

EARTH 2150

COLLECTORS EDITION

SDK MANUAL

LIZENZBESTIMMUNGEN

Durch den Kauf der Software erwerben Sie das nicht ausschließliche, zeitlich nicht begrenzte im Folgenden definierte Benutzungsrecht. Das Benutzungsrecht erlischt bei Verstoß gegen die Benutzungsregeln.

1. Sie dürfen die Software auf Ihren eigenen Computern installieren und benutzen. Eine Sicherheitskopie der Software dürfen Sie nur für eigene Zwecke erstellen. Eine Weitergabe einer Kopie in jeglicher Form, d.h. die Weitergabe auf nicht Originaldatenträgern, Internet oder sonstigen Onlinediensten, ist nicht zulässig.
2. Sie können Ihr Nutzungsrecht an dieser Software an eine andere Person übertragen, indem Sie Ihre erworbene Originalsoftware nichtgewerblich verkaufen oder verschenken. Jedoch verfällt dann Ihr Nutzungsrecht und Sie müssen die Sicherheitskopie und alle Programmdateien der Software auf Ihren Computern löschen. Bei Nichtbefolgung verstößen Sie gegen das Urheberrechtsgesetz.
3. In der Software verwendete Grafiken, Filmsequenzen, Sounds und Musiken unterliegen einem eigenen urheberrechtlichen Schutz, unabhängig von Blackstar Interactive.
4. Jeglicher Eingriff in die Software, der Dekomplizierung, Deassemblierung oder Veränderung dient, ist untersagt. Copyright-Vermerke und Markennamen dieser Software dürfen nicht gelöscht, überschrieben oder auf andere Art verändert werden.
5. Die gewerbliche Benutzung ist nicht gestattet. Das Verleihen, Vermieten, Vorführen oder Verbinden mit anderer Software ist nicht erlaubt.
6. Innerhalb der gesetzlichen Gewährleistungsfrist gewährleistet Blackstar Interactive die Lauffähigkeit der Software. Dazu wurde die Software sorgfältig auf den die Mindestanforderungen erfüllenden verschiedenen üblichen Standard-Konfigurationen getestet. Es gibt jedoch keine Software, die auf allen möglichen Konfigurationen problemlos abläuft. Daher kann Blackstar Interactive für etwaige Störungen, die durch andere Programme, Computerkomponenten oder Bedienungsfehlern verursacht wurden, keine Haftung übernehmen.
7. Falls die Software wider den Benutzerregeln gewerblich genutzt wird, haftet Blackstar Interactive nicht für eventuelle Folgeschäden dieser gewerblichen Nutzung.
8. Blackstar Interactive haftet nicht für die Konsequenzen jeglicher Datenverluste.
9. Die Haftung pro verkaufter Softwareeinheit ist betragsmäßig maximal auf den Kaufpreis der Software beschränkt.
10. Die in der Software aufgeführten geschützten Bezeichnungen unterliegen dem Eigentum ihrer Besitzer. Die Benutzung der Bezeichnungen ist daher mit ihren Eigentümern abzuklären.
11. Sollten eine oder mehrere Klauseln dieser Nutzungsregeln unzulässig sein, oder gegen geltendes Recht verstößen, ist diese so abzuändern, dass sie dennoch dem ursprünglichen Sinne entspricht.
12. Es gelten für die Nutzungsregeln ausschließlich die Rechtsnormen der Bundesrepublik Deutschland.
13. Der Gerichtsstand von Blackstar Interactive befindet sich in Worms am Rhein.

Earth Moon Editor Developer Guide

TABLE OF CONTENTS

1. Earth C Documentation.....	4
EarthC language description.....	5
Objects hierarchy.....	8
Scripts directory structure.....	22
Creating campaign step by step.....	196
Object identifiers.....	198
EarthC tool.....	202
WDCreator tool.....	202
LangC tool.....	203
Tips&Tricks.....	204
2. Moon C Documentation.....	206
MoonC language description.....	207
Objects hierarchy.....	208
Scripts directory structure.....	210
Creating campaign step by step.....	333
Object identifiers.....	334
MoonC tool.....	339
WDCreator tool.....	340
LangC tool.....	340
Tips&Tricks.....	341

Earth 2150

EarthC documentation

Version 1.0 for Earth 2150 version 2.8.7

Copyright © 1999-2002 ZUXXEZ Entertainment AG

This documentation and all tools distributed with it, are used on your own risk and we offer NO support of ANY kind. We do not guarantee that all functions/methods/structures etc. will behave exactly in the way it is described in this documentation. Badly designed custom scripts may cause game crashes or make you unable to play over net.

!! Welcome to the EarthC documentation.!!

EarthC is a developer tool for Earth 2150 which allows you to create your own campaigns, tutorials, gametypes, AI players and unit scripts. This documentation contains sources of all scripts used in Earth 2150. You can use it to learn how EarthC works before creating your own scripts, then you can use it as the base for your scripts. You can also create your own versions of the three standard Earth 2150 campaigns by changing missions, adding new ones or removing boring ones (we hope there aren't any :-)).

EarthC is a programming language so you need basic programming skills to write your own scripts but included sources should help you a lot.

We have included also two other usefull tools: WDCompiler for putting files into a single '*.wd' file and LangC which helps you organize and localize strings in scripts. All tools are located in the Tools subdirectory.

EarthC language description

EarthC is a relatively simply language heavily base on C++. We use it to control game's objects, modify AI behaviors, create missions and many many more....

The basic structure of EarthC class

BaseClassIdentifier "Script name"

```
{  
    consts definitions  
    member definitions  
    functions definitions  
    states definitions  
    events definitions  
    commans definitions  
}
```

"Script name" is a name which this script has in game

Keywords

Control Structures

Local Variable Definitions

Operators

BaseClassIdentifier

Unit scripts:

aircraft
builder
carrier
civil
harvester
repairer
sapper
supplier
tank

Other scripts:

platoon
player
campaign
mission

CONSTS

Consts section has the following structure:

```
const
{
    Name1=234;
    xxxx=123;
}
```

Class Members

possible members types:

int (integer 32-bits signed)
enum (a set of strings)
unit
unitex (usually used in mission class)

punit (used only in platoon)

player

platoon

You can also use other classes but you never get a pointer to the object of this class so they are useless as members

aircraft

builder

carrier

civil

harvester

repairer

sapper

supplier

tank

campaign

mission

Enums declarations:

```
enum MemberIdentifier
{
    "string0",
    "string1",
    multi:
    "stringmulti"
}
```

Variable declared as enum behaves like integer but can additionally be attached to Command (see Commands) and create multistate command button.

examples:

```
int m_nk;
unit m_uUnit;
enum lights
{
    "Lights AUTO",
    "Lights ON",
    "Lights OFF",
    multi:
    "Lights Mode"
}
```

Functions

User defined functions can have any number of parameters of any type but they always return int.

examples:

```
int FindWay(int nX,int nY,int nZ)
{
    int k;
    for(k=0;k<123;k++)
    {
        ....
    }
}
```

States

User can define as many states as he needs. Current state is called periodically every few game ticks. Current state can be changed using instruction:

```
return TheNameOfNewState[,delay]; //inside state
delay is an optional delay (in game ticks) of the next state call (default is 20)
or
state NewStateName; //inside command
typical state declaration:
state StateName
{
}
additionally if you need to reference state X1 in state X2 and state X2 in state X1
you can declare state:
state X1; //here declaration
state X2
{
    return X1;
}
state X1
{
    return X2;
}
```

IMPORTANT: First state declared or defined in file becomes current state until you use any current state changing instruction

Events

Events are predefined callbacks which Earth code calls to inform script about specific problems like a shortage of ammunition or being hit by enemy etc.

event EventName(parameters)

```
{
    [...]
}
```

events usually returns true or false

Commands

Command structure
command CommandName(parameters) [button EnumName | "ButtonName"] [description "DescriptionText"] [priority number] [hotkey] [hidden]
{
 [...]
}

CommandName - see CommandObject

priority - controls order of buttons in game interface (0-255)
hotkey - use hotkey for this button
hidden - don't display button on interface
examples

command Attack(unit uTarget) button "Attack"
{
}
//-----

enum m_eLights
{
 "Lights AUTO",
 "Lights ON",
 "Lights OFF",
 multi:
 "Lights Mode"
}
[...]
//depend on value of m_eLight one of Lights auto(0)/on(1)/off(2) or mode(en-
thing else then 0,1 or 2) will be displayed on game panel.
command SetLights(int nMode) button m_eLights priority 204
{
}

Keywords

if
then
else
while
for
do
int
event
state
command
function
true
false
return
string - not implemented
break
assert - not implemented
verify - not implemented
enum
consts
button
null
multi
description
hotkey
hidden
priority

Control Structures

EarthC accept the following control structures:
block instruction:

```
{  
  instruction;  
  ...  
}  
if (condition)  
  instruction;  
if (condition)  
  instruction;  
else
```

instruction;
while (condition)
 instruction;
do
 instruction;
while (condition);
for(expression;condition;expression)
 instruction;
you can also use simplified forms e.g. for(;;)
break;

Local Variables

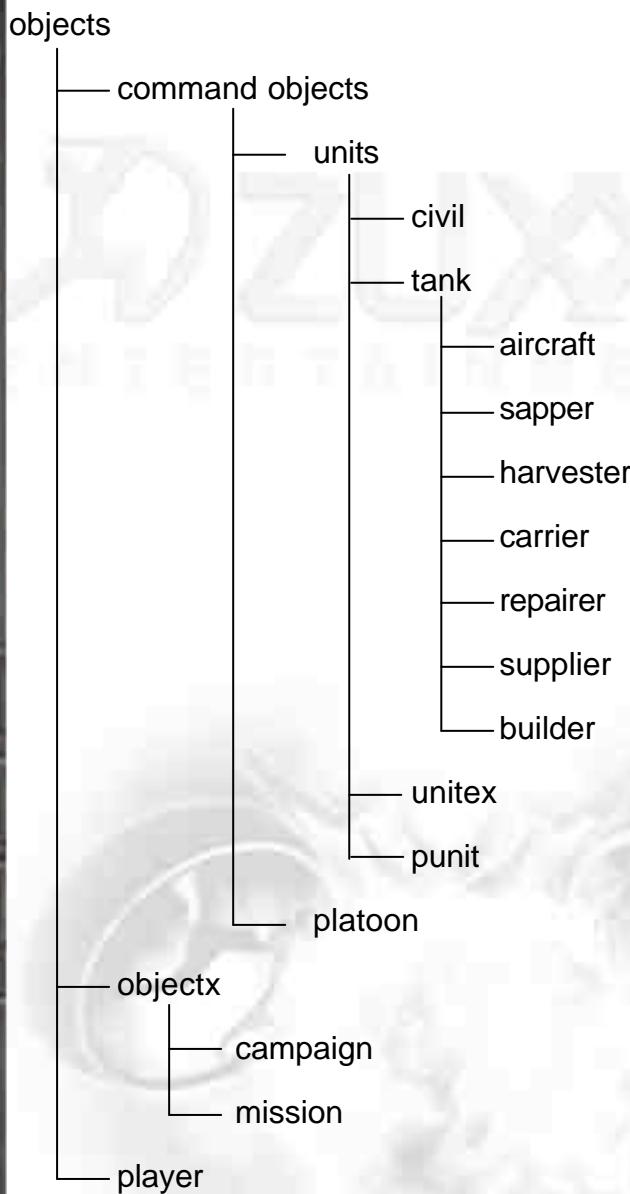
Local variables are defined as in C at the beginning of each function, command, state or event

examples:
state x
{
 int k;
 unit uTarget;
 if (k==9)
 k=3;
 unit uObject; //illegal
 command (int k)
{
 int k;//remember if you assign the same name to local variable and function
 parameter, function parameter won't be accessible

Operators

arithmetic operators:
+ //unary and dual
- //unary and dual
/
%
*
++
--
comparision operators:
<
>
<=
>=
==
!=
logical operators:
!
&&
||
bitwise operators
~
&
|
assign operators
=
member selector
.
parenthesis
(
)
operator precedence
1. () .
2. ! ~ +(unary) - (unary) ++ --
3. * / % &
4. + - |
5. < > <= >= == !=
6. &&
7. ||
8. =

Objects hierarchy



object

Description

Base object for all other objects. Contains couple basic functions.
It is special object and has no name so variable of this type can't be declared in script.

Functions

```
void TraceD(string strText)
void TraceD(int nValue)
int GetEntranceX(unit uTarget)
int GetEntranceY(unit uTarget)
int GetEntranceZ(unit uTarget)
int GetMaxSideColor()
int Distance(int nX1, int nY1, int nX2, int nY2)
```

[For further information take a look at our online support!](#)

commandobject

Description

Object commandobject contains commands common for units and platoon. It is special object and has no name so variable of this type can't be declared in script.

Functions

```
void ChangedCommandValue()
```

Commands

```
int Initialize()
int Uninitialize()
int Stop()
int Land()
int Move(int nX, int nY, int nZ)
int Attack(unit uTarget)
int AttackOnPoint(int nX, int nY, int nZ)
int Escort(unit uTarget)
int HoldPosition()
int Patrol(int nX, int nY, int nZ)
int SetHarvestPoint(int nX, int nY, int nZ)
int Repair(unit uTarget)
int Convert(unit uTarget)
int Repaint(unit uTarget)
int Upgrade(unit uTarget)
int SetUpgradeWeapons(int nMode)
int SetUpgradeChassis(int nMode)
int SetUpgradeShield(int nMode)
int SetRepaintSideColor(int nSideColor)
int SetContainerSource(unit uBuilding)
int GetSingleContainer(unit uContainer)
int SetContainerDestination(unit uBuilding)
int Enter(unit uObject)
int SendSupplyRequest()
int FlyForSupply()
int BuildBuilding(int nX, int nY, int nZ, int nAlpha, int nBuildingID)
int BuildTrench(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int BuildFlatTerrain(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int BuildWall(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int BuildWideBridge(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int BuildNarrowBridge(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int BuildWideTunnel(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int BuildNarrowTunnel(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int MineTerrainClose(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int MineTerrainMedium(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int MineTerrainFar(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int MineTerrainLine(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int SetTransporterSupplyCenter(unit uBuilding)
int MoveToSupplyCenterForLoading()
int KeepFormation()
int SetLights(int nLightsMode)
int SetAttackMode(int nAttackMode)
int SetMovementMode(int nMovementMode)
int DisposePlatoon()
int AddUnitToPlatoon(unit uUnit)
int RemoveUnitFromPlatoon(unit uUnit)
int RotateLeft()
int RotateRight()
int UserNoParam0()
int UserNoParam1()
int UserNoParam2()
int UserNoParam3()
int UserNoParam4()
```

```
int UserNoParam5()
int UserNoParam6()
int UserNoParam7()
int UserNoParam8()
int UserNoParam9()
int UserOneParam0(int nMode)
int UserOneParam1(int nMode)
int UserOneParam2(int nMode)
int UserOneParam3(int nMode)
int UserOneParam4(int nMode)
int UserOneParam5(int nMode)
int UserOneParam6(int nMode)
int UserOneParam7(int nMode)
int UserOneParam8(int nMode)
int UserOneParam9(int nMode)
int UserObject0(unit uObject)
int UserObject1(unit uObject)
int UserObject2(unit uObject)
int UserObject3(unit uObject)
int UserObject4(unit uObject)
int UserObject5(unit uObject)
int UserObject6(unit uObject)
int UserObject7(unit uObject)
int UserObject8(unit uObject)
int UserObject9(unit uObject)
int UserPoint0(int nX, int nY, int nZ)
int UserPoint1(int nX, int nY, int nZ)
int UserPoint2(int nX, int nY, int nZ)
int UserPoint3(int nX, int nY, int nZ)
int UserPoint4(int nX, int nY, int nZ)
int UserPoint5(int nX, int nY, int nZ)
int UserPoint6(int nX, int nY, int nZ)
int UserPoint7(int nX, int nY, int nZ)
int UserPoint8(int nX, int nY, int nZ)
int UserPoint9(int nX, int nY, int nZ)
int UserLine0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserLine9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserDoubleLine9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int UserArea9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
int SpecialNextAngle()
int SpecialSetRepaintSideColorDialog()
int SpecialChangeUnitsScript()
```

Enums

Target types:
findTargetWaterUnit
findTargetFlyingUnit
findTargetNormalUnit
findTargetBuildingUnit
findTargetAnyUnit
findTargetWall
Side selection:
findEnemyUnit
findAlliedUnit
findNeutralUnit
findOurUnit
Kind selection:
findNearestUnit
findWeakestUnit
findStrongestUnit

findDisabledUnit
findSickestUnit
Destination selection:
findDestinationCivilUnit
findDestinationArmedUnit
findDestinationRepairerUnit
findDestinationSupplyUnit
findDestinationAnyUnit
End cannon fire status:
endCannonFireNoTarget
endCannonFireNotInRange
endCannonFireNoAmmo

Cannon types:
cannonTypeLaser
cannonTypeLaserSDI
cannonTypeElectric
cannonTypeMachine
cannonTypeCannon
cannonTypeRocket
cannonTypeAutoRocket
cannonTypeBallisticRocket
cannonTypeSonic
cannonTypeAntiRocket
cannonTypeDropBomb
cannonTypeBomb
cannonTypePelon
cannonTypePlasma
cannonTypeSuperPlasma

Target location:
notInRange
inRangeBadAngleAlpha
inRangeBadAngleBeta
inRangeBadHit
inRangeGoodHit

For further information take a look at our online support!

unitobject

Description:

Object unit is base object for units of different types (like tanks, builders, harvers etc.). This object implements functions common for all types of units. Usually functions which are returning pointer to objects are returning unit type (at least in objects derived from unit). Unit is not base type for user unit scripts. You should use particular objects like civil, tank, repainer etc.

Functions

int GetIFF()
int GetIFFNumber()
int IsBuilding()
int IsLandTarget()
int IsLandUnit()
int IsHelicopter()
int IsBuilder()
int IsCarrier()
int IsHarvester()
int IsRepairer()
int IsSapper()
int IsSupplier()
int IsPassive()
int GetBuildingType()
void CallFreeze(int nFreezeTicks)
int IsFrozen()
void NextCommand(int nPrevCommandStatus)
int HaveEmptyCall()
int GetSideColor()
int GetPlayerSideColor()
int IsBlackFog(int nX, int nY, int nZ)
int IsResourceInPoint(int nX, int nY, int nZ)
int GetResourcesCount(int nX, int nY, int nZ)
int IsContainerInPoint(int nX, int nY, int nZ)
int IsEnemy(unit uTarget)
int IsNeutral(unit uTarget)
int IsAlliance(unit uTarget)
int IsVisible(unit uTarget)
int GetWorldLeft()
int GetWorldTop()
int GetWorldRight()
int GetWorldBottom()
int DistanceTo(int nX, int nY)
void CallMoveToPoint(int nX, int nY, int nZ)
void CallMoveToPointForce(int nX, int nY, int nZ)

void CallMoveAndLandToPointForce(int nX, int nY, int nZ)
void CallMoveLowToPoint(int nX, int nY, int nZ)
void CallMoveLowToPointForce(int nX, int nY, int nZ)
void CallMoveOneField(int nX, int nY, int nZ)
void CallMoveCountFields(int nX, int nY, int nZ, int nCount)
void CallMoveInsideObject(unit uTarget)
void CallStopMoving()
void CallLand()
void CallTurnToAngle(int nAlphaAngle)
void CallGetFlyingSupply()
int IsMoving()
int IsGettingSupply()
int IsLive()
int IsDisabled()
int GetHP()
int GetMaxHP()
int GetLocationX()
int GetLocationY()
int GetLocationZ()
int GetAlphaAngle()
int GetMoveLocationX()
int GetMoveLocationY()
int GetMoveLocationZ()
int GetMoveLocationAngle()
void SetLightsMode(int nMode)
int IsFreePoint(int nX, int nY, int nZ)
void SendSupplyRequest()
unit GetAttacker()
void ClearAttacker()
void SetTargetToObject(unit uTarget)
int GetAngleToTarget(unit uTarget)
int GetAngleToPoint(int nX, int nY, int nZ)
unit FindTarget(int nTargetType, int nSideSelection, int nKindSelection, int nDestinationSelection)
void BuildTargetsArray(int nTargetType, int nSideSelection, int nDestinationSelection)
void SortFoundTargetsArray()
int GetTargetsCount()
int StartEnumTargetsArray()
unit GetNextTarget()
void EndEnumTargetsArray()
unit FindClosestEnemy()
unit FindClosestEnemyEx(int nTargetType)
unit FindClosestEnemyUntoBuilding(int nTargetType)
void AllowScriptWithdraw(int nAllow)
int IsAllowingWithdraw()
int FindSupplyCenterPlace()
int GetFoundSupplyCenterPlaceX()
int GetFoundSupplyCenterPlaceY()
int GetFoundSupplyCenterPlaceZ()
int GetCurrSupplyCenterAssemblyPositionX()
int GetCurrSupplyCenterAssemblyPositionY()
int GetCurrSupplyCenterAssemblyPositionZ()
int IsOnWorkingSupplyCenter()
int IsWorkingSupplyCenterInPoint(int nX, int nY, int nZ)
int HaveBanner()
int HaveRadar()
int HaveScreamer()
int HaveShadow()
void MessageTalk(int nMessageNum)
int InPlatoon()
int IsPlayer()

Events

int OnHit()
int OnCannonLowAmmo(int nCannonNum)
int OnCannonNoAmmo(int nCannonNum)
int OnCannonFoundTarget(int nCannonNum, unit uTarget)
int OnCannonEndFire(int nCannonNum, int nEndStatus)
int OnFreezeForSupplyOrRepair(int nFreezeTicks)
int OnTransportedToWorld()
int OnConvertedToWorldPlayer()
int OnKilledSupplyCenterBuilding()

Enums

Building types:
buildingNormal
buildingPowerPlant
buildingSolarPower
buildingEnergyTransmitter
buildingEnergyBattery
buildingBase
buildingFactory
buildingWaterBase

```

buildingSupplyCenter
buildingMine
buildingRefinery
buildingResourceTransportBase
buildingResearchCenter
buildingHeadquarter
buildingBBC
buildingPlasmaControl
buildingTeleport
buildingSDDefence
buildingShadowTower
buildingRadar
buildingSilos
buildingPlasmaCannon
buildingGenerator
buildingSolarBattery
buildingWeatherControl
buildingTunnelEntrance
buildingCopula
buildingLaserWall
buildingSpaceStation
buildingTransportCenter
buildingMiningRefinery
buildingMiningResourceTransportBase
buildingBaseFactory (LC)
buildingBadRaceBuilding
buildingWaitingForCopulaOpen
buildingTransportedDown

```

[For further information take a look at our online support!](#)

civil object

Description

Scripts of type civil are attached to units which are not armored and has no special possibilities. It can be fake, chasis with radar or screamer etc.

tank object

Description

Scripts of type tank are attached to units with some cannons. If chasis have attached cannon and radar it have tank type. If chasis have attached cannon and special equipment (like carrier or repainer) it have type depend on special equipment type (carrier or repainer). If special chassis (like sapper chassis) have attached cannon it still have special chassis type (sapper).

Functions

```

int GetCannonsCount()
int GetCannonType(int nCannonNum)
int GetCannonShootRange(int nCannonNum)
int GetCannonSightRange(int nCannonNum)
void CannonFireToTarget(int nCannonNum, unit uTarget, int nShootCount)
void CannonFireGround(int nCannonNum, int nx, int ny, int nz, int nShootCount)
int IsTargetInCannonRange(int nCannonNum, unit uTarget)
int IsPointInCannonRange(int nCannonNum, int nx, int ny, int nz)
void SetCannonFireMode(int nCannonNum, int nFireMode)
void StopCannonFire(int nCannonNum)
int CannonRequiresSupply(int nCannonNum)
int CanCannonFireToAircraft(int nCannonNum)
int GetCannonAngleToTarget(int nCannonNum, unit uTarget)
int GetCannonAngleToPoint(int nCannonNum, int nx, int ny, int nz)
int CanCannonTurnFullAngle(int nCannonNum)
int HaveCannonsMissingAmmo()
int GetAmmoCount()

```

[For further information take a look at our online support!](#)

aircraft object

Description

Object aircraft is a special version of tank object for those units which are flyable.

sapper object

Description

Scripts of type sapper are attached to Sapper units.

Functions

```

void CallPutMine()
int IsPuttingMine()
int HaveMines()
int GetMinesCount()
int IsGoodPointForMine(int nx, int ny, int nz)
int AddMineAreaClose(int nx1, int ny1, int nz1, int nx2, int ny2, int nz2)
int AddMineAreaFar(int nx1, int ny1, int nz1, int nx2, int ny2, int nz2)
int AddMineLine(int nx1, int ny1, int nz1, int nx2, int ny2, int nz2)
int AddMinePoint(int nx, int ny, int nz)
void ResetMinePoints()
int HaveMinePoints()
int GetCurrMinePointX()
int GetCurrMinePointY()
int GetCurrMinePointZ()
int NextMinePoint()
int IsCurrentMineArea()
int IsCurrentMineAreaClose()
int IsCurrentMineAreaMedium()
int IsCurrentMineAreaFar()
int IsCurrentMineLineAndPoints()

```

[For further information take a look at our online support!](#)

harvester object

Description

Scripts of type harvester are attached to Harvester units.

Functions

```

void CallHarvest()
void CallPutResource()
int IsHarvesting()
int IsPuttingResource()
int SetHarvesterBuilding(unit uBuilding)
unit GetHarvesterBuilding()
void SetCurrentHarvestPoint(int nx, int ny, int nz)
void InvalidateCurrentHarvestPoint()
int GetHarvesterBuildingPutLocationX()
int GetHarvesterBuildingPutLocationY()
int GetHarvesterBuildingPutLocationZ()
int HaveFullResources()
int HaveSomeResources()
int FindResource()
int GetFoundResourceX()
int GetFoundResourceY()
int GetFoundResourceZ()
unit FindHarvesterRefineryBuilding()
unit FindHarvesterTransportBaseBuilding()

```

[For further information take a look at our online support!](#)

carrier object

Description

Scripts of type carrier are attached to Carrier units (chassis with carrier equipment).

Functions

```

void CallGetContainer()
void CallPutContainer()
void CallGetSingleContainer(int nx, int ny, int nz)
int IsGettingContainer()
int IsPuttingContainer()
int HaveContainer()
int SetSourceBuilding(unit uBuilding)
unit GetSourceBuilding()
int GetSourceBuildingTakeLocationX()
int GetSourceBuildingTakeLocationY()
int GetSourceBuildingTakeLocationZ()
int SetDestinationBuilding(unit uBuilding)
unit GetDestinationBuilding()
int GetDestinationBuildingPutLocationX()
int GetDestinationBuildingPutLocationY()
int GetDestinationBuildingPutLocationZ()
int GetContainerTakeLocationX(int nx, int ny, int nz)
int GetContainerTakeLocationY(int nx, int ny, int nz)
int GetContainerTakeLocationZ(int nx, int ny, int nz)
int FindContainerMineBuilding()
unit FindContainerRefineryBuilding()
unit FindContainerTransportBaseBuilding()
unit FindSingleContainer()

```

[For further information take a look at our online support!](#)

repairer object

Description

Scripts of type repairer are attached to Repairer units (chassis with repairer equipment).

Functions

```
void CallRepair(unit uTarget)
void CallConvert(unit uTarget)
void CallRepaint(unit uTarget)
void CallUpgrade(unit uTarget)
int IsRepairing()
int IsConverting()
int IsRepainting()
int IsUpgrading()
int CanRepairTanks()
int CanRepairBuildings()
int CanConvertTanks()
int CanConvertBuildings()
int CanRepaint()
int CanUpgrade()
int CanBeRepaired(unit uTarget)
int CanBeConverted(unit uTarget)
int CanBeRepainted(unit uTarget)
int CanBeUpgraded(unit uTarget)
int IsInGoodPointForOperateOnTarget(unit uTarget)
int IsGoodPointForOperateOnTarget(unit uTarget, int nX, int nY, int nZ)
int GetOperateOnTargetLocationX(unit uTarget)
int GetOperateOnTargetLocationY(unit uTarget)
int GetOperateOnTargetLocationZ(unit uTarget)
int SetRepaintSideColor(int nSideColor)
void SetUpgradeWeapons(int nNotUpgrade)
void SetUpgradeChassis(int nNotUpgrade)
void SetUpgradeShield(int nNotUpgrade)
```

[For further information take a look at our online support!](#)

supplier object

Description

Scripts of type supplier are attached to Supplier units.

Functions

```
void CallLoadAmmo()
void CallPutAmmo()
int IsLoadingAmmo()
int IsPuttingAmmo()
int HaveObjectsForSupply()
int SetSupplyCenterBuilding(unit uBuilding)
unit GetSupplyCenterBuilding()
int GetSupplyCenterLoadPositionX()
int GetSupplyCenterLoadPositionY()
int GetSupplyCenterLoadPositionZ()
int GetSupplyCenterAssemblyPositionX()
int GetSupplyCenterAssemblyPositionY()
int GetSupplyCenterAssemblyPositionZ()
unit GetCurrentObjectForSupply()
int CanCurrentObjectBeSupplied()
int WasCurrentObjectSupplied()
int NextObjectForSupply()
int GetCurrentPutSupplyPositionX()
int GetCurrentPutSupplyPositionY()
int GetCurrentPutSupplyPositionZ()
int IsInGoodCurrentPutSupplyLocation()
void CheckCurrentPutSupplyLocation()
unit FindSupplyCenter()
```

[For further information take a look at our online support!](#)

builder object

Description

Scripts of type builder are attached to Builder units.

Functions

```
void CallBuildBuilding()  
void CallBuildCurrentElement()  
int IsBuildWorking()  
void SetBuildTypeBuildBuilding(int nX, int nY, int nZ, int nAlphaAngle, int nBuildingID)  
void SetBuildTypeBuildElements(int nBuildType)  
int GetCurrentBuildType()  
int IsGoodPointForCurrentBuild()  
int IsGoodPointForCurrentBuild(int nX, int nY, int nZ)  
int GetCurrentBuildLocationX()  
int GetCurrentBuildLocationY()  
int GetCurrentBuildLocationZ()  
int AddElementsLine(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
int AddElementPoint(int nX, int nY, int nZ)  
int IsGoodElementPointForBuild(int nX, int nY, int nZ)  
void RemoveAllElements()  
int GetCurrentElementX()  
int GetCurrentElementY()  
int GetCurrentElementZ()  
int HaveElementsForBuild()  
int NextElementPoint()
```

Enums

Build type enums:
buildNone
buildBuilding
buildWall
buildTrench
buildFlatTerrain
buildWideBridge
buildNarrowBridge
buildWideTunnel
buildNarrowTunnel

[For further information take a look at our online support!](#)

unitex object

Description

Object unitex is a special object used in mission and player scripts. Using this object you can give orders to units and f.e. restore it's HP. All functions in this object are public.

Functions

```
int GetHP()  
int GetMaxHP()  
void RegenerateHP()  
int GetAmmo()  
int GetMaxAmmo()  
void RegenerateAmmo()  
int GetShield()  
int GetMaxShield()  
void RegenerateShield()  
int GetElectronics()  
int GetMaxElectronics()  
void RegenerateElectronics()  
int GetTemperature()  
int GetMaxTemperature()  
void RegenerateTemperature()  
int IsnWorld(int nWorldNum)  
void ChangePlayer(player pNewPlayer)  
void Repaint(player pColorPlayer)  
void CommandStop()  
void CommandLand()  
void CommandMove(int nX, int nY, int nZ)  
void CommandAttack(unit uTarget)  
void CommandAttackOnPoint(int nX, int nY, int nZ)  
void CommandEscort(unit uEscortTarget)  
void CommandHoldPosition()  
void CommandPatrol(int nX, int nY, int nZ)  
void CommandSetHarvestPoint(int nX, int nY, int nZ)  
void CommandRepair(unit uTarget)  
void CommandConvert(unit uTarget)  
void CommandRepaint(unit uTarget)  
void CommandUpgrade(unit uTarget)  
void CommandSetContainerSource(unit uBuildingMine)  
void CommandGetSingleContainer(unit uContainer)  
void CommandSetContainerDestination(unit uRefineryOrTransportBase)
```

```
void CommandEnter(unit uBuildingOrUnitsTransporter)  
void CommandSendSupplyRequest()  
void CommandFlyForSupply()  
void CommandSetTransporterSupplyCenter(unit uSupplyCenter)  
void CommandMoveToSupplyCenterForLoading()  
void CommandUserNoParam()  
void CommandUserNoParam1()  
void CommandUserNoParam2()  
void CommandUserNoParam3()  
void CommandUserNoParam4()  
void CommandUserNoParam5()  
void CommandUserNoParam6()  
void CommandUserNoParam7()  
void CommandUserNoParam8()  
void CommandUserNoParam9()  
void CommandUserObject0(unit uTarget)  
void CommandUserObject1(unit uTarget)  
void CommandUserObject2(unit uTarget)  
void CommandUserObject3(unit uTarget)  
void CommandUserObject4(unit uTarget)  
void CommandUserObject5(unit uTarget)  
void CommandUserObject6(unit uTarget)  
void CommandUserObject7(unit uTarget)  
void CommandUserObject8(unit uTarget)  
void CommandUserObject9(unit uTarget)  
void CommandUserPoint0(int nX, int nY, int nZ)  
void CommandUserPoint1(int nX, int nY, int nZ)  
void CommandUserPoint2(int nX, int nY, int nZ)  
void CommandUserPoint3(int nX, int nY, int nZ)  
void CommandUserPoint4(int nX, int nY, int nZ)  
void CommandUserPoint5(int nX, int nY, int nZ)  
void CommandUserPoint6(int nX, int nY, int nZ)  
void CommandUserPoint7(int nX, int nY, int nZ)  
void CommandUserPoint8(int nX, int nY, int nZ)  
void CommandUserPoint9(int nX, int nY, int nZ)  
void CommandUserLine0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserLine9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserDoubleLine9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)  
void CommandUserArea9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
```

[For further information take a look at our online support!](#)

punit object

Description

Object punit is a special object used in platoon scripts. Using functions from this object you can give commands to units from platoon. All functions in this object are public.

Functions

```
void CommandStop()  
void CommandLand()  
void CommandMove(int nX, int nY, int nZ)  
void CommandAttack(unit uTarget)  
void CommandAttackOnPoint(int nX, int nY, int nZ)  
void CommandEscort(unit uEscortTarget)  
void CommandHoldPosition()  
void CommandPatrol(int nX, int nY, int nZ)  
void CommandSetHarvestPoint(int nX, int nY, int nZ)
```



```

void CommandRepair(unit uTarget)
void CommandConvert(unit uTarget)
void CommandRepaint(unit uTarget)
void CommandUpgrade(unit uTarget)
void CommandSetContainerSource(unit uBuildingMine)
void CommandGetSingleContainer(unit uContainer)
void CommandSetContainerDestination(unit uRefineryOrTransportBase)
void CommandEnter(unit uBuildingOrUnitsTransporter)
void CommandSendSupplyRequest()
void CommandFlyForSupply()
void CommandSetTransporterSupplyCenter(unit uSupplyCenter)
void CommandMoveToSupplyCenterForLoading()
void CommandUserNoParam0()
void CommandUserNoParam1()
void CommandUserNoParam2()
void CommandUserNoParam3()
void CommandUserNoParam4()
void CommandUserNoParam5()
void CommandUserNoParam6()
void CommandUserNoParam7()
void CommandUserNoParam8()
void CommandUserNoParam9()
void CommandUserObject0(unit uTarget)
void CommandUserObject1(unit uTarget)
void CommandUserObject2(unit uTarget)
void CommandUserObject3(unit uTarget)
void CommandUserObject4(unit uTarget)
void CommandUserObject5(unit uTarget)
void CommandUserObject6(unit uTarget)
void CommandUserObject7(unit uTarget)
void CommandUserObject8(unit uTarget)
void CommandUserObject9(unit uTarget)
void CommandUserPoint0(int nX, int nY, int nZ)
void CommandUserPoint1(int nX, int nY, int nZ)
void CommandUserPoint2(int nX, int nY, int nZ)
void CommandUserPoint3(int nX, int nY, int nZ)
void CommandUserPoint4(int nX, int nY, int nZ)
void CommandUserPoint5(int nX, int nY, int nZ)
void CommandUserPoint6(int nX, int nY, int nZ)
void CommandUserPoint7(int nX, int nY, int nZ)
void CommandUserPoint8(int nX, int nY, int nZ)
void CommandUserPoint9(int nX, int nY, int nZ)
void CommandUserLine0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserLine9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserDoubleLine9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea0(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea1(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea2(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea3(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea4(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea5(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea6(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea7(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea8(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
void CommandUserArea9(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)

int IsAlliance(unit uTarget)
int IsVisible(unit uTarget)
int GetWorldLeft()
int GetWorldTop()
int GetWorldRight()
int GetWorldBottom()
int DistanceTo(int nX, int nY)
int HaveBanner()
int HaveRadar()
int HaveScreamer()
int HaveShadow()
unit FindTarget(int nUnit, int nTargetType, int nSideSelection, int nKindSelection,
int nDestinationSelection)
void BuildTargetsArray(int nUnit, int nTargetType, int nSideSelection, int
nDestinationSelection)
void SortFoundTargetsArray(int nUnit)
int GetTargetsCount(int nUnit)
int StartEnumTargetsArray(int nUnit)
unit GetNextTarget(int nUnit)
void EndEnumTargetsArray(int nUnit)
unit FindClosestEnemy(int nUnit)
unit FindClosestEnemyEx(int nUnit, int nTargetType)
void KeepFormation(int nKeep)
void Dispose()
int GetCannonsCount(int nUnit)
int GetCannonType(int nUnit, int nCannonNum)
int GetCannonShootRange(int nUnit, int nCannonNum)
int GetCannonSightRange(int nUnit, int nCannonNum)
void CannonFireToTarget(int nUnit, int nCannonNum, unit uTarget, int
nShootCount)
void CannonFireGround(int nUnit, int nCannonNum, int nX, int nY, int nZ, int
nShootCount)
int IsTargetInCannonRange(int nUnit, int nCannonNum, unit uTarget)
int IsPointInCannonRange(int nUnit, int nCannonNum, int nX, int nY, int nZ)
void SetCannonFireMode(int nUnit, int nCannonNum, int nMode)
void StopCannonFire(int nUnit, int nCannonNum)
void SendSupplyRequest(int nUnit)
void SetLightsMode(int nUnit, int nLightsMode)
void AddUnitToPlatoon(unit uUnitToAdd)
void RemoveUnitFromPlatoon(unit uUnitToRemove)
void CallMoveToPoint(int nX, int nY, int nZ)
int IsMoving()
void CallStopMoving()
void CallTurnToAngle(int nUnit, int nAngle)
int GetLocationX()
int GetLocationY()
int GetLocationZ()
int IsFrozen()
unit GetAttacker(int nUnit)
void ClearAttacker(int nUnit)
int GetCannonAngleToTarget(int nUnit, int nCannonNum, unit uTarget)
int GetCannonAngleToPoint(int nUnit, int nCannonNum, int nX, int nY, int nZ)
int GetUnitsCount()
unit GetUnitPosition()
void SetUnitPosition(int nUnit, int nDx, int nDy)
int GetPositionX(int nUnit)
int GetPositionY(int nUnit)
int GetPositionZ(int nUnit)
void SetLeader(int nUnit)
int InPlatoon(unit uUnit)
void EnableFeatures(int nFlags, int nEnable)
void Turn(int nAngle)
void SwapUnits(int nOne, int nTwo)
int GetType()

Enums
Platoon features:
platoonHQDefense
platoonKeepFormation
platoonFreeUnits

Platoon types:
typeHelicopters
typeTanks
typeShips


```

[For further information take a look at our online support!](#)

platoon object

Functions

```

void SetAllCannons(unit uTarget)
void SetCannon(int nUnit, int nCannonNum, unit uTarget)
void NextCommand(int nPrevCommandStatus)
int GetSideColor()
int GetPlayerSideColor()
int IsEnemy(unit uTarget)
int IsNeutral(unit uTarget)

```

[For further information take a look at our online support!](#)

objectX object

Description

Special unnamed object. It contains functions which are common for campaign and mission object. There are no scripts of this type.

Functions

(All functions are static)



```

void SaveCameraPosition()
void RestoreCameraPosition()
void SaveCameraPosition2(int nCameraNum)
void RestoreCameraPosition2(int nCameraNum)
void LookAt(int nX, int nY, int nZ, int nAlpha, int nViewNum, int nTunnel, int nWorldNum)
void LookAt(int nX, int nY, int nZ, int nAlpha, int nViewNum, int nTunnel, int nWorldNum, int nCameraNum)
void LookAt(int nX, int nY, int nZ, int nAlpha, int nViewNum, int nTunnel, int nWorldNum, int nDelay)
void LookAt(int nX, int nY, int nZ, int nAlpha, int nViewNum, int nTunnel, int nWorldNum, int nDelay)
void EnableCameraMovement(int nEnable) - not implemented
void EnableInterface(int nEnable) - not implemented
void ShowVideo(string strVideoName)
void MessageBox(string strMessage)
void SetAvailableWorlds(int nWorldsMask)
int GetAvailableWorlds()
void EndGame(string strOutro)
void SetTime(int nDayTick)
int GetDifficultyLevel()
void SendCustomEvent(int nParam1, int nParam2, int nParam3, int nParam4, int nParam5)

```

Enums

Player races:

raceUCS
raceED
raceLC

[For further information take a look at our online support!](#)

campaign object

Functions

(All functions are static)

```

int GetCampaignTime()
void RegisterMission(int nMissinNum, string strMissionLevel, string
strMissionScript, string strShortBriefing, int nFlags, int nLongitude, int nLatitude,
int nDistanceToBase1, int nDistanceToBase2, int nDistanceToBase3, int
nNextMission0, int nNextMission1, int nNextMission2, int nNextMission3)
void RegisterMission(int nMissinNum, string strMissionLevel, string
strMissionScript, string strShortBriefing, int nFlags, int nLongitude, int nLatitude,
int nDistanceToBase1, int nDistanceToBase2, int nDistanceToBase3, int
nNextMission0, int nNextMission1, int nNextMission2)
void RegisterMission(int nMissinNum, string strMissionLevel, string
strMissionScript, string strShortBriefing, int nFlags, int nLongitude, int nLatitude,
int nDistanceToBase1, int nDistanceToBase2, int nDistanceToBase3, int
nNextMission0, int nNextMission1)
void RegisterMission(int nMissinNum, string strMissionLevel, string
strMissionScript, string strShortBriefing, int nFlags, int nLongitude, int nLatitude,
int nDistanceToBase1, int nDistanceToBase2, int nDistanceToBase3, int
nNextMission0)
void RegisterMission(int nMissinNum, string strMissionLevel, string
strMissionScript, string strShortBriefing, int nFlags, int nLongitude, int nLatitude,
int nDistanceToBase1, int nDistanceToBase2, int nDistanceToBase3)
void EnableMission(int nMissionNum, int nEnable)
void ActivateMission(int nMissionNum, int nActivate)
void EnableMissionsEq(int nFlagsAndValue, int nFlagsEqValue, int nEnable)
void EnableMissions(int nFlagsAndValue, int nEnable)
void ActivateMissionsEq(int nFlagsAndValue, int nFlagsEqValue, int nActivate)
void ActivateMissions(int nFlagsAndValue, int nActivate)
int IsMissionEnabled(int nMissionNum)
int IsMissionActive(int nMissionNum)
int AreMissionsEnabled(int nFlagsAndValue, int nFlagsEqValue)
int AreMissionsActive(int nFlagsAndValue)
int AreMissionsEnabledEq(int nFlagsAndValue, int nFlagsEqValue)
int AreMissionsActiveEq(int nFlagsAndValue, int nFlagsEqValue)
int GetNextMission(int nMissionNum, int nNextPosition)
int GetCurrentMission()
void LoadMission(int nWorldNum, int nMissionNum)
void LoadBase(int nWorldNum, int nMissionNum, int nOwner)
void SetActivePlayerAndWorld(int nPlayerNum, int nWorldNum)
void SetTimer(int nTimerNum, int nInterval)
void EnableChooseMissionButton(int nEnable)
void CreateGamePlayer(int nPlayerNum, int nRace, int nPlayerType, string
strScript)
int GetMoneySentToOrbit()
int GetSeason()
void SetSeason(int nSeason)
void SetMissionState(int nMissionNum, int nState)
int GetMissionState(int nMissionNum)

```

Events

```

void StartMission(int nMissionNum)
void EndMission(int nMissionNum, int nResult)
void EnableNextMission(int nMissionNum, int nNextMissionNum, int nEnable)
void Timer0()
void Timer1()
void Timer2()
void Timer3()
void Timer4()
void Timer5()
void Timer6()
void Timer7()
int CustomEvent0(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent1(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent2(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent3(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent4(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent5(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent6(int nParam1, int nParam2, int nParam3, int nParam4)
int CustomEvent7(int nParam1, int nParam2, int nParam3, int nParam4)

```

Enums

Player types:
playerAI
playerLocal
Mission states:
stateAvailable
stateFailed
stateSkipped
stateAccomplished

[For further information take a look at our online support!](#)

mission object

Functions

```

int GetWorldNum()
void EnableNextMission(int nMissionNum, int nEnable)
void SetTimer(int nTimerNum, int nInterval)
void EndMission(int nResult)
int GetMissionTime()
player GetPlayer(int nPlayerNum)
void EnableGoal(int nGoalNum, int nEnable)
void SetGoalState(int nGoalNum, int nState)
int GetGoalState(int nGoalNum)
void Rain(int nX, int nY, int nRange, int nUpTime, int nConstTime, int nDownTime,
int nIntensity)
void Snow(int nX, int nY, int nRange, int nUpTime, int nConstTime, int nDownTime,
int nIntensity)
void Storm(int nX, int nY, int nRange, int nUpTime, int nConstTime, int nDownTime,
int nRainIntensity, int nStormIntensity, nStormPower)
void MeteorRain(int nX, int nY, int nRange, int nUpTime, int nConstTime, int
nDownTime, int nIntensity, int nPower)
void Wind(int nUpTime, int nConstTime, int nDownTime, int nIntensity, int
nAlphaAngle)
void Lighting(int nX, int nY, int nPower)
void Meteor(int nX, int nY, int nPower)
void KillArea(int nIFFMask, int nX, int nY, int nZ, int nRange)
void DamageArea(int nIFFMask, int nX, int nY, int nZ, int nRange)
void ShowArea(int nIFFMask, int nX, int nY, int nZ, int nRange)
void HideArea(int nIFFMask, int nX, int nY, int nZ, int nRange)
unitex GetUnit(int nX, int nY)
unitex GetUnit(int nX, int nY, int nZ)
int GetLeft()
int GetRight()
int GetTop()
int GetBottom()
void EnableEndMissionButton(int nEnable)
void EnableEndMissionButton(int nEnable, int nResult)
void EnableMissionGoalsButton(int nEnable)
int IsGoalEnabled(int nGoalNum)
void CallCamera()
void ResourcesPerContainer(int nResourcesPerContainer)
int PointExist(int nMarkNum)
int GetPointX(int nMarkNum)
int GetPointY(int nMarkNum)
int GetPointZ(int nMarkNum)
int ResourcesLeftInMoney()
void CreateArtifact(string strArtifactID, int nX, int nY, int nZ, int nArtifactNumber,
int nArtifactSpecialType)
void CreateArtifact(string strArtifactID, int nX, int nY, int nZ, int nArtifactNumber,
int nArtifactSpecialType, int nPlayerNum)
void RegisterGoal(int nGoalNum, string strGoalText)

```



Commands

```
int Initialize()
int Uninitialize()
int Combo1(int nSelItem)
int Combo2(int nSelItem)
int Combo3(int nSelItem)
int Combo4(int nSelItem)
int Combo5(int nSelItem)
int Combo6(int nSelItem)
```

Events

Enums

Goal states:

goalAchieved
goalNotAchieved
goalFailed

Artefact types:

artefactSpecialAIOther
artefactSpecialAINewAreaLocation
artefactSpecialAIEnemyMainBaseLocation

Building types:

buildingNormal
buildingPowerPlant
buildingSolarPower
buildingEnergyTransmitter
buildingEnergyBattery
buildingBase
buildingFactory
buildingWaterBase
buildingSupplyCenter
buildingMine
buildingRefinery
buildingResourceTransportBase
buildingResearchCenter
buildingHeadquarter
buildingHQ
buildingPlasmaControl
buildingTeleport
buildingSDIDefense
buildingShadowTower
buildingRadar
buildingSilos
buildingPlasmaCannon
buildingGenerator
buildingSolarBattery
buildingWeatherControl
buildingTunnelEntrance
buildingCopula
buildingLaserWall
buildingSpaceStation
buildingTransportCenter
buildingMiningRefinery
buildingMiningResourceTransportBase
buildingBaseFactory (LC)
buildingBadRaceBuilding
buildingTransportForCoOpOpen
buildingTransportedDown

Chasis flags:

chassisTank
chassisAmphibianTank
chassisShip
chassisHelicopter
chassisAny

Unit flags:

unitMiner
unitBuilder
unitHarvester
unitSupplier
unitFake
unitArmed
unitShadow
unitScreamer
unitObserver
unitCarrier
unitRepairer
unitAny

Player AI features:

aiBuildMiningBuildings
aiBuildMiningUnits
aiControlMiningUnits
aiBuildSupplyBuildings
aiBuildSupplyUnits
aiControlSupplyUnits
aiBuildPowerBuildings
aiBuildResearchBuildings



```
aControlResearches  
aControlDefense  
aControlOffense  
aiBuildBuilders  
aiBuildRepairers  
aiBuildShadows  
aiBuildScreamers  
aiBuildSuperAttack  
aiEnabled  
aiBuildTanks  
aiBuildShips  
aiBuildHelicopters  
aiBuildCivilFactory  
aiBuildMilitaryFactory  
aiBuildShipyard  
aiBuildBuildings  
aiBuildSpecialUnits  
aiRush  
aiUpgradeCannons  
aiEverything  
aiDefenseTowers  
aiRejectAlliance
```

Player AI features 2 (same as AI features plus):

```
ai2BuildingTowers  
ai2BuildFromStartingPoint  
ai2BuildSappers  
ai2ChooseEnemy  
ai2NeutralAI  
ai2BNSSendResult
```

Player races:

```
raceUCS  
raceED  
raceLC
```

[For further information take a look at our online support!](#)

player object

Functions

(All functions are public)

```
void AddMoney(int nMoney)  
void SetMoney(int nMoney)  
int GetMoney()  
void SetEnemy(player pNewEnemyPlayer)  
void SetNeutral(player pNewNeutralPlayer)  
void SetAlly(player pNewAllyPlayer)  
void SetScriptData(int nDataIndex, int nDataValue)  
int GetScriptData(int nDataIndex)  
void SetScriptUnit(int nUnitIndex, unitex uUnit)  
unitex GetScriptUnit(int nUnitIndex)  
void GiveAllUnitsTo(player pPlayer)  
void GiveAllUnitsTo(player pPlayer, int nX, int nY, int nRange) - not implemented  
void GiveAllBuildingsTo(player pPlayer)  
void GiveAllBuildingsTo(player pPlayer, int nX, int nY, int nRange) - not implemented  
void GiveTo(player pPlayer, unit uUnit)  
int GetNumberOfUnits()  
int GetNumberOfUnits(int nX, int nY, int nRange) - not implemented  
int GetNumberOfBuildings()  
int GetNumberOfBuildings(int nBuildingType)  
int GetNumberOfBuildings(int nX, int nY, int nRange) - not implemented  
void EnableAI(int nEnable)  
void EnableAIFeatures(int nFlags, int nEnable)  
void EnableAIFeatures2(int nFlags, int nEnable)  
int GetStartingPointX()  
int GetStartingPointY()  
int IsPointLocated(int nX, int nY, int nZ)  
int IsPointLocated(int nX, int nY)  
void Defeat()  
void Victory()  
void LookAt(int nX, int nY, int nZ, int nAlpha, int nView, int nTunnel)  
void DelayedLookAt(int nX, int nY, int nZ, int nAlpha, int nView, int nTunnel, int nDelay, int nDirection)  
void SetName(string strPlayerName)  
string GetName()  
int GetMoneyCollectedByNow()  
int GetMoneySentToBase()  
int GetMoneySentToOrbit()  
void EnableSendingToBase(int nEnable)  
void EnableMilitaryUnitsLimit(int nEnable)  
void SetMilitaryUnitsLimit(int nUnitsLimit)  
void GetMilitaryUnitsLimit()  
void SetPointToAssemble(int nPointNum, int nX, int nY, int nZ)  
void SetPointToGuard(int nPointNum, int nX, int nY, int nZ)  
void SetPointToAvoid(int nPointNum, int nX, int nY, int nZ)  
void SetPointToQuarter(int nPointNum, int nX, int nY, int nZ)  
int GetIFF()  
int GetFNumber()  
void EnableResearch(string strResearchID, int nEnable)  
void AddResearch(string strResearchID)  
void EnableBuilding(string strBuildingID, int nEnable)  
unitex CreateUnit(int nX, int nY, int nZ, string strScript, string strChasis, string strWeapon0, string strWeapon1, string strWeapon2, string strWeapon3)  
int GetRace()  
void CreateDefaultUnit(int nX, int nY, int nZ)  
void RussianAttack(int nX, int nY, int nZ)  
void Attack(int nX, int nY, int nZ)  
void Attack(unit uTarget)  
void RussianAttack(unit uTarget)  
void ChooseEnemy(player pEnemyPlayer)  
void SetMaxTankPlatoonSize(int nSize)  
void SetMaxHelicopterPlatoonSize(int nSize)  
void SetMaxShipPlatoonSize(int nSize)  
void SetNumberOfOffensiveTankPlatoons(int nCount)  
void SetNumberOfOffensiveShipPlatoons(int nCount)  
void SetNumberOfOffensiveHelicopterPlatoons(int nCount)  
void SetNumberOfDefensiveTankPlatoons(int nCount)  
void SetNumberOfDefensiveShipPlatoons(int nCount)  
void SetNumberOfDefensiveHelicopterPlatoons(int nCount)  
void LoadScript(string strScript)  
void EnableStatistics(int nEnable)  
void CopyResearches(player pFromPlayer)  
void AddToMainTargetList(unit uTarget)  
void ClearMainTargetList()  
void AddToPreferenceList(int nBuildingType)  
void ClearPreferenceList()  
void SetMaxAttackFrequency(int nFrequency)  
int GetMaxAttackFrequency()  
int IsAlive()  
int IsEnemy(player pPlayer)  
int IsAlly(player pPlayer)
```



```
int IsNeutral(player pPlayer)
int GetMoneyFromFinishedMissions()
unitex CreateUnitEx(int nX, int nY, int nZ, string strScript, string strChasis, string
strWeapon0, string strWeapon1, string strWeapon2, string strWeapon3)
unitex CreateUnitEx(int nX, int nY, int nZ, string strScript, string strChasis, string
strWeapon0, string strWeapon1, string strWeapon2, string strWeapon3, int
nShieldUpdateNum)
player GetMainEnemy()
void SetAllowGiveUnits(int bSet)
int IsAllowGiveUnits()
void SetAllowGiveMoney(int bSet)
int IsAllowGiveMoney()
```



Enums



Player AI features:



```
aiBuildMiningBuildings
aiBuildMiningUnits
aiControlMiningUnits
aiBuildSupplyBuildings
aiBuildSupplyUnits
aiControlSupplyUnits
aiBuildPowerBuildings
aiBuildResearchBuildings
aiControlResearches
aiControlDefense
aiControlOffense
aiBuildBuilders
aiBuildRepairers
aiBuildShadows
aiBuildScreamers
aiBuildSuperAttack
aiEnabled
aiBuildTanks
aiBuildShips
aiBuildHelicopters
aiBuildCivilFactory
aiBuildMilitaryFactory
aiBuildShipyard
aiBuildBuildings
aiBuildSpecialUnits
aiRush
aiUpgradeCannons
aiEverything
aiDefenseTowers
aiRejectAlliance
```



Player AI features 2 (same as AI features plus):



```
ai2BuildingTowers
ai2BuildFromStartingPoint
ai2BuildSappers
ai2ChooseEnemy
ai2NeutralAI
ai2BNSSendResult
```



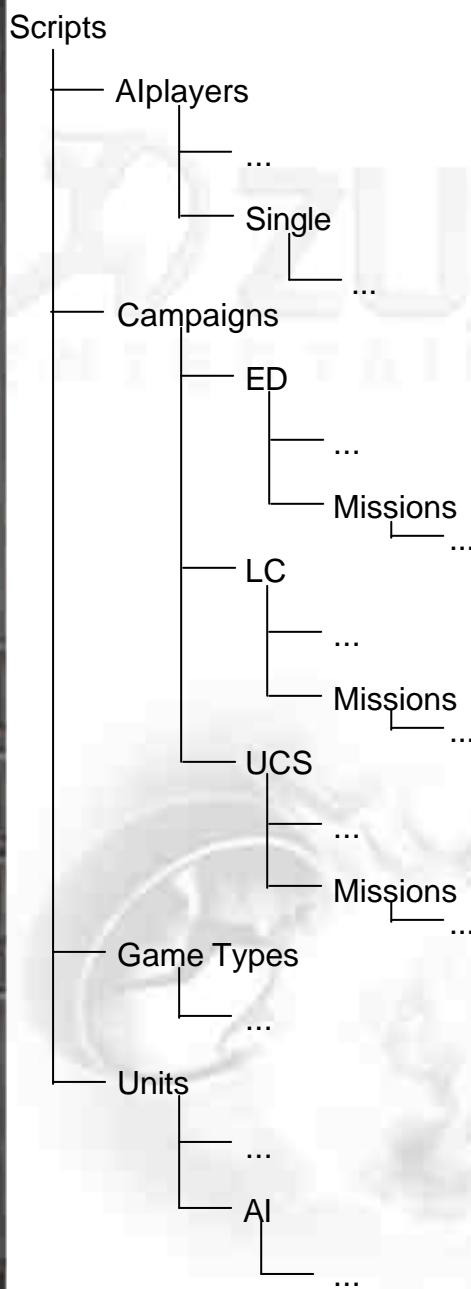
Player races:



```
raceUCS
raceED
raceLC
```

[For further information take a look at our online support!](#)

Scripts directory structure



AIPlayers

Easy.ec

```
player "translateAIPlayerEasy"
{
    state Initialize;
    state SetLimitState;
    state Nothing;

    state Initialize
    {
        if(GetRace() == 1)//ucs
        {
            SetName("translateAIPlayerNameUCSEasy");
            if(GetFFNumber() == 1
                || GetFFNumber() == 6)SetName("translateAINameEasyUCS1");
            if(GetFFNumber() == 2
                || GetFFNumber() == 7)SetName("translateAINameEasyUCS2");
            if(GetFFNumber() == 3
                || GetFFNumber() == 8)SetName("translateAINameEasyUCS3");
            if(GetFFNumber() == 4
                || GetFFNumber() == 9)SetName("translateAINameEasyUCS4");
        }
        if(GetRace() == 2)//ed
        {
            SetName("translateAIPlayerNameEDEasy");
            if(GetFFNumber() == 1
                || GetFFNumber() == 6)SetName("translateAINameEasyED1");
            if(GetFFNumber() == 2
                || GetFFNumber() == 7)SetName("translateAINameEasyED2");
            if(GetFFNumber() == 3
                || GetFFNumber() == 8)SetName("translateAINameEasyED3");
            if(GetFFNumber() == 4
                || GetFFNumber() == 9)SetName("translateAINameEasyED4");
        }
        if(GetRace() == 3)//lc
        {
            SetName("translateAIPlayerNameLCEasy");
            if(GetFFNumber() == 1
                || GetFFNumber() == 6)SetName("translateAINameEasyLC1");
            if(GetFFNumber() == 2
                || GetFFNumber() == 7)SetName("translateAINameEasyLC2");
            if(GetFFNumber() == 3
                || GetFFNumber() == 8)SetName("translateAINameEasyLC3");
            if(GetFFNumber() == 4
                || GetFFNumber() == 9)SetName("translateAINameEasyLC4");
        }
        EnableAIFeatures(aiUpgradeCannons, false);
        EnableAIFeatures(aiRush, false);
        EnableAIFeatures(aiBuildSuperAttack, false);

        SetMaxTankPlatoonSize(3);
        SetMaxHelicopterPlatoonSize(3);
        SetMaxShipPlatoonSize(3);

        SetNumberOffensiveTankPlatoons(1);
        SetNumberOffensiveShipPlatoons(1);
        SetNumberOffensiveHelicopterPlatoons(1);

        SetNumberOfDefensiveTankPlatoons(3);
        SetNumberOfDefensiveShipPlatoons(0);
        SetNumberOfDefensiveHelicopterPlatoons(1);

        SetMaxAttackFrequency(400);
        return SetLimitState, 10;
    }
    state SetLimitState
    {
        if(GetMoney() < 20000) AddMoney(20000 - GetMoney());
        EnableMilitaryUnitsLimit(false);
        return Nothing, 6000;
    }
    state Nothing
    {
        return Nothing, 100000;
    }
}
```

Medium.ec

```
)player "translateAIPlayerMedium"
{
    state Initialize;
    state SetLimitState;
    state Nothing;
    state Initialize
```

```
{ if(GetRace() == 1)//ucs
    {
        SetName("translateAIPlayerNameUCSHard");
        if(GetFFNumber() == 1
            || GetFFNumber() == 6)SetName("translateAINameHardUCS1");
        if(GetFFNumber() == 2
            || GetFFNumber() == 7)SetName("translateAINameHardUCS2");
        if(GetFFNumber() == 3
            || GetFFNumber() == 8)SetName("translateAINameHardUCS3");
        if(GetFFNumber() == 4
            || GetFFNumber() == 9)SetName("translateAINameHardUCS4");
    }
    if(GetRace() == 2)//ed
    {
        SetName("translateAIPlayerNameEDMedium");
        if(GetFFNumber() == 1
            || GetFFNumber() == 6)SetName("translateAINameMediumED1");
        if(GetFFNumber() == 2
            || GetFFNumber() == 7)SetName("translateAINameMediumED2");
        if(GetFFNumber() == 3
            || GetFFNumber() == 8)SetName("translateAINameMediumED3");
        if(GetFFNumber() == 4
            || GetFFNumber() == 9)SetName("translateAINameMediumED4");
    }
    if(GetRace() == 3)//lc
    {
        SetName("translateAIPlayerNameLCMedium");
        if(GetFFNumber() == 1
            || GetFFNumber() == 6)SetName("translateAINameMediumLC1");
        if(GetFFNumber() == 2
            || GetFFNumber() == 7)SetName("translateAINameMediumLC2");
        if(GetFFNumber() == 3
            || GetFFNumber() == 8)SetName("translateAINameMediumLC3");
        if(GetFFNumber() == 4
            || GetFFNumber() == 9)SetName("translateAINameMediumLC4");
    }
    SetMaxTankPlatoonSize(4);
    SetMaxHelicopterPlatoonSize(4);
    SetMaxShipPlatoonSize(4);

    SetNumberOffensiveTankPlatoons(5);
    SetNumberOffensiveShipPlatoons(2);
    SetNumberOffensiveHelicopterPlatoons(2);

    SetNumberOfDefensiveTankPlatoons(2);
    SetNumberOfDefensiveShipPlatoons(0);
    SetNumberOfDefensiveHelicopterPlatoons(1);

    EnableAIFeatures(aiRush, false);
    EnableAIFeatures(aiBuildSuperAttack, false);
    SetMaxAttackFrequency(400);
    return SetLimitState, 10;
}
state SetLimitState
{
    if(GetMoney() < 20000) AddMoney(20000 - GetMoney());
    EnableMilitaryUnitsLimit(false);
    return Nothing, 6000;
}
state Nothing
{
    return Nothing, 100000;
}
```

Hard.ec

```
)player "translateAIPlayerHard"
{
    int nAttackMode;
    state Initialize;
    state SetLimitState;
    state Nothing;

    state Initialize
    {
        if(GetRace() == 1)//ucs
        {
            SetName("translateAIPlayerNameUCSHard");
            if(GetFFNumber() == 1
                || GetFFNumber() == 6)SetName("translateAINameHardUCS1");
            if(GetFFNumber() == 2
                || GetFFNumber() == 7)SetName("translateAINameHardUCS2");
            if(GetFFNumber() == 3
                || GetFFNumber() == 8)SetName("translateAINameHardUCS3");
            if(GetFFNumber() == 4
                || GetFFNumber() == 9)SetName("translateAINameHardUCS4");
        }
    }
```

```

        }
        if((GetRace()==2)//ed
        {
            SetName("translateAIPlayerNameEDHard");
            if(GetFFNumber()==1
            || GetFFNumber()==6)SetName("translateAINameHardED1");
            if(GetFFNumber()==2
            || GetFFNumber()==7)SetName("translateAINameHardED2");
            if(GetFFNumber()==3
            || GetFFNumber()==8)SetName("translateAINameHardED3");
            if(GetFFNumber()==4
            || GetFFNumber()==9)SetName("translateAINameHardED4");
        }
        if((GetRace()==3)//lc
        {
            SetName("translateAIPlayerNameLCHard");
            if(GetFFNumber()==1
            || GetFFNumber()==6)SetName("translateAINameHardLC1");
            if(GetFFNumber()==2
            || GetFFNumber()==7)SetName("translateAINameHardLC2");
            if(GetFFNumber()==3
            || GetFFNumber()==8)SetName("translateAINameHardLC3");
            if(GetFFNumber()==4
            || GetFFNumber()==9)SetName("translateAINameHardLC4");
        }
        SetMaxTankPlatoonSize(6);
        SetMaxHelicopterPlatoonSize(6);
        SetMaxShipPlatoonSize(6);

        SetNumberOfOffensiveTankPlatoons(5);
        SetNumberOfOffensiveShipPlatoons(2);
        SetNumberOfOffensiveHelicopterPlatoons(2);

        SetNumberOfDefensiveTankPlatoons(2);
        SetNumberOfDefensiveShipPlatoons(0);
        SetNumberOfDefensiveHelicopterPlatoons(1);

        if((GetFFNumber()&1)==1)EnableAIFeatures(aiBuildSuperAttack,false);
        nAttackMode=0;
        SetMaxAttackFrequency(400);
        return SetLimitState,10;
    }
    state SetLimitState
    {
        if(GetMoney()<20000) AddMoney(20000-GetMoney());
        EnableMilitaryUnitsLimit(false);
        return Nothing,6000;
    }
    state Nothing
    {
        if(!nAttackMode)
        {
            nAttackMode=1;
            SetMaxAttackFrequency(5);
            return Nothing,600//30s
        }
        nAttackMode=0;
        SetMaxAttackFrequency(800);
        return Nothing,6000//5min 1min=60*20=1200
    }
}

Expert.ec
player "translateAIPlayerExpert"
{
    int nAttackMode;
    state Initialize;
    state SetLimitState;
    state Nothing;

    state Initialize
    {
        SetName("translateAIPlayerExpert");

        SetMaxTankPlatoonSize(6);
        SetMaxHelicopterPlatoonSize(6);
        SetMaxShipPlatoonSize(6);

        SetNumberOfOffensiveTankPlatoons(4);
        SetNumberOfOffensiveShipPlatoons(1);
        SetNumberOfOffensiveHelicopterPlatoons(4);

        SetNumberOfDefensiveTankPlatoons(1);
        SetNumberOfDefensiveShipPlatoons(0);
    }

    SetNumberOfOffensiveHelicopterPlatoons(1);

    if(GetFFNumber()==1
    || GetFFNumber()==6)SetNumberOfOffensiveTankPlatoons(2);
    if(GetFFNumber()==2
    || GetFFNumber()==7)SetNumberOfDefensiveTankPlatoons(0);
    if(GetFFNumber()==3
    || GetFFNumber()==8)SetNumberOfDefensiveTankPlatoons(5);
    if(GetFFNumber()==4
    || GetFFNumber()==9){SetMaxTankPlatoonSize(3);SetMaxHelicopterPlatoonSize(8);}

    AddResearch("RES_MCH2");
    AddResearch("RES_MCH3");
    AddResearch("RES_MCH4");
    AddResearch("RES_MSR2");
    AddResearch("RES_MSR3");
    AddResearch("RES_MSR4");
    AddResearch("RES_MM2");
    AddResearch("RES_MM3");
    AddResearch("RES_MM4");

    if(GetRace()==1)//UCS
    {
        AddResearch("RES_UCS_UOH2");
        AddResearch("RES_UCS_UOH3");
        AddResearch("RES_UCS_UAH1");
        AddResearch("RES_UCS_UAH2");
        AddResearch("RES_UCS_UAH3");
        AddResearch("RES_UCS_GARG1");
        AddResearch("RES_UCS_MB2");
        AddResearch("RES_UCS_MB3");
        AddResearch("RES_UCS_MB4");
        AddResearch("RES_UCS_MG2");
        AddResearch("RES_UCS_MG3");
        AddResearch("RES_UCS_MG4");
    }

    if(GetRace()==2)//ed
    {
        AddResearch("RES_ED_MSC2");
        AddResearch("RES_ED_MSC3");
        AddResearch("RES_ED_MSC4");
        AddResearch("RES_ED_MHC2");
        AddResearch("RES_ED_MHC3");
        AddResearch("RES_ED_MHC4");
        AddResearch("RES_ED_MB2");
        AddResearch("RES_ED_MB3");
        AddResearch("RES_ED_MB4");
        AddResearch("RES_ED_UT2");
        AddResearch("RES_ED_UT3");
        AddResearch("RES_ED_UMT1");
        AddResearch("RES_ED_UMT2");
        AddResearch("RES_ED_UMT3");
        AddResearch("RES_ED_UM11");
        AddResearch("RES_ED_UM12");
        AddResearch("RES_ED_UA11");
    }

    if(GetRace()==3)//lc
    {
        AddResearch("RES_LC_ULU2");
        AddResearch("RES_LC_ULU3");
        AddResearch("RES_LC_UMO2");
        AddResearch("RES_LC_UMO3");
        AddResearch("RES_LC_UME1");
    }

    nAttackMode=0;
    SetMaxAttackFrequency(400);

    return SetLimitState,10;
}

state SetLimitState
{
    if(GetMoney()<20000) AddMoney(20000-GetMoney());
    EnableMilitaryUnitsLimit(false);
    return Nothing,6000;
}

state Nothing
{
    //if(GetMoney()<20000)AddMoney(20000);

    if(!nAttackMode)

```

```

    {
        nAttackMode=1;
        SetMaxAttackFrequency(20);
        return Nothing,200//1min=60*20=1200
    }
    nAttackMode=0;
    SetMaxAttackFrequency(800);
    return Nothing,5000//1min=60*20=1200
}
}

```

Single

Single Easy

player "translateAIPlayerEasy"

```

{
    state Initialize;
    state Nothing;

    state Initialize
    {
        if(GetRace() == 1)//ucs
        {
            SetName("translateAIPlayerNameUCSEasy");
            if(GetIFFNumber() == 1)
                SetName("translateAINameEasyUCS1");
            if(GetIFFNumber() == 2)
                SetName("translateAINameEasyUCS2");
            if(GetIFFNumber() == 3)
                SetName("translateAINameEasyUCS3");
            if(GetIFFNumber() == 4)
                SetName("translateAINameEasyUCS4");
        }
        if(GetRace() == 2)//ed
        {
            SetName("translateAIPlayerNameEDEasy");
            if(GetIFFNumber() == 1)
                SetName("translateAINameEasyED1");
            if(GetIFFNumber() == 2)
                SetName("translateAINameEasyED2");
            if(GetIFFNumber() == 3)
                SetName("translateAINameEasyED3");
            if(GetIFFNumber() == 4)
                SetName("translateAINameEasyED4");
        }
        if(GetRace() == 3)//lc
        {
            SetName("translateAIPlayerNameLCEasy");
            if(GetIFFNumber() == 1)
                SetName("translateAINameEasyLC1");
            if(GetIFFNumber() == 2)
                SetName("translateAINameEasyLC2");
            if(GetIFFNumber() == 3)
                SetName("translateAINameEasyLC3");
            if(GetIFFNumber() == 4)
                SetName("translateAINameEasyLC4");
        }
        EnableAIFeatures(aiUpgradeCannons,false);
        EnableAIFeatures(aiRush,false);

        SetMaxTankPlatoonSize(3);
        SetMaxHelicopterPlatoonSize(3);
        SetMaxShipPlatoonSize(3);

        SetNumberOfOffensiveTankPlatoons(1);
        SetNumberOfOffensiveShipPlatoons(1);
        SetNumberOfOffensiveHelicopterPlatoons(1);

        SetNumberOfDefensiveTankPlatoons(3);
        SetNumberOfDefensiveShipPlatoons(0);
        SetNumberOfDefensiveHelicopterPlatoons(2);
        SetMaxAttackFrequency(800);
        return Nothing;
    }
    state Nothing
    {
        return Nothing,100000;
    }
}

```

Single Medium

player "translateAIPlayerMedium"

```
{
}
```

```

state Initialize;
state Nothing;

state Initialize
{
    if(GetRace() == 1)//ucs
    {
        SetName("translateAIPlayerNameUCSMedium");
        if(GetIFFNumber() == 1)
            SetName("translateAINameMediumUCS1");
        if(GetIFFNumber() == 2)
            SetName("translateAINameMediumUCS2");
        if(GetIFFNumber() == 3)
            SetName("translateAINameMediumUCS3");
        if(GetIFFNumber() == 4)
            SetName("translateAINameMediumUCS4");
    }
    if(GetRace() == 2)//ed
    {
        SetName("translateAIPlayerNameEDMedium");
        if(GetIFFNumber() == 1)
            SetName("translateAINameMediumED1");
        if(GetIFFNumber() == 2)
            SetName("translateAINameMediumED2");
        if(GetIFFNumber() == 3)
            SetName("translateAINameMediumED3");
        if(GetIFFNumber() == 4)
            SetName("translateAINameMediumED4");
    }
    if(GetRace() == 3)//lc
    {
        SetName("translateAIPlayerNameLCMedium");
        if(GetIFFNumber() == 1)
            SetName("translateAINameMediumLC1");
        if(GetIFFNumber() == 2)
            SetName("translateAINameMediumLC2");
        if(GetIFFNumber() == 3)
            SetName("translateAINameMediumLC3");
        if(GetIFFNumber() == 4)
            SetName("translateAINameMediumLC4");
    }
}

SetMaxTankPlatoonSize(4);
SetMaxHelicopterPlatoonSize(4);
SetMaxShipPlatoonSize(4);

SetNumberOfOffensiveTankPlatoons(3);
SetNumberOfOffensiveShipPlatoons(2);
SetNumberOfOffensiveHelicopterPlatoons(2);

SetNumberOfDefensiveTankPlatoons(4);
SetNumberOfDefensiveShipPlatoons(0);
SetNumberOfDefensiveHelicopterPlatoons(2);
EnableAIFeatures(aiRush,false);
SetMaxAttackFrequency(400);
return Nothing;
}

state Nothing
{
    return Nothing,100000;
}
}

```

Single Hard

player "translateAIPlayerHard"

```

{
    int nAttackMode;
    state Initialize;
    state Nothing;

    state Initialize
    {
        if(GetRace() == 1)//ucs
        {
            SetName("translateAIPlayerNameUCSHard");
            if(GetIFFNumber() == 1)
                SetName("translateAINameHardUCS1");
            if(GetIFFNumber() == 2)
                SetName("translateAINameHardUCS2");
            if(GetIFFNumber() == 3)
                SetName("translateAINameHardUCS3");
            if(GetIFFNumber() == 4)
                SetName("translateAINameHardUCS4");
        }
    }
}

```

```

        }
        if(GetRace() == 2) //ed
        {
            SetName("translateAIPlayerNameEDHard");
            if(GetFFNumber() == 1)
                || GetFFNumber() == 6) SetName("translateAINameHardED1");
            if(GetFFNumber() == 2)
                || GetFFNumber() == 7) SetName("translateAINameHardED2");
            if(GetFFNumber() == 3)
                || GetFFNumber() == 8) SetName("translateAINameHardED3");
            if(GetFFNumber() == 4)
                || GetFFNumber() == 9) SetName("translateAINameHardED4");
        }
        if(GetRace() == 3) //lc
        {
            SetName("translateAIPlayerNameLCHard");
            if(GetFFNumber() == 1)
                || GetFFNumber() == 6) SetName("translateAINameHardLC1");
            if(GetFFNumber() == 2)
                || GetFFNumber() == 7) SetName("translateAINameHardLC2");
            if(GetFFNumber() == 3)
                || GetFFNumber() == 8) SetName("translateAINameHardLC3");
            if(GetFFNumber() == 4)
                || GetFFNumber() == 9) SetName("translateAINameHardLC4");
        }
        SetMaxTankPlatoonSize(6);
        SetMaxHelicopterPlatoonSize(6);
        SetMaxShipPlatoonSize(6);

        SetNumberOfOffensiveTankPlatoons(4);
        SetNumberOfOffensiveShipPlatoons(4);
        SetNumberOfOffensiveHelicopterPlatoons(4);

        SetNumberOfDefensiveTankPlatoons(4);
        SetNumberOfDefensiveShipPlatoons(0);
        SetNumberOfDefensiveHelicopterPlatoons(3);
        SetMaxAttackFrequency(400);
        return Nothing, 6000;
    }
    state Nothing
    {
        if(lnAttackMode)
        {
            nAttackMode = 1;
            SetMaxAttackFrequency(20);
            return Nothing, 200;
        }
        nAttackMode = 0;
        SetMaxAttackFrequency(800);
        return Nothing, 6000; //1min=60*20=1200
    }
}

```

Campaigns

ED

CampaignED.ec

```

campaign "translateCampaignED"
{
    consts
    {
        raceUCS = 1;
        raceED = 2;
        raceLC = 3;

        mis211 = 0;
        mis212 = 1;
        mis213 = 2;
        mis214 = 3;
        mis215 = 4;
        mis221 = 5;
        mis222 = 6;
        mis223 = 7;
        mis224 = 8;
        mis231 = 9;
        mis232 = 10;
        mis233 = 11;
        mis234 = 12;
        mis235 = 13;
        mis241 = 14;
        mis242 = 15;
        mis243 = 16;
        mis244 = 17;
        mis251 = 18;
        mis252 = 19;
        mis253 = 20;
        mis254 = 21;
        mis261 = 22;
        mis262 = 23;
        mis263 = 24;
        mis264 = 25;
        mis265 = 26;
        mis271 = 27;
        mis272 = 28;
        mis273 = 29;
    }

    baseUCS = 30;
    baseED = 31;
    baseLC = 32;

    timeFlagWinter = 1;
    timeFlagEarlySpring = 2;
    timeFlagSpring = 4;
    timeFlagSummer = 8;
    timeFlagDesert = 16;
    timeFlagVolcano = 32;
    timeFlagHeavyVolcano = 64;
    baseFlag = 255;

    // pieriadz przewidywane do budowy statku na poczatku okresu
    /*cashEarlySpring = 50000;
    cashSpring = 100000;
    cashSummer = 150000;
    cashDesert = 200000;
    cashVolcano = 250000;
    spaceShipPrice = 300000;
    */
    // czas zmiany poru roku
    timeEarlySpring = 400000; //= 20000 sec = 333 min = 5h 33 min
    timeSpring = 800000;
    timeSummer = 1200000;
    timeDesert = 1600000;
    timeVolcano = 2000000;
    timeHeavyVolcano = 2400000;
    timeEarthDestroyed = 2800000; // = 38h 30min
}

int n_TimeFlag;
int n_CashCounter;
int b_showVideo;
int b_showVideo2;

state Initialize;
state Start;
state Nothing;

state Initialize
{
    int i;
    RegisterMission(baseUCS, "baseUCS", "ED\\Missions\\baseUCSscript", "");
    baseFlag, -84, 41, 0, 0);
    RegisterMission(baseED, "baseED", "ED\\Missions\\baseEDscript", "",
    baseFlag, 50, 60, 0, 0);
    RegisterMission(baseLC, "baseLC", "ED\\Missions\\baseLCscript", "",
    baseFlag, 120, 8, 0, 0);
    // nr lnd script preBriefing
    timeFlags x y d1 d2 d3 next1 next2 next3 next4
    RegisterMission(mis211, "I211", "ED\\Missions\\Mission211",
    "translateBriefingShort211", timeFlagWinter, 60, 60, 60, 60,
    mis212,mis213);
    RegisterMission(mis212, "I212", "ED\\Missions\\Mission212",
    "translateBriefingShort212", timeFlagWinter, 60, 78, 90, 90, 90,
    mis215,mis214);
    RegisterMission(mis213, "I213", "ED\\Missions\\Mission213",
    "translateBriefingShort213", timeFlagWinter, 90, 35, 90, 90, 90);
    RegisterMission(mis214, "I214", "ED\\Missions\\Mission214",
    "translateBriefingShort214", timeFlagWinter, 120, 85, 120, 120, 120);
    RegisterMission(mis215, "I215", "ED\\Missions\\Mission215",
    "translateBriefingShort215", timeFlagWinter, 161, 60, 90, 90, 90,
    mis221,mis223,mis224);
    //-----
    RegisterMission(mis221, "I221", "ED\\Missions\\Mission221",
    "translateBriefingShort221", timeFlagEarlySpring, 105, 52, 90, 90, 90,
    mis223,mis224);
}

```

```

        RegisterMission(mis222, "I222", "ED\Missions\Mission222",
"translateBriefingShort222", timeFlagEarlySpring,-115, 65,300,300,300,
mis231,mis233);
        RegisterMission(mis223, "I223", "ED\Missions\Mission223",
"translateBriefingShort223", timeFlagEarlySpring,-150, 65,250,250,250,
mis222);
        RegisterMission(mis224, "I224", "ED\Missions\Mission224",
"translateBriefingShort224", timeFlagEarlySpring, -140, 40,150,150,150,
mis232,mis234);
    //-----
    RegisterMission(mis231, "I231", "ED\Missions\Mission231",
"translateBriefingShort231", timeFlagSpring, -90, 42,300,300,300,
mis232,mis235);
        RegisterMission(mis232, "I232", "ED\Missions\Mission232",
"translateBriefingShort232", timeFlagSpring, -122, 11,150,150,150,
mis241,mis242,mis243);
        RegisterMission(mis233, "I233", "ED\Missions\Mission233",
"translateBriefingShort233", timeFlagSpring, -74, 41,300,300,300,
baseUCS);
        RegisterMission(mis234, "I234", "ED\Missions\Mission234",
"translateBriefingShort234", timeFlagSpring, -52, -2,360,360,360,
mis241,mis242,mis243);
        RegisterMission(mis235, "I235", "ED\Missions\Mission235",
"translateBriefingShort235", timeFlagSpring, -110, 40,360,360,360);
    //-----
    RegisterMission(mis241, "I241", "ED\Missions\Mission241",
"translateBriefingShort241", timeFlagSummer, -80, 9,330,30,330,30);
        RegisterMission(mis242, "I242", "ED\Missions\Mission242",
"translateBriefingShort242", timeFlagSummer, 77, 15, 90, 90, 90);
        RegisterMission(mis243, "I243", "ED\Missions\Mission243",
"translateBriefingShort243", timeFlagSummer, 47,-18,250,250,250,
mis244);
        RegisterMission(mis244, "I244", "ED\Missions\Mission244",
"translateBriefingShort244", timeFlagSummer, 133,-13,300,300,300,
mis251,mis252);
    //-----
    RegisterMission(mis251, "I251", "ED\Missions\Mission251",
"translateBriefingShort251", timeFlagDesert, 35,15,250,250,250,
mis253);
        RegisterMission(mis252, "I252", "ED\Missions\Mission252",
"translateBriefingShort252", timeFlagDesert, 135,-25,300,300,300,
mis254);
        RegisterMission(mis253, "I253", "ED\Missions\Mission253",
"translateBriefingShort253", timeFlagDesert, 31,27,100,100,100,
mis261,mis262,mis263);
        RegisterMission(mis254, "I254", "ED\Missions\Mission254",
"translateBriefingShort254", timeFlagDesert, 166,-45,300,300,300,
mis264,mis265);
    //-----
    RegisterMission(mis261, "I261", "ED\Missions\Mission261",
"translateBriefingShort261", timeFlagVolcano, 13,-3,250,250,250);
        RegisterMission(mis262, "I262", "ED\Missions\Mission262",
"translateBriefingShort262", timeFlagVolcano, 26,-27,260,260,260);
        RegisterMission(mis263, "I263", "ED\Missions\Mission263",
"translateBriefingShort263", timeFlagVolcano, -15, 20,250,250,250,
mis271,mis272,mis273);
        RegisterMission(mis264, "I264", "ED\Missions\Mission264",
"translateBriefingShort264", timeFlagVolcano, -72, 3,150,150,150);
        RegisterMission(mis265, "I265", "ED\Missions\Mission265",
"translateBriefingShort265", timeFlagVolcano, -75,-5,150,150,150,
mis271,mis272,mis273);
    //-----
    RegisterMission(mis271, "I271", "ED\Missions\Mission271",
"translateBriefingShort271", timeFlagHeavyVolcano,-70, -5,360,360,360);
        RegisterMission(mis272, "I272", "ED\Missions\Mission272",
"translateBriefingShort272", timeFlagHeavyVolcano, -70,-30,360,360,360);
        RegisterMission(mis273, "I273", "ED\Missions\Mission273",
"translateBriefingShort273", timeFlagHeavyVolcano, -77,-8,360,360,360);
    //-----
    n_CashCounter = timeFlagEarlySpring;
    n_TimeFlag = timeFlagWinter;
    b_showVideo = true;
    b_showVideo2 = false;

CreateGamePlayer(1,raceUCS,playerAl,null);
CreateGamePlayer(2,raceED,playerLocal,null);
CreateGamePlayer(3,raceLC,playerAl,null);

LoadBase(1,baseUCS,1);

        LoadBase(2,baseED,2);
        LoadBase(3,baseLC,3);
        SetTime(100);
        SetActivePlayerAndWorld(2,2);/player, world
        SetAvailableWorlds(1 | 4);/misja i baza ed(swiat nr 2)

        SetSeason(1);
        EnableMission(mis211,true);
        ActivateMissions(timeFlagWinter,true);

        return Start,240;
    //-----
state Start
{
    EnableChooseMissionButton(true);
    return Nothing,50;
}
//-----
state Nothing
{
    if(b_showVideo2)
    {
        b_showVideo2 = false;

        if(n_TimeFlag == timeFlagWinter)
            ShowVideo("CS214");

        if(n_TimeFlag == timeFlagSpring)
            ShowVideo("CS215");

        if(n_TimeFlag == timeFlagDesert)
            ShowVideo("CS216");

        if(n_TimeFlag == timeFlagVolcano)
            ShowVideo("CS217");
    }
    return Nothing,50;
}
//-----
event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    TraceD(Mission);
    LoadMission(0,iMission);

    EnableMission(iMission,false);

    if(b_showVideo)
    {
        b_showVideo = false;
        b_showVideo2 = true;
    }
}
//-----
event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
    if(bEnable<2)
    {
        EnableMission(GetNextMission(iMission,iNextNr),bEnable);
        // SetMissionState(iMission,stateSkipped);
    }
    if(bEnable==2)//enable UCS base
        SetAvailableWorlds(GetAvailableWorlds() | 2);
    if(bEnable==3)//enable LC base
        SetAvailableWorlds(GetAvailableWorlds() | 8);
    if(bEnable==4)//you have won the game
    {
        EndGame("Video\OutroED.wd1");
    }
    if(bEnable==5)//you loose the game
    {
        EndGame("Video\OutroLost.wd1");
    }
}
//-----
event EndMission(int iMission, int nResult) //
{
    if(nResult==true)
        SetMissionState(iMission,stateAccomplished);
    else
        SetMissionState(iMission,stateFailed);

    if(!AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
    {

```

```

if(n_TimeFlag != timeFlagHeavyVolcano)
{
    n_TimeFlag = n_TimeFlag*2;
    b_showVideo = true;
}
}

if(AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
{
    ActivateMissionsEq(n_TimeFlag,n_TimeFlag,true);
}
else
{
    SendCustomEvent(0,0,0,0,0);
    return;
}
SetSeason(1);
if(n_TimeFlag==timeFlagEarlySpring)SetSeason(2);
if(n_TimeFlag==timeFlagSpring) SetSeason(3);
if(n_TimeFlag==timeFlagSummer) SetSeason(4);
if(n_TimeFlag==timeFlagDesert) SetSeason(5);
if(n_TimeFlag==timeFlagVolcano) SetSeason(6);
if(n_TimeFlag==timeFlagHeavyVolcano) SetSeason(7);

EnableChooseMissionButton(true);
}

/*event EndMission(int iMission, int nResult) // old campaign structure
{
int nTime;
int nNewTimeFlag;
nTime = GetCampaignTime();
//SetMissionState(nr,stan);stateAccomplished,stateFailed, stateSkipped
if(nResult==true)
SetMissionState(Mission,stateAccomplished);
else
SetMissionState(iMission,stateFailed);

nNewTimeFlag = timeFlagWinter;
if(nTime > timeEarlySpring)
nNewTimeFlag = timeFlagEarlySpring;
if(nTime > timeSpring)
nNewTimeFlag = timeFlagSpring;
if(nTime > timeSummer)
nNewTimeFlag = timeFlagSummer;
if(nTime>timeDesert)
nNewTimeFlag = timeFlagDesert;
if(nTime>timeVolcano)
nNewTimeFlag = timeFlagVolcano;
if(nTime>timeHeavyVolcano)
nNewTimeFlag = timeFlagHeavyVolcano;

if(nNewTimeFlag != n_TimeFlag) //zmiana czasu
{
nNewTimeFlag = n_TimeFlag*2;
if(AreMissionsEnabledEq(nNewTimeFlag,nNewTimeFlag))
{
    EnableMissionsEq(n_TimeFlag,n_TimeFlag,false);
    n_TimeFlag = nNewTimeFlag; //
    b_showVideo = true;
}
}
else
{
if(!AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
{
    n_TimeFlag = n_TimeFlag*2;
}
}
if(AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
{
    ActivateMissionsEq(n_TimeFlag,n_TimeFlag,true);
}
else
{
    SendCustomEvent(0,0,0,0,0);
    return;
}
SetSeason(1);
if(n_TimeFlag==timeFlagEarlySpring)SetSeason(2);
if(n_TimeFlag==timeFlagSpring) SetSeason(3);
if(n_TimeFlag==timeFlagSummer) SetSeason(4);
if(n_TimeFlag==timeFlagDesert) SetSeason(5);
if(n_TimeFlag==timeFlagVolcano) SetSeason(6);
if(n_TimeFlag==timeFlagHeavyVolcano) SetSeason(7);

EnableChooseMissionButton(true);
}

EnableChooseMissionButton(true);
}

} */
}

//*****
TutorialED.ec

campaign "translateCampaignTutorialED"
{
state Initialize;
state Nothing;

state Initialize
{
    CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");
    CreateGamePlayer(2,raceED,playerLocal,"TestGameAI");

    SetTime(100);

RegisterMission(0,"TutorialED","ED\\Missions\\TutorialEDmis","","0,0,0,0,0,0);
LoadMission(0,0);
SetAvailableWorlds(1);
SetActivePlayerAndWorld(2,0);
return Nothing;
}
state Nothing
{
    return Nothing,200;
}
}

CampaignED-Demo.ec

campaign "translateCampaignED"
{
consts
{
    raceUCS=1;
    raceED=2;
    raceLC=3;

    mis211 = 0;

    baseUCS = 30;
    baseED = 31;
    baseLC = 32;

    timeFlagWinter = 1;
    timeFlagEarlySpring = 2;
    timeFlagSpring = 4;
    timeFlagSummer = 8;
    timeFlagDesert = 16;
    timeFlagVolcano = 32;
    timeFlagHeavyVolcano = 64;
    baseFlag = 255;

    // czas zmiany por
    timeEarlySpring = 400000; //= 20000 sec = 333 min = 5h 33 min
    timeSpring = 800000;
    timeSummer = 1200000;
    timeDesert = 1600000;
    timeVolcano = 2000000;
    timeHeavyVolcano = 2400000;
    timeEarthDestroyed=2800000; // = 38h 30min
}

int n_TimeFlag;
int n_CashCounter;
int b_showVideo;
int b_showVideo2;

state Initialize;
state Start;
state Nothing;

state Initialize
{
    int i;

    RegisterMission(baseUCS,"baseUCS","ED\\Missions\\baseUCSscript","");
    baseFlag,-84,41,0,0,0);
    RegisterMission(baseED,"baseED","ED\\Missions\\baseEDscriptDemo","");
    baseFlag,50,60,0,0,0);
    RegisterMission(baseLC,"baseLC","ED\\Missions\\baseLCscript","");
}

```

Missions

baseEDscript.ec

```
baseFlag          120, 8.0.0.0;
//          nr   Ind   script      preBriefing
timeFlags        x,y d1 d2 d3 next1 next2 next3 next4
RegisterMission(mis211, "Z11", "ED\{Missions\}Mission211Demo",
"translateBriefingShort211", timeFlagWinter,           60, 60, 60, 60, 60);
//-----  
  
n_CashCounter = timeFlagEarlySpring;
n_TimeFlag = timeFlagWinter;
b_showVideo = true;
b_showVideo2 = false;  
  
CreateGamePlayer(1,raceUCS,playerAI,null);
CreateGamePlayer(2,raceED,playerLocal,null);
CreateGamePlayer(3,raceLC,playerAI,null);  
  
LoadBase(1,baseUCS,1);
LoadBase(2,baseED,2);
LoadBase(3,baseLC,3);
SetTime(100);
SetActivePlayerAndWorld(2,2); //player, world
SetAvailableWorlds(1 | 4); //misja i baza ed(swiat nr 2)  
  
SetSeason(1);
EnableMission(mis211,true);
ActivateMissions(timeFlagWinter,true);  
  
return Start,240;
}  
//-----  
state Start
{
    EnableChooseMissionButton(true);
    return Nothing,50;
}
//-----  
state Nothing
{
    if(b_showVideo2)
    {
        b_showVideo2 = false;

        if(n_TimeFlag == timeFlagWinter)
            ShowVideo("CS214");
    }
    return Nothing,50;
}
//-----  
event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    TraceD(iMission);
    LoadMission(0,iMission);

    EnableMission(iMission,false);

    if(b_showVideo)
    {
        b_showVideo = false;
        b_showVideo2 = true;
    }
}
//-----  
event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
}
//-----  
event EndMission(int iMission, int nResult) //
{
    int nTime;
    int nNewTimeFlag;
    nTime = GetCampaignTime();
    //SetMissionState(nr,stan); stateAccomplished, stateFailed, stateSkipped
    if(nResult==true)
        SetMissionState(iMission,stateAccomplished);
    else
        SetMissionState(iMission,stateFailed);
}
//*****  
  
mission "translateMissionEDBaseED"
{
    consts
    {
        oneSeasonMoney = 50000; //CR
        oneSeasonTime = 150000; //clk = 2h05min
        noOfSeasons = 20;
        fleetCost = 1000000;
        firstPhaseCost = 100000;
        phaseCost = 200000;
        clicksPerDay = 16383;
    }
    player p_Player;
    int nReport;
    int bMissionsFinished;
    int EndGameResult;

    state Initialize;
    state ShowBriefing;
    state ShowStartFirstMissionBriefing;
    state Nothing;
    state EndGameState;

    state Initialize
    {
        nReport = 0;
        RegisterGoal(0,"translateGoalSendMoneyToSpace",fleetCost,0);
        RegisterGoal(1,"translateGoalEDCampaignPhase1");
        RegisterGoal(2,"translateGoalEDCampaignPhase2");
        RegisterGoal(3,"translateGoalEDCampaignPhase3");
        RegisterGoal(4,"translateGoalEDCampaignPhase4");
        RegisterGoal(5,"translateGoalEDCampaignPhase5");
        EnableGoal(0,true);
        EnableGoal(1,true);
        EnableGoal(2,true);
        EnableGoal(3,true);
        EnableGoal(4,true);
        EnableGoal(5,true);

        p_Player = GetPlayer(2);
        p_Player.SetMoney(5000);
        p_Player.SetMilitaryUnitsLimit(10000);

        SetTimer(0,oneSeasonTime);

        p_Player.SetScriptData(0,fleetCost);

        //weapons
        p_Player.EnableResearch("RES_ED_WCH2",false);
        p_Player.EnableResearch("RES_ED_WCA2",false);
        p_Player.EnableResearch("RES_ED_WH1",false);
        p_Player.EnableResearch("RES_ED_WH2",false);
        p_Player.EnableResearch("RES_ED_WSR1",false);
        p_Player.EnableResearch("RES_ED_WSR2",false);
        p_Player.EnableResearch("RES_ED_MMR1",false);
        p_Player.EnableResearch("RES_ED_WHR1",false);
        p_Player.EnableResearch("RES_ED_WSL1",false);
        p_Player.EnableResearch("RES_ED_WH1",false);
        p_Player.EnableResearch("RES_ED_WSI1",false);
        p_Player.EnableResearch("RES_ED_AB1",false);

        //ammo
        p_Player.EnableResearch("RES_MCH2",false);
        p_Player.EnableResearch("RES_ED_MSC2",false);
        p_Player.EnableResearch("RES_ED_MH2",false);
        p_Player.EnableResearch("RES_MR2",false);
        p_Player.EnableResearch("RES_MMR2",false);
        p_Player.EnableResearch("RES_ED_MHR",false);
        p_Player.EnableResearch("RES_ED_MB2",false);
        //chassis
        p_Player.EnableResearch("RES_ED_UHT1",false);
        p_Player.EnableResearch("RES_ED_UBT1",false);

        p_Player.EnableResearch("RES_ED_UT2",false);
        p_Player.EnableResearch("RES_ED_UT3",false);
        p_Player.EnableResearch("RES_ED_UMT1",false);
        p_Player.EnableResearch("RES_ED_UM11",false);
        p_Player.EnableResearch("RES_ED_UMW1",false);
    }
}
```

```

p_Player.EnableResearch("RES_ED_UHW1",false);
p_Player.EnableResearch("RES_ED_USS2",false);
p_Player.EnableResearch("RES_ED_UHS1",false);

p_Player.EnableResearch("RES_ED_UA11",false);
p_Player.EnableResearch("RES_ED_UA12",false);
p_Player.EnableResearch("RES_ED_UA21",false);
p_Player.EnableResearch("RES_ED_UA22",false);
p_Player.EnableResearch("RES_ED_UA31",false);
p_Player.EnableResearch("RES_ED_UA41",false);

//special
p_Player.EnableResearch("RES_ED_BMD",false);
p_Player.EnableResearch("RES_ED_BHD",false);
p_Player.EnableResearch("RES_ED_RePhant",false);
p_Player.EnableResearch("RES_ED_RePhant2",false);
p_Player.EnableResearch("RES_ED_SCR",false);
p_Player.EnableResearch("RES_ED_SGen",false);

bMissionsFinished=false;
//BUILDINGS
p_Player.EnableBuilding("EDBTC",false);

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),4,0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
nReport = 0;
AddBriefing("translateStartCampaign",fleetCost);

Show(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,8000,800
,5);
return ShowStartFirstMissionBriefing,140;
}
//-----
state ShowStartFirstMissionBriefing
{
AddBriefing("translateStartFirstMission");
return Nothing,50;
}
//-----
state Nothing
{
int nSendedMoney;
int nReTime;
int i;
int k;
nSendedMoney = p_Player.GetMoneySentToOrbit();
}

RegisterGoal(0,"translateGoalSendMoneyToSpace",fleetCost,(nSendedMoney*100)
/fleetCost);
for(=1;i<6;j=i+1)
{
k=firstPhaseCost+(phaseCost*(i-1));
if(nSendedMoney>=k)
{
SetGoalState(i,goalAchieved);
}
}
if(nSendedMoney>=fleetCost)
{
SetGoalState(0,goalAchieved);
AddBriefing("translateCampaignAccomplished");
EndGameState=4;
return EndGameState,3;
}
if(!p_Player.GetNumberOfBuildings(buildingSpaceStation))
{
}

AddBriefing("translateCampaignEDSpacePortDestroyed",p_Player.GetName());
EndGameState=5;
return EndGameState,3;
}

if(bMissionsFinished &&
(nSendedMoney+p_Player.GetMoney())+p_Player.GetMoneyFromFinishedMissions(
))<fleetCost)
{
AddBriefing("translateCampaignFailed");
EndGameState=5;
}

return EndGameState,3;
}

SetConsoleText("translateCampaignEDRemainingTime",((oneSeasonTime*noOfSeas
ons) - GetMissionTime())/clicksPerDay);/XXXMD
return Nothing,600;
}
//-----
state EndGameState
{
EnableNextMission(0,EndGameResult);
return EndGameState,600;
}

//-----
event Timer0() //wolany co 200 000 cykli< ustawnione funkcja SetTimer w state
Initialise
{
int nSendedMoney;
int nProjectStatus;
int nMoneyForNextSeason;

nReport = nReport + 1;

if(nReport>noOfSeasons)
{
AddBriefing("translateCampaignFailed");
EnableNextMission(0,5);
return;
}

nSendedMoney = p_Player.GetMoneySentToOrbit(); // to ma tu byc
nProjectStatus = (nSendedMoney*100)/fleetCost;
nMoneyForNextSeason = ((nReport+1)*oneSeasonMoney) -
nSendedMoney;
if(nMoneyForNextSeason<10000) nMoneyForNextSeason=10000;

if(nSendedMoney < oneSeasonMoney*nReport)

AddBriefing("translateDelayReport",nReport,nProjectStatus,oneSeasonMoney*nRe
port,nSendedMoney,nMoneyForNextSeason);
else

AddBriefing("translateReport",nReport,nProjectStatus,oneSeasonMoney*nReport,
nSendedMoney,nMoneyForNextSeason);
}

event CustomEvent0(int k1,int k2,int k3,int k4) //XXXMD
{
bMissionsFinished=true;
}
}



## baseEDscriptDemo.ec


mission "translateMissionEDBaseED"
{
consts
{
oneSeasonMoney = 50000; //CR
oneSeasonTime = 150000; //clk = 2h05min
noOfSeasons = 20;
fleetCost = 1000000;
firstPhaseCost = 100000;
phaseCost = 200000;
clicksPerDay = 16383;
}

player p_Player;
int nReport;

state Initialize;
state ShowBriefing;
state ShowStartFirstMissionBriefing;
state Nothing;
state EndDemo;

state Initialize
{
nReport = 0;
RegisterGoal(0,"translateGoalSendMoneyToSpace",fleetCost,0);
EnableGoal(0,true);
p_Player = GetPlayer(2);
p_Player.SetMoney(2500);
}
}

```

```

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY() +9.6,0,
20,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY()
-4.6,0,20,0,60,1);

SetTimer(0,oneSeasonTime);

p_Player.SetScriptData(fleetCost);
p_Player.SetScriptData(1,0);

//weapons
//p_Player.EnableResearch("RES_ED_WCH2",false);
//p_Player.EnableResearch("RES_ED_WCA2",false);
p_Player.EnableResearch("RES_ED_WSL1",false);
//p_Player.EnableResearch("RES_ED_WSR1",false);
p_Player.EnableResearch("RES_ED_WSI1",false);
p_Player.EnableResearch("RES_ED_WMR1",false);
p_Player.EnableResearch("RES_ED_WHC1",false);
p_Player.EnableResearch("RES_ED_WH1",false);
p_Player.EnableResearch("RES_ED_WH1L",false);
p_Player.EnableResearch("RES_ED_WHR1",false);
p_Player.EnableResearch("RES_ED_UHT1",false);
//ammo
p_Player.EnableResearch("RES_MCH2",false);
p_Player.EnableResearch("RES_ED_MSC2",false);
p_Player.EnableResearch("RES_ED_MH2",false);
p_Player.EnableResearch("RES_MSR2",false);
p_Player.EnableResearch("RES_MM2",false);
p_Player.EnableResearch("RES_ED_MHR2",false);
p_Player.EnableResearch("RES_ED_MHR4",false);

//chassis
//p_Player.EnableResearch("RES_ED_UMT1",false);
p_Player.EnableResearch("RES_ED_JMW1",false);
//p_Player.EnableResearch("RES_ED_UM1",false);
p_Player.EnableResearch("RES_ED_UHT1",false);
p_Player.EnableResearch("RES_ED_UBT1",false);
p_Player.EnableResearch("RES_ED_UJ1",false);
p_Player.EnableResearch("RES_ED_UA12",false);
p_Player.EnableResearch("RES_ED_UA21",false);
p_Player.EnableResearch("RES_ED_UA31",false);
p_Player.EnableResearch("RES_ED_UA41",false);
p_Player.EnableResearch("RES_ED_UJS2",false);
p_Player.EnableResearch("RES_ED_UHS1",false);
//special
//p_Player.EnableResearch("RES_ED_RephHand",false);
p_Player.EnableResearch("RES_ED_RephHand2",false);
p_Player.EnableResearch("RES_ED_SCN",false);
p_Player.EnableResearch("RES_ED_SGen",false);
p_Player.EnableResearch("RES_ED_BMD",false);
p_Player.EnableResearch("RES_ED_BHD",false);

p_Player.EnableBuilding("EDBHQ",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBTC",false);
return ShowBriefing,100;
}

state ShowBriefing
{
    nReport = 0;
    AddBriefing("translateStartCampaign",fleetCost);

Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,2500,800
,8);
    return ShowStartFirstMissionBriefing,140;
}

state ShowStartFirstMissionBriefing
{
    AddBriefing("translateStartFirstMission");
    return Nothing,50;
}

state Nothing
{
    int nSendedMoney;
    nSendedMoney = p_Player.GetMoneySentToOrbit();
}

RegisterGoal(0,"translateGoalSendMoneyToSpace",fleetCost,(nSendedMoney*100
)/fleetCost);

if((p_Player.GetNumberOfBuildings(buildingSpaceStation))
{
    {
        AddBriefing("translateCampaignEDSpacePortDestroyed",p_Player.GetName());
        return EndDemo,100;
    }

    if(p_Player.GetScriptData(1)==1)
    {
        AddBriefing("translateCampaignAccomplishedDemo");
        return EndDemo,100;
    }
    return Nothing,100;
}

state EndDemo
{
    EndGame(null);
//EndMission(true);
}

event Timer0() //wolany co 200 000 cykli< ustawione funkcja SetTimer w state
Initial
{
    int nSendedMoney;
    int nProjectStatus;
    int nMoneyForNextSeason;

    nReport = nReport+1;

    if(nReport>noOfSeasons)
    {
        AddBriefing("translateCampaignFailed");
        EnableNextMission(0,5);
        return;
    }

    nSendedMoney = p_Player.GetMoneySentToOrbit();// to ma tu byc
    nProjectStatus = (nSendedMoney*100)/fleetCost;
    nMoneyForNextSeason = ((nReport+1)*oneSeasonMoney) -
nSendedMoney;
    if(nMoneyForNextSeason<0) nMoneyForNextSeason=0;

    if(nSendedMoney < oneSeasonMoney*nReport)

AddBriefing("translateDelayReport",nReport,nProjectStatus,oneSeasonMoney*nRe
port,nSendedMoney,nMoneyForNextSeason);
    else

AddBriefing("translateReport",nReport,nProjectStatus,oneSeasonMoney*nReport,
nSendedMoney,nMoneyForNextSeason);
    }

    event Timer1() //wolany co 5000 cykli
    {

Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,2500,800
,10);
    }
}
}

mission "translateMissionEDBaseLC"
{
player p_PlayerLC;

state initialize;
state Nothing;
state Initialize
{
    p_PlayerLC=GetPlayer(3);
    p_PlayerLC.SetMoney(10000);

    p_PlayerLC.EnableAIFeatures(aiBuildTanks,false);
    p_PlayerLC.EnableAIFeatures(aiBuildShips,false);
    p_PlayerLC.EnableAIFeatures(aiBuildHelicopters,false);

    p_PlayerLC.EnableAIFeatures(aiBuildSpecialUnits,false);

//weapons
p_PlayerLC.EnableResearch("RES_LC_WCH2",false);
p_PlayerLC.EnableResearch("RES_LC_WSR2",false);
}

```

```

p_PlayerLC.EnableResearch("RES_LC_WMR1",false);
p_PlayerLC.EnableResearch("RES_LC_WSL1",false);
p_PlayerLC.EnableResearch("RES_LC_WHL1",false);
p_PlayerLC.EnableResearch("RES_LC_WSS1",false);
p_PlayerLC.EnableResearch("RES_LC_WH1",false);
p_PlayerLC.EnableResearch("RES_LC_WARTILLERY",false);
//CHASSIS
p_PlayerLC.EnableResearch("RES_LC_LMO2",false);
p_PlayerLC.EnableResearch("RES_LC_UCR1",false);
p_PlayerLC.EnableResearch("RES_LC_UCU1",false);
p_PlayerLC.EnableResearch("RES_LC_UME1",false);
p_PlayerLC.EnableResearch("RES_LC_UBO1",false);
//SPECIAL
p_PlayerLC.EnableResearch("RES_LC_BMD",false);
p_PlayerLC.EnableResearch("RES_LC_BHD",false);
p_PlayerLC.EnableResearch("RES_LC_BWC",false);
p_PlayerLC.EnableResearch("RES_LC_SGen",false);
p_PlayerLC.EnableResearch("RES_LC_SHR1",false);
p_PlayerLC.EnableResearch("RES_LC_REG1",false);
p_PlayerLC.EnableResearch("RES_LC_SO1",false);
//AMMO
p_PlayerLC.EnableResearch("RES_MCH2",false);
p_PlayerLC.EnableResearch("RES_MSR2",false);
p_PlayerLC.EnableResearch("RES_MMR2",false);
return Nothing;
}
state Nothing
{
}
}

```

baseUCSscript.ec

mission "translateMissionEDBaseUCS"

```

{
player p_PlayerUCS;

state Initialize;
state Nothing;
state Initialize
{
    p_PlayerUCS = GetPlayer(1);
    p_PlayerUCS.SetMoney(10000);

    p_PlayerUCS.EnableAIFeatures(aiBuildTanks,false);
    p_PlayerUCS.EnableAIFeatures(aiBuildShips,false);
    p_PlayerUCS.EnableAIFeatures(aiBuildHelicopters,false);
    p_PlayerUCS.EnableAIFeatures(aiBuildSpecialUnits,false);

    //weapons
    p_PlayerUCS.EnableResearch("RES_UCS_WCH2",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WSR1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WSG1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WHG1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WMR1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WSP1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WH1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WAPB1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_WSD",false);
//CHASSIS
    p_PlayerUCS.EnableResearch("RES_UCS_UML1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_UHL1",false);
    p_PlayerUCS.EnableResearch("RES_UCSUBL1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_UML1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_USB1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_USM1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_UOH2",false);
    p_PlayerUCS.EnableResearch("RES_UCS_UAH1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_GARG1",false);
    p_PlayerUCS.EnableResearch("RES_UCS_BOMBER21",false);
    p_PlayerUCS.EnableResearch("RES_UCS_BOMBER31",false);
//AMMO
    p_PlayerUCS.EnableResearch("RES_MCH2",false);
    p_PlayerUCS.EnableResearch("RES_MSR2",false);
    p_PlayerUCS.EnableResearch("RES_MMR2",false);
    p_PlayerUCS.EnableResearch("RES_UCS_MB2",false);
    p_PlayerUCS.EnableResearch("RES_UCS_MG2",false);
//SPECIAL
    p_PlayerUCS.EnableResearch("RES_UCS_BMD",false);
    p_PlayerUCS.EnableResearch("RES_UCS_BHD",false);
    p_PlayerUCS.EnableResearch("RES_UCS_RePnd",false);
    p_PlayerUCS.EnableResearch("RES_UCS_SGen",false);
    p_PlayerUCS.EnableResearch("RES_UCS_SHD",false);

```

```

        return Nothing;
    }
    state Nothing
    {
    }
}

```

Mission211.ec

mission "translateMission211"

```

{
consts
{
    findResource = 0;
    sendToBase20000 = 1;
    destroyEnemyUnits = 2;
    destroyEnemyBase = 3;
    buildLandingZone = 4;
    callContTran = 5;
}

```

```

player p_Enemy;
player p_Player;

```

```

int n_ResourceX;
int n_ResourceY;
int n_BaseCX;
int n_BaseCY;
int n_BaseCX2;
int n_BaseCY2;
int bShowFailed;
int bCheckEndMission;
int bVictory;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(findResource,"translateGoalFindResources");
    RegisterGoal(sendToBase20000,"translateGoalSend20000",0);
    RegisterGoal(destroyEnemyUnits,"translateGoalDestroyEnemyUnits");
    RegisterGoal(destroyEnemyBase,"translateGoalDestroyEnemyBase");
    RegisterGoal(buildLandingZone,"translateGoal211a");
    RegisterGoal(callContTran,"translateGoal211b");
    EnableGoal(findResource,true);
    EnableGoal(buildLandingZone,true);
    EnableGoal(callContTran,true);

```

```

    n_ResourceX = GetPointX(0);
    n_ResourceY = GetPointY(0);
    n_BaseCX = GetPointX(1);
    n_BaseCY = GetPointY(1);

```

```

    n_BaseCX2 = GetPointX(6);
    n_BaseCY2 = GetPointY(6);

```

```

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(3);

```

```

    p_Enemy.EnableStatistics(false);

```

```

    p_Player.SetMoney(10000);
    p_Enemy.SetMoney(20000);

```

```

    if(GetDifficultyLevel()==0)
        p_Enemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        p_Enemy.LoadScript("single\singleHard");

```

```

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy.EnableAIFeatures(aiEnabled,false);

```

```

    p_Player.EnableResearch("RES_ED_UST2",true);

```

// 1st tab

```

    p_Player.EnableBuilding("EDBPP",true);

```

```

    p_Player.EnableBuilding("EDBBA",false);

```

```

p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBM",true);
p_Player.EnableBuilding("EDBTC",true);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHO",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBEN1",false);
p_Player.EnableBuilding("EDBLZ",true);

CreateArtefact("NEASPECIAL1",n_BaseLCX,n_BaseLCY,0,0,artefactSpecialAINewAreaLocation);

bShowFailed=true;
bCheckEndMission=false;
bVictory=false;
EnableNextMission(0,true);

SetTimer(0,100);
SetTimer(1,6000);

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);

return ShowBriefing,150;//15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing211a");
    Snow(n_ResourceX,n_ResourceY,45,400,5000,800,10);
    return Mining,100;
}
//-----
state Mining
{
    if(GetGoalState(findResource)!= goalAchieved &&
    p_Player.IsPointLocated(n_ResourceX,n_ResourceY,0))
    {
        SetGoalState(findResource,goalAchieved);
        EnableGoal(sendToBase20000,true);
    }
    if(GetGoalState(sendToBase20000)==goalAchieved && p_Player.GetMoneySentToBase(>=20000)
    {
        SetGoalState(sendToBase20000,goalAchieved);
        if(!IsGoalEnabled(destroyEnemyUnits) &&
        !IsGoalEnabled(destroyEnemyBase))
        {
            AddBriefing("translateAccomplished211a");
            bVictory=true;
            EnableEndMissionButton(true);
        }
        //EnableSendingToBase(false);//XXXMD to wykorzystywac.
    }
    if(!IsGoalEnabled(destroyEnemyUnits) &&
    (p_Player.IsPointLocated(GetPointX(2),GetPointY(2),0) ||
    p_Player.IsPointLocated(GetPointX(3),GetPointY(3),0) ||
    p_Player.IsPointLocated(GetPointX(4),GetPointY(4),0) ||
    p_Player.IsPointLocated(GetPointX(5),GetPointY(5),0)))
    {
        p_Enemy.EnableStatistics(true);
        EnableGoal(destroyEnemyUnits,true);
        AddBriefing("translateBriefing211b");
        p_Enemy.EnableAIFeatures(aiEnabled,true);
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        p_Enemy.EnableAIFeatures(aiBuildMiningBuildings,false);
        bVictory=false;
        EnableEndMissionButton(false);
        // 1st tab
        p_Player.EnableBuilding("EDBBA",true);
        p_Player.EnableBuilding("EDBFA",true);
        p_Player.EnableBuilding("EDBAB",true);
        // 2nd tab
        p_Player.EnableBuilding("EDBRE",true);
    }
    if(!IsGoalEnabled(destroyEnemyBase) &&

```

(p_Player.IsPointLocated(n_BaseLCX,n_BaseLCY,0) ||
p_Player.IsPointLocated(n_BaseLCX2,n_BaseLCY2,0)) //baza wroga
{
 EnableGoal(destroyEnemyBase,true);
 Snow(n_BaseLCX,n_BaseLCY,30,400,5000,800,10);
 p_Enemy.EnableAIFeatures(aiControlOffense,true);
 AddBriefing("translateBriefing211c");
}
}

if(GetGoalState(destroyEnemyUnits)!=goalAchieved &&
p_Enemy.GetNumberOfUnits(<)7)
{
 SetGoalState(destroyEnemyUnits,goalAchieved);
}
if(GetGoalState(destroyEnemyBase)!=goalAchieved &&
lp_Enemy.GetNumberOfBuildings())
{
 p_Enemy.EnableAIFeatures(aiControlOffense,false);
 p_Enemy.EnableAIFeatures(aiControlDefense,false);
}
p_Enemy.RussianAttack(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
0);
 SetGoalState(destroyEnemyBase,goalAchieved);
}
if(GetGoalState(destroyEnemyUnits)==goalAchieved &&
GetGoalState(destroyEnemyBase)==goalAchieved &&
GetGoalState(sendToBase20000)==goalAchieved)
{
 AddBriefing("translateAccomplished211b");
 bVictory=true;
 EnableEndMissionButton(true);
 return Nothing;
}
return Mining,200;
}
//-----
state Nothing
{
 return Nothing, 500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialise
{
 RegisterGoal(sendToBase20000,"translateGoalSend20000",p_Player.GetMoneySe
ntToBase());
 if(GetGoalState(buildLandingZone)!=goalAchieved &&
 p_Player.GetNumberOfBuildings(buildingTransportCenter))
 {
 SetGoalState(buildLandingZone,goalAchieved);
 }
 if(GetGoalState(chassisTank|unitCarrier)>=2)
 {
 SetGoalState(callContTran,goalAchieved);
 }
 if(bShowFailed)
 {
 if((ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase
())<20000)
 {
 bShowFailed=false;
 SetGoalState(sendToBase20000,goalFailed);
 AddBriefing("translateFailed211a");
 EnableEndMissionButton(true,false);
 bVictory=false;
 return Nothing;
 }
 }
 if(bCheckEndMission)
 {
 bCheckEndMission=false;
 if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
 {
 if(bVictory)
 EndMission(true);
 else
 AddBriefing("translateFailed211b");
 EndMission(false);
 }
 }
}

```

        }
    }
}

//-----
event Timer1() //wolany co 6000 cykli 5min
{
    Snow(n_ResourceX,n_ResourceY,45,400,2500,800,5);
    Snow(n_BaseLCX,n_BaseLCY,20,400,2500,800,5);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event Artefact(int allD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    EnableNextMission(1,true);
    return true; //usuwa sie
}
}
}

```

Mission211Demo.ec

```

mission "translateMission211"
{
    consts
    {
        findResource = 0;
        sendToBase20000 = 1;
        destroyEnemyUnits = 2;
        destroyEnemyBase = 3;
        buildLandingZone = 4;
        callContTran = 5;
    }

    player p_Enemy;
    player p_Player;

    int n_ResourceX;
    int n_ResourceY;
    int n_BaseLCX;
    int n_BaseLCY;
    int n_BaseLCX2;
    int n_BaseLCY2;
    int bShowFailed;
    int bCheckEndMission;
    int bVictory;
    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;
}
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(findResource,"translateGoalFindResources");
    RegisterGoal(sendToBase20000,"translateGoalSend20000",0);
    RegisterGoal(destroyEnemyUnits,"translateGoalDestroyEnemyUnits");
    RegisterGoal(destroyEnemyBase,"translateGoalDestroyEnemyBase");
    RegisterGoal(buildLandingZone,"translateGoal211a");
    RegisterGoal(callContTran,"translateGoal211b");
    EnableGoal(findResource,true);
    EnableGoal(buildLandingZone,true);
    EnableGoal(callContTran,true);

    n_ResourceX = GetPointX(0);
    n_ResourceY = GetPointY(0);
    n_BaseLCX = GetPointX(1);
    n_BaseLCY = GetPointY(1);

    n_BaseLCX2 = GetPointX(6);
    n_BaseLCY2 = GetPointY(6);

    p_Player = GetPlayer(2);
}

p_Enemy = GetPlayer(3);
p_Enemy.EnableStatistics(false);
p_Player.SetMoney(10000);
p_Enemy.SetMoney(20000);

if(GetDifficultyLevel()==0)
    p_Enemy.LoadScript("single\singleEasy");
if(GetDifficultyLevel()==1)
    p_Enemy.LoadScript("single\singleMedium");
if(GetDifficultyLevel()==2)
    p_Enemy.LoadScript("single\singleHard");

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy.EnableAIFeatures(aiEnabled,false);

p_Player.EnableResearch("RES_ED_UT2",true);

// 1st tab
p_Player.EnableBuilding("EDBPP",true);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBM",true);
p_Player.EnableBuilding("EDBTC",true);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHO",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBN1",false);
p_Player.EnableBuilding("EDBLZ",true);

CreateArtefact("NEASPECIAL1",n_BaseLCX,n_BaseLCY,0,0,artefactSpecialAINewAreaLocation);

bShowFailed=true;
bCheckEndMission=false;
bVictory=false;
EnableNextMission(0,true);

SetTimer(0,100);
SetTimer(1,6000);

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);

return ShowBriefing,150://15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing211a");
    Snow(n_ResourceX,n_ResourceY,45,400,5000,800,10);
    return Mining,100;
}
//-----
state Mining
{
    if(GetGoalState(findResource)==goalAchieved &&
       p_Player.IsPointLocated(n_ResourceX,n_ResourceY,0))
    {
        SetGoalState(findResource,goalAchieved);
        EnableGoal(sendToBase20000,true);
    }
    if(GetGoalState(sendToBase20000)==goalAchieved
       && p_Player.GetMoneySentToBase()>=20000)
    {
        SetGoalState(sendToBase20000,goalAchieved);
        if(!IsGoalEnabled(destroyEnemyUnits) &&
           !IsGoalEnabled(destroyEnemyBase))
        {
            AddBriefing("translateAccomplished211a");
            EnableEndMissionButton(true);
            bVictory=true;
            p_Player.SetScriptData(1,1);
        }
    }
}

```

```

        } //EnableSendingToBase(false); //XXXMD to wykorzystywac.
    }
    if(!IsGoalEnabled(destroyEnemyUnits) &&
    (p_Player.IsPointLocated(GetPointX(2),GetPointY(2),0) ||
    p_Player.IsPointLocated(GetPointX(3),GetPointY(3),0) ||
    p_Player.IsPointLocated(GetPointX(4),GetPointY(4),0) ||
    p_Player.IsPointLocated(GetPointX(5),GetPointY(5),0)))
    {
        p_Enemy.EnableStatistics(true);
        EnableGoal(destroyEnemyUnits,true);
        AddBriefing("translateBriefing211b");
        p_Enemy.EnableAIFeatures(aiEnabled,true);
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        p_Enemy.EnableAIFeatures(aiBuildMiningBuildings,false);
        EnableEndMissionButton(false);
        bVictory=false;
        // 1st tab
        p_Player.EnableBuilding("EDBBA",true);
        p_Player.EnableBuilding("EDBFA",true);
        p_Player.EnableBuilding("EDBAB",true);
        // 2nd tab
        p_Player.EnableBuilding("EDBRE",true);
    }
    if(!IsGoalEnabled(destroyEnemyBase) &&
    (p_Player.IsPointLocated(n_BaseLCX,n_BaseLCY,0) ||
    p_Player.IsPointLocated(n_BaseLC2,n_BaseLCY2,0))) //baza wroga
    {
        EnableGoal(destroyEnemyBase,true);
        Snow(n_BaseLCX,n_BaseLCY,30,400,5000,800,10);
        p_Enemy.EnableAIFeatures(aiControlOffense,true);
        AddBriefing("translateBriefing211c");
    }

    if(GetGoalState(destroyEnemyUnits)!=goalAchieved &&
    p_Enemy.GetNumberOfUnits(<7)
    {
        SetGoalState(destroyEnemyUnits,goalAchieved);
    }

    if(GetGoalState(destroyEnemyBase)!=goalAchieved &&
    lp_Enemy.GetNumberOfBuildings())
    {
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        p_Enemy.EnableAIFeatures(aiControlDefense,false);
        p_Enemy.RussianAttack(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),0);
        SetGoalState(destroyEnemyBase,goalAchieved);
    }

    if(GetGoalState(destroyEnemyUnits)==goalAchieved &&
    GetGoalState(destroyEnemyBase)==goalAchieved &&
    GetGoalState(sendToBase20000)==goalAchieved)
    {
        AddBriefing("translateAccomplished211b");
        EnableEndMissionButton(true);
        bVictory=true;
        p_Player.SetScriptData(1,1);
        return Nothing;
    }

    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase20000,"translateGoalSend20000",p_Player.GetMoneySentToBase());
if(GetGoalState(buildLandingZone)!=goalAchieved &&
p_Player.GetNumberOfBuildings(buildingTransportCenter))
{
    SetGoalState(buildLandingZone,goalAchieved);
}
if(GetGoalState(callContTran)!=goalAchieved &&
p_Player.GetNumberOfUnits(chassisTank | unitCarrier)>=2)
{
    SetGoalState(callContTran,goalAchieved);
}

} //bShowFailed
{
if((ResourcesLeftInMoney() + p_Player.GetMoney() + p_Player.GetMoneySentToBase
() )<20000)
{
    bShowFailed=false;
    SetGoalState(sendToBase20000,goalFailed);
    AddBriefing("translateFailed211a");
    EnableEndMissionButton(true,false);
    p_Player.SetScriptData(1,1);
    bVictory=false;
    return Nothing;
}
}
if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        if(bVictory)
            EndMission(true);
        else
        {
            AddBriefing("translateFailed211b");
            EndMission(false);
        }
        p_Player.SetScriptData(1,1);
    }
}
//-----
event Timer1() //wolany co 6000 cykli 5min
{
    Snow(n_ResourceX,n_ResourceY,45,400,2500,800,5);
    Snow(n_BaseLCX,n_BaseLCY,20,400,2500,800,5);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event Artefact(int aID,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    EnableNextMission(1,true);
    return true; //usuwa sie
}
}

Mission212.ec
mission "translateMission212"
{/Antarktyka klif
consts
{
    findEnemy = 0;
    findEnemyBase = 1;
    destroyEnemyBase = 2;
}
player p_Enemy;
player p_Neutral;
player p_Player;
int n_Enemy1X;
int n_Enemy1Y;
int n_Enemy2X;
int n_Enemy2Y;
int n_EnemyBaseX;
int n_EnemyBaseY;
int n_OldBaseX;
int n_OldBaseY;
int n_OldBaseZ;
int n_CheckEndMission;
state Initialize;
state ShowBriefing;
state Nothing;
//-----
state Initialize
{
}

```

```

player tmpPlayer;
if(!PointExist(0))AddBriefing("translateMarkPointDsnExist",0);
if(!PointExist(1))AddBriefing("translateMarkPointDsnExist",1);
if(!PointExist(2))AddBriefing("translateMarkPointDsnExist",2);
if(!PointExist(3))AddBriefing("translateMarkPointDsnExist",3);
//----- Goals -----
RegisterGoal(findEnemy,"translateGoal212a");
RegisterGoal(findEnemyBase,"translateGoalFindEnemyBase");
RegisterGoal(destroyEnemyBase,"translateGoalDestroyEnemyBase");
EnableGoal(findEnemy,true);
//----- Temporary players -----
tmpPlayer = GetPlayer(3);
tmpPlayer.EnableStatistics(false);
tmpPlayer = GetPlayer(6);
tmpPlayer.EnableStatistics(false);
tmpPlayer.EnableAIFeatures(aiEnabled,false);
//----- Players -----
p_Player = GetPlayer(2);
p_Enemy = GetPlayer(1);
p_Neutral = GetPlayer(4);

//----- AI -----
p_Neutral.EnableStatistics(false);

p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);
p_Enemy.SetNeutral(p_Neutral);
if(GetDifficultyLevel()==0)
{
    p_Enemy.LoadSceneScript("single\singleEasy");
    p_Enemy.EnableAIFeatures(aiUpgradeCannons,false);
}
if(GetDifficultyLevel()==1)
    p_Enemy.LoadSceneScript("single\singleMedium");
if(GetDifficultyLevel()==2)
{
    p_Enemy.LoadSceneScript("single\singleHard");

p_Enemy.CreateDefaultUnit(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),0);
}
p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy.EnableAIFeatures(aiControlOffense,false);

//----- Money -----
p_Player.SetMoney(15000);
p_Enemy.SetMoney(10000);
//----- Researches -----
p_Enemy.EnableResearch("RES_UCS_WSR1",true);

p_Player.EnableResearch("RES_ED_WCH2",true);
p_Player.EnableResearch("RES_MCH2",true);
p_Player.EnableResearch("RES_ED_LA11",true);
p_Player.EnableResearch("RES_ED_UT3",true);
//----- Buildings -----
// 1st tab
p_Player.EnableBuilding("EDBPP",true);
p_Player.EnableBuilding("EDDBA",true);
p_Player.EnableBuilding("EDBFA",true);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",true);
// 2nd tab
p_Player.EnableBuilding("EDBRE",true);
p_Player.EnableBuilding("EDBM",true);
p_Player.EnableBuilding("EDBTC",true);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab
p_Player.EnableBuilding("EDBRC",true);
p_Player.EnableBuilding("EDBHQ",true);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBEN1",true);
p_Player.EnableBuilding("EDBLZ",true);
//----- Units -----
//----- Artefacts -----
CreateArtefact("NEASPECIAL1",GetPointX(3),GetPointY(3),0,0,artefactSpecialAInewAreaLocation);
//----- Timers -----
SetTimer(0,100);
SetTimer(1,1000);
//----- Variables -----
n_Enemy1X = GetPointX(0);
n_Enemy1Y = GetPointY(0);
n_Enemy2X = GetPointX(1);
n_Enemy2Y = GetPointY(1);
n_EnemyBaseX = GetPointX(2);
n_EnemyBaseY = GetPointY(2);
n_OldBaseX = p_Neutral.GetStartingPointX();
n_OldBaseY = p_Neutral.GetStartingPointY();
bCheckEndMission=false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,15,0);
return ShowBriefing,280;
}
//-----
state ShowBriefing
{
Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),20,50,5000,30,7);
EnableNextMission(0,true);
AddBriefing("translateBriefing212a");
return Nothing,100;
}
//-----
state Nothing
{
if(GetGoalState(findEnemy) != goalAchieved)
{
    if(p_Player.IsPointLocated(n_Enemy1X,n_Enemy1Y,0))
    {
        SetGoalState(findEnemy,goalAchieved);
        EnableGoal(findEnemyBase,true);
        CallCamera();
        AddBriefing("translateBriefing212b");
    }
    if(p_Player.IsPointLocated(n_Enemy2X,n_Enemy2Y,0))
    {
        SetGoalState(findEnemy,goalAchieved);
        EnableGoal(findEnemyBase,true);
        CallCamera();
        AddBriefing("translateBriefing212b");
    }
}
if(GetGoalState(findEnemyBase)!=goalAchieved && p_Player.IsPointLocated(n_EnemyBaseX,n_EnemyBaseY,0))
{
    Snow(n_EnemyBaseX,n_EnemyBaseY,20,50,8000,50,7);
    SetGoalState(findEnemyBase,goalAchieved);
    EnableGoal(destroyEnemyBase,true);
    CallCamera();
    AddBriefing("translateBriefing212c");
}
if(n_OldBaseX && p_Player.IsPointLocated(n_OldBaseX,n_OldBaseY,0))
{
    n_OldBaseX=0;
    AddBriefing("translateBriefing212d");
}
return Nothing,100;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed212");
            EndMission(false);
        }
        if(GetGoalState(destroyEnemyBase)!=goalAchieved && (p_Enemy.GetNumberOfUnits()<5) && (p_Enemy.GetNumberOfBuildings(buildingPowerPlant)&& p_Enemy.GetNumberOfBuildings(buildingRefinery))&& p_Enemy.GetNumberOfBuildings() < 5)
        {
            SetGoalState(destroyEnemyBase,goalAchieved);
            AddBriefing("translateAccomplished212");
            EnableEndMissionButton(true);
        }
    }
}
//-----
```

```

event Timer1() //wolany co 1000 cykli
{
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Enemy.SetEnemy(p_Neutral);
}
//-----
event Artefact(int alD,player pPlayer)
{
    if(piPlayer!=p_Player) return false;
    EnableNextMission(1,true);
    return true; //usuwa sie
}
}

```

Mission213.ec

```

mission "translateMission213"
{/Himalaya baza LC
consts
{
    findLostPlatoon = 0;
    findEnemy = 1;
    destroyEnemyMines = 2;
    sendToBase10000 = 3;
}

player pEnemy;
player pMines;
player pPlayer;

int n_PlatoonX;
int n_PlatoonY;
int n_EnemyPosX;
int n_EnemyPosY;
int bCheckEndMission;
int bAIEnabled;

state Initialize;
state ShowBriefing;
state FindLostPlatoon;
state FindEnemy;
state DestroyEnemy;
state Evacuate;
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    if((!PointExist(0))AddBriefing("translateMarkPointDoesNotExist",0);
    if((!PointExist(1))AddBriefing("translateMarkPointDoesNotExist",1);
    if((!PointExist(2))AddBriefing("translateMarkPointDoesNotExist",2);
    if((!PointExist(3))AddBriefing("translateMarkPointDoesNotExist",3);

    RegisterGoal(findLostPlatoon,"translateGoal213a");
    RegisterGoal(findEnemy,"translateGoal213b");
    RegisterGoal(destroyEnemyMines,"translateGoal213c");
    RegisterGoal(sendToBase10000,"translateGoalSend10000",0);
    EnableGoal(findLostPlatoon,true);

    pPlayer = GetPlayer(2);
    pEnemy = GetPlayer(3);
    pMines = GetPlayer(5);
    pPlayer.SetMoney(10000);

    if(GetDifficultyLevel()==0)
    {
        pEnemy.LoadScript("single\singleEasy");

```

```

        pEnemy.EnableAIFeatures(aiUpgradeCannons,false);
        pEnemy.SetMoney(20000);
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy.LoadScript("single\singleMedium");
        pEnemy.SetMoney(30000);
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy.LoadScript("single\singleHard");
        pEnemy.SetMoney(100000);
    }
}

pMines.LoadScript("single\singleEasy");
pMines.EnableAIFeatures(aiEnabled,false);

pEnemy.EnableAIFeatures(aiDefenseTowers,false);
pEnemy.EnableAIFeatures(aiControlOffense,false);
pEnemy.EnableAIFeatures(aiBuildBuildings,false);

pPlayer.EnableAIFeatures(aiEnabled,false);

n_PlatoonX = GetPointX(2);
n_PlatoonY = GetPointY(0);
n_EnemyPosX = GetPointX(1);
n_EnemyPosY = GetPointY(1);

pEnemy.SetPointToAssemble(0,GetPointX(2),GetPointY(2),0);
pEnemy.SetPointToAssemble(1,GetPointX(3),GetPointY(3),0);

pEnemy.EnableResearch("RES_LC_BMD",false);
pEnemy.EnableResearch("RES_LC_WCH2",true);
pEnemy.EnableResearch("RES_MCH2",true);
pEnemy.EnableResearch("RES_LC_UAE1",true);

pPlayer.EnableResearch("RES_MCH2",true);
pPlayer.EnableResearch("RES_ED_UST3",true);
pPlayer.EnableResearch("RES_ED_UA11",true);
pPlayer.EnableResearch("RES_ED_UA12",true);

// 1st tab
pPlayer.EnableBuilding("EDBPP",false);
pPlayer.EnableBuilding("EDBBA",false);
pPlayer.EnableBuilding("EDBFA",false);
pPlayer.EnableBuilding("EDBWB",false);
pPlayer.EnableBuilding("EDBAA",false);
// 2nd tab
pPlayer.EnableBuilding("EDBRC",false);
pPlayer.EnableBuilding("EDBMF",false);
pPlayer.EnableBuilding("EDBTG",false);
// 3rd tab
pPlayer.EnableBuilding("EDBTS",false);
// 4th tab
pPlayer.EnableBuilding("EDBRC",false);
pPlayer.EnableBuilding("EDBHQ",false);
pPlayer.EnableBuilding("EDBRA",false);
pPlayer.EnableBuilding("EDBEN1",false);
pPlayer.EnableBuilding("EDBLZ",false);

bAIEnabled=false;

SetTimer(0,100);
SetTimer(1,6000);
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing213a");
    return FindLostPlatoon,100;
}

state FindLostPlatoon
{
    if(pPlayer.IsPointLocated(n_PlatoonX,n_PlatoonY,0))
    {
        // 1st tab
        pPlayer.EnableBuilding("EDBPP",true);
        pPlayer.EnableBuilding("EDBBA",true);
        pPlayer.EnableBuilding("EDBFA",true);
    }
}

```

```

pPlayer.EnableBuilding("EDWB",false);
pPlayer.EnableBuilding("EDBDB",true);
// 2nd tab
pPlayer.EnableBuilding("EDBRB",true);
pPlayer.EnableBuilding("EDBMB",true);
pPlayer.EnableBuilding("EDBCC",true);
// 3rd tab
pPlayer.EnableBuilding("EDBTS",false);
// 4th tab
pPlayer.EnableBuilding("EDBRC",true);
pPlayer.EnableBuilding("EDBHQ",true);
pPlayer.EnableBuilding("EDBRA",false);
pPlayer.EnableBuilding("EDBEN",true);
pPlayer.EnableBuilding("EDBLZ",true);

SetGoalState(findLostPlatoon, goalAchieved);
EnableGoal(findEnemy,true);
CallCamera();
pPlayer.LookAt(n_PlatoonX,n_PlatoonY,10,0,20,0);
AddBriefing("translateBriefing213b");
return FindEnemy,200;
}

return FindLostPlatoon,100;
}
//-----
state FindEnemy
{
if(pPlayer.IsPointLocated(n_EnemyPosX,n_EnemyPosY,0))
{
    EnableGoal(destroyEnemyMines,true);
    EnableGoal(sendToBase10000,true);
    SetGoalState(findEnemy,goalAchieved);
    AddBriefing("translateBriefing213c");
    pEnemy.SetMoney(pEnemy.GetMoney() + 30000);
    return DestroyEnemy,200;
}
return FindEnemy,100;
}
//-----
state DestroyEnemy
{
if((ResourcesLeftInMoney() + pPlayer.GetMoney() + pPlayer.GetMoneySentToBase()) < 10000)
{
    SetGoalState(sendToBase10000, goalFailed);
    AddBriefing("translateFailed213a");
    EnableEndMissionButton(true);
    return Evacuate;
}
if(GetGoalState(destroyEnemyMines) == goalAchieved & lpMines.GetNumberOfBuildings(buildingMine))
{
    SetGoalState(destroyEnemyMines, goalAchieved);
}
if((bAiEnabled && pPlayer.GetMoneySentToBase() > 1000)
{
    bAiEnabled = true;
    pEnemy.EnableAIFeatures(aiControlOffense,true);
}
if(GetGoalState(sendToBase10000) == goalAchieved & pPlayer.GetMoneySentToBase() >= 10000)
{
    SetGoalState(sendToBase10000, goalAchieved);
}
if((GetGoalState(destroyEnemyMines) == goalAchieved & & GetGoalState(sendToBase10000) == goalAchieved)
{
    AddBriefing("translateAccomplished213");
    EnableEndMissionButton(true);
    return Evacuate;
}
return DestroyEnemy,100;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    RegisterGoal(sendToBase10000,"translateGoalSend10000",pPlayer.GetMoneySentToBase());
    if(bCheckEndMission)
    {
        bCheckEndMission = false;
        if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed213b");
            EndMission(false);
        }
    }
}
//-----
event Timer1() //wolany co 6000 cykli = 5 min
{
    Snow(n_EnemyPosX,n_EnemyPosY,30,400,2500,800,8);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}
}

Mission214.ec
mission "translateMission214"
{/Antarktyka centrum - stara baza Rosyjska
consts
{
    findBase = 0;
    findHangar = 1;
    findCommandCenter = 2;
    sendToBase50000 = 3;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Neutral;
player p_Player;

int bShowFailed;
int bSwitchOn;
int bCheckEndMission;
int n_HangarX;
int n_HangarY;
int n_CenterX;
int n_CenterY;

//-----
state Initialize;
state ShowBriefing;
state LocateUnits;
state ExploringBase;
state Mining;
state Evacuate;

state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(findBase,"translateGoal214a");
    RegisterGoal(findHangar,"translateGoal214b");
    RegisterGoal(findCommandCenter,"translateGoal214c");
    RegisterGoal(sendToBase50000,"translateGoalSend50000");

    EnableGoal(findBase,true);
    EnableGoal(findHangar,true);
}

n_HangarX = GetPointX(0);
n_HangarY = GetPointY(0);
n_CenterX = GetPointX(1);
n_CenterY = GetPointY(1);

p_Player = GetPlayer(2);
}

```

```

p_Neutral= GetPlayer(4);
p_Enemy1 = GetPlayer(1);
p_Enemy2 = GetPlayer(5);
p_Enemy3 = GetPlayer(6);

p_Player.SetMoney(10000);
p_Enemy1.SetMoney(25000);
p_Enemy2.SetMoney(25000);
p_Enemy3.SetMoney(25000);

if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
    p_Enemy3.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    p_Enemy3.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
    p_Enemy3.LoadScript("single\singleHard");
}

p_Enemy1.SetAlly(p_Enemy2);
p_Enemy2.SetAlly(p_Enemy3);
p_Enemy3.SetAlly(p_Enemy1);

p_Player.SetNeutral(p_Neutral);
p_Neutral.SetNeutral(p_Player);

p_Neutral.EnableStatistics(false);

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);

p_Player.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableAIFeatures(aiEnabled,false);

bShowFailed=false;
bSwitchOnA=false;
bCheckEndMission=false;

SetTimer(0,100);
SetTimer(1,6000);

CreateArtefact("NEASPECIAL2",n_CenterX,n_CenterY,1,0,artefactSpecialAIOther);

p_Enemy1.EnableResearch("RES_UCS_WCH2",true);
p_Enemy1.EnableResearch("RES_MCH2",true);
p_Enemy1.EnableResearch("RES_UCS_GARG1",true);

p_Enemy2.EnableResearch("RES_UCS_WSP1",false);
p_Enemy3.EnableResearch("RES_UCS_WSP1",false);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_ED_WCA2",true);
p_Player.EnableResearch("RES_ED_MSC2",true);
p_Player.EnableResearch("RES_ED_UMT1",true);
p_Player.EnableResearch("RES_ED_RepHand",true);

// 1st tab
p_Player.EnableBuilding("EDBPP",false);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBMT",false);
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab

p_Player.EnableBuilding("EDBHQ", false);
p_Player.EnableBuilding("EDBRA", false);
p_Player.EnableBuilding("EDBN1",true);
p_Player.EnableBuilding("EDBLZ",true);

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,128,20,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,20,0,60,0);
    return ShowBriefing,200;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing214a");
    return LocateUnits,100;
}
//-----

state LocateUnits
{
    if(GetGoalState(findBase)!=goalAchieved &&
p_Player.IsPointLocated(n_HangarX,n_HangarY,0))
    {
        SetGoalState(findBase, goalAchieved);
    }
    if(p_Player.IsPointLocated(n_HangarX,n_HangarY,1))
    {
        p_Player.LookAt(n_HangarX,n_HangarY,10,0,20,1);
        SetGoalState(findHangar, goalAchieved);
        EnableGoal(findCommandCenter,true);
        bSwitchOnA=true;
        AddBriefing("translateBriefing214b");
        return ExploringBase,100;
    }
    return LocateUnits,100;
}
//-----
state ExploringBase
{
    return ExploringBase,100;
}
//-----
state Mining
{
    if(p_Player.GetMoneySentToBase()>=50000)
    {
        SetGoalState(sendToBase50000, goalAchieved);
        AddBriefing("translateAccomplished214");
        EnableEndMissionButton(true),
        return Evacuate,500;
    }
    return Mining,100;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
    if(GetDifficultyLevel()!=0)
    {
        if(u_Unit.GetFFNumber()==p_Enemy1.GetFFNumber())
        {
            p_Enemy2.Attack(n_HangarX,n_HangarY,0);
            p_Enemy3.Attack(n_HangarX,n_HangarY,0);
            p_Enemy2.EnableAIFeatures(aiControlOffense,true);
            p_Enemy3.EnableAIFeatures(aiControlOffense,true);
        }
        if(u_Unit.GetFFNumber()==p_Enemy2.GetFFNumber())
        {
            p_Enemy1.Attack(n_HangarX,n_HangarY,0);
            p_Enemy3.Attack(n_HangarX,n_HangarY,0);
        }
    }
}

```

```

        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy3.EnableAIFeatures(aiControlOffense,true);
    }
    if(n_Unit.GetFFNumber()==p_Enemy3.GetFFNumber())
    {
        p_Enemy1.Attack(n_HangarX,n_HangarY);
        p_Enemy2.Attack(n_HangarX,n_HangarY);
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy2.EnableAIFeatures(aiControlOffense,true);
    }
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
RegisterGoal(sendToBase50000,"translateGoalSend50000",p_Player.GetMoneySentToBase());
if(bShowFailed)
{
if(ResourcesLeftInMoney()>=p_Player.GetMoney()+p_Player.GetMoneySentToBase())
{
    bShowFailed=false;
    SetGoalState(sendToBase50000,goalFailed);
    AddBriefing("translateFailed214a");
    EnableEndMissionButton(true);
}
}
if(bCheckEndMission)
{
bCheckEndMission=false;
if(p_Player.GetNumberOfBuildings() && p_Player.GetNumberOfUnits())
{
    AddBriefing("translateFailed214b");
    EndMission(false);
}
}
}
//-----
event Timer1()
{
Snow(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),30,400,2500,800,10);
Snow(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),30,400,2500,800,10);
Snow(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),30,400,2500,800,10);
Snow(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),30,400,2500,800,10);
}
//-----
event Artefact(int alid,player piPlayer)
{
if(piPlayer == pPlayer)
{
    p_Player.EnableBuilding("EDBPP",true);
    p_Player.EnableBuilding("EDBBA",true);
    p_Player.EnableBuilding("EDBFA",true);
    p_Player.EnableBuilding("EDBW",true);
    p_Player.EnableBuilding("EDBAB",true);
    // 2nd tab
    p_Player.EnableBuilding("EDBRE",true);
    p_Player.EnableBuilding("EDBM",true);
    p_Player.EnableBuilding("EDBTC",true);
    // 3rd tab
    p_Player.EnableBuilding("EDBSE",true);
    // 4th tab
    p_Player.EnableBuilding("EDBHO",true);
    p_Player.EnableBuilding("EDBEN1",true);
    p_Player.EnableBuilding("EDBLZ",true);
}
}
//-----
p_Player.LookAt(n_CenterX,n_CenterY,10,0,20,1);
SetGoalState(findCommandCenter,goalAchieved);
EnableGoal(sendToBase50000,true);
AddBriefing("translateBriefing214c");
p_Neutral.GiveAllUnitsTo(p_Player);
p_Neutral.GiveAllBuildingsTo(p_Player);

```

```

        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        bShowFailed=true;
        state Mining;
        return true; //usuwa sie
    }
    return false;
}
//-----
event EndMission()
{
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.SetEnemy(p_Player);
    p_Enemy2.EnableResearch("RES_UCS_WSP1",true);
    p_Enemy3.EnableResearch("RES_UCS_WSP1",true);
}
}



## Mission215.ec


mission "translateMission215"
//Projekt LASER - siberia
consts
{
    evacuateBase = 0;
    build3ResCent = 1;
    evacuatePrototype = 2;
}
player p_Enemy;
player p_Player;
player p_Neutral;
player p_Neutral2;
unitex p_Protoype;
unitex p_Builder;

int n_EvacuatePointX;
int n_EvacuatePointY;
int nCheckEndMission;
int nAttackCounter;

state Initialize;
state ShowBriefing;
state EvacuateBase;
state BuildUpBase;
state EvacuatePrototype;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(evacuateBase,"translateGoal215a");
    RegisterGoal(build3ResCent,"translateGoal215b");
    RegisterGoal(evacuatePrototype,"translateGoal215c");
    EnableGoal(evacuateBase,true);

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(3);
    p_Neutral = GetPlayer(4);
    p_Neutral2 = GetPlayer(8);
    p_Neutral.EnableStatistics(false);
    p_Neutral2.EnableStatistics(false);

    p_Protoype = GetUnit(GetPointX(1),GetPointY(1),1);
    p_Builder = GetUnit(GetPointX(2),GetPointY(2),0);

    p_Player.SetMoney(0);
    p_Enemy.SetMoney(30000);

    p_Enemy.SetNeutral(p_Neutral);
    p_Player.SetNeutral(p_Neutral);
    p_Player.SetNeutral(p_Neutral2);
    p_Neutral.SetNeutral(p_Player);
    p_Neutral2.SetNeutral(p_Player);
    p_Enemy.SetEnemy(p_Neutral2);

    p_Player.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
        p_Enemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy.LoadScript("single\singleMedium");
}

```

```

if(GetDifficultyLevel()==2)
    p_Enemy.LoadScript("single\\singleHard");

p_Enemy.EnableAIFeatures(aiControlDefense,false);
p_Enemy.EnableAIFeatures(aiControlOffense);
p_Enemy.RussianAttack(GetPointX(2),GetPointY(2),0);

p_Neutral.EnableAIFeatures(aiEnabled,false);
p_Neutral2.EnableAIFeatures(aiEnabled,false);

p_Enemy.SetPointToAssemble(0,GetPointX(3),GetPointY(3),0);
p_Enemy.SetPointToAssemble(1,GetPointX(4),GetPointY(4),0);
p_Enemy.SetPointToAssemble(2,GetPointX(5),GetPointY(5),0);

n_EvacuatePointX = GetPointX(0);
n_EvacuatePointY = GetPointY(0);

p_Enemy.EnableResearch("RES_LC_BHD",false);
p_Enemy.EnableResearch("RES_LC_BMD",false);

p_Enemy.EnableResearch("RES_LC_WSL1",true);
p_Enemy.EnableResearch("RES_LC_WCH2",true);
p_Enemy.EnableResearch("RES_MCH2",true);
p_Enemy.EnableResearch("RES_LC_JM02",true);
p_Enemy.EnableResearch("RES_LC_JM1",true);

p_Player.EnableResearch("RES_ED_WSR1",true);
p_Player.EnableResearch("RES_ED_MSC2",true);
p_Player.EnableResearch("RES_ED_LMT1",true);
p_Player.EnableResearch("RES_ED_JA11",true);
p_Player.EnableResearch("RES_ED_JA12",true);
p_Player.EnableResearch("RES_ED_RepHand",true);

// 1st tab
p_Player.EnableBuilding("EDBPP",false);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBM",false);
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHQ",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBEN1",true);
p_Player.EnableBuilding("EDBLZ",true);

SetTimer(0,100);
SetTimer(1,6000);
bCheckEndMission = false;

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),8,0,20,
0);
p_Player.SetMilitaryUnitsLimit(15000);
return ShowBriefing,100;
}

state ShowBriefing
{
    AddBriefing("translateBriefing215a");
    return EvacuateBase,100;
    p_Enemy.SetEnemy(p_Neutral2);
}

state EvacuateBase
{
    nAttackCounter=nAttackCounter+1;
    if(nAttackCounter>15)
    {
        nAttackCounter=0;
        p_Enemy.RussianAttack(GetPointX(2),GetPointY(2),0);
    }
    p_Enemy.SetEnemy(p_Neutral2);

if(Distance(p_Builder.GetLocationX(),p_Builder.GetLocationY(),n_EvacuatePointX,
_n_EvacuatePointY) < 18)
    {
        SetGoalState(evacuateBase,goalAchieved);
        EnableGoal(build3ResCent,true);
        p_Player.SetMoney(15000);
        AddBriefing("translateBriefing215b");
    }

    // 1st tab
    p_Player.EnableBuilding("EDBPP",true);
    p_Player.EnableBuilding("EDBBA",true);
    p_Player.EnableBuilding("EDBWA",true);
    p_Player.EnableBuilding("EDBWB",true);
    p_Player.EnableBuilding("EDBAB",true);
    // 2nd tab
    p_Player.EnableBuilding("EDBRE",true);
    p_Player.EnableBuilding("EDBM",true);
    p_Player.EnableBuilding("EDBTC",true);
    // 3rd tab
    p_Player.EnableBuilding("EDBST",true);
    // 4th tab
    p_Player.EnableBuilding("EDBRC",true);
    p_Player.EnableBuilding("EDBHQ",true);
    p_Player.EnableBuilding("EDBRA",true);
    p_Player.EnableBuilding("EDBEN1",true);
    p_Player.EnableBuilding("EDBLZ",true);

    p_Enemy.EnableAIFeatures(aiControlDefense,true);
    return BuildUpBase,200;
}
}

state BuildUpBase
{
if(p_Player.GetNumberOfBuildings(buildingResearchCenter)>2)
{
    SetGoalState(build3ResCent,goalAchieved);
    p_Protoype.ChangePlayer(p_Player);
    EnableGoal(evacuatePrototype,true);
    AddBriefing("translateBriefing215c");
    p_Enemy.EnableAIFeatures(aiControlOffense,true);
    return EvacuatePrototype;
}
/*if(lp_Player.GetMoney())
{
    EnableNextMission(1,true);
    EnableNextMission(2,true);
    AddBriefing("translateFailed215c");
    EndMission(false);
}
*/
return BuildUpBase,200;
}

state EvacuatePrototype
{
if(Distance(p_Protoype.GetLocationX(),p_Protoype.GetLocationY(),n_EvacuateP
ointX,n_EvacuatePointY) < 5)
{
    SetGoalState(evacuatePrototype,goalAchieved);
    EnableNextMission(0,true);
    AddBriefing("translateAccomplished215");
    EnableEndMissionButton(true);
    return Final,100;
}
return EvacuatePrototype,100;
}

state Final
{
    return Final,500;
}

event Timer0() //wolny co 100 cykl< ustwione funkcja SetTimer w state
Initial
{
if(GetGoalState(evacuateBase)==goalAchieved && lp_Builder.IsLive())
{
    AddBriefing("translateFailed215a");
    EnableNextMission(1,true);
    EnableNextMission(2,true);
    EndMission(false);
}
if(lp_Protoype.IsLive())
{
    AddBriefing("translateFailed215b");
    EnableNextMission(1,true);
    EnableNextMission(2,true);
}
}

```

```

        EndMission(false);
    }

    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed215d");
            EnableNextMission(1,true);
            EnableNextMission(2,true);
            EndMission(false);
        }
    }
}

event Timer1() //wolany co 6000 cykli 5min
{
    Snow(n_EvacuatePointX,n_EvacuatePointY,40,400,2500,800,5);
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event EndMission()
{
    p_Enemy.SetEnemy(p_Neutral);
    p_Player.SetEnemy(p_Neutral);
    p_Player.SetEnemy(p_Neutral2);
    p_Neutral.SetEnemy(p_Player);
    p_Neutral2.SetEnemy(p_Player);
}
}
}

```

Mission221.ec

```

mission "translateMission221"
{/*
Laser weapon tests
*/
const
{
    destroyDummies = 0;
    destroyEnemy = 1;
}

player p_Enemy;
player p_Player;
player p_Dummy;
player p_Neutral;

state Initialize;
state ShowBriefing;
state DestroyDummies;
state DestroyEnemy;
state Evacuate;

//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(destroyDummies,"translateGoal221a");
    RegisterGoal(destroyEnemy,"translateGoal221b");
    EnableGoal(destroyDummies,true);

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(3);
    p_Dummy = GetPlayer(4);
    p_Neutral = GetPlayer(8);
    p_Player.SetMoney(0);
    p_Enemy.SetMoney(0);
    p_Dummy.SetMoney(0);
    p_Neutral.SetMoney(0);
}

p_Neutral.EnableStatistics(false);
}

```

```

p_Dummy.EnableStatistics(false);

if(GetDifficultyLevel()==0)
{
    p_Enemy.LoadScript("single\singleEasy");
    p_Enemy.SetMoney(15000);
}
if(GetDifficultyLevel()==1)
{
    p_Enemy.LoadScript("single\singleMedium");
    p_Enemy.SetMoney(30000);
}
if(GetDifficultyLevel()==2)
{
    p_Enemy.LoadScript("single\singleHard");
    p_Enemy.SetMoney(50000);
}

p_Enemy.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableAIFeatures(aiEnabled,false);
p_Player.EnableAIFeatures(aiEnabled,false);
p_Dummy.EnableAIFeatures(aiEnabled,false);

p_Dummy.SetNeutral(p_Player);
p_Dummy.SetNeutral(p_Neutral);
p_Neutral.SetNeutral(p_Player);
p_Neutral.SetNeutral(p_Dummy);
p_Player.SetNeutral(p_Neutral);

p_Enemy.SetPointToAssemble(0,GetPointX(0),GetPointY(0),0);
p_Enemy.SetPointToAssemble(1,GetPointX(1),GetPointY(1),0);
p_Enemy.SetPointToAssemble(2,GetPointX(2),GetPointY(2),0);

p_Enemy.EnableResearch("RES_LC_WSR2",true);
p_Enemy.EnableResearch("RES_LC_REG1",true);

p_Player.EnableBuilding("EDBRA",false);

SetTimer(0,100);
ShowArea(4,GetPointX(5),GetPointY(5),0,1);
CallCamera(),);

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
    return ShowBriefing,100;
}
state ShowBriefing
{
    AddBriefing("translateBriefing221a");
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    return DestroyDummies,100;
}

//-----
state DestroyDummies
{
    if(lp_Dummy.GetNumberOfUnits())
    {
        p_Neutral.GiveAllUnitsTo(p_Player);
        p_Neutral.GiveAllBuildingsTo(p_Player);
        p_Player.EnableResearch("RES_ED_WSL1",true);
        SetGoalState(destroyDummies,goalAchieved);
        EnableGoal(destroyEnemy,true);
        AddBriefing("translateBriefing221b");
        p_Player.SetMoney(2000);
        p_Enemy.EnableAIFeatures(aiEnabled,true);
        return DestroyEnemy,200;
    }
    return DestroyDummies,100;
}

//-----
state DestroyEnemy
{
    if(p_Enemy.GetNumberOfUnits()<10 && lp_Enemy.GetNumberOfBuildings())
    {
        SetGoalState(destroyEnemy,goalAchieved);
        AddBriefing("translateAccomplished221");
        EnableEndMissionButton(true);
        return Evacuate;
    }
    return DestroyEnemy,100;
}

```

```

//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    if((p_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings()))
    {
        AddBriefing("translateFailed221");
        EndMission(false);
    }
}
//-----
event EndMission()
{
    p_Dummy.SetEnemy(p_Player);
    p_Dummy.SetEnemy(p_Neutral);
    p_Neutral.SetEnemy(p_Player);
    p_Neutral.SetEnemy(p_Dummy);
    p_Player.SetEnemy(p_Neutral);
}
}

Mission222.ec
mission "translateMission222"
//Destroy UCS research facility
const
{
    recoverComputerData=0;
    destroyResearchFacility = 1;
}

player p_Enemy1;
player p_Enemy2;
player p_GoalEnemy;
player p_Player;

state Initialize;
state ShowBriefing;
state RecoverComputerData;
state DestroyResearchFacility;
state Evacuate;

//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(recoverComputerData,"translateGoal222a");
    RegisterGoal(destroyResearchFacility,"translateGoal222b");
    EnableGoal(recoverComputerData,true);

    //           name      x,y,z, nr,typ
CreateArtefact("NEASPECIAL2",GetPointX(0),GetPointY(0),1,0,artefactSpecialAI0ther);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(5);
    p_GoalEnemy = GetPlayer(4);

    p_Player.SetMoney(20000);
    p_Enemy1.SetMoney(20000);
    p_Enemy2.SetMoney(20000);
    p_GoalEnemy.SetMoney(20000);

    p_Player.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }
}

p_GoalEnemy.LoadScript("single\singleEasy");
p_GoalEnemy.LoadScript("single\singleEasy");
p_GoalEnemy.EnableAIFeatures(aiControlOffense,false);
p_GoalEnemy.EnableAIFeatures(aiControlOffense,false);
p_GoalEnemy.EnableAIFeatures(aiControlOffense,false);

p_GoalEnemy.SetMaxTankPlatoonSize(3);
p_GoalEnemy.SetMaxShipPlatoonSize(4);

p_GoalEnemy.SetNumberOffensiveTankPlatoons(0);
p_GoalEnemy.SetNumberOffensiveShipPlatoons(0);
p_GoalEnemy.SetNumberOffensiveHelicopterPlatoons(0);

p_GoalEnemy.SetNumberDefensiveTankPlatoons(3);
p_GoalEnemy.SetNumberDefensiveShipPlatoons(0);
p_GoalEnemy.SetNumberDefensiveHelicopterPlatoons(0);

p_Enemy1.SetNeutral(p_Enemy2);
p_Enemy2.SetNeutral(p_Enemy1);

p_Enemy1.SetPointToAssemble(0,GetPointX(1),GetPointY(1,0));
p_Enemy1.SetPointToAssemble(1,GetPointX(2),GetPointY(2,0));
p_Enemy2.SetPointToAssemble(0,GetPointX(3),GetPointY(3,0));
p_Enemy2.SetPointToAssemble(1,GetPointX(4),GetPointY(4,0));

ShowArea(4,p_GoalEnemy.GetStartingPointX(),p_GoalEnemy.GetStartingPointY(),0,2);

ShowArea(4,p_Enemy1.GetStartingPointX(),p_Enemy1.GetStartingPointY(),0,2);

ShowArea(4,p_Enemy2.GetStartingPointX(),p_Enemy2.GetStartingPointY(),0,2);

p_Enemy1.EnableResearch("RES_UCS_WASR1",true);
p_Enemy1.EnableResearch("RES_UCS_WCH2",true);
p_Enemy1.EnableResearch("RES_MCH2",true);
p_Enemy1.EnableResearch("RES_UCS_UOH2",true);
p_Enemy1.EnableResearch("RES_UCS_RepHand",true);

p_Enemy2.CopyResearches(p_Enemy1);
p_GoalEnemy.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_ED_UMW1",true);
p_Player.EnableResearch("RES_ED_USS2",true);
p_Player.EnableResearch("RES_ED_UA21",true);

p_Player.EnableBuilding("EDBRA",false);

SetTimer(0,200);

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}

state ShowBriefing
{
    AddBriefing("translateBriefing222a");
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    return RecoverComputerData,100;
}

state RecoverComputerData
{
    if(GetGoalState(recoverComputerData)==goalAchieved)
    {
        return DestroyResearchFacility,200;
    }
    return RecoverComputerData,100;
}

state DestroyResearchFacility
{
    if(lp_GoalEnemy.GetNumberOfBuildings(buildingResearchCenter))
    {
        p_Player.EnableBuilding("EDBT",true);
        SetGoalState(destroyResearchFacility,goalAchieved);
        AddBriefing("translateAccomplished222");
        EnableEndMissionButton(true);
    }
}

```

```

        return Evacuate,500;
    }
    return DestroyResearchFacility,100;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event Timer0()
{
    if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed222");
        EndMission(false);
    }
}
//-----
event EndMission()
{
    p_Enemy1.SetEnemy(p_Enemy2);
    p_Enemy2.SetEnemy(p_Enemy1);
}
//-----
event Artefact(int alld,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    p_Enemy1.EnableAIFeatures(aiControlOffense,true);
    p_Enemy2.EnableAIFeatures(aiControlOffense,true);
    SetGoalState(recoverComputerData,goalAchieved);
    EnableGoal(destroyResearchFacility,true);

    if(lp_GoalEnemy.GetNumberOfBuildings(buildingResearchCenter))
    {
        SetGoalState(destroyResearchFacility,goalAchieved);
        AddBriefing("translateAccomplished222");
        EnableEndMissionButton(true);
        state Evacuate;
    }
    else
    {
        AddBriefing("translateBriefing222b");
        state DestroyResearchFacility;
    }
    return true; //usuwa sie
}
}

```

Mission223.ec

```

mission "translateMission223"
{/Alaska - Mine 100 000 resources
const
{
    destroyEnemyBase = 0;
    sendToBase100000 = 1;
}

player p_Enemy1;
player p_Enemy2;
player p_GoalEnemy;
player p_Player;
int bShowFailed;

state Initialize;
state ShowBriefing;
state DestroyEnemyBase;
state ShowVideoState;
state Mining;
state Evacuate;
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(destroyEnemyBase,"translateGoal223a");
    RegisterGoal(sendToBase100000,"translateGoal223b",0);
    EnableGoal(destroyEnemyBase,true);
    EnableGoal(sendToBase100000,true);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
}

```

```

p_Enemy2 = GetPlayer(5);
p_GoalEnemy = GetPlayer(4);

p_Player.SetEnemy(p_GoalEnemy);

p_Player.SetMoney(20000);
p_Enemy1.SetMoney(20000);
p_Enemy2.SetMoney(20000);
p_GoalEnemy.SetMoney(12000);

if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\\singleEasy");
    p_Enemy2.LoadScript("single\\singleEasy");
    p_GoalEnemy.LoadScript("single\\singleEasy");
    p_GoalEnemy.EnableAIFeatures(aiDefenseTowers,false);
}
if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\\singleMedium");
    p_Enemy2.LoadScript("single\\singleMedium");
    p_GoalEnemy.LoadScript("single\\singleMedium");
    p_GoalEnemy.EnableAIFeatures(aiDefenseTowers,false);
}
if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\\singleHard");
    p_Enemy2.LoadScript("single\\singleHard");
    p_GoalEnemy.LoadScript("single\\singleMedium");
    p_GoalEnemy.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiControlOffense,fase);
    p_Enemy2.EnableAIFeatures(aiControlOffense,fase);
    p_GoalEnemy.EnableAIFeatures(aiControlOffense,fase);
}

p_Player.EnableAIFeatures(aiEnabled,fase);
p_Enemy1.EnableAIFeatures(aiControlOffense,fase);
p_Enemy2.EnableAIFeatures(aiControlOffense,fase);
p_GoalEnemy.EnableAIFeatures(aiControlOffense,fase);

p_Enemy1.SetNumberOfOffensiveTankPlatoons(4);
p_Enemy2.SetNumberOfOffensiveTankPlatoons(0);

p_GoalEnemy.SetMaxTankPlatoonSize(3);

p_GoalEnemy.SetNumberOfOffensiveTankPlatoons(0);
p_GoalEnemy.SetNumberOfOffensiveShipPlatoons(0);
p_GoalEnemy.SetNumberOfOffensiveHelicopterPlatoons(0);

p_GoalEnemy.SetNumberOfDefensiveTankPlatoons(4);
p_GoalEnemy.SetNumberOfDefensiveShipPlatoons(0);
p_GoalEnemy.SetNumberOfDefensiveHelicopterPlatoons(0);

```

ShowArea(4,p_GoalEnemy.GetStartingPointX(),p_GoalEnemy.GetStartingPointY(),0,2);

```

p_Enemy1.EnableResearch("RES_UCS_WSG1",true);
p_Enemy1.EnableResearch("RES_MSR2",true);
p_Enemy1.EnableResearch("RES_UCS_UML1",true);
p_Enemy1.EnableResearch("RES_UCS_BMD",true);

p_Enemy2.CopyResearches(p_Enemy1);
p_GoalEnemy.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_ED_WSR2",true);
p_Player.EnableResearch("RES_ED_UA21",true);
p_Player.EnableResearch("RES_ED_BMD",true);

// 1st tab
p_Player.EnableBuilding("EDBPP",true);
p_Player.EnableBuilding("EDBBA",true);
p_Player.EnableBuilding("EDBFA",true);
p_Player.EnableBuilding("EDBWB",true);
p_Player.EnableBuilding("EDBAB",true);
// 2nd tab
p_Player.EnableBuilding("EDBRE",true);
p_Player.EnableBuilding("EDBM",true);
p_Player.EnableBuilding("EDBTC",true);
// 3rd tab
p_Player.EnableBuilding("EDBTS",true);
// 4th tab
p_Player.EnableBuilding("EDBHQ",true);
p_Player.EnableBuilding("EDBRA",true);
p_Player.EnableBuilding("EDBEN1",true);
p_Player.EnableBuilding("EDBLZ",true);

```

```

SetTimer(0,200);
SetTimer(1,6000);
bShowFailed=true;
CallCamera1();
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
AddBriefing("translateBriefing223a");
//EnableEndMissionButton(true);//XXXMD!!!!!!usunac
return DestroyEnemyBase,120;
}

//-----
state DestroyEnemyBase
{
if(lp_GoalEnemy.GetNumberOfBuildings())
{
EnableNextMission(0,true);
SetGoalState(destroyEnemyBase,goalAchieved);
AddBriefing("translateBriefing223b");//transmija ze program badawczy.
p_Enemy1.EnableAIFeatures(aiControlOffense,true);
p_Enemy2.EnableAIFeatures(aiControlOffense,true);
return ShowVideoState,20;
}
return DestroyEnemyBase,100;
}

//-----
state ShowVideoState
{
ShowVideo("CS210");
return Mining,500;
}

//-----
state Mining
{
if(GetGoalState(sendToBase100000)!=goalAchieved &&
p_Player.GetMoneySentToBase()>=100000)
{
SetGoalState(sendToBase100000,goalAchieved);
AddBriefing("translateAccomplished223");
EnableEndMissionButton(true);
return Evacuate,500;
}
return Mining,100;
}

//-----
state Evacuate
{
return Evacuate,500;
}

//-----
event Timer0()
{
RegisterGoal(sendToBase100000,"translateGoal223b",p_Player.GetMoneySentToB
ase());
if(p_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
{
AddBriefing("translateFailed223b");
EndMission(false);
}
if(bShowFailed)
{
// SetConsoleText("%i<%0>, p;<%1>
s;<%2>";ResourcesLeftInMoney(),p_Player.GetMoney(),p_Player.GetMoneySentTo
Base());
if((ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase
())<100000)
{
bShowFailed=false;
SetGoalState(sendToBase100000,goalFailed);
AddBriefing("translateFailed223a");
EnableEndMissionButton(true);
return Evacuate;
}
}
}

//-----
event Timer1()

```

```

{
Snow(p_GoalEnemy.GetStartingPointX(),p_GoalEnemy.GetStartingPointY(),40,400,
2500,800,4);
}
}
```

Mission224.ec

```

mission "translateMission224"
{
consts
{
destroyEnemy1 = 0;
destroyEnemy2 = 1;
destroyEnemy3 = 2;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Player;
int bSwitchOnA;
int bCheckEndMission;
state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;
//-----
state Initialize
{
player tmpPlayer;
tmpPlayer = GetPlayer(1);
tmpPlayer.EnableStatistics(false);

RegisterGoal(destroyEnemy1,"translateGoal224a");
RegisterGoal(destroyEnemy2,"translateGoal224b");
RegisterGoal(destroyEnemy3,"translateGoal224c");
EnableGoal(destroyEnemy1,true);
EnableGoal(destroyEnemy2,true);
EnableGoal(destroyEnemy3,true);

p_Player = GetPlayer(2);
p_Enemy1 = GetPlayer(3);
p_Enemy2 = GetPlayer(4);
p_Enemy3 = GetPlayer(5);

p_Player.SetMoney(20000);
p_Enemy1.SetMoney(20000);
p_Enemy2.SetMoney(20000);
p_Enemy3.SetMoney(20000);

p_Player.EnableAIFeatures(aiEnabled,false);

if(GetDifficultyLevel()==0)
{
p_Enemy1.LoadScript("single\singleEasy");
p_Enemy2.LoadScript("single\singleEasy");
p_Enemy3.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel()==1)
{
p_Enemy1.LoadScript("single\singleMedium");
p_Enemy2.LoadScript("single\singleMedium");
p_Enemy3.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
p_Enemy1.LoadScript("single\singleHard");
p_Enemy2.LoadScript("single\singleHard");
p_Enemy3.LoadScript("single\singleHard");
}

p_Enemy2.SetNumberOfOffensiveTankPlatoons(0);
p_Enemy2.SetNumberOfOffensiveShipPlatoons(0);
p_Enemy2.SetNumberOfOffensiveHelicopterPlatoons(0);

p_Enemy3.SetNumberOfDefensiveTankPlatoons(0);
p_Enemy3.SetNumberOfDefensiveShipPlatoons(0);
p_Enemy3.SetNumberOfDefensiveHelicopterPlatoons(0);

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);
}
```

```

p_Enemy1.EnableResearch("RES_LC_WSR2",true);
p_Enemy1.EnableResearch("RES_MS2",true);
p_Enemy1.EnableResearch("RES_LC_REG1",true);
p_Enemy1.EnableResearch("RES_LC_SGEN",true);
p_Enemy1.EnableResearch("RES_LC_BWC",true);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_ED_WSR2",true);
p_Player.EnableResearch("RES_ED_UA21",true);
p_Player.EnableResearch("RES_ED_UIS2",true);
p_Player.EnableResearch("RES_ED_UNW1",true);
p_Player.EnableResearch("RES_ED_BMD",true);

// 1st tab
p_Player.EnableBuilding("EDBPP",true);
p_Player.EnableBuilding("EDDBAA",true);
p_Player.EnableBuilding("EDBFA",true);
p_Player.EnableBuilding("EDBWB",true);
p_Player.EnableBuilding("EDBBA",true);
// 2nd tab
p_Player.EnableBuilding("EDBRE",true);
p_Player.EnableBuilding("EDBMB",true);
p_Player.EnableBuilding("EDBTC",true);
// 3rd tab
p_Player.EnableBuilding("EDBST",true);
// 4th tab
p_Player.EnableBuilding("EDBRC",true);
p_Player.EnableBuilding("EDBHQ",true);
p_Player.EnableBuilding("EDBRA",true);
p_Player.EnableBuilding("EDBEN1",true);
p_Player.EnableBuilding("EDBLZ",true);

bShitchOnAI=true;
bCheckEndMission = false;

SetTimer(0,100);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);

Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),10,500,5000,500,10);
EnableNextMission(1,true); //nie wlaczac 0 bo generuje to trudny do uchwyca blad
return ShowBriefing,100;
}

state ShowBriefing
{
    AddBriefing("translateBriefing224");
    return Fighting,100;
}

state Fighting
{
    if(bShitchOnAI &&
(p_Player.IsPointLocated(p_Enemy1.GetStartingPointX(),p_Enemy1.GetStartingPointY(),0)) ||

p_Player.IsPointLocated(p_Enemy2.GetStartingPointX(),p_Enemy2.GetStartingPointY(),0)) ||
p_Player.IsPointLocated(p_Enemy3.GetStartingPointX(),p_Enemy3.GetStartingPointY(),0))
    {
        bShitchOnAI=false;
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy3.EnableAIFeatures(aiControlOffense,true);
    }

    if(GetGoalState(destroyEnemy1)!=goalAchieved &&
lp_Enemy1.GetNumberOfBuildings()))
    {
        SetGoalState(destroyEnemy1, goalAchieved);
    }
    if(GetGoalState(destroyEnemy2)!=goalAchieved &&
lp_Enemy2.GetNumberOfBuildings()))
    {
        SetGoalState(destroyEnemy2, goalAchieved);
    }
    if(GetGoalState(destroyEnemy3)!=goalAchieved &&
lp_Enemy3.GetNumberOfBuildings()))
    {
        SetGoalState(destroyEnemy3, goalAchieved);
    }
    if(GetGoalState(destroyEnemy1)==goalAchieved &&
GetGoalState(destroyEnemy2)==goalAchieved &&
GetGoalState(destroyEnemy3)==goalAchieved)
    {
        AddBriefing("translateAccomplished224");

        EnableEndMissionButton(true);
        return Evacuate;
    }
    return Fighting, 100;
}

state Evacuate
{
    return Evacuate, 500;
}

event Timer0()
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed224");
            EndMission(false);
        }
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
    if(bShitchOnAI)
    {
        bShitchOnAI=false;
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy3.EnableAIFeatures(aiControlOffense,true);
    }
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
    if(bShitchOnAI)
    {
        bShitchOnAI=false;
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy3.EnableAIFeatures(aiControlOffense,true);
    }
}

mMission231.ec
ission "translateMission231"
//Destroy second UCS research facility
consts
{
    destroyResearchFacility = 0;
    searchOutAlienBase = 1;
}

player p_Enemy1;
player p_Enemy2;
player p_GoalEnemy;
player p_Player;

state Initialize;
state ShowBriefing;
state DestroyResearchFacility;
state Evacuate;

//-----
state Initialize
{
}

```

```

player tmpPlayer;
//----- Temporary players -----
tmpPlayer = GetPlayer(3);
tmpPlayer.EnableStatistics(false);
//----- Goals -----
RegisterGoal(destroyResearchFacility,"translateGoal231a");
RegisterGoal(searchOutAlienBase,"translateGoal231b");
EnableGoal(destroyResearchFacility,true);

//----- Players -----
p_Player = GetPlayer(2);
p_Enemy1 = GetPlayer(1);
p_Enemy2 = GetPlayer(5);
p_GoalEnemy = GetPlayer(4);
//----- AI -----
p_Player.EnableAIFeatures(aiEnabled,false);
//----- Money -----
p_Player.SetMoney(20000);
p_Enemy1.SetMoney(20000);
p_Enemy2.SetMoney(50000);
p_GoalEnemy.SetMoney(20000);

if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_GoalEnemy.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_GoalEnemy.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_GoalEnemy.LoadScript("single\singleHard");
}

p_Enemy2.LoadScript("single\singleHard");

p_GoalEnemy.SetNumberOfOffensiveTankPlatoons(0);
p_GoalEnemy.SetNumberOfOffensiveShipPlatoons(0);
p_GoalEnemy.SetNumberOfOffensiveHelicopterPlatoons(0);

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy1.SetPointToAssemble(0,GetPointX(2),GetPointY(2),0);
p_Enemy1.SetPointToAssemble(1,GetPointX(3),GetPointY(3),0);

p_GoalEnemy.EnableAIFeatures(aiControlOffense,false);

p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);

p_Enemy2.setMaxTankPlatoonSize(5);

p_Enemy2.SetNumberOfOffensiveTankPlatoons(0);
p_Enemy2.SetNumberOfOffensiveShipPlatoons(0);
p_Enemy2.SetNumberOfOffensiveHelicopterPlatoons(0);

p_Enemy2.SetNumberOfDefensiveTankPlatoons(4);
p_Enemy2.SetNumberOfDefensiveShipPlatoons(0);
p_Enemy2.SetNumberOfDefensiveHelicopterPlatoons(0);

//----- Researches -----
p_Enemy1.EnableResearch("RES_UCS_WSP1",true);
p_Enemy1.EnableResearch("RES_UCS_UML1",true);

p_Enemy2.CopyResearches(p_Enemy1);
p_GoalEnemy.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_ED_WSR1",true);
p_Player.EnableResearch("RES_ED_UA22",true);
//----- Buildings -----
//----- Units -----
//----- Artefacts -----
//----- name      x,y,z, nr,typ
CreateArtifact("NEASPECIAL2",GetPointX(0),GetPointY(0),1.0,artifactSpecialAIne
wAreaLocation);
//----- Timers -----
SetTimer(0,200);
//----- Variables -----
//----- Camera -----
ShowArea(4,p_GoalEnemy.GetStartingPointX(),p_GoalEnemy.GetStartingPointY(),0,
2);

ShowArea(4,GetPointX(1),GetPointY(1),0.2);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6.0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing231a");
    EnableNextMission(0,true);
    return DestroyResearchFacility,200;
}

//-----
state DestroyResearchFacility
{
    if(GetGoalState(searchOutAlienBase)==goalAchieved &&
       GetGoalState(destroyResearchFacility)==goalAchieved)
    {
        AddBriefing("translateAccomplished231b");
        EnableEndMissionButton(true);
        return Evacuate,500;
    }

    if(GetGoalState(destroyResearchFacility)==goalAchieved &&
       lp_GoalEnemy.GetNumberOfBuildings(buildingResearchCenter))
    {
        SetGoalState(destroyResearchFacility, goalAchieved);
        EnableGoal(searchOutAlienBase,true);
        AddBriefing("translateBriefing231b");
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
    }
}

return DestroyResearchFacility,100;
}

//-----
state Evacuate
{
    return Evacuate,500;
}

//-----
event Timer0()
{
    if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed231");
        EndMission(false);
    }
}

//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer==p_Player) return false;
    SetGoalState(searchOutAlienBase, goalAchieved);
    EnableNextMission(1,true);
    return true; //usuwa sie
}

}



## Mission232.ec


mission "translateMission232"
//Malou island
consts
{
    sendToBase = 0;
    nNeededResources=100000;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Enemy4;
player p_Player;

int bShowFailed;
int bCheckEndMission;

state Initialize;
state ShowBriefing;
state Mining;
state Nothing;

```

```

//-----
state Initialize
{
    int enemyStartingMoney;
    int playerStartingMoney;

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);
    p_Enemy3 = GetPlayer(4);
    p_Enemy4 = GetPlayer(5);

    RegisterGoal(sendToBase,"translateGoalSend100000",0);
    EnableGoal(sendToBase,true);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy3.LoadScript("single\singleEasy");
        p_Enemy4.LoadScript("single\singleEasy");
        p_Enemy3.EnableAIFeatures(aiControlOffense,false);
        p_Enemy4.EnableAIFeatures(aiControlOffense,false);
        playerStartingMoney=30000;
        enemyStartingMoney=20000;
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy3.LoadScript("single\singleMedium");
        p_Enemy4.LoadScript("single\singleMedium");
        playerStartingMoney=20000;
        enemyStartingMoney=30000;
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy3.LoadScript("single\singleHard");
        p_Enemy4.LoadScript("single\singleHard");
        playerStartingMoney=10000;
        enemyStartingMoney=50000;
    }
}

p_Player.SetMoney(playerStartingMoney);
p_Enemy1.SetMoney(enemyStartingMoney);
p_Enemy2.SetMoney(enemyStartingMoney);
p_Enemy3.SetMoney(enemyStartingMoney);
p_Enemy4.SetMoney(enemyStartingMoney);

p_Player.EnableAIFeatures(aiEnabled);
p_Enemy1.EnableAIFeatures(aiRush,false);
p_Enemy2.EnableAIFeatures(aiRush,false);
p_Enemy3.EnableAIFeatures(aiRush,false);
p_Enemy4.EnableAIFeatures(aiRush,false);

p_Enemy1.EnableResearch("RES_UCS_WASR1",true);
p_Enemy1.EnableResearch("RES_UCS_WCH2",true);
p_Enemy1.EnableResearch("RES_UCS_WSP1",true);
p_Enemy1.EnableResearch("RES_UCS_WSG1",true);
p_Enemy1.EnableResearch("RES_MCL2",true);
p_Enemy1.EnableResearch("RES_MS2",true);
p_Enemy1.EnableResearch("RES_UCS_MG2",true);
p_Enemy1.EnableResearch("RES_UCS_UML1",true);
p_Enemy1.EnableResearch("RES_UCS_UHL1",true);
p_Enemy1.EnableResearch("RES_BMD",true);
p_Enemy1.EnableResearch("RES_UCS_BHD",true);

p_Enemy3.CopyResearches(p_Enemy1);

p_Enemy2.EnableResearch("RES_LC_WSR2",true);
p_Enemy2.EnableResearch("RES_LC_WSS1",true);
p_Enemy2.EnableResearch("RES_LC_WMR1",true);
p_Enemy2.EnableResearch("RES_MS2",true);
p_Enemy2.EnableResearch("RES_LC_UCR1",true);
p_Enemy2.EnableResearch("RES_LC_UBO1",true);
p_Enemy2.EnableResearch("RES_LC_REG1",true);
p_Enemy2.EnableResearch("RES_LC_SGEN",true);
p_Enemy2.EnableResearch("RES_LC_SHR1",true);
p_Enemy2.EnableResearch("RES_LC_BWC",true);

p_Enemy4.CopyResearches(p_Enemy2);
}

//-----
p_Player.EnableResearch("RES_ED_WMR1",true);
p_Player.EnableResearch("RES_MSR2",true);
p_Player.EnableResearch("RES_ED_UHW1",true);
p_Player.EnableResearch("RES_ED_BHD",true);

SetTimer(0,100);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);

bShowFailed=true;
bCheckEndMission=false;
p_Player.SetMilitaryUnitsLimit(20000);
return ShowBriefing,150://15 sec
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing232",nNeededResources);
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    EnableNextMission(2,true);
    return Mining,200;
}

//-----
state Mining
{
    if((GetGoalState(sendToBase)!=goalAchieved &&
p_Player.GetMoneySentToBase()>=nNeededResources)
    {
        SetGoalState(sendToBase,goalAchieved);
        AddBriefing("translateAccomplished232");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event Timer0() //wolany co 100 cykli< ustawiione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase,"translateGoalSend100000",p_Player.GetMoneySentToBase());
if(bShowFailed)
{
    if(ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase()
()<nNeededResources)
    {
        bShowFailed=false;
        SetGoalState(sendToBase,goalFailed);
        AddBriefing("translateFailed232a");
        EnableEndMissionButton(true);
        return Nothing;
    }
}

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lp_Player.GetNumberOfWorkUnits() && lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed232b");
        EndMission(false);
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

```

Mission233.ec

```
mission "translateMission233"
{/Spy Run
consts
{
    findData1 = 0;
    findData2 = 1;
    findData3 = 2;
    backToLandingZone=3;
}

player p_Enemy1;
player p_Neutral;
player p_Player;
unitex p_Spy;

state Initialize;
state ShowBriefing;
state Spy;
state Evacuate;

//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    tmpPlayer = GetPlayer(8);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(findData1,"translateGoal233a");
    RegisterGoal(findData2,"translateGoal233b");
    RegisterGoal(findData3,"translateGoal233c");
    RegisterGoal(backToLandingZone,"translateGoal233d");

    EnableGoal(findData1,true);
    EnableGoal(findData2,true);
    EnableGoal(findData3,true);

    //      name      x,y,z, nr,typ
CreateArtifact("NEASPECIAL2",GetPointX(1),GetPointY(1),1,0,artefactSpecialAIOther);
CreateArtifact("NEASPECIAL2",GetPointX(2),GetPointY(2),1,1,artefactSpecialAIOther);
CreateArtifact("NEASPECIAL2",GetPointX(3),GetPointY(3),1,2,artefactSpecialAIOther);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Neutral = GetPlayer(8);

    p_Player.SetMoney(0);
    p_Enemy1.SetMoney(10000);
    p_Neutral.SetMoney(0);

    p_Spy = GetUnit(GetPointX(0),GetPointY(0),0);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
    }

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Neutral.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiControlOffense,false);
    p_Enemy1.EnableAIFeatures(aiControlDefense,false);

    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);

    SetTimer(0,200);
}

CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing233a");
    return Spy,200;
}
//-----
state Spy
{
    if(GetGoalState(findData1)==goalAchieved &&
       GetGoalState(findData2)==goalAchieved &&
       GetGoalState(findData3)==goalAchieved)
    {
        p_Neutral.GiveAllBuildingsTo(p_Player);
        p_Enemy1.GiveAllUnitsTo(p_Player);
        EnableGoal(backToLandingZone,true);
        AddBriefing("translateBriefing233b"); //back to lz
        return Evacuate,50;
    }
    return Spy,100;
}
//-----
state Evacuate
{
    if(GetGoalState(backToLandingZone)!=goalAchieved &&
       Distance(p_Spy.GetLocationX(),p_Spy.GetLocationY(),p_Player.GetStartingPointX(),p_Player.GetStartingPointY()) < 5)
    {
        SetGoalState(backToLandingZone, goalAchieved);
        //EnableNextMission(0,2); //enable UCS base
        AddBriefing("translateAAccomplished233");
        EnableEndMissionButton(true);
    }
    return Evacuate,50;
}
//-----
event Timer0()
{
    if(!p_Spy.IsLive())
    {
        AddBriefing("translateFailed233");
        EndMission(false);
    }
}
//-----
event Artefact(int aiD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(aiD==0)
        SetGoalState(findData1, goalAchieved);
    if(aiD==1)
        SetGoalState(findData2, goalAchieved);
    if(aiD==2)
        SetGoalState(findData3, goalAchieved);
    return true; //usuwa sie
}
//-----
event EndMission()
{
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
}
```

Mission234.ec

```
mission "translateMission234"
{/Amazonka
consts
{
    sendToBase = 0;
    nNeededResources=100000;
}

player p_Enemy1;
```

```

player p_Enemy2;
player p_Player;

int bShowFailed;
int bCheckEndMission;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;

//-----
state Initialize
{
    int enemyStartingMoney;
    int playerStartingMoney;

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);

    RegisterGoal(sendToBase, "translateGoalSend100000", 0);
    EnableGoal(sendToBase, true);

    if(GetDifficultyLevel() == 0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        playerStartingMoney = 30000;
        enemyStartingMoney = 20000;
    }
    if(GetDifficultyLevel() == 1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        playerStartingMoney = 20000;
        enemyStartingMoney = 30000;
    }
    if(GetDifficultyLevel() == 2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        playerStartingMoney = 10000;
        enemyStartingMoney = 50000;
    }

    p_Player.SetMoney(playerStartingMoney);
    p_Enemy1.SetMoney(enemyStartingMoney);
    p_Enemy2.SetMoney(enemyStartingMoney);

    p_Player.EnableAIFeatures(aiEnabled, false);
    p_Enemy1.EnableAIFeatures(aiRush, false);
    p_Enemy2.EnableAIFeatures(aiRush, false);

    p_Enemy1.EnableResearch("RES_UCS_WASR1", true);
    p_Enemy1.EnableResearch("RES_UCS_WCH2", true);
    p_Enemy1.EnableResearch("RES_UCS_WSP1", true);
    p_Enemy1.EnableResearch("RES_UCS_WSG1", true);
    p_Enemy1.EnableResearch("RES_MCH2", true);
    p_Enemy1.EnableResearch("RES_MSR2", true);
    p_Enemy1.EnableResearch("RES_UCS_MG2", true);
    p_Enemy1.EnableResearch("RES_UCS_UML1", true);
    p_Enemy1.EnableResearch("RES_UCS_UHL1", true);
    p_Enemy1.EnableResearch("RES_UCS_BMD", true);
    p_Enemy1.EnableResearch("RES_UCS_BHD", true);

    p_Enemy2.EnableResearch("RES_LC_WSR2", true);
    p_Enemy2.EnableResearch("RES_LC_WST1", true);
    p_Enemy2.EnableResearch("RES_LC_WMR1", true);
    p_Enemy2.EnableResearch("RES_MSR2", true);
    p_Enemy2.EnableResearch("RES_LC_UCR1", true);
    p_Enemy2.EnableResearch("RES_LC_UBO1", true);
    p_Enemy2.EnableResearch("RES_LC_REG1", true);
    p_Enemy2.EnableResearch("RES_LC_SGEN", true);
    p_Enemy2.EnableResearch("RES_LC_SHR1", true);
    p_Enemy2.EnableResearch("RES_LC_BWC", true);

    p_Player.EnableResearch("RES_ED_WMR1", true);
    p_Player.EnableResearch("RES_ED_UHW1", true);
    p_Player.EnableResearch("RES_MSR2", true);
    p_Player.EnableResearch("RES_ED_UA22", true);
    p_Player.EnableResearch("RES_ED_BHD", true);
}

SetTimer(0,100);
CallCamera();
```

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6.0,20,0);

```

    bShowFailed=true;
    bCheckEndMission=false;
    p_Player.SetMilitaryUnitsLimit(20000);
    return ShowBriefing,150://15 sec
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing234",nNeededResources);
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    EnableNextMission(2,true);
    return ShowBriefing,200;
}

//-----
state Mining
{
    if(GetGoalState(sendToBase) == goalAchieved && p_Player.GetMoneySentToBase() >= nNeededResources)
    {
        SetGoalState(sendToBase, goalAchieved);
        AddBriefing("translateAccomplished234");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}

//-----
state Nothing
{
    return Nothing, 500;
}

event Timer0() //wolany co 100 cykli< ustawiione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase, "translateGoalSend100000", p_Player.GetMoneySentToBase);
}

if(bShowFailed)
{
    if((ResourcesLeftInMoney() + p_Player.GetMoney() + p_Player.GetMoneySentToBase) < nNeededResources)
    {
        bShowFailed=false;
        SetGoalState(sendToBase, goalFailed);
        AddBriefing("translateFailed234a");
        EnableEndMissionButton(true);
        return Nothing;
    }
}

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if((p_Player.GetNumberOfWorkUnits() && lp_Player.GetNumberOfBuildings()) && AddBriefing("translateFailed234b"));
    EndMission(false);
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
```

Mission235.ec
mission "translateMission235"
{/Find UFO
consts

```

    {
        findUFO = 0;
        findStartingArtifact1 = 1;
        findStartingArtifact2 = 2;
        findStartingArtifact3 = 3;
        findStartingArtifact4 = 4;
        backToLandingZone = 5;
    }

    player p_Enemy;
    player p_Player;
    player p_Neutral;
    unitex p_UFO;

    state Initialize;
    state ShowBriefing;
    state Searching;
    state SearchingArtefacts;
    state Evacuate;
}

//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(findUFO,"translateGoal235a");
    RegisterGoal(findStartingArtifact1,"translateGoal235b");
    RegisterGoal(findStartingArtifact2,"translateGoal235c");
    RegisterGoal(findStartingArtifact3,"translateGoal235d");
    RegisterGoal(findStartingArtifact4,"translateGoal235e");
    RegisterGoal(backToLandingZone,"translateGoal235f");

    EnableGoal(findUFO,true);

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(1);
    p_Neutral = GetPlayer(6);

    p_Neutral.EnableStatistics(false);

    p_UFO = GetUnit(GetPointX(0),GetPointY(0),1);

    p_Player.SetMoney(0);
    p_Enemy.SetMoney(40000);

    p_Player.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Player);

    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);

    p_Neutral.SetNeutral(p_Enemy);
    p_Enemy.SetNeutral(p_Neutral);

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Neutral.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
        p_Enemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        p_Enemy.LoadScript("single\singleHard");

    SetTimer(0,100);
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);

    return ShowBriefing,100;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing235a");
    return Searching,100;
}

//-----
state Searching
{
    if(p_Player.IsPointLocated(GetPointX(0),GetPointY(0),1))
    {
        SetGoalState(findUFO,goalAchieved);
        EnableGoal(findStartingArtifact1,true);
        EnableGoal(findStartingArtifact2,true);
        EnableGoal(findStartingArtifact3,true);
        EnableGoal(findStartingArtifact4,true);
    }

    CreateArtefact("NEASPECIAL2",GetPointX(1),GetPointY(1),1,0,artefactSpecialAI0ther);
    CreateArtefact("NEASPECIAL2",GetPointX(2),GetPointY(2),1,1,artefactSpecialAI0ther);
    CreateArtefact("NEASPECIAL2",GetPointX(3),GetPointY(3),1,2,artefactSpecialAI0ther);
    CreateArtefact("NEASPECIAL2",GetPointX(4),GetPointY(4),1,3,artefactSpecialAI0ther);

    CallCamera();
    p_Player.LookAt(GetPointX(0),GetPointY(0),6,0,20,1);
    AddBriefing("translateBriefing235b");
    return SearchingArtefacts,200;
}
return Searching,200;
}

//-----
state SearchingArtefacts
{
    if(GetGoalState(findStartingArtifact1)==goalAchieved &&
       GetGoalState(findStartingArtifact2)==goalAchieved &&
       GetGoalState(findStartingArtifact3)==goalAchieved &&
       GetGoalState(findStartingArtifact4)==goalAchieved)
    {
        p_Neutral.GiveAllUnitsTo(p_Player);
        EnableGoal(backToLandingZone,true);
        AddBriefing("translateBriefing235c"); //back to lz
        return Evacuate,50;
    }
    return SearchingArtefacts,200;
}

//-----
state Evacuate
{
    if(GetGoalState(backToLandingZone)!=goalAchieved &&
       Distance(p_UFO.GetLocationX(),p_UFO.GetLocationY(),p_Player.GetStartingPointX(),p_Player.GetStartingPointY()) < 5)
    {
        CallCamera();
    }

    p_Player.LookAt(p_UFO.GetLocationX(),p_UFO.GetLocationY(),6,0,20,p_UFO.GetLocationZ());
    SetGoalState(backToLandingZone,goalAchieved);
    AddBriefing("translateAccomplished235");
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Neutral);

    EnableEndMissionButton(true);
}
return Evacuate,50;
}

//-----
event Timer0() //wolany co 100 cykl< ustwione funkcja SetTimer w state
Initialize
{
    if(!p_UFO.IsLive() && GetGoalState(backToLandingZone)== goalFailed)
    {
        SetGoalState(backToLandingZone,goalFailed);
        AddBriefing("translateFailed235a");
        p_Neutral.SetEnemy(p_Player);
        p_Player.SetEnemy(p_Neutral);
        p_Neutral.SetEnemy(p_Enemy);
        p_Enemy.SetEnemy(p_Neutral);
        EnableEndMissionButton(true);
    }

    if(!p_Player.GetNumberOfBuildings() &&
       GetGoalState(backToLandingZone)== goalFailed)
    {
        SetGoalState(backToLandingZone,goalFailed);
        AddBriefing("translateFailed235b");
        p_Neutral.SetEnemy(p_Player);
        p_Player.SetEnemy(p_Neutral);
    }
}

```

```

    p_Neutral.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Neutral);
    EndMission(false);
}

//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(alD==0)
        SetGoalState(findStartingArtifact1, goalAchieved);
    if(alD==1)
        SetGoalState(findStartingArtifact2, goalAchieved);
    if(alD==2)
        SetGoalState(findStartingArtifact3, goalAchieved);
    if(alD==3)
        SetGoalState(findStartingArtifact4, goalAchieved);
    return true; //usuwa sie
}
}

```

Mission241.ec

mission "translateMission241"

```

{//Panama
consts
{
    sendToBase = 0;
    nNeededResources=50000;
}

```

```

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Player;

```

```

int bShowFailed;
int bCheckEndMission;
int bStartOffense;
int bBlowUpTheBridge;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;

```

//-----

state Initialize

```

{
    int enemyStartingMoney;
    int playerStartingMoney;
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(4);
    p_Enemy3 = GetPlayer(5);
}
```

```

RegisterGoal(sendToBase,"translateGoalSend50000",0);
EnableGoal(sendToBase,true);

```

```

if(GetDifficultyLevel()==0)
{

```

```

    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
    p_Enemy3.LoadScript("single\singleEasy");
    playerStartingMoney=30000;
    enemyStartingMoney=20000;
}

```

```

if(GetDifficultyLevel()==1)
{

```

```

    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    p_Enemy3.LoadScript("single\singleEasy");
    playerStartingMoney=20000;
    enemyStartingMoney=30000;
}

```

```

if(GetDifficultyLevel()==2)
{

```

```

    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
}
```

```

    p_Enemy3.LoadScript("single\singleMedium");
    playerStartingMoney=10000;
    enemyStartingMoney=50000;
}

p_Player.SetMoney(playerStartingMoney);
p_Enemy1.SetMoney(enemyStartingMoney);
p_Enemy2.SetMoney(enemyStartingMoney);
p_Enemy3.SetMoney(enemyStartingMoney);

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiRush,false);
p_Enemy2.EnableAIFeatures(aiRush,false);
p_Enemy3.EnableAIFeatures(aiRush,false);

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);

p_Enemy1.EnableResearch("RES_UCS_WHP1",true);
p_Enemy1.EnableResearch("RES_UCS_UBS1",true);
p_Enemy1.EnableResearch("RES_UCS_USM1",true);
p_Enemy1.EnableResearch("RES_UCS_UHM1",true);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_ED_WHL1",true);
p_Player.EnableResearch("RES_MMR2",true);
p_Player.EnableResearch("RES_ED_UHT1",true);
p_Player.EnableResearch("RES_ED_UHS1",true);

bStartOffense=true;
bBlowUpTheBridge=true;

SetTimer(0,100);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),0,6,0,20,
0);
bShowFailed=true;
bCheckEndMission=false;
return ShowBriefing,150;//15 sec
}
//-----
state ShowBriefing
{
ShowArea(4,p_Enemy2.GetStartingPointX()),p_Enemy2.GetStartingPointY(),0,2);
AddBriefing("translateBriefing241",nNeededResources);
return Mining,200;
}
//-----
state Mining
{
    if(bBlowUpTheBridge &&
p_Player.IsPointLocated((GetPointX(1),GetPointY(1),0))
    {
        bBlowUpTheBridge=false;
        // IFFmask.x,y,z,range
        KillArea(65535,GetPointX(0),GetPointY(0),0,4,);
    }
    if(bStartOffense && p_Player.GetMoneySentToBase()>=10000)
    {
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy2.EnableAIFeatures(aiControlOffense,true);
        bStartOffense=false;
    }
    if(GetGoalState(sendToBase)==goalAchieved &&
p_Player.GetMoneySentToBase()>=nNeededResources)
    {
        SetGoalState(sendToBase, goalAchieved);
        AddBriefing("translateAccomplished241");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawnione funkcja SetTimer w state

```

```

Initialize
{
    RegisterGoal(sendToBase,"translateGoalSend50000",p_Player.GetMoneySentToBase());
    if(bShowFailed)
    {
        if((ResourcesLeftInMoney() + p_Player.GetMoney() + p_Player.GetMoneySentToBase()) < nNeededResources)
        {
            bShowFailed=false;
            SetGoalState(sendToBase, goalFailed);
            AddBriefing("translateFailed241a");
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if((p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings()))
        {
            AddBriefing("translateFailed241b");
            EndMission(false);
        }
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----



Mission242.ec
mission "translateMission242"
{/Indie
consts
{
    sendToBase = 0;
    nNeededResources=50000;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Player;

int bShowFailed;
int bCheckEndMission;
int bStartOffense;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;

//-----
state Initialize
{
    int enemyStartingMoney;
    int playerStartingMoney;

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);
    p_Enemy3 = GetPlayer(4);

    RegisterGoal(sendToBase,"translateGoalSend50000",0);
    EnableGoal(sendToBase,true);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy3.LoadScript("single\singleEasy");
        playerStartingMoney=30000;
        enemyStartingMoney=20000;
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy3.LoadScript("single\singleEasy");
        playerStartingMoney=20000;
        enemyStartingMoney=30000;
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy3.LoadScript("single\singleMedium");
        playerStartingMoney=10000;
        enemyStartingMoney=50000;
    }
}

p_Player.SetMoney(playerStartingMoney);
p_Enemy1.SetMoney(enemyStartingMoney);
p_Enemy2.SetMoney(enemyStartingMoney);
p_Enemy3.SetMoney(enemyStartingMoney);

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiRush,false);
p_Enemy2.EnableAIFeatures(aiRush,false);
p_Enemy3.EnableAIFeatures(aiRush,false);

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);

p_Enemy1.EnableResearch("RES_UCS_WMR1",true);
p_Enemy1.EnableResearch("RES_UCS_UM1",true);

p_Enemy2.EnableResearch("RES_LC_WHS1",true);
p_Enemy2.EnableResearch("RES_MMR2",true);
p_Enemy2.EnableResearch("RES_LC_UCU1",true);
p_Enemy2.EnableResearch("RES_LC_SO1",true);

p_Enemy3.CopyResearches(p_Enemy1);

p_Player.EnableResearch("RES_MSR2",true);
p_Player.EnableResearch("RES_ED_UM1",true);
p_Player.EnableResearch("RES_ED_RepHand2",true);

bStartOffense=true;
SetTimer(0,100);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
bShowFailed=true;
bCheckEndMission=false;
return ShowBriefing,150://15 sec
}
//-----
state ShowBriefing
{
    ShowArea(4,p_Enemy3.GetStartingPointX(),p_Enemy3.GetStartingPointY(),0,2);
    AddBriefing("translateBriefing242",nNeededResources);
    return Mining,200;
}
//-----
state Mining
{
    if(bStartOffense && p_Player.GetMoneySentToBase()>=10000)
    {
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy2.EnableAIFeatures(aiControlOffense,true);
        bStartOffense=false;
    }
    if(GetGoalState(sendToBase)!=goalAchieved && p_Player.GetMoneySentToBase()>=nNeededResources)
    {
        SetGoalState(sendToBase, goalAchieved);
        AddBriefing("translateAccomplished242");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}

```

```

//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase,"translateGoalSend50000",p_Player.GetMoneySentToBase());
e());

    if(bShowFailed)
    {
        if((ResourcesLeftInMoney() + p_Player.GetMoney() + p_Player.GetMoneySentToBase()) < nNeededResources)
        {
            bShowFailed=false;
            SetGoalState(sendToBase, goalFailed);
            AddBriefing("translateFailed242a");
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed242b");
            EndMission(false);
        }
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

Mission243.ec
mission "translateMission243"
{/Madagascar
consts
{
    recoverArtefact=0;
}

player p_Enemy1;
player p_Enemy2;
player p_GoalEnemy;
player p_Player;

state Initialize;
state ShowBriefing;
state RecoverArtefact;
state Evacuate;

//-----
state Initialize
{
    RegisterGoal(recoverArtefact,"translateGoal243");
    EnableGoal(recoverArtefact,true);

    //      name      x,y,z, nr,typ
CreateArtefact("NEASPECIAL2",GetPointX(0),GetPointY(0),1,0,artefactSpecialAI0ther);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);

    p_Player.SetMoney(20000);
    p_Enemy1.SetMoney(40000);
    p_Enemy2.SetMoney(40000);
}

p_Player.EnableAIFeatures(aiEnabled,false);

if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\\singleEasy");
    p_Enemy2.LoadScript("single\\singleEasy");
}
if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\\singleMedium");
    p_Enemy2.LoadScript("single\\singleMedium");
}
if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\\singleHard");
    p_Enemy2.LoadScript("single\\singleHard");
}

p_Enemy1.SetPointToAssemble(0,GetPointX(1),GetPointY(1,0));
p_Enemy1.SetPointToAssemble(1,GetPointX(2),GetPointY(2,0));

p_Player.EnableResearch("RES_ED_WHG1",true);
p_Player.EnableResearch("RES_MSR2",true);
p_Player.EnableResearch("RES_ED_UHT1",true);

p_Enemy1.EnableResearch("RES_UCS_WHG1",true);
p_Enemy1.EnableResearch("RES_UCS_SGEN",true);
p_Enemy1.EnableResearch("RES_UCS_UMI1",true);

p_Enemy2.EnableResearch("RES_LC_WHS1",true);
p_Enemy2.EnableResearch("RES_MMR2",true);
p_Enemy2.EnableResearch("RES_LC_UCU1",true);
p_Enemy2.EnableResearch("RES_LC_SOBI",true);

SetTimer(0,200);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing243");
    EnableNextMission(0,true);
    return RecoverArtefact,100;
}

state RecoverArtefact
{
    if(GetGoalState(recoverArtefact)==goalAchieved)
    {
        EnableEndMissionButton(true);
        return Evacuate,500;
    }
    return RecoverArtefact,100;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event Timer0()
{
    if((lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings()))
    {
        AddBriefing("translateFailed243");
        EndMission(false);
    }
}
//-----
event Artefact(int idD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    SetGoalState(recoverArtefact,goalAchieved);
    p_Player.EnableResearch("RES_ED_SGen",true);
    AddBriefing("translateAccomplished243");
    return true; //usuwa sie
}
}

```

Mission244.ec

```
mission "translateMission244"
//Evacuate Ion cannon prototype
consts
{
    evacuatePrototype = 0;
}

player p_Enemy1;
player p_Enemy2;
player p_Neutral;
player p_Player;
unitex p_Protoype;

int n_EvacuatePointX;
int n_EvacuatePointY;

state Initialize;
state ShowVideoState;
state ShowBriefing;
state EvacuatePrototype;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    tmpPlayer.EnableAIFeatures(aiEnabled,false);

    tmpPlayer = GetPlayer(4);
    tmpPlayer.EnableStatistics(false);
    tmpPlayer.EnableAIFeatures(aiEnabled,false);

    RegisterGoal(evacuatePrototype,"translateGoal244");
    EnableGoal(evacuatePrototype,true);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);
    p_Neutral = GetPlayer(8);

    p_Protoype = GetUnit(GetPointX(0),GetPointY(0),1);
    p_Neutral.EnableStatistics(false);

    p_Player.SetMoney(0);
    p_Enemy1.SetMoney(30000);
    p_Enemy2.SetMoney(0);

    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);

    if(GetDifficultyLevel()==0)
        p_Enemy1.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy1.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        p_Enemy1.LoadScript("single\singleHard");

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiControlOffense,false);
    p_Enemy1.EnableAIFeatures(aiControlDefense,false);
    p_Enemy2.EnableAIFeatures(aiEnabled,false);
    p_Neutral.EnableAIFeatures(aiEnabled,false);

    p_Player.EnableResearch("RES_ED_WHC1",true);
    p_Player.EnableResearch("RES_ED_WHL1",true);
    p_Player.EnableResearch("RES_MM2",true);
    p_Player.EnableResearch("RES_MSR2",true);
    p_Player.EnableResearch("RES_ED_UHT1",true);
    p_Player.EnableResearch("RES_ED_UHS1",true);
    p_Player.EnableResearch("RES_ED_JM1",true);
    p_Player.EnableResearch("RES_ED_JMW1",true);
    p_Player.EnableResearch("RES_ED_ReHand2",true);

    p_Enemy1.EnableResearch("RES_UCS_WHP1",true);
    p_Enemy1.EnableResearch("RES_UCS_WMR1",true);
    p_Enemy1.EnableResearch("RES_UCS_WHG1",true);
    p_Enemy1.EnableResearch("RES_UCS_SGEN",true);
    p_Enemy1.EnableResearch("RES_UCS_UBS1",true);
    p_Enemy1.EnableResearch("RES_UCS_USM1",true);
    p_Enemy1.EnableResearch("RES_UCS_UAH1",true);

    n_EvacuatePointX = p_Neutral.GetStartingPointX();
    n_EvacuatePointY = p_Neutral.GetStartingPointY();

    SetTimer(0,100);
    CallCamera();

    p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
    1);
    return ShowVideoState,100;
}
//-----
state ShowVideoState
{
    ShowVideo("CS212");
    return ShowBriefing,20;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    AddBriefing("translateBriefing244");
    return EvacuatePrototype,100;
}
//-----
state EvacuatePrototype
{
    if(Distance(p_Protoype.GetLocationX(),p_Protoype.GetLocationY(),n_EvacuatePointX,n_EvacuatePointY) < 5)
    {
        SetGoalState(evacuatePrototype, goalAchieved);
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy1.EnableAIFeatures(aiControlDefense,true);

        p_Neutral.GiveAllUnitsTo(p_Player);
        p_Neutral.GiveAllBuildingsTo(p_Player);
        p_Player.EnableResearch("RES_ED_WS1",true);
        AddBriefing("translateAccomplished244");
        p_Neutral.SetEnemy(p_Player);
        p_Player.SetEnemy(p_Neutral);
        EnableEndMissionButton(true);
        return Final,100;
    }
    return EvacuatePrototype,50;
}
//-----
state Final
{
    return Final,500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initial
{
    if(!p_Protoype.IsLive())
    {
        AddBriefing("translateFailed244");
        p_Neutral.SetEnemy(p_Player);
        p_Player.SetEnemy(p_Neutral);
        EndMission(false);
    }
}
}



## Mission251.ec



```
mission "translateMission251"
//Escort convoy
consts
{
 escortConvoy = 0;
 destroyUCSBase = 1;
}

player p_Enemy1;
player p_Enemy2;
player p_Neutral;
player p_Player;
unitex p_ConvoyCraft1;
unitex p_ConvoyCraft2;
unitex p_ConvoyCraft3;
unitex p_ConvoyCraft4;
```



55


```

```

unitex p_ConvoyEscort1;
unitex p_ConvoyEscort2;
int nWayPoint;

state Initialize;
state ShowBriefing;
state OnTheWay;
state Fight;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(escortConvoy,"translateGoal251a");
    RegisterGoal(destroyUCSBase,"translateGoal251b");
    EnableGoal(escortConvoy,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(4);
    p_Neutral = GetPlayer(8);
    //----- AI -----
    p_Neutral.EnableStatistics(false);
    p_Enemy2.EnableStatistics(false);
    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);

    p_Neutral.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_Neutral);
    p_Neutral.ChooseEnemy(p_Enemy1);

    if(GetDifficultyLevel()==0)
        p_Enemy1.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy1.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        p_Enemy1.LoadScript("single\singleHard");

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiControlOffense,false);
    p_Enemy1.EnableAIFeatures(aiControlDefense,false);
    p_Enemy2.EnableAIFeatures(aiEnabled,false);
    p_Neutral.EnableAIFeatures(aiEnabled,false);

    p_NeutralSetName("translateAppName251");
    //----- Money -----
    p_Player.SetMoney(0);
    p_Enemy1.SetMoney(30000);
    p_Enemy2.SetMoney(0);
    //----- Researches -----
    p_Player.EnableResearch("RES_ED_MHC2",true);
    p_Player.EnableResearch("RES_ED_SCR",true);
    p_Enemy1.EnableResearch("RES_UCS_SHD",true);

    p_Enemy2.CopyResearches(p_Enemy1);
    //----- Buildings -----
    //----- Units -----
    p_ConvoyCraft1 = GetUnit(GetPointX(0),GetPointY(0,0));
    p_ConvoyCraft2 = GetUnit(GetPointX(1),GetPointY(1,0));
    p_ConvoyCraft3 = GetUnit(GetPointX(2),GetPointY(2,0));
    p_ConvoyCraft4 = GetUnit(GetPointX(3),GetPointY(3,0));
    p_ConvoyEscort1 = GetUnit(GetPointX(4),GetPointY(4,0));
    p_ConvoyEscort2 = GetUnit(GetPointX(5),GetPointY(5,0));
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,200);
    //----- Variables -----
    nWayPoint=5;
    //----- Camera -----
    CallCamera();

    p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    ShowArea(4,GetPointX(6),GetPointY(6),0,2);
    ShowArea(4,GetPointX(7),GetPointY(7),0,2);
    ShowArea(4,GetPointX(8),GetPointY(8),0,2);
    ShowArea(4,GetPointX(9),GetPointY(9),0,2);
    ShowArea(4,GetPointX(10),GetPointY(10),0,2);
    EnableNextMission(0,true);
    AddBriefing("translateBriefing251a");
    return OnTheWay,300;
}
//-----
state OnTheWay
{
    if(nWayPoint>9)
    {
        nWayPoint=10;
    }
    if(p_ConvoyCraft1.DistanceTo(GetPointX(10),GetPointY(10)) < 5 &&
       p_ConvoyCraft2.DistanceTo(GetPointX(10),GetPointY(10)) < 5 &&
       p_ConvoyCraft3.DistanceTo(GetPointX(10),GetPointY(10)) < 5 &&
       p_ConvoyCraft4.DistanceTo(GetPointX(10),GetPointY(10)) < 5 )
    {
        p_Neutral.GiveAllUnitsTo(p_Player);
        p_Neutral.GiveAllBuildingsTo(p_Player);
        SetGoalState(escortConvoy,goalAchieved);
        EnableGoal(destroyUCSBase,true);
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Enemy1.EnableAIFeatures(aiControlDefense,true);
        p_Player.SetMoney(20000);
        AddBriefing("translateBriefing251b");
        return Fight,100;
    }
    if(p_ConvoyCraft1.DistanceTo(GetPointX(nWayPoint),GetPointY(nWayPoint)) < 5 &&
       p_ConvoyCraft2.DistanceTo(GetPointX(nWayPoint),GetPointY(nWayPoint)) < 5 &&
       p_ConvoyCraft3.DistanceTo(GetPointX(nWayPoint),GetPointY(nWayPoint)) < 5 &&
       p_ConvoyCraft4.DistanceTo(GetPointX(nWayPoint),GetPointY(nWayPoint)) < 5 )
    {
        nWayPoint=nWayPoint+1;
        if(nWayPoint==9)
        {
            p_Player.SetMoney(2000);
            AddBriefing("translateBriefing251c");
        }
    }
}
p_ConvoyCraft1.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
p_ConvoyCraft2.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
p_ConvoyCraft3.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
p_ConvoyCraft4.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
p_ConvoyEscort1.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
p_ConvoyEscort2.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
return OnTheWay,300;
}
//-----
state Fight
{
    if((p_Player.GetNumberOfUnits() &&p_Player.GetNumberOfBuildings())
       AddBriefing("translateFailed251b");
       EndMission(false);
    )
    if(
        (p_Enemy1.GetNumberOfUnits()<6) &&
        p_Enemy1.GetNumberOfBuildings())
    {
        SetGoalState(destroyUCSBase,goalAchieved);
        AddBriefing("translateAccomplished251");
        EnableEndMissionButton(true);
        return Final,500;
    }
    return Fight,200;
}

```

```

        }

        //-----
        state Final
        {
            return Final,500;
        }

        //-----
        event Timer0() //wolany co 200 cykli< ustawione funkcja SetTimer w state
        Initialize
        {
            if(GetGoalState(escortConvoy)!=goalAchieved &&
            (lp_ConvoyCraft1.lsLive() || 
            lp_ConvoyCraft2.lsLive() || 
            lp_ConvoyCraft3.lsLive() || 
            lp_ConvoyCraft4.lsLive()))
            {
                AddBriefing("translateFailed251a");
                EndMission(false);
            }
        }

        event EndMission()
        {
            p_Neutral.SetEnemy(p_Player);
            p_Player.SetEnemy(p_Neutral);
            p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
        }
    }
}

```

Mission252.ec

```

mission "translateMission252"
{//Australia -RedRock destroy UCS base
const
{
    destroyEnemyBase = 0;
}
player p_Enemy;
player p_Player;

int bCheckEndMission;

state Initialize;
state ShowBriefing;
state Nothing;

//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(destroyEnemyBase,"translateGoalDestroyEnemyBase");
    EnableGoal(destroyEnemyBase,true);

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(1);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\\singleEasy");
        p_Enemy.EnableAIFeatures(aiUpgradeCannons,false);
        p_Player.SetMoney(30000);
        p_Enemy.SetMoney(20000);
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy.LoadScript("single\\singleMedium");
        p_Player.SetMoney(20000);
        p_Enemy.SetMoney(30000);
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\\singleHard");
    }
}

p_Enemy.CreateDefaultUnit(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),0);
p_Player.SetMoney(15000);
p_Enemy.SetMoney(50000);
}

```

```

        }

        p_Player.EnableAIFeatures(aiEnabled,false);

        p_Player.EnableResearch("RES_ED_AMR1",true);
        p_Player.EnableResearch("RES_ED_UA41",true);
        p_Player.EnableResearch("RES_ED_SCR",true);

        p_Enemy.EnableResearch("RES_UCS_BOMBER21",true);
        p_Enemy.EnableResearch("RES_UCS_SHD",true);

        SetTimer(0,100);

        bCheckEndMission=false;
        CallCamera();

        p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
        return ShowBriefing,100;
    }
state ShowBriefing
{
    EnableNextMission(true);
    AddBriefing("translateBriefing252");
    return Nothing,100;
}

//-----
state Nothing
{
    return Nothing,100;
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed252");
            EndMission(false);
        }
        if(GetGoalState(destroyEnemyBase)==goalAchieved &&
        (p_Enemy.GetNumberOfUnits()<6) &&
        lp_Player.GetNumberOfBuildings())
        {
            SetGoalState(destroyEnemyBase,goalAchieved);
            AddBriefing("translateAccomplished252");
            EnableEndMissionButton(true);
        }
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

```

Mission253.ec

```

mission "translateMission253"
{//Egypt - destroy UCS base (and LC)
const
{
    destroyUCSBase = 0;
    destroyLCBase = 1;
}
player p_EnemyUCS;
player p_EnemyLC;
player p_Player;

int bCheckEndMission;
int bLCUnitDestroyed;
int bUCSUnitDestroyed;

state Initialize;
state ShowBriefing;
state Searching;
}

```

```

state LCAlied;
state UCSFight;
state FinalFight;
state Nothing;

//-----
state Initialize
{
    RegisterGoal(destroyUCSBase,"translateGoal253a");
    RegisterGoal(destroyLCBase,"translateGoal253b");
    EnableGoal(destroyUCSBase,true);
}

p_Player = GetPlayer(2);
p_EnemyUCS = GetPlayer(1);
p_EnemyLC = GetPlayer(3);

if(GetDifficultyLevel()==0)
{
    p_EnemyUCS.LoadScript("single\singleEasy");
    p_EnemyUCS.EnableAIFeatures(aiUpgradeCannons,false);
    p_EnemyLC.LoadScript("single\singleEasy");
    p_EnemyLC.EnableAIFeatures(aiUpgradeCannons,false);
    p_Player.SetMoney(30000);
    p_EnemyUCS.SetMoney(20000);
    p_EnemyLC.SetMoney(20000);
}
if(GetDifficultyLevel()==1)
{
    p_EnemyUCS.LoadScript("single\singleMedium");
    p_EnemyLC.LoadScript("single\singleMedium");
    p_Player.SetMoney(20000);
    p_EnemyUCS.SetMoney(40000);
    p_EnemyLC.SetMoney(40000);
}
if(GetDifficultyLevel()==2)
{
    p_EnemyUCS.LoadScript("single\singleHard");
    p_EnemyLC.LoadScript("single\singleHard");
    p_Player.SetMoney(15000);
    p_EnemyUCS.SetMoney(50000);
    p_EnemyLC.SetMoney(50000);
}

p_EnemyLCSetName("Alia Tiosh");
p_Player.EnableAIFeatures(aiEnabled,false);
p_EnemyLC.EnableAIFeatures(aiControlOffense,false);
p_EnemyUCS.EnableAIFeatures(aiControlOffense,false);
p_Player.EnableResearch("RES_ED_WH2",true);
p_EnemyUCS.EnableResearch("RES_UCS邬BL1",true);
p_EnemyUCS.EnableResearch("RES_UCS_BOMBER21",true);
p_EnemyLC.EnableResearch("RES_LC_WHL1",true);
p_EnemyLC.EnableResearch("RES_LC_WHS1",true);
p_EnemyLC.EnableResearch("RES_LC_WARTILLERY",true);
p_EnemyLC.EnableResearch("RES_MMZ2",true);

bLCunitDestroyed = false;
bUCSunitDestroyed = false;

SetTimer(0,100);

bCheckEndMission=false;
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
ShowArea(4,GetPointX(0),GetPointY(0),0.4);
p_Player.SetMilitaryUnitsLimit(30000);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(2,true);
    AddBriefing("translateBriefing253a");
    return Searching,100;
}
//-----
state Searching
{
    return Nothing,512;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
if(bLCunitDestroyed)
{
    bLCunitDestroyed=false;
    AddBriefing("translateBriefing253e");
    EnableGoal(destroyLCBase,true);
    p_EnemyLC.EnableAIFeatures(aiControlOffense,true);
    p_EnemyUCS.EnableAIFeatures(aiControlOffense,true);
    return FinalFight,500;
}
if(bUCSunitDestroyed)
{
    bUCSunitDestroyed=false;
    p_EnemyLC.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_EnemyLC);
    p_EnemyLC.ChooseEnemy(p_EnemyUCS);
    p_EnemyUCS.SetEnemy(p_EnemyLC);
    p_EnemyUCS.EnableAIFeatures(aiControlOffense,true);
    p_EnemyUCS.EnableAIFeatures(aiControlOffense,true);
    AddBriefing("translateBriefing253b");
    return LCAlied,200;
}
return Searching,100;
}
//-----
state LCAlied
{
    if(bLCunitDestroyed)
    {
        bLCunitDestroyed=false;
        AddBriefing("translateBriefing253c");
        EnableGoal(destroyLCBase,true);
        p_EnemyLC.SetEnemy(p_Player);
        p_Player.SetEnemy(p_EnemyLC);
        p_EnemyLC.ChooseEnemy(p_Player);
        return FinalFight,500;
    }
    if(GetGoalState(destroyUCSBase)==goalAchieved)
    {
        AddBriefing("translateBriefing253d");
        bLCunitDestroyed = true;
        return LCAlied,400;
    }
    if(GetGoalState(destroyLCBase)==goalAchieved)
    {
        return UCSFight,200;
    }
    return LCAlied,200;
}
//-----
state UCSFight
{
    if(GetGoalState(destroyUCSBase)==goalAchieved)
    {
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        AddBriefing("translateAccomplished253b");
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    return UCSFight,200;
}
//-----
state FinalFight
{
    if(GetGoalState(destroyUCSBase)==goalAchieved &
    GetGoalState(destroyLCBase)==goalAchieved)
    {
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        AddBriefing("translateAccomplished253a");
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    return FinalFight,100;
}
//-----
state Nothing
{
    return Nothing,512;
}
//-----
```

```

Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(!p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed254");
            EndMission(false);
        }
        if(GetGoalState(destroyUCSBase)==goalAchieved &&
           lp_EnemyUCS.GetNumberOfBuildings() &&
           p_EnemyUCS.GetNumberOfUnits(<6)
        {
            SetGoalState(destroyUCSBase,goalAchieved);
        }
        if(GetGoalState(destroyLCBase)!=goalAchieved &&
           lp_EnemyLC.GetNumberOfBuildings() &&
           p_EnemyLC.GetNumberOfUnits(<6)
        {
            SetGoalState(destroyLCBase,goalAchieved);
        }
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    unit pAttacker;
    bCheckEndMission=true;
    pAttacker = u_Unit.GetAttacker();
    if(pAttacker==null) return;
    if(pAttacker.GetIFFNumber()!=p_Player.GetIFFNumber()) return;
    if(u_Unit.GetIFFNumber()==p_EnemyLC.GetIFFNumber())
    {
        bLCunitDestroyed=true;
        return;
    }
    if(u_Unit.GetIFFNumber()==p_EnemyUCS.GetIFFNumber())
        bUCunitDestroyed=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    unit uAttacker;
    bCheckEndMission=true;
    uAttacker = u_Unit.GetAttacker();
    if(uAttacker==null) return;
    if(uAttacker.GetIFFNumber()!=p_Player.GetIFFNumber())
    {
        return;
    }
    if(u_Unit.GetIFFNumber()==p_EnemyLC.GetIFFNumber())
    {
        bLCunitDestroyed=true;
        return;
    }
    if(u_Unit.GetIFFNumber()==p_EnemyUCS.GetIFFNumber())
        bUCunitDestroyed=true;
}
//-----
event EndMission()
{
    p_EnemyLC.SetEnemy(p_Player);
    p_Player.SetEnemy(p_EnemyLC);
    p_EnemyLC.EnableAIFeatures(aiRejectAlliance,true);
}
}

state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(destroyEnemyBase,"translateGoalDestroyEnemyBase");
    EnableGoal(destroyEnemyBase,true);

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(1);

    p_Enemy.EnableResearch("RES_UCS_UBL1",true);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\singleEasy");
        p_Enemy.EnableAIFeatures(aiUpgradeCannons,false);
        p_Player.SetMoney(30000);
        p_Enemy.SetMoney(20000);
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy.LoadScript("single\singleMedium");
        p_Player.SetMoney(20000);
        p_Enemy.SetMoney(30000);
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\singleHard");
    }
}
p_Enemy.CreateDefaultUnit(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),0);
    p_Player.SetMoney(15000);
    p_Enemy.SetMoney(50000);
}
p_Player.EnableAIFeatures(aiEnabled,false);

SetTimer(0,100);
bCheckEndMission=false;
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
    p_Player.SetMilitaryUnitsLimit(30000);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    AddBriefing("translateBriefing254");
    return Nothing,100;
}
//-----
state Nothing
{
    return Nothing,100;
}
//-----
event Timer0() //wolany co 100 cykl< ustawiione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed254");
            EndMission(false);
        }
        if(GetGoalState(destroyEnemyBase)==goalAchieved &&
           (p_Enemy.GetNumberOfUnits(<6) &&
           lp_Enemy.GetNumberOfBuildings()))
        {
            SetGoalState(destroyEnemyBase,goalAchieved);
            AddBriefing("translateAccomplished254");
            EnableEndMissionButton(true);
        }
    }
}

```

Mission254.ec

```

mission "translateMission254"
{[Australia - Wyspy destroy UCS base
consts
{
    destroyEnemyBase = 0;
}
player p_Enemy;
player p_Player;

int bCheckEndMission;

state Initialize;
state ShowBriefing;
state Nothing;

```

```

    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

```

Mission261.ec

```

mission "translateMission261"
{//Kongo - destroy UCS base
consts
{
    destroyUCSBase = 0;
    destroyMMBase = 1;
}
player p_EnemyUCS;
player p_EnemyMM;
player p_Player;

int bCheckEndMission;
int bUCSunitDestroyed;

state Initialize;
state ShowBriefing;
state Starting;
state Fight;
state Nothing;

//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(destroyUCSBase,"translateGoal261a");
    RegisterGoal(destroyMMBase,"translateGoal261b");
    EnableGoal(destroyUCSBase,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(2);
    p_EnemyUCS = GetPlayer(1);
    p_EnemyMM = GetPlayer(8);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_EnemyUCS.LoadScript("single\singleEasy");
        p_EnemyUCS.EnableAIFeatures(aiUpgradeCannons,false);
        p_EnemyMM.LoadScript("single\singleEasy");
        p_EnemyMM.EnableAIFeatures(aiUpgradeCannons,false);
        p_Player.SetMoney(3000);
        p_EnemyUCS.SetMoney(20000);
        p_EnemyMM.SetMoney(20000);
    }
    if(GetDifficultyLevel()==1)
    {
        p_EnemyUCS.LoadScript("single\singleMedium");
        p_EnemyMM.LoadScript("single\singleMedium");
        p_Player.SetMoney(20000);
        p_EnemyUCS.SetMoney(30000);
        p_EnemyMM.SetMoney(30000);
    }
    if(GetDifficultyLevel()==2)
    {
        p_EnemyUCS.LoadScript("single\singleHard");
        p_EnemyMM.LoadScript("single\singleHard");
        p_Player.SetMoney(15000);
        p_EnemyUCS.SetMoney(50000);
        p_EnemyMM.SetMoney(50000);
    }
    p_Player.EnableAIFeatures(aiEnabled,false);
    p_EnemyMM.EnableAIFeatures(aiControlOffense,false);
    p_EnemyUCS.EnableAIFeatures(aiControlOffense,false);

    p_EnemyMM.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_EnemyMM);
    p_EnemyMM.chooseEnemy(p_EnemyUCS);
}

```

```

//----- Money -----
//----- Researches -----
p_Player.EnableResearch("RES_ED_UBT1",true);
//----- Buildings -----
//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,1200);
//----- Variables -----
bUCSunitDestroyed = false;
bCheckEndMission=false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6.0,20,0);
}
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing261a");
    return Starting,100;
}
//-----
state Starting
{
    if(bUCSunitDestroyed)
    {
        bUCSunitDestroyed=false;
        EnableGoal(destroyMMBase,true);
        p_EnemyMM.SetEnemy(p_Player);
        p_Player.SetEnemy(p_EnemyMM);

        p_EnemyMM.SetNeutral(p_EnemyUCS);
        p_EnemyUCS.SetNeutral(p_EnemyMM);

        p_EnemyMM.EnableAIFeatures(aiControlOffense,true);
        p_EnemyUCS.EnableAIFeatures(aiControlOffense,true);
        AddBriefing("translateBriefing261b");
        return Fight,500;
    }
    return Starting,100;
}
//-----
state Fight
{
    if(bCheckEndMission)
    {
        if(GetGoalState(destroyUCSBase)==goalAchieved && GetGoalState(destroyMMBase)==goalAchieved)
        {
            AddBriefing("translateAccomplished261");
            p_EnemyMM.SetEnemy(p_EnemyUCS);
            p_EnemyUCS.SetEnemy(p_EnemyMM);
            p_EnemyMM.SetEnemy(p_Player);
            p_Player.SetEnemy(p_EnemyMM);
            EnableEndMissionButton(true);
            return Nothing,500;
        }
    }
    return Fight,100;
}
//-----
state Nothing
{
    return Nothing,512;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(!p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed261");
            p_EnemyMM.SetEnemy(p_EnemyUCS);
            p_EnemyUCS.SetEnemy(p_EnemyMM);
            p_EnemyMM.SetEnemy(p_Player);
            p_Player.SetEnemy(p_EnemyMM);
            EndMission(false);
        }
        if(GetGoalState(destroyUCSBase)!=goalAchieved &&

```

```

        lp_EnemyUCS.GetNumberOfBuildings() &&
        p_EnemyUCS.GetNumberOfUnits()<6)
    {
        SetGoalState(destroyUCSBase.goalAchieved);
    }

    if(GetGoalState(destroyMMBase)!=goalAchieved &&
        lp_EnemyMM.GetNumberOfBuildings() &&
        p_EnemyMM.GetNumberOfUnits()<6)
    {
        SetGoalState(destroyMMBase.goalAchieved);
    }
}

//-----
event Timer1() //wolany co 10min=600sec=12000
{
    bUCSunitDestroyed=true;
}
//-----
event UnitDestroyed(unit u_Unit)
{
    unit uAttacker;
    bCheckEndMission=true;
    uAttacker = u_Unit.GetAttacker();
    if(uAttacker==null) return;
    if(uAttacker.GetIFFNumber()!=p_Player.GetIFFNumber()) return;
    if(u_Unit.GetIFFNumber()==p_EnemyUCS.GetIFFNumber())
    {
        bUCSunitDestroyed=true;
        return;
    }
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    unit uAttacker;
    bCheckEndMission=true;
    uAttacker = u_Unit.GetAttacker();
    if(uAttacker == null) return;
    if(uAttacker.GetIFFNumber()!=p_Player.GetIFFNumber()) return;
    if(u_Unit.GetIFFNumber()==p_EnemyUCS.GetIFFNumber())
    {
        bUCSunitDestroyed=true;
        return;
    }
}
//-----
event EndMission()
{
    p_EnemyMM.SetEnemy(p_Player);
    p_Player.SetEnemy(p_EnemyMM);

    p_EnemyMM.SetEnemy(p_EnemyUCS);
    p_EnemyUCS.SetEnemy(p_EnemyMM);
    p_EnemyMM.EnableAIFeatures(aiRejectAlliance,true);
}
}



## Mission262.ec


mission "translateMission262"
{/Lesotho - Africa recover siloses.
consts
{
    destroySiloses = 0;
    launchTime = 48000;//40 min
    clicksPerDay = 16383;
}
player p_EnemyUCS;
player p_EnemyMM;
player p_Player;

int bCheckEndMission;
int bShowBriefing;

state Initialize;
state ShowBriefing;
state Fight;
state ShowVideoState;
state Nothing;

//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
}

```

```

        return Fight,100;
    }
}

state ShowVideoState
{
    ShowVideo("CS211");
    EnableEndMissionButton(true);
    return Nothing,512;
}
}

state Nothing
{
    return Nothing,512;
}
}

event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    int nTime;
    int nDays;
    int nHours;
    nTime = launchTime-GetMissionTime();
    if((nTime<0 || GetGoalState(destroySilos)==goalAchieved) return;
    nDays=nTime/clicksPerDay;
    nHours = ((nTime-nDays*clicksPerDay)*24)/clicksPerDay;
    SetConsoleText("translateMissionMessage262",nDays,nHours);

    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(p_Player.GetNumberOfUnits() &&p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed262");
            p_EnemyMM.SetEnemy(p_EnemyUCS);
            p_EnemyUCS.SetEnemy(p_EnemyMM);
            EndMission(false);
        }
    }
}

event Timer1() //wolany co 10min=600sec=12000
{
    if(bShowbriefing)
    {
        bShowBriefing=false;
        AddBriefing("translateBriefing262b");
        SetConsoleText("");
        p_EnemyMM.EnableAIFeatures(aiEnabled,true);
    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

```

Mission263.ec

```

mission "translateMission263"
{//Algier - destroy UCS water base
const
{
    destroyEnemy1Base = 0;
    destroyEnemy2Base = 1;
}
player p_Enemy1;
player p_Enemy2;
player p_Player;

int bCheckEndMission;
int bShowEnd;

state Initialize;
state ShowBriefing;
state Nothing;
}

```

```

state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(destroyEnemy1Base,"translateGoal263a");
    RegisterGoal(destroyEnemy2Base,"translateGoal263b");
    EnableGoal(destroyEnemy1Base,true);
    EnableGoal(destroyEnemy2Base,true);

    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(4);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy1.EnableAIFeatures(aiUpgradeCannons,false);
        p_Enemy2.EnableAIFeatures(aiUpgradeCannons,false);
        p_Player.SetMoney(30000);
        p_Enemy1.SetMoney(20000);
        p_Enemy2.SetMoney(20000);
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Player.SetMoney(20000);
        p_Enemy1.SetMoney(30000);
        p_Enemy2.SetMoney(30000);
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Player.SetMoney(15000);
        p_Enemy1.SetMoney(50000);
        p_Enemy2.SetMoney(50000);
    }

    p_Player.EnableResearch("RES_ED_AB1",true);
    p_Player.EnableResearch("RES_ED_WHR1",true);
    p_Player.EnableResearch("RES_ED_MB2",true);
    p_Player.EnableResearch("RES_ED_MHR2",true);
    p_Player.EnableResearch("RES_ED_UA31",true);
    p_Player.EnableResearch("RES_ED_UTB1",true);

    p_Enemy1.EnableResearch("RES_UCS_PC",true);
    p_Enemy1.EnableResearch("RES_UCS_WSD",true);
    p_Enemy1.EnableResearch("RES_UCS_MB2",true);
    p_Enemy1.EnableResearch("RES_UCS_BOMBER31",true);

    p_Enemy2.CopyResearches(p_Enemy1);

    p_Player.EnableAIFeatures(aiEnabled,false);

    SetTimer(0,100);
    bCheckEndMission=false;
    bShowEnd=true;
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
p_Player.SetMilitaryUnitsLimit(40000);
return ShowBriefing,100;
}

state ShowBriefing
{
    AddBriefing("translateBriefing263");
    return Nothing,100;
}

state Nothing
{
    return Nothing,100;
}

event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
}

```

```

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lp_Player.GetNumberOfUnits() &&lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed263");
        EndMission(false);
    }
    if(GetGoalState(destroyEnemy1Base)==goalAchieved &&
    (p_Enemy1.GetNumberOfUnits(<6) &&
    lp_Enemy1.GetNumberOfBuildings()))
    {
        SetGoalState(destroyEnemy1Base, goalAchieved);
    }
    if(GetGoalState(destroyEnemy2Base)==goalAchieved &&
    (p_Enemy2.GetNumberOfUnits(<6) &&
    lp_Enemy2.GetNumberOfBuildings()))
    {
        SetGoalState(destroyEnemy2Base, goalAchieved);
    }
    if(bShowEnd &&
    GetGoalState(destroyEnemy1Base)==goalAchieved &&
    GetGoalState(destroyEnemy2Base)==goalAchieved)
    {
        bShowEnd=false;
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        EnableNextMission(2,true);
        AddBriefing("translateAccomplished263");
        EnableEndMissionButton(true);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

```

```

----- Players -----
p_Player = GetPlayer(2);
p_Enemy1 = GetPlayer(1);
p_Enemy2 = GetPlayer(4);
p_Enemy3 = GetPlayer(5);
p_Neutral = GetPlayer(8);

----- AI -----
if(GetDifficultyLevel()==0)
    p_Enemy1.LoadSceneScript("single\singleEasy");
if(GetDifficultyLevel()==1)
    p_Enemy1.LoadSceneScript("single\singleMedium");
if(GetDifficultyLevel()==2)
    p_Enemy1.LoadSceneScript("single\singleHard");

p_Enemy2.EnableStatistics(false);
p_Enemy3.EnableStatistics(false);

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy1.EnableAIFeatures(aiControlDefense,false);
p_Enemy2.EnableAIFeatures(aiEnabled,false);
p_Enemy3.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableAIFeatures(aiEnabled,false);

p_Neutral.EnableAIFeatures(aiRejectAlliance,false);
p_Neutral.ChooseEnemy(p_Enemy1);
p_Player.SetAlly(p_Neutral);
----- Money -----
p_Player.SetMoney(0);
p_Enemy1.SetMoney(40000);
p_Enemy2.SetMoney(0);

----- Researches -----
----- Buildings -----
----- Units -----
p_Guide = GetUnit(GetPointX(0),GetPointY(0),0);
p_Builder = GetUnit(p_Player.GetStartingPointX());
p_Player.GetStartingPointY());0);

----- Artifacts -----
----- Timers -----
SetTimer(0,200);

----- Variables -----
nWayPoint=0;
bShowFailed=true;

----- Camera -----
CallCamera();

Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
);
return ShowBriefing,100;
}
}
state ShowBriefing
{
    AddBriefing("translateBriefing264a");
    return OnTheWay,200;
}
}
state OnTheWay
{
    if(nWayPoint==1)
        Em3.RussianAttack(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
0);

    p_Neutral.IsPointLocated(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),0)
    {
        p_Neutral.GiveAllUnitsTo(p_Player);
        SetGoalState(goToDestinationPoint,goalAchieved);
        EnableGoal(sendToBase50000,true);
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
        p_Player.SetMoney(20000);
        AddBriefing("translateBriefing264b");
        return Fight,100;
    }
}

Distance(p_Guide.GetLocationX(),p_Guide.GetLocationY(),GetPointX(nWayPoint),
)(PointY(nWayPoint))< &&

```

Mission264.ec

```

mission "translateMission264"
//Columbia
consts
{
    goToDestinationPoint = 0;
    sendToBase50000 = 1;
}
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;

player p_Neutral;
player p_Player;

unitex p_Guide;
unitex p_Builder;

int nWayPoint;
int bShowFailed;

state Initialize;
state ShowBriefing;
state OnTheWay;
state Fight;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(goToDestinationPoint,"translateGoal264");
    RegisterGoal(sendToBase50000,"translateGoalSend50000");
    EnableGoal(goToDestinationPoint,true);
}

//----- Temporary players -----
tmpPlayer = GetPlayer(3);
tmpPlayer.EnableStatistics(false);

```

```

Distance(p_Guide.GetLocationX(),p_Guide.GetLocationY(),p_Builder.GetLocationX()
),p_Builder.GetLocationY())<3)
{
    nWayPoint = nWayPoint+1;
    if(nWayPoint>9)
    {
        nWayPoint=10;
    }
}

p_Guide.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint));
ShowArea(4,p_Guide.GetLocationX(),p_Guide.GetLocationY(),p_Guide.GetLocationZ(),1);
return OnTheWay,100;
}

//-----
state Fight
{
    if(p_Player.GetNumberOfUnits()> 6)
p_Enemy!.EnableAIFeatures(aiControlOffense,true);

    if(GetGoalState(sendToBase50000)==goalAchieved &&
p_Player.GetMoneySentToBase()>=50000)
    {
        SetGoalState(sendToBase50000, goalAchieved);
        AddBriefing("translateAccomplished264");
        EnableEndMissionButton(true);
        return Final,500;
    }
    return Fight,200;
}
//-----
state Final
{
    return Final,500;
}

//-----
event Timer0() //wolany co 200 cykli< ustawione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase50000,"translateGoalSend50000",p_Player.GetMoneySe
ntToBase());
    if(bShowFailed)
    {
        if((ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase
())<50000)
        {
            bShowFailed=false;
            SetGoalState(sendToBase50000, goalFailed);
            AddBriefing("translateFailed264a");
            EnableEndMissionButton(true);
        }
        if(lp_Player.GetNumberOfUnits() &&lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed264b");
            EndMission(false);
        }
    }
}

//-----
event EndMission()
{
    p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
}

}

player p_Enemy;
player p_Player;
int bCheckEndMission;
state Initialize;
state ShowBriefing;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    RegisterGoal(destroyEnemyBase,"translateGoal265");
    EnableGoal(destroyEnemyBase,true);

    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(1);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\singleEasy");
        p_Enemy.EnableAIFeatures(aiUpgradeCannons,false);
        p_Player.SetMoney(30000);
        p_Enemy.SetMoney(20000);
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy.LoadScript("single\singleMedium");
        p_Player.SetMoney(20000);
        p_Enemy.SetMoney(30000);
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\singleHard");
    }

    p_Enemy.CreateDefaultUnit(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPoin
tY(),0);
    p_Player.SetMoney(15000);
    p_Enemy.SetMoney(50000);
    p_Player.EnableResearch("RES_ED_AB1",true);
    p_Player.EnableResearch("RES_ED_WH1",true);
    p_Player.EnableResearch("RES_ED_MB2",true);
    p_Player.EnableResearch("RES_ED_MHR2",true);
    p_Player.EnableResearch("RES_ED_UA31",true);
    p_Player.EnableResearch("RES_ED_UT1",true);

    p_Enemy.EnableResearch("RES_UCS_PC",true);
    p_Enemy.EnableResearch("RES_UCS_WSD",true);
    p_Enemy.EnableResearch("RES_UCS_MB2",true);
    p_Enemy.EnableResearch("RES_UCS_BOMBER31",true);

    p_Player.EnableAIFeatures(aiEnabled,false);

    SetTimer(0,100);
    bCheckEndMission=false;
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
p_Player.SetMilitaryUnitsLimit(40000);
return ShowBriefing,100;

state ShowBriefing
{
    AddBriefing("translateBriefing265");
    return Nothing,100;
}

//-----
state Nothing
{
    return Nothing,100;
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

```

Mission265.ec

```

mission "translateMission265"
{Andas - destroy ucs base
consts
{
    destroyEnemyBase = 0;
}

```

```

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed265");
        EndMission(false);
    }
    if(GetGoalState(destroyEnemyBase)==goalAchieved &&
       (p_Enemy.GetNumberOfUnits(<6) &&
       !p_Enemy.GetNumberOfBuildings()))
    {
        SetGoalState(destroyEnemyBase, goalAchieved);
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        EnableNextMission(2,true);
        AddBriefing("translateAccomplished265");
        EnableEndMissionButton(true);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

}



## Mission271.ec


mission "translateMission271"
{
    consts
    {
        sendToBase = 0;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Player;
    int bSitchOnA;
    int nNeedeResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;

    //-----
    state Initialize
    {
        p_Player = GetPlayer(2);
        p_Enemy1 = GetPlayer(1);
        p_Enemy2 = GetPlayer(3);

        fleetCost = p_Player.GetScriptData(0);
        nNeedeResources = fleetCost - p_Player.GetMoneySentToOrbit();

        if(nNeedeResources > 100000)
            nNeedeResources=100000;
        RegisterGoal(sendToBase,"translateGoal271",nNeedeResources,0);
        EnableGoal(sendToBase,true);

        p_Player.SetMoney(10000);
        p_Enemy1.SetMoney(50000);
        p_Enemy2.SetMoney(50000);

        if(GetDifficultyLevel()==0)
        {
            p_Enemy1.LoadScript("single\singleEasy");
            p_Enemy2.LoadScript("single\singleEasy");
        }
        if(GetDifficultyLevel()==1)
        {
            p_Enemy1.LoadScript("single\singleMedium");
            p_Enemy2.LoadScript("single\singleMedium");
        }
    }
}

if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
}

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiEnabled,true);
p_Enemy2.EnableAIFeatures(aiEnabled,true);

//weapons
p_Player2.EnableResearch("RES_LC_WCH2",true);
p_Enemy2.EnableResearch("RES_LC_WSR2",true);
p_Enemy2.EnableResearch("RES_LC_WMR1",true);
p_Enemy2.EnableResearch("RES_LC_WSL1",true);
p_Enemy2.EnableResearch("RES_LC_WHL1",true);
p_Enemy2.EnableResearch("RES_LC_WSS1",true);
p_Enemy2.EnableResearch("RES_LC_WHIS1",true);
p_Enemy2.EnableResearch("RES_LC_WARTILLERY",true);
//CHASSIS
p_Enemy2.EnableResearch("RES_LC_LMO2",true);
p_Enemy2.EnableResearch("RES_LC_UCR1",true);
p_Enemy2.EnableResearch("RES_LC_UCU1",true);
p_Enemy2.EnableResearch("RES_LC_UME1",true);
p_Enemy2.EnableResearch("RES_LC_REG1",true);
p_Enemy2.EnableResearch("RES_LC_SOB1",true);
//SPECIAL
p_Enemy2.EnableResearch("RES_LC_BWC",true);
p_Enemy2.EnableResearch("RES_LC_SGen",true);
p_Enemy2.EnableResearch("RES_LC_SHR1",true);
p_Enemy2.EnableResearch("RES_LC_REG1",true);
p_Enemy2.EnableResearch("RES_LC_SOB1",true);
//AMMO
p_Enemy2.EnableResearch("RES_MCH2",true);
p_Enemy2.EnableResearch("RES_MSR2",true);
p_Enemy2.EnableResearch("RES_MMR2",true);

SetTimer(0,100);
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
bShowFailed=true;
bCheckEndMission=false;
return ShowBriefing,150;/15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing271",nNeedeResources);
    return Mining,200;
}
//-----
state Mining
{
    if(GetGoalState(sendToBase)==goalAchieved &&
       p_Player.GetMoneySentToBase()>=nNeedeResources)
    {
        SetGoalState(sendToBase, goalAchieved);
        AddBriefing("translateAccomplished271");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykl< ustawione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase,"translateGoal271",nNeedeResources,p_Player.GetMoneySentToBase());

    if(bShowFailed)
    {
        if((ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase())<20000)
        {
            bShowFailed=false;
            SetGoalState(sendToBase, goalFailed);
            AddBriefing("translateFailed271a");
        }
    }
}

```

```

        EnableEndMissionButton(true);
        return Nothing;
    }
}

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(p_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed271b");
        EndMission(false);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

Mission272.ec
mission "translateMission272"
{
    consts
    {
        sendToBase = 0;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Player;
    int bShiftOnAI;
    int nNeedeResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;
}

state Initialize
{
    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);

    fleetCost = p_Player.GetScriptData(0);
    nNeedeResources = fleetCost - p_Player.GetMoneySentToOrbit();

    if(nNeedeResources > 100000)
        nNeedeResources=100000;
    RegisterGoal(sendToBase,"translateGoal272",nNeedeResources,0);
    EnableGoal(sendToBase,true);

    p_Player.SetMoney(10000);
    p_Enemy1.SetMoney(50000);
    p_Enemy2.SetMoney(50000);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }
}

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiEnabled,true);
p_Enemy2.EnableAIFeatures(aiEnabled,true);

//weapons
p_Enemy2.EnableResearch("RES_LC_WCH2",true);
p_Enemy2.EnableResearch("RES_LC_WSR2",true);
p_Enemy2.EnableResearch("RES_LC_WMR1",true);
p_Enemy2.EnableResearch("RES_LC_WSL1",true);
p_Enemy2.EnableResearch("RES_LC_WHL1",true);
p_Enemy2.EnableResearch("RES_LC_WSS1",true);
p_Enemy2.EnableResearch("RES_LC_WH51",true);
p_Enemy2.EnableResearch("RES_LC_WARTILLERY",true);
//CHASSIS
p_Enemy2.EnableResearch("RES_LC_UMO2",true);
p_Enemy2.EnableResearch("RES_LC_UCR1",true);
p_Enemy2.EnableResearch("RES_LC_UCU1",true);
p_Enemy2.EnableResearch("RES_LC_UME1",true);
p_Enemy2.EnableResearch("RES_LC_UBO1",true);
//SPECIAL
p_Enemy2.EnableResearch("RES_LC_BWC",true);
p_Enemy2.EnableResearch("RES_LC_SGen",true);
p_Enemy2.EnableResearch("RES_LC_SHR1",true);
p_Enemy2.EnableResearch("RES_LC_REG1",true);
p_Enemy2.EnableResearch("RES_LC_SOBI1",true);
//AMMO
p_Enemy2.EnableResearch("RES_MCH2",true);
p_Enemy2.EnableResearch("RES_MSR2",true);
p_Enemy2.EnableResearch("RES_MM2",true);

SetTimer(0,100);
CallCamera();
```

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
 bShowFailed=true;
 bCheckEndMission=false;
 return ShowBriefing,150://15 sec

//-----
 state ShowBriefing
 {
 AddBriefing("translateBriefing272",nNeedeResources);
 return Mining,200;

//-----
 state Mining
 {
 if(GetGoalState(sendToBase)!=goalAchieved &&
 p_Player.GetMoneySentToBase()>=nNeedeResources)
 {
 SetGoalState(sendToBase,goalAchieved);
 AddBriefing("translateAccomplished272");
 EnableEndMissionButton(true);
 return Nothing, 500;

}
 return Mining,200;

//-----
 state Nothing
 {
 return Nothing, 500;

}

//-----
 event Timer0() //wolany co 100 cykli< ustawnione funkcja SetTimer w state
 Initialize
{

RegisterGoal(sendToBase,"translateGoal272",nNeedeResources,p_Player.GetMoneySentToBase());

if(bShowFailed)

{

if((ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase())<20000)

{

bShowFailed=false;
 SetGoalState(sendToBase,goalFailed);
 AddBriefing("translateFailed272a");
 EnableEndMissionButton(true);
 return Nothing;

}

}
 if(bCheckEndMission)

{

```

bCheckEndMission=false;
if((lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
{
    AddBriefing("translateFailed272b");
    EndMission(false);
}
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

Mission273.ec
mission "translateMission273"
{
    consts
    {
        sendToBase = 0;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Player;
    int bShitchOnA;
    int nNeedeResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;

    //-----
    state Initialize
    {
        p_Player = GetPlayer(2);
        p_Enemy1 = GetPlayer(1);
        p_Enemy2 = GetPlayer(3);

        fleetCost = p_Player.GetScriptData(0);
        nNeedeResources = fleetCost - p_Player.GetMoneySentToOrbit();

        if(nNeedeResources > 100000)
            nNeedeResources=100000;
        RegisterGoal(sendToBase,"translateGoal273",nNeedeResources,0);
        EnableGoal(sendToBase,true);

        p_Player.SetMoney(10000);
        p_Enemy1.SetMoney(50000);
        p_Enemy2.SetMoney(50000);

        if(GetDifficultyLevel()==0)
        {
            p_Enemy1.LoadScript("single\singleEasy");
            p_Enemy2.LoadScript("single\singleEasy");
        }
        if(GetDifficultyLevel()==1)
        {
            p_Enemy1.LoadScript("single\singleMedium");
            p_Enemy2.LoadScript("single\singleMedium");
        }
        if(GetDifficultyLevel()==2)
        {
            p_Enemy1.LoadScript("single\singleHard");
            p_Enemy2.LoadScript("single\singleHard");
        }

        //weapons
        p_Enemy2.EnableResearch("RES_LC_WCH2",true);
        p_Enemy2.EnableResearch("RES_LC_WSR2",true);
        p_Enemy2.EnableResearch("RES_LC_WMP1",true);
        p_Enemy2.EnableResearch("RES_LC_WSL1",true);
        p_Enemy2.EnableResearch("RES_LC_WH1",true);
        p_Enemy2.EnableResearch("RES_LC_WSS1",true);
    }
}

p_Enemy2.EnableResearch("RES_LC_WHS1",true);
p_Enemy2.EnableResearch("RES_LC_WARTILLERY",true);
//CHASSIS
p_Enemy2.EnableResearch("RES_LC_UMO2",true);
p_Enemy2.EnableResearch("RES_LC_UCR1",true);
p_Enemy2.EnableResearch("RES_LC_UCU1",true);
p_Enemy2.EnableResearch("RES_LC_UME1",true);
p_Enemy2.EnableResearch("RES_LC_UBO1",true);
//SPECIAL
p_Enemy2.EnableResearch("RES_LC_BWC",true);
p_Enemy2.EnableResearch("RES_LC_SGen",true);
p_Enemy2.EnableResearch("RES_LC_SHR1",true);
p_Enemy2.EnableResearch("RES_LC_REG1",true);
p_Enemy2.EnableResearch("RES_LC_SOBI",true);
//AMMO
p_Enemy2.EnableResearch("RES_MCH2",true);
p_Enemy2.EnableResearch("RES_MSR2",true);
p_Enemy2.EnableResearch("RES_MMR2",true);

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiEnabled,true);
p_Enemy2.EnableAIFeatures(aiEnabled,true);

SetTimer(0,100);
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
bShowFailed=true;
bCheckEndMission=false;
return ShowBriefing,150://15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing273",nNeedeResources);
    return Mining,200;
}
//-----
state Mining
{
    if(GetGoalState(sendToBase)==goalAchieved &&
p_Player.GetMoneySentToBase()>=nNeedeResources)
    {
        SetGoalState(sendToBase, goalAchieved);
        AddBriefing("translateAccomplished273");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawiione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase,"translateGoal273",nNeedeResources,p_Player.GetMoneySentToBase());
}

if(bShowFailed)
{
    if(ResourcesLeftInMoney()<20000)
    {
        bShowFailed=false;
        SetGoalState(sendToBase, goalFailed);
        AddBriefing("translateFailed273a");
        EnableEndMissionButton(true);
        return Nothing;
    }
}
if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed273b");
        EndMission(false);
    }
}
}

```

```

}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----

```

TutorialEDmis.ec

mission "translateTutorialED"

```

{
    consts
    {
        buildPowerPlant = 0;
        buildVechProdCent = 1;
        buildMine = 2;
        buildRefinery = 3;
        buildTransporters = 4;
        harvestResources = 5;
        buildLeapProdCent = 6;
        buildArmy = 7;
        findEnemy = 8;
        destroyEnemy = 9;
    }
    player p_EnemyUCS;
    player p_Player;

    int nUCSUnitsCount;
    int nBuildingCount;
    int nMoney;
    int nStartX;
    int nStartY;

    //*****
    state Initialize;
    state Start;
    state MoveCamera;
    state BuildPowerPlant;
    state BuildVechProdCent;
    state BuildMine;
    state BuildRefinery;
    state BuildTransporters;
    state HarvestResources;
    state BuildLeapProdCent;
    state BuildArmy;
    state More;
    state EndState;

    state Initialize
    {
        RegisterGoal(buildPowerPlant,"translateGoalTutorialED_PP");
        RegisterGoal(buildVechProdCent,"translateGoalTutorialED_BA");
        RegisterGoal(buildMine,"translateGoalTutorialED_MI");
        RegisterGoal(buildRefinery,"translateGoalTutorialED_Re");
        RegisterGoal(buildTransporters,"translateGoalTutorialED_OH");
        RegisterGoal(harvestResources,"translateGoalTutorialED_Mining");
        RegisterGoal(buildLeapProdCent,"translateGoalTutorialED_FA");
        RegisterGoal(buildArmy,"translateGoalTutorialED_tanks");
        RegisterGoal(findEnemy,"translateGoalTutorialED_findEnemy");
        RegisterGoal(destroyEnemy,"translateGoalTutorialED_destroyEnemy");
    }
}

p_EnemyUCS=GetPlayer(1);
p_Player=GetPlayer(2);

p_EnemyUCS.SetMoney(0);
p_Player.SetMoney(20000);

p_EnemyUCS.EnableAI(false);
p_Player.EnableAI(false);

//weapons
p_Player.EnableResearch("RES_ED_ACH2",false);
p_Player.EnableResearch("RES_ED_WSL1",false);
p_Player.EnableResearch("RES_ED_WSL1",false);
p_Player.EnableResearch("RES_ED_WSR2",false);
p_Player.EnableResearch("RES_ED_WMR1",false);
p_Player.EnableResearch("RES_ED_WH1",false);
p_Player.EnableResearch("RES_ED_WH1",false);

```

```

p_Player.EnableResearch("RES_ED_WHL1",false);
p_Player.EnableResearch("RES_ED_WHR1",false);
p_Player.EnableResearch("RES_ED_AB1",false);
//ammo
p_Player.EnableResearch("RES_ED_MHC2",false);
p_Player.EnableResearch("RES_ED_MB2",false);
p_Player.EnableResearch("RES_ED_MSC3",false);
p_Player.EnableResearch("RES_MM2",false);
p_Player.EnableResearch("RES_MS2",false);
p_Player.EnableResearch("RES_ED_MHR2",false);
p_Player.EnableResearch("RES_ED_MHR4",false);
//chassis
p_Player.EnableResearch("RES_ED_UST3",false);
p_Player.EnableResearch("RES_ED_UMT2",false);
p_Player.EnableResearch("RES_ED_UMW1",false);
p_Player.EnableResearch("RES_ED_UHM1",false);
p_Player.EnableResearch("RES_ED_UHT1",false);
p_Player.EnableResearch("RES_ED_UHW1",false);
p_Player.EnableResearch("RES_ED_UBT1",false);
p_Player.EnableResearch("RES_ED_UA11",false);
p_Player.EnableResearch("RES_ED_UA21",false);
p_Player.EnableResearch("RES_ED_UA31",false);
p_Player.EnableResearch("RES_ED_UA41",false);
p_Player.EnableResearch("RES_ED_USS2",false);
p_Player.EnableResearch("RES_ED_UHS1",false);
//special
p_Player.EnableResearch("RES_ED_SCR",false);
p_Player.EnableResearch("RES_ED_SGen",false);
p_Player.EnableResearch("RES_ED_BMD",false);
p_Player.EnableResearch("RES_ED_BHD",false);
p_Player.EnableResearch("RES_ED_RepHand2",false);

```

```

// 1st tab
p_Player.EnableBuilding("EDBPP",false);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBMF",false);
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHO",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBEN1",false);
p_Player.EnableBuilding("EDBLZ",false);

```

```

nStartX = p_Player.GetStartingPointX();
nStartY = p_Player.GetStartingPointY();

```

```

LookAt(nStartX,nStartY,6.0,20.0,0);
SetTimer(0,6000);
SetTimer(1,100);
nBuildingCount=0;
return Start,100;
}
//-----

```

```

state Start
{
    AddBriefing("translateTutorialED_moveCamera");
    nUCSUnitsCount=p_EnemyUCS.GetNumberOfUnits()/2;
    return MoveCamera,600;
}
//-----

```

```

state MoveCamera
{
    p_Player.EnableBuilding("EDBPP",true);
    EnableGoal(buildPowerPlant,true);
    AddBriefing("translateTutorialED_PP");
    return BuildPowerPlant,100;
}
//-----

```

```

state BuildPowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingPowerPlant))
    {
        p_Player.EnableBuilding("EDBBA",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
    }
}

```

```

SetGoalState(buildPowerPlant.goalAchieved);
EnableGoal(buildVechProdCent,true);
if((p_Player.GetNumberOfBuildings(buildingBase))
    AddBriefing("translateTutorialED_BA");
return BuildVechProdCent,100;
}
if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
{
    p_Player.CreateUnitEx(nStartX,nStartY,
0,null,"EDUBU1",null,null,null,null);
}
if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);

return BuildPowerPlant,60;
}

//-----
state BuildVechProdCent
{
if(p_Player.GetNumberOfBuildings(buildingBase))
{
    p_Player.EnableBuilding("EDBMA",true);
    nBuildingCount=p_Player.GetNumberOfBuildings();
    SetGoalState(buildVechProdCent.goalAchieved);
    EnableGoal(buildMine,true);
    if((p_Player.GetNumberOfBuildings(buildingMine))
        AddBriefing("translateTutorialED_MI");
    return BuildMine,100;
}
if((p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
{
    p_Player.CreateUnitEx(nStartX,nStartY,
0,null,"EDUBU1",null,null,null,null);
}
if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);

if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=nBuildingCount+1;
    AddBriefing("translateTutorialED_BA_Fool");
}
return BuildVechProdCent,60;
}

//-----
state BuildMine
{
if(p_Player.GetNumberOfBuildings(buildingMine))
{
    p_Player.EnableBuilding("EDBRE",true);
    nBuildingCount=p_Player.GetNumberOfBuildings();
    SetGoalState(buildMine.goalAchieved);
    EnableGoal(buildRefinery,true);
    if((p_Player.GetNumberOfBuildings(buildingRefinery))
        AddBriefing("translateTutorialED_RE");
    return BuildRefinery,100;
}
if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=nBuildingCount+1;
    AddBriefing("translateTutorialED_MI_Fool");
}
if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
return BuildMine,100;
}

//-----
state BuildRefinery
{
if(p_Player.GetNumberOfBuildings(buildingRefinery))
{
    nBuildingCount=p_Player.GetNumberOfBuildings();
    SetGoalState(buildRefinery.goalAchieved);
    EnableGoal(buildTransports,true);
    if(p_Player.GetNumberOfUnits(chassisTank | unitCarrier)<2)
        AddBriefing("translateTutorialED_OH");
    return BuildTransports,100;
}
if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=nBuildingCount+1;
    AddBriefing("translateTutorialED_RE_Fool");
}
if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
return BuildRefinery,100;
}

//-----
state BuildTransports
{
if(p_Player.GetNumberOfUnits(chassisTank | unitCarrier)>=2)
{
    SetGoalState(buildTransports.goalAchieved);
    EnableGoal(harvestResources,true);
    AddBriefing("translateTutorialED_Mining");
    nMoney=p_Player.GetMoney();
    return HarvestResources,100;
}
if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
return BuildTransports,100;
}

//-----
state HarvestResources
{
if(nMoney < p_Player.GetMoney())
{
    p_Player.EnableBuilding("EDBFA",true);
    SetGoalState(harvestResources.goalAchieved);
    EnableGoal(buildWeapProdCent,true);
    if((p_Player.GetNumberOfBuildings(buildingFactory))
        AddBriefing("translateTutorialED_FA");
    return BuildWeapProdCent,100;
}
return HarvestResources,100;
}

//-----
state BuildWeapProdCent
{
if(p_Player.GetNumberOfBuildings(buildingFactory))
{
    SetGoalState(buildWeapProdCent.goalAchieved);
    EnableGoal(buildArmy,true);
    nBuildingCount=p_Player.GetNumberOfBuildings();
    AddBriefing("translateTutorialED_tanks");
    return BuildArmy;
}
if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=nBuildingCount+1;
}
return BuildWeapProdCent,100;
}

//-----
state BuildArmy
{
if(p_Player.GetNumberOfUnits(chassisTank | unitArmed)>=5)
{
    SetGoalState(buildArmy.goalAchieved);
    EnableGoal(findEnemy,true);
    AddBriefing("translateTutorialED_findEnemy");
    return More,600;
}
return BuildArmy,200;
}

//-----
state More
{
p_Player.EnableBuilding("EDBAB",true);
p_Player.EnableBuilding("EDBST",true);
p_Player.EnableBuilding("EDBRC",true);
p_Player.EnableBuilding("EDBEN",true);
AddBriefing("translateTutorialED_more");
return EndState,100;
}

//-----
state EndState
{
return EndState,500;
}

//-----
event Timer0()
{
}

```

```

Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),40,400,5000,800
,8);
}

//-----
event Timer1()
{
    if(GetGoalState(findEnemy)!=goalAchieved &&
p_Player.IsPointLocated(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStartin
gPointY(),0))
    {
        //LookAt(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStartingPointY(),6,0,2
,0,0,0);
        SetGoalState(findEnemy,goalAchieved);
        EnableGoal(destroyEnemy,true);
        AddBriefing("translateTutorialED_destroyEnemy");
    }
    if(GetGoalState(destroyEnemy)!=goalAchieved &&
p_EnemyUCS.GetNumberOfUnits())
    {
        SetGoalState(destroyEnemy,goalAchieved);
        AddBriefing("translateTutorialED_Victory");
    }
}

```

LC

CampaignLC-Demo.ec

```

campaign "translateCampaignLC"
{
    consts
    {
        raceUCS=1;
        raceED=2;
        raceLC=3;

        mis314 = 0;

        baseUCS = 24;
        baseED = 25;
        baseLC = 26;

        timeFlagWinter = 1;
        timeFlagEarlySpring = 2;
        timeFlagSpring = 4;
        timeFlagSummer = 8;
        timeFlagDesert = 16;
        timeFlagVolcano = 32;
        timeFlagHeavyVolcano = 64;
        baseFlag = 255;

        // pieniadze przewidywane do budowy statku na poczatku okresu
        cashEarlySpring = 50000;
        cashSpring = 100000;
        cashSummer = 150000;
        cashDesert = 200000;
        cashVolcano = 250000;
        spaceShipPrice = 300000;

        // czas zmiany por roku
        timeEarlySpring = 150000; //2h 5min
        timeSpring = 300000;
        timeSummer = 450000;
        timeDesert = 600000;
        timeVolcano = 750000;
        timeHeavyVolcano = 900000;
        timeEarthDestroyed= 1050000;
    }

    int n_TimeFlag;
    int n_CashCounter;
    int b_showVideo;
    int b_showVideo2;

    state Initialize;
    state Start;
    state Nothing;

    state Initialize
    {
        int i;
    }

```

```

        RegisterMission(baseUCS,"baseUCS","LC\(\Missions\)\LCbaseUCSscript","");
        baseFlag,-84,41,0,0,0);
        RegisterMission(baseED,"baseED","LC\(\Missions\)\LCbaseEDscript","");
        baseFlag,50,60,0,0,0);
        RegisterMission(baseLC,"baseLC","LC\(\Missions\)\LCbaseLCscriptDemo","");
        baseFlag,120,8,0,0,0);
        // nr Ind script preBriefing
        timeFlags x y d1 d2 d3 next1 next2 next3 next4
        RegisterMission(mis314, "1314", "LC\(\Missions\)\Mission314Demo",
        "translateBriefingShort314", timeFlagWinter, 120, 85, 20, 20, 20);

        n_CashCounter = timeFlagEarlySpring;
        n_TimeFlag = timeFlagWinter;
        b_showVideo = true;
        b_showVideo2 = false;

        CreateGamePlayer(1,raceUCS,playerAI,null);
        CreateGamePlayer(2,raceED,playerAI,null);
        CreateGamePlayer(3,raceLC,playerLocal,null);

        LoadBase(1,baseUCS,1);
        LoadBase(2,baseED,2);
        LoadBase(3,baseLC,3);

        SetActivePlayerAndWorld(3,3); //player, world
        SetAvailableWorlds(1 | 8); //misja i baza LC(swiat nr 3)

        EnableMission(mis314,true);
        ActivateMissions(timeFlagWinter,true);
        SetSeason(1);

        return Start,700;
    }
}

state Start
{
    EnableChooseMissionButton(true);
    return Nothing,50;
}

state Nothing
{
    if(b_showVideo2)
    {
        b_showVideo2 = false;

        if(n_TimeFlag == timeFlagWinter)
            ShowVideo("CS314");

        if(n_TimeFlag == timeFlagSpring)
            ShowVideo("CS315");

        if(n_TimeFlag == timeFlagDesert)
            ShowVideo("CS316");

        if(n_TimeFlag == timeFlagVolcano)
            ShowVideo("CS317");
    }
    return Nothing,50;
}

event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    LoadMission(0,iMission);

    EnableMission(iMission,false);

    if(b_showVideo)
    {
        b_showVideo = false;
        b_showVideo2 = true;
    }
}

event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
}

event EndMission(int iMission, int nResult) //
{
    int nTime;
    int nNewTimeFlag;
    nTime = GetCampaignTime();
    if(nResult==true)

```

```

        SetMissionState(iMission,stateAccomplished);
    else
        SetMissionState(iMission,stateFailed);
    }
//*****
```

CampaignLC.ec

```

campaign "translateCampaignLC"
{
    consts
    {
        raceUCS=1;
        raceED=2;
        raceLC=3;

        mis311 = 0;
        mis312 = 1;
        mis313 = 2;
        mis314 = 3;
        mis315 = 4;
        mis321 = 5;
        mis322 = 6;
        mis323 = 7;
        mis331 = 8;
        mis333 = 9;
        mis334 = 10;
        mis341 = 11;
        mis342 = 12;
        mis343 = 13;
        mis344 = 14;
        mis351 = 15;
        mis352 = 16;
        mis353 = 17;
        mis354 = 18;
        mis361 = 19;
        mis362 = 20;
        mis363 = 21;
        mis371 = 22;
        mis372 = 23;

        baseUCS = 24;
        baseED = 25;
        baseLC = 26;

        timeFlagWinter = 1;
        timeFlagEarlySpring = 2;
        timeFlagSpring = 4;
        timeFlagSummer = 8;
        timeFlagDesert = 16;
        timeFlagVolcano = 32;
        timeFlagHeavyVolcano = 64;
        baseFlag = 255;

        // plieniadze przewidywane do budowy statku na poczatku okresu
        cashEarlySpring = 50000;
        cashSpring = 100000;
        cashSummer = 150000;
        cashDesert = 200000;
        cashVolcano = 250000;
        spaceShipPrice = 300000;

        // czas zmiany por roku
        timeEarlySpring = 150000; //2h 5min
        timeSpring = 300000;
        timeSummer = 450000;
        timeDesert = 600000;
        timeVolcano = 750000;
        timeHeavyVolcano = 900000;
        timeEarthDestroyed= 1050000;
    }

    int n_TimeFlag;
    int n_CashCounter;
    int b_showVideo;
    int b_showVideo2;

    state Initialize;
    state Start;
    state Nothing;
```

```

        state Initialize
        {
            int i;

            RegisterMission(baseUCS,"baseUCS","LC\(\Missions\)\LCbaseUCScript","");
            baseFlag, -84, 41,0,0);
            RegisterMission(baseED,"baseED","LC\(\Missions\)\LCbaseEDscript","");
            baseFlag, 50, 60,0,0);
            RegisterMission(baseLC,"baseLC","LC\(\Missions\)\LCbaseLCscript","");
            baseFlag, 120, 8,0,0,0);
            //           nr   lnd   script      preBriefing
            timeFlags x y d1 d2 d3 next1 next2 next3 next4
            RegisterMission(mis311, "311", "LC\(\Missions\)\Mission311",
            "translateBriefingShort311", timeFlagWinter, 60, 60, 60, 60,
            mis312,mis314,mis315);
            RegisterMission(mis312, "312", "LC\(\Missions\)\Mission312",
            "translateBriefingShort312", timeFlagWinter, 105, 25, 20, 20, mis323);
            RegisterMission(mis313, "313", "LC\(\Missions\)\Mission313",
            "translateBriefingShort313", timeFlagWinter, 90, 35, 90, 90);
            RegisterMission(mis314, "314", "LC\(\Missions\)\Mission314",
            "translateBriefingShort314", timeFlagWinter, 120, 85,20,20, mis322);
            RegisterMission(mis315, "315", "LC\(\Missions\)\Mission315",
            "translateBriefingShort315", timeFlagWinter, 161, 60, 90, 90);
        //-----
```

```

            RegisterMission(mis321, "321", "LC\(\Missions\)\Mission321",
            "translateBriefingShort321", timeFlagEarlySpring, 105, 52,0,0,0);
            RegisterMission(mis322, "322", "LC\(\Missions\)\Mission322",
            "translateBriefingShort322", timeFlagEarlySpring,-100, 40,0,0, mis321);
            RegisterMission(mis323, "323", "LC\(\Missions\)\Mission323",
            "translateBriefingShort323", timeFlagEarlySpring, 105, 25,0,0, mis333);
        //-----
```

```

            RegisterMission(mis331, "331", "LC\(\Missions\)\Mission331",
            "translateBriefingShort331", timeFlagSpring, -90, 42,0,0,0,
            mis341,mis342,mis343,mis344);
            RegisterMission(mis333, "333", "LC\(\Missions\)\Mission333",
            "translateBriefingShort333", timeFlagSpring, 105, 25,0,0, mis334);
            RegisterMission(mis334, "334", "LC\(\Missions\)\Mission334",
            "translateBriefingShort334", timeFlagSpring, -52,-2,0,0, mis331);
        //-----
```

```

            RegisterMission(mis341, "341", "LC\(\Missions\)\Mission341",
            "translateBriefingShort341", timeFlagSummer, -47,-18,0,0, mis353);
            RegisterMission(mis342, "342", "LC\(\Missions\)\Mission342",
            "translateBriefingShort342", timeFlagSummer, 77, 15,0,0, mis354);
            RegisterMission(mis343, "343", "LC\(\Missions\)\Mission343",
            "translateBriefingShort343", timeFlagSummer, 47,-18,0,0, mis351);
            RegisterMission(mis344, "344", "LC\(\Missions\)\Mission344",
            "translateBriefingShort344", timeFlagSummer, 133,-13,0,0,0);
        //-----
```

```

            RegisterMission(mis351, "351", "LC\(\Missions\)\Mission351",
            "translateBriefingShort351", timeFlagDesert, 35,-15,0,0, mis352);
            RegisterMission(mis352, "352", "LC\(\Missions\)\Mission352",
            "translateBriefingShort352", timeFlagDesert, -47,-18,0,0, mis361);
            RegisterMission(mis353, "353", "LC\(\Missions\)\Mission353",
            "translateBriefingShort353", timeFlagDesert, 31, 30,0,0,0);
            RegisterMission(mis354, "354", "LC\(\Missions\)\Mission354",
            "translateBriefingShort354", timeFlagDesert, 13, -3,0,0, mis362);
        //-----
```

```

            RegisterMission(mis361, "361", "LC\(\Missions\)\Mission361",
            "translateBriefingShort361", timeFlagVolcano, 26,-27,0,0,0, mis363);
            RegisterMission(mis362, "362", "LC\(\Missions\)\Mission362",
            "translateBriefingShort362", timeFlagVolcano, 26,-27,0,0,0, mis363);
            RegisterMission(mis363, "363", "LC\(\Missions\)\Mission363",
            "translateBriefingShort363", timeFlagVolcano, -75,-5,0,0,0, mis371,mis372);
        //-----
```

```

            RegisterMission(mis371, "371", "LC\(\Missions\)\Mission371",
            "translateBriefingShort371", timeFlagHeavyVolcano,-55,-2,0,0,0);
            RegisterMission(mis372, "372", "LC\(\Missions\)\Mission372",
            "translateBriefingShort372", timeFlagHeavyVolcano,-70,-30,0,0,0);
        //-----
```

```

        n_CashCounter = timeFlagEarlySpring;
        n_TimeFlag = timeFlagWinter;
        b_showVideo = true;
        b_showVideo2 = false;

        CreateGamePlayer(1,raceJCS.playerAI,null);
        CreateGamePlayer(2,raceED.playerAI,null);
        CreateGamePlayer(3,raceLC.playerLocal,null);
```

```

LoadBase(1,baseLC,1);
LoadBase(2,baseED,2);
LoadBase(3,baseLC,3);

SetActivePlayerAndWorld(3,3);//player, world
SetAvailableWorlds(1 | 8)//misja i baza LC(swiat nr 3)

EnableMission(mis311,true);
EnableMission(mis312,true);
ActivateMissions(timeFlagWinter,true);
SetSeason(1);

return Start,700;
}

//-----
state Start
{
    EnableChooseMissionButton(true);
    return Nothing,50;
}
//-----

state Nothing
{
    if(b_showVideo2)
    {
        b_showVideo2 = false;

        if(n_TimeFlag == timeFlagWinter)
            ShowVideo("CS314");

        if(n_TimeFlag == timeFlagSpring)
            ShowVideo("CS315");

        if(n_TimeFlag == timeFlagDesert)
            ShowVideo("CS316");

        if(n_TimeFlag == timeFlagVolcano)
            ShowVideo("CS317");
    }
    return Nothing,50;
}
//-----

event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    TraceD(iMission);

    LoadMission(0,iMission);

    EnableMission(iMission,false);

    if(b_showVideo)
    {
        b_showVideo = false;
        b_showVideo2 = true;
    }
}
//-----

event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
    if(bEnable<2)
        EnableMission(GetNextMission(iMission,iNextNr),bEnable);
    if(bEnable==2)//enable ED base
        SetAvailableWorlds(GetAvailableWorlds() | 4);
    if(bEnable==3)//enable UCS base
        SetAvailableWorlds(GetAvailableWorlds() | 2);

    if(bEnable==4)//you have win the game
    {
        EndGame("Video\\OutroLC.wd1");
    }
    if(bEnable==5)//you loose the game
    {
        EndGame("Video\\OutroLost.wd1");
    }
    if(bEnable==6)//you loose the game hero is killed
    {
        EndGame(null);
    }
}
//-----

event EndMission(int iMission, int nResult) //
{
    if(nResult==true)
        SetMissionState(iMission,stateAccomplished);
    else
        SetMissionState(iMission,stateFailed);
}

if(!AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
{
    if(n_TimeFlag != timeFlagHeavyVolcano)
    {
        n_TimeFlag = n_TimeFlag*2;
        b_showVideo = true;
    }
}

if(AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
{
    ActivateMissionsEq(n_TimeFlag,n_TimeFlag,true);
}
else
{
    SendCustomEvent(0,0,0,0,0);
    return;
}

SetSeason(1);
if(n_TimeFlag==timeFlagEarlySpring)SetSeason(2);
if(n_TimeFlag==timeFlagSpring) SetSeason(3);
if(n_TimeFlag==timeFlagSummer) SetSeason(4);
if(n_TimeFlag==timeFlagDesert) SetSeason(5);
if(n_TimeFlag==timeFlagVolcano) SetSeason(6);
if(n_TimeFlag==timeFlagHeavyVolcano) SetSeason(7);

EnableChooseMissionButton(true);
}

//*****

```

TutorialLC.ec

campaign "translateTutorialLC"

```

{
    state Initialize;
    state Nothing;

    state Initialize
    {
        CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");
        CreateGamePlayer(3,raceLC,playerLocal,"TestGameAI");

        RegisterMission(0,"TutorialLC","LC\\Missions\\TutorialLCmis","","0,0,0,0,0,0);
        LoadMission(0,0);
        SetAvailableWorlds(1);
        SetActivePlayerAndWorld(3,0);
        SetTime(100);
        return Nothing;
    }

    state Nothing
    {
        return Nothing,200;
    }
}

```

Missions

LcbaseEDscript.ec

mission "translateMissionBaseED"

```

{
    player p_Player;

    state Initialize;
    state Nothing;

    state Initialize
    {
        p_Player = GetPlayer(2);
        p_Player.SetMoney(25000);

        p_Player.EnableAIFeatures(aiBuildTanks,false);
        p_Player.EnableAIFeatures(aiBuildShips,false);
        p_Player.EnableAIFeatures(aiBuildHelicopters,false);

        p_Player.EnableAIFeatures(aiBuildSpecialUnits,false);

        //weapons
        p_Player.EnableResearch("RES_ED_ACH2",false);
        p_Player.EnableResearch("RES_ED_WCA2",false);
        p_Player.EnableResearch("RES_ED_WHC1",false);
    }
}

```

```

p_Player.EnableResearch("RES_ED_WSR1",false);
p_Player.EnableResearch("RES_ED_WMR1",false);
p_Player.EnableResearch("RES_ED_AMR1",false);
p_Player.EnableResearch("RES_ED_WHR1",false);
p_Player.EnableResearch("RES_ED_WHL1",false);
p_Player.EnableResearch("RES_ED_WSL1",false);
p_Player.EnableResearch("RES_ED_AB1",false);

//ammo
p_Player.EnableResearch("RES_MCH2",false);
p_Player.EnableResearch("RES_MSC2",false);

//chassis
p_Player.EnableResearch("RES_ED_IJHT1",false);
p_Player.EnableResearch("RES_ED_UBT1",false);
p_Player.EnableResearch("RES_ED_UAM1",false);
p_Player.EnableResearch("RES_ED_UHW1",false);

p_Player.EnableResearch("RES_ED_UA11",false);
p_Player.EnableResearch("RES_ED_UA21",false);
p_Player.EnableResearch("RES_ED_UA31",false);
p_Player.EnableResearch("RES_ED_UA41",false);
//special
p_Player.EnableResearch("RES_ED_SCR",false);
p_Player.EnableResearch("RES_ED_SGen",false);
return Nothing,100;
}

//-----
state Nothing
{
    return Nothing,600;
}
}

```

LCbaseLCscript.ec

```

mission "translateMissionBaseLC"
{
    consts
    {
        oneSeasonMoney = 25000; //CR
        oneSeasonTime = 1500000;/clk = 1h 20min tak naprawde odpowiada
        to polowce sezonu
        noOfSeasons = 20;
        fleetCost = 500000;
        firstPhaseCost = 50000;
        phaseCost = 100000;
        clicksPerDay = 16383;
    }

    player pPlayer;

    int nReport;
    int EndGameResult;
    int bMissionsFinished;

    state Initialize;
    state ShowBriefing1;
    state ShowBriefing2;
    state ShowStartFirstMissionBriefing;
    state Nothing;
    state EndGameState;

    state Initialize
    {
        unitex uHero;
        //----- Goals -----
        RegisterGoal(0,"translateGoalCCSendMoneyToSpace",fleetCost,0);
        RegisterGoal(1,"translateGoalCCampaignPhase1");
        RegisterGoal(2,"translateGoalCCampaignPhase2");
        RegisterGoal(3,"translateGoalCCampaignPhase3");
        RegisterGoal(4,"translateGoalCCampaignPhase4");
        RegisterGoal(5,"translateGoalCCampaignPhase5");
        EnableGoal(0,true);
        EnableGoal(1,true);
        EnableGoal(2,true);
        EnableGoal(3,true);
        EnableGoal(4,true);
        EnableGoal(5,true);
        //----- Temporary players -----
        //----- Players -----
        pPlayer = GetPlayer(3);
        //----- AI -----
    }

    pPlayer.SetScriptData(0,fleetCost);
    pPlayer.SetMilitaryUnitsLimit(10000);
    //----- Money -----
    pPlayer.SetMoney(5000);
    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WCH2",false);
    pPlayer.EnableResearch("RES_LC_ACH2",false);
    pPlayer.EnableResearch("RES_LC_WSR2",false);
    pPlayer.EnableResearch("RES_LC_ASR1",false);
    pPlayer.EnableResearch("RES_LC_WMR1",false);
    pPlayer.EnableResearch("RES_LC_AMR1",false);
    pPlayer.EnableResearch("RES_LC_WSL1",false);
    pPlayer.EnableResearch("RES_LC_WHL1",false);
    pPlayer.EnableResearch("RES_LC_WSS1",false);
    pPlayer.EnableResearch("RES_LC_WHS1",false);
    pPlayer.EnableResearch("RES_LC_WAS1",false);
    pPlayer.EnableResearch("RES_LC_WARTILLERY",false);
    //CHASS
    pPlayer.EnableResearch("RES_LC_UMO2",false);
    pPlayer.EnableResearch("RES_LC_UCR1",false);
    pPlayer.EnableResearch("RES_LC_UCU1",false);
    pPlayer.EnableResearch("RES_LC_UME1",false);
    pPlayer.EnableResearch("RES_LC_UBO1",false);
    //SPECIAL
    pPlayer.EnableResearch("RES_LC_BMD",false);
    pPlayer.EnableResearch("RES_LC_BHD",false);
    pPlayer.EnableResearch("RES_LC_BWC",false);
    pPlayer.EnableResearch("RES_LC_SDIEF",false);
    pPlayer.EnableResearch("RES_LC_SGen",false);
    pPlayer.EnableResearch("RES_LC_MGen",false);
    pPlayer.EnableResearch("RES_LC_HGen",false);
    pPlayer.EnableResearch("RES_LC_SHR1",false);
    pPlayer.EnableResearch("RES_LC_REG1",false);
    pPlayer.EnableResearch("RES_LC_SOBI",false);
    //AMMO
    pPlayer.EnableResearch("RES_MCH2",false);
    pPlayer.EnableResearch("RES_MSR2",false);
    pPlayer.EnableResearch("RES_MSR3",false);
    pPlayer.EnableResearch("RES_MSRA",false);
    pPlayer.EnableResearch("RES_MMTR2",false);
    pPlayer.EnableResearch("RES_MMTR3",false);
    pPlayer.EnableResearch("RES_MMTR4",false);
    //----- Buildings -----
    pPlayer.EnableBuilding("LCBSR",false);
    pPlayer.EnableBuilding("LLASERWALL",false);
    //----- Units -----
    uHero = GetUnit(GetPointX(0),GetPointY(0),0);
    pPlayer.SetScriptUnit(0,uHero);

    uHero.CommandMove(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),0);
    CallCamera();
    SelectUnit(uHero,false);
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(OneSeasonTime);
    //----- Variables -----
    nReport = 0;
    //----- Camera -----
    CallCamera();

    uHero.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
    return ShowBriefing1,100;
}

//-----
state ShowBriefing1
{
    nReport = 0;
    AddBriefing("translateStartCampaignLC1",pPlayer.GetName());
    Snow(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),30,400,8000,800,5);
    return ShowBriefing2,500;
}

//-----
state ShowBriefing2
{
    AddBriefing("translateStartCampaignLC2",pPlayer.GetName(),fleetCost);
    return ShowStartFirstMissionBriefing,100;
}

//-----
state ShowStartFirstMissionBriefing
{
    AddBriefing("translateStartFirstMission");
    return Nothing,50;
}

```

```

state HeroKilled
{
    ShowVideo("CSKilled");
    return EndGameState,3;
}
//-----
state Nothing
{
    unitex uHero;
    int nSendedMoney;
    int nReportTime;
    int i;
    nSendedMoney = pPlayer.GetMoneySentToOrbit();
}

RegisterGoal(0,"translateGoalLCSendMoneyToSpace",fleetCost,(nSendedMoney*10
0)/fleetCost);
    RegisterGoal(1,"translateGoalLCCampaignPhase1");
    RegisterGoal(2,"translateGoalLCCampaignPhase2");
    RegisterGoal(3,"translateGoalLCCampaignPhase3");
    RegisterGoal(4,"translateGoalLCCampaignPhase4");
    RegisterGoal(5,"translateGoalLCCampaignPhase5");

    for(i=1;i<6;i++)
    {
        if(GetGoalState(i)!=goalAchieved && (nSendedMoney>=((phaseCost*(i-
1))+firstPhaseCost)))
        {
            SetGoalState(i,goalAchieved);
        }
    }

    if(bMissionsFinished &&
(nSendedMoney+pPlayer.GetMoney()+pPlayer.GetMoneyFromFinishedMissions())<fleetCost)
    {
        AddBriefing("translateCampaignLCFailed");
        EndGameResult=5;
        return EndGameState,3;
    }

    if(nSendedMoney>=fleetCost)
    {
        SetGoalState(0,goalAchieved);
        AddBriefing("translateCampaignLCAccomplished",pPlayer.GetName());
        EndGameResult=4;
        return EndGameState,3;
    }
}

SetConsoleText("translateCampaignLCRemainingTime",((oneSeasonTime*noOfSeas
ons) - GetMissionTime())/clicksPerDay);/XXMD
uHero = pPlayer.GetScriptUnit(0);
if(uHero==null || !uHero.IsLive())
{
    AddBriefing("translateCampaignLCHeroKilled",pPlayer.GetName());
    EndGameResult=6;
    return HeroKilled,2;
}
if(pPlayer.GetNumberOfBuildings(buildingSpaceStation))
{
}

AddBriefing("translateCampaignLCspacePortDestroyed",pPlayer.GetName());
    EndGameResult=5;
    return EndGameState,3;
}
return Nothing,600;
}
//-----
state EndGameState
{
    EnableNextMission(0,EndGameResult);
    return EndGameState,600;
}
//-----
event Timer0() //wolany co 200 000 cykli< ustawiione funkcja SetTimer w state
Initialize
{
    int nSendedMoney;
    int nProjectStatus;
    int nMoneyForNextSeason;
    int nPrecentDelivered;

    nReport = nReport+1;

    if(nReport>noOfSeasons)
    {
        AddBriefing("translateCampaignLCFailed",pPlayer.GetName());
        EndGameResult=5;
        return EndGameState;
    }
}

nSendedMoney = pPlayer.GetMoneySentToOrbit();// to ma tu byc
nProjectStatus = (nSendedMoney*100)/fleetCost;
nMoneyForNextSeason = ((nReport+1)*oneSeasonMoney) -
nSendedMoney;
    if(nMoneyForNextSeason<10000) nMoneyForNextSeason=10000;

    nPrecentDelivered = (nSendedMoney*100)/(oneSeasonMoney*nReport);
    if(nSendedMoney < oneSeasonMoney*nReport)
        AddBriefing("translateLCDelayReport",pPlayer.GetName(),nReport,nProjectStatus,n
MoneyForNextSeason,nPrecentDelivered);
        else

AddBriefing("translateLCReport",pPlayer.GetName(),nReport,nProjectStatus,nMoney
ForNextSeason);
}
event CustomEvent0(int k1,int k2,int k3,int k4) //XXMD
{
    bMissionsFinished=true;
}
}

LbaseLCscriptD
mission "translateMissionBaseLC"
{
    consts
    {
        oneSeasonMoney = 25000;//CR
        oneSeasonTime = 150000;//clk = 1h 20min tak naprawde to odpowiada
        to polowce sezonu
        noOfSeasons = 20;
        fleetCost = 500000;
        firstPhaseCost = 50000;
        phaseCost = 100000;
        clicksPerDay = 16383;
    }

    player pPlayer;
    int nReport;
    int EndGameResult;

    state Initialize;
    state ShowBriefing1;
    state ShowBriefing2;
    state ShowStartFirstMissionBriefing;
    state Nothing;
    state EndGameState;

    state Initialize
    {
        unitex uHero;
        //----- Goals -----
        RegisterGoal(0,"translateGoalLCSendMoneyToSpace",fleetCost,0);
        RegisterGoal(1,"translateGoalLCCampaignPhase1");
        RegisterGoal(2,"translateGoalLCCampaignPhase2");
        RegisterGoal(3,"translateGoalLCCampaignPhase3");
        RegisterGoal(4,"translateGoalLCCampaignPhase4");
        RegisterGoal(5,"translateGoalLCCampaignPhase5");
        EnableGoal(0,true);
        EnableGoal(1,true);
        EnableGoal(2,true);
        EnableGoal(3,true);
        EnableGoal(4,true);
        EnableGoal(5,true);
        //----- Temporary players -----
        //----- Players -----
        pPlayer = GetPlayer(3);
        //----- AI -----
        pPlayer.SetScriptData(0,fleetCost);
        pPlayer.SetMilitaryUnitsLimit(10000);
        //----- Money -----
        pPlayer.SetMoney(5000);
        //----- Researches -----
        pPlayer.EnableResearch("RES_LC_WCH2",false);
        pPlayer.EnableResearch("RES_LC_ACH2",false);
        pPlayer.EnableResearch("RES_LC_WSR2",false);
        pPlayer.EnableResearch("RES_LC_ASRI",false);
        pPlayer.EnableResearch("RES_LC_WMR1",false);
        pPlayer.EnableResearch("RES_LC_AMR1",false);
    }
}

```

```

pPlayer.EnableResearch("RES_LC_WSL1",false);
pPlayer.EnableResearch("RES_LC_WHL1",false);
pPlayer.EnableResearch("RES_LC_WSS1",false);
pPlayer.EnableResearch("RES_LC_WH1",false);
pPlayer.EnableResearch("RES_LC_WAS1",false);
pPlayer.EnableResearch("RES_LC_WARTILLERY",false);
//CHASS
pPlayer.EnableResearch("RES_LC_UMO2",false);
pPlayer.EnableResearch("RES_LC_UCR1",false);
pPlayer.EnableResearch("RES_LC_UCU1",false);
pPlayer.EnableResearch("RES_LC_UME1",false);
pPlayer.EnableResearch("RES_LC_UBO1",false);
//SPECIAL
pPlayer.EnableResearch("RES_LC_BMD",false);
pPlayer.EnableResearch("RES_LC_BHD",false);
pPlayer.EnableResearch("RES_LC_BWC",false);
pPlayer.EnableResearch("RES_LC_SDIEF",false);
pPlayer.EnableResearch("RES_LC_Scen",false);
pPlayer.EnableResearch("RES_LC_Mon",false);
pPlayer.EnableResearch("RES_LC_HGn",false);
pPlayer.EnableResearch("RES_LC_SHR1",false);
pPlayer.EnableResearch("RES_LC_REG1",false);
pPlayer.EnableResearch("RES_LC_SOB1",false);
//AMMO
pPlayer.EnableResearch("RES_MCH2",false);
pPlayer.EnableResearch("RES_MSR2",false);
pPlayer.EnableResearch("RES_MSR3",false);
pPlayer.EnableResearch("RES_MSR4",false);
pPlayer.EnableResearch("RES_MMRC",false);
pPlayer.EnableResearch("RES_MMRC2",false);
pPlayer.EnableResearch("RES_MMRC4",false);
//----- Buildings -----
pPlayer.EnableBuilding("LCBSR",false);
pPlayer.EnableBuilding("LLASERWAL",false);
//----- Units -----
uHero = GetUnit(GetPointX(0),GetPointY(0),0);
pPlayer.SetScriptUnit(0,uHero);

uHero.CommandMove(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),0);
//----- Artefacts -----
//----- Timers -----
SetTimer(0,oneSeasonTime);
//----- Variables -----
nReport = 0;
pPlayer.SetScriptData(1,0);
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing1,100;
}
//----- ShowBriefing1
{
nReport = 0;
AddBriefing("translateStartCampaignLc1",pPlayer.GetName());
Snow(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),30,400,8000,800,5);
return ShowBriefing2,500;
}
//----- ShowBriefing2
{
AddBriefing("translateStartCampaignLc2",pPlayer.GetName(),fleetCost);
return ShowStartFirstMissionBriefing,100;
}
//----- ShowStartFirstMissionBriefing
{
AddBriefing("translateStartFirstMission");
return Nothing,50;
}
//----- state HeroKilled
{
ShowVideo("CSKilling");
return EndgameState,3;
}
//----- state EndDemo
{
EndGame(null);
}
//----- state Nothing

```

{ unitex uHero;

SetConsoleText("translateCampaignLCRemainingTime",((oneSeasonTime*noOfSeasons) - GetMissionTime())/clicksPerDay);

```

if(pPlayer.GetScriptData(1)==1)
{
    AddBriefing("translateCampaignAccomplishedDemo");
    return EndDemo,100;
}

uHero = pPlayer.GetScriptUnit(0);
if(uHero==null || !uHero.IsLive())
{
    AddBriefing("translateCampaignLCHeroKilled",pPlayer.GetName());
    return EndDemo,100;
}

if(lpPlayer.GetNumberOfBuildings(buildingSpaceStation))
{
    AddBriefing("translateCampaignLCSpacePortDestroyed",pPlayer.GetName());
    return EndDemo,100;
}

return Nothing,100;
}
//-----
state EndGameState
{
    EnableNextMission(0,EndGameState);
    return EndGameState,600;
}
//-----
event Timer0() //wolny co 200 000 cykli< ustawiione funkcja SetTimer w state
Initialze
{
    int nSendedMoney;
    int nProjectStatus;
    int nMoneyForNextSeason;
    int nPrecentDelivered;

    nReport = nReport+1;

    if(nReport>noOfSeasons)
    {
        AddBriefing("translateCampaignLCFailed",pPlayer.GetName());
        Endgame Endgame=5;
        return EndGameState;
    }

    nSendedMoney = pPlayer.GetMoneySentToOrbit(); //to ma tu byc
    nProjectStatus = (nSendedMoney*100)/fleetCost;
    nMoneyForNextSeason = ((nReport+1)*oneSeasonMoney) -
nSeasonMoney;
    if((nMoneyForNextSeason<10000) nMoneyForNextSeason=10000;

    nPrecentDelivered = (nSendedMoney*100)/(oneSeasonMoney*nReport);
    if(nSendedMoney < oneSeasonMoney*nReport)

    AddBriefing("translateLCDelayReport",pPlayer.GetName(),nReport,nProjectStatus,nMoneyForNextSeason,nPrecentDelivered);
    else

    AddBriefing("translateLCReport",pPlayer.GetName(),nReport,nProjectStatus,nMoneyForNextSeason);
    }
}

LcbaseUCSscript.ec
mission "translateMissionBaseUCS"
{
player p_PlayerUCS;

state Initialize;
state Nothing;
state Initialize
{
    p_PlayerUCS = GetPlayer(1);
    p_PlayerUCS.SetMoney(10000);

    p_PlayerUCS.EnableAIFeatures(aiBuildTanks,false);
    p_PlayerUCS.EnableAIFeatures(aiBuildShips,false);
    p_PlayerUCS.EnableAIFeatures(aiBuildHelicopters,false);
}

```

```

p_PlayerUCS.EnableAIFeatures(aiBuildSpecialUnits,false);

//weapons
p_PlayerUCS.EnableResearch("RES_UCS_WMR1",false);
p_PlayerUCS.EnableResearch("RES_UCS_WAMR1",false);
p_PlayerUCS.EnableResearch("RES_UCS_WSP1",false);
p_PlayerUCS.EnableResearch("RES_UCS_WHPI",false);
p_PlayerUCS.EnableResearch("RES_UCS_WPC",false);
p_PlayerUCS.EnableResearch("RES_UCS_WAPB1",false);
p_PlayerUCS.EnableResearch("RES_UCS_WSD",false);
//CHASSIS
p_PlayerUCS.EnableResearch("RES_UCS_UHL1",false);
p_PlayerUCS.EnableResearch("RES_UCSUBL1",false);
p_PlayerUCS.EnableResearch("RES_UCS_UMI1",false);
p_PlayerUCS.EnableResearch("RES_UCS_UAH1",false);
p_PlayerUCS.EnableResearch("RES_UCS_BOMBER21",false);
p_PlayerUCS.EnableResearch("RES_UCS_BOMBER31",false);
//AMMO
p_PlayerUCS.EnableResearch("RES_UCS_MSR2",false);
p_PlayerUCS.EnableResearch("RES_UCS_MM2",false);
p_PlayerUCS.EnableResearch("RES_UCS_MB2",false);
//SPECIAL
p_PlayerUCS.EnableResearch("RES_UCS_BMD",false);
p_PlayerUCS.EnableResearch("RES_UCS_BHD",false);
p_PlayerUCS.EnableResearch("RES_UCS_SGen",false);
p_PlayerUCS.EnableResearch("RES_UCS_SHD",false);

return Nothing;
}
state Nothing
{
}
}

Mission311.ec
mission "translateMission311"
{
consts
{
    findResource = 0;
    sendToBase20000 = 1;
    backToBase = 2;
}
player pEnemy;
player pPlayer;

int n_ResourceX;
int n_ResourceY;
int bShowFailed;
int b_AIActivated;
int bCheckEndMission;
int bVictory;

state Initialize;
state ShowBriefing;
state Searching;
state Mining;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(findResource,"translateGoalFindResources");
    RegisterGoal(sendToBase20000,"translateGoalSend20000");
    RegisterGoal(backToBase,"translateGoalHeroBackToBase");
    EnableGoal(findResource,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(2);

    //----- AI -----
    if(GetDifficultyLevel()==0)
        pEnemy.LoadSceneScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        pEnemy.LoadSceneScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        pEnemy.LoadSceneScript("single\singleHard");

    pEnemy.EnableAIFeatures(aiControlOffense,false);
}

pEnemy.EnableAIFeatures(aiControlDefense,false);
pPlayer.EnableAIFeatures(aiEnabled,false);
//----- Money -----
pPlayer.SetMoney(10000);
pEnemy.SetMoney(20000);

//----- Researches -----
pPlayer.EnableResearch("RES_MCH2",true);
//----- Buildings -----
pPlayer.EnableBuilding("LCBBF",false);
pPlayer.EnableBuilding("LCBP",false);
pPlayer.EnableBuilding("LCBBA",false);
pPlayer.EnableBuilding("LCBMR",false);
pPlayer.EnableBuilding("LCBSR",false);
pPlayer.EnableBuilding("LCBC",false);
pPlayer.EnableBuilding("LCBAB",false);
pPlayer.EnableBuilding("LCBGA",false);
pPlayer.EnableBuilding("LCBDE",false);
pPlayer.EnableBuilding("LCBHQ",false);
pPlayer.EnableBuilding("LCBSD",false);
pPlayer.EnableBuilding("LCBWC",false);
pPlayer.EnableBuilding("LCBSS",false);
pPlayer.EnableBuilding("LCBLZ",false);

//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,10000);

//----- Variables -----
n_ResourceX = GetPointX(1);
n_ResourceY = GetPointY(1);
b_AIActivated=false;
bCheckEndMission=false;
bShowFailed=true;
bVictory=false;

//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,200;//15 sec
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    EnableNextMission(2,true);
    Snow(n_ResourceX,n_ResourceY,310,500,5000,500,10);
    AddBriefing("translateBriefing311a",pPlayer.GetName());
    return Searching,100;
}
//-----
state Searching
{
    if(pPlayer.IsPointLocated(n_ResourceX,n_ResourceY))
    {
        SetGoalState(findResource,goalAchieved);
        EnableGoal(sendToBase20000,true);
        pPlayer.EnableBuilding("LCBPP",true);
        pPlayer.EnableBuilding("LCBAA",true);
        pPlayer.EnableBuilding("LCBSR",true);
        pPlayer.EnableBuilding("LCBLZ",true);
        AddBriefing("translateBriefing311b",pPlayer.GetName());
        return Mining,200;
    }
    return Searching,100;
}
//-----
state Mining
{
    if(pPlayer.GetMoneySendToBase()>=20000)
    {
        SetGoalState(sendToBase20000,goalAchieved);
        AddBriefing("translateAccomplished311",pPlayer.GetName());
        EnableEndMissionButton(true);
        bVictory=true;
        return Nothing,500;
    }
    return Mining,200;
}
//-----
```

```

state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase20000,"translateGoalSend20000",pPlayer.GetMoneySent
ToBase());
if(b_AIActivated &&
(pPlayer.IsPointLocated(GetPointX(2),GetPointY(2)) ||
pPlayer.IsPointLocated(GetPointX(3),GetPointY(3),0) ||
pPlayer.IsPointLocated(GetPointX(4),GetPointY(4),0) ||
pPlayer.IsPointLocated(GetPointX(5),GetPointY(5),0) ||
(pPlayer.GetMoneySentToBase()>=15000-(GetDifficultyLevel()*2500))))
{
    b_AIActivated=true;
    AddBriefing("translateBriefing311",pPlayer.GetName());
    pEnemy.EnableAIFeatures(aiControlOffense,true);
}

pPlayer.EnableBuilding("LCBPP",true);
pPlayer.EnableBuilding("LCBAA",true);
pPlayer.EnableBuilding("LCBBF",true);
pPlayer.EnableBuilding("LCBMR",true);
pPlayer.EnableBuilding("LCBAB",true);
pPlayer.EnableBuilding("LCBHQ",true);
pPlayer.EnableBuilding("LCBLZ",true);
}

if(bShowFailed)
{
    if(IsGoalEnabled(sendToBase20000))
        if((ResourcesLeftInMoney() + pPlayer.GetMoney() + pPlayer.GetMoneySentToBase())<20000)
        {
            bShowFailed=false;
            SetGoalState(sendToBase20000, goalFailed);
            AddBriefing("translateFailed311",pPlayer.GetName());
            EnableEndMissionButton(true);
            bVictory=false;
            return Nothing;
        }
}

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
    {
        if(bVictory)
            EndMission(true);
        else
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }
    }
}

event Timer1() //wolany co 6000 cykli
{
    Snow(n_ResourceX,n_ResourceY,10,400,2500,800,10);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----



{ makeTests = 0;
backToBase = 1;
}

player pPlayer;
player pNeutral;
player pTowers;
player pEnemy;

unitex uHero;
unitex uTester;

int bTestPhase;

state Initialize;
state ShowBriefing;
state Testing;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(makeTests,"translateGoal312");
    RegisterGoal(backToBase,"translateGoalHeroBackToBase" 0);
    EnableGoal(makeTests,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(2);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pNeutral = GetPlayer(5);
    pTowers = GetPlayer(10);
    pEnemy = GetPlayer(1);

    //----- AI -----
    pPlayer.EnableStatistics(false);
    pNeutral.EnableStatistics(false);
    pTowers.EnableStatistics(false);
    pEnemy.EnableStatistics(false);

    pPlayer.EnableAIFeatures(aiEnabled,false);
    pNeutral.EnableAIFeatures(aiEnabled,false);
    pTowers.EnableAIFeatures(aiEnabled,false);

    pNeutral.EnableAIFeatures(aiRejectAlliance,false);
    pTowers.EnableAIFeatures(aiRejectAlliance,false);

    pPlayer.SetAlly(pNeutral);
    pPlayer.SetAlly(pTowers);

    pTowers.SetEnemy(pNeutral);
    pNeutral.SetNeutral(pTowers);

    pNeutral.SetEnemy(pEnemy);
    pEnemy.SetNeutral(pNeutral);
    //----- Money -----
    pPlayer.SetMoney(0);
    //----- Researches -----
    pPlayer.EnableResearch("RES_MCH2",true);
    //----- Buildings -----
    pPlayer.EnableBuilding("LCBFF",false);
    pPlayer.EnableBuilding("LCBP",false);
    pPlayer.EnableBuilding("LCBAA",false);
    pPlayer.EnableBuilding("LCBMR",false);
    pPlayer.EnableBuilding("LCBSR",false);
    pPlayer.EnableBuilding("LCBC",false);
    pPlayer.EnableBuilding("LCBAB",false);
    pPlayer.EnableBuilding("LCBGA",false);
    pPlayer.EnableBuilding("LCBE",false);
    pPlayer.EnableBuilding("LCBHQ",false);
    pPlayer.EnableBuilding("LCBSD",false);
    pPlayer.EnableBuilding("LCBWC",false);
    pPlayer.EnableBuilding("LCBSS",false);
    pPlayer.EnableBuilding("LCBLZ",true);
    pPlayer.EnableBuilding("LCLASERWALL",false);
    //----- Units -----
    uHero=pPlayer.GetScriptUnit(0);
    uTester=GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));
    //----- Artifacts -----
    //----- Timers -----
}

```

Mission312.ec

mission "translateMission312"
{
 consts
}

```

SetTimer(0,100);
//----- Variables -----
bTestPhase=0;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;//5 sec
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing312a",pPlayer.GetName());
    return Testing,100;
}
//-----
state Testing
{
    if(bTestPhase==0 &&
       uHero.IsInWorld(GetWorldNum()) &&
       uHero.DistanceTo((GetPointX(1),GetPointY(1))<8)
    {
        uHero.ChangePlayer(pNeutral);
        uHero.CommandMove((GetPointX(1),GetPointY(1),0));
        bTestPhase=1;
        return Testing,100;
    }

    if(bTestPhase==1 &&
       uHero.DistanceTo((GetPointX(1),GetPointY(1))==0)
    {
        AddBriefing("translateBriefing312b",pPlayer.GetName());//start speed
        uHero.CommandMove((GetPointX(2),GetPointY(2),0));
        uTester.CommandMove((GetPointX(3),GetPointY(3),0));
        bTestPhase=2;
        return Testing,100;
    }

    if(bTestPhase==2 &&
       uHero.DistanceTo((GetPointX(2),GetPointY(2))==0)
    {
        AddBriefing("translateBriefing312c",pPlayer.GetName());//end speed test
        start hp test
        uHero.CommandMove((GetPointX(6),GetPointY(6),0));
        bTestPhase=3;
        return Testing,100;
    }

    if(bTestPhase==3 &&
       uHero.DistanceTo((GetPointX(6),GetPointY(6))==0)
    {
        bTestPhase=4;
        return Testing,200;
    }

    if(bTestPhase==4)
    {
        uHero.CommandMove((GetPointX(4),GetPointY(4),0));
        bTestPhase=5;
        return Testing,100;
    }

    if(bTestPhase==5 &&
       uHero.DistanceTo((GetPointX(4),GetPointY(4))==0)
    {
        AddBriefing("translateBriefing312d",pPlayer.GetName());//end hp test
        start fire test
        uHero.CommandMove((GetPointX(5),GetPointY(5),0));
        bTestPhase=6;
        return Testing,100;
    }

    if(bTestPhase==6 &&
       lpEnemy.GetNumberOfUnits())
    {
        uHero.ChangePlayer(pPlayer);
        SetGoalState(makeTests,goalAchieved);
        EnableGoal(backToBase,true);
        AddBriefing("translateAccomplished312",pPlayer.GetName());//end tests
        bTestPhase=7;
        return Nothing,100;
    }
    return Testing,100;
}

//-----
state Nothing
{
    if(GetGoalState(backToBase)!=goalAchieved &&
       lpHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}
//-----
event UnitDestroyed(unit u_Unit)
{
}
//-----
event BuildingDestroyed(unit u_Unit)
{
}
//-----
event EndMission()
{
    pNeutral.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pNeutral);

    pTowers.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pTowers);
}
}


```

Mission313.ec

```

mission "translateMission313"
{
    consts
    {
        sendToBase20000 = 0;
    }

    player pEnemy;
    player pPlayer;

    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;
    //-----
    state Initialize
    {
        player tmpPlayer;
        //----- Goals -----
        RegisterGoal(sendToBase20000,"translateGoalSend20000",0);
        EnableGoal(sendToBase20000,true);

        //----- Temporary players -----
        tmpPlayer = GetPlayer(1);
        tmpPlayer.EnableStatistics(false);

        //----- Players -----
        pPlayer = GetPlayer(3);
        pEnemy = GetPlayer(2);

        //----- AI -----
        if(GetDifficultyLevel()==0)
            pEnemy.LoadScript("single\\singleEasy");
        if(GetDifficultyLevel()==1)
            pEnemy.LoadScript("single\\singleMedium");
        if(GetDifficultyLevel()==2)
            pEnemy.LoadScript("single\\singleHard");

        pPlayer.EnableAIFeatures(aiEnabled,false);
        //pEnemy.AddToMainTargetList();
        //----- Money -----
        pPlayer.SetMoney(10000);
        pEnemy.SetMoney(20000);

        //----- Researches -----
    }
}
```

```

pPlayer.EnableResearch("RES_LC_WCH2",true);
pPlayer.EnableResearch("RES_LC_ACH2",true);
pPlayer.EnableResearch("RES_LC_UME1",true);
pPlayer.EnableResearch("RES_MSR2",true);
pPlayer.EnableResearch("RES_LC_BMD",true);

pEnemy.EnableResearch("RES_ED_WCH2",true);
pEnemy.EnableResearch("RES_MCH2",true);
pEnemy.EnableResearch("RES_ED_UA11",true);

//----- Buildings -----
//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,10000);

//----- Variables -----
bCheckEndMission=false;
bShowFailed=true;

//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,200;/15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing313",pPlayer.GetName());
    return Mining,100;
}
//-----
state Mining
{
    if(pPlayer.GetMoneySentToBase()>=20000)
    {
        SetGoalState(sendToBase20000,goalAchieved);
        AddBriefing("translateAccomplished313",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase20000,"translateGoalSend20000",pPlayer.GetMoneySentToBase());
    if(bShowFailed)
    {
        if(IsGoalEnabled(sendToBase20000))
        {
            if((ResourcesLeftInMoney()+pPlayer.GetMoney()+pPlayer.GetMoneySentToBase())<20000)
            {
                bShowFailed=false;
                SetGoalState(sendToBase20000,goalFailed);
                AddBriefing("translateFailed313",pPlayer.GetName());
                EnableEndMissionButton(true);
                return Nothing;
            }
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if((pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings()))
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }
    }
}
//-----
event Timer1() //wolany co 6000 cykli
{
    Snow(GetPointX(0),GetPointY(0),30,400,2500,800,10);
}

```

```

}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}
//-----
```

Mission314.ec

```

mission "translateMission314"
{
    consts
    {
        deliverDocuments = 0;
        backToBase = 1;
    }

    player pEnemy;
    player pPlayer;
    player pNeutral;

    unitex uHero;

    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Searching;
    state Nothing;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(deliverDocuments,"translateGoal314");
    RegisterGoal(backToBase,"translateGoalHeroBackToBase");
    EnableGoal(deliverDocuments,true);
    //----- Temporary players -----
}

//----- Players -----
pPlayer = GetPlayer(3);
pEnemy = GetPlayer(2);
pNeutral = GetPlayer(1);
//----- AI -----
if(GetDifficultyLevel()==0)
{
    pEnemy.LoadScript("single\singleEasy");
    pEnemy.EnableAIFeatures(aiControlDefense,false);
}
if(GetDifficultyLevel()==1)
    pEnemy.LoadScript("single\singleMedium");
if(GetDifficultyLevel()==2)
    pEnemy.LoadScript("single\singleHard");

pNeutral.LoadScript("single\singleHard");
pNeutral.EnableAIFeatures(aiControlOffense,false);
pNeutral.EnableAIFeatures(aiControlOffense,false);

pPlayer.EnableAIFeatures(aiEnabled,false);

pNeutral.SetNeutral(pPlayer);
pPlayer.SetNeutral(pNeutral);
pNeutral.ChooseEnemy(pEnemy);
pNeutral.SetEnemy(pEnemy);
//----- Money -----
pPlayer.SetMoney(0);
pEnemy.SetMoney(20000);
pNeutral.SetMoney(30000);
//----- Research -----
pEnemy.EnableResearch("RES_ED_WCH2",true);
pEnemy.EnableResearch("RES_ED_ACH2",true);
pEnemy.EnableResearch("RES_MCH2",true);
pEnemy.EnableResearch("RES_ED_UA11",true);
//----- Buildings -----
pPlayer.EnableBuilding("LCBBF",false);
pPlayer.EnableBuilding("LCBPF",false);
pPlayer.EnableBuilding("LCBBA",false);
pPlayer.EnableBuilding("LCBMR",false);
pPlayer.EnableBuilding("LCBSR",false);
}

```

```

pPlayer.EnableBuilding("LCBRC",false);
pPlayer.EnableBuilding("LCBAB",false);
pPlayer.EnableBuilding("LCBGA",false);
pPlayer.EnableBuilding("LCBDE",false);
pPlayer.EnableBuilding("LCBSQ",false);
pPlayer.EnableBuilding("LCBSU",false);
pPlayer.EnableBuilding("LCBSZ",false);
pPlayer.EnableBuilding("LCLASERWALL",false);
//----- Units -----
uHero=pPlayer.GetScriptUnit(0);
//----- Artifacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission=false;
//----- Camera -----
CallCamera(a);

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6.0,20.0);
return ShowBriefing,100;//5 sec
}
//----- ShowBriefing -----
{
AddBriefing("translateBriefing314",pPlayer.GetName());
return Searching,100;
}
//----- Searching -----
{
if(uHero)
{
if(uHero.IsInWorld(GetWorldNum())
&& uHero.DistanceTo(GetPointX(0),GetPointY(0))<10
&& uHero.GetLocationZ())
{
SetGoalState(deliverDocuments,goalAchieved);
EnableGoal(backToBase,true);
AddBriefing("translateAccomplished314",pPlayer.GetName());
EnableNextMission(0,true);
return Nothing,200;
}
}
return Searching,100;
}
//----- Nothing -----
{
if(GetGoalState(backToBase)!=goalAchieved &&
uHero.IsInWorld(GetWorldNum()))
{
SetGoalState(backToBase,goalAchieved);
EnableEndMissionButton(true);
}
return Nothing, 500;
}
//----- Timer0() //wolany co 100 cykli< ustawiione funkcja SetTimer w state
Initiate
{
if(bCheckEndMission)
{
bCheckEndMission=false;
if(!pPlayer.GetNumberOfUnits() && !pPlayer.GetNumberOfBuildings())
{
if(GetGoalState(backToBase)!=goalAchieved)
{
AddBriefing("translateFailedNoUnits",pPlayer.GetName());
EndMission(false);
}
else
EndMission(true);
}
if(pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings())
{
pPlayer.EnableBuilding("LCBLZ",true);
if(pPlayer.GetMoney()<500) pPlayer.AddMoney(500);
}
}
}
//----- UnitDestroyed -----
event UnitDestroyed(unit u_Unit)
{

```

```

bCheckEndMission=true;
}
//----- BuildingDestroyed -----
{
bCheckEndMission=true;
}
//----- EndMission -----
{
pNeutral.SetEnemy(pPlayer);
pPlayer.SetEnemy(pNeutral);
}
}



## Mission314Demo.ec


mission "translateMission314"
{
consts
{
deliverDocuments = 0;
backToBase = 1;
}
player pEnemy;
player pPlayer;
player pNeutral;
unitex uHero;
int bCheckEndMission;

state Initialize;
state ShowBriefing;
state Searching;
state Nothing;
//----- Initialize -----
{
//----- Goals -----
RegisterGoal(deliverDocuments,"translateGoal314");
RegisterGoal(backToBase,"translateGoalHeroBackToBase");
EnableGoal(deliverDocuments,true);
//----- Temporary players -----
//----- Players -----
pPlayer = GetPlayer(3);
pEnemy = GetPlayer(2);
pNeutral = GetPlayer(1);
//----- AI -----
if(GetDifficultyLevel()==0)
{
pEnemy.LoadScene("single\singleEasy");
pEnemy.EnableAIFeatures(aiControlDefense,false);
}
if(GetDifficultyLevel()==1)
pEnemy.LoadScene("single\singleMedium");
if(GetDifficultyLevel()==2)
pEnemy.LoadScene("single\singleHard");

pNeutral.LoadScene("single\singleHard");
pEnemy.EnableAIFeatures(aiControlOffense,false);
pNeutral.EnableAIFeatures(aiControlOffense,false);

pPlayer.EnableAIFeatures(aiEnabled,false);

pNeutral.SetNeutral(pPlayer);
pPlayer.SetNeutral(pNeutral);
pNeutral.ChooseEnemy(pEnemy);
pNeutral.SetEnemy(pEnemy);
//----- Money -----
pPlayer.SetMoney(0);
pEnemy.SetMoney(2000);
pNeutral.SetMoney(3000);
//----- Researches -----
pEnemy.EnableResearch("RES_ED_WCH2",true);
pEnemy.EnableResearch("RES_ED_ACH2",true);
pEnemy.EnableResearch("RES_MCH2",true);
pEnemy.EnableResearch("RES_ED_UA11",true);
//----- Buildings -----
pPlayer.EnableBuilding("LCBF",false);
pPlayer.EnableBuilding("LCBP",false);
pPlayer.EnableBuilding("LCBA",false);

```

```

pPlayer.EnableBuilding("LCBMR",false);
pPlayer.EnableBuilding("LCBSR",false);
pPlayer.EnableBuilding("LCBRC",false);
pPlayer.EnableBuilding("LCBAB",false);
pPlayer.EnableBuilding("LCBGA",false);
pPlayer.EnableBuilding("LCBDE",false);
pPlayer.EnableBuilding("LCBHQ",false);
pPlayer.EnableBuilding("LCBSD",false);
pPlayer.EnableBuilding("LCBWIC",false);
pPlayer.EnableBuilding("LCBSS",false);
pPlayer.EnableBuilding("LCBLZ",false);
pPlayer.EnableBuilding("LCLASERWALL",false);
//----- Units -----
uHero=pPlayer.GetScriptUnit();
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission=false;

//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;/5 sec
}
//----- Briefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing314",pPlayer.GetName());
    return Searching,100;
}
//----- Searching -----
state Searching
{
    if(uHero)
    {
        if(uHero.IsInWorld(GetWorldNum())
            && uHero.DistanceTo(GetPointX(0),GetPointY(0))<10
            && uHero.GetLocationZ())
        {
            SetGoalState(deliverDocuments,goalAchieved);
            EnableGoal(backToBase,true);
            AddBriefing("translateAccomplished314",pPlayer.GetName());
            return Nothing,200;
        }
    }
    return Searching,100;
}
//----- Nothing -----
state Nothing
{
    if(GetGoalState(backToBase)!=goalAchieved &&
uHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
        pPlayer.SetScriptData(1,1);
    }
    return Nothing, 500;
}
//----- Timer -----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if((pPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            pPlayer.SetScriptData(1,1);
            if(GetGoalState(backToBase)==goalAchieved)
            {
                AddBriefing("translateFailedNoUnits",pPlayer.GetName());
                EndMission(false);
            }
            else
                EndMission(true);
        }
        if(pPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            pPlayer.EnableBuilding("LCBLZ",true);
            if(pPlayer.GetMoney()<500) pPlayer.AddMoney(500);
        }
    }
}
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    pNeutral.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pNeutral);
}
}

Mission315.ec
mission "translateMission315"
{
    consts
    {
        destroyEnemyBase = 0;
    }

    player pEnemy1;
    player pEnemy2;
    player pPlayer;

    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(destroyEnemyBase,"translateGoal315");
    EnableGoal(destroyEnemyBase,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy1 = GetPlayer(2);
    pEnemy2 = GetPlayer(8);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        pEnemy1.LoadScript("single\singleEasy");
        pEnemy2.LoadScript("single\singleEasy");
        pEnemy2.EnableAIFeatures(aiControlDefense,false);
        pEnemy2.EnableAIFeatures(aiBuildBuildings,false);
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy1.LoadScript("single\singleMedium");
        pEnemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy1.LoadScript("single\singleHard");
        pEnemy2.LoadScript("single\singleHard");
    }
    pEnemy2.EnableAIFeatures(aiControlOffense,false);
    pPlayer.EnableAIFeatures(aiEnabled,false);
    //----- Money -----
    if(GetDifficultyLevel()==2)
        pPlayer.SetMoney(10000);
    else
        pPlayer.SetMoney(20000);
    pEnemy1.SetMoney(20000);
    pEnemy2.SetMoney(20000);
    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WSR2",true);
    pPlayer.EnableResearch("RES_LC_UNO2",true);
    pPlayer.EnableResearch("RES_LC_BMD",true);
}

```

```

pEnemy1.EnableResearch("RES_ED_WCA2",true);
pEnemy1.EnableResearch("RES_ED_MSC2",true);
pEnemy1.EnableResearch("RES_ED_UMT1",true);

pEnemy2.CopyResearches(pEnemy1);
//----- Buildings -----
//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission=false;

//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;/5 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing315",pPlayer.GetName());
    return Nothing, 500;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//----- event Timer0() //wolny co 100 cykli< ustawiione funkcja SetTimer w state Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(GetGoalState(destroyEnemyBase))=goalAchieved &&
lpEnemy2.GetNumberOfBuildings()
        {
            SetGoalState(destroyEnemyBase, goalAchieved);
            AddBriefing("translateAccomplished315",pPlayer.GetName());
            EnableEndMissionButton(true);
        }
        if(lpPlayer.GetNumberOfUnits() && !lpPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
}
}

state Initialize;
state ShowBriefing;
state Fighting;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(destroyEnemyBase,"translateGoal321");
    RegisterGoal(backToBase,"translateGoalHeroBackToBase");
    EnableGoal(destroyEnemyBase,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);

    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(2);

    //----- AI -----
    pPlayer.EnableAIFeatures(aiEnabled,false);
    if(GetDifficultyLevel()==0)
        pEnemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        pEnemy.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        pEnemy.LoadScript("single\singleHard");

    pPlayer.EnableAIFeatures(aiEnabled,false);
//pEnemy.AddToMainTargetList();

    //----- Money -----
    pPlayer.SetMoney(10000);
    pEnemy.SetMoney(20000);

    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WSL1",true);
    pPlayer.EnableResearch("RES_LC_BHD",true);

    pEnemy.EnableResearch("RES_ED_WSL1",true);
    //----- Buildings -----
    //----- Units -----
    uHero=pPlayer.GetScriptUnit(0);
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    SetTimer(1,10000);
    //----- Variables -----
    bCheckEndMission=false;
    bBlowUpBase=true;
    nBlowUpCounter=0;
    //----- Camera -----
    CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,200;/15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing321a",pPlayer.GetName());
    return Fighting,100;
}
//-----
state Fighting
{
    int timeToExplode;
    if(bBlowUpBase &&
uHero.IsInWorld(GetWorldNum()) &&
uHero.DistanceTo((GetPointX(0),GetPointY(0))<10 &&
uHero.GetLocationZ()))
    {
        bBlowUpBase=false;
        if(GetDifficultyLevel()==0)
            nBlowUpCounter=30;
        if(GetDifficultyLevel()==1)
            nBlowUpCounter=20;
        if(GetDifficultyLevel()==2)
            nBlowUpCounter=10;
        AddBriefing("translateBriefing321b",pPlayer.GetName());
        return Fighting,20;
    }
    if(nBlowUpCounter)
}

```

Mission321.ec

```

mission "translateMission321"
{
    consts
    {
        destroyEnemyBase = 0;
        backToBase = 1;
    }

    player pEnemy;
    player pPlayer;

    unitex uHero;

    int bBlowUpBase;
    int nBlowUpCounter;
    int bCheckEndMission;
}
```

```

    {
        nBlowUpCounter=nBlowUpCounter-1;
        if((nBlowUpCounter)
        {
            KillArea(4,GetPointX(0),GetPointY(0),0,20-(GetDifficultyLevel()*5));
            SetConsoleText("");
        }
        else
            SetConsoleText("translateMessage321",nBlowUpCounter);
        return Fighting,20;
    }
    return Fighting,200;
}
//-----
state Nothing
{
    if(GetGoalState(backToBase)|=goalAchieved &&
luHero.lshWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(GetGoalState(destroyEnemyBase)|=goalAchieved &&
lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }

        if(GetGoalState(destroyEnemyBase)|=goalAchieved &&
lpEnemy.GetNumberOfBuildings())
        {
            SetGoalState(destroyEnemyBase,goalAchieved);
            EnableGoal(backToBase,true);
            AddBriefing("translateAccomplished321",pPlayer.GetName());
            state Nothing;
        }
    }
}
//-----
event Timer1() //wolany co 6000 cykli
{
    Snow(GetPointX(0),GetPointY(0),30,400,2500,800,3);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

Mission322.ec
mission "translateMission322"
{
    consts
    {
        meetConvoy = 0;
        escortConvoy = 1;
        backToBase = 2;
    }

    player pPlayer;
    player pConvoy;
    player pUCS1;
    player pUCS2;
    player pEnemy1;
    player pEnemy2;
    player pEnemy3;
}

unitex uHero;
unitex uConvoyCraft1;
unitex uConvoyCraft2;
unitex uConvoyCraft3;

int bTestPhase;
int b1Ready;
int b2Ready;
int b3Ready;

state Initialize;
state ShowBriefing;
state ReachBase1;
state ReachBase2;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(meetConvoy,"translateGoal322a");
    if(GetDifficultyLevel()==0)
        RegisterGoal(escortConvoy,"translateGoal322b",33);
    if(GetDifficultyLevel()==1)
        RegisterGoal(escortConvoy,"translateGoal322b",66);
    if(GetDifficultyLevel()==2)
        RegisterGoal(escortConvoy,"translateGoal322b",100);
    RegisterGoal(backToBase,"translateGoalHeroBackToBase",0);
    EnableGoal(meetConvoy,true);
    //----- Temporary players -----
    // tmpPlayer = GetPlayer(2);
    // tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pConvoy = GetPlayer(6);
    pUCS1 = GetPlayer(1);
    pUCS2 = GetPlayer(4);
    pEnemy1 = GetPlayer(2);
    pEnemy2 = GetPlayer(7);
    pEnemy3 = GetPlayer(8);

    //----- AI -----
    pEnemy2.EnableStatistics(false);
    pEnemy3.EnableStatistics(false);

    pEnemy1.LoadScript("single\singleEasy");
    pUCS1.LoadScript("single\singleEasy");
    pUCS2.LoadScript("single\singleMedium");
    pConvoy.LoadScript("single\singleHard");

    pPlayer.EnableAIFeatures(aiEnabled,false);

    pEnemy1.EnableAIFeatures(aiControlOffense,false);
    pEnemy2.EnableAIFeatures(aiRush,false);
    pEnemy3.EnableAIFeatures(aiRush,false);

    pEnemy2.SetMaxTankPlatoonSize(4);
    pEnemy2.SetNumberOffensiveTankPlatoons(3);
    pEnemy2.SetNumberDefensiveTankPlatoons(0);

    pEnemy3.SetMaxTankPlatoonSize(4);
    pEnemy3.SetNumberOffensiveTankPlatoons(3);
    pEnemy3.SetNumberDefensiveTankPlatoons(0);

    pEnemy1.SetNeutral(pEnemy2);
    pEnemy1.SetNeutral(pEnemy3);

    pEnemy2.SetNeutral(pEnemy1);
    pEnemy2.SetNeutral(pEnemy3);

    pEnemy3.SetNeutral(pEnemy1);
    pEnemy3.SetNeutral(pEnemy2);

    pConvoy.EnableAIFeatures(aiEnabled,false);
    pUCS1.EnableAIFeatures(aiControlOffense,false);
    pUCS2.EnableAIFeatures(aiControlOffense,false);

    pUCS1.EnableAIFeatures(aiRejectAlliance,false);
    pUCS2.EnableAIFeatures(aiRejectAlliance,false);
    pConvoy.EnableAIFeatures(aiRejectAlliance,false);

    pPlayer.SetAlly(pUCS1);
    pPlayer.SetAlly(pUCS2);
    pPlayer.SetAlly(pConvoy);
}

```

```

pUCS1.SetAlly(pUCS2);
pUCS1.SetAlly(pConvoy);
pUCS2.SetAlly(pConvoy);

pPlayer.EnableAIFeatures(aiEnabled,false);
//----- Money -----
pPlayer.SetMoney(0);
pEnemy1.SetMoney(20000);
//----- Researches -----
pPlayer.EnableResearch("RES_MSR3",true);
pPlayer.EnableResearch("RES_LC_ASR1",true);
pPlayer.EnableResearch("RES_LC_SOB2",true);

pUCS1.EnableResearch("RES_UCS_WSP1",true);
pUCS1.EnableResearch("RES_MSR2",true);
pUCS1.EnableResearch("RES_UCS_BMD",true);

pEnemy1.EnableResearch("RES_ED_WSR1",true);
//----- Buildings -----
pPlayer.EnableBuilding("LCBBF",false);
pPlayer.EnableBuilding("LCBPP",false);
pPlayer.EnableBuilding("LCBBA",false);
pPlayer.EnableBuilding("LCBMR",false);
pPlayer.EnableBuilding("LCBSR",false);
pPlayer.EnableBuilding("LCBRC",false);
pPlayer.EnableBuilding("LCBAB",false);
pPlayer.EnableBuilding("LCBGA",false);
pPlayer.EnableBuilding("LCBDE",false);
pPlayer.EnableBuilding("LCBHQ",false);
pPlayer.EnableBuilding("LCBSD",false);
pPlayer.EnableBuilding("LCBWC",false);
pPlayer.EnableBuilding("LCBSS",false);
pPlayer.EnableBuilding("LCBLZ",true);
pPlayer.EnableBuilding("LLASERWALL",false);
//----- Units -----
uHero=pPlayer.GetScriptUnit(0);

uConvoyCraft1=GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));
uConvoyCraft2=GetUnit(GetPointX(1),GetPointY(1),GetPointZ(1));
uConvoyCraft3=GetUnit(GetPointX(2),GetPointY(2),GetPointZ(2));

//----- Artefacts -----
//----- Timers -----
//----- Variables -----
bTestPhase=0;
b1Ready = false;
b2Ready = false;
b3Ready = false;

//----- Camera -----
CallCamera(1);

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;//5 sec
}
}
state ShowBriefing
{
    AddBriefing("translateBriefing322a",pPlayer.GetName());//dojedz do bazy 1 tam czeka konwoj

    if(GetDifficultyLevel()==1)
    {
        pEnemy2.RussianAttack(GetPointX(4),GetPointY(4),0);
    }

    if(GetDifficultyLevel()==2)
    {
        pEnemy2.RussianAttack(GetPointX(4),GetPointY(4),0);
        pEnemy3.RussianAttack(GetPointX(5),GetPointY(5),0);
    }

    pEnemy2.EnableAIFeatures(aiEnabled,false);
    pEnemy3.EnableAIFeatures(aiEnabled,false);
    pConvoy.EnableAIFeatures(aiEnabled,false);
    return ReachBase1,100;
}
}
state ReachBase1
{
    if(uHero.IsInWorld(GetWorldNum()) && uHero.DistanceTo((getPointX(0),getPointY(0))<5)
    SetGoalState(meetConvoy.goalAchieved);
    EnableGoal(escortConvoy,true);
    if(GetDifficultyLevel()==0)
        AddBriefing("translateBriefing322b",pPlayer.GetName(),33);//oto dwaj konwoj zaprowadz do bazy 2 33% musi przeczy
    if(GetDifficultyLevel()==1)
        AddBriefing("translateBriefing322b",pPlayer.GetName(),66);//oto dwaj konwoj zaprowadz go do bazy 2 66% musi przeczy
    if(GetDifficultyLevel()==2)
        AddBriefing("translateBriefing322b",pPlayer.GetName(),100);//oto dwaj konwoj zaprowadz go do bazy 2 100% musi przeczy
    return ReachBase2,100;
}
}
state ReachBase2
{
    if(uHero.IsInWorld(GetWorldNum()))
    {
        if(uConvoyCraft1.IsLive())
            uConvoyCraft1.CommandMove(uHero.GetLocationX(),1,uHero.GetLocationY(),uHero.GetLocationZ());
        if(uConvoyCraft2.IsLive())
            uConvoyCraft2.CommandMove(uHero.GetLocationX(),2,uHero.GetLocationY(),uHero.GetLocationZ());
        if(uConvoyCraft3.IsLive())
            uConvoyCraft3.CommandMove(uHero.GetLocationX(),3,uHero.GetLocationY(),uHero.GetLocationZ());
    }

    if(!b1Ready && uConvoyCraft1.IsLive() && uConvoyCraft1.DistanceTo((GetPointX(3),GetPointY(3))<15)
    {
        b1Ready = true;
    }

    if(!b2Ready && uConvoyCraft2.IsLive() && uConvoyCraft2.DistanceTo((GetPointX(3),GetPointY(3))<15)
    {
        b2Ready = true;
    }

    if(!b3Ready && uConvoyCraft3.IsLive() && uConvoyCraft3.DistanceTo((GetPointX(3),GetPointY(3))<15)
    {
        b3Ready = true;
    }

    if((GetDifficultyLevel()==1 && pConvoy.GetNumberOfUnits(<2) || GetDifficultyLevel()==2 && pConvoy.GetNumberOfUnits(<3) || pConvoy.GetNumberOfUnits()==0)
    {
        SetGoalState(escortConvoy.goalFailed);
        EnableGoal(backToBase,true);
        AddBriefing("translateFailed322",pPlayer.GetName());//convoy destroyed
        return Nothing,100;
    }

    if((b1Ready || !uConvoyCraft1.IsLive()) && (b2Ready || !uConvoyCraft2.IsLive()) && (b3Ready || !uConvoyCraft3.IsLive()))
    {
        if(uConvoyCraft1.IsLive()) uConvoyCraft1.ChangePlayer(pUCS2);
        if(uConvoyCraft2.IsLive()) uConvoyCraft2.ChangePlayer(pUCS2);
        if(uConvoyCraft3.IsLive()) uConvoyCraft3.ChangePlayer(pUCS2);
        pUCS2.SetMoney(20000);
        SetGoalState(escortConvoy.goalAchieved);
        EnableGoal(backToBase,true);
        AddBriefing("translateAccomplished322",pPlayer.GetName());//convoy on
    }

    EnableNextMission(0,true);
    return Nothing,100;
}

return ReachBase2,100;
}

state Nothing
{
    if((GetGoalState(backToBase)!=goalAchieved && uHero.IsInWorld(GetWorldNum())))
    {
        SetGoalState(backToBase.goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}

```

```

        }
    }-----  

    event UnitDestroyed(unit u_Unit)  

    {  

    }-----  

    event BuildingDestroyed(unit u_Unit)  

    {  

    }-----  

    event EndMission()  

    {  

        pUCS1.SetEnemy(pPlayer);  

        pPlayer.SetEnemy(pUCS1);  

        pUCS2.SetEnemy(pPlayer);  

        pPlayer.SetEnemy(pUCS2);  

        pConvoy.SetEnemy(pPlayer);  

        pPlayer.SetEnemy(pConvoy);  

    }-----  

}



## Mission323.ec


mission "translateMission323"
{
    consts
    {
        makeTests = 0;
        backToBase = 1;
    }

    player pPlayer;
    player pNeutral;
    player pTowers;
    player pEnemy;

    unitex uPrototype;
    unitex uTester;

    int bTestPhase;

    state Initialize;
    state ShowBriefing;
    state Testing;
    state Nothing;
}-----  

state Initialize
{
    player tmpPlayer;
    //----- Goals -----  

    RegisterGoal(makeTests,"translateGoal323");
    EnableGoal(makeTests,true);
    //----- Temporary players -----  

    tmpPlayer = GetPlayer(2);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----  

    pPlayer = GetPlayer(3);
    pNeutral = GetPlayer(5);
    pTowers = GetPlayer(10);
    pEnemy = GetPlayer(1);
  

    //----- AI -----  

    pPlayer.EnableAIFeatures(aiEnabled,false);
    pNeutral.EnableAIFeatures(aiEnabled,false);
    pTowers.EnableAIFeatures(aiEnabled,false);
    pEnemy.EnableAIFeatures(aiEnabled,false);

    pPlayer.EnableAIFeatures(aiRejectAlliance,false);
    pTowers.EnableAIFeatures(aiRejectAlliance,false);
    pEnemy.EnableAIFeatures(aiRejectAlliance,false);

    pPlayer.SetAlly(pNeutral);
    pPlayer.SetAlly(pTowers);

    pEnemy.SetNeutral(pPlayer);
    pEnemy.SetNeutral(pNeutral);
}-----  

pNeutral.SetNeutral(pTowers);
//----- Money -----
pPlayer.SetMoney(0);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_SGEN",true);
//----- Buildings -----
  

pPlayer.EnableBuilding("LCBF",false);
pPlayer.EnableBuilding("LCBP",false);
pPlayer.EnableBuilding("LCBA",false);
pPlayer.EnableBuilding("LCMR",false);
pPlayer.EnableBuilding("LCSR",false);
pPlayer.EnableBuilding("LCRC",false);
pPlayer.EnableBuilding("LCAB",false);
pPlayer.EnableBuilding("LCGA",false);
pPlayer.EnableBuilding("LCDE",false);
pPlayer.EnableBuilding("LCHQ",false);
pPlayer.EnableBuilding("LCBS",false);
pPlayer.EnableBuilding("LCWC",false);
pPlayer.EnableBuilding("LCBS",false);
pPlayer.EnableBuilding("LCBLZ",false);
pPlayer.EnableBuilding("LCLASERWALL",false);
//----- Units -----
uPrototypeOf = GetUnit(GetPointX(1),GetPointY(1),GetPointZ(0));
uTester = GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));
CallCamera();
SelectUnit(uPrototypeOf,false);
//----- Artifacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bTestPhase=0;
  

//----- Camera -----
CallCamera();
pPlayer.LookAt(GetPointX(1),GetPointY(1),6,0,20,0);
return ShowBriefing,100;//5 sec
}-----  

state ShowBriefing
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing323a",pPlayer.GetName());
    return Testing,100;
}-----  

state Testing
{
    if(bTestPhase==0)
    {
        bTestPhase=1;
    }

    if(bTestPhase==1)
    {
        AddBriefing("translateBriefing323b",pPlayer.GetName());//start speed test
        uPrototypeOf.CommandMove(GetPointX(2),GetPointY(2),0);
        uTester.CommandMove(GetPointX(3),GetPointY(3),0);
        bTestPhase=2;
        return Testing,100;
    }

    if(bTestPhase==2 && uPrototypeOf.DistanceTo(GetPointX(2),GetPointY(2))==0)
    {
        AddBriefing("translateBriefing323c",pPlayer.GetName());//end speed test
        start hp test
        uPrototypeOf.CommandMove(GetPointX(6),GetPointY(6),0);
        bTestPhase=3;
        return Testing,100;
    }

    if(bTestPhase==3 && uPrototypeOf.DistanceTo(GetPointX(6),GetPointY(6))==0)
    {
        bTestPhase=4;
        return Testing,100;
    }

    if(bTestPhase==4)
    {
        uPrototypeOf.CommandMove(GetPointX(4),GetPointY(4),0);
        bTestPhase=5;
        return Testing,100;
    }
}

```

```

        }

        if(bTestPhase==5 &&
           uPrototype.DistanceTo(GetPointX(4),GetPointY(4))==0)
        {
            AddBriefing("translateBriefing323d",pPlayer.GetName());//end hp test
            start fire test
            uPrototype.CommandMove(GetPointX(5),GetPointY(5),0);
            uPrototype.ChangePlayer(pPlayer);
            CallCamera();
            SelectUnit(uPrototype,false);
            bTestPhase=6;
            return Testing,100;
        }

        if(bTestPhase==6 && pEnemy.GetNumberOfUnits()<3)
        {
            KillArea(8,uPrototype.GetLocationX(),uPrototype.GetLocationY(),0,1);
            bTestPhase=7;
            return Testing,100;
        }

        if(bTestPhase==7)
        {
            SetGoalState(makeTests, goalFailed);
            AddBriefing("translateAccomplished323",pPlayer.GetName());//end tests
            EnableEndMissionButton(true,false);
            return Nothing,100;
        }
        return Testing,100;
    }
}

state Nothing
{
    return Nothing, 500;
}
}

event Timer0() //wolany co 100 cykli< ustawnione funkcja SetTimer w state
Initialize
{
}
}

//-----
event UnitDestroyed(unit u_Unit)
{
}

//-----
event BuildingDestroyed(unit u_Unit)
{
}

//-----
event EndMission()
{
    pNeutral.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pNeutral);

    pTowers.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pTowers);

    pEnemy.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pEnemy);
}
}

```

Mission331.ec

```

mission "translateMission331"
{
    consts
    {
        meetConvoy = 0;
        escortConvoy = 1;
        backToBase = 2;
    }

    player pPlayer;
    player pConvoy;
    player pEnemy;

    unitex uHero;

    unitex uConvoyCraft1;
    unitex uConvoyCraft2;
    unitex uConvoyCraft3;
    unitex uConvoyCraft4;

    int bEDAsk;
}

int b1Ready;
int b2Ready;
int b3Ready;
int b4Ready;

state Initialize;
state ShowBriefing;
state ReachBase1;
state ReachBase2;
state Nothing;
//-----
state Initialize
{
    //player tmpPlayer;
    //----- Goals -----
    RegisterGoal(meetConvoy,"translateGoal331a");
    RegisterGoal(escortConvoy,"translateGoal331b",100);
    RegisterGoal(backToBase,"translateGoalHeroBackToBase");
    EnableGoal(meetConvoy,true);
    //----- Temporary players -----
    // tmpPlayer = GetPlayer(2);
    // tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pConvoy = GetPlayer(1);
    pEnemy = GetPlayer(2);

    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(20000);
    pPlayer.EnableAIFeatures(aiEnabled,false);
    pEnemy.EnableAIFeatures(aiEnabled,false);
    pConvoy.EnableAIFeatures(aiEnabled,false);

    pConvoy.EnableAIFeatures(aiRejectAlliance,false);

    pPlayer.SetAlly(pConvoy);
    //----- Money -----
    pPlayer.SetMoney(0);
    pConvoy.SetMoney(20000);
    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_REG1",true);
    pConvoy.EnableResearch("RES_UCS_UAH1",true);
    //----- Buildings
    pPlayer.EnableBuilding("LCBBF",false);
    pPlayer.EnableBuilding("LCBP",false);
    pPlayer.EnableBuilding("LCBBA",false);
    pPlayer.EnableBuilding("LCBMR",false);
    pPlayer.EnableBuilding("LCBSR",false);
    pPlayer.EnableBuilding("LCBCF",false);
    pPlayer.EnableBuilding("LCBAB",false);
    pPlayer.EnableBuilding("LCBGA",false);
    pPlayer.EnableBuilding("LCBDE",false);
    pPlayer.EnableBuilding("LCBHQ",false);
    pPlayer.EnableBuilding("LCBSD",false);
    pPlayer.EnableBuilding("LCBWC",false);
    pPlayer.EnableBuilding("LCBSS",false);
    pPlayer.EnableBuilding("LCBLZ",true);
    pPlayer.EnableBuilding("LCLASERWALL",false);
    //----- Units -----
    uHero=pPlayer.GetScriptedUnit(0);

    uConvoyCraft1 = GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));
    uConvoyCraft2 = GetUnit(GetPointX(1),GetPointY(1),GetPointZ(1));
    uConvoyCraft3 = GetUnit(GetPointX(2),GetPointY(2),GetPointZ(2));
    uConvoyCraft4 = GetUnit(GetPointX(3),GetPointY(3),GetPointZ(3));
    //----- Artifacts -----
    //----- Timers -----
    //----- Variables -----
    bEDAsk=true;
    b1Ready = false;
    b2Ready = false;
    b3Ready = false;
    b4Ready = false;
    //----- Camera -----
    CallCamera();

    pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
    return ShowBriefing,100;/5 sec
}
}

state ShowBriefing
{
    AddBriefing("translateBriefing331a",pPlayer.GetName());//dojedz do bazy 1 tam czeka konwo
    return ReachBase1,100;
}

```

```

//-----
state ReachBase1
{
    if(uHero.IsInWorld(GetWorldNum()) &&
       uHero.DistanceTo((GetPointX(0),GetPointY(0))<5)
    {
        SetGoalState(meetConvoy.goalAchieved);
        EnableGoal(escortConvoy,true);
        if(GetDifficultyLevel()==0)
            AddBriefing("translateBriefing331b",pPlayer.GetName());//oto twoj
        konwoj zaprowadz go do bazy 2 100% musi przebyc
        return ReachBase2,100;
    }
    return ReachBase1,100;
}
//-----
state ReachBase2
{
    if(bEDAsk && GetMissionTime()>6000)//5 min
    {
        bEDAsk=false;
        AddBriefing("translateBriefing331c",pPlayer.GetName());//ED here
        destroy this convoy and join us.
    }

    if(!uConvoyCraft1.IsLive() && !uConvoyCraft2.IsLive() &&
       !uConvoyCraft3.IsLive() && !uConvoyCraft4.IsLive())
    {
        SetGoalState(escortConvoy.goalFailed);
        EnableGoal(backToBase,true);
        pEnemy.EnableAIFeatures(aiRejectAlliance,false);
        pPlayer.SetAll(pEnemy);
        pConvoy.SetEnemy(pPlayer);
        pPlayer.SetEnemy(pConvoy);
        AddBriefing("translateAccomplished331b",pPlayer.GetName());//UCS con-
voy destroyed
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        return Nothing,100;
    }

    if(uHero.IsInWorld(GetWorldNum()))
    {
        if(uConvoyCraft1.IsLive())
            uConvoyCraft1.CommandMove(uHero.GetLocationX())-
1,uHero.GetLocationY(),uHero.GetLocationZ());
        if(uConvoyCraft2.IsLive())
            uConvoyCraft2.CommandMove(uHero.GetLocationX())-
2,uHero.GetLocationY(),uHero.GetLocationZ());
        if(uConvoyCraft3.IsLive())
            uConvoyCraft3.CommandMove(uHero.GetLocationX(),uHero.GetLocationY())-
1,uHero.GetLocationZ());
        if(uConvoyCraft4.IsLive())
            uConvoyCraft4.CommandMove(uHero.GetLocationX(),uHero.GetLocationY())-
2,uHero.GetLocationZ());
    }

    if(lb1Ready && uConvoyCraft1.IsLive() &&
       uConvoyCraft1.DistanceTo((GetPointX(4),GetPointY(4))<15)
    {
        b1Ready = true;
    }
    if(lb2Ready && uConvoyCraft2.IsLive() &&
       uConvoyCraft2.DistanceTo((GetPointX(4),GetPointY(4))<15)
    {
        b2Ready = true;
    }
    if(lb3Ready && uConvoyCraft3.IsLive() &&
       uConvoyCraft3.DistanceTo((GetPointX(4),GetPointY(4))<15)
    {
        b3Ready = true;
    }
    if(lb4Ready && uConvoyCraft4.IsLive() &&
       uConvoyCraft4.DistanceTo((GetPointX(4),GetPointY(4))<15)
    {
        b4Ready = true;
    }

    if(b1Ready && b2Ready && b3Ready && b4Ready)
    {
        SetGoalState(meetConvoy.goalAchieved);
        pConvoy.EnableAIFeatures(aiEnabled,true);
        pConvoy.EnableAIFeatures(aiControlOffense,false);
        EnableGoal(backToBase,true);
    }
}

AddBriefing("translateAccomplished331a",pPlayer.GetName());//convoy
on place UCS alliance still kept
EnableNextMission(2,true);
EnableNextMission(3,true);
return Nothing,100;
}
return ReachBase2,100;
}

//-----
state Nothing
{
    if(GetGoalState(backToBase)!==goalAchieved &&
       uHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}
//-----
event UnitDestroyed(unit u_Unit)
{
}
//-----
event BuildingDestroyed(unit u_Unit)
{
}
//-----
event EndMission()
{
    pEnemy.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pEnemy);

    pConvoy.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pConvoy);
}
}



## Mission333.ec


mission "translateMission333"
{
    consts
    {
        destroyPrototype = 0;
        backToBase = 1;
    }

    player pPlayer;
    player pTraitor;
    player pEnemy;

    unitex uPrototype;
    unitex uHero;

    int nReflex;
    int nWayPoint;

    state Initialize;
    state ShowBriefing;
    state Preparing;
    state Runaway;
    state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal("destroyPrototype","translateGoal333");//destroy prototype
    RegisterGoal("backToBase","translateGoalHeroBackToBase",0);
    EnableGoal("destroyPrototype,true");
    //----- Temporary players -----
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pTraitor = GetPlayer(5);
    pEnemy = GetPlayer(2);

    //----- AI -----
    pPlayer.EnableStatistics(false);
    pTraitor.EnableStatistics(false);
    pEnemy.EnableStatistics(false);
}
}

```

```

pPlayer.EnableAIFeatures(aiEnabled,false);
pTraitor.EnableAIFeatures(aiEnabled,false);

pEnemy.SetNeutral(pTraitor);
pTraitor.SetNeutral(pEnemy);

pEnemy.EnableAIFeatures(aiControlOffense,false);
pEnemy.SetMaxTankPlatoonSize(6);
pEnemy.SetMaxHelicopterPlatoonSize(0);
pEnemy.SetMaxShipPlatoonSize(0);

pEnemy.SetNumberOfOffensiveTankPlatoons(10);
pEnemy.SetNumberOfOffensiveShipPlatoons(0);
pEnemy.SetNumberOfOffensiveHelicopterPlatoons(0);

pEnemy.SetNumberOfDefensiveTankPlatoons(0);
pEnemy.SetNumberOfDefensiveShipPlatoons(0);
pEnemy.SetNumberOfDefensiveHelicopterPlatoons(0);
pEnemy.SetMaxAttackFrequency(200);

//----- Money -----
pPlayer.SetMoney(0);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_WSS1",true);
pPlayer.EnableResearch("RES_LC_WSL1",true);
pPlayer.EnableResearch("RES_LC_BWC",true);
pPlayer.EnableResearch("RES_LC_MGEN",true);
pPlayer.EnableResearch("RES_LC_BHD",true);
//----- Buildings -----
pPlayer.EnableBuilding("LCBBP",false);
pPlayer.EnableBuilding("LCBPP",false);
pPlayer.EnableBuilding("LCBAA",false);
pPlayer.EnableBuilding("LCBMR",false);
pPlayer.EnableBuilding("LCBSR",false);
pPlayer.EnableBuilding("LCBRC",false);
pPlayer.EnableBuilding("LCBAB",false);
pPlayer.EnableBuilding("LCBGA",false);
pPlayer.EnableBuilding("LCBDE",false);
pPlayer.EnableBuilding("LCBHQ",false);
pPlayer.EnableBuilding("LCBSD",false);
pPlayer.EnableBuilding("LCBW",false);
pPlayer.EnableBuilding("LCBSS",false);
pPlayer.EnableBuilding("LCLBLZ",false);
pPlayer.EnableBuilding("LCLASERWAL",false);
//----- Units -----
uPrototype = GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));
uHero = pPlayer.GetScriptUnit(0);
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
nWayPoint=1;
//----- Camera -----
CallCamera();
pPlayer.LookAt(GetPointX(0),GetPointY(0),6,0,20,0);
return ShowBriefing,100//5 sec
}
//----- Show Briefing -----
state ShowBriefing
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing333",pPlayer.GetName());//zdrada!!!! zabij go
tyko hero moze to uzytic
    return Preparing,100;
}
//----- State Preparing -----
state Preparing
{
    if(pPlayer.GetNumberofUnits()>0)
    {
        KillArea(8,uPrototype.GetLocationX(),uPrototype.GetLocationY(),0,15);
        CallCamera();
        pPlayer.LookAt(GetPointX(0)+3,GetPointY(0)-5,8,0,20,0);
        return Runaway,100;
    }
    return Preparing,100;
}
//----- State Runaway -----
state Runaway
{
    if(uHero.DistanceTo(GetPointX(nWayPoint),GetPointY(nWayPoint))<15
    || pPlayer.IsPointLocated(GetPointX(nWayPoint),GetPointY(nWayPoint),0))
    {
        nWayPoint=nWayPoint+1;
        if(nWayPoint>5) nWayPoint=5;
        if(uPrototype.DistanceTo(GetPointX(5),GetPointY(5))==0 &&
uPrototype.GetLocationZ() == 1)
        {
            pEnemy.EnableAIFeatures(aiControlOffense,true);
            KillArea(4,GetPointX(6),GetPointY(6),0,1);
            SetGoalState(destroyPrototype,goalAchieved);
            EnableGoal(backToBase,true);
            AddBriefing("translateFailed333",pPlayer.GetName());//prototype esca-
ped
            return Nothing,100;
        }
        if(!uPrototype.IsLive())
        {
            pEnemy.EnableAIFeatures(aiControlOffense,true);
            SetGoalState(destroyPrototype,goalAchieved);
            EnableGoal(backToBase,true);
            AddBriefing("translateAccomplished333",pPlayer.GetName());//prototype
destroyed
            return Nothing,100;
        }
    }
    uPrototype.CommandMove(GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(
nWayPoint));
    if(GetDifficultyLevel()==0)
        return Runaway,400;//EASY
    if(GetDifficultyLevel()==1)
        return Runaway,200;//MEDIUM
    return Runaway,100;//HARD
}
//----- State Nothing -----
state Nothing
{
    if(GetGoalState(backToBase)!=goalAchieved &&
uHero.InWorld(GetWorldNum()))
    {
        ShowVideo("CS312");
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}

//----- Event Timer0 -----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}
//----- Event UnitDestroyed -----
event UnitDestroyed(unit u_Unit)
{
}
//----- Event BuildingDestroyed -----
event BuildingDestroyed(unit u_Unit)
{
}
//----- Event EndMission -----
event EndMission()
{
    pEnemy.SetEnemy(pTraitor);
    pTraitor.SetEnemy(pEnemy);
}
}



## Mission334.ec


mission "translateMission334"
{
    consts
    {
        sendToBase50000 = 0;
        backToBase = 1;
    }
    player pEnemy;
    player pEDStrikeForces;
    player pPlayer;
    unitex uHero;
    int bShowFailed;
    int bCheckEndMission;
}

```

```

int nAttackCounter;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal("sendToBase50000", "translateGoalSend50000", 0);
    RegisterGoal("backToBase", "translateGoalHeroBackToBase");
    EnableGoal("sendToBase50000", true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(2);
    pEDStrikeForces = GetPlayer(4);
    //----- AI -----
    if(GetDifficultyLevel() == 0)
        pEnemy.LoadScript("single\\singleEasy");
    if(GetDifficultyLevel() == 1)
        pEnemy.LoadScript("single\\singleMedium");
    if(GetDifficultyLevel() == 2)
        pEnemy.LoadScript("single\\singleHard");

    pEDStrikeForces.LoadScript("single\\singleHard");
    pEDStrikeForces.SetMaxTankPlatoonSize(6);
    pEDStrikeForces.SetNumberOffensiveTankPlatoons(10);
    pEDStrikeForces.SetNumberOfDefensiveTankPlatoons(0);
    pEDStrikeForces.EnableAIFeatures(aiControlOffense, false);
    pEDStrikeForces.EnableAIFeatures(aiControlDefense, false);
    pEDStrikeForces.SetNeutral(pEnemy);
    pEnemy.SetNeutral(pEDStrikeForces);

    pEnemy.EnableAIFeatures(aiControlOffense, false);

    pPlayer.EnableAIFeatures(aiEnabled, false);
    //----- Money -----
    pPlayer.SetMoney(10000);
    pEnemy.SetMoney(20000);

    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WSL2", true);
    pPlayer.EnableResearch("RES_MSR4", true);
    pPlayer.EnableResearch("RES_LC_SHR1", true);

    pEnemy.EnableResearch("RES_ED_WSL1", true);
    //----- Buildings -----
    //----- Units -----
    uHero = pPlayer.GetScriptUnit(0);
    //----- Artefacts -----
    //----- Variables -----
    bCheckEndMission = false;
    bShowFailed = true;
    if(GetDifficultyLevel() == 0)
        nAttackCounter = 48 / hours;
    if(GetDifficultyLevel() == 1)
        nAttackCounter = 36;
    if(GetDifficultyLevel() == 2)
        nAttackCounter = 24;

    //----- Timers -----
    SetTimer(0, 100);
    SetTimer(1, 16383 / 24);
    //----- Camera -----
    CallCamera();
}

pPlayer.LookAt(pPlayer.GetStartingPointX(), pPlayer.GetStartingPointY(), 6, 0, 20, 0);
return ShowBriefing, 200; // 15 sec
}
//-----
state ShowBriefing
{
    EnableNextMission(0, true);

    Rain(pPlayer.GetStartingPointX(), pPlayer.GetStartingPointY(), 10, 400, 5000, 800, 5);
    AddBriefing("translateBriefing334", pPlayer.GetName(), nAttackCounter); // masz
    //tyko <0> hours for destroying the bridges
}

return Mining, 200;
}
//-----
state Mining
{
    if(pPlayer.GetMoneySentToBase() >= 50000)
    {
        SetGoalState("sendToBase50000", goalAchieved);
        EnableGoal("backToBase", true);
        AddBriefing("translateAccomplished334", pPlayer.GetName());
        return Nothing, 500;
    }
    return Mining, 200;
}
//-----
state Nothing
{
    if(GetGoalState("backToBase") != goalAchieved &&
    luHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState("backToBase", goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}
//-----
event Timer0() // wolany co 100 cykli< ustwione funkcja SetTimer w state
Initialize
{
    RegisterGoal("sendToBase50000", "translateGoalSend50000", pPlayer.GetMoneySentToBase());
    if(bShowFailed)
    {
        if(GetGoalState("sendToBase50000") != goalAchieved)
        {
            if(ResourcesLeftInMoney() + pPlayer.GetMoneySentToBase() + pPlayer.GetMoney() < 40000)
            {
                bShowFailed = false;
                SetGoalState("sendToBase50000", goalFailed);
                EnableGoal("backToBase", true);
                AddBriefing("translateFailed334", pPlayer.GetName());
                state Nothing;
            }
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission = false;
        if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailedNoUnits", pPlayer.GetName());
            EndMission(false);
        }
    }
}
//-----
event Timer1() // wolany co 1
{
    if(nAttackCounter > 0)
    {
        nAttackCounter = nAttackCounter - 1;
    }
    SetConsoleText("translateMessage334", nAttackCounter / 24, nAttackCounter % 24);
    if(nAttackCounter)
    {
        pEDStrikeForces.RussianAttack(pPlayer.GetStartingPointX(), pPlayer.GetStartingPointY(), 0, 0);
        AddBriefing("translateBriefing334b", pPlayer.GetName());
        pEnemy.EnableAIFeatures(aiControlOffense, true);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}

```

```

//-----
}



## Mission341.ec


mission "translateMission341"
{/Rio
consts
{
    escortNeo = 0;
    destroyEnemyBase = 1;
    backToBase = 2;
}

player pPlayer;
player pNeo;
player pEnemy;

unitex uHero;
unitex uNeo;

int bCaptureMechs;
int bCheckEndMission;

state Initialize;
state ShowBriefing;
state ReachEnemyBase;
state DestroyEnemyBase;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(escortNeo,"translateGoal341a");
    RegisterGoal(destroyEnemyBase,"translateGoal341b");
    RegisterGoal(backToBase,"translateGoalHeroBackToBase",0);
    EnableGoal(escortNeo,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(2);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pNeo = GetPlayer(8);
    pEnemy = GetPlayer(1);

    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(30000);
    pPlayer.EnableAIFeatures(aiEnabled,false);

    pNeo.EnableStatistics(false);
    pNeo.EnableAIFeatures(aiEnabled,false);
    pNeo.EnableAIFeatures(aiRejectAlliance,false);

    pEnemy.EnableAIFeatures(aiRush,false);
    pEnemy.EnableAIFeatures(aiControlOffense,false);

    pPlayer.SetAlly(pNeo);
    //----- Money -----
    pPlayer.SetMoney(0);
    //----- Researches -----
    tmpPlayer.EnableResearch("RES_ED_WSH1",true);
    tmpPlayer.EnableResearch("RES_ED_WH1",true);
    tmpPlayer.EnableResearch("RES_ED_UHL1",true);
    tmpPlayer.EnableResearch("RES_ED_UHW1",true);
    tmpPlayer.EnableResearch("RES_ED_UA1",true);
    tmpPlayer.EnableResearch("RES_ED_SGEN",true);

    pEnemy.EnableResearch("RES_UCS_WMR1",true);
    pEnemy.EnableResearch("RES_UCS_WHG1",true);
    pEnemy.EnableResearch("RES_MMR2",true);
    pEnemy.EnableResearch("RES_UCS_UHL1",true);
    pEnemy.EnableResearch("RES_UCS_BOMBER21",true);
    pEnemy.EnableResearch("RES_UCS_BHD",true);
    pEnemy.EnableResearch("RES_UCS_SGEN",true);
    //----- Buildings -----
    pPlayer.EnableBuilding("LCBDF",false);
    pPlayer.EnableBuilding("LCBHQ",false);
    pPlayer.EnableBuilding("LCBSD",false);
    pPlayer.EnableBuilding("LCBWC",false);
    pPlayer.EnableBuilding("LCBS5",false);
    pPlayer.EnableBuilding("LCBLZ",true);
    pPlayer.EnableBuilding("LCLASERWALL",false);
    //----- Units -----
    uHero=pPlayer.GetScriptUnit(0);
    uNeo=GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));
}

//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCaptureMechs = true;
bCheckEndMission = false;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;/5 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing341a",pPlayer.GetName());//zaprowadz Neo w
pobite Headquarter w bazie UCS
return ReachEnemyBase,100;
}
//-----
state ReachEnemyBase
{
    if(uNeo.DistanceTo(GetPointX(1),GetPointY(1))<15)
    {
        pEnemy.GiveAllUnitsTo(pPlayer);
        //----- EnableGoal(destroyEnemyBase,true);
        AddBriefing("translateBriefing341b",pPlayer.GetName());//oto moj przyja-
ciemu nasze nowe zabawki teraz mozemy zniszczyć ta baze i wykorzystac surow-
ce;
        return DestroyEnemyBase,100;
    }

    if(uHero.IsInWorld(GetWorldNum()))
    {
        if(uNeo.IsLive())
            uNeo.CommandMove(uHero.GetLocationX(),uHero.GetLocationY(),+2,uHero.GetLoc-
ationZ());
    }
    return ReachEnemyBase,300;
}
//-----
state DestroyEnemyBase
{
    if(bCheckEndMission)
    {
        if(!pEnemy.GetNumberOfBuildings())
        {
            SetGoalState(escortNeo,goalAchieved);
            SetGoalState(destroyEnemyBase,goalAchieved);
            EnableGoal(backToBase,true);
            AddBriefing("translateAccomplished341",pPlayer.GetName());///enemy
base destroyed
            EnableNextMission(0,true);
            return Nothing,100;
        }
        return DestroyEnemyBase,100;
    }

    if((GetGoalState(backToBase)!=goalAchieved &&
    !uHero.IsInWorld(GetWorldNum())))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawiione funkcja SetTimer w state

```

```

Initialize
{
    if(GetGoalState(escortNeo)!=goalFailed && !uNeo.IsLive())
    {
        SetGoalState(escortNeo,goalFailed);
        EnableGoal(backToBase,true);
        AddBriefing("translateFailed341",pPlayer.GetName());//Neo killed
        EnableNextMission(0,false);
        state Nothing;
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    pNeo.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pNeo);
}
}

```

Mission342.ec

```

mission "translateMission342"
{/India
consts
{
    sendToBase50000 = 0;
}

player pEnemy1;
player pEnemy2;
player pEnemy3;
player pEnemy4;
player pEnemy5;
player pPlayer;

int bShowFailed;
int bCheckEndMission;
int nTimer;
int nAttack;

state Initialize;
state ShowBriefing;
state Mining;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(sendToBase50000,"translateGoalSend50000",0);
    EnableGoal(sendToBase50000,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(1);
    tmpPlayer.EnableStatistics(false);
    tmpPlayer = GetPlayer(2);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy1 = GetPlayer(4);
    pEnemy2 = GetPlayer(5);
    pEnemy3 = GetPlayer(6);
    pEnemy4 = GetPlayer(7);
    pEnemy5 = GetPlayer(8);

    //----- AI -----
    pEnemy1.EnableStatistics(false);
    pEnemy2.EnableStatistics(false);
    pEnemy3.EnableStatistics(false);
    pEnemy4.EnableStatistics(false);
    pEnemy5.EnableStatistics(false);

    pEnemy1.setMaxTankPlatoonSize(4);
    pEnemy2.setMaxTankPlatoonSize(4);
    pEnemy3.setMaxTankPlatoonSize(4);
    pEnemy4.setMaxTankPlatoonSize(4);
}

```

```

pEnemy5.setMaxTankPlatoonSize(4);

pEnemy1.setNumberOfDefensiveTankPlatoons(3-GetDifficultyLevel());
pEnemy2.setNumberOfDefensiveTankPlatoons(3-GetDifficultyLevel());
pEnemy3.setNumberOfDefensiveTankPlatoons(3-GetDifficultyLevel());
pEnemy4.setNumberOfDefensiveTankPlatoons(3-GetDifficultyLevel());
pEnemy5.setNumberOfDefensiveTankPlatoons(3-GetDifficultyLevel());

pEnemy1.setNumberOfOffensiveTankPlatoons(1+GetDifficultyLevel());
pEnemy2.setNumberOfOffensiveTankPlatoons(1+GetDifficultyLevel());
pEnemy3.setNumberOfOffensiveTankPlatoons(1+GetDifficultyLevel());
pEnemy4.setNumberOfOffensiveTankPlatoons(1+GetDifficultyLevel());
pEnemy5.setNumberOfOffensiveTankPlatoons(1+GetDifficultyLevel());

pEnemy1.EnableAIFeatures(aiControlOffense,false);
pEnemy2.EnableAIFeatures(aiControlOffense,false);
pEnemy3.EnableAIFeatures(aiControlOffense,false);
pEnemy4.EnableAIFeatures(aiControlOffense,false);
pEnemy5.EnableAIFeatures(aiControlOffense,false);

pEnemy1.setNeutral(pEnemy5);
pEnemy5.setNeutral(pEnemy1);

pEnemy2.setNeutral(pEnemy3);
pEnemy3.setNeutral(pEnemy2);

pEnemy2.setNeutral(pEnemy4);
pEnemy4.setNeutral(pEnemy2);

pEnemy1.setName("Naroshi");
pEnemy2.setName("Taga Ri");
pEnemy3.setName("Kon Hi");
pEnemy4.setName("Wi Jan Ho");
pEnemy5.setName("Tar Gan");

pPlayer.EnableAIFeatures(aiEnabled,false);
//----- Money -----
pPlayer.setMoney(10000);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_WMR1",true);
pPlayer.EnableResearch("RES_MMR2",true);
pPlayer.EnableResearch("RES_LC_UCR1",true);
pPlayer.EnableResearch("RES_LC_HGEN",true);

//----- Buildings -----
//----- Units -----
//----- Artifacts -----
//----- Variables -----
bCheckEndMission=false;
bShowFailed=true;
nAttack=0;
if(GetDifficultyLevel()==0)
    nTimer=9000;
if(GetDifficultyLevel()==1)
    nTimer=6000;
if(GetDifficultyLevel()==2)
    nTimer=3600;
//----- Timers -----
SetTimer(0,100);
SetTimer(1,nTimer);

//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6.0,20.0);
return ShowBriefing,200;/15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing342",pPlayer.GetName());
    EnableNextMission(0,true);
    return Mining,100;
}
//-----
state Mining
{
    if(pPlayer.getMoney$entToBase()>=50000)
    {
        SetGoalState(sendToBase50000,goalAchieved);
        AddBriefing("translateAccomplished342",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    return Mining,200;
}

```

```

//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase50000,"translateGoalSend50000",pPlayer.GetMoneySent
ToBase());
    if(bShowFailed)
    {
        if(IsGoalEnabled(sendToBase50000))
        {

if((ResourcesLeftInMoney()+pPlayer.GetMoneySentToBase())+pPlayer.GetMoney())<
50000)
        {
            bShowFailed=false;
            SetGoalState(sendToBase50000,goalFailed);
            AddBriefing("translateFailed342",pPlayer.GetName());
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }
    }
}
//-----
event Timer1()
{
    nAttack=nAttack+1;
    if(nAttack==1)
    {
        pEnemy1.EnableAIFeatures(aiControlOffense,true);
        pEnemy1.EnableStatistics(true);
    }
    if(nAttack==2)
    {
        pEnemy2.EnableAIFeatures(aiControlOffense,true);
        pEnemy2.EnableStatistics(true);
    }
    if(nAttack==3)
    {
        pEnemy3.EnableAIFeatures(aiControlOffense,true);
        pEnemy3.EnableStatistics(true);
    }
    if(nAttack==4)
    {
        pEnemy4.EnableAIFeatures(aiControlOffense,true);
        pEnemy4.EnableStatistics(true);
    }
    if(nAttack==5)
    {
        pEnemy5.EnableAIFeatures(aiControlOffense,true);
        pEnemy5.EnableStatistics(true);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----



{ destroyEnemyForces=0;
}

player pEnemy;
player pAlly;
player pPlayer;

int bCheckEndMission;
int nNeoAttack;

state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(destroyEnemyForces,"translateGoal343");
    EnableGoal(destroyEnemyForces,true);
    //----- Temporary players -----
    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(2);
    pAlly = GetPlayer(1);
    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(30000);
    pPlayer.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        pEnemy.LoadScript("single\singleEasy");
        pAlly.LoadScript("single\singleHard");
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy.LoadScript("single\singleMedium");
        pAlly.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy.LoadScript("single\singleHard");
        pAlly.LoadScript("single\singleEasy");
    }
    pAlly.EnableAIFeatures(aiRejectAlliance,false);
    pPlayer.SetAlly(pAlly);
    pAlly.ChooseEnemy(pEnemy);
    pAlly.SetEnemy(pEnemy);

    pEnemy.ChooseEnemy(pPlayer);
    pEnemy.SetEnemy(pAlly);

    //----- Money -----
    pPlayer.SetMoney(20000);
    pEnemy.SetMoney(120000);
    pAlly.SetMoney(120000);
    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WMR1",true);
    pPlayer.EnableResearch("RES_MMR2",true);
    pPlayer.EnableResearch("RES_LC_UCR1",true);
    pPlayer.EnableResearch("RES_LC_HGEN",true);

    pEnemy.EnableResearch("RES_ED_WMR1",true);
    pEnemy.EnableResearch("RES_ED_UHT1",true);
    pEnemy.EnableResearch("RES_ED_UHW1",true);
    pEnemy.EnableResearch("RES_ED_UA41",true);
    pEnemy.EnableResearch("RES_ED_SGEN",true);

    pAlly.EnableResearch("RES_UCS_WMR1",true);
    pAlly.EnableResearch("RES_UCS_WHG1",true);
    pAlly.EnableResearch("RES_MM2",true);
    pAlly.EnableResearch("RES_UCS_UHL1",true);
    pAlly.EnableResearch("RES_UCS_BOMBER21",true);
    pAlly.EnableResearch("RES_UCS_BHD",true);
    pAlly.EnableResearch("RES_UCS_SGEN",true);

    //----- Buildings -----
    //----- Units -----
    //----- Artefacts -----
    //----- Timers -----
    if(GetDifficultyLevel()==0)
        SetTimer(0,24000); //20min
    if(GetDifficultyLevel()==1)
        SetTimer(0,12000); //10min
}

```

Mission343.ec

mission "translateMission343"
{/Madagaskar
consts

```

if(GetDifficultyLevel()==2)
SetTimer(0,6000); //5 min
//----- Variables -----
bCheckEndMission=false;
nNeoAttack=0;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
AddBriefing("translateBriefing343",pPlayer.GetName());
EnableNextMission(0,true);
return Fighting,100;
}
//-----
state Fighting
{
if(bCheckEndMission)
{
bCheckEndMission=false;
if(pEnemy.GetNumberOfUnits()<8 && lpEnemy.GetNumberOfBuildings())
{
SetGoalState(destroyEnemyForces,goalAchieved);
AddBriefing("translateAccomplished343",pPlayer.GetName());
EnableEndMissionButton(true);
return Evacuate,500;
}
if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
{
AddBriefing("translateFailedNoUnits",pPlayer.GetName());
EndMission(false);
}
}
return Fighting,100;
}
//-----
state Evacuate
{
return Evacuate,500;
}
//-----
event UnitDestroyed(unit u_Unit)
{
bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
bCheckEndMission=true;
}
//-----
event Timer0()
{
nNeoAttack=nNeoAttack+1;
if(nNeoAttack==1)
{
AddBriefing("translateBriefing343b",pPlayer.GetName());
pAlly.GiveAllUnitsTo(pEnemy);
}
if(nNeoAttack==3)
{
AddBriefing("translateBriefing343c",pPlayer.GetName());
pAlly.GiveAllUnitsTo(pEnemy);
}
}
//-----
event EndMission()
{
pAlly.EnableAIFeatures(aiRejectAlliance,true);
pPlayer.SetEnemy(pAlly);
pAlly.SetEnemy(pPlayer);
}
//-----
//----- Money -----
pPlayer.SetMoney(20000);
pEnemy.SetMoney(120000);
pAlly.SetMoney(120000);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_WHL1",true);
pPlayer.EnableResearch("RES_LC_UCR1",true);

pEnemy.EnableResearch("RES_ED_UHT1",true);
pEnemy.EnableResearch("RES_ED_UHW1",true);
pEnemy.EnableResearch("RES_ED_UA41",true);
pEnemy.EnableResearch("RES_ED_SGEN",true);

pAlly.EnableResearch("RES_UCS_WMR1",true);
pAlly.EnableResearch("RES_UCS_WHG1",true);
pAlly.EnableResearch("RES_UCS_UHL1",true);
pAlly.EnableResearch("RES_UCS_BOMBER21",true);
pAlly.EnableResearch("RES_UCS_SGEN",true);
pAlly.EnableResearch("RES_UCS_BHD",true);
//----- Buildings -----
//----- Units -----
//----- Artefact -----
//----- Timers -----
if(GetDifficultyLevel()==0)

```

Mission344.ec

```
mission "translateMission344"  
{//Australia  
    consts
```

```

nNeoAttack=0;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
AddBriefing("translateBriefing344a",pPlayer.GetName());
return Fighting,100;
}
//-----
state Fighting
{
if(bCheckEndMission)
{
bCheckEndMission=false;
if(!pEnemy.GetNumberOfBuildings())
{
SetGoalState(destroyEnemyForces.goalAchieved);
AddBriefing("translateAccomplished344",pPlayer.GetName());
EnableEndMissionButton(true);
return Evacuate,500;
}
if(!pPlayer.GetNumberOfUnits() && !pPlayer.GetNumberOfBuildings())
{
AddBriefing("translateFailedNoUnits",pPlayer.GetName());
EndMission(false);
}
}
return Fighting,100;
}
//-----
state Evacuate
{
return Evacuate,500;
}
//-----
event UnitDestroyed(unit u_Unit)
{
bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
bCheckEndMission=true;
}
//-----//--



event Timer0()
{
nNeoAttack=nNeoAttack+1;
if(nNeoAttack==1)
{
AddBriefing("translateBriefing344b",pPlayer.GetName());
pAlly.GiveAllUnitsTo(pEnemy);
}
if(nNeoAttack==3)
{
AddBriefing("translateBriefing344c",pPlayer.GetName());
pAlly.GiveAllUnitsTo(pEnemy);
}
}
//-----
event EndMission()
{
pAlly.EnableAIFeatures(aiRejectAlliance,true);
pPlayer.SetEnemy(pAlly);
pAlly.SetEnemy(pPlayer);
}
}
//-----



mission "translateMission351"
{/Mozambik kill Neo - tylko LC poniewaz UCS sie boi
consts
{
destroyEDBase = 0;
destroyNeoHome = 1;
}
player pEnemy;
player pPlayer;
unitex uNeoHome;
int bNeoFirstAttack;
int bNeoSecondAttack;
state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;
//-----
state Initialize
{
player tmpPlayer;
//----- Goals -----
RegisterGoal(destroyEDBase,"translateGoal351a");
RegisterGoal(destroyNeoHome,"translateGoal351b");
EnableGoal(destroyEDBase,true);
EnableGoal(destroyNeoHome,true);
//----- Temporary players -----
tmpPlayer = GetPlayer(1);
tmpPlayer.EnableStatistics(false);
//----- Players -----
pPlayer = GetPlayer(3);
pEnemy = GetPlayer(2);
//----- AI -----
pPlayer.SetMilitaryUnitsLimit(40000);
pPlayer.EnableAIFeatures(aiEnabled,false);

if(GetDifficultyLevel()==0)
{
pEnemy.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel()==1)
{
pEnemy.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
pEnemy.LoadScript("single\singleHard");
}

//----- Money -----
pPlayer.SetMoney(20000);
pEnemy.SetMoney(170000);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_WHL1",true);
pPlayer.EnableResearch("RES_MM3",true);
pPlayer.EnableResearch("RES_LC_REG2",true);

pEnemy.EnableResearch("RES_ED_WHC1",true);
pEnemy.EnableResearch("RES_ED_UA31",true);
pEnemy.EnableResearch("RES_ED_EM11",true);
pEnemy.EnableResearch("RES_ED_SCR",true);

//----- Buildings -----
//----- Units -----
uNeoHome=GetUnit(GetPointX(0),GetPointY(0),0);
//----- Artefacts -----
//----- Timers -----
SetTimer(0,200);
SetTimer(1,18000); //7 min
//----- Variables -----
bNeoFirstAttack=true;
bNeoSecondAttack=false;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
AddBriefing("translateBriefing351",pPlayer.GetName());
EnableNextMission(0,true);
return Fighting,100;
}
//-----
state Fighting
{
}

```

Mission351.ec

```

mission "translateMission351"
{/Mozambik kill Neo - tylko LC poniewaz UCS sie boi
consts
{
destroyEDBase = 0;
destroyNeoHome = 1;
}

```

```

if(GetGoalState(destroyEDBase)==goalAchieved &&
GetGoalState(destroyNeoHome)==goalAchieved)
{
    AddBriefing("translateAccomplished351a",pPlayer.GetName());
    EnableEndMissionButton(true);
    return Evacuate,500;
}

if(GetGoalState(destroyNeoHome)!=goalAchieved && !uNeoHome.IsLive())
{
    SetGoalState(destroyNeoHome,goalAchieved);
    AddBriefing("translateAccomplished351b",pPlayer.GetName());
}

if(GetGoalState(destroyEDBase)==goalAchieved &&
lpEnemy.GetNumberOfBuildings())
{
    SetGoalState(destroyEDBase,goalAchieved);
}
return Fighting,200;
}

//----- Evacuate -----
state Evacuate
{
    return Evacuate,500;
}
//----- Mission352.ec -----
event Timer0()
{
    if(!lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
    {
        AddBriefing("translateFailedNoUnits",pPlayer.GetName());
        EndMission(false);
    }
}
}

//----- Mission352.ec -----
mission "translateMission352"
{/Rio de Janeiro
const
{
    killNeo = 0;
    backToBase = 1;
}

player pPlayer;
player pAlly;
player pEnemy;

unitex uNeo;
unitex uHero;

int nReflex;
int nWayPoint;

state Initialize;
state ShowBriefing;
state Preparing;
state Runaway;
state Nothing;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(killNeo,"translateGoal352");//kill Neo
    RegisterGoal(backToBase,"translateGoalHeroBackToBase",0);
    EnableGoal(killNeo,true);
    //----- Temporary players -----
    pPlayer = GetPlayer(3);
    pAlly = GetPlayer(1);
    pEnemy = GetPlayer(2);

    //----- AI -----
    pPlayer.EnableAIFeatures(aiEnabled,false);
    pAlly.EnableAIFeatures(aiEnabled,false);
    pEnemy.EnableAIFeatures(aiEnabled,false);

    pAlly.EnableAIFeatures(aiRejectAlliance,false);
    pAlly.SetEnemy(pEnemy);
    pAlly.ChooseEnemy(pAlly);
    pEnemy.SetEnemy(pAlly);
    pPlayer.SetAlly(pAlly);
}

//----- Money -----
pPlayer.SetMoney(0);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_UCU1",true);

pEnemy.EnableResearch("RES_ED_WHL1",true);
pEnemy.EnableResearch("RES_ED_UBL1",true);
pEnemy.EnableResearch("RES_UCS_UML1",true);
pEnemy.EnableResearch("RES_LCS_BOMBER31",true);
pEnemy.EnableResearch("RES_LCS_SHD",true);
//----- Buildings -----
pPlayer.EnableBuilding("LCBFF",false);
pPlayer.EnableBuilding("LCBPF",false);
pPlayer.EnableBuilding("LCBA",false);
pPlayer.EnableBuilding("LCBMR",false);
pPlayer.EnableBuilding("LCBS",false);
pPlayer.EnableBuilding("LCBC",false);
pPlayer.EnableBuilding("LCBAB",false);
pPlayer.EnableBuilding("LCBGA",false);
pPlayer.EnableBuilding("LCBDE",false);
pPlayer.EnableBuilding("LCBHQ",false);
pPlayer.EnableBuilding("LCBDS",false);
pPlayer.EnableBuilding("LCBWC",false);
pPlayer.EnableBuilding("LCBLZ",false);
pPlayer.EnableBuilding("LCASERWALL",false);
//----- Units -----
uNeo = GetUnit((GetPointX(0),GetPointY(0),GetPointZ(0));
uHero = pPlayer.GetScriptUnit(0);
//----- Artifacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
nWayPoint=1;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;//5 sec
}
//----- ShowBriefing -----
state ShowBriefing
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing352",pPlayer.GetName());//zabij go neo on
rzejal mechy i rozwala baze naszych przyjaciol
    return Preparing,100;
}
//----- Preparing -----
state Preparing
{
    if(pPlayer.GetNumberOfUnits()>0)
    {
        return Runaway,100;
    }
    return Preparing,100;
}
//----- Runaway -----
state Runaway
{
    if(pPlayer.
IsPointLocated((GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayPoint)))
)
    {
        nWayPoint=nWayPoint+1;
        if(nWayPoint>5) nWayPoint=5;
    }
    if(!uNeo.IsLive())
    {
        SetGoalState(killNeo, goalAchieved);
        if(uHero.IsInWorld(GetWorldNum()))
        {
            EnableGoal(backToBase,true);
        }
        AddBriefing("translateAccomplished352",pPlayer.GetName());//Neo is kill-
led
        return Nothing,100;
    }
    uNeo.CommandMove((GetPointX(nWayPoint),GetPointY(nWayPoint),GetPointZ(nWayP-
oint));
}

```

```

if(GetDifficultyLevel() == 0)
    return Runaway_300; //EASY
if(GetDifficultyLevel() == 1)
    return Runaway_200; //MEDIUM
return Runaway_100; //HARD
}
-----
state Nothing
{
    if(GetGoalState(backToBase) != goalAchieved &&
uHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}

//----- event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
}
//
//----- event UnitDestroyed(unit u_Unit)
{
}
//
event BuildingDestroyed(unit u_Unit)
{
}
//
//----- event EndMission()
{
    pEnemy.SetEnemy(pAllied);
    pAllied.SetEnemy(pEnemy);
}
}



## Mission353.ec


mission "translateMission353"
{
    consts
    {
        escortNeo = 0;
        destroyEnemyBase = 1;
        backToBase = 2;
    }

    player pPlayer;
    player pNeo;
    player pEnemy;
    player pDummy;

    unitex uHero;
    unitex uNeo;

    int bCaptureMechs;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state ReachEnemyBase;
    state DestroyEnemyBase;
    state Nothing;
    //
    state Initialize
    {
        player tmpPlayer;
        //----- Goals -----
        RegisterGoal(escortNeo,"translateGoal353a");
        RegisterGoal(destroyEnemyBase,"translateGoal353b");
        RegisterGoal(backToBase,"translateGoalHeroBackToBase",0);
        EnableGoal(escortNeo,true);
        //----- Temporary players -----
        tmpPlayer = GetPlayer(2);
        tmpPlayer.EnableStatistics(false);
        //----- Players -----
        pPlayer = GetPlayer(3);
        pNeo = GetPlayer(8);
        pEnemy = GetPlayer(1);
        pDummy = GetPlayer(4);
    }

    //----- AI -----
    pPlayer.EnableAIFeatures(aiEnabled,false);
    pPlayer.SetMoney(20000);
    pEnemy.SetMoney(20000);

    if(GetDifficultyLevel() == 0)
    {
        pEnemy.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel() == 1)
    {
        pEnemy.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel() == 2)
    {
        pEnemy.LoadScript("single\singleHard");
    }

    pDummy.EnableAIFeatures(aiEnabled,false);
    pDummy.EnableStatistics(false);

    pNeo.EnableStatistics(false);
    pNeo.EnableAIFeatures(aiEnabled,false);
    pNeo.EnableAIFeatures(aiRejectAlliance,false);

    pPlayer.SetAlly(pNeo);
    //----- Money -----
    pPlayer.SetMoney(0);
    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_UCU1",true);

    tmpPlayer.EnableResearch("RES_ED_WHCI",true);
    tmpPlayer.EnableResearch("RES_ED_WHLI",true);
    tmpPlayer.EnableResearch("RES_ED_UMLI",true);
    tmpPlayer.EnableResearch("RES_ED_UA2I",true);
    tmpPlayer.EnableResearch("RES_ED_SCR",true);

    pEnemy.EnableResearch("RES_UCS_WHP1",true);
    pEnemy.EnableResearch("RES_UCSUBL1",true);
    pEnemy.EnableResearch("RES_UCS_LUMI1",true);
    pEnemy.EnableResearch("RES_UCS_BOMBER31",true);
    pEnemy.EnableResearch("RES_UCS_SHD",true);
    //----- Buildings -----
    pPlayer.EnableBuilding("LCBBF",false);
    pPlayer.EnableBuilding("LCBP",false);
    pPlayer.EnableBuilding("LCBA",false);
    pPlayer.EnableBuilding("LCBMR",false);
    pPlayer.EnableBuilding("LCBSR",false);
    pPlayer.EnableBuilding("LCBRC",false);
    pPlayer.EnableBuilding("LCAB",false);
    pPlayer.EnableBuilding("LCBGA",false);
    pPlayer.EnableBuilding("LCBDE",false);
    pPlayer.EnableBuilding("LCBHQ",false);
    pPlayer.EnableBuilding("LCBSD",false);
    pPlayer.EnableBuilding("LCBW",false);
    pPlayer.EnableBuilding("LCSS",false);
    pPlayer.EnableBuilding("LCBLZ",true);
    pPlayer.EnableBuilding("LCLASERWALL",false);
    //----- Units -----
    uHero = pPlayer.GetScriptUnit(0);
    uNeo = GetUnit(GetPointX(0),GetPointY(0),GetPointZ(0));

    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    //----- Variables -----
    bCaptureMechs = true;
    bCheckEndMission = false;
    //----- Camera -----
    CallCamera();
}

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;/5 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing353",pPlayer.GetName()); //zaprowadz Neo w poblize Headquarter w bazie UCS
    return ReachEnemyBase,100;
}
//-----
state ReachEnemyBase
{
}

```

```

if(uNeo.IsLive() && uNeo.DistanceTo(GetPointX(1),GetPointY(1))<10)
{
    pEnemy.GiveAllUnitsTo(pDummy);
    KillArea(2,GetPointX(1),GetPointY(1),0,20);
    pDummy.GiveAllUnitsTo(pEnemy);
    SetGoalState(escortNeo,goalAchieved);
    EnableGoal(destroyEnemyBase,true);
}

uNeo.CommandMove(pNeo.GetStartingPointX(),pNeo.GetStartingPointY(),0);
return DestroyEnemyBase,100;
}

if(uHero.IsInWorld(GetWorldNum()))
{
    if(uNeo.IsLive())
}

uNeo.CommandMove(uHero.GetLocationX(),uHero.GetLocationY()+2,uHero.GetLoc
ationZ());
}
}

state DestroyEnemyBase
{
    return DestroyEnemyBase,100;
}

//-----
state Nothing
{
    if(GetGoalState(backToBase)!=goalAchieved &&
uHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 500;
}
}

event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }

        if(GetGoalState(destroyEnemyBase)!=goalAchieved &&
lpEnemy.GetNumberOfBuildings())
        {
            SetGoalState(destroyEnemyBase,goalAchieved);
            EnableGoal(backToBase,true);
            AddBriefing("translateAccomplished353",pPlayer.GetName());//enemy
base destroyed
            state Nothing;
        }

        if(GetGoalState(escortNeo)!=goalFailed && luNeo.IsLive())
        {
            SetGoalState(escortNeo,goalFailed);
            AddBriefing("translateFailed353",pPlayer.GetName());//Neo Killed
        }
    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event EndMission()
{
    pNeo.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pNeo);
}
}

```

Mission354.ec

```

mission "translateMission354"
{
    consts
    {
        destroyEnemyBase = 0;
        backToBase = 1;
    }

    player pEnemy;
    player pPlayer;
    player pCannon;

    unitex uHero;
    unitex uCannon;
    unitex uPP;

    int bCanCaptureCannon;
    int bCannonFireCounter;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Fighting;
    state ShowVideoState;
    state Nothing;
}

state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(destroyEnemyBase, "translateGoal354");
    RegisterGoal(backToBase, "translateGoalHeroBackToBase");
    EnableGoal(destroyEnemyBase,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(2);
    tmpPlayer.EnableStatistics(false);

    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(1);
    pCannon = GetPlayer(6);
    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(30000);
    pCannon.EnableStatistics(false);
    pCannon.EnableAIFeatures(aiEnabled,false);

    pPlayer.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
        pEnemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        pEnemy.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        pEnemy.LoadScript("single\singleHard");

    pPlayer.EnableAIFeatures(aiEnabled,false);
    //----- Money -----
    pPlayer.SetMoney(10000);
    pEnemy.SetMoney(40000);

    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WHS1",true);
    pPlayer.EnableResearch("RES_LC_WAS1",true);
    pPlayer.EnableResearch("RES_MMR3",true);
    pPlayer.EnableResearch("RES_LC_UCU1",true);
    pPlayer.EnableResearch("RES_LC_AMR1",true);
    pPlayer.EnableResearch("RES_LC_UBO1",true);
    pPlayer.EnableResearch("RES_LC_REG2",true);

    tmpPlayer.EnableResearch("RES_UCS_WHP1",true);
    tmpPlayer.EnableResearch("RES_UCS_WAPB1",true);
    tmpPlayer.EnableResearch("RES_UCS_UBL1",true);
    tmpPlayer.EnableResearch("RES_UCS_UM1",true);
    tmpPlayer.EnableResearch("RES_UCS_BOMBER3",true);
    tmpPlayer.EnableResearch("RES_UCS_SHD",true);

    //----- Buildings -----
    //----- Units -----
    uHero=pPlayer.GetScriptUnit(0);
    uCannon = GetUnit((GetPointX(0),GetPointY(0),0));
    uPP = GetUnit((GetPointX(1),GetPointY(1),0));
    //----- Artifacts -----
    //----- Timers -----
    SetTimer(0,100);
}

```

```

SetTimer(1,10000);
//----- Variables -----
bCheckEndMission=false;
bCanCaptureCannon=true;
bCannonFireCounter=0;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
ShowArea(8,GetPointX(0),GetPointY(0),0,3);
return ShowBriefing,200;//15 sec
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing354a",pPlayer.GetName());//zniszcz baze ED
many kody do ich dziala plazmowego mozesz je przejac musisz tylko tam sie
dostac
    return Fighting,100;
}
//-----
state Fighting
{
    int timeToExplode;
    if(bCanCaptureCannon &&
        uHero.IsInWorld(GetWorldNum()) &&
        uHero.DistanceTo(GetPointX(0),GetPointY(0))<10 &&
        !uHero.GetLocationZ())
    {
        bCanCaptureCannon=false;
        bCannonFireCounter=5;
        AddBriefing("translateBriefing354b",pPlayer.GetName());
        pCannon.GiveAllBuildingsTo(pPlayer);
        return Fighting,20;
    }
    if(bCannonFireCounter)
    {
        bCannonFireCounter=bCannonFireCounter-1;
        if(bCannonFireCounter==2)
        {
            uCannon.ChangePlayer(pEnemy);
            uPP.ChangePlayer(pEnemy);
            KillArea(2,GetPointX(0),GetPointY(0),0,10);
        }
        if(!bCannonFireCounter)
        {
            AddBriefing("translateBriefing354c",pPlayer.GetName());
            SetConsoleText("");
        }
        else
            SetConsoleText("translateMessage354",bCannonFireCounter+50);//tik
tik time remaining
        return Fighting,20;
    }
    return Fighting,200;
}
//-----
state ShowVideoState
{
    ShowVideo("CS3 10");
    EnableEndMissionButton(true);
    return Nothing,100;
}
//-----
state Nothing
{
    if(GetGoalState(backToBase)!=goalAchieved &&
uHero.IsInWorld(GetWorldNum()))
    {
        SetGoalState(backToBase,goalAchieved);
        EnableEndMissionButton(true);
    }
    return Nothing, 100;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
        bCheckEndMission=false;
}

```

```

if(pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings())
{
    AddBriefing("translateFailedNoUnits",pPlayer.GetName());
    EndMission(false);
}

if(GetGoalState(destroyEnemyBase)!=goalAchieved &&
pEnemy.GetNumberOfBuildings()<3)
{
    SetGoalState(destroyEnemyBase, goalAchieved);
    EnableGoal(backToBase,true);
    AddBriefing("translateAccomplished354",pPlayer.GetName());
    state ShowVideoState;
}

//-----
event Timer1() //wolany co 6000 cykli
{
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

Mission361.ec
mission "translateMission361"
//Madagascar
consts
{
    destroyEDForces=0;
    destroyUCSForces=1;
}
player pEnemy;
player pAlly;
player pPlayer;

int bCheckEndMission;
int bBetrayal;
int makeBetrayal;

state Initialize;
state ShowBriefing;
state Fighting;
state ShowVideoState;
state Evacuate;

//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(destroyEDForces,"translateGoal361a");
    RegisterGoal(destroyUCSForces,"translateGoal361b");
    EnableGoal(destroyEDForces,true);
    //----- Temporary players -----
    //----- Players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(2);
    pAlly = GetPlayer(1);
    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(40000);
    pPlayer.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        pEnemy.LoadScript("single\singleEasy");
        pAlly.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy.LoadScript("single\singleMedium");
        pAlly.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy.LoadScript("single\singleHard");
    }
}

```

```

    pAlly.LoadScript("single\singleHard");
}

pAlly.EnableAIFeatures(aiRejectAlliance,false);
pAlly.SetEnemy(pEnemy);
pAlly.ChooseEnemy(pEnemy);
pEnemy.SetEnemy(pAlly);
pPlayer.SetAlly(pAlly);

pPlayer.SetMilitaryUnitsLimit(40000);
//----- Money -----
pPlayer.SetMoney(20000);
pEnemy.SetMoney(150000);
pAlly.SetMoney(150000);
//----- Researches -----
pPlayer.EnableResearch("RES_LC_WH1",true);
pPlayer.EnableResearch("RES_LC_WAS1",true);
pPlayer.EnableResearch("RES_LC_AMR1",true);
pPlayer.EnableResearch("RES_LC_UBO1",true);
pPlayer.EnableResearch("RES_LC_SDDEF",true);
pPlayer.EnableResearch("RES_LC_WARTILLERY",true);
pPlayer.EnableResearch("RES_MMRA",true);

pEnemy.EnableResearch("RES_ED_WHR1",true);
pEnemy.EnableResearch("RES_ED_UTB1",true);

pAlly.EnableResearch("RES_UCS_PC",true);
pAlly.EnableResearch("RES_UCS_WSD",true);
pAlly.EnableResearch("RES_UCS_WAPB1",true);
pAlly.EnableResearch("RES_UCS_MB2",true);
//----- Buildings -----
//----- Units -----
//----- Artifacts -----
//----- Timers -----
if(GetDifficultyLevel()==0)
    SetTimer(0,24000)/20min
if(GetDifficultyLevel()==1)
    SetTimer(0,12000)/10min
if(GetDifficultyLevel()==2)
    SetTimer(0,6000)/5 min
//----- Variables -----
bCheckEndMission=false;
bBetrayal=true;
makeBetrayal=false;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing361a",pPlayer.GetName());
    EnableNextMission(0,true);
    return Fighting,100;
}
//-----
state Fighting
{
    if(makeBetrayal)
    {
        makeBetrayal=false;
        EnableGoal(destroyUCSForces,true);
        pAlly.EnableAIFeatures(aiRejectAlliance,true);
        pAlly.SetEnemy(pPlayer);
        pEnemy.SetEnemy(pPlayer);
        pPlayer.SetEnemy(pAlly);
        pAlly.ChooseEnemy(pPlayer);
        pEnemy.ChooseEnemy(pPlayer);
        AddBriefing("translateBriefing361b",pPlayer.GetName());
    }

    if(GetGoalState(destroyEDForces)==goalAchieved &
        GetGoalState(destroyUCSForces)==goalAchieved )
    {
        AddBriefing("translateAccomplished361c",pPlayer.GetName());
        return ShowVideoState,20;
    }

    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(GetGoalState(destroyEDForces)!=goalAchieved &&
            !pEnemy.GetNumberOfUnits() &&
            !pEnemy.GetNumberOfBuildings())
        {
            SetGoalState(destroyEDForces,goalAchieved);
            AddBriefing("translateAccomplished361a",pPlayer.GetName());//ED forces destroyed
        }
        if(GetGoalState(destroyUCSForces)!=goalAchieved &&
            !pAlly.GetNumberOfUnits() &&
            !pAlly.GetNumberOfBuildings())
        {
            SetGoalState(destroyUCSForces,goalAchieved);
            AddBriefing("translateAccomplished361b",pPlayer.GetName());//UCS forces destroyed Traitors are dead now
        }
        if(!pPlayer.GetNumberOfUnits() && !pPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }
    }
}

return Fighting,100;
}
//-----
state ShowVideoState
{
    ShowVideo("CS311");
    EnableEndMissionButton(true);
    return Evacuate,500;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event Timer0()
{
    if(bBetrayal)
    {
        bBetrayal=false;
        makeBetrayal=true;
    }
}
//-----
event EndMission()
{
    pAlly.EnableAIFeatures(aiRejectAlliance,true);
    pPlayer.SetEnemy(pAlly);
    pAlly.SetEnemy(pPlayer);
}
}

Mission362.ec
mission "translateMission362"
{/Madagascar
consts
{
    destroyEDForces=0;
    destroyUCSForces=1;
}

player pEnemy;
player pAlly;
player pPlayer;

int bCheckEndMission;
int bBetrayal;
int makeBetrayal;

state Initialize;
state ShowBriefing;
state Fighting;
state ShowVideoState;

```

```

state Evacuate;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(destroyEDForces,"translateGoal362a");
    RegisterGoal(destroyUCSForces,"translateGoal362b");
    EnableGoal(destroyUCSForces,true);
    //----- Temporary players -----
    pPlayer = GetPlayer(3);
    pEnemy = GetPlayer(1);
    pAlly = GetPlayer(2);
    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(40000);
    pPlayer.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        pEnemy.LoadScript("single\singleEasy");
        pAlly.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy.LoadScript("single\singleMedium");
        pAlly.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy.LoadScript("single\singleHard");
        pAlly.LoadScript("single\singleHard");
    }

    pAlly.EnableAIFeatures(aiRejectAlliance,false);
    pAlly.SetEnemy(pEnemy);
    pAlly.ChooseEnemy(pEnemy);
    pEnemy.SetEnemy(pAlly);
    pPlayer.SetAlly(pAlly);

    //----- Money -----
    pPlayer.SetMoney(20000);
    pEnemy.SetMoney(150000);
    pAlly.SetMoney(150000);
    //----- Researches -----
    pPlayer.EnableResearch("RES_LC_WHL1",true);
    pPlayer.EnableResearch("RES_LC_SDIDEF",true);
    pPlayer.EnableResearch("RES_LC_WARTILLERY",true);
    pPlayer.EnableResearch("RES_MMR4",true);
    pPlayer.EnableResearch("RES_LC_PEG2",true);

    pAlly.EnableResearch("RES_ED_WHR1",true);
    pAlly.EnableResearch("RES_ED_UBT1",true);

    pEnemy.EnableResearch("RES_UCS_PC",true);
    pEnemy.EnableResearch("RES_UCS_WSD",true);
    pEnemy.EnableResearch("RES_UCS_WAPB1",true);
    pEnemy.EnableResearch("RES_UCS_MB2",true);
    //----- Buildings -----
    //----- Units -----
    //----- Artifacts -----
    //----- Timers -----
    if(GetDifficultyLevel()==0)
        SetTimer(0,24000)//20min
    if(GetDifficultyLevel()==1)
        SetTimer(0,12000)//10min
    if(GetDifficultyLevel()==2)
        SetTimer(0,6000)//2.5 min
    //----- Variables -----
    bCheckEndMission=false;
    bBetrayal=true;
    makeBetrayal = false;
    //----- Camera -----
    CallCamera();
}

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing362a",pPlayer.GetName());
    EnableNextMission(0,true);
    return Fighting,100;
}
//-----



state Fighting
{
    if(makeBetrayal)
    {
        makeBetrayal=false;
        EnableGoal(destroyEDForces,true);
        pAlly.EnableAIFeatures(aiRejectAlliance,true);
        pAlly.SetEnemy(pPlayer);
        pEnemy.SetEnemy(pPlayer);
        pPlayer.SetEnemy(pAlly);
        pAlly.ChooseEnemy(pPlayer);
        pEnemy.ChooseEnemy(pPlayer);
        AddBriefing("translateBriefing362b",pPlayer.GetName());
    }

    if(GetGoalState(destroyEDForces)==goalAchieved &&
       GetGoalState(destroyUCSForces)==goalAchieved )
    {
        AddBriefing("translateAccomplished362c",pPlayer.GetName());
        return ShowVideoState,20;
    }

    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(GetGoalState(destroyUCSForces)==goalAchieved &&
           |pEnemy.GetNumberOfUnits() &&
           |pEnemy.GetNumberOfBuildings()|)
        {
            SetGoalState(destroyUCSForces,goalAchieved);
            AddBriefing("translateAccomplished362a",pPlayer.GetName());//UCS
forces destroyed
        }
        if(GetGoalState(destroyEDForces)!=goalAchieved &&
           |pAlly.GetNumberOfUnits() &&
           |pAlly.GetNumberOfBuildings()|)
        {
            SetGoalState(destroyEDForces,goalAchieved);
            AddBriefing("translateAccomplished362b",pPlayer.GetName());//ED
forces destroyed
        }
        if(|pPlayer.GetNumberOfUnits() && |pPlayer.GetNumberOfBuildings()|)
        {
            AddBriefing("translateFailedNoUnits",pPlayer.GetName());
            EndMission(false);
        }
    }
    return Fighting,100;
}
//-----
state ShowVideoState
{
    ShowVideo("CS311");
    EnableEndMissionButton(true);
    return Evacuate,500;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----



event Timer0()
{
    if(bBetrayal)
    {
        bBetrayal=false;
        makeBetrayal=true;
    }
}
//-----
event EndMission()
{
    pAlly.EnableAIFeatures(aiRejectAlliance,true);
}

```

```

    pPlayer.SetEnemy(pAlly);
    pAlly.SetEnemy(pPlayer);
}
//-----
}
```

Mission363.ec

```

mission "translateMission363"
{
    consts
    {
        sendToBase = 0;
    }

    player pEnemy1;
    player pEnemy2;
    player pPlayer;
    int bShitchOnA;
    int nNeedeResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;

//-----
state Initialize
{
    //----- Goals -----
    pPlayer = GetPlayer(3);
    fleetCost = pPlayer.GetScriptData(0);
    nNeedeResources = fleetCost - pPlayer.GetMoneySentToOrbit();
    if(nNeedeResources > 100000)
        nNeedeResources=100000;

    RegisterGoal(sendToBase,"translateGoal363",nNeedeResources,0);
    EnableGoal(sendToBase,true);
    //----- Temporary players -----
    //----- Players -----
    pEnemy1 = GetPlayer(1);
    pEnemy2 = GetPlayer(2);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        pEnemy1.LoadScript("single\singleEasy");
        pEnemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy1.LoadScript("single\singleMedium");
        pEnemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy1.LoadScript("single\singleHard");
        pEnemy2.LoadScript("single\singleHard");
    }

    pPlayer.EnableAIFeatures(aiEnabled,false);
    pEnemy1.EnableAIFeatures(aiEnabled,true);
    pEnemy2.EnableAIFeatures(aiEnabled,true);
    //----- Money -----
    pPlayer.SetMoney(20000);
    pEnemy1.SetMoney(150000);
    pEnemy2.SetMoney(150000);
    //----- Researches -----
    //----- Buildings -----
    //----- Units -----
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    //----- Variables -----
    bShowFailed=true;
    bCheckEndMission=false;
    //----- Camera -----
    CallCamera();
}

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,150;/15 sec
}
//-----
```

```

state ShowBriefing
{
    AddBriefing("translateBriefing363",pPlayer.GetName(),nNeedeResources);
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    return Mining,200;
}
//-----
state Mining
{
    if(GetGoalState(sendToBase)==goalAchieved &&
    pPlayer.GetMoneySentToBase()>=nNeedeResources)
    {
        SetGoalState(sendToBase,goalAchieved);
        AddBriefing("translateAccomplished363",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase,"translateGoal363",nNeedeResources,pPlayer.GetMoney
    SentToBase());;

    if(bShowFailed)
    {
        if((ResourcesLeftInMoney())+pPlayer.GetMoneySentToBase()+pPlayer.GetMoney())
        <nNeedeResources)
        {
            bShowFailed=false;
            SetGoalState(sendToBase,goalFailed);
            AddBriefing("translateFailed363a",pPlayer.GetName());
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(pPlayer.GetNumberOfUnits() && !pPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed363b",pPlayer.GetName());
            EndMission(false);
        }
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}
//-----
```

Mission371.ec

```

mission "translateMission371"
{
    consts
    {
        sendToBase = 0;
    }

    player pEnemy1;
    player pEnemy2;
    player pPlayer;
    int bShitchOnA;
    int nNeedeResources;
```

```

int fleetCost;
int bShowFailed;
int bCheckEndMission;

state Initialize;
state ShowBriefing;
state Mining;
state Nothing;

//-----
state Initialize
{
    //----- Goals -----
    pPlayer = GetPlayer(3);
    fleetCost = pPlayer.GetScriptData(0);
    nNeedResources = fleetCost - pPlayer.GetMoneySentToOrbit();
    if(nNeedResources > 100000)
        nNeedResources=100000;

    RegisterGoal(sendToBase,"translateGoal371",nNeedResources,0);
    EnableGoal(sendToBase,true);
    //----- Temporary players -----
    //----- Players -----
    pEnemy1 = GetPlayer(1);
    pEnemy2 = GetPlayer(2);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        pEnemy1.LoadScript("single\singleEasy");
        pEnemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy1.LoadScript("single\singleMedium");
        pEnemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy1.LoadScript("single\singleHard");
        pEnemy2.LoadScript("single\singleHard");
    }

    pPlayer.EnableAIFeatures(aiEnabled,false);
    pEnemy1.EnableAIFeatures(aiEnabled,true);
    pEnemy2.EnableAIFeatures(aiEnabled,true);
    //----- Money -----
    pPlayer.SetMoney(10000);
    pEnemy1.SetMoney(15000);
    pEnemy2.SetMoney(15000);
    //----- Researches -----
    //----- Buildings -----
    //----- Units -----
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    //----- Variables -----
    bShowFailed=true;
    bCheckEndMission=false;
    //----- Camera -----
    CallCamera();
}

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,150;//15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing371",pPlayer.GetName(),nNeedResources);
    return Mining,200;
}
//-----
state Mining
{
    if(GetGoalState(sendToBase)!=goalAchieved &&
    pPlayer.GetMoneySentToBase()>=nNeedResources)
    {
        SetGoalState(sendToBase,goalAchieved);
        AddBriefing("translateAccomplished371",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}

}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase,"translateGoal371",nNeedResources,pPlayer.GetMoney
    SentToBase());
}

if(bShowFailed)
{
    if((ResourcesLeftInMoney()+pPlayer.GetMoneySentToBase())+pPlayer.GetMoney())
    <nNeedResources)
    {
        bShowFailed=false;
        SetGoalState(sendToBase,goalFailed);
        AddBriefing("translateFailed371a",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing;
    }
}

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed371b",pPlayer.GetName());
        EndMission(false);
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

}

mission "translateMission372"
{
    consts
    {
        sendToBase = 0;
    }

    player pEnemy1;
    player pEnemy2;
    player pPlayer;
    int bSwitchOnA;
    int nNeedResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;

    //-----
    state Initialize
    {
        //----- Goals -----
        pPlayer = GetPlayer(3);
        fleetCost = pPlayer.GetScriptData(0);
        nNeedResources = fleetCost - pPlayer.GetMoneySentToOrbit();
        if(nNeedResources > 100000)
            nNeedResources=100000;

        RegisterGoal(sendToBase,"translateGoal372",nNeedResources,0);
        EnableGoal(sendToBase,true);
        //----- Temporary players -----
        //----- Players -----
    }
}

```

```

pEnemy1 = GetPlayer(1);
pEnemy2 = GetPlayer(2);
//----- AI -----
if(GetDifficultyLevel() == 0)
{
    pEnemy1.LoadScript("single\singleEasy");
    pEnemy2.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel() == 1)
{
    pEnemy1.LoadScript("single\singleMedium");
    pEnemy2.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel() == 2)
{
    pEnemy1.LoadScript("single\singleHard");
    pEnemy2.LoadScript("single\singleHard");
}

pPlayer.EnableAIFeatures(aiEnabled,false);
pEnemy1.EnableAIFeatures(aiEnabled,true);
pEnemy2.EnableAIFeatures(aiEnabled,true);
//----- Money -----
pPlayer.SetMoney(20000);
pEnemy1.SetMoney(150000);
pEnemy2.SetMoney(150000);
//----- Researches -----
//----- Buildings -----
//----- Units -----
//----- Artifacts -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bShowFailed=true;
bCheckEndMission=false;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20.0);
    return ShowBriefing,150://15 sec
}
//----- ShowBriefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing372",pPlayer.GetName(),nNeedResources);
    return Mining,200;
}
//----- Mining -----
state Mining
{
    if(GetGoalState(sendToBase)|=goalAchieved &&
pPlayer.GetMoneySentToBase()>=nNeedResources)
    {
        SetGoalState(sendToBase, goalAchieved);
        AddBriefing("translateAccomplished372",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//----- Nothing -----
state Nothing
{
    return Nothing, 500;
}
//----- Timer0 -----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    RegisterGoal(sendToBase,"translateGoal372",nNeedResources,pPlayer.GetMoney
SendToBase());
}

if(bShowFailed)
{
    if((ResourcesLeftInMoney() + pPlayer.GetMoneySentToBase() + pPlayer.GetMoney())<nNeedResources)
    {
        bShowFailed=false;
        SetGoalState(sendToBase, goalFailed);
        AddBriefing("translateFailed372a",pPlayer.GetName());
        EnableEndMissionButton(true);
        return Nothing;
    }
}

```

```

if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lpPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed372b",pPlayer.GetName());
        EndMission(false);
    }
}
//----- event UnitDestroyed(unit u_Unit) -----
{
    bCheckEndMission=true;
}
//----- event BuildingDestroyed(unit u_Unit) -----
{
    bCheckEndMission=true;
}
//----- TutorialLCmis.ec -----
mission "translateTutorialLC"
{
    consts
    {
        buildPowerPlant = 0;
        upgradePowerPlant = 1;
        buildBattery = 2;
        buildMainBase = 3;
        buildMine = 4;
        buildArmy = 5;
        findEnemy = 6;
        destroyEnemy = 7;
    }
    player p_EnergyUCS;
    player p_Player;
    int nUCSUnitsCount;
    int nBuildingCount;
    int tmpCounter;
    int nUnitCount;
    //*****
    state Initialize;
    state Start;
    state MoveCamera;
    state BuildPowerPlant;
    state UpgradePowerPlant;
    state BuildBattery;
    state BuildMainBase;
    state BuildMine;
    state BuildArmy;
    state More;
    state EndState;
    state Initialize
    {
        RegisterGoal(buildPowerPlant,"translateGoalTutorialLC_PP");
        RegisterGoal(upgradePowerPlant,"translateGoalTutorialLC_PP_UPG");
        RegisterGoal(buildBattery,"translateGoalTutorialLC_BA");
        RegisterGoal(buildMainBase,"translateGoalTutorialLC_MB");
        RegisterGoal(buildMine,"translateGoalTutorialLC_M");
        RegisterGoal(buildArmy,"translateGoalTutorialLC_tanks");
        RegisterGoal(findEnemy,"translateGoalTutorialLC_findEnemy");
        RegisterGoal(destroyEnemy,"translateGoalTutorialLC_destroyEnemy");
        nUnitCount=0;
        p_EnergyUCS=GetPlayer(1);
        p_Player=GetPlayer(3);
    }
    p_EnergyUCS.SetMoney(10000);
    p_Player.SetMoney(20000);

    p_EnergyUCS.EnableAIFeatures(aiEnabled,false);
    p_Player.EnableAIFeatures(aiEnabled,false);

    p_Player.EnableResearch("RES_MMR2" false);
    //p_Player.EnableResearch("RES_LC_WCH2" false);
    p_Player.EnableResearch("RES_LC_ACH2" false);
    //p_Player.EnableResearch("RES_LC_WSR1" false);
    p_Player.EnableResearch("RES_LC_WSR2" false);
    p_Player.EnableResearch("RES_LC_ASR1" false);
}

```

```

//p_Player.EnableResearch("RES_LC_WSL1",false);
p_Player.EnableResearch("RES_LC_WSL2",false);
p_Player.EnableResearch("RES_LC_WSS1",false);
p_Player.EnableResearch("RES_LC_WMR1",false);
p_Player.EnableResearch("RES_LC_WH1",false);
p_Player.EnableResearch("RES_LC_WH51",false);

p_Player.EnableResearch("RES_LC_WAS1",false);
//p_Player.EnableResearch("RES_LC_UMO2",false);
p_Player.EnableResearch("RES_LC_UCR1",false);
p_Player.EnableResearch("RES_LC_UCU1",false);

p_Player.EnableResearch("RES_LC_UU3",false);
p_Player.EnableResearch("RES_LC_UMO3",false);
p_Player.EnableResearch("RES_LC_UME1",false);
p_Player.EnableResearch("RES_LC_UBO1",false);

p_Player.EnableResearch("RES_LC_BMD",false);
p_Player.EnableResearch("RES_LC_BHD",false);
p_Player.EnableResearch("RES_LC_SDIDEF",false);
//p_Player.EnableResearch("RES_LC_SGen",false);
p_Player.EnableResearch("RES_LC_MGen",false);
p_Player.EnableResearch("RES_LC_MGen",false);
p_Player.EnableResearch("RES_LC_SRH1",false);
p_Player.EnableResearch("RES_LC_REG1",false);
p_Player.EnableResearch("RES_LC_SOBI",false);
p_Player.EnableResearch("RES_LC_SDIDEF",false);
p_Player.EnableResearch("RES_LC_BWC",false);

p_Player.EnableBuilding("LCBPP",false);
p_Player.EnableBuilding("LCBF",false);
p_Player.EnableBuilding("LCBAA",false);
p_Player.EnableBuilding("LCBMR",false);
p_Player.EnableBuilding("LCBSR",false);
p_Player.EnableBuilding("LCBR",false);
p_Player.EnableBuilding("LCAB",false);
p_Player.EnableBuilding("LCBGA",false);
p_Player.EnableBuilding("LCBDE",false);
p_Player.EnableBuilding("LCBHQ",false);
p_Player.EnableBuilding("LCBSD",false);
p_Player.EnableBuilding("LCBWC",false);
p_Player.EnableBuilding("LCBLZ",false);
p_Player.EnableBuilding("LCLASERWALL",false);

SetTimer(0,100);

LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0,0);
nBuildingCount=0;
tmpCounter=0;
return Start,100;
}

state Start
{
    AddBriefing("translateTutorialLC_moveCamera");
    nJCSUnitsCount=p_EnemyJCS.GetNumberOfUnits()/2;
    return MoveCamera,500;
}

state MoveCamera
{
    EnableGoal(buildPowerPlant,true);
    p_Player.EnableBuilding("LCBPP",true);
    AddBriefing("translateTutorialLC_PP");
    return BuildPowerPlant,100;
}

state BuildPowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingSolarPower))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildPowerPlant,goalAchieved);
        EnableGoal(upgradePowerPlant,true);
        AddBriefing("translateTutorialLC_PP_UPG");
        return UpgradePowerPlant,100;
    }
    return BuildPowerPlant,100;
}

state UpgradePowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingSolarBattery)>3)
    {
        p_Player.EnableBuilding("LCBBA",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(upgradePowerPlant,goalAchieved);
        EnableGoal(buildBattery,true);
        if(!p_Player.GetNumberOFBuildings(buildingEnergyBattery))
            AddBriefing("translateTutorialLC_BA");
        if(p_Player.GetNumberOfUnits())
            if(p_Player.GetNumberOFBuildings(buildingEnergyBattery))
                return BuildBattery,100;
    }
    return UpgradePowerPlant,20;// sec
}

state BuildBattery
{
    if(p_Player.GetNumberOfBuildings(buildingEnergyBattery))
    {
        p_Player.EnableBuilding("LCBBF",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildBattery,goalAchieved);
        EnableGoal(buildMainBase,true);
        if(!p_Player.GetNumberOFBuildings(buildingBaseFactory))
            AddBriefing("translateTutorialLC_BF");
        return BuildMainBase,100;
    }
    if(p_Player.GetNumberOfUnits())
        if(p_Player.GetNumberOFBuildings(buildingEnergyBattery))
            return BuildBattery,50;
}

state BuildMainBase
{
    if(p_Player.GetNumberOfBuildings(buildingBaseFactory))
    {
        p_Player.EnableBuilding("LCBMR",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildMainBase,goalAchieved);
        EnableGoal(buildMine,true);
        p_EnemyJCS.LoadSceneScript("single\singleEasy");
        p_EnemyJCS.EnableAIFeatures(aiEnabled,true);
        p_EnemyJCS.EnableAIFeatures(aiBuildBuildings,false);
        p_EnemyJCS.EnableAIFeatures(aiControlOffense,false);
        p_EnemyJCS.EnableAIFeatures(aiBuildTanks | aiBuildShips | aiBuildHelicopters,false);
    }
    if(p_Player.GetNumberOfUnits())
        if(p_Player.GetNumberOFBuildings(buildingMine))
            AddBriefing("translateTutorialLC_MI");
        return BuildMine,100;
}

state BuildMine
{
    if(p_Player.GetNumberOfBuildings(buildingMiningRefinery))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildMine,goalAchieved);
        EnableGoal(buildArmy,true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
    }
}

```

```

AddBriefing("translateTutorialLC_tanks");

nUnitCount=p_Player.GetNumberOfUnits(chassisAmphibianTank | unitArmed);
    if(nUnitCount>3)nUnitCount=3;
    return BuildArmy,50;
}
if(!p_Player.GetNumberOfUnits())
{
    p_Player.CreateUnitEx(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
    0,null,"LCULU1","LCWCH1",null,null,0);
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialLC_M1_Fool");
    }
    return BuildMine,50;
}

//-----
state BuildArmy
{
    if(p_Player.GetNumberOfUnits(chassisAmphibianTank | unitArmed)>=nUnitCount+
    4)
    {
        SetGoalState(buildArmy.goalAchieved);
        EnableGoal(findEnemy,true);
        AddBriefing("translateTutorialLC_findEnemy");
        return More,600;
    }
    return BuildArmy,100;
}

state More
{
    p_Player.EnableBuilding("LCBAB",true);
    p_Player.EnableBuilding("LCBGA",true);
    p_Player.EnableBuilding("LCBRC",true);
    p_Player.EnableBuilding("CLASERWALL",true);
    AddBriefing("translateTutorialLC_more");
    return EndState,100;
}

state EndState
{
    return EndState,500;
}

event Timer0()
{
    if(GetGoalState(findEnemy)==goalAchieved &&
    p_Player.IsPointLocated(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStartin
    gPointY(),0))
    {
        SetGoalState(findEnemy.goalAchieved);
        EnableGoal(destroyEnemy,true);
        AddBriefing("translateTutorialLC_destroyEnemy");
    }
    if(GetGoalState(destroyEnemy)==goalAchieved &&
    lp_EnemyUCS.GetNumberOfUnits())
    {
        SetGoalState(destroyEnemy.goalAchieved);
        AddBriefing("translateTutorialLC_Victory");
    }
}



## UCS



### CampaignUCS.ec


campaign "translateCampaignUCS"
{
    consts
    {
        raceUCS=1;
        raceED=2;
        raceLC=3;
    }

    mis111 = 0;
    mis112 = 1;
    mis113 = 2;
    mis114 = 3;
    mis121 = 4;
    mis122 = 5;
    mis123 = 6;
    mis124 = 7;
    mis131 = 8;
    mis132 = 9;
    mis133 = 10;
    mis134 = 11;
    mis141 = 12;
    mis142 = 13;
    mis143 = 14;
    mis144 = 15;
    mis151 = 16;
    mis152 = 17;
    mis153 = 18;
    mis165 = 19;
    mis171 = 20;
    mis172 = 21;

    baseUCS = 22;
    baseED = 23;
    baseLC = 24;

    timeFlagWinter = 1;
    timeFlagEarlySpring = 2;
    timeFlagSpring = 4;
    timeFlagSummer = 8;
    timeFlagDesert = 16;
    timeFlagVolcano = 32;
    timeFlagHeavyVolcano = 64;
    baseFlag = 255;

    // pieniadze przewidywane do budowy statku na poczatku okresu
    cashEarlySpring = 50000;
    cashSpring = 100000;
    cashSummer = 150000;
    cashDesert = 200000;
    cashVolcano = 250000;
    spaceShipPrice = 300000;

    // czas zmiany poru roku
    timeEarlySpring = 150000; //2h 5min
    timeSpring = 300000;
    timeSummer = 450000;
    timeDesert = 600000;
    timeVolcano = 750000;
    timeHeavyVolcano = 900000;
    timeEarthDestroyed= 1050000;
}

int n_TimeFlag;
int n_CashCounter;
int b_showVideo;
int b_showVideo2;

state Initialize;
state Start;
state Nothing;

state Initialize
{
    int i;

    RegisterMission(baseUCS,"baseUCS","UCS\\Missions\\UCSbaseUCSscript","");
    baseFlag, -84, 41,0,0,0);
    RegisterMission(baseED,"baseED","UCS\\Missions\\UCSbaseEDscript","");
    baseFlag, 50, 60,0,0,0);
    RegisterMission(baseLC,"baseLC","UCS\\Missions\\UCSbaseLCscript","");
    baseFlag, 120, 8,0,0,0);

    //          nr      lnd   script   preBriefing
    timeFlags   x   y   d1   d2   next1 next2 next3 next4
    RegisterMission(mis111, "111", "UCS\\Missions\\Mission111",
    "translateBriefingShort111", timeFlagWinter,     60, 60, 60, 60,
    mis112,mis113);
    RegisterMission(mis112, "112", "UCS\\Missions\\Mission112",
    "translateBriefingShort112", timeFlagWinter,     60, 78, 90, 90, 90,
    mis114,mis121);
    RegisterMission(mis113, "113", "UCS\\Missions\\Mission113",
    "translateBriefingShort113", timeFlagWinter, -100, 40, 90, 90, 90, mis122);
    RegisterMission(mis114, "114", "UCS\\Missions\\Mission114",
    "translateBriefingShort114", timeFlagWinter,     120, 85,120,120,120);
}

```

```

//-----
RegisterMission(mis121, "121", "UCS\\Missions\\Mission121",
"translateBriefingShort121", timeFlagEarlySpring, 105, 52,0,0, mis131);
RegisterMission(mis122, "122", "UCS\\Missions\\Mission122",
"translateBriefingShort122", timeFlagEarlySpring,-100, 40,0,0,
mis134,mis123);
RegisterMission(mis123, "123", "UCS\\Missions\\Mission123",
"translateBriefingShort123", timeFlagEarlySpring,-150, 65,0,0, mis124);
RegisterMission(mis124, "124", "UCS\\Missions\\Mission124",
"translateBriefingShort124", timeFlagEarlySpring, 140, 40,0,0,
mis132,mis133);
//-----

RegisterMission(mis131, "131", "UCS\\Missions\\Mission131",
"translateBriefingShort131", timeFlagSpring, -90, 42,0,0, mis142);
RegisterMission(mis132, "132", "UCS\\Missions\\Mission132",
"translateBriefingShort132", timeFlagSpring, 90, 35,0,0);
RegisterMission(mis133, "133", "UCS\\Missions\\Mission133",
"translateBriefingShort133", timeFlagSpring, -74, 41,0,0);
RegisterMission(mis134, "134", "UCS\\Missions\\Mission134",
"translateBriefingShort134", timeFlagSpring, -100, 40,0,0);
//-----

RegisterMission(mis141, "141", "UCS\\Missions\\Mission141",
"translateBriefingShort141", timeFlagSummer, -47,-18,0,0, mis152);
RegisterMission(mis142, "142", "UCS\\Missions\\Mission142",
"translateBriefingShort142", timeFlagSummer, 77, 15,0,0,0,
mis141,mis143,mis144);
RegisterMission(mis143, "143", "UCS\\Missions\\Mission143",
"translateBriefingShort143", timeFlagSummer, -47,-18,0,0, mis153);
RegisterMission(mis144, "144", "UCS\\Missions\\Mission144",
"translateBriefingShort144", timeFlagSummer, 133,-13,0,0,0, mis153);
//-----

RegisterMission(mis151, "151", "UCS\\Missions\\Mission151",
"translateBriefingShort151", timeFlagDesert, 35,-15,0,0, mis165);
RegisterMission(mis152, "152", "UCS\\Missions\\Mission152",
"translateBriefingShort152", timeFlagDesert, 135,-25,0,0, mis151);
RegisterMission(mis153, "153", "UCS\\Missions\\Mission153",
"translateBriefingShort153", timeFlagDesert, 31, 30,0,0, mis151);
//-----

RegisterMission(mis165, "165", "UCS\\Missions\\Mission165",
"translateBriefingShort165", timeFlagVolcano, -75,-5,0,0, mis171,mis172);
//-----

RegisterMission(mis171, "171", "UCS\\Missions\\Mission171",
"translateBriefingShort171", timeFlagHeavyVolcano,-72, 3,0,0);
RegisterMission(mis172, "172", "UCS\\Missions\\Mission172",
"translateBriefingShort172", timeFlagHeavyVolcano,-70,-30,0,0);
//-----



n_CashCounter = timeFlagEarlySpring;
n_TimeFlag = timeFlagWinter;
b_showVideo = true;
b_showVideo2 = false;

CreateGamePlayer(1,raceUCS,playerLocal,null);
CreateGamePlayer(2,raceED,playerAI,null);
CreateGamePlayer(3,raceLC,playerAI,null);

LoadBase(1,baseUCS,1);
LoadBase(2,baseED,2);
LoadBase(3,baseLC,3);

SetActivePlayerAndWorld(1,1);//player, world
SetAvailableWorlds(1|2)//misja i baza UCS(swiat nr 1)

EnableMission(mis111,true);
ActivateMissions(timeFlagWinter,true);
SetSeason(1);

return Start,240;
}

state Start
{
    EnableChooseMissionButton(true);
    return Nothing,50;
}

state Nothing
{
    if(b_showVideo2)
    {
        b_showVideo2 = false;
        if(n_TimeFlag == timeFlagWinter)
            ShowVideo("CS114");
        if(n_TimeFlag == timeFlagSpring)
            ShowVideo("CS115");
        if(n_TimeFlag == timeFlagDesert)
            ShowVideo("CS116");
        if(n_TimeFlag == timeFlagVolcano)
            ShowVideo("CS117");
    }
    return Nothing,50;
}

event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    TraceID(iMission);

    LoadMission(0,iMission);

    EnableMission(iMission,false);

    if(b_showVideo)
    {
        b_showVideo = false;
        b_showVideo2 = true;
    }
}

event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
    if(bEnable<-2)
        EnableMission(GetNextMission(iMission,iNextNr),bEnable);
    if(bEnable==2)//enable ED base
        SetAvailableWorlds(GetAvailableWorlds() | 4);
    if(bEnable==3)//enable LC base
        SetAvailableWorlds(GetAvailableWorlds() | 8);
    if(bEnable==4)//you have win the game
    {
        EndGame("Video\\OutroUCS.wd1");
    }
    if(bEnable==5)//you loose the game
    {
        EndGame("Video\\OutroLost.wd1");
    }
}

event EndMission(int iMission, int nResult) //
{
    if(nResult==true)
        SetMissionState(iMission,stateAccomplished);
    else
        SetMissionState(iMission,stateFailed);

    if(!AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
    {
        if(n_TimeFlag != timeFlagHeavyVolcano)
        {
            n_TimeFlag = n_TimeFlag*2;
            b_showVideo = true;
        }
    }

    if(AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
    {
        ActivateMissionsEq(n_TimeFlag,n_TimeFlag,true);
    }
    else
    {
        SendCustomEvent(0,0,0,0);
        return;
    }

    SetSeason(1);
    if(n_TimeFlag==timeFlagEarlySpring)SetSeason(2);
    if(n_TimeFlag==timeFlagSpring) SetSeason(3);
    if(n_TimeFlag==timeFlagSummer) SetSeason(4);
    if(n_TimeFlag==timeFlagDesert) SetSeason(5);
    if(n_TimeFlag==timeFlagVolcano) SetSeason(6);
    if(n_TimeFlag==timeFlagHeavyVolcano) SetSeason(7);
}

```

```

        EnableChooseMissionButton(true);
    }

}

//*****
TutorialUCS.ec
campaign "translateTutorialUCS"
{
    state Initialize;
    state Nothing;

    state Initialize
    {
        CreateGamePlayer(1,raceUCS.playerLocal,null);
        CreateGamePlayer(2,raceED.playerAI,null);

        SetTime(100);

        RegisterMission("ITutorialUCS","UCS\Missions\\TutorialUCSmis","","0,0,0,0,0");
        LoadMission(0,0);
        SetAvailableWorlds(1);
        SetActivePlayerAndWorld(1,0);
        return Nothing;
    }
    state Nothing
    {
        return Nothing,200;
    }
}

}

//*****
Missions

Mission111.ec
mission "translateMission111"
{
    consts
    {
        findResource = 0;
        sendToBase20000 = 1;
    }

    player p_Enemy;
    player p_Player;

    int n_ResourceX;
    int n_ResourceY;
    int n_EnemyBaseX;
    int n_EnemyBaseY;
    int bShowFailed;
    int b_AIActivated;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Searching;
    state Mining;
    state Nothing;
}

state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(findResource,"translateGoalFindResources");
    RegisterGoal(sendToBase20000,"translateGoalSend20000",0);
    EnableGoal(findResource,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);

    //----- AI -----
    p_Enemy.EnableStatistics(false);

    if(GetDifficultyLevel()==0)
        p_Enemy.LoadScript("single\\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy.LoadScript("single\\singleMedium");
}

if(GetDifficultyLevel()==2)
    p_Enemy.LoadScript("single\\singleHard");

p_Enemy.EnableAIFeatures(aiControlOffense,false);
p_Enemy.EnableAIFeatures(aiControlDefense,false);

//----- Money -----
p_Player.SetMoney(10000);
p_Enemy.SetMoney(30000);

//----- Researches -----
p_Player.EnableResearch("RES_UCS_USL2",true);
//----- Buildings -----
// 1st tab
p_Player.EnableBuilding("UCSBPP",true);
p_Player.EnableBuilding("UCSBAA",true);
p_Player.EnableBuilding("UCSBAF",false);
p_Player.EnableBuilding("UCSBWB",false);
p_Player.EnableBuilding("UCSBAF",false);
// 2nd tab
p_Player.EnableBuilding("UCSBRF",false);
p_Player.EnableBuilding("UCSBTB",true);
// 3rd tab
p_Player.EnableBuilding("UCSBST",false);
// 4th tab
p_Player.EnableBuilding("UCSBCR",false);
p_Player.EnableBuilding("UCSBEN",false);
p_Player.EnableBuilding("UCSBTE",false);
p_Player.EnableBuilding("UCSBHO",false);
p_Player.EnableBuilding("UCSBEN",false);
p_Player.EnableBuilding("UCSBLZ",true);

//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,1000);

//----- Variables -----
n_ResourceX = GetPointX(1);
n_ResourceY = GetPointY(1);
n_EnemyBaseX = GetPointX(0);
n_EnemyBaseY = GetPointY(0);
b_AIactivated=false;
bCheckEndMission=false;
bShowFailed=true;

//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,200;//15 sec
}

state ShowBriefing
{
    Snow(n_ResourceX,n_ResourceY,310,500,5000,500,10);
    AddBriefing("translateBriefing111a");
    return Searching,100;
}

state Searching
{
    if(p_Player.IsPointLocated(n_ResourceX,n_ResourceY))
    {
        SetGoalState(findResource,goalAchieved);
        EnableGoal(sendToBase20000,true);
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        return Mining,200;
    }
    return Searching,100;
}

state Mining
{
    if(p_Player.GetMoneySentToBase()>=20000)
    {
        SetGoalState(sendToBase20000,goalAchieved);
        AddBriefing("translateAccomplished111");
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    if(!b_AIActivated &&
}

```

```

(p_Player.GetMoneySentToBase()>=15000 || 
p_Player.IsPointLocated(GetPointX(2),GetPointY(2),GetPointZ(2)) || 
p_Player.IsPointLocated(GetPointX(3),GetPointY(3),GetPointZ(3)) || 
p_Player.IsPointLocated(GetPointX(4),GetPointY(4),GetPointZ(4)) || 
p_Player.IsPointLocated(GetPointX(5),GetPointY(5),GetPointZ(5))) 
{ 
    b_AIActivated=true;
    AddBriefing("translateBriefing111b");
    p_Enemy.EnableAIFeatures(aiControlOffense,true);
    p_Enemy.EnableAIFeatures(aiControlDefense,true);
    // 1st tab
    p_Player.EnableBuilding("UCSBBA",true);
    p_Player.EnableBuilding("UCSBA",true);
    p_Player.EnableBuilding("UCSBAB",true);
    // 2nd tab
    p_Player.EnableBuilding("UCSBRE",true);
    p_Enemy.EnableStatistics(true);
}
return Mining,200;
}

//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    RegisterGoal(sendToBase20000,"translateGoalSend20000",p_Player.GetMoneySentToBase());
    if(bShowFailed)
    {
        if(IsGoalEnabled(sendToBase20000))
        {
if((ResourcesLeftInMoney()+p_Player.GetMoney()+p_Player.GetMoneySentToBase())<20000)
        {
            bShowFailed=false;
            SetGoalState(sendToBase20000,goalFailed);
            AddBriefing("translateFailed111a");
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed111b");
            EndMission(false);
        }
    }
}
event Timer1() //wolany co 1000 cykli
{
    if(GetMissionTime()==1000)
Snow(n_ResourceX,n_ResourceY,10,400,5000,800,10);
    if(GetMissionTime()==12000)
Snow(n_EnemyBaseX,n_EnemyBaseY,10,400,5000,800,10);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----



//Antarktyka klif
consts
{
    defendFor20min = 0;
    clicksPerDay = 16383;
}
player p_Enemy;
player p_Enemy2;
player p_Player;
int bCheckEndMission;
int bShowAc;

state Initialize;
state ShowBriefing;
state Nothing;

//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(defendFor20min,"translateGoal112",0,0);
    EnableGoal(defendFor20min,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);
    p_Enemy2 = GetPlayer(4);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleHard");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleHard");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    p_Enemy2.SetMaxTankPlatoonSize(6);
    p_Enemy2.SetNumberOfOffensiveTankPlatoons(10);
    p_Enemy2.SetNumberOfDefensiveTankPlatoons(0);
    p_Enemy2.EnableAIFeatures(aiControlOffense,false);
    p_Enemy2.EnableAIFeatures(aiControlDefense,false);

    p_Player.EnableAIFeatures(aiEnabled,false);

    p_Enemy2.SetNeutral(p_Enemy);
    p_Enemy2.SetNeutral(p_Enemy2);
    p_Player.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Player);
    //----- Money -----
    p_Player.SetMoney(5000);
    p_Enemy.SetMoney(50000);
    //----- Researches -----
    p_Player.EnableResearch("RES_UCS_USL3",true);
    p_Player.EnableResearch("RES_UCS_WCH2",true);
    p_Player.EnableResearch("RES_UCS_WSG2",true);

    p_Enemy.EnableResearch("RES_ED_WCH2",true);
    p_Enemy.EnableResearch("RES_MCH2",true);
    p_Enemy.EnableResearch("RES_ED_UA1",true);
    //----- Building -----
    p_Player.EnableBuilding("UCSBBA",false);
    p_Player.EnableBuilding("UCSBA",false);
    p_Player.EnableBuilding("UCSBAB",false);
    p_Player.EnableBuilding("UCSBRE",false);
    p_Player.EnableBuilding("UCSBT",false);
    p_Player.EnableBuilding("UCSBET",false);
    p_Player.EnableBuilding("UCSBF",false);
    p_Player.EnableBuilding("UCSBC",false);
    p_Player.EnableBuilding("UCSBAB",false);
    p_Player.EnableBuilding("UCSBWB",false);
    p_Player.EnableBuilding("UCSBFO",false);
    p_Player.EnableBuilding("UCSBST",true);
    p_Player.EnableBuilding("UCSBB",false);
    p_Player.EnableBuilding("UCSBPB",false);
}

```

Mission112.ec

mission "translateMission112"

```

p_Player.EnableBuilding("UCSBPC",false);
p_Player.EnableBuilding("UCSBHQ",false);
p_Player.EnableBuilding("UCSBSD",false);
p_Player.EnableBuilding("UCSBSH",false);
p_Player.EnableBuilding("UCSBE",false);
p_Player.EnableBuilding("UCSBN1",false);
p_Player.EnableBuilding("UCSBS1",false);
p_Player.EnableBuilding("UCSBLZ",false);
//----- Units -----
//----- Artifacts -----
//----- Timers -----
SetTimer(0,1000);
SetTimer(1,1000);
SetTimer(2,clicksPerDay*2);
//----- Variables -----
bCheckEndMission=false;
bShowAc=true;
//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    AddBriefing("translateBriefing112");
    return Nothing,100;
}

//-----
state Nothing
{
    return Nothing,600;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    int nTime;
    int nTimeDays;
    int nTimeHours;
    nTime = ((clicksPerDay*2) - GetMissionTime());
    if(nTime>=0)
    {
        nTimeDays = nTime / clicksPerDay;
        nTimeHours = (nTime % clicksPerDay)*24/clicksPerDay;
    }
    RegisterGoal(defendFor20min,"translateGoal112",nTimeDays,nTimeHours);
    SetConsoleText("translateMessage112",nTimeDays,nTimeHours);
    if(Time==0)SetConsoleText("1");
}
if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(lp_Player.GetNumberOfUnits() &&p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed112");
        EndMission(false);
    }
    if(bShowAc && lp_Enemy.GetNumberOfUnits() &&
lp_Enemy.GetNumberOfBuildings())
    {
        bShowAc=false;
        AddBriefing("translateAccomplished112");
        p_Player.SetMoney(p_Player.GetMoney()+5000);
        p_Player.EnableBuilding("UCSBLZ",true);
        EnableEndMissionButton(true);
    }
}
//-----
event Timer1() //wolany co 6000 cykli
{
    if(GetDifficultyLevel()!=0)
p_Enemy.RussianAttack(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
0);
Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),20,50,2500,30,7
);
}
//-----
//-----
event Timer2() //wolany co 20min = 20*60*20 = 1200*20 = 24000
{
    if(bShowAc)
    {
        bShowAc=false;
        if(GetDifficultyLevel()!=0)
        {
            p_Enemy2.EnableAIFeatures(aiRush,true);
            p_Enemy2.EnableAIFeatures(aiControlOffense,true);
        }
        SetGameState(defendFor20min,goalAchieved);
        AddBriefing("translateAccomplished112");
        p_Player.SetMoney(p_Player.GetMoney()+5000);
        p_Player.EnableBuilding("UCSBLZ",true);
        EnableEndMissionButton(true);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    p_Enemy2.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Enemy2);
}
}


```

```

p_Player.SetNeutral(p_Neutral);
p_Neutral.SetNeutral(p_Player);
p_Player.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableAIFeatures(aiEnabled,false);
//----- Money -----
p_Player.SetMoney(0);
//----- Researches -----
//----- Buildings -----
//----- Units -----
p_Tester1 = GetUnit(GetPointX(0),GetPointY(0),0);
p_Tester2 = GetUnit(GetPointX(1),GetPointY(1),0);
CallCamera();
SelectUnit(p_Tester1,false);
//----- Artifacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,6000);
//----- Variables -----
n_TeleportOutX = GetPointX(2);
n_TeleportOutY = GetPointY(2);
bCheckEndMission = false;
bShowBriefing = false;
//----- Camera -----
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),8,0,20,0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing113a");
    return TestEntrance1,20;
}

//-----
state TestEntrance1
{
    if(bShowBriefing)
    {
        bShowBriefing = false;
        AddBriefing("translateBriefing113b");
        return TestEntrance2,20;
    }
    if(p_Tester1.GetLocationZ() == 1)
    {
        bShowBriefing = true;
    }

p_Player.LookAt(p_Tester1.GetLocationX(),p_Tester1.GetLocationY(),8,0,20,1);
    return TestEntrance1,80;
}
//-----
state TestEntrance2
{
    if(bShowBriefing)
    {
        bShowBriefing = false;
        KillArea(65535,p_Tester1.GetLocationX(),p_Tester1.GetLocationY(),0,0,1);
        return TestEntranceBriefing,100;
    }
    if(p_Tester1.GetLocationZ() == 0)
    {
        bShowBriefing = true;
    }

p_Player.LookAt(p_Tester1.GetLocationX(),p_Tester1.GetLocationY(),8,0,20,0);
    return TestEntrance2,20;
}
//-----
state TestEntranceBriefing
{
    SetGoalState(testTunnelEntrance,goalAchieved);
    EnableGoal(testTeleport,true);
    AddBriefing("translateBriefing113c");
    p_Neutral.GiveAllBuildingsTo(p_Player);
    p_Neutral.GiveAllUnitsTo(p_Player);
    CallCamera();
    SelectUnit(p_Tester2,false);
    return TestTeleportStart,20;
}
//-----
state TestTeleportStart
{
    p_Player.DelayedLookAt(GetPointX(1),GetPointY(1),6,0,20,0,60,1);
    return TestTeleport,20;
}
state TestTeleport
{
    if(p_Tester2.DistanceTo(n_TeleportOutX,n_TeleportOutY) < 4)
    {
        p_Player.DelayedLookAt(n_TeleportOutX,n_TeleportOutY,8,128,20,0,60,1);
        return BlowUpTeleport,100;
    }
    return TestTeleport,20;
}
//-----
state BlowUpTeleport
{
    KillArea(65535,n_TeleportOutX,n_TeleportOutY,0,8);
    return LastBriefing,20;
}
//-----
state LastBriefing
{
    SetGoalState(testTeleport,goalAchieved);
    AddBriefing("translateBriefing113d");
    EnableEndMissionButton(true);
    return Final,500;
}

//-----
state Final
{
    return Final,500;
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission = false;
        if(!p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed113");
            EndMission(false);
        }
    }
}

//-----
event Timer1() //wolany co 6000 cykli 5min
{
    Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),40,400,2500,800,5);
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}

//-----
event EndMission()
{
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.SetEnemy(p_Player);
}



## Mission114.ec


mission "translateMission114"
{
    consts
    {
        findResource = 0;
        sendToBase30000 = 1;
    }

    player pEnemy;
    player pPlayer;
}

```

```

int bShowFailed;
int bCheckEndMission;
int bSwitchAI;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;
//-----
state Initialize
{
    int nStartAttack;
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(findResource,"translateGoalFindResources");
    RegisterGoal(sendToBase30000,"translateGoalSend30000",0);
    EnableGoal(findResource,true);
    EnableGoal(sendToBase30000,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(1);
    pEnemy = GetPlayer(2);
    //----- AI -----
    pPlayer.SetMilitaryUnitsLimit(15000);
    if(GetDifficultyLevel()==0)
    {
        pEnemy.LoadScript("single\singleEasy");
        nStartAttack = 15;
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy.LoadScript("single\singleMedium");
        pEnemy.SetNumberOfOffensiveTankPlatoons(2);
        nStartAttack = 15;
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy.LoadScript("single\singleHard");
        nStartAttack = 15;
    }
}

//pEnemy.SetNumberOfOffensiveHelicopterPlatoons(0);
//pEnemy.SetNumberOfDefensiveHelicopterPlatoons(0);

pPlayer.EnableAIFeatures(aiEnabled,false);
pEnemy.EnableAIFeatures(aiEnabled,true);
pEnemy.EnableAIFeatures(aiControlOffense,false);
//----- Money -----
pPlayer.SetMoney(10000);
pEnemy.SetMoney(10000);
//----- Researches -----
pEnemy.EnableResearch("RES_ED_WCA2",true);
pEnemy.EnableResearch("RES_ED_MSC2",true);
pEnemy.EnableResearch("RES_ED_UMT1",true);
pEnemy.EnableResearch("RES_ED_UA12",true);
pEnemy.EnableResearch("RES_ED_RePhand",true);

pPlayer.EnableResearch("RES_UCS_WACH2",true);
pPlayer.EnableResearch("RES_UCS_WSR1",true);
pPlayer.EnableResearch("RES_MCH2",true);
pPlayer.EnableResearch("RES_UCS_NG2",true);
pPlayer.EnableResearch("RES_UCS_GARG1",true);
//----- Buildings -----
pPlayer.EnableBuilding("UCSBTE",false);
pPlayer.EnableBuilding("UCSBEN1",false);
//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,6000);
SetTimer(2,nStartAttack*20*60);
//----- Variables -----
bShowFailed=true;
bSwitchAI=false;
bCheckEndMission=false;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,150;/15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing114");
    Snow(pPlayer.GetStartingPointX()-10,pPlayer.GetStartingPointY()+10,30,400,5000,800,10);
    return Mining,100;
}
//-----
state Mining
{
    if(GetGoalState(findResource)==goalAchieved && pPlayer.IsPointLocated(GetPointX(0),GetPointY(0),0))
    {
        SetGoalState(findResource, goalAchieved);
    }
    if(GetGoalState(sendToBase30000)==goalAchieved && pPlayer.GetMoneySentToBase()>=30000)
    {
        SetGoalState(sendToBase30000, goalAchieved);
        AddBriefing("translateAccomplished114");
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    RegisterGoal(sendToBase30000,"translateGoalSend30000",pPlayer.GetMoneySentToBase());
    if(bSwitchAI && pPlayer.GetMoneySentToBase()>15000)
    {
        pEnemy.EnableAIFeatures(aiControlOffense,true);
        bSwitchAI=false;
    }
    if(bShowFailed)
    {
        if((ResourcesLeftInMoney() + pPlayer.GetMoneySentToBase() + pPlayer.GetMoney()) < 30000)
        {
            bShowFailed=false;
            SetGoalState(sendToBase30000, goalFailed);
            AddBriefing("translateFailed114a");
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(ipPlayer.GetNumberOfUnits() && lpPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed114b");
            EndMission(false);
        }
    }
}
//-----
event Timer1() //wolany co 6000 cykli 5min
{
    Snow(pPlayer.GetStartingPointX()-10,pPlayer.GetStartingPointY()+10,30,400,5000,800,10);
}
//-----
event Timer2() //wolany co 6000 cykli 5min
{
    pEnemy.EnableAIFeatures(aiControlOffense,true);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    if(bSwitchAI)
    {
        if(GetDifficultyLevel()!=0)
            pEnemy.EnableAIFeatures(aiControlOffense,true);
        bSwitchAI=false;
    }
    bCheckEndMission=true;
}

```

```

}
}

event BuildingDestroyed(unit u_Unit)
{
    if(bSwitchAI)
    {
        if(GetDifficultyLevel()!=0)
            p_Enemy.EnableAIFeatures(aiControlOffense,true);
        bSwitchAI=false;
    }
    bCheckEndMission=true;
}
}

```

Mission121.ec

mission "translateMission121"
{/Baikal - zniszczyc centrum naukowe ED

```

consts
{
    establishCommunication = 0;
    destroyEDBase = 1;
}

```

```

player p_Enemy;
player p_Player;
player p_Mechs;
player p_ED2;

```

```

int bShowFailed;
int bCheckEndMission;
int bShowVideo;

```

```

state Initialize;
state ShowBriefing;
state StartLaser;
state EnableMechs;
state EnableAI;
state Fighting;
state Nothing;
//-----
state Initialize
{

```

```

    player tmpPlayer;
    //----- Goals -----

```

```

    RegisterGoal(establishCommunication,"translateGoal121a");
    RegisterGoal(destroyEDBase,"translateGoal121b");

```

```

    EnableGoal(enableCommunication,true);
    EnableGoal(destroyEDBase,true);

```

```

    //----- Temporary players -----

```

```

    tmpPlayer = GetPlayer(3);

```

```

    tmpPlayer.EnableStatistics(false);

```

```

    //----- Players -----

```

```

    p_Player = GetPlayer(1);

```

```

    p_Enemy = GetPlayer(2);

```

```

    p_Mechs = GetPlayer(4);

```

```

    p_ED2 = GetPlayer(8);

```

```

    //----- AI -----

```

```

    p_Mechs.EnableStatistics(false);

```

```

    p_ED2.EnableStatistics(false);

```

```

    p_Mechs.SetNeutral(p_Enemy);

```

```

    p_Mechs.SetNeutral(p_ED2);

```

```

    p_Enemy.SetNeutral(p_Mechs);

```

```

    p_Enemy.SetNeutral(p_ED2);

```

```

    p_ED2.SetNeutral(p_Mechs);

```

```

    if(GetDifficultyLevel()==0)

```

```

        p_Enemy.LoadScript("single\singleEasy");

```

```

    if(GetDifficultyLevel()==1)

```

```

        p_Enemy.LoadScript("single\singleMedium");

```

```

    if(GetDifficultyLevel()==2)

```

```

        p_Enemy.LoadScript("single\singleHard");

```

```

    p_Player.EnableAIFeatures(aiEnabled,false);

```

```

    p_Enemy.EnableAIFeatures(aiEnabled,true);

```

```

    p_Enemy.EnableAIFeatures(aiControlOffense,false);

```

```

    p_Mechs.EnableAIFeatures(aiEnabled,false);

```

```

    p_ED2.EnableAIFeatures(aiEnabled,false);

```

```

    p_Mechs.EnableAIFeatures(aiRejectAlliance,false);

```

```

    p_Player.SetAlly(p_Mechs);

```

```

    p_Player.SetMilitaryUnitsLimit(15000);

```

```

    //----- Money -----

```

```

    p_Player.SetMoney(10000);

```

```

    p_Enemy.SetMoney(30000);

```

//----- Researches -----

```

p_Enemy.EnableResearch("RES_ED_WSL1",true);

```

```

p_Enemy.EnableResearch("RES_ED_UA21",true);

```

```

p_Enemy.EnableResearch("RES_ED_UW11",true);

```

```

p_Player.EnableResearch("RES_UCS_WASR1",true);

```

```

p_Player.EnableResearch("RES_UCS_UOH2",true);

```

```

p_Player.EnableResearch("RES_UCS_UML1",true);

```

```

p_Player.EnableResearch("RES_UCS_Rephand",true);

```

//----- Buildings -----

```

p_Player.EnableBuilding("UCSBTE",false);

```

```

p_Player.EnableBuilding("UCSBEN1",false);

```

//----- Units -----

//----- Artefacts -----

//----- Timers -----

```

SetTimer(0,100);

```

```

SetTimer(1,6000);

```

//----- Variables -----

```

bShowFailed = true;

```

```

bCheckEndMission = false;

```

```

bShowVideo=true;

```

//----- Camera -----

```

CallCamera();

```

```

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);

```

```

    return ShowBriefing,100;//15 sec
}

```

state ShowBriefing

```

{
    AddBriefing("translateBriefing121");
    EnableNextMission(0,true);
    return StartLaser,1200;
}

```

//-----

state StartLaser

```

{
    p_Enemy.SetEnemy(p_Mechs);
    return EnableMechs,400;
}

```

//-----

state EnableMechs

```

{
    SetGoalState(establishCommunication,goalAchieved);
}

```

```

p_Player.DelayedLookAt(p_Mechs.GetStartingPointX(),p_Mechs.GetStartingPointY(),6,0,15,0,60,1);

```

```

    p_Mechs.GiveAllUnitsTo(p_Player);

```

```

    p_ED2.GiveAllUnitsTo(p_Enemy);

```

```

    p_ED2.GiveAllBuildingsTo(p_Enemy);
}

```

```

p_Enemy.RussianAttack(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),0,0);

```

```

    return EnableAI,6000;
}

```

//-----

state EnableAI

```

{
    p_Enemy.EnableAIFeatures(aiControlOffense,true);
    return Fighting,200;
}

```

//-----

state Fighting

```

{
    return Fighting,200;
}

```

//-----

state Nothing

```

{
    if(bShowVideo)

```

```

        ShowVideo("CS111");

```

```

        bShowVideo=false;
}

```

```

return Nothing, 500;
}

```

event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state

Initialize

```

{
    if(bCheckEndMission)

```

```

        bCheckEndMission=false;
}

```

```

if(!p_Player.GetNumberofUnits() && lp_Player.GetNumberOfBuildings())

```

112

```

    {
        AddBriefing("translateFailed121");
        EndMission(false);
    }
    if(GetGoalState(destroyEDBase)!=goalAchieved &&
       lp_Enemy.GetNumberOfUnits() && lp_Enemy.GetNumberOfBuildings())
    {
        SetGoalState(destroyEDBase,goalAchieved);
        AddBriefing("translateAccomplished121");
        EnableEndMissionButton(true);
    }
}

//-----
event Timer1() //wolny co 6000 cykli 5min
{
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY()-
10,30,400,5000,800,10);
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    p_Mechs.SetEnemy(p_Enemy);
    p_Mechs.SetEnemy(p_ED2);
    p_Enemy.SetEnemy(p_Mechs);
    p_ED2.SetEnemy(p_Mechs);
}
}

Mission122.ec
mission "translateMission122"
{/Projekt Teleport - part II
consts
{
    testTunnelEntrance = 0;
    testTeleport = 1;
    destroyMech = 2;
}

player pPlayer;
player pNeutral;
player pSecurity;

unitex uTester1;
unitex uTeleport;

int n_TeleportOutX;
int n_TeleportOutY;
int bCheckEndMission;
int bShowBriefing;

state Initialize;
state ShowBriefing;
state TestEntrance1;
state TestEntrance2;
state TestEntranceBriefing;
state TestTeleport;
state TeleportBriefing;
state DestroyTech;
state LastBriefing;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(testTunnelEntrance,"translateGoal122a");
    RegisterGoal(testTeleport,"translateGoal122b");
    RegisterGoal(destroyMech,"translateGoal122c");
    EnableGoal(testTunnelEntrance,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
}

tmpPlayer = GetPlayer(2);
tmpPlayer.EnableStatistics(false);
//----- Players -----
pPlayer = GetPlayer(1);
pNeutral = GetPlayer(6);
pSecurity = GetPlayer(4);
//----- AI -----
pNeutral.EnableStatistics(false);
pSecurity.EnableStatistics(false);

pPlayer.SetNeutral(pNeutral);
pNeutral.SetNeutral(pPlayer);

pPlayer.SetNeutral(pSecurity);
pSecurity.SetNeutral(pPlayer);

pNeutral.SetNeutral(pSecurity);
pSecurity.SetNeutral(pNeutral);

pPlayer.EnableAIFeatures(aiEnabled,false);
pNeutral.EnableAIFeatures(aiEnabled,false);
pSecurity.EnableAIFeatures(aiEnabled,false);
//----- Money -----
pPlayer.SetMoney(10000);
//----- Researches -----
//----- Buildings -----
pPlayer.EnableBuilding("UCSBA",false);
pPlayer.EnableBuilding("UCSFA",false);
pPlayer.EnableBuilding("UCSPP",false);
pPlayer.EnableBuilding("UCSBPR",false);
pPlayer.EnableBuilding("UCSBTB",false);
pPlayer.EnableBuilding("UCSBET",false);
pPlayer.EnableBuilding("UCSRF",false);
pPlayer.EnableBuilding("UCSRC",false);
pPlayer.EnableBuilding("UCSAB",false);
pPlayer.EnableBuilding("UCSWB",false);
pPlayer.EnableBuilding("UCSFO",false);
pPlayer.EnableBuilding("UCSBST",false);
pPlayer.EnableBuilding("UCSBTF",false);
pPlayer.EnableBuilding("UCSPB",false);
pPlayer.EnableBuilding("UCSPC",false);
pPlayer.EnableBuilding("UCSHQ",false);
pPlayer.EnableBuilding("UCBSD",false);
pPlayer.EnableBuilding("UCBSH",false);
pPlayer.EnableBuilding("UCSBE",false);
pPlayer.EnableBuilding("UCSBEN1",true);
pPlayer.EnableBuilding("UCSSS",false);
pPlayer.EnableBuilding("UCSLZ",false);
pPlayer.EnableBuilding("UCSBE",false);
//----- Units -----
uTester1 = GetUnit(GetPointX(0),GetPointY(0));
uTester1 = GetUnit(GetPointX(1),GetPointY(1));
//----- Artefacts -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,6000);
//----- Variables -----
n_TeleportOutX = GetPointX(2);
n_TeleportOutY = GetPointY(2);
bCheckEndMission = false;
bShowBriefing = false;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),8,0,20,0);
SelectUnit(uTester1,false);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    AddBriefing("translateBriefing122a");
    return TestEntrance1,20;
}

state TestEntrance1
{
    if(bShowBriefing)
    {
        bShowBriefing = false;
        AddBriefing("translateBriefing122b");
        return TestEntrance2,20;
    }
}

```

```

if(uTester1.GetLocationZ() == 1)
{
    bShowBriefing = true;
}

pPlayer.LookAt(uTester1.GetLocationX(), uTester1.GetLocationY(), 8, 0, 20, 1);
return TestEntrance1, 80;
}

state TestEntrance2
{
    if(uTester1.GetLocationZ() == 0)
    {
        pPlayer.LookAt(uTester1.GetLocationX(), uTester1.GetLocationY(), 8, 0, 20, 1);
        return TestEntranceBriefing, 100;
    }
    return TestEntrance2, 20;
}

state TestEntranceBriefing
{
    SetGoalState(testTunnelEntrance.goalAchieved);
    EnableGoal(testTeleport, true);
    AddBriefing("translateBriefing12c"); //OK it works
    pNeutral.GiveAllBuildingsTo(pPlayer);
    return TestTeleport, 20;
}

state TestTeleport
{
    if(uTester1.DistanceTo(n_TeleportOutX, n_TeleportOutY) < 4)
    {
        pPlayer.DelayedLookAt(n_TeleportOutX, n_TeleportOutY, 8, 0, 20, 0, 20, 1);
        return TeleportBriefing, 100;
    }
    return TestTeleport, 20;
}

state TeleportBriefing
{
    uTester1.ChangePlayer(GetPlayer(2));
    uTeleport.ChangePlayer(GetPlayer(6));
    pSecurity.GiveAllUnitsTo(pPlayer);
    SetGoalState(testTeleport.goalAchieved);
    EnableGoal(destroyMech, true);
    AddBriefing("translateBriefing12d");
    return DestroyMech, 50;
}

state DestroyMech
{
    if(!uTester1.IsLive())
    {
        return LastBriefing, 80;
    }
    return DestroyMech, 50;
}

state LastBriefing
{
    SetGoalState(destroyMech.goalAchieved);
    AddBriefing("translateAccomplished122");
    EnableEndMissionButton(true);
    return Final, 500;
}

state Final
{
    return Final, 500;
}

event Timer0() //wolany co 100 cykli< ustawiione funkcja SetTimer w state Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if((pPlayer.GetNumberOfBuildings()))
        {
            AddBriefing("translateFailed122");
            EndMission(false);
        }
    }
}

```

```

event Timer1() //wolany co 6000 cykli 5min
{
}

Snow(pPlayer.GetStartingPointX(), pPlayer.GetStartingPointY(), 40, 400, 2500, 800, 5)
;

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event EndMission()
{
    pPlayer.SetEnemy(pNeutral);
    pNeutral.SetEnemy(pPlayer);

    pPlayer.SetEnemy(pSecurity);
    pSecurity.SetEnemy(pPlayer);

    pNeutral.SetEnemy(pSecurity);
    pSecurity.SetEnemy(pNeutral);
}

}



## Mission123.ec


mission "translateMission123"
//Alasca track ed tanks
consts
{
    destroyEDForces = 0;
}

player pEnemy;
player pPlayer;
int nWayPoint;

state Initialize;
state ShowBriefing;
state OnTheWay;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    //-----goals-----
    RegisterGoal(destroyEDForces, "translateGoal123");
    EnableGoal(destroyEDForces, true);

    //-----temporary players-----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    //-----players-----
    pPlayer = GetPlayer(1);
    pEnemy = GetPlayer(2);
    //-----AI-----
    pPlayer.EnableAIFeatures(aiEnabled, false);
    pEnemy.EnableAIFeatures(aiEnabled, false);
    pPlayer.SetMilitaryUnitsLimit(15000);
    //-----money-----
    pPlayer.SetMoney(10000);
    pEnemy.SetMoney(0);
    //-----researches-----
    pEnemy.EnableResearch("RES_ED_Rephand2", true);

    pPlayer.EnableResearch("RES_MS2", true);
    pPlayer.EnableResearch("RES_UCS_UML1", true);
    pPlayer.EnableResearch("RES_UCS_BMD", true);
    //-----buildings-----
    pPlayer.EnableBuilding("UCSBEN1", false);
    //-----timers-----
    SetTimer(0, 200);
    //-----variables-----
    nWayPoint = 0;

    //-----camera-----
    CallCamera();
}

pPlayer.LookAt(pPlayer.GetStartingPointX(), pPlayer.GetStartingPointY(), 6, 0, 20, 0);

```

```

        return ShowBriefing,120;
    }
    //-----
    state ShowBriefing
    {
        EnableNextMission(0,true);
        AddBriefing("translateBriefing123");
        return OnTheWay,600;
    }
    //-----
    state OnTheWay
    {
        if(pPlayer.IsPointLocated(GetPointX(nWayPoint),GetPointY(nWayPoint),0))
        {
            if(nWayPoint==0)
                ShowVideo("CS112");
            nWayPoint=nWayPoint+1;
            if(nWayPoint>12)
            {
                nWayPoint=12;
            }
        }
        if(nWayPoint)
            pEnemy.RussianAttack(GetPointX(nWayPoint),GetPointY(nWayPoint),0);
    }
    if(!pPlayer.GetNumberOfUnits() && !pPlayer.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed123");
        EndMission(false);
    }
    if(!pEnemy.GetNumberOfUnits())
    {
        SetGoalState(destroyEDForces, goalAchieved);
        AddBriefing("translateAccomplished123");
        EnableEndMissionButton(true);
        return Final,500;
    }
    return OnTheWay,200;
}
//-----
state Final
{
    return Final,500;
}

//-----
event EndMission()
{
}
}

```

Mission124.ec

```

mission "translateMission124"
(/neo home
consts
{
    findStolenPlans = 0;
    destroyNeoHome = 1;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Player;

unitex u_NeoHome;
unitex u_Mech1;
unitex u_Mech2;

int bShowFailed;
int bCheckEndMission;

state Initialize;
state ShowBriefing;
state NeoMessage;
state ShowBriefing2;
state Fighting;
state Nothing;
//-----
state Initialize
{
    int nEnemyMoney;
    player tmpPlayer;
    //-----
    //-----goals-----
    RegisterGoal(findStolenPlans,"translateGoal124a");
    RegisterGoal(destroyNeoHome,"translateGoal124b",0);
    EnableGoal(findStolenPlans,true);
    //-----temporary players-----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //-----players-----
    p_Player = GetPlayer(1);
    p_Enemy1 = GetPlayer(2);
    p_Enemy2 = GetPlayer(4);
    p_Enemy3 = GetPlayer(5);
    //-----AI-----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy3.LoadScript("single\singleEasy");
        p_Enemy1.EnableAIFeatures(aiControlOffense,false);
        p_Enemy2.EnableAIFeatures(aiControlOffense,false);
        p_Enemy3.EnableAIFeatures(aiControlOffense,false);
        p_Enemy2.EnableAIFeatures(aiDefenseTowers,false);
        p_Enemy2.EnableAIFeatures2(aiBuildingTowers,false);
        nEnemyMoney=10000;
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy3.LoadScript("single\singleMedium");
        p_Enemy1.EnableAIFeatures(aiControlOffense,false);
        p_Enemy2.EnableAIFeatures(aiControlOffense,false);
        p_Enemy3.EnableAIFeatures(aiDefenseTowers,false);
        nEnemyMoney=20000;
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy3.LoadScript("single\singleHard");
        p_Enemy1.EnableAIFeatures(aiControlOffense,false);
        nEnemyMoney=30000;
    }
    p_Enemy1.SetNeutral(p_Enemy2);
    p_Enemy1.SetNeutral(p_Enemy3);
    p_Enemy2.SetNeutral(p_Enemy1);
    p_Enemy2.SetNeutral(p_Enemy3);
    p_Enemy3.SetNeutral(p_Enemy1);
    p_Enemy3.SetNeutral(p_Enemy2);
    p_Player.EnableAIFeatures(aiEnabled,false);
    //-----money-----
    p_Player.SetMoney(30000-nEnemyMoney/2);
    p_Enemy1.SetMoney(nEnemyMoney);
    p_Enemy2.SetMoney(nEnemyMoney);
    p_Enemy3.SetMoney(nEnemyMoney);
    //-----researches-----
    p_Enemy1.EnableResearch("RES_ED_WSR1",true);
    p_Enemy1.EnableResearch("RES_ED_BMD",true);
    p_Enemy2.CopyResearches(p_Enemy1);
    p_Enemy3.CopyResearches(p_Enemy1);

    p_Player.EnableResearch("RES_UCS_UMI1",true);
    p_Player.EnableResearch("RES_UCS_USZ2",true);
    p_Player.EnableResearch("RES_UCS_UBS1",true);
    p_Player.EnableResearch("RES_UCS_RepHand2",true);
    //-----buildings-----
    p_Player.EnableBuilding("UCSBTE",false);
    //-----units-----
    u_NeoHome = GetUnit(GetPointX(1),GetPointY(1),0);
    u_Mech1 = GetUnit(GetPointX(2),GetPointY(2),0);
    u_Mech2 = GetUnit(GetPointX(3),GetPointY(3),0);
    //-----Artefacts-----
    CreateArtifact("NEASPECIAL1",GetPointX(0),GetPointY(0),0,0,artefactSpecialAI0th
er);
    //-----Timers-----
    SetTimer(0,100);
    //-----variables-----
}

```

```

bShowFailed=true;
bCheckEndMission=false;
//-----camera-----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,150;//15 sec
}
//-----
state ShowBriefing
{
AddBriefing("translateBriefing124a");
Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY()+10,30,400,5000,
800,10);
return NeoMessage,600;
}
//-----
state NeoMessage
{
AddBriefing("translateBriefing124b");
u_Mech1.ChangePlayer(p_Enemy1);
u_Mech2.ChangePlayer(p_Enemy2);
return ShowBriefing,200;
}
//-----
state ShowBriefing
{
EnableGoal(destroyNeoHome,true);
AddBriefing("translateBriefing124c");
ShowArea(2,GetPointX(1),GetPointY(1),0,4);
return Fighting,200;
}
//-----
state Fighting
{
if(GetGoalState(destroyNeoHome)==goalAchieved && !u_NeoHome.IsLive())
{
SetGoalState(destroyNeoHome, goalAchieved);
AddBriefing("translateBriefing124d");
p_Player.GiveAllUnitsTo(p_Enemy1);
}
if(GetGoalState(destroyNeoHome)==goalAchieved &&
GetGoalState(findStolenPlans)==goalNotAchieved)
{
AddBriefing("translateAccomplished124");
EnableEndMissionButton(true);
EnableNextMission(0,true);
return Nothing,500;
}
return Fighting,200;
}
//-----
state Nothing
{
return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
if(bCheckEndMission)
{
bCheckEndMission=false;

if(GetGoalState(destroyNeoHome)==goalAchieved &&
GetGoalState(findStolenPlans)==goalNotAchieved)
return;

if(!p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
{
AddBriefing("translateFailed124");
EnableNextMission(1,true);
EndMission(false);
}
}
}
//-----
event UnitDestroyed(unit u_Unit)
{
bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
}
}
//-----
bCheckEndMission=true;
}
//-----
event Artefact(int alD,player piPlayer)
{
if(piPlayer==p_Player) return false;
if(GetMissionTime()<1200*(60-(GetDifficultyLevel()*15))) //60,45,30
minut
{
SetGoalState(findStolenPlans,goalAchieved);
AddBriefing("translateBriefing124e");
}
else
{
SetGoalState(findStolenPlans,goalFailed);
AddBriefing("translateBriefing124f");
}
return true; //usuwa sie
}
}



## Mission131.ec


mission "translateMission131"
{
consts
{
destroyEnemy = 0;
}

player pEnemy;
player pPlayer;

unitex uMainBaseLC;

int bCheckEndMission;
state Initialize;
state ShowBriefing;
state Fighting;
state Nothing;
//-----
state Initialize
{
player tmpPlayer;
//----- Goals -----
RegisterGoal(destroyEnemy,"translateGoal131");
EnableGoal(destroyEnemy,true);
//----- Temporary players -----
tmpPlayer = GetPlayer(2);
tmpPlayer.EnableStatistics(false);
//----- Players -----
pPlayer = GetPlayer(1);
pEnemy = GetPlayer(3);
//----- AI -----
pPlayer.SetMilitaryUnitsLimit(20000);
if(GetDifficultyLevel()==0)
{
pEnemy.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel()==1)
{
pEnemy.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
pEnemy.LoadScript("single\singleHard");
}

pPlayer.EnableAIFeatures(false);
pPlayer.SetMilitaryUnitsLimit(20000);
//----- Money -----
pPlayer.SetMoney(10000);
pEnemy.SetMoney(20000);
//----- Researches -----
pPlayer.EnableResearch("RES_UCS_WSP1",true);
pPlayer.EnableResearch("RES_UCS_UOH3",true);
//----- Buildings -----
if(pPlayer.GetScriptData(5)==12)
pPlayer.EnableBuilding("UCSSTE",false);
//----- Units -----
uMainBaseLC = GetUnit(GetPointX(0),GetPointY(0),0);
//----- Artefacts -----
//----- Timers -----
SetTimer(0,200);
}
}

```

```

//----- Variables -----
bCheckEndMission=false;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,150://15 sec
}
//----- state ShowBriefing -----
{
    EnableNextMission(0,true);
    AddBriefing("translateBriefing131");
    return Fighting,100;
}
//----- state Fighting -----
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;

        if(!uMainBaseLC.IsLive())
        {
            EnableGoal(destroyEnemy,false);
            pEnemy.EnableAIFeatures(aiRejectAlliance,false);
            pEnemy.EnableAIFeatures(aiEnabled,false);
            pPlayer.SetAI(pEnemy);
            AddBriefing("translateAccomplished131",pEnemy.GetName());
            EnableEndMissionButton(true);
            return Nothing,500;
        }

        if((pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings()))
        {
            AddBriefing("translateFailed131");
            EndMission(false);
        }
    }
    return Fighting,200;
}
//----- state Nothing -----
{
    return Nothing, 500;
}
//----- event UnitDestroyed(unit uUnit) -----
{
    bCheckEndMission=true;
}
//----- event BuildingDestroyed(unit uUnit) -----
{
    bCheckEndMission=true;
}
//----- event EndMission() -----
{
    pPlayer.SetEnemy(pEnemy);
    pEnemy.SetEnemy(pPlayer);
}
}



## Mission132.ec


mission "translateMission132"
{/uwlomic Grizzlich
consts
{
    enablePrototypes = 0;
    destroyEnemyBase = 1;
}

player p_Enemy;
player p_Neutral1; // landing zone
player p_Neutral2; // grizzly
player p_Player;

int bCheckEndMission;
int bEnableLZ;

state initialize;
state ShowBriefing;
state Fighting;

state Nothing;
//-----
state Initialize
{
    int nEnemyMoney;
    player tmpPlayer;

    //-----goals-----
    RegisterGoal(enablePrototypes,"translateGoal132a");
    RegisterGoal(destroyEnemyBase,"translateGoal132b");
    EnableGoal(enablePrototypes,true);

    //-----temporary players-----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);

    //-----players-----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);
    p_Neutral1 = GetPlayer(6);
    p_Neutral2 = GetPlayer(8);

    //-----AI-----
    p_Neutral1.EnableStatistics(false);
    p_Neutral2.EnableStatistics(false);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\\singleEasy");
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        nEnemyMoney=10000;
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy.LoadScript("single\\singleMedium");
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        nEnemyMoney=20000;
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\\singleHard");
        nEnemyMoney=30000;
    }
}

p_Neutral1.EnableAIFeatures(aiEnabled,false);
p_Neutral2.EnableAIFeatures(aiEnabled,false);
p_Player.EnableAIFeatures(aiEnabled,false);

p_Neutral1.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral1);
p_Neutral2.SetNeutral(p_Enemy);
p_Player.SetNeutral(p_Neutral2);
p_Enemy.SetNeutral(p_Neutral2);

//-----money-----
p_Player.SetMoney(0);
p_Enemy.SetMoney(nEnemyMoney);

//-----researches-----
p_Enemy.EnableResearch("RES_ED_WMR1",true);
p_Enemy.EnableResearch("RES_MSR2",true);
p_Enemy.EnableResearch("RES_ED_UHW1",true);
p_Enemy.EnableResearch("RES_ED_BHD",true);

p_Player.EnableResearch("RES_UCS_WHG1",true);
p_Player.EnableResearch("RES_UCS_UHL1",true);
p_Player.EnableResearch("RES_UCS_BHD",true);

if(p_Player.GetScriptData(5)==12)
    p_Player.EnableBuilding("UCSBTE",false);

//-----units-----
//-----Artefacts-----

CreateArtifact("NEASPECIAL1",GetPointX(0),GetPointY(0),1,0,artifactSpecialAI0th
er);
//-----Timers-----
SetTimer(0,100);
//-----variables-----
bCheckEndMission=false;
bEnableLZ=true;
//-----camera-----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,150://15 sec
}
//----- state ShowBriefing -----
}

```

```

    {
        AddBriefing("translateBriefing132a");
        Rain(p_Player.GetStartingPointX(), p_Player.GetStartingPointY()-
20,30,400,5000,800,5);
        return Fighting,600;
    }
}

state Fighting
{
    if(!IsGoalEnabled(destroyEnemyBase) &&
    lp_Enemy.GetNumberOfBuildings())
    {
        SetGoalState(destroyEnemyBase,goalAchieved);
        AddBriefing("translateAccomplished123");
        EnableEndMissionButton(true);
        p_Neutral1.GiveAllBuildingsTo(p_Player);
        return Nothing,100;
    }
    return Fighting,200;
}
}

state Nothing
{
    return Nothing, 500;
}
}

event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;

        if(p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed132");
            EndMission(false);
        }
    }
}
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

event BuildingDestroyed(unit u_Unit)
{
    if(bEnableLZ)
    {
        bEnableLZ=false;
        p_Neutral1.GiveAllBuildingsTo(p_Player); //uwaga destroyed przychodzi
tez gdy budynek jest konwertowany.
    }
    bCheckEndMission=true;
}
}

event EndMission()
{
    p_Neutral1.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral1);
    p_Neutral2.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Neutral2);
}
}

event Artefact(int allD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    SetGoalState(enablePrototypes,goalAchieved);
    p_Neutral2.GiveAllUnitsTo(p_Player);
    EnableGoal(destroyEnemyBase,true);
    AddBriefing("translateBriefing132b");
    return true; //usuwa sie
}
}

defendMainBase3 = 3;
}

player pEnemy1;
player pEnemy2;
player pEnemy3;
player pPlayer;

unitex uMainBase1;
unitex uMainBase2;
unitex uMainBase3;

int bCheckEndMission;
state Initialize;
state ShowBriefing;
state Fighting;
state Nothing;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(destroyEnemy,"translateGoal133a");
    RegisterGoal(defendMainBase1,"translateGoal133b");
    RegisterGoal(defendMainBase2,"translateGoal133c");
    RegisterGoal(defendMainBase3,"translateGoal133d");
    EnableGoal(destroyEnemy,true);
    EnableGoal(defendMainBase1,true);
    EnableGoal(defendMainBase2,true);
    EnableGoal(defendMainBase3,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    pPlayer = GetPlayer(1);
    pEnemy1 = GetPlayer(2);
    pEnemy2 = GetPlayer(4);
    pEnemy3 = GetPlayer(5);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        pEnemy1.LoadScript("single\singleEasy");
        pEnemy2.LoadScript("single\singleEasy");
        pEnemy3.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        pEnemy1.LoadScript("single\singleMedium");
        pEnemy2.LoadScript("single\singleMedium");
        pEnemy3.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        pEnemy1.LoadScript("single\singleHard");
        pEnemy2.LoadScript("single\singleHard");
        pEnemy3.LoadScript("single\singleHard");
    }
    pEnemy1.EnableAIFeatures(aiControlOffense,false);
    pEnemy1.EnableAIFeatures(aiControlDefense,false);
    pEnemy2.EnableAIFeatures(aiControlOffense,false);
    pEnemy2.EnableAIFeatures(aiControlDefense,false);
    pEnemy3.EnableAIFeatures(aiControlOffense,false);
    pEnemy3.EnableAIFeatures(aiControlDefense,false);

    pEnemy1.SetNeutral(pEnemy2);
    pEnemy1.SetNeutral(pEnemy3);
    pEnemy2.SetNeutral(pEnemy1);
    pEnemy2.SetNeutral(pEnemy3);
    pEnemy3.SetNeutral(pEnemy1);
    pEnemy3.SetNeutral(pEnemy2);
    pPlayer.EnableAIFeatures(aiEnabled,false);
    //----- Money -----
    pPlayer.SetMoney(10000);
    pEnemy1.SetMoney(20000);
    pEnemy2.SetMoney(20000);
    pEnemy3.SetMoney(20000);
    //----- Researches -----
    pEnemy1.EnableResearch("RES_ED_WMR1",true);
    pEnemy1.EnableResearch("RES_MSR2",true);
    pEnemy1.EnableResearch("RES_ED_UHW1",true);
    pEnemy1.EnableResearch("RES_ED_BHD",true);

    pEnemy2.CopyResearches(pEnemy1);
    pEnemy3.CopyResearches(pEnemy1);
}

```

Mission133.ec

mission "translateMission133"

```

{
    consts
    {
        destroyEnemy = 0;
        defendMainBase1 = 1;
        defendMainBase2 = 2;
    }
}

```

```

pPlayer.EnableResearch("RES_UCS_WHG1",true);
pPlayer.EnableResearch("RES_UCS_UHL1",true);
pPlayer.EnableResearch("RES_UCS_BHD",true);
//----- Buildings -----
pPlayer.EnableBuilding("UCSBTE",false);
//----- Units -----
uMainBase1 = GetUnit(GetPointX(0),GetPointY(0),0);
uMainBase2 = GetUnit(GetPointX(1),GetPointY(1),0);
uMainBase3 = GetUnit(GetPointX(2),GetPointY(2),0);
//----- Artifacts -----
//----- Timers -----
SetTimer(0,1200);
//----- Variables -----
bCheckEndMission=false;
//----- Camera -----
CallCamera();
pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,150;/15 sec
}
//----- Show Briefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing133");
    return Fighting,100;
}
//----- Fighting -----
state Fighting
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;

        if(GetGoalState(defendMainBase1)!=goalFailed && !uMainBase1.IsLive())
            SetGoalState(defendMainBase1,goalFailed);

        if(GetGoalState(defendMainBase2)!=goalFailed && !uMainBase2.IsLive())
            SetGoalState(defendMainBase2,goalFailed);

        if(GetGoalState(defendMainBase3)!=goalFailed && !uMainBase3.IsLive())
            SetGoalState(defendMainBase3,goalFailed);

        if(!pEnemy1.GetNumberOfUnits() && !pEnemy1.GetNumberOfBuildings()
            && !pEnemy2.GetNumberOfUnits() && !pEnemy2.GetNumberOfBuildings()
            && !pEnemy3.GetNumberOfUnits() && !pEnemy3.GetNumberOfBuildings())
        {
            SetGoalState(destroyEnemy,goalAchieved);
        }

        if(!pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed133");
            EndMission(false);
        }
        if(GetGoalState(defendMainBase1)==goalFailed &&
            GetGoalState(defendMainBase2)==goalFailed &&
            GetGoalState(defendMainBase2)==goalFailed
        )
        {
            AddBriefing("translateFailed133");
            EnableEndMissionButton(true,false);
            return Nothing,500;
        }
    }
    if(GetGoalState(destroyEnemy)==goalAchieved)
    {
        if(GetGoalState(defendMainBase1)==goalFailed)
            SetGoalState(defendMainBase1,goalAchieved);
        if(GetGoalState(defendMainBase2)==goalFailed)
            SetGoalState(defendMainBase2,goalAchieved);
        if(GetGoalState(defendMainBase3)==goalFailed)
            SetGoalState(defendMainBase3,goalAchieved);
        AddBriefing("translateAccomplished133");
        EnableEndMissionButton(true);
        return Nothing,500;
    }
    return Fighting,200;
}
//----- Nothing -----
state Nothing
{
    return Nothing, 500;
}
//----- Timer0 -----
event Timer0()
{
    pEnemy1.RussianAttack(GetPointX(0),GetPointY(0),0);
    pEnemy2.RussianAttack(GetPointX(1),GetPointY(1));
    pEnemy3.RussianAttack(GetPointX(2),GetPointY(2));
}
//----- Unit Destroyed -----
event UnitDestroyed(unit uUnit)
{
    bCheckEndMission=true;
}
//----- Building Destroyed -----
event BuildingDestroyed(unit uUnit)
{
    bCheckEndMission=true;
}
//----- End Mission -----
event EndMission()
{
    pEnemy1.SetEnemy(pEnemy2);
    pEnemy1.SetEnemy(pEnemy3);
    pEnemy2.SetEnemy(pEnemy1);
    pEnemy2.SetEnemy(pEnemy3);
    pEnemy3.SetEnemy(pEnemy1);
    pEnemy3.SetEnemy(pEnemy2);
}
}

mission "translateMission134"
//Projekt Teleport - last stage
consts
{
    testTeleport = 0;
}

player p_Player;
player p_Security;
unitex p_Tester1;

int n_TeleportInX;
int n_TeleportInY;
int n_TeleportOutX;
int n_TeleportOutY;
int bCheckEndMission;

state Initialize;
state ShowBriefing;
state TestTeleport;
state TeleportBriefing;
state TestTeleport2;
state LastBriefing;
state Final;
//-----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(testTeleport,"translateGoal134");
    EnableGoal(testTeleport,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(2);
    tmpPlayer.EnableStatistics(false);
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Security = GetPlayer(4);
    //----- AI -----
    p_Security.EnableStatistics(false);
    p_Player.SetNeutral(p_Security);
    p_Security.SetNeutral(p_Player);
    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Security.EnableAIFeatures(aiEnabled,false);
    //----- Money -----
    p_Player.SetMoney(5000);
    //----- Researches -----
    //----- Buildings -----
    if(p_Player.GetScriptData(5)==12)

```

Mission134.ec

```

    p_Player.EnableBuilding("UCSBTE"false);
    //----- Units -----
    p_Tester1 = GetUnit(GetPointX(0),GetPointY(0),0);
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    SetTimer(1,6000);
    //----- Variables -----
    n_TeleportInX = GetPointX(1);
    n_TeleportInY = GetPointY(1);
    n_TeleportOutX = GetPointX(2);
    n_TeleportOutY = GetPointY(2);
    bCheckEndMission = false;
    //----- Camera -----
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),8,0,20,0);
{
    SelectUnit(p_Tester1,false);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),40,400,2500,800,5);
    AddBriefing("translateBriefing134a");
    return TestTeleport,20;
}
//-----
state TestTeleport
{
    if(p_Tester1.DistanceTo(n_TeleportOutX,n_TeleportOutY)<4)
    {
        p_Player.DelayedLookAt(n_TeleportOutX,n_TeleportOutY,8,0,20,0,20,1);
        return TeleportBriefing,100;
    }
    return TestTeleport,20;
}
//-----
state TeleportBriefing
{
    AddBriefing("translateBriefing134b");
    return TestTeleport2,50;
}
//-----
state TestTeleport2
{
    if(p_Tester1.DistanceTo(n_TeleportInX,n_TeleportInY)<4)
    {
        p_Player.DelayedLookAt(n_TeleportInX,n_TeleportInY,8,0,20,0,20,1);
        return LastBriefing,80;
    }
    return TestTeleport2,20;
}
//-----
state LastBriefing
{
    SetGoalState(testTeleport.goalAchieved);
    AddBriefing("translateAccomplished134");
    p_Player.SetScriptData(5,12)//to oznacz ze wynaleziono teleport
    EnableEndMissionButton(true);
    return Final,500;
}
//-----
state Final
{
    return Final,500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        AddBriefing("translateFailed134");
        EndMission(false);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{

```

```

    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    p_Player.SetEnemy(p_Security);
    p_Security.SetEnemy(p_Player);
}
}



## Mission141.ec


mission "translateMission141"
{/Madagascar
consts
{
    recoverArtifact=0;
}
player p_Enemy1;
player p_Enemy2;
player p_Player;
state Initialize;
state ShowBriefing;
state RecoverArtifact;
state Evacuate;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(recoverArtifact,"translateGoal141");
    EnableGoal(recoverArtifact,true);
    //----- Temporary players -----
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy1 = GetPlayer(2);
    p_Enemy2 = GetPlayer(3);
    //----- AI -----
    p_Player.SetMilitaryUnitsLimit(30000);
    p_Player.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }
    p_Enemy1.SetPointToAssemble(0,GetPointX(1),GetPointY(1),0);
    p_Enemy1.SetPointToAssemble(1,GetPointX(2),GetPointY(2),0);
    //----- Money -----
    p_Player.SetMoney(20000);
    p_Enemy1.SetMoney(20000);
    p_Enemy2.SetMoney(20000);
    //----- Researches -----
    p_Enemy1.EnableResearch("RES_ED_WHL1",true);
    p_Enemy1.EnableResearch("RES_MMR2",true);
    p_Enemy1.EnableResearch("RES_ED_UHT1",true);
    p_Player.EnableResearch("RES_UCS_WH1",true);
    p_Player.EnableResearch("RES_UCS_UAH2",true);
    p_Player.EnableResearch("RES_UCS_SGEN",true);
    p_Enemy2.EnableResearch("RES_LC_WHS1",true);
    p_Enemy2.EnableResearch("RES_LC_WHL1",true);
    p_Enemy2.EnableResearch("RES_LC_WMR1",true);
    p_Enemy2.EnableResearch("RES_MMR2",true);
    p_Enemy2.EnableResearch("RES_LC_UCR1",true);
    p_Enemy2.EnableResearch("RES_LC_UBO1",true);
    //----- Buildings -----
}
```

```

//----- Units -----
//----- Artefacts -----
//      name      x,y,z, nr,typ
CreateArtefact("NEASPECIAL2",GetPointX(0),GetPointY(0),1,0,artefactSpecialAI0th
en);
//----- Timers -----
SetTimer(0,200);
//----- Variables -----
//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
AddBriefing("translateBriefing141");
EnableNextMission(0,true);
return RecoverArtifact,100;
}

//-----
state RecoverArtifact
{
if(GetGoalState(recoverArtifact)==goalAchieved)
{
EnableEndMissionButton(true);
return Evacuate,500;
}
return RecoverArtifact,100;
}
//-----
state Evacuate
{
return Evacuate,500;
}
//-----
event Timer0()
{
if((p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings()))
{
AddBriefing("translateFailed141");
EndMission(false);
}
}
//-----
event Artefact(int alD,player pIPlayer)
{
if(pIPlayer!=p_Player) return false;
SetGoalState(recoverArtifact,goalAchieved);
p_Player.EnableResearch("RES_UCS_SGen",true);
AddBriefing("translateAccomplished141");
return true; //uswa sie
}
}

```

Mission142.ec

mission "translateMission142"

```

{
consts
{
protectMeeting = 0;
escortLC = 1;
attackTime = 1200;
briefingTime = 3400;
endMeetingTime = 6000;
}

player pEnemy;
player pLC;
player pLCunits;
player pPlayer;
player pUCSunits;

unitex uLC;
unitex uUCS;

int bCheckEndMission;
int nMissionStep;

```

```

state Initialize;
state ShowBriefing;
state Fighting;
state Nothing;
//-----
state Initialize
{
//----- Goals -----
RegisterGoal(protectMeeting,"translateGoal142a");
RegisterGoal(escortLC,"translateGoal142b");
EnableGoal(protectMeeting,true);

//----- Temporary players -----
//----- Players -----
pPlayer = GetPlayer(1);
pEnemy = GetPlayer(2);
pLC = GetPlayer(3);
pLCunits = GetPlayer(4);
pUCSunits = GetPlayer(6);
//----- AI -----
pLCunits.EnableStatistics(false);
pUCSunits.EnableStatistics(false);

if(GetDifficultyLevel()==0)
{
pEnemy.LoadScript("single\singleEasy");
}
if(GetDifficultyLevel()==1)
{
pEnemy.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
pEnemy.LoadScript("single\singleHard");
}
pLC.LoadScript("single\singleHard");
pLCunits.LoadScript("single\singleHard");
pUCSunits.LoadScript("single\singleHard");

pPlayer.EnableAIFeatures(aiEnabled,false);

pEnemy.EnableAIFeatures(aiControlOffense,false);
//pEnemy.EnableAIFeatures(aiControlDefense,false);
pLC.EnableAIFeatures(aiControlOffense,false);

pLCunits.EnableAIFeatures(aiEnabled,false);
pUCSunits.EnableAIFeatures(aiEnabled,false);

pLC.EnableAIFeatures(aiRejectAlliance,false);
pLCunits.EnableAIFeatures(aiRejectAlliance,false);
pUCSunits.EnableAIFeatures(aiRejectAlliance,false);

pLC.SetEnemy(pEnemy);
pLCunits.SetEnemy(pEnemy);
pUCSunits.SetEnemy(pEnemy);

pPlayer.SetAlly(pLCunits);
pPlayer.SetAlly(pUCSunits);

pUCSunits.SetAlly(pLC);
pUCSunits.SetAlly(pLCunits);

pLCunits.ChooseEnemy(pEnemy);
pUCSunits.ChooseEnemy(pEnemy);
pLC.ChooseEnemy(pEnemy);

pLCunits.SetNeutral(pEnemy);

pEnemy.SetEnemy(pLCunits);
pEnemy.SetEnemy(pLCunits);
pEnemy.SetEnemy(pLC);

//----- Money -----
pPlayer.SetMoney(10000);
pEnemy.SetMoney(30000);
pLC.SetMoney(30000);
//----- Research -----
pEnemy.EnableResearch("RES_ED_UMI1",true);

pLC.EnableResearch("RES_LC_REG1",true);
pLC.EnableResearch("RES_LC_SHR1",true);

pPlayer.EnableResearch("RES_UCS_WMR1",true);
pPlayer.EnableResearch("RES_MMR2",true);
pPlayer.EnableResearch("RES_UCS_UAH1",true);

```

```

//----- Buildings -----
//----- Units -----
uLC = GetUnit(GetPointX(0),GetPointY(0),0);
uUCS = GetUnit(GetPointX(1),GetPointY(1),0);
//----- Artifacts -----
//----- Timers -----
if(GetDifficultyLevel()==0)
    SetTimer(0,12000);
if(GetDifficultyLevel()==1)
    SetTimer(0,9000);
if(GetDifficultyLevel()==2)
    SetTimer(0,6000);

//----- Variables -----
bCheckEndMission=false;
nMissionStep=0;
//----- Camera -----
CallCamera();

pPlayer.LookAt(pPlayer.GetStartingPointX(),pPlayer.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;//15 sec
}

state ShowBriefing
{
    AddBriefing("translateBriefing142a");
    uLC.CommandMove(GetPointX(2),GetPointY(2),0);
    uUCS.CommandMove(GetPointX(2),GetPointY(2),0);
    EnableNextMission(0,true);
    return Fighting,100;
}

state Fighting
{
    if(GetMissionTime()>=attackTime && InMissionStep)
    {
        nMissionStep=1;
        pEnemy.Attack(GetPointX(2),GetPointY(2),0);
    }

    if(GetMissionTime()>=briefingTime && nMissionStep==1)
    {
        nMissionStep=2;
        AddBriefing("translateBriefing142b");//neo here -> protekt meeting
    }
    if(GetMissionTime()>=endMeetingTime && nMissionStep==2)
    {
        nMissionStep=3;
        SetGoalState(protectMeeting,goalAchieved);
        pPlayer.SetMly(pLC);
        EnableGoal(ecosrtLC,true);
        uLC.ChangePlayer(pPlayer);
        uUCS.ChangePlayer(pPlayer);
        AddBriefing("translateBriefing142c");//meeting is over -> escort LC to
the base
    }

    if(uLC.DistanceTo(GetPointX(5),GetPointY(5))<14)
    {
        uLC.ChangePlayer(pLC);
        SetGoalState(ecosrtLC,goalAchieved);
        AddBriefing("translateAccomplished142");
        EnableNextMission(0,false);
        EnableNextMission(1,true);
        EnableNextMission(2,true);
        EnableEndMissionButton(true);
        return Nothing;
    }

    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(!uLC.IsLive())
        {
            SetGoalState(ecosrtLC,goalFailed);
            pLC.SetEnemy(pPlayer);
            pPlayer.SetEnemy(pLC);
            AddBriefing("translateFailed142a",pPlayer.GetName());
            EnableEndMissionButton(true,false);
            return Nothing;
        }

        if((pPlayer.GetNumberOfUnits() && pPlayer.GetNumberOfBuildings()))
        {
            AddBriefing("translateFailed142b");
            EndMission(false);
        }
    }
}

state Nothing
{
    return Nothing, 500;
}

event Timer0()
{
    if(nMissionStep>2)
    {
        pEnemy.Attack(uLC.GetLocationX(),uLC.GetLocationY(),uLC.GetLocationZ());
    }
}

event UnitDestroyed(unit uUnit)
{
    bCheckEndMission=true;
}

event BuildingDestroyed(unit uUnit)
{
    bCheckEndMission=true;
}

event EndMission()
{
    pLC.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pLC);

    pLCunits.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pLCunits);

    pUCSunits.SetEnemy(pUCSunits);
    pUCSunits.SetEnemy(pLCunits);

    pLCunits.SetEnemy(pUCSunits);
    pUCSunits.SetEnemy(pLCunits);

    pUCSunits.SetEnemy(pPlayer);
    pPlayer.SetEnemy(pUCSunits);
}
}

```

Mission143.ec

```

mission "translateMission143"
//Madagascar
consts
{
    defendLCBase=0;
    destroyEnemyForces=1;
    recoverArtifact = 2;
}

player p_Enemy;
player p_Ally;
player p_Player;

state Initialize;
state ShowBriefing;
state Fighting;
state ShowVideoState;
state Evacuate;

state Initialize
{
    //----- Goals -----
    RegisterGoal(defendLCBase,"translateGoal143a");
    RegisterGoal(destroyEnemyForces,"translateGoal143b");
    RegisterGoal(recoverArtifact,"translateGoal143c");

    EnableGoal(defendLCBase,true);
    EnableGoal(destroyEnemyForces,true);
    //----- Temporary players -----
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);
    p_Ally = GetPlayer(3);
    //----- AI -----
    p_Player.SetMilitaryUnitsLimit(30000);
}

```

```

p_Player.EnableAIFeatures(aiEnabled,false);

if(GetDifficultyLevel()==0)
{
    p_Enemy.LoadScript("single\\singleEasy");
    p_Ally.LoadScript("single\\singleHard");
}
if(GetDifficultyLevel()==1)
{
    p_Enemy.LoadScript("single\\singleMedium");
    p_Ally.LoadScript("single\\singleMedium");
}
if(GetDifficultyLevel()==2)
{
    p_Enemy.LoadScript("single\\singleHard");
    p_Ally.LoadScript("single\\singleEasy");
}

p_Enemy.SetPointToAssemble(0,GetPointX(1),GetPointY(1),0);
p_Enemy.SetPointToAssemble(1,GetPointX(2),GetPointY(2),0);

p_Ally.EnableAIFeatures(aiRejectAlliance,false);
p_Ally.ChooseEnemy(p_Enemy);
p_Ally.SetEnemy(p_Enemy);

p_Enemy.ChooseEnemy(p_Ally);
p_Enemy.SetEnemy(p_Ally);

p_Player.SetAlly(p_Ally);
//----- Money -----
p_Player.SetMoney(20000);
p_Enemy.SetMoney(30000);
p_Ally.SetMoney(30000);
//----- Researches -----
p_Enemy.EnableResearch("RES_ED_WHCI",true);
p_Enemy.EnableResearch("RES_ED_UHT1",true);

p_Player.EnableResearch("RES_UCS_WHP1",true);
p_Player.EnableResearch("RES_UCS_UAH2",true);

p_Ally.EnableResearch("RES_LC_WHIS1",true);
p_Ally.EnableResearch("RES_LC_WHL1",true);
p_Ally.EnableResearch("RES_LC_WMR1",true);
p_Ally.EnableResearch("RES_MMRC2",true);
p_Ally.EnableResearch("RES_LC_UCR1",true);
p_Ally.EnableResearch("RES_LC_UBO1",true);
//----- Buildings -----
//----- Units -----
//----- Artifacts -----
//----- Timers -----
SetTimer(0,200);
//----- Variables -----
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing143a");
    EnableNextMission(0,true);
    return Fighting,100;
}

//-----
state Fighting
{
    if((p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed143b");
        EndMission(false);
    }
    if((p_Ally.GetNumberOfBuildings())
    {
        SetGoalState(defendLCBase,goalFailed);
        AddBriefing("translateFailed143a");
        EnableEndMissionButton(true,false);
        return Evacuate,500;
    }
    if((p_Enemy.GetNumberOfUnits() && p_Enemy.GetNumberOfBuildings())
    {
        SetGoalState(defendLCBase,goalAchieved);
    }
}

SetGoalState(destroyEnemyForces,goalAchieved);
EnableGoal(recoverArtifact,true);
AddBriefing("translateBriefing143b");

CreateArtifact("NEASPECIAL2",GetPointX(0),GetPointY(0),1,0,artefactSpecialAI0ther);
    return ShowVideoState,20;
}
}
//-----
state ShowVideoState
{
    ShowVideo("CS110");
    return Evacuate,500;
}
}
//-----
state Evacuate
{
    return Evacuate,500;
}
}
//-----
event Timer0()
{
    if((p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed143b");
        EndMission(false);
    }
}
}
//-----
event EndMission()
{
    p_Ally.EnableAIFeatures(aiRejectAlliance,true);
    p_Player.SetEnemy(p_Ally);
    p_Ally.SetEnemy(p_Player);
}
}
//-----
event Artefact(int aid,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    SetGoalState(recoverArtifact,goalAchieved);
    p_Player.EnableResearch("RES_UCS_SGen",true);
    AddBriefing("translateAccomplished143");
    EnableEndMissionButton(true);
    return true; //usuwa sie
}
}

Mission144.ec
mission "translateMission144"
{/Australia
consts
{
    destroyEDBase = 0;
}
player p_Enemy;
player p_Ally;
player p_Player;
int bNeoFirstAttack;
int bNeoSecondAttack;

state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;
}
}
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(destroyEDBase,"translateGoal144");
    EnableGoal(destroyEDBase,true);
    //----- Temporary players -----
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);
    p_Ally = GetPlayer(3);
    //----- AI -----
    p_Player.SetMilitaryUnitsLimit(30000);
    p_Player.EnableAIFeatures(aiEnabled,false);
}
}

```

```

if(GetDifficultyLevel()==0)
{
    p_Enemy.LoadScript("single\singleEasy");
    p_Ally.LoadScript("single\singleHard");
}
if(GetDifficultyLevel()==1)
{
    p_Enemy.LoadScript("single\singleMedium");
    p_Ally.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
    p_Enemy.LoadScript("single\singleHard");
    p_Ally.LoadScript("single\singleEasy");
}

p_Ally.EnableAIFeatures(aiRejectAlliance,false);
p_Player.SetAlly(p_Ally);
p_Ally.ChooseEnemy(p_Enemy);
p_Ally.SetEnemy(p_Enemy);
p_Enemy.SetEnemy(p_Ally);

----- Money -----
p_Player.SetMoney(20000);
p_Enemy.SetMoney(30000);
p_Ally.SetMoney(30000);
----- Researches -----
p_Enemy.EnableResearch("RES_ED_WSI1",true);
p_Enemy.EnableResearch("RES_MMR2",true);
p_Enemy.EnableResearch("RES_ED_UHT1",true);
p_Enemy.EnableResearch("RES_ED_UHS1",true);

p_Player.EnableResearch("RES_UCS_WAMR1",true);
p_Player.EnableResearch("RES_UCS_BOMBER21",true);

----- Buildings -----
----- Units -----
----- Artifacts -----
----- Timers -----
SetTimer(0,200);
SetTimer(1,18000); //7 min
----- Variables -----
bNeoFirstAttack=true;
bNeoSecondAttack=false;
----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
    return ShowBriefing,100;
}
-----
state ShowBriefing
{
    AddBriefing("translateBriefing144a");
    EnableNextMission(0,true);
    return Fighting,100;
}

-----
state Fighting
{
    if((p_Enemy.GetNumberOfUnits() && !p_Enemy.GetNumberOfBuildings()))
    {
        bNeoSecondAttack=false;
        bNeoFirstAttack=false;

        p_Player.SetAlly(p_Ally);
        p_Ally.EnableAIFeatures(aiControlOffense,false);

        SetGoalState(destroyEDBase,goalAchieved);
        AddBriefing("translateAccomplished144");
        EnableEndMissionButton(true);
        return Evacuate,500;
    }
    return Fighting,100;
}
-----
state Evacuate
{
    return Evacuate,500;
}
-----
event Timer0()
{
    if(p_Player.GetNumberOfUnits() && !p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed144");
        EndMission(false);
    }
}
-----
event Timer1()
{
    if(bNeoSecondAttack)
    {
        bNeoSecondAttack=false;
        // AddBriefing("translateBriefing144c");
        // p_Player.GiveAllUnitsTo(p_Enemy);
    }
    if(bNeoFirstAttack)
    {
        bNeoFirstAttack=false;
        bNeoSecondAttack=true;
        AddBriefing("translateBriefing144b");
        p_Player.GiveAllUnitsTo(p_Enemy);
    }
}
-----
event EndMission()
{
    p_Ally.EnableAIFeatures(aiRejectAlliance,true);
    p_Player.SetEnemy(p_Ally);
    p_Ally.SetEnemy(p_Player);
}

```

Mission151.ec

```

mission "translateMission151"
//Mozambik kill Neo
consts
{
    destroyEDBase = 0;
    destroyNeoHome = 1;
}
player p_Enemy1;
player p_Enemy2;
player p_Player;
unitex uNeoHome;
int bNeoFirstAttack;
int bNeoSecondAttack;
state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;
-----
state Initialize
{
    player tmpPlayer;
    ----- Goals -----
    RegisterGoal(destroyEDBase,"translateGoal151a");
    RegisterGoal(destroyNeoHome,"translateGoal151b");
    EnableGoal(destroyEDBase,true);
    EnableGoal(destroyNeoHome,true);
    ----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    ----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy1 = GetPlayer(2);
    p_Enemy2 = GetPlayer(8);
    ----- AI -----
    p_Player.SetMilitaryUnitsLimit(40000);
    p_Player.EnableAIFeatures(aiEnabled,false);
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
    }
}

```

```

    p_Enemy2.LoadScript("single\singleHard");
}
if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
}
if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleEasy");
}

//----- Money -----
p_Player.SetMoney(20000);
p_Enemy1.SetMoney(30000);
p_Enemy2.SetMoney(30000);
//----- Research -----
p_Player.EnableResearch("RES_UCS邬BL1",true);
p_Player.EnableResearch("RES_UCS_UAH3",true);
p_Player.EnableResearch("RES_UCS_SHD",true);

p_Enemy1.EnableResearch("RES_ED_WH2",true);
p_Enemy1.EnableResearch("RES_ED_UTB1",true);
p_Enemy1.EnableResearch("RES_ED_SCR",true);

p_Enemy2.CopyResearches(p_Enemy1);
//----- Buildings -----
//----- Units -----
uNeoHome=GetUnit(GetPointX(0),GetPointY(0,0));
//----- Artefacts -----
//----- Timers -----
SetTimer(0,200);
SetTimer(1,18000); //7 min
//----- Variables -----
bNeoFirstAttack=true;
bNeoSecondAttack=false;
//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6.0,20,0);
return ShowBriefing,100;
}
//----- ShowBriefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing151a");
    EnableNextMission(0,true);
    return Fighting,100;
}

//----- Fighting -----
state Fighting
{
    if((GetGoalState(destroyEDBase)==goalAchieved & GetGoalState(destroyNeoHome)== goalAchieved)
    {
        AddBriefing("translateAccomplished151c");
        EnableEndMissionButton(true);
        return Evacuate,500;
    }

    if((GetGoalState(destroyNeoHome)==goalAchieved && !uNeoHome.IsLive()))
    {
        bNeoSecondAttack=false;
        bNeoFirstAttack=false;
        SetGoalState(destroyNeoHome,goalAchieved);
        AddBriefing("translateAccomplished151b");
    }

    if((GetGoalState(destroyEDBase)!=goalAchieved && lp_Enemy1.GetNumberOfUnits() && lp_Enemy1.GetNumberOfBuildings())
    {
        lp_Enemy2.GetNumberOfUnits() && lp_Enemy2.GetNumberOfBuildings())
        {
            SetGoalState(destroyEDBase,goalAchieved);
            AddBriefing("translateAccomplished151a");
        }
    }

    return Fighting,100;
}
//----- Evacuate -----
state Evacuate
{
    return Evacuate,500;
}

//----- Timer0 -----
event Timer0()
{
    if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed151");
        EndMission(false);
    }
}
//----- Timer1 -----
event Timer1()
{
    if(uNeoHome.IsLive())
    {
        if(bNeoSecondAttack)
        {
            bNeoSecondAttack=false;
            AddBriefing("translateBriefing151c");
            p_Player.GiveAllUnitsTo(p_Enemy1);
        }
        if(!bNeoFirstAttack)
        {
            bNeoFirstAttack=false;
            bNeoSecondAttack=true;
            AddBriefing("translateBriefing151b");
            p_Player.GiveAllUnitsTo(p_Enemy1);
        }
    }
}

Mission152.ec
mission "translateMission152"
{/Australia
consts
{
    destroyEDForces = 0;
}

player p_Enemy;
player p_Player;

int bNeoFirstAttack;
int bNeoSecondAttack;

state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;

//----- Initialize -----
state Initialize
{
    player tmpPlayer;
    //----- Goals -----
    RegisterGoal(destroyEDForces,"translateGoal152");
    EnableGoal(destroyEDForces,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);
    //----- AI -----
    p_Player.SetMilitaryUnitsLimit(30000);
    p_Player.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==-1)
    {
        p_Enemy.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\singleHard");
    }

    //----- Money -----
    p_Player.SetMoney(20000);
    p_Enemy.SetMoney(40000);
    //----- Research -----
}

```

```

p_Player.EnableResearch("RES_UCS_WAMR1",true);
p_Player.EnableResearch("RES_UCS_BOMBER21",true);

p_Enemy.EnableResearch("RES_ED_AMR1",true);
p_Enemy.EnableResearch("RES_ED_MHC2",true);
p_Enemy.EnableResearch("RES_ED_UA41",true);

//----- Buildings -----
//----- Units -----
//----- Artifacts -----
//----- Timers -----
SetTimer(0,200);
SetTimer(1,18000); //7 min
//----- Variables -----
bNeoFirstAttack=true;
bNeoSecondAttack=false;
//----- Camera -----
CallCamera();}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing152a");
    EnableNextMission(0,true);
    return Fighting,100;
}

//-----
state Fighting
{
    if((p_Enemy.GetNumberOfUnits() && p_Enemy.GetNumberOfBuildings()))
    {
        bNeoSecondAttack=false;
        bNeoFirstAttack=false;
        SetGoalState(destroyEDForces,goalAchieved);
        AddBriefing("translateAccomplished152");
        EnableEndMissionButton(true);
        return Evacuate,500;
    }
    return Fighting,100;
}

//-----
state Evacuate
{
    return Evacuate,500;
}

//-----
event Timer0()
{
    if((p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings()))
    {
        AddBriefing("translateFailed152");
        EndMission(false);
    }
}

//-----
event Timer1()
{
    if(bNeoSecondAttack)
    {
        bNeoSecondAttack=false;
        AddBriefing("translateBriefing152c");
        p_Player.GiveAllUnitsTo(p_Enemy);
    }
    if(bNeoFirstAttack)
    {
        bNeoFirstAttack=false;
        bNeoSecondAttack=true;
        AddBriefing("translateBriefing152b");
        p_Player.GiveAllUnitsTo(p_Enemy);
    }
}

}

Mission153.ec
mission "translateMission153"
{/Egypt
consts
{
    destroyEDBase = 0;
    destroyCBase = 1;
}

player p_Enemy;
player p_Ally;
player p_Player;

int bNeoAttack;
int bLCBreakAlliance;
int maxPlatoons;

state Initialize;
state ShowBriefing;
state Fighting;
state Evacuate;

//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(destroyEDBase,"translateGoal153a");
    RegisterGoal(destroyCBase,"translateGoal153b");
    EnableGoal(destroyEDBase,true);
    //----- Temporary players -----
    //----- Players -----
    p_Player = GetPlayer(1);
    p_Enemy = GetPlayer(2);
    p_Ally = GetPlayer(3);
    //----- AI -----
    p_Ally.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_Ally);

    p_Ally.ChooseEnemy(p_Enemy);
    p_Enemy.ChooseEnemy(p_Ally);
    p_Ally.SetEnemy(p_Enemy);
    p_Enemy.SetEnemy(p_Ally);

    p_Player.SetMilitaryUnitsLimit(30000);
    p_Player.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\singleEasy");
        p_Ally.LoadScript("single\singleHard");
        maxPlatoons=2;
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy.LoadScript("single\singleMedium");
        p_Ally.LoadScript("single\singleMedium");
        maxPlatoons=4;
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy.LoadScript("single\singleHard");
        p_Ally.LoadScript("single\singleEasy");
        maxPlatoons=7;
    }

    //----- Money -----
    p_Player.SetMoney(20000);
    p_Enemy.SetMoney(40000);
    p_Ally.SetMoney(40000);
    //----- Researches -----
    p_Enemy.EnableResearch("RES_ED_AMR1",true);
    p_Enemy.EnableResearch("RES_ED_MHC2",true);
    p_Enemy.EnableResearch("RES_ED_UA41",true);

    //----- Buildings -----
    //----- Units -----
    //----- Artifacts -----
    //----- Timers -----
    SetTimer(0,200);
    SetTimer(1,18000); //7 min XXXMD to wlaczyc
    //SetTimer(1,1800); //0.7 min
    //----- Variables -----
    bNeoAttack=true;
    bLCBreakAlliance=false;
    //----- Camera -----
    CallCamera();}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,100;
}

```

Mission153.ec

```
mission "translateMission153"
{//Egypt
    consts
    {
        destroyEDBase = 0;
        destroyCBase = 1;
```

```

state ShowBriefing
{
    AddBriefing("translateBriefing153a");
    EnableNextMission(0,true);
    return Fighting,100;
}

//-----
state Fighting
{
    if(GetGoalState(destroyLCBase)!=goalAchieved &&
    p_Ally.GetNumberOfBuildings(<5)
    {
        SetGoalState(destroyLCBase,goalAchieved);
    }
    if(GetGoalState(destroyEDBase)!=goalAchieved &&
    p_Enemy.GetNumberOfBuildings(<5)
    {
        SetGoalState(destroyEDBase,goalAchieved);
    }
    if(GetGoalState(destroyEDBase)==goalAchieved &&
    GetGoalState(destroyLCBase)==goalAchieved)
    {
        SetGoalState(destroyEDBase,goalAchieved);
        AddBriefing("translateAccomplished153");
        EnableEndMissionButton(true);
        return Evacuate,500;
    }
    return Fighting,100;
}
//-----
state Evacuate
{
    return Evacuate,500;
}
//-----
event Timer0()
{
    if((p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed153");
        EndMission(false);
    }
}
//-----
event Timer1()
{
    if(bLCBreakAlliance)
    {
        p_Enemy.EnableAIFeatures(aiControlOffense,true);
        bLCBreakAlliance=false;
        p_Ally.SetEnemy(p_Player);
        p_Ally.ChooseEnemy(p_Player);
        p_Enemy.ChooseEnemy(p_Player);

        p_Player.SetEnemy(p_Ally);
        EnableGoal(destroyLCBase,true);
        AddBriefing("translateBriefing153c");
    }
    if(bNeoAttack)
    {
        bNeoAttack=false;
        bLCbreakAlliance=true;
        AddBriefing("translateBriefing153b");
        p_Enemy.SetNumberOfOffensiveTankPlatoons(maxPlatoons);
        p_Player.GiveAllUnitsTo(p_Enemy);

        p_Enemy.RussianAttack(p_Ally.GetStartingPointX(),p_Ally.GetStartingPointY(),0);
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        SetTimer(1,6000);
    }
}

Mission165.ec
mission "translateMission165"
{
    consts
    {
        sendToBase = 0;
    }

    player p_Enemy1;
    player p_Enemy2;

    player p_Player;
    int bShowOnAI;
    int nNeedResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;
}

state Initialize;
state ShowBriefing;
state Mining;
state Nothing;

state Initialize;
state Initialize
{
    //----- Goals -----
    p_Player = GetPlayer(1);
    fleetCost = p_Player.GetScriptData(0);
    nNeedResources = fleetCost - p_Player.GetMoneySentToOrbit();
    if(nNeedResources > 100000)
        nNeedResources=100000;

    RegisterGoal(sendToBase,"translateGoal165",nNeedResources,0);
    EnableGoal(sendToBase,true);
    //----- Temporary players -----
    //----- Players -----
    p_Enemy1 = GetPlayer(2);
    p_Enemy2 = GetPlayer(3);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiEnabled,true);
    p_Enemy2.EnableAIFeatures(aiEnabled,true);
    //----- Money -----
    p_Player.SetMoney(10000);
    p_Enemy1.SetMoney(50000);
    p_Enemy2.SetMoney(50000);
    //----- Researches -----
    p_Enemy1.EnableResearch("RES_ED_WHR1",true);
    p_Enemy1.EnableResearch("RES_ED_AB1",true);
    p_Enemy1.EnableResearch("RES_ED_MB2",true);
    p_Enemy1.EnableResearch("RES_ED_MHR2",true);
    p_Enemy1.EnableResearch("RES_ED_UA31",true);

    p_Player.EnableResearch("RES_UCS_PC",true);
    p_Player.EnableResearch("RES_UCS_WSD",true);
    p_Player.EnableResearch("RES_UCS_WAPB1",true);
    p_Player.EnableResearch("RES_UCS_MB2",true);
    p_Player.EnableResearch("RES_UCS_BOMBER31",true);

    p_Enemy2.EnableResearch("RES_LC_WARTILLERY",true);
    p_Enemy2.EnableResearch("RES_LC_UCU1",true);

    //----- Buildings -----
    //----- Units -----
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    //----- Variables -----
    bShowFailed=true;
    bCheckEndMission=false;
    //----- Camera -----
    CallCamera();

    p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6.0,20,0);
    return ShowBriefing,150;//15 sec
}
//-----

```

```

state ShowBriefing
{
    EnableNextMission(0,true);
    EnableNextMission(1,true);
    AddBriefing("translateBriefing165",nNeedResources);
    return Mining,200;
}
//-----
state Mining
{
    if(GetGoalState(sendToBase)!=goalAchieved && p_Player.GetMoneySentToBase()>=nNeedResources)
    {
        SetGoalState(sendToBase,goalAchieved);
        AddBriefing("translateAccomplished165");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    RegisterGoal(sendToBase,"translateGoal165",nNeedResources,p_Player.GetMoneySentToBase());
    if(bShowFailed)
    {
        if(ResourcesLeftInMoney()+p_Player.GetMoneySentToBase()+p_Player.GetMoney())
        (<nNeedResources)
        {
            bShowFailed=false;
            SetGoalState(sendToBase,goalFailed);
            AddBriefing("translateFailed165a");
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed165b");
            EndMission(false);
        }
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
```

Mission171.ec

mission "translateMission171"

```

{
    const
    {
        sendToBase = 0;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Player;
    int bShitchOnAI;
    int nNeedResources;
    int fleetCost;
    int bShowFailed;
```

```

int bCheckEndMission;
state Initialize;
state ShowBriefing;
state Mining;
state Nothing;
//-----
state Initialize
{
    //----- Goals -----
    p_Player = GetPlayer(1);
    fleetCost = p_Player.GetScriptData(0);
    nNeedResources = fleetCost - p_Player.GetMoneySentToOrbit();
    if(nNeedResources > 100000)
        nNeedResources=100000;

    RegisterGoal(sendToBase,"translateGoal171",nNeedResources,0);
    EnableGoal(sendToBase,true);
    //----- Temporary players -----
    //----- Players -----
    p_Enemy1 = GetPlayer(2);
    p_Enemy2 = GetPlayer(3);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiEnabled,true);
    p_Enemy2.EnableAIFeatures(aiEnabled,true);
    //----- Money -----
    p_Player.SetMoney(10000);
    p_Enemy1.SetMoney(50000);
    p_Enemy2.SetMoney(50000);
    //----- Researches -----
    //----- Buildings -----
    //----- Units -----
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    //----- Variables -----
    bShowFailed=true;
    bCheckEndMission=false;
    //----- Camera -----
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
return ShowBriefing,150;//15 sec
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing171",nNeedResources);
    return Mining,200;
}
//-----
state Mining
{
    if(GetGoalState(sendToBase)!=goalAchieved && p_Player.GetMoneySentToBase()>=nNeedResources)
    {
        SetGoalState(sendToBase,goalAchieved);
        AddBriefing("translateAccomplished171");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}
//-----
state Nothing
{
```

```

        return Nothing, 500;
    }
}

event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
}

RegisterGoal(sendToBase,"translateGoal171",nNeedResources,p_Player.GetMoney
ySentToBase());
if(bShowFailed)
{
    if((ResourcesLeftInMoney() + p_Player.GetMoneySentToBase() + p_Player.GetMoney
()) < nNeedResources)
    {
        bShowFailed=false;
        SetGoalState(sendToBase,goalFailed);
        AddBriefing("translateFailed171a");
        EnableEndMissionButton(true);
        return Nothing;
    }
}
if(bCheckEndMission)
{
    bCheckEndMission=false;
    if(!p_Player.GetNumberOfUnits() && !p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed171b");
        EndMission(false);
    }
}
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

Mission172.ec
mission "translateMission172"
{
    consts
    {
        sendToBase = 0;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Player;
    int bSwitchOnAl;
    int nNeedResources;
    int fleetCost;
    int bShowFailed;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Mining;
    state Nothing;

    //-----
    state Initialize
    {
        //----- Goals -----
        p_Player = GetPlayer(1);
        fleetCost = p_Player.GetScriptData(0);
        nNeedResources = fleetCost - p_Player.GetMoneySentToOrbit();
        if(nNeedResources > 100000)
            nNeedResources=100000;

        RegisterGoal(sendToBase,"translateGoal172",nNeedResources,0);
        EnableGoal(sendToBase,true);
        //----- Temporary players -----
        //----- Players -----
        p_Enemy1 = GetPlayer(2);
        p_Enemy2 = GetPlayer(3);
    }

    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }

    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiEnabled,true);
    p_Enemy2.EnableAIFeatures(aiEnabled,true);
    //----- Money -----
    p_Player.SetMoney(10000);
    p_Enemy1.SetMoney(5000);
    p_Enemy2.SetMoney(50000);
    //----- Researches -----
    //----- Buildings -----
    //----- Units -----
    //----- Artefacts -----
    //----- Timers -----
    SetTimer(0,100);
    //----- Variables -----
    bShowFailed=true;
    bCheckEndMission=false;
    //----- Camera -----
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,150;//15 sec
}

state ShowBriefing
{
    AddBriefing("translateBriefing172",nNeedResources);
    return Mining,200;
}

state Mining
{
    if(GetGoalState(sendToBase)==goalAchieved &&
    p_Player.GetMoneySentToBase() >= nNeedResources)
    {
        SetGoalState(sendToBase,goalAchieved);
        AddBriefing("translateAccomplished172");
        EnableEndMissionButton(true);
        return Nothing, 500;
    }
    return Mining,200;
}

state Nothing
{
    return Nothing, 500;
}

event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{

RegisterGoal(sendToBase,"translateGoal172",nNeedResources,p_Player.GetMoney
ySentToBase()));

if(bShowFailed)
{
    if((ResourcesLeftInMoney() + p_Player.GetMoneySentToBase() + p_Player.GetMoney
()) < nNeedResources)
    {
        bShowFailed=false;
        SetGoalState(sendToBase,goalFailed);
        AddBriefing("translateFailed172a");
        EnableEndMissionButton(true);
        return Nothing;
    }
}
if(bCheckEndMission)

```

```

    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed172b");
            EndMission(false);
        }
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----




## TutorialUCSmis.ec


mission "translateTutorialUCS"
{
    consts
    {
        buildPowerPlant = 0;
        buildVechProdCent = 1;
        buildRefinery = 3;
        buildTransporters = 4;
        harvestResources = 5;
        buildWeapProdCent = 6;
        buildArmy = 7;
        findEnemy = 8;
        destroyEnemy = 9;
    }

    player p_Player;
    player p_Enemy;

    int nDelayCount;
    int nBuildingCount;
    int nMoney;
    int nStartX;
    int nStartY;

    //*****
    state Initialize;
    state Start;
    state MoveCamera;
    state BuildPowerPlant;
    state BuildVechProdCent;
    state BuildRefinery;
    state BuildHarvesters;
    state HarvestResources;
    state BuildWeapProdCent;
    state BuildArmy;
    state More;
    state EndState;

    state Initialize
    {
        RegisterGoal(buildPowerPlant,"translateGoalTutorialUCS_PP");
        RegisterGoal(buildVechProdCent,"translateGoalTutorialUCS_BA");
        RegisterGoal(buildRefinery,"translateGoalTutorialUCS_RF");
        RegisterGoal(buildTransporters,"translateGoalTutorialUCS_OH");
        RegisterGoal(harvestResources,"translateGoalTutorialUCS_Mining");
        RegisterGoal(buildWeapProdCent,"translateGoalTutorialUCS_FA");
        RegisterGoal(buildArmy,"translateGoalTutorialUCS_tanks");
        RegisterGoal(findEnemy,"translateGoalTutorialUCS_findEnemy");
        RegisterGoal(destroyEnemy,"translateGoalTutorialUCS_destroyEnemy");

        p_Player=GetPlayer(1);
        p_Enemy=GetPlayer(2);

        p_Player.SetMoney(20000);
        p_Enemy.SetMoney(20000);

        p_Player.EnableAIFeatures(aiEnabled,false);
        p_Enemy.EnableAIFeatures(aiEnabled,true);
        p_Enemy.EnableAIFeatures(aiControlOffense,false);

        //weapons
    }
}

//-----
p_Player.EnableResearch("RES_UCS_WACH2",false);
p_Player.EnableResearch("RES_UCS_WSR2",false);
p_Player.EnableResearch("RES_UCS_WASR1",false);
p_Player.EnableResearch("RES_UCS_WSP2",false);
p_Player.EnableResearch("RES_UCS_WHG1",false);
p_Player.EnableResearch("RES_UCS_WSD",false);
//ammo
p_Player.EnableResearch("RES_UCS_WAPB1",false);
p_Player.EnableResearch("RES_MMR2",false);
p_Player.EnableResearch("RES_UCS_MB2",false);
p_Player.EnableResearch("RES_UCS_MG2",false);
//chassis
p_Player.EnableResearch("RES_UCS_GARG1",false);
p_Player.EnableResearch("RES_UCS_USL3",false);
p_Player.EnableResearch("RES_UCS_USL2",false);
p_Player.EnableResearch("RES_UCS_UML3",false);
p_Player.EnableResearch("RES_UCS_UHL1",false);
p_Player.EnableResearch("RES_UCS_UH1",false);
p_Player.EnableResearch("RES_UCS_US1",false);
p_Player.EnableResearch("RES_UCS_USB1",false);
p_Player.EnableResearch("RES_UCS_USM1",false);
p_Player.EnableResearch("RES_UCS_UAH1",false);
p_Player.EnableResearch("RES_UCS_BOMBER21",false);
//sPECIAL
p_Player.EnableResearch("RES_UCS_BMD",false);
p_Player.EnableResearch("RES_UCS_BHD",false);

p_Player.EnableResearch("RES_UCS_RepHand2",false);
p_Player.EnableResearch("RES_UCS_SGen",false);
p_Player.EnableResearch("RES_UCS_SHD",false);

p_Player.EnableResearch("RES_UCS_WSD",false);
p_Player.EnableResearch("RES_UCS_PC",false);

// 1st tab
p_Player.EnableBuilding("UCSBPP",false);
p_Player.EnableBuilding("UCSBET",false);
p_Player.EnableBuilding("UCSBAA",false);
p_Player.EnableBuilding("UCSBAF",false);
p_Player.EnableBuilding("UCSBWB",false);
p_Player.EnableBuilding("UCSBAF",false);
// 2nd tab
p_Player.EnableBuilding("UCSBPF",false);
p_Player.EnableBuilding("UCSBTB",false);
// 3rd tab
p_Player.EnableBuilding("UCSBST",false);
// 4th tab
p_Player.EnableBuilding("UCSBCR",false);
p_Player.EnableBuilding("UCSBTE",false);
p_Player.EnableBuilding("UCSBHQ",false);
p_Player.EnableBuilding("UCSBEN",false);
p_Player.EnableBuilding("UCSBLZ",false);

nStartX = p_Player.GetStartingPointX();
nStartY = p_Player.GetStartingPointY();
LookAt(nStartX,nStartY,6,0,20,0);
SetTimer(0,6000); //5min
SetTimer(1,100); //5sec
nBuildingCount=0;

return Start;
}

//-----
state Start
{
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,2500,800,10);
    AddBriefing("translateTutorialUCS_moveCamera");
    return MoveCamera,600;
}

//-----

state MoveCamera
{
    EnableGoal(buildPowerPlant,true);
    p_Player.EnableBuilding("UCSBPP",true);
    AddBriefing("translateTutorialUCS_PP");
    return BuildPowerPlant,100;
}
//-----
```

```

state BuildPowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingPowerPlant))
    {
        p_Player.EnableBuilding("UCSBB4",true);
        p_Player.EnableBuilding("UCSBET",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildPowerPlant,goalAchieved);
        EnableGoal(buildVechProdCent,true);
        if(p_Player.GetNumberOfBuildings(buildingBase))
            AddBriefing("translateTutorialUCS_BA");
        return BuildVechProdCent,100;
    }
    if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
    {
        p_Player.CreateUnitEx(nStartX,nStartY,
0,null,"UCSUBU1",null,null,null,null);
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildPowerPlant,50;
}
//-----

state BuildVechProdCent
{
    if(p_Player.GetNumberOfBuildings(buildingBase))
    {
        p_Player.EnableBuilding("UCSBRF",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildVechProdCent,goalAchieved);
        EnableGoal(buildRefinery,true);
        if(p_Player.GetNumberOfBuildings(buildingRefinery))
            AddBriefing("translateTutorialUCS_RF");
        return BuildRefinery,100;
    }
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialUCS_BA_Fool");
    }
    if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
    {
        p_Player.CreateUnitEx(nStartX,nStartY,
0,null,"UCSUBU1",null,null,null,null);
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildVechProdCent,50;
}
//-----

state BuildRefinery
{
    if(p_Player.GetNumberOfBuildings(buildingRefinery))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildRefinery,goalAchieved);
        EnableGoal(buildTransporters,true);
        if(p_Player.GetNumberOfUnits(chassisTank | unitCarrier)<2)
            AddBriefing("translateTutorialUCS_OH");
        return BuildHarvesters,100;
    }
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialUCS_RF_Fool");
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildRefinery,50;
}
//-----

state BuildHarvesters
{
    if(p_Player.GetNumberOfUnits(unitHarvester | chassisAny)>=2)
    {
        SetGoalState(buildTransporters,goalAchieved);
        EnableGoal(harvestResources,true);
        AddBriefing("translateTutorialUCS_Mining");
        nMoney=p_Player.GetMoney();
        return HarvestResources,100;
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildHarvesters,100;
}
//-----


state HarvestResources
{
    if(nMoney < p_Player.GetMoney())
    {
        p_Player.EnableBuilding("UCSBA4",true);
        SetGoalState(harvestResources,goalAchieved);
        EnableGoal(buildWeapProdCent,true);
        if(p_Player.GetNumberOfBuildings(buildingFactory))
            AddBriefing("translateTutorialUCS_FA");
        return BuildWeapProdCent,100;
    }
    return HarvestResources,50;
}
//-----


state BuildWeapProdCent
{
    if(p_Player.GetNumberOfBuildings(buildingFactory))
    {
        SetGoalState(buildWeapProdCent,goalAchieved);
        EnableGoal(buildArmy,true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        AddBriefing("translateTutorialUCS_tanks");
        return BuildArmy;
    }
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
    }
    return BuildWeapProdCent,100;
}
//-----


state BuildArmy
{
    if(p_Player.GetNumberOfUnits(chassisTank | unitArmed)>=5)
    {
        SetGoalState(buildArmy,goalAchieved);
        EnableGoal(findEnemy,true);
        if(GetGoalState(findEnemy)==goalAchieved)
            AddBriefing("translateTutorialUCS_findEnemy");
        return More,600;
    }
    return BuildArmy,200;
}
//-----


state More
{
    p_Player.EnableBuilding("UCSBET",true);
    p_Player.EnableBuilding("UCSAB1",true);
    p_Player.EnableBuilding("UCSBS1",true);
    p_Player.EnableBuilding("UCSBC1",true);
    p_Player.EnableBuilding("UCSBEN1",true);
    AddBriefing("translateTutorialUCS_more");
    return EndState,100;
}
//-----


state EndState
{
    return EndState,500;
}
//-----


event Timer0()
{
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,2500,800,
10);
}
//-----


event Timer1()
{
    if(GetGoalState(findEnemy)!=goalAchieved &&
p_Player.IsPointLocated(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(
),0))
    {
        //LookAt(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),6,0,20,0,0);
        SetGoalState(findEnemy,goalAchieved);
        EnableGoal(destroyEnemy,true);
        AddBriefing("translateTutorialUCS_destroyEnemy");
    }
}
//-----



```

```

if(GetGoalState(destroyEnemy))!=goalAchieved &&
lp_Enemy.GetNumberOfUnits())
{
    SetGoalState(destroyEnemy.goalAchieved);
    AddBriefing("translateTutorialUCS_Victory");
}
}



### UCSbaseEDscript.ec


mission "translateMissionUCSBaseED"
{
player p_Player;
state Initialize;
state Nothing;
state Initialize
{
    p_Player = GetPlayer(2);
    p_Player.SetMoney(25000);

    p_Player.EnableAIFeatures(aiBuildTanks,false);
    p_Player.EnableAIFeatures(aiBuildShips,false);
    p_Player.EnableAIFeatures(aiBuildHelicopters,false);

    p_Player.EnableAIFeatures(aiBuildSpecialUnits,false);

//weapons
    p_Player.EnableResearch("RES_ED_WCH2",false);
    p_Player.EnableResearch("RES_ED_WCA2",false);
    p_Player.EnableResearch("RES_ED_WSL1",false);
    p_Player.EnableResearch("RES_ED_WSR1",false);
    p_Player.EnableResearch("RES_ED_WSH1",false);
    p_Player.EnableResearch("RES_ED_WMR1",false);
    p_Player.EnableResearch("RES_ED_MR1",false);
    p_Player.EnableResearch("RES_ED_WH1",false);
    p_Player.EnableResearch("RES_ED_WH2",false);
    p_Player.EnableResearch("RES_ED_WHL1",false);
    p_Player.EnableResearch("RES_ED_WHRL1",false);

//ammo
    p_Player.EnableResearch("RES_MCH2",false);
    p_Player.EnableResearch("RES_ED_MSC2",false);
    p_Player.EnableResearch("RES_ED_MHC2",false);
    p_Player.EnableResearch("RES_MS2",false);
    p_Player.EnableResearch("RES_MM2",false);
    p_Player.EnableResearch("RES_ED_MHR2",false);
    p_Player.EnableResearch("RES_ED_MB2",false);

//chassis
    p_Player.EnableResearch("RES_ED_UMT1",false);
    p_Player.EnableResearch("RES_ED_UMW1",false);
    p_Player.EnableResearch("RES_ED_UML1",false);
    p_Player.EnableResearch("RES_ED_UHT1",false);
    p_Player.EnableResearch("RES_ED_UHW1",false);
    p_Player.EnableResearch("RES_ED_UTB1",false);
    p_Player.EnableResearch("RES_ED_UA1",false);
    p_Player.EnableResearch("RES_ED_UA12",false);
    p_Player.EnableResearch("RES_ED_UA21",false);
    p_Player.EnableResearch("RES_ED_UA31",false);
    p_Player.EnableResearch("RES_ED_UA41",false);

//special
    p_Player.EnableResearch("RES_ED_RepHand",false);
    p_Player.EnableResearch("RES_ED_RepHand2",false);
    p_Player.EnableResearch("RES_ED_SCR",false);
    p_Player.EnableResearch("RES_ED_SGen",false);
    p_Player.EnableResearch("RES_ED_BMD",false);
    p_Player.EnableResearch("RES_ED_BHD",false);
    return Nothing,100;
}

//-----
state Nothing
{
    return Nothing,600;
}
}

```

UCSbaseLCscript.ec
mission "translateMissionUCSBaseLC"

```

player p_PlayerLC;
state Initialize;
state Nothing;
state Initialize
{
    p_PlayerLC=GetPlayer(3);
    p_PlayerLC.SetMoney(10000);

    p_PlayerLC.EnableAIFeatures(aiBuildTanks,false);
    p_PlayerLC.EnableAIFeatures(aiBuildShips,false);
    p_PlayerLC.EnableAIFeatures(aiBuildHelicopters,false);
    p_PlayerLC.EnableAIFeatures(aiBuildSpecialUnits,false);

//weapons
    p_PlayerLC.EnableResearch("RES_LC_WMR1",false);
    p_PlayerLC.EnableResearch("RES_LC_WHL1",false);
    p_PlayerLC.EnableResearch("RES_LC_WHS1",false);
    p_PlayerLC.EnableResearch("RES_LC_WARTILLERY",false);
//CHASSIS
    p_PlayerLC.EnableResearch("RES_LC_UCR1",false);
    p_PlayerLC.EnableResearch("RES_LC_UCU1",false);
    p_PlayerLC.EnableResearch("RES_LC_UBO1",false);
//SPECIAL
    p_PlayerLC.EnableResearch("RES_LC_SHR1",false);
    p_PlayerLC.EnableResearch("RES_LC_REG1",false);
//AMMO
    p_PlayerLC.EnableResearch("RES_MMR2",false);
    return Nothing;

}
state Nothing
{
}
}



### UCSbaseUCSscript.ec


mission "translateMissionUCSBaseUCS"
{
const
{
    oneSeasonMoney = 50000;//CR
    oneSeasonTime = 150000://clk = 2h 5min tak naprawde to odpowida to
polowce sezony
    noOfSeasons = 20;
    fleetCost = 1000000;
    firstPhaseCost = 100000;
    phaseCost = 200000;
    clicksPerDay = 16383;
}
player p_Player;
int nReport;
int nMissionsFinished;
int EndGameResult;

state Initialize;
state ShowBriefing;
state ShowStartFirstMissionBriefing;
state Nothing;
state EndGameState;

state Initialize
{
    //----- Goals -----
    RegisterGoal("0",translateGoalUCSSendMoneyToSpace",fleetCost,0);
    RegisterGoal("1",translateGoalUCSCampaignPhase1");
    RegisterGoal("2",translateGoalUCSCampaignPhase2");
    RegisterGoal("3",translateGoalUCSCampaignPhase3");
    RegisterGoal("4",translateGoalUCSCampaignPhase4");
    RegisterGoal("5",translateGoalUCSCampaignPhase5);
    EnableGoal(0,true);
    EnableGoal(1,true);
    EnableGoal(2,true);
    EnableGoal(3,true);
    EnableGoal(4,true);
    EnableGoal(5,true);
    //----- Temporary players -----
    //----- Players -----
    p_Player = GetPlayer(1);
    //----- AI -----
    p_Player.SetScriptData(0,fleetCost);
    p_Player.SetMilitaryUnitsLimit(10000);
    //----- Money -----
    p_Player.SetMoney(5000);
    //----- Researches -----
}
```

```

p_Player.EnableResearch("RES_UCS_WCH2",false);
p_Player.EnableResearch("RES_UCS_WACH2",false);
p_Player.EnableResearch("RES_UCS_WSG2",false);
p_Player.EnableResearch("RES_UCS_WHG1",false);
p_Player.EnableResearch("RES_UCS_WSR1",false);
p_Player.EnableResearch("RES_UCS_WASR1",false);
p_Player.EnableResearch("RES_UCS_WMR1",false);
p_Player.EnableResearch("RES_UCS_WAMP1",false);
p_Player.EnableResearch("RES_UCS_WHP1",false);
p_Player.EnableResearch("RES_UCS_WAPB1",false);
p_Player.EnableResearch("RES_UCS_WSD",false);
p_Player.EnableResearch("RES_UCS_PC",false);

p_Player.EnableResearch("RES_MCH2",false);
p_Player.EnableResearch("RES_MSR2",false);
p_Player.EnableResearch("RES_MMRC2",false);
p_Player.EnableResearch("RES_UCS_MB2",false);
p_Player.EnableResearch("RES_UCS_MG2",false);

p_Player.EnableResearch("RES_UCS_USL2",false);
p_Player.EnableResearch("RES_UCS_USL3",false);
p_Player.EnableResearch("RES_UCS_UML1",false);
p_Player.EnableResearch("RES_UCS_UHL1",false);
p_Player.EnableResearch("RES_UCSUBL1",false);
p_Player.EnableResearch("RES_UCSUML1",false);
p_Player.EnableResearch("RES_UCSUH1",false);
p_Player.EnableResearch("RES_UCSUH2",false);
p_Player.EnableResearch("RES_UCSUH3",false);
p_Player.EnableResearch("RES_UCSUH2",false);
p_Player.EnableResearch("RES_UCSUH3",false);
p_Player.EnableResearch("RES_UCS_UBS1",false);
p_Player.EnableResearch("RES_UCS_UAH1",false);
p_Player.EnableResearch("RES_UCS_UAH2",false);
p_Player.EnableResearch("RES_UCS_UAH3",false);
p_Player.EnableResearch("RES_UCS_GARG1",false);
p_Player.EnableResearch("RES_UCS_BOMBER21",false);
p_Player.EnableResearch("RES_UCS_BOMBER31",false);

p_Player.EnableResearch("RES_UCS_BMD",false);
p_Player.EnableResearch("RES_UCS_BHD",false);
p_Player.EnableResearch("RES_UCS_Rephand",false);
p_Player.EnableResearch("RES_UCS_Rephand2",false);
p_Player.EnableResearch("RES_UCS_SGEN",false);
p_Player.EnableResearch("RES_UCS_SHD",false);

//----- Buildings -----
p_Player.EnableBuilding("UCSBEN1",false);
p_Player.EnableBuilding("UCSBTE",false);
//----- Units -----
//----- Artefacts -----
//----- Timers -----
SetTimer(0,oneSeasonTime);
//----- Variables -----
nReport = 0;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
    return ShowBriefing,100;
}
//----- state ShowBriefing -----
{
    nReport = 0;
    AddBriefing("translateStartCampaignUCS",fleetCost,p_Player.GetName());
}

Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,8000,800
,5);
    return ShowStarFirstMissionBriefing,140;
}
//----- state ShowStarFirstMissionBriefing -----
{
    AddBriefing("translateStartFirstMission");
    return Nothing,500;
}
//----- state Nothing -----
{
    int nSendedMoney;
    int nRemTime;
    int i;
    nSendedMoney = p_Player.GetMoneySentToOrbit();
}

RegisterGoal(0,"translateGoalSendMoneyToSpace",fleetCost,(nSendedMoney*100)

```

```

/fleetCost);
    RegisterGoal(1,"translateGoalUCSCampaignPhase1");
    RegisterGoal(2,"translateGoalUCSCampaignPhase2");
    RegisterGoal(3,"translateGoalUCSCampaignPhase3");
    RegisterGoal(4,"translateGoalUCSCampaignPhase4");
    RegisterGoal(5,"translateGoalUCSCampaignPhase5");

    for(i=1;i<6;i+=1)
    {
        if(GetGoalState(i)!=goalAchieved && (nSendedMoney>=((phaseCost*(i-
1))+firstPhaseCost)))
        {
            SetGoalState(i,goalAchieved);
        }
    }

    if(bMissionsFinished &&
(nSendedMoney+p_Player.GetMoney())+p_Player.GetMoneyFromFinishedMissions(
))<fleetCost)
    {
        AddBriefing("translateCampaignUCSFailed");
        EndGameResult=5;
        return EndGameState,3;
    }

    if(lp_Player.GetNumberOfBuildings(buildingSpaceStation))
    {
        AddBriefing("translateCampaignUCSSpacePortDestroyed",p_Player.GetName());
        EndGameResult=5;
        return EndGameState,3;
    }

    if(nSendedMoney>=fleetCost)
    {
        SetGoalState(0,goalAchieved);
        AddBriefing("translateCampaignUCSAccomplished",p_Player.GetName());
        EndGameResult=4;
        return EndGameState,3;
    }
}

```

```

SetConsoleText("translateCampaignUCSRemainingTime",((oneSeasonTime*nOfSea-
sons) - GetMissionTime())//clicksPerDay)//XXXMD
return Nothing,600;
}
//-----
state EndGameState
{
    EnableNextMission(0,EndGameState);
    return EndGameState,600;
}

```

```

//-----
event Timer0() //wolany co 200 000 cykli< ustawione funkcja SetTimer w state
Initialize
{
    int nSendedMoney;
    int nProjectStatus;
    int nMoneyForNextSeason;

    nReport = nReport+1;
}

```

```

if(nReport>noOfSeasons)
{
    AddBriefing("translateCampaignUCSFailed",p_Player.GetName());
    EnableNextMission(0,5);
    return;
}

nSendedMoney = p_Player.GetMoneySentToOrbit();// to ma tu byc
nProjectStatus = ((nSendedMoney*100)/fleetCost);
nMoneyForNextSeason = ((nReport+1)*oneSeasonMoney) -
nSendedMoney;
if(nMoneyForNextSeason<10000) nMoneyForNextSeason=10000;
if(nSendedMoney < oneSeasonMoney*nReport)

```

```

AddBriefing("translateUCSDelayReport",p_Player.GetName(),nReport,nProjectStatu-
s,nMoneyForNextSeason,((nSendedMoney*100)/(oneSeasonMoney*nReport)));
else

```

```

AddBriefing("translateUCSReport",p_Player.GetName(),nReport,nProjectStatus,nM-
oneyForNextSeason);
}

event CustomEvent0(int k1,int k2,int k3,int k4) //XXXMD

```

```

    {
        bMissionsFinished=true;
    }
}

```

GameTypes

Arena.ec

```
mission "translateGameTypeArena"
```

```

{
    int bDemo;
    int nTimeLimit;

    int bCheckBuilding;
    int nMeteor;
    int nPlayerInConsole;
    enum comboTechLevel
    {
        "translateGameMenuTechLevelLow",
        "translateGameMenuTechLevelMedium",
        "translateGameMenuTechLevelHigh",
        "translateGameMenuTechLevel"
    }
    enum comboStartingUnits
    {
        "1",
        "2",
        "3",
        multi: "translateGameMenuStartingUnits"
    }
    enum comboTime
    {
        "translateGameMenuTimeLimitNoLimit",
        "translateGameMenuTimeLimit15min",
        "translateGameMenuTimeLimit30min",
        "translateGameMenuTimeLimit45min",
        "translateGameMenuTimeLimit1h",
        "translateGameMenuTimeLimit15h",
        multi: "translateGameMenuTimeLimit"
    }
}

//***** F U N C T I O N S *****
function int CreateArtefacts(player rPlayer)
{
    int i;
    int x;
    int y;
    if(!rPlayer) return false;
    x = rPlayer.GetStartingPointX();
    y = rPlayer.GetStartingPointY();

    CreateArtifact("NEAMAMMO",x, y ,0,-1,-1);
    CreateArtifact("NEAMAMMO",x-1,y ,0,-1,-1);
    CreateArtifact("NEAMAMMO",x+1,y ,0,-1,-1);

    CreateArtifact("NEAENERGY",x, y ,1,0,-1,-1);
    CreateArtifact("NEAENERGY",x-1,y +1,0,-1,-1);
    CreateArtifact("NEAENERGY",x+1,y +1,0,-1,-1);

    CreateArtifact("NEAMAXHP",x, y -1,0,-1,-1);
    CreateArtifact("NEAMAXHP",x-1,y -1,0,-1,-1);
    CreateArtifact("NEAMAXHP",x+1,y -1,0,-1,-1);

    CreateArtifact("NEASHOWMAP0",x, y+2,0,-1,-1);
    return true;
}

function int CreateStartingUnits(player rPlayer,player rPlayer2)
{
    int x;
    int y;
    if(!rPlayer) return false;
    if(!rPlayer2) return false;
    x = rPlayer2.GetStartingPointX();
    y = rPlayer2.GetStartingPointY();

    if(comboTechLevel==0)
    {
        if (rPlayer.GetRace() == raceED)
        {
            rPlayer.CreateUnitEx(x,y ,0,null,"EDUMT3","EDWSL1",null,null,null,0);
            if(comboStartingUnits>0)
                rPlayer.CreateUnitEx(x+1,y ,
                    0,null,"EDUST3","EDWCA2",null,null,null);
            if(comboStartingUnits>1)
                rPlayer.CreateUnitEx(x-1,y ,
                    0,null,"EDUMW3","EDWSR3",null,null,null,0);
            if (rPlayer.GetRace() == raceLC)
            {
                rPlayer.CreateUnitEx(x,y ,0,null,"LCUMO2","LCWSL1",null,null,null,0);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,
                        y,0,null,"LCUMO3","LCWSR3",null,null,null);
                if(comboStartingUnits>1)
                    rPlayer.CreateUnitEx(x-1,
                        y,0,null,"LCUMO2","LCWS2",null,null,null,0);
            }
            if (rPlayer.GetRace() == raceUCS)
            {
                rPlayer.CreateUnitEx(x,y,0,null,"UCSML3","UCSWSP2",null,null,null,2);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,y ,0,null,"UCSUSL3","UCSWTR3",null,null,null);
                if(comboStartingUnits>1)
                    rPlayer.CreateUnitEx(x-1,y ,
                        0,null,"UCSML3","UCSWMR1",null,null,null,2);
            }
        }
        if(comboTechLevel==1)
        {
            if (rPlayer.GetRace() == raceED)
            {
                rPlayer.CreateUnitEx(x,y ,0,null,"EDUHT3","EDWMR3",null,null,null,1);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,y ,
                        0,null,"EDUHW2","EDWSL2",null,2);
                if(comboStartingUnits>1)
                    rPlayer.CreateUnitEx(x-1,y ,
                        0,null,"EDUHT3","EDWHC2",null,"EDWSR3",null,1);
            }
            if (rPlayer.GetRace() == raceLC)
            {
                rPlayer.CreateUnitEx(x,y ,0,null,"LCUCR1","LCWMR3",null,null,null,2);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,
                        y,0,null,"LCUCR1","LCWHL1",null,"LCWSL1",null,2);
                if(comboStartingUnits>1)
                    rPlayer.CreateUnitEx(x-1,y ,0,null,"LCUCR1","LCWHS2",null,null,null,2);
            }
            if (rPlayer.GetRace() == raceUCS)
            {
                rPlayer.CreateUnitEx(x,y ,0,null,"UCSUHL3","UCSWBMR3",null,"UCSWSP2",null,2);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,y ,0,null,"UCSUHL3","UCSWBSR3",null,"UCSWSSR3",null,2);
            }
        }
        if(comboTechLevel==2)
        {
            if (rPlayer.GetRace() == raceED)
            {
                rPlayer.CreateUnitEx(x,y ,0,null,"EDUHW2","EDWHL3",null,"EDWSL3",null,2);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,y ,
                        0,null,"EDUBT2","EDWMR3","EDWMR3",null,null,1);
                if(comboStartingUnits>1)
                    rPlayer.CreateUnitEx(x-1,y ,
                        0,null,"EDUHT3","EDWHC2",null,"EDWSR3",null,1);
            }
            if (rPlayer.GetRace() == raceLC)
            {
                rPlayer.CreateUnitEx(x,y,0,null,"LCUCR3","LCWHL2",null,"LCWSL2",null,2);
                if(comboStartingUnits>0)
                    rPlayer.CreateUnitEx(x+1,y ,
                        0,null,"LCUCU3","LCWMR3",null,null,1);
            }
        }
    }
}
```

```

if(comboStartingUnits>1)
    rPlayer.CreateUnitEx(x,y,0,null,"UCU3","LWHL2","LCWHL2","LCWSS2","LCWSS2",1);
}
if (rPlayer.GetRace() == raceUCS)
{
}

rPlayer.CreateUnitEx(x,y,0,null,"UCSUBL2","UCSWBHP3","UCSWSSP2","UCSWSSP2",n,
all,2);
if(comboStartingUnits>0)
    rPlayer.CreateUnitEx(x+1,y,
0,null,"UCSUBL2","UCSWBMR3","UCSWSSR3",null,null,2);
if(comboStartingUnits>1)
    rPlayer.CreateUnitEx(x-
1,y,0,null,"UCSUBL2","UCSWBHP3","UCSWSSP2","UCSWSSP2",null,2);
}
rPlayer.SetScriptData(1,rPlayer.GetNumberOfUnits());
return true;
}

-----
```

```

state Initialize;
state Nothing;

state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

    if(comboTime==0)nTimeLimit=0;
    if(comboTime==1)nTimeLimit=15*60*20;
    if(comboTime==2)nTimeLimit=30*60*20;
    if(comboTime==3)nTimeLimit=45*60*20;
    if(comboTime==4)nTimeLimit=60*60*20;
    if(comboTime==5)nTimeLimit=90*60*20;

    bCheckBuilding=false;
    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer==null)
        {
            rPlayer.SetAllowGiveUnits(false);
            rPlayer.EnableAIFeatures2(ai2ZNSendResult,false); //nie wysylac rezul-
tatow do EARTH NETU
        }
        rPlayer.AddResearch("RES_MSR2");
        rPlayer.AddResearch("RES_MSR3");
        rPlayer.AddResearch("RES_MSR4");
        rPlayer.AddResearch("RES_MM2");
        rPlayer.AddResearch("RES_MM3");
        rPlayer.AddResearch("RES_MM4");
        if (rPlayer.GetRace() == raceUCS)
        {
            rPlayer.AddResearch("RES_UCS_SGen");
            rPlayer.AddResearch("RES_UCS_MGen");
            rPlayer.AddResearch("RES_UCS_HGen");
        }
        if (rPlayer.GetRace() == raceED)
        {
            rPlayer.AddResearch("RES_ED_SGen");
            rPlayer.AddResearch("RES_ED_MGen");
            rPlayer.AddResearch("RES_ED_HGen");
        }
        if (rPlayer.GetRace() == raceED)
        {
            rPlayer.AddResearch("RES_LC_SGen");
            rPlayer.AddResearch("RES_LC_MGen");
            rPlayer.AddResearch("RES_LC_HGen");
        }
        if (rPlayer.GetRace() == raceLC)
        {
            rPlayer.AddResearch("RES_LC_SGen");
            rPlayer.AddResearch("RES_LC_MGen");
            rPlayer.AddResearch("RES_LC_HGen");
        }
    }

    rPlayer.setMaxTankPlatoonSize(1);
    rPlayer.setNumberOfOffensiveTankPlatoons(4);
    rPlayer.setNumberOfDefensiveTankPlatoons(0);
    rPlayer.setNumberOfDefensiveShipPlatoons(0);
    rPlayer.setNumberOfDefensiveHelicopterPlatoons(0);
    rPlayer.EnableAIFeatures(aiRush,true);
    rPlayer.EnableAIFeatures(aiBuildBuildings,false);
    rPlayer.EnableAIFeatures2(ai2NeutralAI,false); //kazdy na kazdego
    rPlayer.setMaxAttackFrequency(20);

    rPlayer.setMoney(0);

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);
CreateStartingUnits(rPlayer,rPlayer);
```

```

CreateArtefacts(rPlayer);
rPlayer.setScriptData(0,0);
}

SetTimer(0,23); //respawn
SetTimer(1,1200);
SetTimer(2,53); //console

nMeteor=1;
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    true;
}

event RemoveUnits()
{
    true;
}

event Timer2()
{
    int i;
    int k;
    player rPlayer;
    player rPlayer2;

    rPlayer=null;
    for(i=nPlayerInConsole+1;i<15 && rPlayer==null;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)nPlayerInConsole=i;
    }
    if(rPlayer==null)
    {
        nPlayerInConsole=0;
        for(i=0;i<15 && rPlayer==null;i=i+1)
        {
            rPlayer=GetPlayer(i);
            if(rPlayer!=null)nPlayerInConsole=i;
        }
    }
}

if(rPlayer!=null)
{
    k=1;
    for(i=0;i<15;i=i+1)
    {
        rPlayer2=GetPlayer(i);
        if(rPlayer2!=null && (rPlayer.getScriptData(0) <
rPlayer2.getScriptData(0))) k = k+1;
    }
}

SetConsoleTextCol(rPlayer.getIFFNumber(),"translateArenaStatistics",k,rPlayer.getN
ame(),rPlayer.getScriptData(0)); //XXX dodac kolor
}

event Timer0()
{
    int i;
    int nPlayersCount;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    nPlayersCount=0;
    for(i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.isAlive())
        {
            nPlayersCount=nPlayersCount+1;
            rLastPlayer=rPlayer;
            if(rPlayer.getScriptData(1))
            {
                rPlayer2 = GetPlayer(nPlayerInConsole);
```



```

if(comboTime==0)nTimeLimit=0;
if(comboTime==1)nTimeLimit=15*60*20;
if(comboTime==2)nTimeLimit=30*60*20;
if(comboTime==3)nTimeLimit=45*60*20;
if(comboTime==4)nTimeLimit=60*60*20;
if(comboTime==5)nTimeLimit=90*60*20;

if(comboResources==0) ResourcesPerContainer(16);
if(comboResources==1) ResourcesPerContainer(8);
if(comboResources==2) ResourcesPerContainer(4);
if(comboResources==3) ResourcesPerContainer(2);

bCheckBuilding=false;
for(i=0;i<15;i+=1)
{
    rPlayer=GetPlayer(i);

    if(rPlayer==null)
    {
        if(comboAlliedVictory)
            rPlayer.EnableAIFeatures2(ai2BNSSendResult,false); //nie wysylac
    }
}

rezultatow do EARTH NETU

if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

rPlayer.SetMoney(nStartingMoney);

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);
if((rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings()))

rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX()+1,rPlayer.GetStartingPointY()
,0);

if(rPlayer.GetRace()==1)

CreateArtefact("NEABANNERUCS",rPlayer.GetStartingPointX(),rPlayer.GetStartingPo
intY(),0,i,artefactSpecialAIOther,j);
if((rPlayer.GetRace()==2)

CreateArtefact("NEABANNEDR",rPlayer.GetStartingPointX(),rPlayer.GetStartingPoi
ntY(),0,i,artefactSpecialAIOther,j);
if((rPlayer.GetRace()==3)

CreateArtefact("NEABANNERLC",rPlayer.GetStartingPointX(),rPlayer.GetStartingPoi
ntY(),0,i,artefactSpecialAIOther,j);

    }

SetTimer(0,300);
SetTimer(1,1200);

nMeteor=1;
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    false;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

event Artefact(int alD,player piPlayer)
{
    player rPlayer;
    rPlayer=GetPlayer(alD);
    if(rPlayer==null) return false;
    if(rPlayer == piPlayer) return false;
}

if(rPlayer.IsAlly(piPlayer)) return false;
if(rPlayer.IsAlive())
{
    rPlayer.Defeat();
    KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
}
return true;
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenDestroyed;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    rLastPlayer=null;
    iAlivePlayers=0;
    if(comboAlliedVictory)
    {
        bOneHasBeenDestroyed=false;
        for(i=0;i<15;j=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer==null && !rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if ((iCountBuilding==0)rPlayer.Defeat();
                }
                if(rPlayer!=null && !rPlayer.IsAlive())
                    bOneHasBeenDestroyed=true;
            }
        }

        bActiveEnemies=false;
        for(i=0;i<15;j=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer==null && !rPlayer.IsAlive())
            {
                for(j=i+1;j<15;j=j+1)
                {
                    rPlayer2 = GetPlayer(j);
                    if(rPlayer2!=null && rPlayer2.IsAlive() &&
rPlayer.IsAlly(rPlayer2))
                    {
                        bActiveEnemies=true;
                    }
                }
            }
        }
        if(bActiveEnemies) return;
        if(bOneHasBeenDestroyed) return;
    }

    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer==null && rPlayer.IsAlive())
        {
            rPlayer.Victory();
        }
    }
    else
    {
        for(i=0;i<15;j=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer==null && rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if ((iCountBuilding==0)
rPlayer.Defeat();
                else
                {
                    iAlivePlayers=iAlivePlayers+1;
                    rLastPlayer=rPlayer;
                }
            }
        }
    }
}

```

```

        if (iAlivePlayers==1 && rLastPlayer!=null)
        {
            rLastPlayer.Victory();
        }
    }
}

event Timer1()
{
    int i;
    int minLeft;
    player rPlayer;

    bCheckBuilding=true;
    if(nTimeLimit)
    {
        minLeft=(nTimeLimit - GetMissionTime())/1200;

        if(minLeft<0)minLeft=0;
        SetConsoleText("translateScriptTimeLeft",minLeft);
        if(minLeft<1)
        {
            for(i=0;i<15;i=i+1)
            {
                rPlayer=GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    rPlayer.Defeat();
                    KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
                }
            }
            nTimeLimit=0;
        }
    }
}

command Initialize()
{
    comboMoney=0;
    comboResources=1;
    comboStartingUnits=1;
    comboAlliedVictory=1;
    comboUnitsLimit=2;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboMoney
{
    comboMoney = nMode;
}

command Combo2(int nMode) button comboTime
{
    comboTime = nMode;
}

command Combo3(int nMode) button comboResources
{
    comboResources = nMode;
}

command Combo4(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}

command Combo5(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}

command Combo6(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}
}

DestroyEnemyStructures.ec
mission "translateGameTypeDestroyStructures"
{
    int nTimeLimit;
    int nCashRate;
    int nBuildingType;
}

enum comboCashType
{
    "translateScriptMineForMoney",
    "translateScript2500CRmin",
    "translateScript5000CRmin",
    "translateScript10000CRmin",
    "translateScript20000CRmin",
}

multi:
    "translateScriptGainingMoney"
}

/*enum comboStartingUnits
{
    "translateGameMenuStartingUnitsDefault",
    "translateGameMenuStartingUnitsBuilderOnly",
    multi:
        "translateGameMenuStartingUnits"
} */
enum comboAlliedVictory
{
    "translateGameMenuAlliedVictoryNo",
    "translateGameMenuAlliedVictoryYes",
}

multi:
    "translateGameMenuAlliedVictory"
}

enum comboUnitsLimit
{
    "translateGameMenuUnitsLimitNoLimit",
    "translateGameMenuUnitsLimit10000CR",
    "translateGameMenuUnitsLimit20000CR",
    "translateGameMenuUnitsLimit30000CR",
    "translateGameMenuUnitsLimit50000CR",
}

multi:
    "translateGameMenuUnitsLimit"
}

enum comboWinCondition
{
    "translateScriptDestroyAllStructures",
    "translateScriptDestroyMainStructures",
    "translateScriptDestroyPowerPlants",
    "translateScriptDestroyFactories",
}

multi:
    "translateScriptVictoryCondition"
}

enum comboResearchTime
{
    "translateScriptNormalTime",
    "translateScript2xFaster",
    "translateScript4xFaster",
    "translateScript8xFaster",
}

multi:
    "translateScriptResearchTime"
}

enum comboResearchLimit
{
    "translateScriptAllResearches",
    "translateScriptNoBombs",
    "translateScriptNoMassDestructionWeapons",
    "translateScriptNoBombsAndMDW",
}

multi:
    "translateScriptAvailableResearches"
}

state Initialize;
state Nothing;

state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

    nStartingMoney = 25000;

    if(comboCashType==0) nCashRate = 0;
    if(comboCashType==1) nCashRate = 2500;
    if(comboCashType==2) nCashRate = 5000;
    if(comboCashType==3) nCashRate = 10000;
    if(comboCashType==4) nCashRate = 20000;

    if(comboResearchTime==1) SetTimeDivider(2);
    if(comboResearchTime==2) SetTimeDivider(4);
    if(comboResearchTime==3) SetTimeDivider(8);

    ResourcesPerContainer(2);
}

```

```

for(i=0;i<15;i=i+1)
{
    rPlayer=GetPlayer(i);
    if(rPlayer!=null)
    {
        if(comboAlliedVictory)
            rPlayer.EnableAIFeatures2(ai2BNSendResult,false);//nie wysylac
    }
}

resultatow do EARTH NETU

    rPlayer.SetScriptData(0,0);
    if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
    if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
    if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
    if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
    if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

    rPlayer.SetMoney(nStartingMoney);

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0,0,20,0);
    if(!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
;

rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0)
;

    if(comboCashType>0)
    {

rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMini
ngUnits,false);

        rPlayer.EnableBuilding("EDBM%",false);
        rPlayer.EnableBuilding("EDRE%",false);
        rPlayer.EnableBuilding("UCSBR%",false);
        rPlayer.EnableBuilding("LCBMR%",false);

        rPlayer.EnableResearch("RES_UCS_UOH2",false);
        rPlayer.EnableResearch("RES_UCS_UOH3",false);
        rPlayer.EnableResearch("RES_UCS_UAH1",false);
        rPlayer.EnableResearch("RES_UCS_UAH2",false);
        rPlayer.EnableResearch("RES_UCS_UAH3",false);
    }

    if(comboResearchLimit==1 || comboResearchLimit==3)
    {
        //no bombs
        rPlayer.EnableResearch("RES_ED_AB1",false);
        rPlayer.EnableResearch("RES_ED_MB2",false);
        rPlayer.EnableResearch("RES_UCS_WAPB1",false);
        rPlayer.EnableResearch("RES_UCS_MB2",false);
    }

    if(comboResearchLimit==2 || comboResearchLimit==3)
    {
        //no UW
        rPlayer.EnableResearch("RES_ED_WHR1",false);
        rPlayer.EnableResearch("RES_LC_BWC",false);
        rPlayer.EnableResearch("RES_LC_SDIDEF",false);
        rPlayer.EnableResearch("RES_UCS_PC",false);
        rPlayer.EnableResearch("RES_UCS_WSD",false);
    }
}

SetTimer(0,100);
SetTimer(2,1200);
SetTimer(1,400);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    if(comboCashType==0)
        false;
    else
        true;
}

event RemoveUnits()
{
    /* if(comboStartingUnits)
        true;
    else
        */
}

false;/*
true;
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenDestroyed;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;
}

rLastPlayer=null;
iAlivePlayers=0;
bOneHasBeenDestroyed=false;
for(i=0;i<15;i=i+1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null)
    {
        if(rPlayer.IsAlive())
        {
            if(comboWinCondition==0)
                iCountBuilding = rPlayer.GetNumberOfBuildings();

            if(comboWinCondition==1)//main structures
            {
                iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery) +
rPlayer.GetNumberOfBuildings(buildingBase) +
rPlayer.GetNumberOfBuildings(buildingFactory) +
rPlayer.GetNumberOfBuildings(buildingWaterBase) +
rPlayer.GetNumberOfBuildings(buildingSupplyCenter) +
rPlayer.GetNumberOfBuildings(buildingMine) +
rPlayer.GetNumberOfBuildings(buildingRefinery) +
rPlayer.GetNumberOfBuildings(buildingResearchCenter) +
rPlayer.GetNumberOfBuildings(buildingHeadquarter) +
rPlayer.GetNumberOfBuildings(buildingBBC) +
rPlayer.GetNumberOfBuildings(buildingPlasmaControl) +
rPlayer.GetNumberOfBuildings(buildingWeatherControl) +
rPlayer.GetNumberOfBuildings(buildingMiningRefinery) +
rPlayer.GetNumberOfBuildings(buildingBaseFactory);
            }

            if(comboWinCondition==2)//power plants
            {
                iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery);
            }

            if(comboWinCondition==3)//factories
            {
                iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingBase) +
rPlayer.GetNumberOfBuildings(buildingFactory) +
rPlayer.GetNumberOfBuildings(buildingWaterBase) +
rPlayer.GetNumberOfBuildings(buildingBaseFactory);
            }
        }

        if(iCountBuilding) rPlayer.SetScriptData(0,1);
    }

    if (iCountBuilding==0)
    {
        if(rPlayer.GetScriptData(0)==1)
        {
            rPlayer.Defeat();
            KillArea(rPlayer.GetIFF().GetRight()/2,GetBottom()/2,0,128);
        }
        else
        {
            if(rPlayer.GetNumberOfUnits()==0) rPlayer.Defeat();
        }
    }
}

if(!rPlayer.IsAlive()) bOneHasBeenDestroyed=true;
}

bActiveEnemies=false;
}

```

```

for(i=0;i<15;i+=1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        for(j=i+1;j<15;j+=1)
        {
            rPlayer2 = GetPlayer(j);
            if(rPlayer2!=null && rPlayer2.IsAlive() && !(rPlayer.IsAlly(rPlayer2)
&& comboAlliedVictory))
            {
                bActiveEnemies=true;
            }
        }
    }
    if(bActiveEnemies) return;
    if(!bOneHasBeenDestroyed) return;

    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            rPlayer.Victory();
        }
    }
}

event Timer1()
{
    SetConsoleText("");
}

event Timer2()
{
    int i;
    player rPlayer;

    if(comboWinCondition==0)
        SetConsoleText("translateScriptDestroyAllStructures");
    if(comboWinCondition==1)
        SetConsoleText("translateScriptDestroyMainStructures");
    if(comboWinCondition==2)
        SetConsoleText("translateScriptDestroyPowerPlants");
    if(comboWinCondition==3)
        SetConsoleText("translateScriptDestroyFactories");

    for(i=0;i<15;i+=1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            if(rPlayer.GetMoney()<100000)
                rPlayer.AddMoney(nCashRate);
        }
    }
}

command Initialize()
{
    comboCashType=0://mine for money
    //comboStartingUnits=1;
    comboAlliedVictory=1;
    comboWinCondition=0://destroy all structures
    comboUnitsLimit=2;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboCashType
{
    comboCashType = nMode;
}

command Combo2(int nMode) button comboWinCondition
{
    comboWinCondition = nMode;
}

command Combo3(int nMode) button comboResearchTime
{
    comboResearchTime = nMode;
}

command Combo4(int nMode) button comboResearchLimit
{
    comboResearchLimit = nMode;
}

command Combo5(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}

command Combo6(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}

/*
command Combo6(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}*/



}

}

DieHard412.ec
mission "Die Hard 4 1/2"
{
    int bGameEnded;

    enum comboResearchLimit
    {
        "translateScriptAllResearches",
        "translateScriptNoBombs",
        "translateScriptNoMassDestructionWeapons",
        "translateScriptNoBombsAndMDW",
        multi:
        "translateScriptAvailableResearches"
    }

    enum comboWinCondition
    {
        "translateGameTypeHarvestAndKill",
        "translateScriptDestroyAllStructures",
        "translateScriptDestroyMainStructures",
        "translateScriptDestroyPowerPlants",
        "translateScriptDestroyFactories",
        multi:
        "translateScriptVictoryCondition"
    }

    enum comboStartingUnits
    {
        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
        multi:
        "translateGameMenuStartingUnits"
    }

    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit20000CR",
        "translateGameMenuUnitsLimit50000CR",
        "100000 CR",
        multi:
        "translateGameMenuUnitsLimit"
    }

    enum comboAlliedVictory
    {
        "translateGameMenuAlliedVictoryNo",
        "translateGameMenuAlliedVictoryYes",
        multi:
        "translateGameMenuAlliedVictory"
    }

    state Initialize;
    state Nothing;
    state Initialize
}

```

```

    {
        player rPlayer;
        int i;
        int nStartingMoney;

        bGameEnded=false;
        for(i=0;i<15;i+=1)
        {
            rPlayer=GetPlayer(i);

            if(rPlayer!=null)
            {
                if(comboAlliedVictory)
                    rPlayer.EnableAIFeatures2(ai2BNSSendResult,false); //nie wysylac
            rezultatow do EARTH NETU

            rPlayer.SetMoney(100000);

            rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMiningUnits,false);

            rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);

            if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
            if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(20000);
            if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(50000);
            if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(100000);

            if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
        }

        rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0,0);

        rPlayer.EnableBuilding("EDBML",false);
        rPlayer.EnableBuilding("EDBRC",false);
        rPlayer.EnableBuilding("UCSBRF",false);
        rPlayer.EnableBuilding("LCBMR",false);

        rPlayer.EnableResearch("RES_UCS_UOH2",false);
        rPlayer.EnableResearch("RES_UCS_UOH3",false);
        rPlayer.EnableResearch("RES_UCS_UAH1",false);
        rPlayer.EnableResearch("RES_UCS_UAH2",false);
        rPlayer.EnableResearch("RES_UCS_UAH3",false);

        if(comboResearchLimit==1 || comboResearchLimit==3)
        {
            //no bombs
            rPlayer.EnableResearch("RES_ED_AB1",false);
            rPlayer.EnableResearch("RES_ED_MB2",false);
            rPlayer.EnableResearch("RES_UCS_WAPB1",false);
            rPlayer.EnableResearch("RES_UCS_MB2",false);
        }

        if(comboResearchLimit==2 || comboResearchLimit==3)
        {
            //no UW
            rPlayer.EnableResearch("RES_ED_WHR1",false);
            rPlayer.EnableResearch("RES_LC_BWC",false);
            rPlayer.EnableResearch("RES_LC_SDIDEF",false);
            rPlayer.EnableResearch("RES_UCS_PC",false);
            rPlayer.EnableResearch("RES_UCS_WSD",false);
        }

        //neutral stuff
        rPlayer.AddResearch("RES_MCH2");
        rPlayer.AddResearch("RES_MCH3");
        rPlayer.AddResearch("RES_MCH4");
        rPlayer.AddResearch("RES_MR2");
        rPlayer.AddResearch("RES_MR3");
        rPlayer.AddResearch("RES_MR4");
        rPlayer.AddResearch("RES_MM2");
        rPlayer.AddResearch("RES_MM3");
        rPlayer.AddResearch("RES_MM4");

        if(rPlayer.GetRace()==1)//UCS
        {
            rPlayer.AddResearch("RES_UCS_USL2");
            rPlayer.AddResearch("RES_UCS_USL3");
            rPlayer.AddResearch("RES_UCS_UML1");
            rPlayer.AddResearch("RES_UCS_UML2");
            rPlayer.AddResearch("RES_UCS_UML3");
            rPlayer.AddResearch("RES_UCS_UHL1");
            rPlayer.AddResearch("RES_UCS_UHL2");
            rPlayer.AddResearch("RES_UCS_UHL3");
        }
    }

    rPlayer.AddResearch("RES_UCS_UJL1");
    rPlayer.AddResearch("RES_UCS_UJL2");
    rPlayer.AddResearch("RES_UCS_UJL3");
    rPlayer.AddResearch("RES_UCS_UBL1");
    rPlayer.AddResearch("RES_UCS_UBL2");
    rPlayer.AddResearch("RES_UCS_UM1");
    rPlayer.AddResearch("RES_UCS_UM2");
    rPlayer.AddResearch("RES_UCS_US1");
    rPlayer.AddResearch("RES_UCS_US2");
    rPlayer.AddResearch("RES_UCS_US3");
    rPlayer.AddResearch("RES_UCS_GARG1");
    rPlayer.AddResearch("RES_UCS_GARG2");
    rPlayer.AddResearch("RES_UCS_GARG3");
    rPlayer.AddResearch("RES_UCS_BOMBER21");
    rPlayer.AddResearch("RES_UCS_BOMBER22");
    rPlayer.AddResearch("RES_UCS_BOMBER31");
    rPlayer.AddResearch("RES_UCS_BOMBER32");
    rPlayer.AddResearch("RES_UCS_BMD");
    rPlayer.AddResearch("RES_UCS_BHD");
    rPlayer.AddResearch("RES_UCS_WCH2");
    rPlayer.AddResearch("RES_UCS_WCH2");
    rPlayer.AddResearch("RES_UCS_WSR1");
    rPlayer.AddResearch("RES_UCS_WSR2");
    rPlayer.AddResearch("RES_UCS_WG1");
    rPlayer.AddResearch("RES_UCS_WG2");
    rPlayer.AddResearch("RES_UCS_WHG1");
    rPlayer.AddResearch("RES_UCS_WHG2");
    rPlayer.AddResearch("RES_UCS_WMR1");
    rPlayer.AddResearch("RES_UCS_WMR2");
    rPlayer.AddResearch("RES_UCS_WMR3");
    rPlayer.AddResearch("RES_UCS_WAMR1");
    rPlayer.AddResearch("RES_UCS_WSP1");
    rPlayer.AddResearch("RES_UCS_WSP2");
    rPlayer.AddResearch("RES_UCS_WHP1");
    rPlayer.AddResearch("RES_UCS_WHP2");
    rPlayer.AddResearch("RES_UCS_WHP3");

    if (comboResearchLimit!=2&&comboResearchLimit!=3)
        rPlayer.AddResearch("RES_UCS_PC");

    if (comboResearchLimit!=1&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_UCS_WAPB1");
        rPlayer.AddResearch("RES_UCS_WAPB2");
    }

    if (comboResearchLimit!=2&&comboResearchLimit!=3)
        rPlayer.AddResearch("RES_UCS_WSD");
        rPlayer.AddResearch("RES_UCS_RepHand");
        rPlayer.AddResearch("RES_UCS_RepHand2");
        rPlayer.AddResearch("RES_UCS_SGen");
        rPlayer.AddResearch("RES_UCS_MGen");
        rPlayer.AddResearch("RES_UCS_HGen");
        rPlayer.AddResearch("RES_UCS_SHD");
        rPlayer.AddResearch("RES_UCS_SHD2");
        rPlayer.AddResearch("RES_UCS_SHD3");
        rPlayer.AddResearch("RES_UCS_SHD4");

    if (comboResearchLimit!=1&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_UCS_MB2");
        rPlayer.AddResearch("RES_UCS_MB3");
        rPlayer.AddResearch("RES_UCS_MB4");
    }

    if(rPlayer.GetRace()==2)//ed
    {
        rPlayer.AddResearch("RES_ED_UST2");
        rPlayer.AddResearch("RES_ED_UST3");
        rPlayer.AddResearch("RES_ED_UHT1");
        rPlayer.AddResearch("RES_ED_UHT2");
        rPlayer.AddResearch("RES_ED_UHT3");
    }

```

```

rPlayer.AddResearch("RES_ED_UBT1");
rPlayer.AddResearch("RES_ED_UTB2");
rPlayer.AddResearch("RES_ED_UMT1");
rPlayer.AddResearch("RES_ED_UTM2");
rPlayer.AddResearch("RES_ED_UTM3");
rPlayer.AddResearch("RES_ED_UIM1");
rPlayer.AddResearch("RES_ED_UIM2");
rPlayer.AddResearch("RES_ED_UMW1");
rPlayer.AddResearch("RES_ED_UMW2");
rPlayer.AddResearch("RES_ED_UHW1");
rPlayer.AddResearch("RES_ED_UHW2");
rPlayer.AddResearch("RES_ED_USS2");
rPlayer.AddResearch("RES_ED_USS3");
rPlayer.AddResearch("RES_ED_UHS1");
rPlayer.AddResearch("RES_ED_UHS2");
rPlayer.AddResearch("RES_ED_UA11");
rPlayer.AddResearch("RES_ED_UA12");
rPlayer.AddResearch("RES_ED_UA21");
rPlayer.AddResearch("RES_ED_UA22");
rPlayer.AddResearch("RES_ED_UA41");
rPlayer.AddResearch("RES_ED_UA42");
rPlayer.AddResearch("RES_ED_UA31");
rPlayer.AddResearch("RES_ED_UA32");
rPlayer.AddResearch("RES_ED_BMD");
rPlayer.AddResearch("RES_ED_BHD");
rPlayer.AddResearch("RES_ED_WCH2");
rPlayer.AddResearch("RES_ED_ACH2");
rPlayer.AddResearch("RES_ED_WCA2");
rPlayer.AddResearch("RES_ED_WHC1");
rPlayer.AddResearch("RES_ED_WHC2");
rPlayer.AddResearch("RES_ED_WSR1");
rPlayer.AddResearch("RES_ED_WSR2");
rPlayer.AddResearch("RES_ED_WSR3");
rPlayer.AddResearch("RES_ED_ASR1");
rPlayer.AddResearch("RES_ED_ASR2");
rPlayer.AddResearch("RES_ED_WNR");
rPlayer.AddResearch("RES_ED_WMR1");
rPlayer.AddResearch("RES_ED_WMR2");
rPlayer.AddResearch("RES_ED_WMR3");
rPlayer.AddResearch("RES_ED_AMR1");
rPlayer.AddResearch("RES_ED_AMR2");

if (comboResearchLimit!=2&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_ED_WHR1");
    rPlayer.AddResearch("RES_ED_WSL1");
    rPlayer.AddResearch("RES_ED_WSL2");
    rPlayer.AddResearch("RES_ED_WSL3");
    rPlayer.AddResearch("RES_ED_WHL1");
    rPlayer.AddResearch("RES_ED_WHL2");
    rPlayer.AddResearch("RES_ED_WSL3");
    rPlayer.AddResearch("RES_ED_WS11");
    rPlayer.AddResearch("RES_ED_WS12");
    rPlayer.AddResearch("RES_ED_WH11");
    rPlayer.AddResearch("RES_ED_WH12");
}

if (comboResearchLimit!=1&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_ED_AB1");
    rPlayer.AddResearch("RES_ED_AB2");
}
    rPlayer.AddResearch("RES_ED_RepHand");
    rPlayer.AddResearch("RES_ED_RepHand2");
    rPlayer.AddResearch("RES_ED_SGen");
    rPlayer.AddResearch("RES_ED_MGen");
    rPlayer.AddResearch("RES_ED_HGen");
    rPlayer.AddResearch("RES_ED_SCR");
    rPlayer.AddResearch("RES_ED_SCR2");
    rPlayer.AddResearch("RES_ED_SCR3");
    rPlayer.AddResearch("RES_ED_MHR2");
    rPlayer.AddResearch("RES_ED_MHR3");
    rPlayer.AddResearch("RES_ED_MHR4");
}

if (comboResearchLimit!=1&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_ED_MB2");
    rPlayer.AddResearch("RES_ED_MB3");
    rPlayer.AddResearch("RES_ED_MB4");
}
    rPlayer.AddResearch("RES_ED_MSC");
    rPlayer.AddResearch("RES_ED_MSC2");
    rPlayer.AddResearch("RES_ED_MSC4");
    rPlayer.AddResearch("RES_ED_MHC2");
}

rPlayer.AddResearch("RES_ED_MHC3");
rPlayer.AddResearch("RES_ED_MHC4");
}

if(Player.GetRace() == 3)//lc
{
    rPlayer.AddResearch("RES_LC_ULU2");
    rPlayer.AddResearch("RES_LC_ULU3");
    rPlayer.AddResearch("RES_LC_UMO2");
    rPlayer.AddResearch("RES_LC_UMO3");
    rPlayer.AddResearch("RES_LC_UCR1");
    rPlayer.AddResearch("RES_LC_UCR2");
    rPlayer.AddResearch("RES_LC_UCR3");
    rPlayer.AddResearch("RES_LC_UCU1");
    rPlayer.AddResearch("RES_LC_UCU2");
    rPlayer.AddResearch("RES_LC_UCU3");
    rPlayer.AddResearch("RES_LC_UME1");
    rPlayer.AddResearch("RES_LC_UME2");
    rPlayer.AddResearch("RES_LC_UME3");
    rPlayer.AddResearch("RES_LC_UBO1");
    rPlayer.AddResearch("RES_LC_UBO2");
    rPlayer.AddResearch("RES_LC_BMD");
    rPlayer.AddResearch("RES_LC_BHD");
}

if (comboResearchLimit==2&&comboResearchLimit!=3)
    rPlayer.AddResearch("RES_LC_BWC");
if (comboResearchLimit==2&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_LC_SDIDEF");
    rPlayer.AddResearch("RES_LC_WCH2");
    rPlayer.AddResearch("RES_LC_ACH2");
    rPlayer.AddResearch("RES_LC_WSR2");
    rPlayer.AddResearch("RES_LC_WSR3");
    rPlayer.AddResearch("RES_LC_ASR1");
    rPlayer.AddResearch("RES_LC_ASR2");
    rPlayer.AddResearch("RES_LC_WMR1");
    rPlayer.AddResearch("RES_LC_WMR2");
    rPlayer.AddResearch("RES_LC_WMR3");
    rPlayer.AddResearch("RES_LC_AMR1");
    rPlayer.AddResearch("RES_LC_AMR2");
    rPlayer.AddResearch("RES_LC_WSL1");
    rPlayer.AddResearch("RES_LC_WSL2");
    rPlayer.AddResearch("RES_LC_WHL1");
    rPlayer.AddResearch("RES_LC_WHL2");
    rPlayer.AddResearch("RES_LC_WSS1");
    rPlayer.AddResearch("RES_LC_WSS2");
    rPlayer.AddResearch("RES_LC_WH1");
    rPlayer.AddResearch("RES_LC_WH2");
    rPlayer.AddResearch("RES_LC_WAS1");
    rPlayer.AddResearch("RES_LC_WAS2");
    rPlayer.AddResearch("RES_LC_WARTILLERY");
    rPlayer.AddResearch("RES_LC_SGen");
    rPlayer.AddResearch("RES_LC_MGen");
    rPlayer.AddResearch("RES_LC_HGen");
    rPlayer.AddResearch("RES_LC_SHR1");
    rPlayer.AddResearch("RES_LC_SHR2");
    rPlayer.AddResearch("RES_LC_SHR3");
    rPlayer.AddResearch("RES_LC_REG1");
    rPlayer.AddResearch("RES_LC_REG2");
    rPlayer.AddResearch("RES_LC_REG3");
    rPlayer.AddResearch("RES_LC_SOBI");
    rPlayer.AddResearch("RES_LC_SOBI2");
}

SetTimer(0,100);
SetTimer(1,20);
SetTimer(2,1200);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    true;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
}

```

```

        else
            false;
    }

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenCalled;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    rLastPlayer=null;
    iAlivePlayers=0;
    bOneHasBeenCalled=false;
    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.IsAlive())
                if(comboWinCondition==1)
                    iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if(comboWinCondition==1)
                    iCountBuilding = rPlayer.GetNumberOfBuildings();

                if(comboWinCondition==2)//main structures
                {
                    iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery) +
rPlayer.GetNumberOfBuildings(buildingBase) +
rPlayer.GetNumberOfBuildings(buildingFactory) +
rPlayer.GetNumberOfBuildings(buildingWaterBase) +
rPlayer.GetNumberOfBuildings(buildingSupplyCenter) +
rPlayer.GetNumberOfBuildings(buildingMine) +
rPlayer.GetNumberOfBuildings(buildingRefinery) +
rPlayer.GetNumberOfBuildings(buildingResearchCenter) +
rPlayer.GetNumberOfBuildings(buildingHeadquarter) +
rPlayer.GetNumberOfBuildings(buildingBBC) +
rPlayer.GetNumberOfBuildings(buildingPlasmaControl) +
rPlayer.GetNumberOfBuildings(buildingWeatherControl) +
rPlayer.GetNumberOfBuildings(buildingMiningRefinery) +
rPlayer.GetNumberOfBuildings(buildingBaseFactory);
                }
                if(comboWinCondition==3)//power plants
                    iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery);

                if(comboWinCondition==4)//factories
                {
                    iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingBase) +
rPlayer.GetNumberOfBuildings(buildingFactory) +
rPlayer.GetNumberOfBuildings(buildingWaterBase) +
rPlayer.GetNumberOfBuildings(buildingBaseFactory);
                }
                if(iCountBuilding) rPlayer.SetScriptData(0,1);
                if (iCountBuilding==0)
                {
                    if(rPlayer.GetScriptData(0)==1)
                    {
                        rPlayer.Defeat();
                }
            }
        }
    }
}

KillArea(rPlayer.GetIFF().GetRight()/2,GetBottom()/2,0,128);
}
else
{
    if(rPlayer.GetNumberOfUnits()==0) rPlayer.Defeat();
}
}
if(!rPlayer.IsAlive()) bOneHasBeenCalled=true;
}
}

bActiveEnemies=false;
for(i=0;i<15;j=i+1)
{
    rPlayer = GetPlayer(i);
    if((rPlayer!=null && rPlayer.IsAlive()))
    {
        for(j=i+1;j<15;j=j+1)
        {
            rPlayer2 = GetPlayer(j);
            if(rPlayer2!=null && rPlayer2.IsAlive() && !(rPlayer.IsAlly(rPlayer2) && comboAlliedVictory)))
            {
                bActiveEnemies=true;
            }
        }
    }
}
if(bActiveEnemies) return;
if(!bOneHasBeenCalled) return;

for(i=0;i<15;j=i+1)
{
    rPlayer = GetPlayer(i);
    if((rPlayer!=null && rPlayer.IsAlive()))
    {
        rPlayer.Victory();
    }
}
}

event Timer1()
{
    int i;
    player rPlayer;

    for(i=0;i<15;i+=1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMoney()<100000)
                rPlayer.AddMoney(100000 - rPlayer.GetMoney());
        }
    }
}

event Timer2()
{
    if(comboWinCondition==0)
        SetConsoleText("Destroy everything");
    if(comboWinCondition==1)
        SetConsoleText("translateScriptDestroyAllStructures");
    if(comboWinCondition==2)
        SetConsoleText("translateScriptDestroyMainStructures");
    if(comboWinCondition==3)
        SetConsoleText("translateScriptDestroyPowerPlants");
    if(comboWinCondition==4)
        SetConsoleText("translateScriptDestroyFactories");
}

command Initialize()
{
    comboResearchLimit=0;
    comboWinCondition=0;
    comboStartingUnits=1;
    comboAllydVictory=1;
    comboUnitsLimit=2;
}

command Combo1(int nMode) button comboResearchLimit
{
    comboResearchLimit = nMode;
}

```

```

        }
        command Combo2(int nMode) button comboWinCondition
        {
            comboWinCondition = nMode;
        }
        command Combo3(int nMode) button comboStartingUnits
        {
            comboStartingUnits = nMode;
        }
        command Combo4(int nMode) button comboUnitsLimit
        {
            comboUnitsLimit = nMode;
        }
        command Combo5(int nMode) button comboAlliedVictory
        {
            comboAlliedVictory = nMode;
        }
    }

}


```

EarnMoney.ec

```

mission "translateGameTypeEarnMoney"
{
    int nMoneyToEarn;
    int nTimeLimit;

    enum comboMoney
    {
        "translateScript120000CR",
        "translateScript130000CR",
        "translateScript140000CR",
        multi:
        "translateGameMenuStartingMoney"
    }
    enum comboMoneyToEarn
    {
        "translateScript250000CR",
        "translateScript400000CR",
        "translateScript500000CR",
        "translateScript750000CR",
        "translateScript1000000CR",
        multi:
        "translateGameMenuMoneyToGain"
    }
    enum comboTime
    {
        "translateGameMenuTimeLimitNoLimit",
        "translateGameMenuTimeLimit1Min",
        "translateGameMenuTimeLimit30min",
        "translateGameMenuTimeLimit45min",
        "translateGameMenuTimeLimit1h",
        "translateGameMenuTimeLimit15h",
        multi:
        "translateGameMenuTimeLimit"
    }

    enum comboResources
    {
        "translateGameMenuResourcesLow",
        "translateGameMenuResourcesNormal",
        "translateGameMenuResourcesHigh",
        "translateGameMenuResourcesVeryHigh",
        multi:
        "translateGameMenuResources"
    }
    enum comboStartingUnits
    {
        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
        multi:
        "translateGameMenuStartingUnits"
    }

    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit100000CR",
        "translateGameMenuUnitsLimit200000CR",
        "translateGameMenuUnitsLimit300000CR",
        "translateGameMenuUnitsLimit500000CR",
        multi:
        "translateGameMenuUnitsLimit"
    }
}


```

```

state Initialize;
state Nothing;

state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

    if(comboMoney==0) nStartingMoney=20000;
    if(comboMoney==1) nStartingMoney=30000;
    if(comboMoney==2) nStartingMoney=40000;
    if(comboMoney==3) nStartingMoney=50000;
    if(comboMoney==4) nStartingMoney=75000;
    if(comboMoney==5) nStartingMoney=100000;

    nMoneyToEarn=25000;
    if(comboMoneyToEarn==1) nMoneyToEarn=40000;
    if(comboMoneyToEarn==2) nMoneyToEarn=50000;
    if(comboMoneyToEarn==3) nMoneyToEarn=75000;
    if(comboMoneyToEarn==4) nMoneyToEarn=100000;

    if(comboTime==0)nTimeLimit;
    if(comboTime==1)nTimeLimit=15*60*20;
    if(comboTime==2)nTimeLimit=30*60*20;
    if(comboTime==3)nTimeLimit=45*60*20;
    if(comboTime==4)nTimeLimit=60*60*20;
    if(comboTime==5)nTimeLimit=90*60*20;

    if(comboResources==0)ResourcesPerContainer(16);
    if(comboResources==1)ResourcesPerContainer(8);
    if(comboResources==2)ResourcesPerContainer(4);
    if(comboResources==3)ResourcesPerContainer(2);

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer==null)
        {
            rPlayer.SetAllowGiveMoney(false);
            rPlayer.SetMoney(nStartingMoney);
        }
    }
}

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0,0,20,0);

if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
{
    rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0);
}

SetTimer(0,100);
SetTimer(1,1200);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    false;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

event Timer0()
{
    int iLivePlayers;
    int i;
    int bOthersDefeat;
    int iOwnBuilding;
    player rPlayer;
    player rLastPlayer;

    rLastPlayer=null;
}

```

```

iAlivePlayers=0;
bOthersDefeat=false;
for(i=0;i<15;i=i+1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
        if (iCountBuilding>0)
        {
            iAlivePlayers=iAlivePlayers+1;
            rLastPlayer=rPlayer;
            if(rPlayer.GetMoney()>=nMoneyToEarn)
            {
                rPlayer.Victory();
                bOthersDefeat=true;
            }
        }
        else
        {
            rPlayer.Defeat();
        }
    }
}
if (iAlivePlayers==1 && rLastPlayer!=null && rLastPlayer.IsAlive())
{
    rLastPlayer.Victory();
}

if(bOthersDefeat==true)
{
    for(i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            if(rPlayer.GetMoney()<nMoneyToEarn)
            {
                rPlayer.Defeat();
                KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
            }
        }
    }
}
event Timer1()
{
    int i;
    int minLeft;
    player rPlayer;

    if(nTimeLimit)
    {
        minLeft=(nTimeLimit - GetMissionTime())/1200;

        if(minLeft<0)minLeft=0;
        SetConsoleText("translateScriptTimeLeft",minLeft);
        if(minLeft<1)
        {
            for(i=0;i<15;i=i+1)
            {
                rPlayer = GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    rPlayer.Defeat();
                    KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
                }
            }
            nTimeLimit=0;
        }
    }
}

command Initialize()
{
    nTimeLimit=0;
    comboResources=1;
    comboStartingUnits=1;
    comboUnitsLimit=2;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboMoney
{
    comboMoney=nMode;
    if(comboMoneyToEarn < comboMoney) comboMoneyToEarn =
comboMoney;
}

command Combo2(int nMode) button comboMoneyToEarn
{
    comboMoneyToEarn=nMode;
    if(comboMoneyToEarn < comboMoney) comboMoney =
comboMoneyToEarn;
}

command Combo3(int nMode) button comboTime
{
    comboTime = nMode;
}

command Combo4(int nMode) button comboResources
{
    comboResources=nMode;
}

command Combo5(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}

command Combo6(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}

HarvestAndKill.ec
mission "translateGameTypeHarvestAndKill"
{
    int nTimeLimit;
    int bCheckBuilding;
    int nMeteor;

    enum comboMoney
    {
        "translateScript20000CR",
        "translateScript30000CR",
        "translateScript40000CR",
        "translateScript50000CR",
    }

    multi:
        "translateGameMenuStartingMoney"
    }

    enum comboTime
    {
        "translateGameMenuTimeLimitNoLimit",
        "translateGameMenuTimeLimit1Min",
        "translateGameMenuTimeLimit30min",
        "translateGameMenuTimeLimit45min",
        "translateGameMenuTimeLimit1h",
        "translateGameMenuTimeLimit15h",
    }

    multi:
        "translateGameMenuTimeLimit"
    }

    enum comboResources
    {
        "translateGameMenuResourcesLow",
        "translateGameMenuResourcesNormal",
        "translateGameMenuResourcesHigh",
        "translateGameMenuResourcesVeryHigh",
    }

    multi:
        "translateGameMenuResources"
    }

    enum comboStartingUnits
    {
        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
    }

    multi:
        "translateGameMenuStartingUnits"
    }

    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit10000CR",
    }
}

```

```

        "translateGameMenuUnitsLimit20000CR",
        "translateGameMenuUnitsLimit30000CR",
        "translateGameMenuUnitsLimit50000CR",
multi:
    "translateGameMenuUnitsLimit"
}
enum comboAlliedVictory
{
    "translateGameMenuAlliedVictoryNo",
    "translateGameMenuAlliedVictoryYes",
multi:
    "translateGameMenuAlliedVictory"
}
state Initialize;
state Nothing;
state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

if(comboMoney==0)nStartingMoney = 20000;
if(comboMoney==1)nStartingMoney = 30000;
if(comboMoney==2)nStartingMoney = 40000;
if(comboMoney==3)nStartingMoney = 50000;

if(comboTime==0)nTimeLimit=15*60*20;
if(comboTime==1)nTimeLimit=30*60*20;
if(comboTime==2)nTimeLimit=45*60*20;
if(comboTime==3)nTimeLimit=60*60*20;
if(comboTime==4)nTimeLimit=90*60*20;

if(comboResources==0) ResourcesPerContainer(16);
if(comboResources==1) ResourcesPerContainer(8);
if(comboResources==2) ResourcesPerContainer(4);
if(comboResources==3) ResourcesPerContainer(2);

bCheckBuilding=false;
for(i=0;i<15;i+=1)
{
    rPlayer=GetPlayer(i);

    if(rPlayer!=null)
    {
        if(comboAlliedVictory)
            rPlayer.EnableAllFeatures2(ai2BNSendResult,false);//nie wysylac
rezultatow do EARTH NETu

        if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
        if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
        if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
        if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
        if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

        rPlayer.SetMoney(nStartingMoney);

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6.0,20.0);
        if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0)
    }
}
SetTimer(0,100);
SetTimer(1,1200);

nMeteor=1;
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    false;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenDestroyed;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;
    //---meteors
    //rPlayer = GetPlayer(1);
    //IMeteor(rPlayer.GetStartingPointX());
    nMeteor*2,Player.GetStartingPointY(),nMeteor);
    //nMeteor=(nMeteor%10)+1;
    //end of meteors

    if (bCheckBuilding==false)
        return 0;
    rLastPlayer=null;
    iAlivePlayers=0;
    if(comboAlliedVictory)//-----
    {
        bOneHasBeenDestroyed=false;
        for(i=0;i<15;i+=1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if (iCountBuilding==0)rPlayer.Defeat();
            }
            if(rPlayer!=null && !rPlayer.IsAlive())
                bOneHasBeenDestroyed=true;
        }
    }

    bActiveEnemies=false;
    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            for(j=i+1;j<15;j+=1)
            {
                rPlayer2 = GetPlayer(j);
                if(rPlayer2!=null && rPlayer2.IsAlive() &&
rPlayer.IsAlly(rPlayer2))
                {
                    bActiveEnemies=true;
                }
            }
        }
    }
    if(bActiveEnemies) return;
    if(!bOneHasBeenDestroyed) return;

    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            rPlayer.Victory();
        }
    }
else//-----
{
    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
            if (iCountBuilding>0)
            {
                iAlivePlayers=iAlivePlayers+1;
                rLastPlayer=rPlayer;
            }
        }
    }
}
}

```



```

if(comboMoney==2)nStartingMoney = 40000;
if(comboMoney==3)nStartingMoney = 50000;

if(comboTime==0)nTimeLimit=0;
if(comboTime==1)nTimeLimit=15*60*20;
if(comboTime==2)nTimeLimit=30*60*20;
if(comboTime==3)nTimeLimit=45*60*20;
if(comboTime==4)nTimeLimit=60*60*20;
if(comboTime==5)nTimeLimit=90*60*20;

/*nTimeLimit=15*60*20;
if(comboTime==1)nTimeLimit=30*60*20;
if(comboTime==2)nTimeLimit=45*60*20;*/

if(comboResources==0) ResourcesPerContainer(16);
if(comboResources==1) ResourcesPerContainer(8);
if(comboResources==2) ResourcesPerContainer(4);
if(comboResources==3) ResourcesPerContainer(2);

bCheckBuilding=false;
for(=0;<15;=+=1)
{
    rPlayer=GetPlayer(i);

    if(rPlayer!=null)
    {
        if(comboAliedVictory)
            rPlayer.EnableAIFeatures2(ai2BNSendResult,false);/nie wysylac
    rezultatow do EARTH NETU

        if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
        if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
        if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
        if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
        if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

        rPlayer.EnableAIFeatures2(ai2BNSendResult,false);/nie wysylac rezul-
tatow do EARTH NETU

        rPlayer.SetMoney(nStartingMoney);

        rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0,6,20,0);
        if(!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
    }

    rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0)
    ;

    rPlayer.EnableResearch("RES_ED_ACH2",false);
    rPlayer.EnableResearch("RES_ED_WSL1",false);
    rPlayer.EnableResearch("RES_ED_WSL1",false);
    rPlayer.EnableResearch("RES_ED_WSR2",false);
    rPlayer.EnableResearch("RES_ED_ASRI",false);
    rPlayer.EnableResearch("RES_ED_ASRI",false);
    rPlayer.EnableResearch("RES_ED_WHCI",false);
    rPlayer.EnableResearch("RES_ED_WHCI",false);
    rPlayer.EnableResearch("RES_ED_WHL1",false);
    rPlayer.EnableResearch("RES_ED_WHL1",false);
    rPlayer.EnableResearch("RES_ED_WHL1",false);
    rPlayer.EnableResearch("RES_ED_AB1",false);
    /ammo
    rPlayer.EnableResearch("RES_ED_MHC2",false);
    rPlayer.EnableResearch("RES_ED_MB2",false);
    rPlayer.EnableResearch("RES_ED_MSC3",false);
    rPlayer.EnableResearch("RES_ED_NMR2",false);
    rPlayer.EnableResearch("RES_MR3",false);
    rPlayer.EnableResearch("RES_ED_MHR2",false);
    rPlayer.EnableResearch("RES_ED_MHR4",false);
    /chassis
    rPlayer.EnableResearch("RES_ED_UST3",false);
    rPlayer.EnableResearch("RES_ED_UTM2",false);
    rPlayer.EnableResearch("RES_ED_UWV1",false);
    rPlayer.EnableResearch("RES_ED_UM11",false);
    rPlayer.EnableResearch("RES_ED_UHT1",false);
    rPlayer.EnableResearch("RES_ED_UHW1",false);
    rPlayer.EnableResearch("RES_ED_UTB1",false);
    rPlayer.EnableResearch("RES_ED_UA11",false);
    rPlayer.EnableResearch("RES_ED_UA21",false);
    rPlayer.EnableResearch("RES_ED_UA31",false);
    rPlayer.EnableResearch("RES_ED_UA41",false);
    rPlayer.EnableResearch("RES_ED_US22",false);
    rPlayer.EnableResearch("RES_ED_UHS1",false);
    /special
    rPlayer.EnableResearch("RES_ED_SCR",false);
    rPlayer.EnableResearch("RES_ED_SGen",false);
    rPlayer.EnableResearch("RES_ED_BMD",false);
    rPlayer.EnableResearch("RES_ED_BHD",false);

    rPlayer.EnableResearch("RES_LC_WCH2",false);
    rPlayer.EnableResearch("RES_LC_ACH2",false);
    rPlayer.EnableResearch("RES_LC_ASRI",false);
    rPlayer.EnableResearch("RES_LC_WSR2",false);
    //rPlayer.EnableResearch("RES_LC_WSL1",false);
    rPlayer.EnableResearch("RES_LC_WSL2",false);
    rPlayer.EnableResearch("RES_LC_WSS1",false);
    rPlayer.EnableResearch("RES_LC_WMR1",false);
    rPlayer.EnableResearch("RES_LC_WHL1",false);
    rPlayer.EnableResearch("RES_LC_WHST",false);

    rPlayer.EnableResearch("RES_LC_WAS1",false);
    //rPlayer.EnableResearch("RES_LC_UM02",false);
    rPlayer.EnableResearch("RES_LC_UCR1",false);
    rPlayer.EnableResearch("RES_LC_UCU1",false);

    rPlayer.EnableResearch("RES_LC_ULU3",false);
    rPlayer.EnableResearch("RES_LC_UMO3",false);
    rPlayer.EnableResearch("RES_LC_UME1",false);
    rPlayer.EnableResearch("RES_LC_UBO1",false);

    rPlayer.EnableResearch("RES_LC_BMD",false);
    rPlayer.EnableResearch("RES_LC_BHD",false);
    rPlayer.EnableResearch("RES_LC_SDIEF",false);
    //rPlayer.EnableResearch("RES_LC_SGen",false);
    rPlayer.EnableResearch("RES_LC_MGen",false);
    rPlayer.EnableResearch("RES_LC_MGen",false);
    rPlayer.EnableResearch("RES_LC_SHR1",false);
    rPlayer.EnableResearch("RES_LC_REG1",false);
    rPlayer.EnableResearch("RES_LC_SOBI",false);
    rPlayer.EnableResearch("RES_LC_SDIEF",false);
    rPlayer.EnableResearch("RES_LC_BWC",false);
    /weapons
    rPlayer.EnableResearch("RES_UCS_WACH2",false);
    rPlayer.EnableResearch("RES_UCS_WHG1",false);
    rPlayer.EnableResearch("RES_UCS_WSR2",false);
    rPlayer.EnableResearch("RES_UCS_WSP2",false);
    rPlayer.EnableResearch("RES_UCS_WSD",false);
    /ammo
    rPlayer.EnableResearch("RES_UCS_WAPB1",false);
    rPlayer.EnableResearch("RES_UCS_MB2",false);
    /chassis
    rPlayer.EnableResearch("RES_UCS_GARG1",false);
    rPlayer.EnableResearch("RES_UCS_U3L3",false);
    rPlayer.EnableResearch("RES_UCS_UML3",false);
    rPlayer.EnableResearch("RES_UCS_UHL1",false);
    rPlayer.EnableResearch("RES_UCS_UHM1",false);
    rPlayer.EnableResearch("RES_UCS_USS2",false);
    rPlayer.EnableResearch("RES_UCS_UBS1",false);
    rPlayer.EnableResearch("RES_UCS_USM1",false);
    rPlayer.EnableResearch("RES_UCS_UAH1",false);
    rPlayer.EnableResearch("RES_UCS_BOMBER21",false);
    /sPECIAL
    rPlayer.EnableResearch("RES_UCS_BMD",false);
    rPlayer.EnableResearch("RES_UCS_BHD",false);

    rPlayer.EnableResearch("RES_UCS_WSD",false);
    rPlayer.EnableResearch("RES_UCS_PC",false);

    }

    SetTimer(0,100);
    SetTimer(1,1200);

    nMeteor=1;
    return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
}

```

```

        false;
    }

    event RemoveUnits()
    {
        if(comboStartingUnits)
            true;
        else
            false;
    }

    event Timer0()
    {
        int iAlivePlayers;
        int i;
        int j;
        int iCountBuilding;
        int bActiveEnemies;
        int bOneHasBeenCalled;
        player rPlayer;
        player rPlayer2;
        player rLastPlayer;
        //---meteors
        //rPlayer = GetPlayer(1);
        //Meteor(rPlayer.GetStartingPointX());
        nMeteor*2,Player.GetStartingPointY(),nMeteor);
        //nMeteor=(nMeteor%10)+1;
        //end of meteors

        if (bCheckBuilding==false)
            return 0;
        rLastPlayer=null;
        iAlivePlayers=0;
        if(comboAlliedVictory)//-----
        {
            bOneHasBeenCalled=false;
            for(i=0;i<15;i+=1)
            {
                rPlayer = GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                    if (iCountBuilding==0)rPlayer.Defeat();
                }
                if(rPlayer!=null && !rPlayer.IsAlive())
                    bOneBeenCalled=true;
            }
            bActiveEnemies=false;
            for(i=0;i<15;i+=1)
            {
                rPlayer = GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    for(j=i+1;j<15;j=j+1)
                    {
                        rPlayer2 = GetPlayer(j);
                        if(rPlayer2!=null && rPlayer2.IsAlive() &&
rPlayer.IsAlly(rPlayer2))
                        {
                            bActiveEnemies=true;
                        }
                    }
                }
            }
            if(bActiveEnemies) return;
            if(!bOneBeenCalled) return;

            for(i=0;i<15;i+=1)
            {
                rPlayer = GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    rPlayer.Victory();
                }
            }
        }else//-----
        {
            for(i=0;i<15;i+=1)
            {
                rPlayer = GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                    if (iCountBuilding>0)
                    {
                        iAlivePlayers=iAlivePlayers+1;
                        rLastPlayer=rPlayer;
                    }
                    else
                    {
                        rPlayer.Defeat();
                    }
                }
            }
            if (iAlivePlayers==1 && rLastPlayer==null)
            {
                rLastPlayer.Victory();
            }
        }
    }

    event Timer1()
    {
        int i;
        int minLeft;
        player rPlayer;
        bCheckBuilding=true;
        if(nTimeLimit)
        {
            minLeft=(nTimeLimit - GetMissionTime())/1200;
            if(minLeft<0)minLeft=0;
            SelConsoleText("translateScriptTimeLeft",minLeft);
            if(minLeft<1)
            {
                for(i=0;i<15;i=i+1)
                {
                    rPlayer=GetPlayer(i);
                    if(rPlayer!=null && rPlayer.IsAlive())
                    {
                        rPlayer.Defeat();
                        KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
                    }
                }
                AddBriefing("translateCampaignAccomplishedDemo");
                nTimeLimit=0;
            }
        }
    }

    command Initialize()
    {
        comboMoney=0;
        comboResources=1;
        comboStartingUnits=1;
        comboAlliedVictory=1;
        comboUnitsLimit=2;
    }

    command Uninitialize()
    {
        ResourcesPerContainer(8);
    }

    command Combo1(int nMode) button comboMoney
    {
        comboMoney = nMode;
    }

    command Combo2(int nMode) button comboTime
    {
        comboTime = nMode;
    }

    command Combo3(int nMode) button comboResources
    {
        comboResources = nMode;
    }

    command Combo4(int nMode) button comboStartingUnits
    {
        comboStartingUnits = nMode;
    }

    command Combo5(int nMode) button comboUnitsLimit
    {
        comboUnitsLimit = nMode;
    }

```

```

        }

        command Combo6(int nMode) button comboAlliedVictory
        {
            comboAlliedVictory = nMode;
        }
    }

UncleSam.ec
mission "translateGameTypeUncleSam"
{
    int nCashRate;
    int nCashTime;

    int bGameEnded;

    enum comboMoney
    {
        "translateScript10000CR",
        "translateScript15000CR",
        "translateScript20000CR",
        "translateScript30000CR",
        "translateScript40000CR",
        "translateScript50000CR",
    multi:
        "translateGameMenuStartingMoney"
    }

    enum comboCashRate
    {
        "translateScript1000CR",
        "translateScript12500CR",
        "translateScript15000CR",
        "translateScript10000CR",
        "translateScript15000CR",
        "translateScript20000CR",
    multi:
        "translateGameMenuCashRate"
    }

    enum comboCashTime
    {
        "translateGameMenuRateFrequency1min",
        "translateGameMenuRateFrequency3min",
        "translateGameMenuRateFrequency5min",
        "translateGameMenuRateFrequency10min",
        "translateGameMenuRateFrequency15min",
    }

    multi:
        "translateGameMenuRateFrequency"
    }

    enum comboStartingUnits
    {
        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
    multi:
        "translateGameMenuStartingUnits"
    }

    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit10000CR",
        "translateGameMenuUnitsLimit20000CR",
        "translateGameMenuUnitsLimit30000CR",
        "translateGameMenuUnitsLimit50000CR",
    multi:
        "translateGameMenuUnitsLimit"
    }

    enum comboAlliedVictory
    {
        "translateGameMenuAlliedVictoryNo",
        "translateGameMenuAlliedVictoryYes",
    multi:
        "translateGameMenuAlliedVictory"
    }

    state Initialize;
    state Nothing;
    state Initialize
    {
}
}

player rPlayer;
int i;
int nStartingMoney;

if(comboMoney==0)nStartingMoney=10000;
if(comboMoney==1)nStartingMoney=15000;
if(comboMoney==2)nStartingMoney=20000;
if(comboMoney==3)nStartingMoney=30000;
if(comboMoney==4)nStartingMoney=40000;
if(comboMoney==5)nStartingMoney=50000;

if(comboCashRate==0)nCashRate = 1000;
if(comboCashRate==1)nCashRate = 2500;
if(comboCashRate==2)nCashRate = 5000;
if(comboCashRate==3)nCashRate = 10000;
if(comboCashRate==4)nCashRate = 15000;
if(comboCashRate==5)nCashRate = 20000;

if(comboCashTime==0)nCashTime=1*60*20;
if(comboCashTime==1)nCashTime=3*60*20;
if(comboCashTime==2)nCashTime=5*60*20;
if(comboCashTime==3)nCashTime=10*60*20;
if(comboCashTime==4)nCashTime=15*60*20;

bGameEnded=false;
for(i=0;j<15;i=j+1)
{
    rPlayer=GetPlayer(i);

    if(rPlayer!=null)
    {
        if(comboAlliedVictory)
            rPlayer.EnableAIFeatures2(ai2BNSendResult,false); //nie wysylac
    rezultatow do EARTH NETu

    rPlayer.SetMoney(nStartingMoney);

rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMin
ngUnits,false);

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);

if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0)
;

rPlayer.EnableBuilding("EDBMR",false);
rPlayer.EnableBuilding("EDBRE",false);
rPlayer.EnableBuilding("UCSBR",false);
rPlayer.EnableBuilding("LCBMR",false);

rPlayer.EnableResearch("RES_UCS_UOH2",false);
rPlayer.EnableResearch("RES_UCS_UOH3",false);
rPlayer.EnableResearch("RES_UCS_UAH1",false);
rPlayer.EnableResearch("RES_UCS_UAH2",false);
rPlayer.EnableResearch("RES_UCS_UAH3",false);

}

SetTimer(0,100);
SetTimer(1,nCashTime);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    true;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
}

```

Units

AdvancedTank.ec

tank "translateScriptNameTankAdvanced"
 {

{
connect

```
    consts  
    {  
        disableFire=0;  
        enableFire=1;  
    }
```

```
unit m_uTarget;
int m_nCannonsCount;
int m_nTargetGx;
int m_nTargetGy;
int m_nTargetLz;
int m_nStayGx;
int m_nStayGy;
int m_nStayLz;
int m_nSpecialGx; //Patrol point , escort
int m_nSpecialGy;
int m_nSpecialLz;
```

```

int m_nSpecialCounter;
int m_nState;
int bFirstShoot;
unit m_uSpecialUnit;

enum lights
{
    "translateCommandStateLightsAUTO",
    "translateCommandStateLightsON",
    "translateCommandStateLightsOFF",
multi:
    "translateCommandStateLightsMode"
}

enum movementMode
{
    "translateCommandStateFollowEnemy",
    "translateCommandStateHoldArea",
    "translateCommandStateHoldPosition",
multi:
    "translateCommandStateMovement"
}

enum attackMode
{
    "translateCommandStateFireAtWill",
    "translateCommandStateReturnfire",
    "translateCommandStateHoldFire",
multi:
    "translateCommandStateFireMode"
}

enum searchMode
{
    "translateCommandStateNearestTarget",
    "translateCommandStateWeakestTarget",
multi:
    "translateCommandStateTargeting"
}

enum retreatMode
{
    "translateCommandStateNoRetreat",
    "translateCommandStateRetreatAmmo",
    "translateCommandStateRetreatHP50",
    "translateCommandStateRetreatHP25",
multi:
    "translateCommandStateRetreatMode"
}

enum traceMode
{
    "translateCommandStateTraceOFF",
    "translateCommandStateTraceON",
multi:
    "translateCommandStateTraceMode"
}
//***** * F U N C T I O N S *****
//*****
function int SetTarget(unit uTarget)
{
    m_uTarget = uTarget;
    SetTargetObject(uTarget);
    return true;
}
-----
function int GoToPoint()
{
    int nRangeMode;
    int nDx;
    int nDy;
    nRangeMode = IsPointInCannonRange(0,m_nTargetGx, m_nTargetGy,
m_nTargetLz);

    if(nRangeMode == 0) //poza zasiegu
    {
        CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
        return true;
    }
    if (nRangeMode == 4) //in range
    {
        if (IsMoving())
            CallStopMoving();
        return false;
    }
}
}
if(nRangeMode == 1) //w zasiegu ale trzeba odwrocic czolg
{
    if (IsMoving())
        CallStopMoving();
    else
        CallTurnToAngle(GetCannonAngleToPoint(0,m_nTargetGx,
m_nTargetGy, m_nTargetLz));
}
return true;
}
CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
return true;
}
-----
function int GoToTarget()
{
    int nRangeMode;
    int nDx;
    int nDy;
    int nDistance;
    nRangeMode = IsTargetInCannonRange(0, m_uTarget);
    nDistance =
DistanceTo(m_uTarget.GetLocationX(),m_uTarget.GetLocationY());
    if(nRangeMode == notInRange)
    {
        CallMoveToPoint(m_uTarget.GetLocationX(), m_uTarget.GetLocationY(),
m_uTarget.GetLocationZ());
        return true;
    }
    if (nRangeMode == inRangeGoodHit)
    {
        if (IsMoving())
        {
            CallStopMoving();
        }
        return false;
    }
    if(nRangeMode == inRangeBadAngleAlpha) // w zasiegu ale trzeba odwro-
cic czolg
    {
        if (IsMoving())
        {
            CallStopMoving();
        }
        else
        {
            CallTurnToAngle(GetCannonAngleToTarget(0,m_uTarget));
        }
        return true;
    }
    if(nRangeMode == inRangeBadAngleBeta)//w zasiegu strzalu ale zly kat
beta
    {
        if(nDistance<3)//odsunac sie
        {
            CallMoveToPoint(2*m_uTarget.GetLocationX()-
m_uTarget.GetLocationY(),2*m_uTarget.GetLocationZ());
        }
        else
        {
            //jazda do punktu o 90stopni
            m_nTargetGx = m_uTarget.GetLocationX();
            m_nTargetGy = m_uTarget.GetLocationY();
            m_nTargetLz = m_uTarget.GetLocationZ();
            nDx = m_nTargetGx - m_nTargetGy + GetLocationY();
            nDy = m_nTargetGy + m_nTargetGx - GetLocationX();
            CallMoveOneField(nDx, nDy, m_nTargetLz);
        }
        return true;
    }
    //w zasiegu ale cos zaslania
    if(nDistance<3)//odsunac sie
    {
        CallMoveToPoint(2*m_uTarget.GetLocationX()-
m_uTarget.GetLocationY(),2*m_uTarget.GetLocationZ());
    }
    else

```

```

    {
        CallMoveToPoint(m_uTarget.GetLocationX(), m_uTarget.GetLocationY(),
m_uTarget.GetLocationZ());
    }
    return true;
}

//-----
function int TargetInRange()
{
    int nRangeMode;
    int nDx;
    int nDy;
    nRangeMode = IsTargetInCannonRange(0, m_uTarget);

    if (nRangeMode == inRangeGoodHit)
    {
        return true;
    }
    if(nRangeMode == inRangeBadAngleAlpha) //w zasiegu ale trzeba odwro-
cic czolg
    {
        if (IsMoving())
        {
            CallStopMoving();
        }
        else
        {
            CallTurnToAngle(GetCannonAngleToTarget(0,m_uTarget));
        }
        return true;
    }
    return false;
}
//-----
function int EndState()
{
    NextCommand(1);
}
//-----
function int FindBestTarget()
{
    int i;
    int nTargetsCount;
    unit newTarget;

    if(GetCannonType(0) != cannonTypeelon)
    {
        if(!CanCannonFireToAircraft(-1))
        {
            if(searchMode==1)//find weakest target
SetTarget(FindTarget(findTargetWaterUnit | findTargetNormalUnit | findTargetBuildin
gUnit, findEnemyUnit, findNearestUnit, findDestinationAnyUnit));
            else //find closest target
SetTarget(FindClosestEnemyUnitOrBuilding(findTargetWaterUnit |
findTargetNormalUnit));
        }
        else
        {
            if(searchMode==1)//find weakest target
SetTarget(FindTarget(findTargetAnyUnit, findEnemyUnit,
findWeakestUnit, findDestinationAnyUnit));
            else //find closest target
SetTarget(FindClosestEnemyUnitOrBuilding(findTargetWaterUnit |
findTargetNormalUnit | findTargetFlyingUnit));
        }
        if(m_uTarget!=null &&
movementMode==2 &&
(IsTargetInCannonRange(0, m_uTarget)!=inRangeGoodHit))
        {
            if(!CanCannonFireToAircraft(-1))

SetTarget(FindTarget(findTargetWaterUnit | findTargetNormalUnit | findTargetBuildin
gUnit, findEnemyUnit, findNearestUnit, findDestinationAnyUnit));
            else
SetTarget(FindTarget(findTargetAnyUnit, findEnemyUnit,
findNearestUnit, findDestinationAnyUnit));
        }
        return true;
    }
    SetTarget(null);
BuildTargetsArray(findTargetWaterUnit | findTargetNormalUnit | findTargetBuildingUn
it, findEnemyUnit, findDestinationAnyUnit);
SortFoundTargetsArray();
nTargetsCount=GetTargetsCount();
if(nTargetsCount!=0)
{
    StartEnumTargetsArray();
    for(i=0;i<nTargetsCount;j=i+1)
    {
        newTarget = GetNextTarget();
        if(!newTarget.IsEnabled())
        {
            EndEnumTargetsArray();
            SetTarget(newTarget);
            return true;
        }
    }
    EndEnumTargetsArray();
    return false;
}
else
{
    return false;
}
}
//********************************************************************* STATES *****
state Nothing;
state HoldPosition;
state StartMoving;
state Moving;
state AutoAttacking;
state Attacking;
state AttackingPoint;
state Patrol;
state Escort;
state InPlatoonState;
state Retreat;
//-----
state Retreat
{
    if(IsAllowingWithdraw())AllowScriptWithdraw(true);
    if(m_nSpecialCounter)
    {
        SetTargetObject(null);
        m_uTarget = FindClosestEnemy();
        m_nSpecialCounter = (m_nSpecialCounter+1)%16;
    }
    else
    {
        if(IsMoving())
            m_nSpecialCounter = (m_nSpecialCounter+1)%16;
        else
            m_nSpecialCounter = 0;
        if(traceMode)TraceD("R
return Retreat;
    }
    if(m_uTarget)
    {
        if(IsMoving())
            CallStopMoving();
        m_uTarget = null;
        SetTargetObject(null);
        if(traceMode)TraceD("R->N
return Nothing;
    }
    CallMoveToPoint(2*GetLocationX()-
m_uTarget.GetLocationX(),2*GetLocationY() -
m_uTarget.GetLocationY(),GetLocationZ());
    if(traceMode)TraceD("R u
return Retreat;
}

//-----
state InPlatoonState
{
    if(IsAllowingWithdraw())AllowScriptWithdraw(true);
    if(traceMode) TraceD("P\n");
    if(!InPlatoon())
    {
        SetLightsMode(lights);
        SetCannonFireMode(-1, disableFire);
        return Nothing;
    }
    return InPlatoonState;
}

```

```

    }-----\n
state Nothing
{
    if(traceMode)TraceD("N\n");
    if(!IsAllowingWithdraw())AllowScriptWithdraw(true);
    if(lnPlatoon())
    {
        SetCannonFireMode(-1, enableFire);
        return lnPlatoonState;
    }
    if(GetCannonType(0)==cannonTypeBallisticRocket) return Nothing;
    if(movementMode==2) return HoldPosition;
    if(attackMode==2)//hold fire
        return Nothing;
    if (attackMode == 0)//Fire at will
        FindBestTarget();
    if(!lm_uTarget)
    {
        SetTarget(GetAttacker());
        ClearAttacker();
    }
    if (m_uTarget != null)
    {
        m_nStayGx = GetLocationX();
        m_nStayGy = GetLocationY();
        m_nStayLz = GetLocationZ();
        return AutoAttacking;
    }
    return Nothing;
}
//-----\n
state HoldPosition
{
    if(traceMode)TraceD("HP\n");
    if(!IsAllowingWithdraw())AllowScriptWithdraw(false);
    if(movementMode!=2) return Nothing;
    if(m_uTarget)
    {
        if(lm_uTarget.IsLive() || !IsEnemy(m_uTarget) || (GetCannonType(0)
= cannonTypePelon && m_uTarget.IsEnabled()))
        {
            StopCannonFire(-1);
            SetTarget(null);
        }
        else
        {
            if(TargetInRange())
            {
                if(traceMode)TraceD("HP Fire!!!\n");
                CannonFireToTarget(-2, m_uTarget, -1);
            }
            else
                SetTarget(null);
        }
    }
    else
    {
        if(attackMode==2)//hold fire
            return HoldPosition;
        if (attackMode == 0)//Fire at will
            FindBestTarget();
        if(!lm_uTarget)
        {
            SetTarget(GetAttacker());
            ClearAttacker();
        }
    }
    return HoldPosition;
}
//-----\n
state AutoAttacking
{
    int nDistance;
    if(traceMode)TraceD("AA\n");
    // pozostawaj w okolicach punktu
    nDistance =
    Distance(m_nStayGx,m_nStayGy,GetLocationX(),GetLocationY());
    if(movementMode==1 && nDistance > 8)
    {
        if(traceMode)TraceD("nDistance: > 12 !!!!!\n");
        SetTarget(null);
        CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
        if(attackMode==0) SetCannonFireMode(-1, enableFire);
        return StartMoving;
    }
    if(!lm_uTarget.IsLive() || !IsEnemy(m_uTarget) || (GetCannonType(0) ==
cannonTypePelon && m_uTarget.IsEnabled()))
    {
        StopCannonFire(-1);
        SetTarget(null);
    }
    if (m_uTarget)
    {
        if(!GoToTarget())
        {
            if(traceMode)TraceD("AA Fire!!!!\n");
            CannonFireToTarget(-2, m_uTarget, -1);
            return AutoAttacking;
        }
        return AutoAttacking,2;
    }
    else//target not exist
    {
        if(attackMode==0)
            FindBestTarget();
        if(lm_uTarget && attackMode==1)
        {
            SetTarget(GetAttacker());
            ClearAttacker();
        }
        if (m_uTarget != null)
            return AutoAttacking;
        if (nDistance > 0 && movementMode==1)
        {
            CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
            return StartMoving;
        }
        if (IsMoving())
        {
            CallStopMoving();
        }
        return Nothing;
    }
}
//-----\n
state AttackingPoint
{
    if(traceMode)TraceD("AG\n");
    if(GoToPoint())
    {
        return AttackingPoint;
    }
    else
    {
        CannonFireGround(-1, m_nTargetGx, m_nTargetGy, m_nTargetLz, 1);
        return AttackingPoint;
    }
}
//-----\n
state Attacking
{
    if(traceMode)TraceD("A\n");
    if (m_uTarget.IsLive() &&
    (GetCannonType(0) != cannonTypePelon || lm_uTarget.IsEnabled() ||
bFirstShoot))
    {
        if(GoToTarget())
        {
            return Attacking,2;
        }
        else
    }
}

```

```

        bFirstShoot=false;
        CannonFireToTarget(-1, m_uTarget, -1);
        return Attacking;
    }
}
else //target not exist or is disabled
{
    SetTarget(null);
    StopCannonFire(-1);
    if (!IsMoving())
    {
        CallStopMoving();
    }
    EndState();
    return Nothing;
}

//-----
state StartMoving
{
    if(attackMode==0) SetCannonFireMode(-1, enableFire);
    return Moving, 20;
}
//-----
state Moving
{
    if (IsMoving())
    {
        if(GetAttacker()!=null && attackMode==1)
        {
            SetTarget(GetAttacker());
            CannonFireToTarget(-2, m_uTarget, -1);
            ClearAttacker();
            m_uTarget=null;
        }
        if(traceMode) TracedD("Moving\n");
        return Moving;
    }
    else
    {
        if(traceMode) TraceD("Moving -> N\n");
        EndState();
        return Nothing;
    }
}
//-----
state Patrol
{
    if(m_uTarget)
    {
        if(!m_uTarget.IsLive() || IsEnemy(m_uTarget) || (GetCannonType(0)
== cannonTypePelon && m_uTarget.IsEnabled())))
        {
            StopCannonFire(-1);
            SetTarget(null);
        }
        else
        {
            if(GoToTarget())
            {
                return Patrol,2;
            }
            else
            {
                CannonFireToTarget(-2, m_uTarget, -1);
                return Patrol;
            }
        }
    }
    else
    FindBestTarget();
    if (!IsMoving())
    {
        if(m_nSpecialCounter == 1)
        {
            CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
            m_nSpecialCounter = 0;
        }
        else
        {
            CallMoveToPoint(m_nSpecialGx, m_nSpecialGy, m_nSpecialLz);
            m_nSpecialCounter = 1;
        }
    }
}
return Patrol;
}
//-----
state Escort
{
    if((traceMode)TraceD("Escort\n"))
    {
        if(m_uTarget)
        {
            if(!m_uTarget.IsLive() || !IsEnemy(m_uTarget) || (GetCannonType(0)
== cannonTypePelon && m_uTarget.IsEnabled())))
            {
                StopCannonFire(-1);
                SetTarget(null);
            }
            else
            {
                if(GoToTarget())
                {
                    return Escort,2;
                }
                else
                {
                    CannonFireToTarget(-2, m_uTarget, -1);
                    return Escort;
                }
            }
        }
        else
        FindBestTarget();
        if(!m_uSpecialUnit.IsLive())
        {
            m_uSpecialUnit=null;
            if(IsMoving())
            {
                CallStopMoving();
                return Escort;
            }
            EndState();
            return Nothing;
        }
        m_nTargetGx = m_uSpecialUnit.GetLocationX() + m_nSpecialGx;
        m_nTargetGy = m_uSpecialUnit.GetLocationY() + m_nSpecialGy;
        m_nTargetLz = m_uSpecialUnit.GetLocationZ();
        if(Distance(m_nTargetGx, m_nTargetGy, GetLocationX(), GetLocationY()) > 0)
        {
            if(traceMode) TraceD("Escort: updating position\n");
            CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
            return Escort;
        }
        else
        {
            if(!IsMoving())
            {
                CallStopMoving();
                return Escort;
            }
            return Escort;
        }
    }
}
//-----
state Frozen
{
    if (IsFrozen() || IsMoving())
    {
        state Frozen;
    }
    else
    {
        //!!wrocic do tego co robimy
        if(m_nState==1)
        return AttackingPoint;

        if(m_nState==2)
        return Attacking;

        if(m_nState==3)
        {
            if(attackMode==0) SetCannonFireMode(-1, enableFire);
            CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
            return StartMoving;
        }
    }
}

```

```

        }

        if(m_nState==4)
            return Patrol;

        if(m_nState==5)
            return Escort;

        if(m_nState==6)
            return InPlatoonState;

        state Nothing;
    }

//***** E V E N T S *****
//***** zwrocaj true
//false jak nie ma
event OnHit()
{
    if (attackMode == 1 && m_uTarget==null)//Return fire
    {
        SetTarget(GetAttacker());
        CannonFireToTarget(-2, m_uTarget, -1);
    }
    if(retreatMode == 2 || retreatMode == 3)
    {
        if((GetHP()*4)<GetMaxHP())
        {
            m_nSpecialCounter=0;
            state Retreat;
        }
        if(retreatMode == 2 &&((GetHP()*2)<=GetMaxHP()))
        {
            m_nSpecialCounter=0;
            state Retreat;
        }
    }
    true;
}

event OnCannonLowAmmo(int nCannonNum)
{
    SendSupplyRequest();
    true;
}

event OnCannonNoAmmo(int nCannonNum)
{
    if(retreatMode == 1)
    {
        m_nSpecialCounter=0;
        state Retreat;
    }
    true;
}

event OnCannonFoundTarget(int nCannonNum, unit uTarget)
{
    if(GetCannonType(nCannonNum) == cannonTypelon)
    {
        if(uTarget.IsEnabled())
        {
            return true;
        }
    }
    if(attackMode!=0)//hold fire | return fire
        return true;
    return false;//gdyby zwrocic true do dzialka nie strzeli
}

event OnCannonEndFire(int nCannonNum, int nEndStatus)//gdy zniszczony,
poza zasięgiem lub brak amunicji
{
    false;
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    if(state==Frozen) m_nState = 0;
    if(state==AttackingPoint)
        m_nState=1;
    if(state==Attacking)
        m_nState=2;
}

if((state==Moving) || (state==StartMoving))
    m_nState=3;
if(state==Patrol)
    m_nState=4;
if(state==Escort)
    m_nState=5;
if(state==InPlatoonState)
    m_nState=6;
CallFreeze(nFreezeTicks);
state Frozen;
true;
}

event OnTransportedToWorld()
{
    StopCannonFire(-1);
    SetCannonFireMode(-1, disableFire);
    SetTarget(null);
    m_uSpecialUnit = null;
}

event OnConvertedToWorld()
{
    StopCannonFire(-1);
    SetCannonFireMode(-1, disableFire);
    SetTarget(null);
    m_uSpecialUnit = null;
    ClearAttacker();
    state Nothing;
}

//***** C O M M A N D S *****
command Initialize()
{
    m_nCannonsCount = GetCannonsCount();
    traceMode = 0;
    movementMode=0;
    SetCannonFireMode(-1, disableFire);
}

command Uninitialize()
{
    //wykasowac referencje
    SetTarget(null);
    m_uSpecialUnit = null;
}

command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        assert(nMode == 0);
        lights = nMode;
    }
    SetLightsMode(lights);
    NextCommand(0);
}

command SetMovementMode(int nMode) button movementMode description
"translateCommandStateMovementDescription" priority 205
{
    if (nMode == -1)
    {
        movementMode = (movementMode + 1) % 3;
    }
    else
    {
        assert(nMode == 0);
        movementMode = nMode;
    }
    NextCommand(0);
}

command SetAttackMode(int nMode) button attackMode description
"translateCommandStateFireModeDescription" hotkey priority 6
{
    if (nMode == -1)
    {
        attackMode = (attackMode + 1) % 3;
    }
}

```

```

        }
    else
    {
        assert(nMode == 0);
        attackMode = nMode;
    }
    if(attackModel==0)
    {
        SetCannonFireMode(-1, disableFire);
        StopCannonFire(-1);
    }
}

//-----
command UserOneParam1(int nMode) button searchMode description
"translateCommandStateTargetingDescription" priority 8
{
    if (nMode == -1)
    {
        searchMode = (searchMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        searchMode = nMode;
    }
}

//-----
command UserOneParam2(int nMode) button retreatMode description
"translateCommandStateRetreatModeDescription" priority 12
{
    if (nMode == -1)
    {
        retreatMode = (retreatMode + 1) % 4;
    }
    else
    {
        assert(nMode == 0);
        retreatMode = nMode;
    }
}

//-----
command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    SetTarget(null);
    StopCannonFire(-1);
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    if(IsMoving())
        CallStopMoving();
    SetCannonFireMode(-1, disableFire);
    state Nothing;
}

//-----
command HoldPosition() hidden button "translateCommandHoldPosition" description
"translateCommandStopDescription" hotkey priority 20
{
    SetTarget(null);
    StopCannonFire(-1);
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    if(IsMoving())
        CallStopMoving();
    movementMode=2;
    SetCannonFireMode(-1, disableFire);
    ChangedCommandValue();
    state Nothing;
}

//-----
command Move(int nGx, int nGy, int nLz) button "translateCommandMove" description
"translateCommandMoveDescription" hotkey priority 21
{
    SetTarget(null);
    m_nStayGx = nGx;
    m_nStayGy = nGy;
    m_nStayLz = nLz;
    if(attackMode==0) SetCannonFireMode(-1, enableFire);
    CallMoveToPoint(nGx, nGy, nLz);
    state StartMoving;
}

//-----
command Attack(unit uTarget) button "translateCommandAttack" description
"translateCommandAttackDescription" hotkey priority 22
{
    SetTarget(uTarget);
    bFirstShoot=true;
    if(attackMode==0) SetCannonFireMode(-1, enableFire);
    AllowScriptWithdraw(true);
    if(state==Froozen)
    {
        m_nState = 2;
    }
    else
    {
        state Attacking;
    }
}

/*komenda nie wystawiana na zewnatrz*/
command AttackOnPoint(int nX, int nY, int nZ) hidden button
"translateCommandAttack"
{
    SetTarget(null);
    m_nTargetGx = nX;
    m_nTargetGy = nY;
    m_nTargetLz = nZ;
    AllowScriptWithdraw(true);
    SetCannonFireMode(-1, disableFire);
    if(state==Froozen)
    {
        m_nState = 1;
    }
    else
    {
        state AttackingPoint;
    }
}

//-----
command UserNoParam0() button "translateCommandRetreat" description
"translateCommandRetreatDescription" hotkey priority 25
{
    m_nSpecialCounter=0;
    state Retreat;
}

//-----
command SendSupplyRequest() button "translateCommandSupply" description
"translateCommandSupplyDescription" hotkey priority 27
{
    SendSupplyRequest();
}

//-----
command Patrol(int nGx, int nGy, int nLz) button "translateCommandPatrol" description
"translateCommandPatrolDescription" hotkey priority 29
{
    m_nSpecialGx = nGx;
    m_nSpecialGy = nGy;
    m_nSpecialLz = nLz;
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    CallMoveToPoint(nGx, nGy, nLz);
    m_nSpecialCounter = 1;
    SetCannonFireMode(-1, disableFire);
    state Patrol;
}

//-----
command Escort(unit uUnit) button "translateCommandEscort" description
"translateCommandEscortDescription" hotkey priority 31
{
    m_uSpecialUnit=uUnit;
    m_nSpecialGx=GetLocationX()-m_uSpecialUnit.GetLocationX();
    m_nSpecialGy=GetLocationY()-m_uSpecialUnit.GetLocationY();
    if(m_nSpecialGx > 2) m_nSpecialGx=2;
    if(m_nSpecialGx < -2) m_nSpecialGx=-2;
    if(m_nSpecialGy > 2) m_nSpecialGy=2;
    if(m_nSpecialGy < -2) m_nSpecialGy=-2;
    state Escort;
}

//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    CallMoveInsideObject(uEntrance);
    m_nTargetGx = GetEntranceX(uEntrance);
}

```

```

        m_nTargetGy = GetEntranceY(uEntrance);
        m_nTargetLz = GetEntranceZ(uEntrance);
        SetCannonFireMode(-1, disableFire);
        state StartMoving;
    }

//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
//-----
/* command UserOneParam9(int nMode) button traceMode priority 255
if (nMode == -1)
{
    traceMode = (traceMode + 1) % 2;
}
else
{
    assert(nMode == 0);
    traceMode = nMode;
}
*/
//-----

}

state StartMoving
{
    return Moving, 40;
}
//-----
state Moving
{
    if (IsMoving())
    {
        return Moving;
    }
    else
    {
        NextCommand(1);
        return Nothing;
    }
}
state MovingToBuildBuilding
{
    if (IsMoving())
    {
        if(traceMode) TraceD("BuildRobot->IsMoving\n");
        return MovingToBuildBuilding;
    }
    else
    {
        if (IsInGoodPointForCurrentBuild())
        {
            if(traceMode) TraceD("BuildRobot->CallBuildBuilding\n");
            CallBuildBuilding();
            return BuildBuilding;
        }
        else
        {
            //!!jakis licznik?
            m_nMoveToX = GetCurrentBuildLocationX();
            m_nMoveToY = GetCurrentBuildLocationY();
            m_nMoveToZ = GetCurrentBuildLocationZ();
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return MovingToBuildBuilding;
        }
    }
}
state MovingToBuildElement
{
    if (IsMoving())
    {
        return MovingToBuildElement;
    }
    else
    {
        if (IsInGoodPointForCurrentBuild())
        {
            CallBuildCurrentElement();
            return BuildElement;
        }
        else
        {
            //!!jakis licznik?
            m_nMoveToX = GetCurrentBuildLocationX();
            m_nMoveToY = GetCurrentBuildLocationY();
            m_nMoveToZ = GetCurrentBuildLocationZ();
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return MovingToBuildElement;
        }
    }
}
state BuildBuilding
{
    if (IsBuildWorking())
    {
        return BuildBuilding;
    }
    else
    {
        NextCommand(1); //!!1 czy 0?
        return Nothing;
    }
}
state BuildElement

```

Builder.ec

builder "translateScriptNameConstructionVechicle"

```
{
    consts
    {
        buildNone = 0;
        buildBuilding = 1;
        buildWall = 2;
        buildTrench = 3;
        buildFlatTerrain = 4;
        buildWideBridge = 5;
        buildNarrowBridge = 6;
        buildWideTunnel = 7;
        buildNarrowTunnel = 8;
    }
}
```

```
int m_nMoveToX;
int m_nMoveToY;
int m_nMoveToZ;
```

enum lights

```
{
    "translateCommandStateLightsAUTO",
    "translateCommandStateLightsON",
    "translateCommandStateLightsOFF",
    "translateCommandStateLightsMode"
}
```

enum traceMode

```
{
    "translateCommandStateTraceOFF",
    "translateCommandStateTraceON",
    "translateCommandStateTraceMode"
}
```

```
state Initialize;
state Nothing;
state StartMoving;
state Moving;
state MovingToBuildBuilding;
state MovingToBuildElement;
state BuildBuilding;
state BuildElement;
```

```
state Initialize
{
    return Nothing;
}
```

```
state Nothing
{
    return Nothing;
}
```

```

    {
        if (IsBuildWorking())
        {
            return BuildElement;
        }
        else
        {
            if (NextElementPoint())
            {
                m_nMoveToX = GetCurrentBuildLocationX();
                m_nMoveToY = GetCurrentBuildLocationY();
                m_nMoveToZ = GetCurrentBuildLocationZ();
                CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
                return MovingToBuildElement;
            }
            else
            {
                NextCommand(1); //!!1 czy 0?
                return Nothing;
            }
        }
    }
}

//-----
state Froozen
{
    if (IsFroozen())
    {
        state Froozen;
    }
    else
    {
        //!!wrocic do tego co robilismy
        state Nothing;
    }
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    //CallFreeze(nFreezeTicks);
    //state Froozen;
    true;
}

//-----
command Initialize()
{
    false;
}

command Uninitialize()
{
    //wykasowac referencje
    false;
}

//bez nazwy - wywoływanie po wybraniu miejsca
command BuildBuilding(int nX, int nY, int nZ, int nAlpha, int nID) hidden button
"translateCommandBuilding"
{
    SetBuildTypeBuildBuilding(nX, nY, nZ, nAlpha, nID);
    m_nMoveToX = GetCurrentBuildLocationX();
    m_nMoveToY = GetCurrentBuildLocationY();
    m_nMoveToZ = GetCurrentBuildLocationZ();
    CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    state MovingToBuildBuilding;
    true;
}

command BuildTrench(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2) button
"translateCommandTrench" hotkey
{
    SetBuildTypeBuildElements(buildTrench);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

command BuildFlatTerrain(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
button "translateCommandPlatten" hotkey
{
    SetBuildTypeBuildElements(buildFlatTerrain);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

command BuildWall(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2) button
"translateCommandWall" hotkey
{
    SetBuildTypeBuildElements(buildWall);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

command BuildWideBridge(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2) button
"translateCommandBridge2" hotkey
{
    SetBuildTypeBuildElements(buildWideBridge);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

command BuildNarrowBridge(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2) button
"translateCommandBridge1" hotkey
{
    SetBuildTypeBuildElements(buildNarrowBridge);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

command BuildWideTunnel(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2) button
"translateCommandTunnel2" hotkey
{
    SetBuildTypeBuildElements(buildWideTunnel);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

command BuildNarrowTunnel(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2) button
"translateCommandTunnel1" hotkey
{
    SetBuildTypeBuildElements(buildNarrowTunnel);
    if (AddElementsLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMoveToX = GetCurrentBuildLocationX();
        m_nMoveToY = GetCurrentBuildLocationY();
        m_nMoveToZ = GetCurrentBuildLocationZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToBuildElement;
    }
    true;
}

```

```

        }

        command SpecialNextAngle() button "translateCommandTurn" description
        "translateCommandTurnDescription" hotkey priority 50
        {
            //implemented in code
        }

        command Move(int nGx, int nGy, int nLz) hidden button
        "translateCommandMove" description "translateCommandMoveDescription" hotkey
        priority 21
        {
            m_nMoveToX = nGx;
            m_nMoveToY = nGy;
            m_nMoveToZ = nLz;
            CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            state StartMoving;
            true;
        }
        //-----
        command SetLights(int nMode) button lights description
        "translateCommandStateLightsModeDescription" hotkey priority 204
        {
            if (nMode == -1)
            {
                lights = (lights + 1) % 3;
            }
            else
            {
                assert(nMode == 0);
                lights = nMode;
            }
            SetLightsMode(lights);
        }
        //-----
        command Enter(unit uEntrance) hidden button "translateCommandEnter"
        {
            m_nMoveToX = GetEntranceX(uEntrance);
            m_nMoveToY = GetEntranceY(uEntrance);
            m_nMoveToZ = GetEntranceZ(uEntrance);
            CallMoveInsideObject(uEntrance);
            state StartMoving;
            true;
        }
        //-----
        command SpecialChangeUnitsScript() button "translateCommandChangeScript"
        description "translateCommandChangeScriptDescription" hotkey priority 254
        {
            //special command - no implementation
        }
        //-----
        /* command UserOneParam9(int nMode) hidden button traceMode priority
        255
        {
            if (nMode == -1)
            {
                traceMode = (traceMode + 1) % 2;
            }
            else
            {
                assert(nMode == 0);
                traceMode = nMode;
            }
        }*/
    }

}

Carrier.ec
carrier "translateScriptNameContainerTransporter"
{
    int m_nMoveToX;
    int m_nMoveToY;
    int m_nMoveToZ;
    int m_nCurrPutGetX;
    int m_nCurrPutGetY;
    int m_nCurrPutGetZ;
    int m_nContainerX;
    int m_nContainerY;
    int m_nContainerZ;
    int m_nGetContainerFrom;//0 - mine, 1 - single container
    int m_nState;

    enum lights
    {
        "translateCommandStateLightsAUTO",
        "translateCommandStateLightsON",
        "translateCommandStateLightsOFF",
    }

    multi: "translateCommandStateLightsMode"
    }

    state Initialize;
    state Nothing;
    state StartMoving;
    state Moving;
    state MovingToContainerSource;
    state MovingToContainerDestination;
    state GettingContainer;
    state PuttingContainer;
    //-----
    state Initialize
    {
        return Nothing;
    }
    //-----
    state Nothing
    {
        return Nothing;
    }
    //-----
    state StartMoving
    {
        return Moving, 20;
    }
    //-----
    state Moving
    {
        if (!IsMoving())
        {
            return Moving;
        }
        else
        {
            NextCommand(1);
            return Nothing;
        }
    }
    //-----
    state MovingToContainerSource
    {
        int nPosX;
        int nPosY;
        int nPosZ;

        if (!IsMoving())
        {
            return MovingToContainerSource;
        }
        else
        {
            nPosX = GetLocationX();
            nPosY = GetLocationY();
            nPosZ = GetLocationZ();
            //sprawdzic czy dojechalismy tam gdzie nalezalo
            if ((nPosX == m_nCurrPutGetX) && (nPosY == m_nCurrPutGetY) &&
                (nPosZ == m_nCurrPutGetZ))
            {
                if (m_nGetContainerFrom == 0)
                {
                    CallGetContainer();
                }
                else
                {
                    assert m_nGetContainerFrom == 1;
                    CallGetSingleContainer(m_nContainerX, m_nContainerY,
                    m_nContainerZ);
                }
                return GettingContainer;
            }
            else
            {
                //kazac mu tam znowu jechac (imoze jakis licznik zeby nie wywolywal
                w kolko)
                CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
                m_nCurrPutGetZ);
                return MovingToContainerSource;
            }
        }
    }
}

```

```

state MovingToContainerDestination
{
    int nPosX;
    int nPosY;
    int nPosZ;

    if (IsMoving())
    {
        return MovingToContainerDestination;
    }
    else
    {

        nPosX = GetLocationX();
        nPosY = GetLocationY();
        nPosZ = GetLocationZ();
        //sprawdzic czy dojechalem tam gdzie nalezalo
        if ((nPosX == m_nCurrPutGetX) & (nPosY == m_nCurrPutGetY) &
(nPosZ == m_nCurrPutGetZ))
        {
            CallPutContainer();
            return PuttingContainer;
        }
        else
        {
            //kazac mu tam znnow jechac (!moze jakis licznik zeby nie wywoylawa
w kolko)
            CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
            return MovingToContainerDestination;
        }
    }
}

state GettingContainer
{
    unit uBuilding;
    unit uContainer;
    if (IsGettingContainer())
    {
        return GettingContainer;
    }
    else
    {
        if (HaveContainer())
        {
            //jesli biezemy pojedynczy kontener to teraz znalezc nastepny w
poblizu
            if (m_nGetContainerFrom == 1)
            {
                uContainer = FindSingleContainer();
                if (uContainer)
                {

                    m_nContainerX = uContainer.GetLocationX();
                    m_nContainerY = uContainer.GetLocationY();
                    m_nContainerZ = uContainer.GetLocationZ();
                    m_nGetContainerFrom = 1;
                }
                else
                {
                    //juz nie ma nastepnego w poblizu
                    m_nGetContainerFrom = 0;
                }
            }
            //pojechac do budynku przeznaczenia jesli jest
            if (GetDestinationBuilding() != null)
            {
                m_nCurrPutGetX = GetDestinationBuildingPutLocationX();
                m_nCurrPutGetY = GetDestinationBuildingPutLocationY();
                m_nCurrPutGetZ = GetDestinationBuildingPutLocationZ();
                CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);

                return MovingToContainerDestination;
            }
            else
            {
                //sprobowac znalezc ten budynek
                uBuilding = FindContainerRefineryBuilding();
                if (uBuilding != null)
                {
                    SetDestinationBuilding(uBuilding);
                    m_nCurrPutGetX = GetDestinationBuildingPutLocationX();
                    m_nCurrPutGetY = GetDestinationBuildingPutLocationY();
                    m_nCurrPutGetZ = GetDestinationBuildingPutLocationZ();
                    CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
                    return MovingToContainerDestination;
                }
            }
        }
        else
        {
            //z jakiegos powodu nie wzielismy kontenera - poszukac nowej kopalni
            lub pojedynczego kontenera
            if (m_nGetContainerFrom == 0)
            {
                uBuilding = GetSourceBuilding(); //aby sie upewnic czy zabity (i
skasowac referencje w kodzie)
                if (uBuilding == null)
                {
                    //najpierw probujemy znalezc kopalnie
                    uBuilding = FindContainerMineBuilding();
                }
                if (uBuilding != null)
                {
                    SetSourceBuilding(uBuilding);
                    m_nCurrPutGetX = GetSourceBuildingTakeLocationX();
                    m_nCurrPutGetY = GetSourceBuildingTakeLocationY();
                    m_nCurrPutGetZ = GetSourceBuildingTakeLocationZ();
                    m_nGetContainerFrom = 0;
                    CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
                    return MovingToContainerSource;
                }
                else
                {
                    //probujemy znalezc wolny kontener
                    uContainer = FindSingleContainer();
                    if (uContainer)
                    {
                        m_nContainerX = uContainer.GetLocationX();
                        m_nContainerY = uContainer.GetLocationY();
                        m_nContainerZ = uContainer.GetLocationZ();
                        m_nCurrPutGetX =
GetContainerTakeLocationX(m_nContainerX, m_nContainerY, m_nContainerZ);
                        m_nCurrPutGetY =
GetContainerTakeLocationY(m_nContainerX, m_nContainerY, m_nContainerZ);
                        m_nCurrPutGetZ =
GetContainerTakeLocationZ(m_nContainerX, m_nContainerY, m_nContainerZ);
                        m_nGetContainerFrom = 1;
                        CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
                        return MovingToContainerSource;
                    }
                    else
                    {
                        return Nothing;
                    }
                }
            }
            else
            {
                assert m_nGetContainerFrom == 1;
                //najpierw probujemy znalezc nastepny wolny kontener
                uContainer = FindSingleContainer();
                if (uContainer)
                {
                    m_nContainerX = uContainer.GetLocationX();
                    m_nContainerY = uContainer.GetLocationY();
                    m_nContainerZ = uContainer.GetLocationZ();
                    m_nCurrPutGetX = GetContainerTakeLocationX(m_nContainerX,
m_nContainerY, m_nContainerZ);
                    m_nCurrPutGetY = GetContainerTakeLocationY(m_nContainerX,
m_nContainerY, m_nContainerZ);
                    m_nCurrPutGetZ = GetContainerTakeLocationZ(m_nContainerX,
m_nContainerY, m_nContainerZ);
                    m_nGetContainerFrom = 1;
                    CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
                    return MovingToContainerSource;
                }
            }
        }
    }
}

```

```

        else
        {
            uBuilding = GetSourceBuilding(); //aby sie upewnic czy zabity (i
skasowac referencje w kodzie)
            if (uBuilding == null)
            {
                //probujemy znalezc kopalnie
                uBuilding = FindContainerMineBuilding();
            }
            if (uBuilding != null)
            {
                SetSourceBuilding(uBuilding);
                m_nCurrPutGetX = GetSourceBuildingTakeLocationX();
                m_nCurrPutGetY = GetSourceBuildingTakeLocationY();
                m_nCurrPutGetZ = GetSourceBuildingTakeLocationZ();
                m_nGetContainerFrom = 0;
                CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);

                return MovingToContainerSource;
            }
            else
            {
                //probujemy znalezc wolny kontener
                return Nothing;
            }
        }
    }
}

//-----
state PuttingContainer
{
    unit uBuilding;
    unit uContainer;
    if (!IsPuttingContainer())
    {
        return PuttingContainer;
    }
    else
    {
        if (!HaveContainer())
        {
            //pojedz do kopalni lub po pojedynczy kontener
            if ((m_nGetContainerFrom == 1) &&
IsContainerInPoint(m_nContainerX, m_nContainerY, m_nContainerZ))
            {
                m_nGetContainerFrom = 1;
                m_nCurrPutGetX = GetContainerTakeLocationX(m_nContainerX,
m_nContainerY, m_nContainerZ);
                m_nCurrPutGetY = GetContainerTakeLocationY(m_nContainerX,
m_nContainerY, m_nContainerZ);
                m_nCurrPutGetZ = GetContainerTakeLocationZ(m_nContainerX,
m_nContainerY, m_nContainerZ);
                CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);

                return MovingToContainerSource;
            }
            else
            {
                if (GetSourceBuilding() != null)
                {
                    m_nCurrPutGetX = GetSourceBuildingTakeLocationX();
                    m_nCurrPutGetY = GetSourceBuildingTakeLocationY();
                    m_nCurrPutGetZ = GetSourceBuildingTakeLocationZ();
                    m_nGetContainerFrom = 0;
                    CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);

                    return MovingToContainerSource;
                }
                else
                {
                    //probujemy znalezc kopalnie
                    uBuilding = FindContainerMineBuilding();
                    if (uBuilding != null)
                    {
                        SetSourceBuilding(uBuilding);
                        m_nCurrPutGetX = GetSourceBuildingTakeLocationX();
                        m_nCurrPutGetY = GetSourceBuildingTakeLocationY();
                        m_nCurrPutGetZ = GetSourceBuildingTakeLocationZ();
                        m_nGetContainerFrom = 0;
                        CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
                    }
                }
            }
        }
    }
}

//-----
state Frozen
{
    if (!IsFrozen())
    {
        state Frozen;
    }
    else
    {
        //!!wrocic do tego co robilismy
        if (m_nState == 1)
        {
            CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
            state MovingToContainerDestination;
        }
        else
        {
            if (m_nState == 2)
            {
                CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
m_nCurrPutGetZ);
                state MovingToContainerSource;
            }
        }
    }
}

```

```

        }
    else
    {
        if(m_nState==3)
        {
            CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            state StartMoving;
        }
        else
            state Nothing;
    }
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    if(state==GettingContainer || state==PuttingContainer)
    {
    }
    else
    {
        m_nState=0;
        if(state==MovingToContainerDestination)
            m_nState=1;
        if(state==MovingToContainerSource)
            m_nState=2;
        if(state==Moving)
            m_nState=3;
        CallFreeze(nFreezeTicks);
        state Frozen;
    }
    true;
}
//-----
command Initialize()
{
    m_nGetContainerFrom = 0;
    //pozwoli dzialkom strzelac samym (o ile sa jakies)
    SetCannonFireMode(-1, 1);
    false;
}
//-----
command Uninitialize()
{
    //wykasowac referencje
    SetDestinationBuilding(null);
    SetSourceBuilding(null);
    false;
}
//-----

/*bez nazwy - wywolywane po kliknięciu kursorem*/
command SetContainerSource(unit uTarget) hidden button
"translateCommandSetMine"
{
    SetSourceBuilding(uTarget);
    m_nGetContainerFrom = 0;
    if (!HaveContainer())
    {
        m_nCurrPutGetX = GetSourceBuildingTakeLocationX();
        m_nCurrPutGetY = GetSourceBuildingTakeLocationY();
        m_nCurrPutGetZ = GetSourceBuildingTakeLocationZ();
        CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
        m_nCurrPutGetZ);

        state MovingToContainerSource;
    }
    else
    {
        if (GetDestinationBuilding() != null)
        {
            m_nCurrPutGetX = GetDestinationBuildingPutLocationX();
            m_nCurrPutGetY = GetDestinationBuildingPutLocationY();
            m_nCurrPutGetZ = GetDestinationBuildingPutLocationZ();
            CallMoveToPointForce(m_nCurrPutGetX, m_nCurrPutGetY,
            m_nCurrPutGetZ);

            state MovingToContainerDestination;
        }
    }
    NextCommand(1);
    true;
}
//-----
command GetSingleContainer(unit uTarget) hidden button
"translateCommandGetContainer"
{
    m_nContainerX = uTarget.GetLocationX();
    m_nContainerY = uTarget.GetLocationY();
    m_nContainerZ = uTarget.GetLocationZ();
    m_nGetContainerFrom = 1;
    if (!HaveContainer())
    {
        m_nCurPutGetX = GetContainerTakeLocationX(m_nContainerX,
        m_nContainerY, m_nContainerZ);
        m_nCurPutGetY = GetContainerTakeLocationY(m_nContainerX,
        m_nContainerY, m_nContainerZ);
        m_nCurPutGetZ = GetContainerTakeLocationZ(m_nContainerX,
        m_nContainerY, m_nContainerZ);
        CallMoveToPointForce(m_nCurPutGetX, m_nCurPutGetY,
        m_nCurPutGetZ);

        state MovingToContainerSource;
    }
    else
    {
        if (GetDestinationBuilding() != null)
        {
            m_nCurPutGetX = GetDestinationBuildingPutLocationX();
            m_nCurPutGetY = GetDestinationBuildingPutLocationY();
            m_nCurPutGetZ = GetDestinationBuildingPutLocationZ();
            CallMoveToPointForce(m_nCurPutGetX, m_nCurPutGetY,
            m_nCurPutGetZ);

            state MovingToContainerDestination;
        }
    }
    NextCommand(1);
    true;
}
//-----
command SetContainerDestination(unit uTarget) hidden button
"translateCommandSetRefinery"
{
    SetDestinationBuilding(uTarget);
    if (HaveContainer())
    {
        m_nCurPutGetX = GetDestinationBuildingPutLocationX();
        m_nCurPutGetY = GetDestinationBuildingPutLocationY();
        m_nCurPutGetZ = GetDestinationBuildingPutLocationZ();
        CallMoveToPointForce(m_nCurPutGetX, m_nCurPutGetY,
        m_nCurPutGetZ);

        state MovingToContainerDestination;
    }
    else
    {
        if (GetSourceBuilding() != null)
        {
            m_nCurPutGetX = GetSourceBuildingTakeLocationX();
            m_nCurPutGetY = GetSourceBuildingTakeLocationY();
            m_nCurPutGetZ = GetSourceBuildingTakeLocationZ();
            m_nGetContainerFrom = 0;
            CallMoveToPointForce(m_nCurPutGetX, m_nCurPutGetY,
            m_nCurPutGetZ);

            state MovingToContainerSource;
        }
    }
    NextCommand(1);
    true;
}
//-----
command Move(int nGx, int nGy, int nLz) hidden button
"translateCommandMove" description "translateCommandMoveDescription" hotkey
{
    m_nMoveToX = nGx;
    m_nMoveToY = nGy;
    m_nMoveToZ = nLz;
    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    state StartMoving;
    true;
}
//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    m_nMoveToX = GetEntranceX(uEntrance);
    m_nMoveToY = GetEntranceY(uEntrance);
    m_nMoveToZ = GetEntranceZ(uEntrance);
}

```

```

    CallMoveInsideObject(uEntrance);
    state StartMoving;
    true;
}

//-----
command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    CallStopMoving();
    state StartMoving;
    true;
}
//-----

command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        assert(nMode == 0);
        lights = nMode;
    }
    SetLightsMode(lights);
}
//-----

command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
//-----

command UserNoParam0() button "Find"
{
unit uContainer;
uContainer = FindSingleContainer();
if (!uContainer)
{
    m_nContainerX = uContainer.GetLocationX();
    m_nContainerY = uContainer.GetLocationY();
    m_nContainerZ = uContainer.GetLocationZ();
    m_nCurrPutGetX = GetContainerTakeLocationX(m_nContainerX,
m_nContainerY, m_nContainerZ);
    m_nCurrPutGetY = GetContainerTakeLocationY(m_nContainerX,
m_nContainerY, m_nContainerZ);
    m_nCurrPutGetZ = GetContainerTakeLocationZ(m_nContainerX,
m_nContainerY, m_nContainerZ);
    m_nGetContainerFrom = 1;
    CallMoveToPointForcel(m_nCurrPutGetX, m_nCurrPutGetY, m_nCurrPutGetZ);

    state MovingToContainerSource;
}
/*
//-----
}

Civil.ec
civil "translateScriptNameUnarmedVechicle"
{
int m_nStay6x;
int m_nStay6y;
int m_nStay6z;
int m_nSpecial6x; //Patrol point , escort
int m_nSpecial6y;
int m_nSpecial6z;
int m_nSpecialCounter;
unit m_uSpecialUnit;

enum lights
{
    "translateCommandStateLightsAUTO",
    "translateCommandStateLightsON",
    "translateCommandStateLightsOFF",
multi:
    "translateCommandStateLightsMode"
}

enum traceMode
{
    "translateCommandStateTraceOFF",
    "translateCommandStateTraceON",
    "translateCommandStateTraceMode"
}
//***** F U N C T I O N S *****
//-----function int EndState()
{
    NextCommand(1);
}

//***** S T A T E S *****
state Retreat;
state Nothing;
state StartMoving;
state Moving;
state Patrol;
state Escort;

//-----state Retreat
{
    unit uTarget;
    if(!m_nSpecialCounter)
    {
        uTarget = FindClosestEnemy();
        m_nSpecialCounter = (m_nSpecialCounter+1)%16;
    }
    else
    {
        if(!IsMoving())
            m_nSpecialCounter = (m_nSpecialCounter+1)%16;
        else
            m_nSpecialCounter = 0;
        if(traceMode)TraceD("R->N
return Retreat;
    }

    if(!uTarget)
    {
        if(!IsMoving())
            CallStopMoving();
        if(traceMode)TraceD("R u
return Nothing;
    }
    CallMoveToPoint(2*GetLocationX()-uTarget.GetLocationX(),2*GetLocationY()-
uTarget.GetLocationY(),GetLocationZ());
    if(traceMode)TraceD("R u
return Retreat;
    }

//-----state Nothing
{
    if(traceMode)TraceD("N
return Nothing;
}

//-----state StartMoving
{
    return Moving, 20;
}

//-----state Moving
{
    if (IsMoving())
    {
        if(traceMode) TraceD("Moving
        return Moving;
    }
    else
    {
        if(traceMode) TraceD("Moving -> N
        EndState();
        return Nothing;
    }
}
//-----
```

```

state Patrol
{
    if (!IsMoving())
    {
        if(m_nSpecialCounter == 1)
        {
            CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
            m_nSpecialCounter = 0;
        }
        else
        {
            CallMoveToPoint(m_nSpecialGx, m_nSpecialGy, m_nSpecialLz);
            m_nSpecialCounter = 1;
        }
    }
    return Patrol;
}
//-----
state Escort
{
    int nTargetGx;
    int nTargetGy;
    if(traceMode)
        TraceD("Escort" + "\n");

    if(!m_uSpecialUnit.IsLive())
    {
        m_uSpecialUnit=null;
        if(IsMoving())
        {
            CallStopMoving;
            return Escort;
        }
        EndState();
        return Nothing;
    }

    nTargetGx = m_uSpecialUnit.GetLocationX() + m_nSpecialGx;
    nTargetGy = m_uSpecialUnit.GetLocationY() + m_nSpecialGy;
    if(Distance(nTargetGx,nTargetGy,GetLocationX(),GetLocationY()) > 0)
    {
        if(traceMode)
            TraceD("Escort: updating position" + "\n");
        CallMoveToPoint(nTargetGx, nTargetGy, m_uSpecialUnit.GetLocationZ());
        return Escort;
    }
    else
    {
        if(IsMoving())
        {
            CallStopMoving;
            return Escort;
        }
        return Escort;
    }
}
//-----
//-----
state Frozen
{
    if (IsFrozen())
    {
        state Frozen;
    }
    else
    {
        //Wyrocic do tego co robilismy
        state Nothing;
    }
}

//***** E V E N T S *****
//zwroca true
//false jak nie ma
event OnHit()
{
    if((GetHP()*2)<=GetMaxHP())
    {
        m_nSpecialCounter=0;
        state Retreat;
    }
    true;
}

}-----
```

```

event OnCannonLowAmmo(int nCannonNum)
{
    true;
}
//-----
event OnCannonNoAmmo(int nCannonNum)
{
    true;
}
//-----
event OnCannonFoundTarget(int nCannonNum, unit uTarget)
{
    false;//gdby zwrocic true to dzialko nie strzeli
}
//-----
event OnCannonEndFire(int nCannonNum, int nEndStatus)
{
    false;
}
//-----
event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    CallFreeze(nFreezeTicks);
    state Frozen;
    true;
}
//-----
event OnTransportedToNewWorld()
{
}

***** C O M M A N D S *****
command Initialize()
{
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    traceMode = 0;
}
//-----
command Uninitialize()
{
    //wykasowac referencje
    m_uSpecialUnit = null;
}
//-----
command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        lights = nMode;
    }
    SetLightsMode(lights);
}

//-----
command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    if(IsMoving())
    {
        CallStopMoving();
        state Nothing;
    }
}
//-----
command Move(int nGx, int nGy, int nLz) hidden button
"translateCommandMove" description "translateCommandMoveDescription" hotkey
priority 21
{
    CallMoveToPoint(nGx, nGy, nLz);
    state StartMoving;
}
```

```

    command UserNoParam0() button "translateCommandRetreat" description
"translateCommandRetreatDescription" hotkey priority 25
{
    m_nSpecialCounter=0;
    state Retreat;
}

//-----
command Patrol(int nGx, int nGy, int nLz) button "translateCommandPatrol"
description "translateCommandPatrolDescription" hotkey priority 29
{
    m_nSpecialGx = nGx;
    m_nSpecialGy = nGy;
    m_nSpecialLz = nLz;
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    CallMoveToPoint(nGx, nGy, nLz);
    m_nSpecialCounter = 1;
    state Patrol;
}

//-----
command Escort(unit uUnit) button "translateCommandEscort" description
"translateCommandEscortDescription" hotkey priority 31
{
    m_uSpecialUnit=uUnit;
    m_nSpecialGx=GetLocationX()-m_uSpecialUnit.GetLocationX();
    m_nSpecialGy=GetLocationY()-m_uSpecialUnit.GetLocationY();
    if(m_nSpecialGx > 2) m_nSpecialGx=-2;
    if(m_nSpecialGx < -2) m_nSpecialGx=2;
    if(m_nSpecialGy > 2) m_nSpecialGy=-2;
    if(m_nSpecialGy < -2) m_nSpecialGy=2;
    state Escort;
}

//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    CallMoveInsideObject(uEntrance);
    state StartMoving;
}

//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}

/* command UserOneParam9(int nMode) hidden button traceMode priority
255
{
    if(nMode == -1)
    {
        traceMode = (traceMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        traceMode = nMode;
    }
} */
}



## CivilLightsOff.ec

civil "translateScriptNameUnarmedVechicleLightsOff"
{
    int m_nStayGx;
    int m_nStayGy;
    int m_nStayLz;
    int m_nSpecialGx; //Patrol point , escort
    int m_nSpecialGy;
    int m_nSpecialLz;
    int m_nSpecialCounter;
    unit m_uSpecialUnit;

    enum lights
    {
        "translateCommandStateLightsAUTO",
        "translateCommandStateLightsON",
        "translateCommandStateLightsOFF",
    }

    multi:
}

    "translateCommandStateLightsMode"
}

enum traceMode
{
    "translateCommandStateTraceOFF",
    "translateCommandStateTraceON",
}

multi:
    "translateCommandStateTraceMode"
}

***** FUNCTIONS *****
***** STATES *****

//-----
function int EndState()
{
    NextCommand(1);
}

//-----
state Retreat;
state Nothing;
state StartMoving;
state Moving;
state Patrol;
state Escort;

//-----
state Retreat
{
    unit uTarget;
    if(!m_nSpecialCounter)
    {
        uTarget = FindClosestEnemy();
        m_nSpecialCounter = (m_nSpecialCounter+1)%16;
    }
    else
    {
        if(IsMoving())
            m_nSpecialCounter = (m_nSpecialCounter+1)%16;
        else
            m_nSpecialCounter = 0;
        if(traceMode)TraceD("R->N
");
        return Retreat;
    }
}

if(!uTarget)
{
    if(IsMoving())
        CallStopMoving();
    if(traceMode)TraceD("R->N
");
    return Nothing;
}
CallMoveToPoint(2*GetLocationX()-uTarget.GetLocationX(),2*GetLocationY()-
uTarget.GetLocationY(),GetLocationZ());
if(traceMode)TraceD("R u
");
return Retreat;
}

//-----
state Nothing
{
    if(traceMode)TraceD("N
");
    return Nothing;
}

//-----
state StartMoving
{
    return Moving, 20;
}

//-----
state Moving
{
    if (!IsMoving())
    {
        if(traceMode) TraceD("Moving
");
        return Moving;
    }
    else
    {
        if(traceMode) TraceD("Moving -> N
");
        EndState();
    }
}

```

```

        return Nothing;
    }
}
//-----
state Patrol
{
    if (!IsMoving())
    {
        if(m_nSpecialCounter == 1)
        {
            CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
            m_nSpecialCounter = 0;
        }
        else
        {
            CallMoveToPoint(m_nSpecialGx, m_nSpecialGy, m_nSpecialLz);
            m_nSpecialCounter = 1;
        }
    }
    return Patrol;
}
//-----
state Escort
{
    int nTargetGx;
    int nTargetGy;
    if(traceMode)
        TraceD("Escort
\n");
    if(!m_uSpecialUnit.IsLive())
    {
        m_uSpecialUnit=null;
        if(!IsMoving())
        {
            CallStopMoving;
            return Escort;
        }
        EndState();
        return Nothing;
    }
    nTargetGx = m_uSpecialUnit.GetLocationX() + m_nSpecialGx;
    nTargetGy = m_uSpecialUnit.GetLocationY() + m_nSpecialGy;
    if(Distance(nTargetGx,nTargetGy,GetLocationX(),GetLocationY()) > 0)
    {
        if(traceMode)
            TraceD("Escort: updating position
\n");
        CallMoveToPoint(nTargetGx, nTargetGy, m_uSpecialUnit.GetLocationZ());
        return Escort;
    }
    else
    {
        if(IsMoving())
        {
            CallStopMoving;
            return Escort;
        }
        return Escort;
    }
}
//-----
state Frozen
{
    if (!IsFrozen())
    {
        state Frozen;
    }
    else
    {
        //!!wrocic do tego co robilismy
        state Nothing;
    }
}
//*****
//***** E V E N T S *****
//*****
//zwrocic true
//false jak nie ma
event OnHit()
{
    if((GetHP()*2)<=GetMaxHP())
    {
        m_nSpecialCounter=0;
        state Retreat;
    }
    true;
}
//-----
event OnCannonLowAmmo(int nCannonNum)
{
    true;
}
//-----
event OnCannonNoAmmo(int nCannonNum)
{
    true;
}
//-----
event OnCannonFoundTarget(int nCannonNum, unit uTarget)
{
    false;//gdby zwrocic true do dzialko nie strzeli
}
//-----
event OnCannonEndFire(int nCannonNum, int nEndStatus)
{
    false;
}
//-----
event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    CallFreeze(nFreezeTicks);
    state Froozen;
    true;
}
//-----
event OnTransportedToNewWorld()
{
}
//*****
//***** C O M M A N D S *****
//*****
command Initialize()
{
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    traceMode = 0;
    lights=2;
    SetLightsMode(lights);
}
//-----
command Uninitialize()
{
    //wykasowac referencje
    m_uSpecialUnit = null;
}
//-----
command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        lights = nMode;
    }
    SetLightsMode(lights);
}
//-----
command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    if(IsMoving())
    {
        CallStopMoving();
    }
    state Nothing;
}
//-----
command Move(int nGx, int nGy, int nLz) hidden button

```

```

"translateCommandMove" description "translateCommandMoveDescription" hotkey priority 21
{
    CallMoveToPoint(nGx, nGy, nLz);
    state StartMoving;
}
//-----
command UserNoParam0() button "translateCommandRetreat" description "translateCommandRetreatDescription" hotkey priority 25
{
    m_nSpecialCounter=0;
    state Retreat;
}

//-----
command Patrol(int nGx, int nGy, int nLz) button "translateCommandPatrol" description "translateCommandPatrolDescription" hotkey priority 29
{
    m_nSpecialGx = nGx;
    m_nSpecialGy = nGy;
    m_nSpecialLz = nLz;
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    CallMoveToPoint(nGx, nGy, nLz);
    m_nSpecialCounter = 1;
    state Patrol;
}

//-----
command Escort(unit uUnit) button "translateCommandEscort" description "translateCommandEscortDescription" hotkey priority 31
{
    m_uSpecialUnit=uUnit;
    m_nSpecialGx=GetLocationX()-m_uSpecialUnit.GetLocationX();
    m_nSpecialGy=GetLocationY()-m_uSpecialUnit.GetLocationY();
    if(m_nSpecialGx > 2) m_nSpecialGx=-2;
    if(m_nSpecialGx < -2) m_nSpecialGx=2;
    if(m_nSpecialGy > 2) m_nSpecialGy=-2;
    if(m_nSpecialGy < -2) m_nSpecialGy=2;
    state Escort;
}

//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    CallMoveInsideObject(uEntrance);
    state StartMoving;
}

//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript" description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
//-----
/* command UserOneParam9(int nMode) hidden button traceMode priority 255
{
    if (nMode == -1)
    {
        traceMode = (traceMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        traceMode = nMode;
    }
}*/
```

Harvester.ec

```

harvester "translateScriptNameHarvester"
{
    int m_nMoveToX;
    int m_nMoveToY;
    int m_nMoveToZ;
    int m_nHarvestX;
    int m_nHarvestY;
    int m_nHarvestZ;
    int m_bValidHarvest;
    int m_nDestinationX;
    int m_nDestinationY;
```

```

        return WaitForMovingToHarvestPoint;
    }
    else
    {
        //tak dla pewnosci wywolujemy jeszcze raz Call...
        CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
        return MovingToHarvestPoint,40;
    }
}

state MovingToHarvestPoint
{
    int nPosX;
    int nPosY;
    int nPosZ;

    //=====uwaga latka===== harvesterka staje przed punktem w
    //ktorym ma harvestowac nalezaloby sprawdzic czy tam ktos stoi
    if (!IsMoving())
    {
        nPosX = GetLocationX() - m_nHarvestX;
        nPosY = GetLocationY() - m_nHarvestY;
        nPosZ = GetLocationZ() - m_nHarvestZ;
        if (!InPos)
        {
            if ((nPosX<0)nPosX=-nPosX;
            if ((nPosY<0)nPosY=-nPosY;
            if ((nPosX<2 && nPosY<2)
            {
                CallStopMoving();
            }
        }
        //=====
        if (!IsMoving())
        {
            return MovingToHarvestPoint;
        }
        else
        {
            nPosX = GetLocationX();
            nPosY = GetLocationY();
            nPosZ = GetLocationZ();
            if (!m_bValidHarvest && (nPosX == m_nHarvestX) && (nPosY ==
            m_nHarvestY) && (nPosZ == m_nHarvestZ))
            {
                //sprawdzic czy jest tu jeszcze zasob
                if (!IsResourceInPoint(nPosX, nPosY, nPosZ))
                {
                    CallHarvest();
                    return Harvesting;
                }
                else
                {
                    //znalezc jakis zasob
                    if (FindNewHarvestPoint())
                    {
                        CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY,
                        m_nHarvestZ);
                        return MovingToHarvestPoint;
                    }
                    else
                    {
                        return Nothing;
                    }
                }
            }
            else
            {
                //sprawdzic czy nie ma zasobu tu gdzie stojmy (o ile jest rozny od
                slotu budynku)
                if (!IsResourceInPoint(nPosX, nPosY, nPosZ) &&
                (!m_bValidDestination || (nPosX != m_nDestinationX) || (nPosY
                != m_nDestinationY) || (nPosZ != m_nDestinationZ)))
                {
                    //ok kopimy tu gdzie jessem
                    SetHarvestPoint(nPosX, nPosY, nPosZ);
                    CallHarvest();
                    return Harvesting;
                }
                else
                {
                    //znalezc jakis zasob
                    if (FindNewHarvestPoint())
                    {
                        CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY,
                        m_nHarvestZ);
                        return MovingToHarvestPoint;
                    }
                }
            }
        }
    }
}

state CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY,
m_nHarvestZ);
{
    CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
    return MovingToHarvestPoint;
}
}

state MovingToHarvestPoint,101;
{
    CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY,
m_nHarvestZ);
    return MovingToHarvestPoint;
}
}

state MovingToDestinationBuilding
{
    int nPosX;
    int nPosY;
    int nPosZ;
    if (!IsMoving())
    {
        return MovingToDestinationBuilding;
    }
    else
    {
        nPosX = GetLocationX();
        nPosY = GetLocationY();
        nPosZ = GetLocationZ();
        if (!m_bValidDestination && (nPosX == m_nDestinationX) && (nPosY ==
        m_nDestinationY) && (nPosZ == m_nDestinationZ))
        {
            CallPutResource();
            return PuttingResource;
        }
        else
        {
            if (m_bValidDestination)
            {
                //kazac mu tam znrow jechac (moze jakis licznik (+ zmiana
                budynku) zeby nie wywoylac tego w kolko)
                CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
                m_nDestinationZ);
                return MovingToDestinationBuilding,41;
            }
            else
            {
                return Nothing;
            }
        }
    }
}

state Harvesting
{
    unit uBuilding;
    if (!IsHarvesting())
    {
        //jeszcze nie skonczyl
        return Harvesting;
    }
    else
    {
        //skonczyl - jedziemy do budynku
        if (HaveFullResources())
        {
            if (m_bValidDestination && (GetHarvesterBuilding() != null))
            {
                CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
                m_nDestinationZ);
                return MovingToDestinationBuilding;
            }
            else
            {
                //!!znalezc inny budynek
                uBuilding = FindHarvesterRefineryBuilding();
                if (uBuilding != null)
                {
                    SetHarvesterBuilding(uBuilding);
                    m_nDestinationX = GetHarvesterBuildingPutLocationX();
                    m_nDestinationY = GetHarvesterBuildingPutLocationY();
                    m_nDestinationZ = GetHarvesterBuildingPutLocationZ();
                    m_bValidDestination = true;
                    CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
                    m_nDestinationZ);
                    return MovingToDestinationBuilding;
                }
            }
        }
    }
}

```

```

        }
        else
        {
            return Nothing;
        }
    }
    else
    {
        //nie zdolalismy zapelnic calego pojazdu - znalezcz nastepna kratke
        if (FindNewHarvestPoint())
        {
            CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY,
m_nHarvestZ);
            return MovingToHarvestPoint;
        }
        else
        {
            //nie ma juz zasobow - pojedzacy z tym co mamy (o ile mamy) do
            budynku
            if (HaveSomeResources())
            {
                if (m_bValidDestination && (GetHarvesterBuilding() != null))
                {
                    CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);
                    return MovingToDestinationBuilding;
                }
                else
                {
                    //!!znalezcz inny budynek
                    uBuilding = FindHarvesterRefineryBuilding();
                    if (uBuilding != null)
                    {
                        SetHarvesterBuilding(uBuilding);
                        m_nDestinationX = GetHarvesterBuildingPutLocationX();
                        m_nDestinationY = GetHarvesterBuildingPutLocationY();
                        m_nDestinationZ = GetHarvesterBuildingPutLocationZ();
                        m_bValidDestination = true;
                        CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);
                        return MovingToDestinationBuilding;
                    }
                    else
                    {
                        return Nothing;
                    }
                }
            }
            else
            {
                return Nothing;
            }
        }
    }
}

state PuttingResource
{
    unit uBuilding;
    if (IsPuttingResource())
    {
        return PuttingResource;
    }
    else
    {
        if (HaveSomeResources())
        {
            //z jakiegos powodu zostaly resource'y
            if (m_bValidDestination && (GetHarvesterBuilding() != null))
            {
                CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);
                return MovingToDestinationBuilding;
            }
            else
            {
                //!!znalezcz inny budynek
                uBuilding = FindHarvesterRefineryBuilding();
                if (uBuilding != null)
                {
                    SetHarvesterBuilding(uBuilding);

```

```

m_nDestinationX = GetHarvesterBuildingPutLocationX();
m_nDestinationY = GetHarvesterBuildingPutLocationY();
m_nDestinationZ = GetHarvesterBuildingPutLocationZ();
m_bValidDestination = true;
CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);

return MovingToDestinationBuilding;
}
//else przechodzimy dalej
}
if (m_bValidHarvest)
{
    //nie sprawdzamy IsResourceInPoint zeby nie weszla do znajdowania
nowego punktu
    //bo moze znalezc ten na ktorym stojmy i zablokowac slot budynku
    //pojechac tam
    CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
    return MovingToHarvestPoint;
}
else
{
    //znalezcz nowy punkt
    if (FindNewHarvestPoint())
    {
        CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY,
m_nHarvestZ);
        return MovingToHarvestPoint;
    }
    else
    {
        return Nothing;
    }
}
//-----
state Froozen
{
    if (IsFroozed())
    {
        state Froozed;
    }
    else
    {
        //!!wrocic do tego co robilismy
        if(m_nState==1)
        {
            CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
            state MovingToHarvestPoint;
        }
        else
        {
            if(m_nState==2)
            {
                CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);
                state MovingToDestinationBuilding;
            }
            else
            {
                if(m_nState==3)
                {
                    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
                    state StartMoving;
                }
                else
                state Nothing;
            }
        }
    }
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    if(state==WaitForMovingToHarvestPoint || 
state==PuttingResource || 
state==Harvesting)
    {
    }
    else
    {
        m_nState=0;
    }
}

```

```

if(state==MovingToHarvestPoint)
    m_nState=1;
if(state==MovingToDestinationBuilding)
    m_nState=2;
if(state==Moving)
    m_nState=3;
CallFreeze(nFreezeTicks);
state Frozen;
}
true;
true;
}

//-----
command Initialize()
{
    m_bValidHarvest = false;
    m_bValidDestination = false;
    //pozwolic dzialkom strzelac samym (o ile sa jakies)
    SetCannonFireMode(-1, 1);
    false;
}

command Uninitialize()
{
    //wykasowac referencje
    m_bValidHarvest = false; //potrzebne przy przejezdaniu do innego swiata
    m_bValidDestination = false;
    InvalidateCurrentHarvestPoint();
    SetHarvesterBuilding(null);
    false;
}

/*bez nazwy - wywoywane po kliknieciu kursorem*/
command SetHarvestPoint(int nX, int nY, int nZ) hidden button
"translateCommandHarvest"
{
    SetHarvestPoint(nX, nY, nZ);
    if (!HaveFullResources())
    {
        CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
        if (IsHarvesting())
        {
            state WaitForMovingToHarvestPoint;
        }
        else
        {
            state MovingToHarvestPoint;
        }
    }
    else
    {
        if (m_bValidDestination && (GetHarvesterBuilding() != null))
        {
            CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);

            state MovingToDestinationBuilding;
        }
    }
    NextCommand(1);
    true;
}

/*bez nazwy - wywoywane po kliknieciu kursorem*/
command SetContainerDestination(unit uObject) hidden button
"translateCommandSetRefinery"
{
    SetHarvesterBuilding(uObject);
    m_nDestinationX = GetHarvesterBuildingPutLocationX();
    m_nDestinationY = GetHarvesterBuildingPutLocationY();
    m_nDestinationZ = GetHarvesterBuildingPutLocationZ();
    m_bValidDestination = true;
    if (HaveSomeResources())
    {
        CallMoveToPointForce(m_nDestinationX, m_nDestinationY,
m_nDestinationZ);

        state MovingToDestinationBuilding;
    }
    else
    {
        if (m_bValidHarvest)
        {
            CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
        }
    }
}

if (!IsHarvesting())
{
    state WaitForMovingToHarvestPoint;
}
else
{
    state MovingToHarvestPoint;
}
}
NextCommand(1);
true;
}

command Move(int nX, int nY, int nZ) hidden button
"translateCommandMove" description "translateCommandMoveDescription" hotkey priority 21
{
    m_nMoveToX = nX;
    m_nMoveToY = nY;
    m_nMoveToZ = nZ;
    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    state StartMoving;
    true;
}

command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    m_nMoveToX = GetEntranceX(uEntrance);
    m_nMoveToY = GetEntranceY(uEntrance);
    m_nMoveToZ = GetEntranceZ(uEntrance);
    CallMoveInsideObject(uEntrance);
    state StartMoving;
    true;
}

command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    CallStopMoving();
    state StartMoving;
    true;
}
//-----
command Land() button "translateCommandLand" description
"translateCommandLandDescription" hotkey priority 31
{
    if (state==Harvesting) return;
    if (state==PuttingResource) return;
    CallLand();
    state Nothing;
}

//-----
command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        lights = nMode;
    }
    SetLightsMode(lights);
}

//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    /*special command - no implementation
    */
}

command UserNoParam0() button "Find"
{
    if (FindNewHarvestPoint())
    {
        CallMoveAndLandToPoint(m_nHarvestX, m_nHarvestY, m_nHarvestZ);
        state MovingToHarvestPoint;
    }
    true;
}
*/

```

```

}

Helicopter.ec
aircraft "translateScriptNameAircraft"
{
    consts
    {
        nonAttack=0;
        movingOnPosition=1;
        attacking=2;
        evacuating=3;
        attackRange=7;

        disableFire=0;
        findTargetWaterUnit = 1;
        findTargetFlyingUnit = 2;
        findTargetNormalUnit = 4;
        findTargetBuildingUnit = 8;
        findTargetAnyUnit = 15;

        //typ rasy (jaki ch IFF'ow szukamy
        findEnemyUnit = 1;
        //kryterium szukania
        findNearestUnit = 1;
        findWeakestUnit = 2;
        //typ szukanego obiektu
        findDestinationAnyUnit = 15;
        //lslnRange
        noInRange = 0;//poza zasięgiem (za dużą odległość)
        inRangeBadAngleAlpha = 1; //w zasięgu strzału ale zły kat alpha trzeba
        obrócić podwozie
        inRangeBadAngleBeta = 2; //w zasięgu strzału ale zły kat beta
        inRangeBadHit = 3; //w zasięgu strzału ale nie można trafić (cos zasłania)
        inRangeGoodHit = 4;
    }

    unit m_uTarget;
    int m_nTargetGx;
    int m_nTargetGy;
    int m_nTargetLz;
    int m_nStayGx;
    int m_nStayGy;
    int m_nStayLz;
    int m_nAttackGx;
    int m_nAttackGy;
    int m_nAttackPhase;
    int m_nSpecialGx; //Patrol point , escort
    int m_nSpecialGy;
    int m_nSpecialLz;
    int m_nSpecialCounter;
    unit m_uSpecialUnit;
    int m_nState;
    int m_bNeedReload;
    int m_bNeedStop;

    enum lights
    {
        "translateCommandStateLightsAUTO",
        "translateCommandStateLightsON",
        "translateCommandStateLightsOFF",
        multi: "translateCommandStateLightsMode"
    }

    enum movementMode
    {
        "translateCommandStateFollowEnemy",
        "translateCommandStateHoldPosition",
        multi: "translateCommandStateMovement"
    }

    enum attackMode
    {
        "translateCommandStateDynamicAttack",
        "translateCommandStateStaticAttack",
        multi: "translateCommandStateAttackMode"
    }

    enum traceMode
    {
        "translateCommandStateTraceOFF",
        "translateCommandStateTraceON",
        multi: "translateCommandStateTraceMode"
    }
}

multi:
    "translateCommandStateTraceMode"
}

state Nothing;
state HoldPosition;
state StartMoving;
state Moving;
state MovingAfterReload;
state AutoAttacking;
state Attacking;
state AttackingPoint;
state Patrol;
state Escort;
//for flyables
state MovingForGetSupply;
state GettingSupply;

//*********************************************************************F U N C T I O N S *****
//-----
function int SetTarget(unit uTarget)
{
    m_uTarget = uTarget;
    SetTargetObject(uTarget);
    return true;
}
//-----
function int FindBestTarget()
{
    if(!CanCannonFireToAircraft(0))
    {
        SetTarget(FindClosestEnemyUnitOrBuilding(findTargetWaterUnit |
findTargetNormalUnit));
    }
    else
    {
        SetTarget(FindClosestEnemyUnitOrBuilding(findTargetWaterUnit |
findTargetNormalUnit | findTargetFlyingUnit));
    }
}
//-----
function int AttackRun(int bAttackOnPoint)
{
    int nRangeMode;
    int nX;
    int nY;
    int nXAbs;
    int nYAbs;
    int nDistance;
    int nR;

    if(attackMode==1)//position attack
    {
        if(GetCannonType(0)==cannonTypeBomb)//bombowiec
        {
            if(traceMode) TraceD("bombowiec\n");
            if(GetLocationX()==m_nTargetGx &&
GetLocationY()==m_nTargetGy)//nad celom
            {
                if (!IsMoving())
                {
                    if(traceMode) TraceD("CallStopMoving
CallStopMoving();\n");
                    return true;
                }
                else
                {
                    if(traceMode) TraceD("GoToTarget: Target not in range
\n");
                    CallMoveToPoint(m_nTargetGx,m_nTargetGy,0);
                    return false;
                }
            } //koniec bombowca
            if(bAttackOnPoint)
                nRangeMode =
IsPointInCannonRange(0,m_nTargetGx,m_nTargetGy,m_nTargetLz);
            else
                nRangeMode = IsTargetInCannonRange(0, m_uTarget);

            if (nRangeMode == inRangeGoodHit)
            {

```

```

    if(traceMode) TraceD("In range\n");
    if(!isMoving() && m_bNeedStop)
    {
        m_bNeedStop=false;
        if(traceMode) TraceD("CallStopMoving\n");
        CallStopMoving();
    }
    return true;
}

if(nRangeMode == inRangeBadAngleAlpha) //w zasiegu ale trzeba
odwrocic czolg
{
    if(traceMode) TraceD("Rotating tank:");
    if(isMoving())
    {
        if(traceMode) TraceD("CallStopMoving\n");
        if(m_bNeedStop)
        {
            m_bNeedStop=false;
            CallStopMoving();
        }
    }
    else
    {
        if(traceMode) TraceD("CallTurnToAngle\n");
        if(bAttackOnPoint)
    }

    CallTurnToAngle(GetCannonAngleToPoint(0,m_nTargetGx,m_nTargetGy,m_nTargetLz));
    else
        CallTurnToAngle(GetCannonAngleToTarget(0,m_uTarget));
}
return false;
}

if(nRangeMode == inRangeBadHit)
{
    /*if(traceMode) TraceD("Zaslanianie\n");
    nDx = m_nTargetGx - m_nTargetGy + GetLocationY();
    nDy = m_nTargetGy + m_nTargetGx - GetLocationX();
    CallMoveLowToPoint(nDx,nDy,0);*/
    if(traceMode) TraceD("Rotating tank (zaslania);");
    if(!isMoving())
    {
        if(traceMode) TraceD("CallStopMoving\n");
        if(m_bNeedStop)
        {
            m_bNeedStop=false;
            CallStopMoving();
        }
    }
    else
    {
        if(traceMode) TraceD("CallTurnToAngle\n");
        if(bAttackOnPoint)
    }

    CallTurnToAngle(GetCannonAngleToPoint(0,m_nTargetGx,m_nTargetGy,m_nTargetLz));
    else
        CallTurnToAngle(GetCannonAngleToTarget(0,m_uTarget));
}
return true;
}

m_bNeedStop=true;
if(nRangeMode == inRangeBadAngleBeta)
{
    if(traceMode) TraceD("Zla beta\n");
    if(GetLocationX()==m_uTarget.GetLocationX() &&
GetLocationX()==m_uTarget.GetLocationX())
        CallMoveToPoint(GetLocationX(),+3,GetLocationY(), 0);
    else
        CallMoveToPoint(2*GetLocationX(),
m_uTarget.GetLocationX(),2*GetLocationY()-m_uTarget.GetLocationY(), 0);
    return true;
}

nDistance = DistanceTo(m_nTargetGx,m_nTargetGy);
if(nDistance <=1;nDx=-1;)//gdy stojmy nad celem
nDx = GetLocationX()-m_nTargetGx;
nDy = GetLocationY()-m_nTargetGy;
nRangeMode = GetCannonShootRange(0)-1;

nDxAbs = m_nTargetGx+((nRangeMode*nDx)/nDistance);
nDyAbs = m_nTargetGy+((nRangeMode*nDy)/nDistance);

if(nDxAbs==GetLocationX() && nDyAbs==GetLocationY())
{
    nRangeMode = GetCannonShootRange(0)-2;
    nDxAbs = m_nTargetGx+((nRangeMode*nDx)/nDistance);
    nDyAbs = m_nTargetGy+((nRangeMode*nDy)/nDistance);
}

CallMoveLowToPoint(nDxAbs,nDyAbs,0);
if(traceMode)
{
    TraceD("Target not in range dist=");
    Traced(nDistance);
    TraceD("rRange = ");
    TraceD(nRangeMode);
    TraceD("\n");
}
return false;
}

if(m_nAttackPhase==nonAttack)
{
    if(traceMode) TraceD("nonAttack\n");
    nDx=GetLocationX()-m_nTargetGx;
    nDy=GetLocationY()-m_nTargetGy;

    if(nDx<0)(nDxAbs=-nDx;nDx=-attackRange);
    else (nDxAbs=nDx;nDx=attackRange);
    if(nDy<0)(nDyAbs=-nDy;nDy=-attackRange);
    else (nDyAbs=nDy;nDy=attackRange);

    if(nDyAbs>nDxAbs)nDy=0;
    else nDx=0;

    m_nAttackGx = m_nTargetGx+nDx;
    m_nAttackGy = m_nTargetGy+nDy;

    CallMoveToPoint(m_nAttackGx, m_nAttackGy, 0);
    m_nAttackPhase = movingOnPosition;
    return false;
}

if(m_nAttackPhase==movingOnPosition)
{
    if(traceMode) TraceD("movingOnPosition\n");
    if(!isMoving() || (m_nAttackGx==GetLocationX())&&
m_nAttackGy==GetLocationY()))
    {
        CallMoveLowToPoint(2*m_nTargetGx - m_nAttackGx, 2*m_nTargetGy -
m_nAttackGy, 0);
        m_nAttackPhase=attacking;
    }
    return false;
}

if(m_nAttackPhase==attacking)
{
    if(traceMode) TraceD("attacking:\n");
    nDistance =
Distance(m_nAttackGx,m_nAttackGy,GetLocationX(),GetLocationY());
    if(traceMode) { TraceD(nDistance);TraceD("\n");}
    if((GetCannonType(0)!=cannonTypeBomb && nDistance>4) ||
nDistance>8)
    {
        nDx = (m_nTargetGx - m_nAttackGx);
        nDy = (m_nTargetGy - m_nAttackGy);
        m_nAttackGx = m_nTargetGx+nDy;
        m_nAttackGy = m_nTargetGy+nDx;
        m_nAttackPhase=movingOnPosition;//evacuating;
        CallMoveToPoint(m_nAttackGx, m_nAttackGy, 0);
        return false;
    }
    return true;
}

/*if(m_nAttackPhase==evacuating)
{
    if(traceMode) TraceD("evacuating:\n");
    nDistance =
Distance(m_nTargetGx,m_nTargetGy,GetLocationX(),GetLocationY());
    if(traceMode) { TraceD(nDistance);TraceD("\n");}
    if(!isMoving() || nDistance>3)
    {
        m_nAttackPhase=nonAttack;
    }
    return false;
}/*
return false;
}

//-----
function int TargetInRange()

```

```

    {
        int nRangeMode;
        int nDx;
        int nDy;
        nRangeMode = IsTargetInCannonRange(0, m_uTarget);

        if (nRangeMode == inRangeGoodHit)
        {
            return true;
        }
        if(nRangeMode == inRangeBadAngleAlpha) //w zasiegu ale trzeba odwro-
cic czolg
        {
            if(traceMode) TraceD("Rotating aircraft:");
            CallTurnToAngle(GetCannonAngleToTarget(0,m_uTarget));
            return true;
        }
        return false;
    }
    //-----
    function int CheckAmmo()
    {
        if(CannonRequiresSupply(0) && (m_bNeedReload || !GetAmmoCount()) &&
FindSupplyCenterPlace())
        {
            m_nState=0;
            if(state==AttackingPoint)
                m_nState = 1;
            if(state==Attacking)
                m_nState = 2;
            if(state==Moving)
                m_nState = 3;

            m_nSpecialGx = GetLocationX();
            m_nSpecialGy = GetLocationY();

            m_nStayGx = GetFoundSupplyCenterPlaceX();
            m_nStayGy = GetFoundSupplyCenterPlaceY();
            m_nStayLz = GetFoundSupplyCenterPlaceZ();
            //CallMoveAndLandToPointForce(m_nStayGx, m_nStayGy, m_nStayLz);
            CallMoveToPointForce(m_nStayGx, m_nStayGy, m_nStayLz); //dodane
25.01.2000
            return true;
        }
        return false;
    }
    //-----
    function int EndState()
    {
        NextCommand(1);
    }
    //***** STATES *****
    //***** STATES *****

    //-----
    state Nothing
    {
        if(traceMode)TraceD("\n");
        if(movementMode==1) return HoldPosition;
        if(CheckAmmo())return MovingForGetSupply;

        FindBestTarget();
        if(!m_uTarget)
        {
            SetTarget(GetAttacker());
            ClearAttacker();
        }

        if (m_uTarget != null)
        {
            m_nStayGx = GetLocationX();
            m_nStayGy = GetLocationY();
            m_nStayLz = GetLocationZ();
            m_nAttackPhase=nonAttack;
            return AutoAttacking;
        }
        return Nothing;
    }
    //-----
    state HoldPosition
    {
        if(traceMode)TraceD("\n");
        if(CheckAmmo())return MovingForGetSupply;

        if(m_uTarget)
        {
            if(!m_uTarget.IsLive() || !IsEnemy(m_uTarget))
            {
                StopCannonFire(-1);
                SetTarget(null);
                m_nAttackPhase=nonAttack;
            }
            else
            {
                if(TargetInRange())
                {
                    if(traceMode)TraceD("HP Fire!!!\n");
                    CannonFireToTarget(-1, m_uTarget, -1);
                    return HoldPosition;
                }
                else
                {
                    SetTarget(null);
                    return HoldPosition,10;
                }
            }
        }
        else
        {
            FindBestTarget();
            if(!m_uTarget)
            {
                SetTarget(GetAttacker());
                ClearAttacker();
            }
        }
        return HoldPosition;
    }
    //-----
state AutoAttacking
{
    int nDistance;
    if(traceMode)TraceD("AA\n");
    if(CheckAmmo())return MovingForGetSupply;

    // pozostawaj w okolicach punktu
    nDistance =
Distance(m_nStayGx,m_nStayGy,GetLocationX(),GetLocationY());
    if (nDistance > 18)
    {
        if(traceMode)TraceD("nDistance: > 12 !!!\n");
        SetTarget(null);
        m_nAttackPhase=nonAttack;
        CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
        return Moving;
    }

    if(!m_uTarget.IsLive() || !IsEnemy(m_uTarget))
    {
        StopCannonFire(-1);
        SetTarget(null);
        m_nAttackPhase=nonAttack;
    }

    if (m_uTarget)
    {
        m_nTargetGx = m_uTarget.GetLocationX();
        m_nTargetGy = m_uTarget.GetLocationY();
        if(AttackRun(0))
        {
            if(traceMode)TraceD("AA Fire!!!\n");
            CannonFireToTarget(-1, m_uTarget, 2);
        }
        return AutoAttacking;
    }
    else//target not exist
    {
        FindBestTarget();
        if(!m_uTarget)
        {
            SetTarget(GetAttacker());
        }
    }
}

```

```

        ClearAttacker();
    }

    if (m_uTarget != null)
        return AutoAttacking;

    if( nDistance > 0)
    {
        CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
        return Moving;
    }

    if (IsMoving())
    {
        CallStopMoving();
    }
    return Nothing;
}

//-----
state AttackingPoint
{
    if(traceMode)TraceD("AG\n");
    if(CheckAmmo())return MovingForGetSupply;
    if(AttackRun(1))
        CannonFireGround(-1, m_nTargetGx, m_nTargetGy, 0, 2);
    return AttackingPoint;
}
//-----
state Attacking
{
    if(traceMode)TraceD("A\n");
    if(CheckAmmo())return MovingForGetSupply;
    if(m_uTarget.IsLive())
    {
        m_nTargetGx = m_uTarget.GetLocationX();
        m_nTargetGy = m_uTarget.GetLocationY();
        if(AttackRun(0))
            CannonFireToTarget(-1, m_uTarget, 2);
        return Attacking;
    }
    else //target not exist
    {
        SetTarget(null);
        m_nAttackPhase=nonAttack;
        if (IsMoving())
        {
            CallStopMoving();
        }
        EndState();
        return Nothing;
    }
}
//-----
state StartMoving
{
    return Moving, 20;
}
//-----
state Moving
{
    if (IsMoving())
    {
        if(traceMode) TraceD("Moving\n");
        return Moving;
    }
    else
    {
        if(traceMode) TraceD("Moving -> N\n");
        EndState();
        return Nothing;
    }
}
//-----
state Patrol
{
    if(CheckAmmo())return MovingForGetSupply;
    if(m_uTarget)
    {
        if(!m_uTarget.IsLive() || !IsEnemy(m_uTarget))
        {
            StopCannonFire(-1);
            SetTarget(null);
            m_nAttackPhase=nonAttack;
        }
        else
    }
    {
        m_nTargetGx = m_uTarget.GetLocationX();
        m_nTargetGy = m_uTarget.GetLocationY();
        m_nTargetLz = m_uTarget.GetLocationZ();
        if(Distance(m_nTargetGx,m_nTargetGy,(GetLocationX(),GetLocationY())) > 0)
        {
            if(traceMode) TraceD("Escort: updating position\n");
            CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
            return Escort;
        }
        else
        {
            if(IsMoving())
            {
                CallStopMoving();
                return Escort;
            }
        }
        return Escort;
    }
}

```

```

        }

//-----
state Frozen
{
    if (IsFrozen())
    {
        state Frozen;
    }
    else
    {
        //!!wrocic do tego co robilismy
        state Nothing;
    }
}

//-----
//for Flyables
state MovingForGetSupply
{
    if (IsMoving())
    {
        if(traceMode) TraceD("MovingForSupply
\n");
        if ((GetLocationX() == m_nStayGx) && (GetLocationY() ==
m_nStayGy) && (GetLocationZ() == m_nStayLz))
        {
            if(traceMode) TraceD("MovingForSupply CallStopMoving
\n");
            CallStopMoving();
            return MovingForGetSupply;
        }
        return MovingForGetSupply;
    }
    else
    {
        if ((GetLocationX() == m_nStayGx) && (GetLocationY() ==
m_nStayGy) && (GetLocationZ() == m_nStayLz))
        {
            if(traceMode) TraceD("MovingForSupply -> GettingSupply
\n");
            CallGetFlyingSupply();
            return GettingSupply;
        }
        else
        {
            if(traceMode) TraceD("MovingForSupply again
\n");
            //CallMoveAndLandToPointForce(m_nStayGx, m_nStayGy, m_nStayLz);
            //CallMoveToPointForce(m_nStayGx, m_nStayGy, m_nStayLz); //dodane
25.01.2000
            return MovingForGetSupply;
        }
    }
}

state GettingSupply
{
    if (IsGettingSupply())
    {
        if(traceMode) TraceD("GettingSupply
\n");
        return GettingSupply;
    }
    else
    {
        if(traceMode) TraceD("End GettingSupply
\n");
        m_bNeedReload=false;
        if(m_nSpecialGx==GetLocationX() && m_nSpecialGy==GetLocationY())
m_nSpecialGx=m_nSpecialGx+5;
        if(!GetAmmoCount())
        {
            m_nSpecialGx=GetLocationX()-5;
            m_nSpecialGy=GetLocationY();
        }
    }
}

if(m_nSpecialGx){m_nSpecialGx=GetLocationX();}+5;m_nSpecialGy=GetLocationY
()); CallMoveToPoint(m_nSpecialGx,m_nSpecialGy,0);
return MovingAfterReload;

}

//-----
state MovingAfterReload
{
    if (IsMoving())
    {
        if(traceMode) TraceD("MovingAfterReload
\n");
        return MovingAfterReload;
    }
    else
    {
        if(m_nState==1)
            return AttackingPoint,40;
        if(m_nState==2)
            return Attacking,40;
    }
}

if(traceMode) TraceD("MovingAfterReload -> N
\n");
EndState();
return Nothing;
}

//*****
//***** E V E N T S *****
//*****
//zwrocia true
//false jak nikt ma
event OnHit()
{
    true;
}

event OnCannonLowAmmo(int nCannonNum)
{
    true;
}

event OnCannonNoAmmo(int nCannonNum)
{
    m_bNeedReload=true;
    if(CheckAmmo())state MovingForGetSupply;
    true;
}

event OnCannonFoundTarget(int nCannonNum, unit uTarget)
{
    return false;//gdzie zwrocic true to dzialko nie strzeli
}

event OnCannonEndFire(int nCannonNum, int nEndStatus)//gdzie zniszczony,
poza zasiegiem lub brak amunicji
{
    false;
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    CallFreeze(nFreezeTicks);
    state Frozen;
    true;
}

event OnTransportedToWorld()
{
    StopCannonFire(-1);
    SetTarget(null);
    m_uSpecialUnit = null;
}

event OnConvertedToNewPlayer()
{
    StopCannonFire(-1);
    SetTarget(null);
    m_uSpecialUnit = null;
    state Nothing;
}

//*****
//***** C O M M A N D S *****
//*****
command Initialize()
{
    traceMode = 0;
    movementMode = 0;
    attackMode = 1;
    SetCannonFireMode(-1, disableFire);
    m_nAttackGx=-1;
}

```

```

if(!GetAmmoCount())
    m_bNeedReload=true;
else
    m_bNeedReload=false;
}

//-----
command Uninitialize()
{
    //wykasowac referencje
    SetTarget(null);
    m_uSpecialUnit = null;
}

//-----
command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        assert(nMode == 0);
        lights = nMode;
    }
    SetLightsMode(lights);
    NextCommand(0);
}

//-----
command SetMovementMode(int nMode) button movementMode description
"translateCommandStateMovementDescription" priority 205
{
    if (nMode == -1)
    {
        movementMode = (movementMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        movementMode = nMode;
    }
    NextCommand(0);
}

//-----
command SetAttackMode(int nMode) button attackMode description
"translateCommandStateAttackModeDescription" priority 206
{
    if (nMode == -1)
    {
        attackMode = (attackMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        attackMode = nMode;
    }
    NextCommand(0);
}

//-----
command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    SetTarget(null);
    StopCannonFire(-1);
    if(state==GettingSupply) return;
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    if(IsMoving())
        CallStopMoving();
    state Nothing;
}

//-----
command HoldPosition() hidden button "translateCommandHoldPosition" description
"translateCommandHoldPositionDescription" hotkey priority 20
{
    SetTarget(null);
    StopCannonFire(-1);
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    movementMode = 1;
    if(IsMoving())
        CallStopMoving();
}

ChangedCommandValue();
state HoldPosition;
}

//-----
command Move(int nGx, int nGy, int nLz) button "translateCommandMove" description
"translateCommandMoveDescription" hotkey priority 21
{
    SetTarget(null);
    m_nStayGx = nGx;
    m_nStayGy = nGy;
    m_nStayLz = nLz;
    m_nState = 3;
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    CallMoveToPoint(nGx, nGy, nLz);
    state StartMoving;
}

//-----
command Attack(unit uTarget) button "translateCommandAttack" description
"translateCommandAttackDescription" hotkey priority 22
{
    SetTarget(uTarget);
    m_nAttackPhase=nonAttack;
    m_nState = 2;
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    state Attacking;
}

/*komenda nie wystawiana na zewnatrz*/
command AttackOnPoint(int nX, int nY, int nZ) hidden button
"translateCommandAttack"
{
    SetTarget(null);
    m_nTargetGx = nX;
    m_nTargetGy = nY;
    m_nTargetLz = nZ;
    m_nAttackPhase = nonAttack;
    m_nState = 1;
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    state AttackingPoint;
}

//-----
/* command Patrol(int nGx, int nGy, int nLz) button "translateCommandPatrol" description
"translateCommandPatrolDescription" hotkey priority 29
{
    if(state==GettingSupply) return;
    m_nSpecialGx = nGx;
    m_nSpecialGy = nGy;
    m_nSpecialLz = nLz;
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    CallMoveToPoint(nGx, nGy, nLz);
    m_nSpecialCounter = 1;
    state Patrol;
}

//-----
command Escort(unit uUnit) button "translateCommandEscort" description
"translateCommandEscortDescription" hotkey priority 31
{
    if(state==GettingSupply) return;
    m_uSpecialUnit=uUnit;
    m_nSpecialGx=GetLocationX()-m_uSpecialUnit.GetLocationX();
    m_nSpecialGy=GetLocationY()-m_uSpecialUnit.GetLocationY();
    if(m_nSpecialGx > 2) m_nSpecialGx=2;
    if(m_nSpecialGx < -2) m_nSpecialGx=-2;
    if(m_nSpecialGy > 2) m_nSpecialGy=2;
    if(m_nSpecialGy < -2) m_nSpecialGy=-2;
    state Escort;
}

//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    CallMoveInObject(uEntrance);
    m_nTargetGx = GetEntranceX(uEntrance);
    m_nTargetGy = GetEntranceY(uEntrance);
    m_nTargetLz = GetEntranceZ(uEntrance);
    state StartMoving;
}

//-----
/* command UserOneParam9(int nMode) button traceMode priority 255

```

```

    {
        if (nMode == -1)
        {
            traceMode = (traceMode + 1) % 2;
        }
        else
        {
            assert(nMode == 0);
            traceMode = nMode;
        }
    } */
}

//for flyables
command Land() button "translateCommandLand" description
"translateCommandLandDescription" hotkey priority 31
{
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    CallLand();
    state Nothing;
}

command FlyForSupply() button "translateCommandGetSupply" description
"translateCommandGetSupplyDescription" hotkey priority 50
{
    if(state==GettingSupply && IsOnWorkingSupplyCenter()) return;
    if (HaveCannonsMissingAmmo())
    {
        if (FindSupplyCenterPlace())
        {
            m_nSpecialGx = GetLocationX();
            m_nSpecialGy = GetLocationY();

            m_nStayGx = GetFoundSupplyCenterPlaceX();
            m_nStayGy = GetFoundSupplyCenterPlaceY();
            m_nStayLz = GetFoundSupplyCenterPlaceZ();
            //CallMoveAndLandToPointForce(m_nStayGx, m_nStayGy,
m_nStayLz); //jak bylo Force to stawaly w slupku
            CallMoveToPointForce(m_nStayGx, m_nStayGy, m_nStayLz); //dodane
25.01.2000
            state MovingForGetSupply;
        }
        else
        {
            NextCommand(0);
        }
    }
    else
    {
        NextCommand(0);
    }
}
//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
//-----
/*button "Attack"
description "euuhwduihewui"
hotkey // flaga ze ma reagowac na klawisz do tej komendy
priority 7 */
}

Platoon.ec
platoon "translateDefaultPlatoon"
{
    consts
    {
        disableFire=0;
        enableFire=1;
    }

    enum lights
    {
        "translateCommandStateLightsAUTO",
        "translateCommandStateLightsON",
        "translateCommandStateLightsOFF",
    multi:
        "translateCommandStateLightsMode"
    }

    enum traceMode
    {
        "translateCommandStateTraceOFF",
        "translateCommandStateTraceON",
        "translateCommandStateTraceMode"
    }
}

unit m_uTarget;
int m_nTargetGx;
int m_nTargetGy;
int m_nTargetLz;

int bOnTheWay;
int bAuto;
int m_bFindAndDestroyWalls;

function int GoToPoint()
{
    int nRangeMode;
    nRangeMode = IsPointInCannonRange(0,0,m_nTargetGx, m_nTargetGy,
m_nTargetLz);

    if (nRangeMode == 4) //in range
    {
        if (IsMoving())
            CallStopMoving();
        return false;
    }
    CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
    return true;
}

//-----
function int GoToTarget()
{
    int nRangeMode;
    int i;
    for(i=0;i<GetUnitsCount();i=i+1)
    {
        nRangeMode = IsTargetInCannonRange(i,0, m_uTarget);
        if (nRangeMode == inRangeGoodHit)
        {
            if(i) SetLeader(i);
            if (IsMoving())
                CallStopMoving();
            return false;
        }
    }
    CallMoveToPoint(m_uTarget.GetLocationX(), m_uTarget.GetLocationY(),
m_uTarget.GetLocationZ());
    return true;
}

//-----
function int PrepareAttack(unit uTarget)
{
    int i;
    punit member;
    EnableFeatures(platoonFreeUnits,true);
    for(i=0;i<GetUnitsCount();i=i+1)
    {
        member = GetUnit(i);
        member.CommandAttack(uTarget);
    }
}

//-----
function int EndAttack()
{
    int i;
    punit member;
    EnableFeatures(platoonFreeUnits,false);
    for(i=0;i<GetUnitsCount();i=i+1)
    {
        member = GetUnit(i);
        member.CommandStop();
    }
}

//-----
function int EndState()
{
    NextCommand(1);
}

//***** S T A T E S *****
//*****
state FirstState;
state Nothing;

```

```

state StartMoving;
state Moving;
state Attacking;
state AttackingPoint;
//-----
state FirstState //pluton jest w tym stanie dopukni nie zobaczy pierwszego
wroga w tym stanie rozgląda sie wszyscy bo pluton moze byc rozproszony
{
    int i;
    if(traceMode) TraceD("F");

    //m_uTarget = GetAttacker(-1);
    //ClearAttacker(-1);

    if(traceMode) TraceD(".");
    if(m_uTarget)
    {
        for(i=0;i<GetUnitsCount() && !m_uTarget;i++)
        {
            m_uTarget =
FindTarget(i,findTargetWaterUnit | findTargetBuildingRuin | findTargetNormalUnit | fin
dTargBuildingUnit,findEnemyUnit,findNearestUnit,findDestinationAnyUnit);
            if (m_uTarget != null) SetLeader(i);
        }
    }
    if(traceMode) TraceD(".");
    if (m_uTarget != null)
    {
        if(traceMode) TraceD("> A      \n");
        bAuto=true;
        PrepareAttack(m_uTarget);
        return Attacking;
    }
    if(traceMode) TraceD(".");
    return FirstState,20;
}
//-----
state Nothing
{
    int i;
    if(traceMode) TraceD("N");
    m_uTarget = GetAttacker(-1);
    ClearAttacker(-1);
    if(traceMode) TraceD(".");
    if(!m_uTarget)
    {
        if(traceMode) TraceD(" FT");
        for(i=0;i<GetUnitsCount() && !m_uTarget;i++)
        {
            if (m_bFindAndDestroyWalls)
                m_uTarget =
FindTarget(0,findTargetWaterUnit | findTargetBuildingRuin | findTargetWall |
findTargetNormalUnit | findTargetBuildingUnit,findEnemyUnit,findNearestUnit,findDe
stinationAnyUnit);
            else
                m_uTarget =
FindTarget(0,findTargetWaterUnit | findTargetBuildingRuin | findTargetWall |
findTargetNormalUnit,findEnemyUnit,findNearestUnit,findDestinationAnyUnit);
            if (m_uTarget != null)
            {
                SetLeader(i);
                if(traceMode) TraceD(i);
            }
        }
    }
    if(traceMode) TraceD(".");
    if (m_uTarget != null)
    {
        if(traceMode) TraceD("> A      \n");
        bAuto=true;
        PrepareAttack(m_uTarget);
        return Attacking;
    }
    if(traceMode) TraceD(".");
    return Nothing,20;
}
//-----
state StartMoving
{
    return Moving, 20;
}
//-----
state Moving

```

```

    {
        if(bOnTheWay)//bOnTheWay jest po to aby nie robic tego przy komendzie
Enter
        {
            if (m_bFindAndDestroyWalls)
                m_uTarget =
FindTarget(0,findTargetWaterUnit | findTargetBuildingRuin | findTargetWall |
findTargetNormalUnit,findEnemyUnit,findNearestUnit,findDestinationAnyUnit);
            else
                m_uTarget =
FindTarget(0,findTargetWaterUnit | findTargetBuildingRuin | findTargetNormalUnit | fi
ndTargetBuildingUnit,findEnemyUnit,findNearestUnit,findDestinationAnyUnit);

            if (m_uTarget != null)
            {
                if(traceMode) TraceD("M -> A      \n");
                bAuto=true;
                PrepareAttack(m_uTarget);
                return Attacking;
            }
        }
        if(traceMode) TraceD("M");
        if (!isMoving())
        {
            if(traceMode) TraceD("      \n");
            return Moving;
        }
        else
        {
            if(traceMode) TraceD("stop      \n");
            bOnTheWay=false;
            EndState();
            return Nothing;
        }
    }
//-----
state AttackingPoint
{
    if(traceMode) TraceD("AP      \n");
    if(GoToPoint())
    {
        return AttackingPoint;
    }
    else
    {
        CannonFireGround(-1,-1, m_nTargetGx, m_nTargetGy, m_nTargetLz, 1);
        return AttackingPoint;
    }
}
//-----
state Attacking
{
    if(traceMode) TraceD("A");
    if (m_uTarget.IsLive())
    {
        if(bAuto &&
Distance((m_uTarget.GetLocationX()),m_uTarget.GetLocationY())>20)
        {
            if(traceMode) TraceD("- out of distance");
            m_uTarget=null;
            EndAttack();
            if(bOnTheWay)
            {
                CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
                if(traceMode) TraceD(" ->M      \n");
                return StartMoving;
            }
            if(traceMode) TraceD(" ->N      \n");
            return Nothing;
        }
        if(traceMode) TraceD("      \n");
        PrepareAttack(m_uTarget);
        return Attacking,80;
    }
    else //target not exist
    {
        m_uTarget=null;
        EndAttack();
        if(bOnTheWay)
        {
            CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
            return StartMoving;
        }
        EndState();
        return Nothing;
    }
}

```

```

        }
    }

//-----
state Frozen
{
    if (IsFrozen())
    {
        state Frozen;
    }
    else
    {
        state Nothing;
    }
}
//-----
//*****
***** C O M M A N D S *****
***** *****
command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    EndAttack();
    m_uTarget = null;
    SetCannonFireMode(-1,-1, enableFire);
    CallStopMoving();
    bOnTheWay=true;
    state Nothing;
}
//-----
command Move(int nGx, int nGy, int nLz) button "translateCommandMove" description
"translateCommandMoveDescription" hotkey priority 21
{
    EndAttack();
    m_uTarget = null;
    SetCannonFireMode(-1,-1, enableFire);
    CallMoveToPoint(nGx, nGy, nLz);
    m_nTargetGx = nGx;
    m_nTargetGy = nGy;
    m_nTargetLz = nLz;
    bOnTheWay=true;
    state StartMoving;
}
//-----
command Attack(unit uTarget) button "translateCommandAttack" description
"translateCommandAttackDescription" hotkey priority 22
{
    PrepareAttack(m_uTarget);
    bAuto=false;
    bOnTheWay=false;
    m_uTarget = uTarget;
    SetCannonFireMode(-1,-1, enableFire);
    state Attacking;
}
//-----
//komenda nie wstawiana na zewnatrz/
command AttackOnPoint(int nX, int nY, int nZ) hidden button
"translateCommandAttack"
{
    bAuto=false;
    bOnTheWay=false;
    m_uTarget = null;
    m_nTargetGx = nX;
    m_nTargetGy = nY;
    m_nTargetLz = nZ;
    state AttackingPoint;
}
//-----
command DisposePlatoon() button "translateCommandPlatoonDispose" description
"translateCommandPlatoonDisposeDescription" hotkey priority 13
{
    Dispose();
}
//-----
command SetLights(int nMode) button lights priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        assert(nMode == 0);
        lights = nMode;
    }
}

SetLightsMode(-1,lights);
NextCommand(0);

//-----
command UserOneParam9(int nMode) button traceMode priority 255
{
    if (nMode == -1)
    {
        traceMode = (traceMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        traceMode = nMode;
    }
}
//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript" description
"translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
}
//-----
command SendSupplyRequest() button "translateCommandSupply" description
"translateCommandSupplyDescription" hotkey priority 27
{
    SendSupplyRequest(-1);
}
//-----
command AddUnitToPlatoon(unit uUnit) button
"translateCommandPlatoonAddUnit" description
"translateCommandPlatoonAddUnitDescription" hotkey priority 11
{
    AddUnitToPlatoon(uUnit);
    NextCommand(1);
}
//-----
command RemoveUnitFromPlatoon(unit uUnit) button
"translateCommandPlatoonRemoveUnit" description
"translateCommandPlatoonRemoveUnitDescription" hotkey priority 12
{
    RemoveUnitFromPlatoon(uUnit);
    NextCommand(1);
}
//-----
command Initialize()
{
    bAuto=false;
    traceMode = 0;
    bOnTheWay=false;
    m_bFindAndDestroyWalls=0;
    SetCannonFireMode(-1,-1, enableFire);
}
//-----
command UserOneParam0(int nMode) hidden button "F2W"
{
    m_bFindAndDestroyWalls=1;
}
//-----
command UserOneParam1(int nMode) hidden button "Don't F2W"
{
    m_bFindAndDestroyWalls=0;
}
//-----
command Uninitialize()
{
    //wykasowac reference
    m_uTarget = null;
}
//-----





## Repairer.ec


repainer "translateScriptNameRepairer"
{
    consts
    {
        findTargetWaterUnit      = 1;
        findTargetFlyingUnit     = 2;
        findTargetNormalUnit     = 4;
    }
}

```

```

findTargetBuildingUnit = 8;
findTargetAnyUnit = 15;

//typ rasy (jakich IFF'ow szukamy
findEnemyUnit = 1;
findAllUnit = 2;
findNeutralUnit = 4;
findOurUnit = 8;

//kryterium szukania
findDisabledUnit = 8;
findDamagedUnit = 16;
//typ szukaneego obiektu
findDestinationCivilUnit = 1;
findDestinationArmedUnit = 2;
findDestinationRepairerUnit = 4;
findDestinationSupplyUnit = 8;
findDestinationAnyUnit = 15;

operationRepair = 0;
operationCapture = 1;
operationRepaint = 2;
operationUpgrade = 3;
}

int m_nMoveToX;
int m_nMoveToY;
int m_nMoveToZ;
unit m_uCurrTarget;
int m_nCurrOperation;//0-repairing, 1-converting, 2-painting, 3-upgrading
int m_nRepairSideColor;
int m_nStayX;
int m_nStayY;
int m_nStayZ;

enum lights
{
    "translateCommandStateLightsAUTO",
    "translateCommandStateLightsON",
    "translateCommandStateLightsOFF",
multi:
    "translateCommandStateLightsMode"
}

enum traceMode
{
    "translateCommandStateTraceOFF",
    "translateCommandStateTraceON",
multi:
    "translateCommandStateTraceMode"
}

enum repairMode
{
    "translateCommandStateDontRepair",
    "translateCommandStateAutoRepair",
multi:
    "translateCommandStateRepairMode"
}

enum captureMode
{
    "translateCommandStateDontCapture",
    "translateCommandStateAutoCapture",
multi:
    "translateCommandStateCaptureMode"
}

enum upgradeWeaponsMode
{
    "translateCommandStateUpgradeWeapons",
    "translateCommandStateDontUpgradeWeapons",
multi:
    "translateCommandStateUpgradeWeaponsMode"
}

enum upgradeChasisMode
{
    "translateCommandStateUpgradeChasis",
    "translateCommandStateDontUpgradeChasis",
multi:
    "translateCommandStateUpgradeChasisMode"
}

enum upgradeShieldMode
{
    "translateCommandStateUpgradeShield",
    "translateCommandStateDontUpgradeShield",
multi:
    "translateCommandStateUpgradeShieldMode"
}

}

//***** F U N C T I O N S *****
function int SetCurrentTarget(unit uTarget)
{
    m_uCurrTarget = uTarget;
    SetTargetObject(m_ucurrTarget);
    return true;
}

function int FindTargetToRepair()
{
    int i;
    int nTargetsCount;
    unit newTarget;

    BuildTargetsArray(findTargetWaterUnit | findTargetNormalUnit | findTargetBuildingUnit, findAllUnit | findOurUnit.findDestinationAnyUnit);
    SortFoundTargetsArray();
    nTargetsCount=GetTargetsCount();
    if(nTargetsCount!=0)
    {
        StartEnumTargetsArray();
        for(i=0;i<nTargetsCount;i+=1)
        {
            if(traceMode) TraceD(".");
            newTarget = GetNextTarget();

            if(!newTarget.IsFrozen() && CanBeRepaired(newTarget))
            {
                if(traceMode) TraceD(".");
                m_nMoveToX = GetOperateOnTargetLocationX(newTarget);
                m_nMoveToY = GetOperateOnTargetLocationY(newTarget);
                m_nMoveToZ = GetOperateOnTargetLocationZ(newTarget);

                if(IsGoodPointForOperateOnTarget(newTarget,m_nMoveToX,m_nMoveToY,m_nMoveToZ))
                {
                    if(traceMode) TraceD("!");
                    EndEnumTargetsArray();
                    SetCurrentTarget(newTarget);
                    return true;
                }
            }
        }
        EndEnumTargetsArray();
        return false;
    }
    else
    {
        return false;
    }
}

function int FindTargetToCapture()
{
    int i;
    int nTargetsCount;
    unit newTarget;

    BuildTargetsArray(findTargetWaterUnit | findTargetNormalUnit | findTargetBuildingUnit, findEnemyUnit.findDestinationAnyUnit);
    SortFoundTargetsArray();
    nTargetsCount=GetTargetsCount();

    if(nTargetsCount!=0)
    {
        if(traceMode) TraceD("T>0");
        StartEnumTargetsArray();
        for(i=0;i<nTargetsCount;i+=1)
        {
            newTarget = GetNextTarget();
            if(CanBeConverted(newTarget) && (DistanceTo(newTarget.GetLocationX(),newTarget.GetLocationY())<7))
            {
                m_nMoveToX = GetOperateOnTargetLocationX(newTarget);
                m_nMoveToY = GetOperateOnTargetLocationY(newTarget);
                m_nMoveToZ = GetOperateOnTargetLocationZ(newTarget);
            }
        }
    }
}

```

```

if(IsGoodPointForOperateOnTarget(newTarget,GetOperateOnTargetLocationX(newTarget),
GetOperateOnTargetLocationY(newTarget),GetOperateOnTargetLocationZ(newTarget)))
{
    EndEnumTargetsArray();
    SetCurrentTarget(newTarget);
    return true;
}
}

EndEnumTargetsArray();
return false;
}

else
{
    return false;
}

//***** STATES *****
//***** STATES *****
//***** STATES *****

state Initialize;
state Nothing;
state StartMoving;
state Moving;
state MovingToTarget;
state Repairing;
state Converting;
state Repainting;
state Upgrading;

state Initialize
{
    return Nothing;
}

state Nothing
{
    if(traceMode) TraceD("N");

    if(IsMoving())
    {
        if(traceMode) TraceD(" IsMoving\n");
        return Nothing;
    }
    if(InPlatoon())
    {
        m_nStayX = GetLocationX();
        m_nStayY = GetLocationY();
        m_nStayZ = GetLocationZ();
    }

    if(repairMode)
    {
        if(FindTargetToRepair())
        {
            m_nCurrOperation = operationRepair;
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            if(traceMode) TraceD("-> MTT\n");
            return MovingToTarget;
        }
    }
    if(captureMode)
    {
        if(traceMode) TraceD("captureMode\n");
        if(FindTargetToCapture())
        {
            if(traceMode) TraceD("Target found\n");
            m_nCurrOperation = operationCapture;
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return MovingToTarget;
        }
    }
    if(InPlatoon())
    {
        if(traceMode) TraceD(" IP\n");
        return Nothing;
    }
    if((m_nStayX && (GetLocationX() != m_nStayX || GetLocationY() != m_nStayY
    || GetLocationZ() != m_nStayZ))
}
}

if(traceMode) TraceD("-> M
SetCurrentTarget(null);
m_nMoveToX = m_nStayX;
m_nMoveToY = m_nStayY;
m_nMoveToZ = m_nStayZ;
CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
return StartMoving;
}

if(traceMode) TraceD("\n");
return Nothing;
}

state StartMoving
{
    return Moving, 20;
}
//-----
state Moving
{
    if (!IsMoving())
    {
        return Moving;
    }
    else
    {
        NextCommand(1);
        return Nothing;
    }
}

state MovingToTarget
{
    if (!IsMoving())
    {
        //!!sprawdzanie czy cel jeszcze mozna naprawic/zdisablowlac i czy sie
        ruszyl (wtedy trzeba zmienic kierunek jazdy)
        if ((m_uCurrTarget.IsFrozen() &&
            ((m_nCurrOperation == operationRepair) &&
            CanBeRepaired(m_uCurrTarget)) ||
            ((m_nCurrOperation == operationCapture) &&
            CanBeConverted(m_uCurrTarget)) ||
            ((m_nCurrOperation == operationRepaint) &&
            CanBeRepainted(m_uCurrTarget)) ||
            ((m_nCurrOperation == operationUpgrade) &&
            CanBeUpgraded(m_uCurrTarget))) ||
            ((m_uCurrTarget.m_nLastMoveTime + 1000) <= m_uNow)
        )
        {
            if(!IsGoodPointForOperateOnTarget(m_uCurrTarget,m_nMoveToX,m_nMoveToY,m_nMoveToZ))
            {
                if(m_uCurrTarget.m_nLastMoveTime + 1000 <= m_uNow)
                {
                    m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
                    m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
                    m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
                }
            }
            if(IsGoodPointForOperateOnTarget(m_uCurrTarget,m_nMoveToX,m_nMoveToY,m_nMoveToZ))
            {
                CallMoveToPointForce(m_nMoveToX, m_nMoveToY,
                m_nMoveToZ);
                return MovingToTarget;
            }
            else
            {
                //!unable to find operation point for target
                m_nCurrOperation = operationRepair;
                CallStopMoving();
                SetCurrentTarget(null);
                NextCommand(1);
                return Nothing;
            }
        }
        else
        {
            m_nCurrOperation = operationRepair;
            CallStopMoving();
            SetCurrentTarget(null);
            NextCommand(1);
            return Nothing;
        }
    }
    return MovingToTarget;
}
else
{
}
}

```

```

    {
        //sprawdzic czy w punkcie w ktorym jessemysmo mozemy zaczac naprawe
        if (!IsGoodPointForOperateOnTarget(m_uCurrTarget))
        {
            if (m_nCurrOperation == operationRepair)
            {
                if (CanBeRepaired(m_uCurrTarget))
                {
                    CallRepair(m_uCurrTarget);
                    return Repairing;
                }
                else
                {
                    SetCurrentTarget(null);
                    NextCommand(1);
                    return Nothing;
                }
            }
            else if (m_nCurrOperation == operationCapture)
            {
                if (CanBeConverted(m_uCurrTarget))
                {
                    CallConvert(m_uCurrTarget);
                    return Converting;
                }
                else
                {
                    SetCurrentTarget(null);
                    NextCommand(1);
                    return Nothing;
                }
            }
            else if (m_nCurrOperation == operationRepaint)
            {
                if (CanBeRepainted(m_uCurrTarget))
                {
                    CallRepaint(m_uCurrTarget, m_nRepaintSideColor);
                    return Repainting;
                }
                else
                {
                    SetCurrentTarget(null);
                    NextCommand(1);
                    return Nothing;
                }
            }
            else
            {
                assert m_nCurrOperation == operationUpgrade;
                if (CanBeUpgraded(m_uCurrTarget))
                {
                    CallUpgrade(m_uCurrTarget);
                    return Upgrading;
                }
                else
                {
                    SetCurrentTarget(null);
                    NextCommand(1);
                    return Nothing;
                }
            }
        }
        else
        {
            m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
            m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
            m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
            if (!IsGoodPointForOperateOnTarget(m_uCurrTarget, m_nMoveToX, m_nMoveToY, m_nMoveToZ))
            {
                CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
                return MovingToTarget;
            }
            else
            ///unable to find operation point for target
            m_nCurrOperation = operationRepair;
            CallStopMoving();
            SetCurrentTarget(null);
            NextCommand(1);
            return Nothing;
        }
    }
    else

```

```

        }

state Upgrading
{
    if (IsUpgrading())
    {
        return Upgrading.5;
    }
    else
    {
        if (m_uCurrTarget.IsLive() && CanBeUpgraded(m_uCurrTarget))
        {
            //jakiegos powodu jeszcze go nie zupgradeowalemis - odjechal ?
            m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
            m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
            m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return MovingToTarget;
        }
        else
        {
            SetCurrentTarget(null);
            NextCommand(1);
            return Nothing;
        }
    }
}

//-----
state Frozen
{
    if (IsFrozen())
    {
        return Frozen;
    }
    else
    {
        //!!wrocic do tego co robilismy
        return Nothing;
    }
}

//=====
event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    CallFreeze(nFreezeTicks);
    state Frozen;
    true;
}

//-----
command Initialize()
{
    int nColor;
    int nMaxColor;
    nMaxColor = GetMaxSideColor();
    m_nRepaintSideColor = GetSideColor();
    for (nColor = m_nRepaintSideColor + 1; nColor <= nMaxColor; nColor = nColor + 1)
    {
        if (IsPlayer(nColor))
        {
            m_nRepaintSideColor = nColor;
            break;
        }
    }
    if (nColor > nMaxColor)
    {
        for (nColor = 0; nColor < m_nRepaintSideColor; nColor = nColor + 1)
        {
            if (IsPlayer(nColor))
            {
                m_nRepaintSideColor = nColor;
                break;
            }
        }
    }
    SetRepaintSideColor(m_nRepaintSideColor);
    repairMode = 1;
    captureMode = 1;
    false;
}

command Uninitialize()
{
    //wykasowac reference
    SetCurrentTarget(null);
    false;
}

/*bez nazwy - wywoływany przez kursor*/
command Repair(unit uTarget) hidden button "translateCommandRepair"
{
    if (CanBeRepaired(uTarget))
    {
        m_nCurrOperation = operationRepair;
        SetCurrentTarget(uTarget);
        m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
        m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
        m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToTarget;
    }
    else
    {
        NextCommand(0);
    }
    true;
}

//-----
/*bez nazwy - wywoływany przez kursor*/
command Convert(unit uTarget) hidden button "translateCommandCapture"
{
    if (CanBeConverted(uTarget))
    {
        m_nCurrOperation = operationCapture;
        SetCurrentTarget(uTarget);
        m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
        m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
        m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToTarget;
    }
    else
    {
        NextCommand(0);
    }
    true;
}

//-----
command Repaint(unit uTarget) button "translateCommandRepaint" description
"translateCommandRepaintDescription" hotkey priority 100
{
    if (CanBeRepainted(uTarget))
    {
        m_nCurrOperation = operationRepaint;
        SetCurrentTarget(uTarget);
        m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
        m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
        m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToTarget;
    }
    else
    {
        NextCommand(0);
    }
    true;
}

//-----
command Upgrade(unit uTarget) button "translateCommandUpgrade" description
"translateCommandUpgradeDescription" hotkey priority 99
{
    if (CanBeUpgraded(uTarget))
    {
        m_nCurrOperation = operationUpgrade;
        SetCurrentTarget(uTarget);
        m_nMoveToX = GetOperateOnTargetLocationX(m_uCurrTarget);
        m_nMoveToY = GetOperateOnTargetLocationY(m_uCurrTarget);
        m_nMoveToZ = GetOperateOnTargetLocationZ(m_uCurrTarget);
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToTarget;
    }
    else
    {
        NextCommand(0);
    }
    true;
}

```

```

command SetUpgradeWeapons(int nMode) hidden button
upgradeWeaponsMode priority 98
{
    if (nMode == -1)
    {
        upgradeWeaponsMode = (upgradeWeaponsMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        upgradeWeaponsMode = nMode;
    }
    SetUpgradeWeapons(upgradeWeaponsMode);
    NextCommand(0);
    true;
}
//-----
command SetUpgradeChasis(int nMode) hidden button upgradeChasisMode
priority 97
{
    if (nMode == -1)
    {
        upgradeChasisMode = (upgradeChasisMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        upgradeChasisMode = nMode;
    }
    SetUpgradeChasis(upgradeChasisMode);
    NextCommand(0);
    true;
}
//-----
command SetUpgradeShield(int nMode) hidden button upgradeShieldMode
priority 96
{
    if (nMode == -1)
    {
        upgradeShieldMode = (upgradeShieldMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        upgradeShieldMode = nMode;
    }
    SetUpgradeShield(upgradeShieldMode);
    NextCommand(0);
    true;
}
//-----
command SetRepaintSideColor(int nNewSideColor) hidden button
"translateCommandSetColor"
{
    m_nRepaintSideColor = nNewSideColor;
    SetRepaintSideColor(m_nRepaintSideColor);
    NextCommand(1);
    true;
}
//-----
command SpecialSetRepaintSideColorDialog() button
"translateCommandSetColor" description "translateCommandSetColorDescription"
hotkey priority 101
{
    //specjalna komenda obslugiwana przez dialog
}
//-----
command Move(int nGx, int nGy, int nLz) button "translateCommandMove" description "translateCommandMoveDescription" hotkey priority 21
{
    SetCurrentTarget(null);
    m_nMoveToX = nGx;
    m_nMoveToY = nGy;
    m_nMoveToZ = nLz;
    m_nStayX = nGx;
    m_nStayY = nGy;
    m_nStayZ = nLz;
    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    state StartMoving;
    true;
}
//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    SetCurrentTarget(null);
    m_nMoveToX = GetEntranceX(uEntrance);
    m_nMoveToY = GetEntranceY(uEntrance);
    m_nMoveToZ = GetEntranceZ(uEntrance);

    CallMoveInsideObject(uEntrance);

    state StartMoving;
    true;
}
}
//-----
command Stop() button "translateCommandStop" description "translateCommandStopDescription" hotkey priority 20
{
    SetCurrentTarget(null);
    CallStopMoving();
    state Nothing;
    true;
}
}
//-----
command UserOneParam0(int nMode) button repairMode description "translateCommandStateRepairModeDescription" priority 190
{
    if (nMode == -1)
    {
        repairMode = (repairMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        repairMode = nMode;
    }
}
}
command UserOneParam1(int nMode) button captureMode description "translateCommandStateCaptureModeDescription" priority 190
{
    if (nMode == -1)
    {
        captureMode = (captureMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        captureMode = nMode;
    }
}

```

```

//-----
command SetLights(int nMode) button lights description
"translateCommandStateLightsModeDescription" hotkey priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
        assert(nMode == 0);
        lights = nMode;
    }
    SetLightsMode(lights);
}
//-----

command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
//-----

/* command UserOneParam9(int nMode) button traceMode priority 255
{
    if (nMode == -1)
    {
        traceMode = (traceMode + 1) % 2;
    }
    else
    {
        assert(nMode == 0);
        traceMode = nMode;
    }
} */
}

Sapper.ec
sapper "translateScriptNameMiner"
{
    int m_nMoveToX;
    int m_nMoveToY;
    int m_nMoveToZ;
    int m_nMinePointX;
    int m_nMinePointY;
    int m_nMinePointZ;
    int m_nState://0-Nothing, 1-Moving, 2-MovingToMinePutPoint
state Initialize;
state Nothing;
state StartMoving;
state Moving;
state MovingToMinePutPoint;
state PuttingMine;
state WaitForMines;

state Initialize
{
    return Nothing;
}

state Nothing
{
    return Nothing;
}

state StartMoving
{
    return Moving, 20;
}
//-----
state Moving
{
    if (!IsMoving())
    {
        return Moving;
    }
    else
    {
        NextCommand(1);
        return Nothing;
    }
}

state MovingToMinePutPoint
{
    if (!IsMoving())
    {
        return MovingToMinePutPoint;
    }
    else
    {
        if ((GetLocationX() == m_nMinePointX) && (GetLocationY() ==
m_nMinePointY) && (GetLocationZ() == m_nMinePointZ))
        {
            if (HaveMines())
            {
                CallPutMine();
                return PuttingMine;
            }
            else
            {
                return WaitForMines;
            }
        }
        else
        {
            //moze zastosowac jakis licznik (po wyzerowaniu ktorego przechodzi-
my do nastepnego punktu)
            //zely w kolko nie wywolywac ponizszego
            CallMoveToPointForce(m_nMinePointX, m_nMinePointY,
m_nMinePointZ);
            return MovingToMinePutPoint;
        }
    }
}

state PuttingMine
{
    if (!IsPuttingMine())
    {
        //jeszcze nie skonczyl
        state PuttingMine;
    }
    else
    {
        if (NextMinePoint())
        {
            m_nMinePointX = GetCurrMinePointX();
            m_nMinePointY = GetCurrMinePointY();
            m_nMinePointZ = GetCurrMinePointZ();
            CallMoveToPointForce(m_nMinePointX, m_nMinePointY,
m_nMinePointZ);
            return MovingToMinePutPoint;
        }
    }
}

```

```

    else
    {
        //nie ma juz wiecej punktow do zaminowania
        NextCommand(1);
        return Nothing;
    }
}

state WaitForMines
{
    if (HaveMines())
    {
        CallPutMine();
        return PuttingMine;
    }
    else
    {
        return WaitForMines;
    }
}

//-----
state Froozen
{
    if (IsFroozen())
    {
        state Froozen;
    }
    else
    {
        if (m_nState == 2)
        {
            CallMoveToPoint(m_nMinePointX, m_nMinePointY, m_nMinePointZ);
            return MovingToMinePutPoint;
        }
        else if (m_nState == 1)
        {
            CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return Moving;
        }
        else
        {
            return Nothing;
        }
    }
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    if (state == Froozen)
    {
        //zostaje poprzedni m_nState
    }
    else if ((state == StartMoving) || (state == Moving))
    {
        m_nState = 1;
    }
    else if ((state == MovingToMinePutPoint) || (state == PuttingMine) || (state == WaitForMines))
    {
        m_nState = 2;
    }
    else
    {
        m_nState = 0;
    }
    CallFreeze(nFreezeTicks);
    state Froozen;
    true;
}

//-----

command Initialize()
{
    //pozwolic dzialkom strzelac samym (o ile sa jakies)
    SetCannonFireMode(-1, 1);
    false;
}

command Uninitialize()
{
    //wykasowac referencje
    false;
}

command MineTerrainClose(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
button "translateCommandMineClose" description
"translateCommandMineCloseDescription" hotkey priority 100
{
    ResetMinePoints();
    if (AddMineAreaClose(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMinePointX = GetCurrMinePointX();
        m_nMinePointY = GetCurrMinePointY();
        m_nMinePointZ = GetCurrMinePointZ();
        CallMoveToPointForce(m_nMinePointX, m_nMinePointY, m_nMinePointZ);
        state MovingToMinePutPoint;
    }
    else
    {
        //nie mozna postawic miny na zadnym punkcie tego obszaru
    }
    true;
}

command MineTerrainMedium(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
button "translateCommandMineMedium" description
"translateCommandMineMediumDescription" hotkey priority 101
{
    ResetMinePoints();
    if (AddMineAreaMedium(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMinePointX = GetCurrMinePointX();
        m_nMinePointY = GetCurrMinePointY();
        m_nMinePointZ = GetCurrMinePointZ();
        CallMoveToPointForce(m_nMinePointX, m_nMinePointY, m_nMinePointZ);
        state MovingToMinePutPoint;
    }
    else
    {
        //nie mozna postawic miny na zadnym punkcie tego obszaru
    }
    true;
}

command MineTerrainFar(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
button "translateCommandMineFar" description
"translateCommandMineFarDescription" hotkey priority 102
{
    ResetMinePoints();
    if (AddMineAreaFar(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMinePointX = GetCurrMinePointX();
        m_nMinePointY = GetCurrMinePointY();
        m_nMinePointZ = GetCurrMinePointZ();
        CallMoveToPointForce(m_nMinePointX, m_nMinePointY, m_nMinePointZ);
        state MovingToMinePutPoint;
    }
    else
    {
        //nie mozna postawic miny na zadnym punkcie tego obszaru
    }
    true;
}

command MineTerrainLine(int nX1, int nY1, int nZ1, int nX2, int nY2, int nZ2)
button "translateCommandMineLine" description
"translateCommandMineLineDescription" hotkey priority 103
{
    ResetMinePoints();
    if (AddMineLine(nX1, nY1, nZ1, nX2, nY2, nZ2))
    {
        m_nMinePointX = GetCurrMinePointX();
        m_nMinePointY = GetCurrMinePointY();
        m_nMinePointZ = GetCurrMinePointZ();
        CallMoveToPointForce(m_nMinePointX, m_nMinePointY, m_nMinePointZ);
        state MovingToMinePutPoint;
    }
    else
    {
        //nie mozna postawic miny na zadnym punkcie tego obszaru
    }
    true;
}

command UserPoint0(int nX, int nY, int nZ) button "translateCommandAddPoint"
description "translateCommandAddPointDescription" hotkey priority 102
{
    //for test
    if (IsCurrentMineArea())
    {
}

```

```

        ResetMinePoints();
    }
    if (AddMinePoint(nX, nY, nZ))
    {
        if ((state != MovingToMinePutPoint) && (state != WaitForMines) &&
(state != PuttingMine))
        {
            m_nMinePointX = GetCurrMinePointX();
            m_nMinePointY = GetCurrMinePointY();
            m_nMinePointZ = GetCurrMinePointZ();
            CallMoveToPointForce(m_nMinePointX, m_nMinePointY,
m_nMinePointZ);
            state MovingToMinePutPoint;
        }
    }
    else
        NextCommand(0);
    true;
}

command Move(int nGx, int nGy, int nLz) button "translateCommandMove" description
"translateCommandMoveDescription" hotkey priority 21
{
    ResetMinePoints();
    m_nMoveToX = nGx;
    m_nMoveToY = nGy;
    m_nMoveToZ = nLz;
    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    state StartMoving;
    true;
}

command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    ResetMinePoints();
    m_nMoveToX = GetEntranceX(uEntrance);
    m_nMoveToY = GetEntranceY(uEntrance);
    m_nMoveToZ = GetEntranceZ(uEntrance);
    CallMoveInsideObject(uEntrance);
    state StartMoving;
    true;
}

command SendSupplyRequest() button "translateCommandSupply" description
"translateCommandSupplyDescription" hotkey priority 27
{
    SendSupplyRequest();
    NextCommand(1);
    true;
}

command Stop() button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    ResetMinePoints();
    CallStopMoving();
    state StartMoving;
    true;
}
//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript" description
"translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}
}

Supplier.ec
supplier "translateScriptNameSupplyTransporter"
{
enum lights
{
    "translateCommandStateLightsAUTO",
    "translateCommandStateLightsON",
    "translateCommandStateLightsOFF",
multi:
    "translateCommandStateLightsMode"
}
enum traceMode
{
    "translateCommandStateTraceOFF",
    "translateCommandStateTraceON",
multi:
    "translateCommandStateTraceMode"
}
int m_nMoveToX;
int m_nMoveToY;
int m_nMoveToZ;

state Initialize;
state Nothing;
state StartMoving;
state Moving;
state MovingToAssemblyPoint;
state MovingToSupplyCenter;
state MovingToObjectForSupply;
state LoadingAmmo;
state PuttingAmmo;

state Initialize
{
    return Nothing;
}

state Nothing
{
    int blsEnd;
    if (HaveObjectsForSupply())
    {
        //kontynujemy zaopatrzenie bo nie mozna zostawic zadnego obiektu
        blsEnd = false;
        while (!blsEnd && !ICanCurrentObjectBeSupplied())
        {
            if (!INextObjectForSupply())
            {
                blsEnd = true;
            }
        }
        if (!blsEnd)
        {
            m_nMoveToX = GetCurrentPutSupplyPositionX();
            m_nMoveToY = GetCurrentPutSupplyPositionY();
            m_nMoveToZ = GetCurrentPutSupplyPositionZ();
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return MovingToObjectForSupply;
        }
    }
    else
    {
        return Nothing;
    }
}
else
{
    return Nothing;
}
}

state StartMoving
{
    return Moving, 20;
}
//-----
state Moving
{
    if(traceMode)TraceD("state Moving
\n");
    if (IsMoving())
    {
        return Moving;
    }
    else
    {
        NextCommand(1);
        return Nothing;
    }
}

state MovingToAssemblyPoint
{
    if(traceMode)TraceD("state MovingToAssemblyPoint
\n");
    if (IsMoving())
    {
        return MovingToAssemblyPoint;
    }
    else
    {
        return Nothing;
    }
}

state MovingToSupplyCenter

```

```

    {
        int nPosX;
        int nPosY;
        int nPosZ;
        if (!IsMoving())
        {
            if ((traceMode)TraceD("M -> SC
                nPosX = GetLocationX();
                nPosY = GetLocationY();
                nPosZ = GetLocationZ();
                if ((nPosX == m_nMoveToX) && (nPosY == m_nMoveToY) && (nPosZ
                == m_nMoveToZ))//added 25.01.2000
                    CallStopMoving();
            )
            return MovingToSupplyCenter;
        }
        else
        {
            nPosX = GetLocationX();
            nPosY = GetLocationY();
            nPosZ = GetLocationZ();
            if ((traceMode)TraceD("M -> SC
                if ((nPosX == m_nMoveToX) && (nPosY == m_nMoveToY) && (nPosZ
                == m_nMoveToZ))
                    if ((traceMode)TraceD("M -> SC OK
                \n");
                    CallLoadAmmo();
                    return LoadingAmmo;
            }
            else
            {
                if ((traceMode)TraceD("M -> SC Again
                \n";
                //CallMoveAndLandToPointForce(m_nMoveToX, m_nMoveToY,
                m_nMoveToZ);
                CallMoveToPointForce(m_nMoveToX, m_nMoveToY,
                m_nMoveToZ);//dodane 09.03.2000
                return MovingToSupplyCenter;
            }
        }
    }

state MovingToObjectForSupply
{
    int nPosX;
    int nPosY;
    int nPosZ;
    int blsEnd;
    if (!IsMoving())
    {
        if (!CanCurrentObjectBeSupplied())
        {
            CallStopMoving();
            return MovingToObjectForSupply;
        }
        CheckCurrentPutSupplyLocation();
        if (m_nMoveToX != GetCurrentPutSupplyPositionX() ||
            m_nMoveToY != GetCurrentPutSupplyPositionY())
        {
            m_nMoveToX = GetCurrentPutSupplyPositionX();
            m_nMoveToY = GetCurrentPutSupplyPositionY();
            m_nMoveToZ = GetCurrentPutSupplyPositionZ();
            CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        }
        //XXXMD zrobic gonienie czolgu
        if ((traceMode)TraceD("state MovingToObjectForSupply 1
        \n");
        return MovingToObjectForSupply;
    }
    else
    {
        if (!IsInGoodCurrentPutSupplyLocation())
        {
            if ((traceMode)TraceD("state MovingToObjectForSupply 2
            CallPutAmmo();
            return PuttingAmmo;
        }
        else
        {
            if ((traceMode)TraceD("state MovingToObjectForSupply 3
            //nie jest w dobrym miejscu
            if (CanCurrentObjectBeSupplied())

```

```

    {
        m_nMoveToX = GetCurrentPutSupplyPositionX();
        m_nMoveToY = GetCurrentPutSupplyPositionY();
        m_nMoveToZ = GetCurrentPutSupplyPositionZ();
        if ((traceMode)TraceD("state MovingToObjectForSupply 4
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        return MovingToObjectForSupply;
    }
}
else
{
    if ((traceMode)TraceD("state MovingToObjectForSupply - Can't be
    supplied\n");
    //nie mozna do niego dojechac (pod ziemią), lub zabity - biezemy
    nastepnego
    blsEnd = false;
    do
    {
        if (!NextObjectForSupply())
        {
            blsEnd = true;
        }
    }
    while (!blsEnd && !CanCurrentObjectBeSupplied());
    if (!blsEnd)
    {
        m_nMoveToX = GetCurrentPutSupplyPositionX();
        m_nMoveToY = GetCurrentPutSupplyPositionY();
        m_nMoveToZ = GetCurrentPutSupplyPositionZ();
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY,
        m_nMoveToZ);
        return MovingToObjectForSupply;
    }
}
else
{
    //nie ma juz obiekow do zaopatrzienia - wracamy do punktu
    zbiorki
    if (GetSupplyCenterBuilding() != null)
    {
        m_nMoveToX = GetSupplyCenterAssemblyPositionX();
        m_nMoveToY = GetSupplyCenterAssemblyPositionY();
        m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
        CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        return MovingToAssemblyPoint;
    }
    else
    {
        return Nothing;
    }
}
}

state LoadingAmmo
{
    int blsEnd;
    if (!IsLoadingAmmo())
    {
        return LoadingAmmo;
    }
    else
    {
        if (HaveObjectsForSupply())
        {
            blsEnd = false;
            while (!blsEnd && !CanCurrentObjectBeSupplied())
            {
                if (!NextObjectForSupply())
                {
                    blsEnd = true;
                }
            }
            if (!blsEnd)
            {
                m_nMoveToX = GetCurrentPutSupplyPositionX();
                m_nMoveToY = GetCurrentPutSupplyPositionY();
                m_nMoveToZ = GetCurrentPutSupplyPositionZ();
                CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
                return MovingToObjectForSupply;
            }
        }
        else
        {
            //nie ma obiekow do zaopatrzenia - wracamy do punktu zbiorki
            if (GetSupplyCenterBuilding() != null)

```

```

    {
        m_nMoveToX = GetSupplyCenterAssemblyPositionX();
        m_nMoveToY = GetSupplyCenterAssemblyPositionY();
        m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
        CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        return MovingToAssemblyPoint;
    }
}
else
{
    return Nothing;
}
}

//nie ma obiekow do zaopatrzenia - wracamy do punktu zbiorki
if (GetSupplyCenterBuilding() != null)
{
    m_nMoveToX = GetSupplyCenterAssemblyPositionX();
    m_nMoveToY = GetSupplyCenterAssemblyPositionY();
    m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    return MovingToAssemblyPoint;
}
else
{
    return Nothing;
}
}

}

state PuttingAmmo
{
    int blsEnd;
    if (!IsPuttingAmmo())
    {
        return PuttingAmmo;
    }
    else
    {
        //sprawdzic czy wrzucono amunicje do biezacego obiektu
        if (WasCurrentObjectSupplied())
        {
            //ok jedziemy do nastepnego
            blsEnd = false;
            do
            {
                if (!INextObjectForSupply())
                {
                    blsEnd = true;
                }
            } while (!blsEnd && !ICanCurrentObjectBeSupplied());
            if (!blsEnd)
            {
                m_nMoveToX = GetCurrentPutSupplyPositionX();
                m_nMoveToY = GetCurrentPutSupplyPositionY();
                m_nMoveToZ = GetCurrentPutSupplyPositionZ();
                CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
                return MovingToObjectForSupply;
            }
        }
        else
        {
            //nie ma juz obiekow do zaopatrzenia - wracamy do punktu zbiorki
            if (GetSupplyCenterBuilding() != null)
            {
                m_nMoveToX = GetSupplyCenterAssemblyPositionX();
                m_nMoveToY = GetSupplyCenterAssemblyPositionY();
                m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
                CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
                return MovingToAssemblyPoint;
            }
            else
            {
                return Nothing;
            }
        }
    }
    //nie wrzucilismy zaopatrzenia do niego bo np. odjechal
    blsEnd = false;
    while (!blsEnd && !ICanCurrentObjectBeSupplied())
    {
        if (!INextObjectForSupply())
    }
}

{
    blsEnd = true;
}
}

if (!blsEnd)
{
    m_nMoveToX = GetCurrentPutSupplyPositionX();
    m_nMoveToY = GetCurrentPutSupplyPositionY();
    m_nMoveToZ = GetCurrentPutSupplyPositionZ();
    CallMoveToPointForce(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    return MovingToObjectForSupply;
}
else
{
    //nie ma obiekow do zaopatrzenia - wracamy do punktu zbiorki
    if (GetSupplyCenterBuilding() != null)
    {
        m_nMoveToX = GetSupplyCenterAssemblyPositionX();
        m_nMoveToY = GetSupplyCenterAssemblyPositionY();
        m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
        CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        return MovingToAssemblyPoint;
    }
    else
    {
        return Nothing;
    }
}
}

//-----
state Frozen
{
    if (!IsFrozen())
    {
        state Frozen;
    }
    else
    {
        //!!wrocić do tego co robilismy
        state Nothing;
    }
}

event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    CallFreeze(nFreezeTicks);
    state Frozen;
    true;
}

event OnKilledSupplyCenterBuilding()
{
    unit uNewSupplyCenter;
    if (!GetSupplyCenterBuilding())//przeniesc do eventu
    {
        uNewSupplyCenter = FindSupplyCenter();
        if ((uNewSupplyCenter != null) &&
SetSupplyCenterBuilding(uNewSupplyCenter))
        {
            //pojedziec do jego punktu zbiorki o ile nie rozwizmy zaopatrzenia
            m_nMoveToX = GetSupplyCenterAssemblyPositionX();
            m_nMoveToY = GetSupplyCenterAssemblyPositionY();
            m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
            CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            return MovingToAssemblyPoint;
        }
    }
}

command Initialize()
{
    false;
}

command Uninitialize()
{
    //wykasowac referencje
    false;
}

command SetTransporterSupplyCenter(unit uSupplyCenter) hidden button
"translateCommandSetSupplyCntr"
{
}

```

```

if((traceMode)TraceD("command SetTransporterSupplyCenter
\n");
if (GetSupplyCenterBuilding() != uSupplyCenter)
{
    if (SetSupplyCenterBuilding(uSupplyCenter))
    {
        //pojechac do jego punktu zbiorki o ile nie rozwozimy zaopatrzenia
        if (!HaveObjectsForSupply())
        {
            m_nMoveToX = GetSupplyCenterAssemblyPositionX();
            m_nMoveToY = GetSupplyCenterAssemblyPositionY();
            m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
            CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
            state MovingToAssemblyPoint;
        }
        NextCommand(1);
    }
    else
    {
        NextCommand(0);
    }
}
else
{
    if (!HaveObjectsForSupply() && (state != MovingToSupplyCenter))
    {
        m_nMoveToX = GetSupplyCenterAssemblyPositionX();
        m_nMoveToY = GetSupplyCenterAssemblyPositionY();
        m_nMoveToZ = GetSupplyCenterAssemblyPositionZ();
        CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
        state MovingToAssemblyPoint;
    }
    NextCommand(1);
}
true;
}

//komenda wydawana tylko przez budynek supplyCenter
command MoveToSupplyCenterForLoading()
{
    if((traceMode)TraceD("command MoveToSupplyCenterForLoading\n");
    if (!HaveObjectsForSupply() && (state != LoadingAmmo))
    {
        m_nMoveToX = GetSupplyCenterLoadPositionX();
        m_nMoveToY = GetSupplyCenterLoadPositionY();
        m_nMoveToZ = GetSupplyCenterLoadPositionZ();
        //CallMoveAndLandToPointForce(m_nMoveToX, m_nMoveToY,
m_nMoveToZ);
        CallMoveToPointForce(m_nMoveToX, m_nMoveToY,
m_nMoveToZ); //dodatajne 09.03.2000
        state MovingToSupplyCenter;
    }
    //komenda wewnetrzna z budynku wiec nie wolamy NextCommand
    true;
}

command Move(int nGx, int nGy, int nLz) hidden button
"translateCommandMove" description "translateCommandMoveDescription" hotkey
priority 21
{
    m_nMoveToX = nGx;
    m_nMoveToY = nGy;
    m_nMoveToZ = nLz;
    CallMoveToPoint(m_nMoveToX, m_nMoveToY, m_nMoveToZ);
    state StartMoving;
    true;
}

command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    m_nMoveToX = GetEntranceX(uEntrance);
    m_nMoveToY = GetEntranceY(uEntrance);
    m_nMoveToZ = GetEntranceZ(uEntrance);
    CallMoveInsideObject(uEntrance);
    state StartMoving;
    true;
}

}

```

true;

}

command Stop() hidden button "translateCommandStop" description "translateCommandStopDescription" hotkey priority 20

{

CallStopMoving();

state Nothing;

true;

}

command Land() button "translateCommandLand" description "translateCommandLandDescription" hotkey priority 31

{

CallLand();

state Nothing;

}

//-----

command SpecialChangeUnitsScript() button "translateCommandChangeScript"

description "translateCommandChangeScriptDescription" hotkey priority 254

{

//special command - no implementation

}

//-----

command SetLights(int nMode) button lights priority 204

{

if (nMode == -1)

{
 lights = (lights + 1) % 3;
 }
 else
 {
 assert(nMode == 0);
 lights = nMode;
 }
 SetLightsMode(lights);
}

}

/* command UserOneParam9(int nMode) hidden button traceMode priority 255

{

if (nMode == -1)

```

{
    traceMode = (traceMode + 1) % 2;
}
else
{
    assert(nMode == 0);
    traceMode = nMode;
}
*/
}

Tank.ec
tank "translateScriptNameTank"
{
    consts
    {
        disableFire=0;
        enableFire=1;
    }

    unit m_uTarget;
    int m_nCannonsCount;
    int m_nTargetGx;
    int m_nTargetGy;
    int m_nTargetLz;
    int m_nStayGx;
    int m_nStayGy;
    int m_nStayLz;
    int m_nSpecialCounter;
    unit m_uSpecialUnit;
    int m_bFindAndDestroyWalls;

    enum lights
    {
        "translateCommandStateLightsAUTO",
        "translateCommandStateLightsON",
        "translateCommandStateLightsOFF",
        multi:
        "translateCommandStateLightsMode"
    }

    enum traceMode
    {
        "translateCommandStateTraceOFF",
        "translateCommandStateTraceON",
        multi:
        "translateCommandStateTraceMode"
    }
//***** F U N C T I O N S *****
//*****
function int SetTarget(unit uTarget)
{
    m_uTarget = uTarget;
    SetTargetObject(uTarget);
    return true;
}
function int GoToPoint()
{
    int nRangeMode;
    int nDx;
    int nDy;
    nRangeMode = IsPointInCannonRange(0,m_nTargetGx, m_nTargetGy,
m_nTargetLz);

    if (nRangeMode == 4) //in range
    {
        if (IsMoving())
            CallStopMoving();
        return false;
    }
    if(nRangeMode == 1) //w zasiegu ale trzeba odwrocić czolg
    {
        if (IsMoving())
            CallStopMoving();
        else
            CallTurnToAngle(GetCannonAngleToPoint(0,m_nTargetGx,
m_nTargetGy, m_nTargetLz));
    }
    return true;
}
CallMoveToPoint(m_nTargetGx, m_nTargetGy, m_nTargetLz);
return true;
}

//-----
function int GoToTarget()
{
    int nRangeMode;
    nRangeMode = IsTargetInCannonRange(0, m_uTarget);

    if (nRangeMode == inRangeGoodHit)
    {
        if(traceMode) TraceD("GoToTarget: In range
\n");
        if (IsMoving())
            CallStopMoving();
        return false;
    }
    if(nRangeMode == inRangeBadAngleAlpha) //w zasiegu ale trzeba odwrocić czolg
    {
        if(traceMode) TraceD("GoToTarget: Rotating tank");
        CallTurnToAngle(GetCannonAngleToTarget(0,m_uTarget));
        return true;
    }
    CallMoveToPoint(m_uTarget.GetLocationX(), m_uTarget.GetLocationY(),
m_uTarget.GetLocationZ());
    return true;
}

//-----
function int EndState()
{
    SetCannonFireMode(-1, disableFire);
    NextCommand(1);
}

//-----
function int FindBestTarget()
{
    int i;
    int nTargetsCount;
    unit newTarget;

    if(GetCannonType(0) != cannonTypelon)
    {
        if(!CanCannonFireToAircraft(-1))
        {
            SetTarget(FindClosestEnemyUnitOrBuilding(findTargetWaterUnit |
findTargetNormalUnit));
        }
        else
        {
            SetTarget(FindClosestEnemyUnitOrBuilding(findTargetWaterUnit |
findTargetNormalUnit | findTargetFlyingUnit));
        }
        return true;
    }
    SetTarget(null);
}

BuildTargetsArray(findTargetWaterUnit | findTargetNormalUnit | findTargetBuildingUn
it, findEnemyUnit, findDestinationAnyUnit);
SortFoundTargetsArray();
nTargetsCount=GetTargetsCount();
if(nTargetsCount!=0)
{
    StartEnumTargetsArray();
    for(i=0;i<nTargetsCount;i+=1)
    {
        newTarget = GetNextTarget();
        if((newTarget.IsEnabled()))
        {
            EndEnumTargetsArray();
            SetTarget(newTarget);
        }
    }
}
}


```

```

        return true;
    }
}
EndEnumTargetsArray();
return false;
}
else
{
    return false;
}
}***** S T A T E S *****
state Nothing;
state InPlatoonState;
state StartMoving;
state Moving;
state AutoAttacking;
state Attacking;
state AttackingPoint;
state Retreat;
state Retreat
{
    if(IsMoving())
        return Retreat,20;
    m_uTarget = FindClosestEnemy();
    SetTargetObject(null);
    if(m_uTarget)
    {
        if(IsMoving())
            CallStopMoving();
        m_uTarget = null;
        SetTargetObject(null);
        if(traceMode)TraceD("R->N
        return Nothing;
    }
    CallMoveToPoint(2*GetLocationX()-
m_uTarget.GetLocationX(),2*GetLocationY()-
m_uTarget.GetLocationY(),2*GetLocationZ());
        if(traceMode)TraceD("R u
        return Retreat,100;
    }
}
state InPlatoonState
{
    if(traceMode) TraceD("IP\n");
    if(!InPlatoon())
    {
        SetLightsMode(lights);
        SetCannonFireMode(-1, disableFire);
        return Nothing;
    }
    return InPlatoonState,40;
}
state Nothing
{
    if(traceMode)TraceD("N
    if(!InPlatoon())
    {
        SetCannonFireMode(-1, enableFire);
        return InPlatoonState;
    }
    SetTarget(GetAttacker());
    ClearAttacker();
    if(m_uTarget)
    {
        FindBestTarget();
    }
    if (m_uTarget != null)
    {
        m_nStayGx = GetLocationX();
        m_nStayGy = GetLocationY();
        m_nStayLz = GetLocationZ();
        return AutoAttacking;
    }
    return Nothing;
}
}-----state AutoAttacking
{
    int nDistance;
    if(traceMode)TraceD("AA
        \n");
    // pozostawaj w okolicach punktu
    nDistance = DistanceTo(m_nStayGx,m_nStayGy);
    if( nDistance > 12 )
    {
        if(traceMode)TraceD("nDistance: > 12 !!!!\n ");
        SetTarget(null);
        CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
        SetCannonFireMode(-1, enableFire);
        return Moving;
    }
    if(m_uTarget.IsLive() || (GetCannonType(0) == cannonTypePelon &&
m_uTarget.IsEnabled()))
    {
        StopCannonFire(-1);
        SetTarget(null);
    }
    if (m_uTarget)
    {
        if(GoToTarget())
        {
            if(traceMode)TraceD("AA Fire!!!!\n");
            CannonFireToTarget(-1, m_uTarget, -1);
            if(GetAttacker()!=null){ SetTarget(GetAttacker()); ClearAttacker(); }
        }
        return AutoAttacking;
    }
    else//target not exist
    {
        SetTarget(GetAttacker());
        ClearAttacker();
        if(m_uTarget)
        {
            FindBestTarget();
        }
        if (m_uTarget != null)
            return AutoAttacking;
    }
    if( nDistance > 0 )
    {
        CallMoveToPoint(m_nStayGx, m_nStayGy, m_nStayLz);
        SetCannonFireMode(-1, enableFire);
        return Moving;
    }
    if (IsMoving())
    {
        CallStopMoving();
    }
    return Nothing;
}
}-----state AttackingPoint
{
    if(traceMode)TraceD("AG
        \n");
    if(GoToPoint())
    {
        return AttackingPoint;
    }
    else
    {
        CannonFireGround(-1, m_nTargetGx, m_nTargetGy, m_nTargetLz, 1);
        return AttackingPoint;
    }
}
}-----state Attacking
{
    if(traceMode)TraceD("A
        \n");
    if (m_uTarget.IsLive())
    {
        if(GoToTarget())

```

```

    {
        return Attacking;
    }
    else
    {
        CannonFireToTarget(-1, m_uTarget, -1);
        return Attacking;
    }
}
else //target not exist
{
    SetTarget(null);
    if (IsMoving())
    {
        CallStopMoving();
    }
    EndState();
    return Nothing;
}

//-----
state StartMoving
{
    return Moving, 20;
}
//-----
state Moving
{
/*ten kawalek jest teraz w plutonie
SetTarget(GetAttacker());
ClearAttacker();*/
    if(!m_uTarget)
    {
        FindBestTarget();
    }

    if (m_uTarget != null)
    {
        return AutoAttacking;
    }/*
    if (IsMoving())
    {
        if(traceMode) TraceD("Moving\n");
        return Moving;
    }
    else
    {
        if(traceMode) TraceD("Moving -> N\n");
        EndState();
        return Nothing;
    }
}
//-----
state Frozen
{
    if (IsFrozen())
    {
        state Frozen;
    }
    else
    {
        //!!wrocic do tego co robilismy
        state Nothing;
    }
}

//*****E V E N T S *****
//*****
//zwrocaj true
//false jak nie ma
event OnHit()
{
    if(GetAmmoCount() && CannonRequiresSupply(-1))
    {
        SetCannonFireMode(-1, enableFire);
        if(traceMode)TraceD("NoAmmo -> R\n");
        state Retreat;
    }
    true;
}

//-----
event OnCannonLowAmmo(int nCannonNum)
{
    true;
}
//-----
event OnCannonNoAmmo(int nCannonNum)
{
    if(CannonRequiresSupply(nCannonNum))// && !InPlatoon()
    {
        m_nSpecialCounter=0;
        SetCannonFireMode(-1, enableFire);
        if(traceMode)TraceD("NoAmmo -> R\n");
        state Retreat;
    }
    true;
}
//-----
event OnCannonFoundTarget(int nCannonNum, unit uTarget)
{
    if(GetCannonType(nCannonNum) == cannonTypePelon)
    {
        if(uTarget.IsEnabled())
        {
            return true;
        }
    }
    return false;//gdby zwrocic true to dzialko nie strzeli
}
//-----
event OnCannonEndFire(int nCannonNum, int nEndStatus)//gdy zniszczony,
poza zasiegiem lub brak amunicji
{
    false;
}
//-----
event OnFreezeForSupplyOrRepair(int nFreezeTicks)
{
    CallFreeze(nFreezeTicks);
    state Froozen;
    true;
}
//-----
event OnTransportedToNewWorld()
{
    StopCannonFire(-1);
    SetCannonFireMode(-1, disableFire);
    SetTarget(null);
    m_uSpecialUnit = null;
}
//-----
event OnConvertedToNewPlayer()
{
    StopCannonFire(-1);
    SetCannonFireMode(-1, disableFire);
    SetTarget(null);
    ClearAttacker();
    m_uSpecialUnit = null;
    state Nothing;
}

//***** C O M M A N D S *****
//*****
command Initialize()
{
    m_nCannonsCount = GetCannonsCount();
    traceMode = 0;
    SetCannonFireMode(-1, disableFire);
}
//-----
command Uninitialize()
{
    //wykasowac referencje
    SetTarget(null);
    m_uSpecialUnit = null;
}
//-----
command SetLights(int nMode) hidden button lights priority 204
{
    if (nMode == -1)
    {
        lights = (lights + 1) % 3;
    }
    else
    {
}

```

```

    assert(nMode == 0);
    lights = nMode;
}
SetLightsMode(lights);
NextCommand(0);
}

//-----
command Stop() hidden button "translateCommandStop" description
"translateCommandStopDescription" hotkey priority 20
{
    SetTarget(null);
    StopCannonFire(-1);
    m_nStayGx = GetLocationX();
    m_nStayGy = GetLocationY();
    m_nStayLz = GetLocationZ();
    if(!isMoving())
        CallStopMoving();
    SetCannonFireMode(-1, disableFire);
    state Nothing;
}

//-----
command Move(int nGx, int nGy, int nLz) hidden button
"translateCommandMove" description "translateCommandMoveDescription" hotkey
priority 21
{
    SetTarget(null);
    m_nStayGx = nGx;
    m_nStayGy = nGy;
    m_nStayLz = nLz;
    SetCannonFireMode(-1, enableFire);
    CallMoveToPoint(nGx, nGy, nLz);
    state StartMoving;
}

//-----
command Attack(unit uTarget) hidden button "translateCommandAttack" des-
cription "translateCommandAttackDescription" hotkey priority 22
{
    if((CanCannonFireToAircraft(-1) || !uTarget.IsHelicopter()) &&
       (GetAmmoCount() || !CannonRequiresSupply(-1)))
    {
        SetTarget(uTarget);
        SetCannonFireMode(-1, enableFire);
        state Attacking;
    }
}

/*komenda nie wystawiana na zewnatrz*/
command AttackOnPoint(int nX, int nY, int nZ) hidden button
"translateCommandAttack"
{
    SetTarget(null);
    m_nTargetGx = nX;
    m_nTargetGy = nY;
    m_nTargetLz = nZ;
    SetCannonFireMode(-1, disableFire);
    state AttackingPoint;
}

//-----
command SendSupplyRequest() hidden button "translateCommandSupply" des-
cription "translateCommandSupplyDescription" hotkey priority 27
{
    SendSupplyRequest();
}

//-----
//-----
command Enter(unit uEntrance) hidden button "translateCommandEnter"
{
    CallMoveInsideObject(uEntrance);
    m_nTargetGx = GetEntranceX(uEntrance);
    m_nTargetGy = GetEntranceY(uEntrance);
    m_nTargetLz = GetEntranceZ(uEntrance);
    SetCannonFireMode(-1, disableFire);
    state StartMoving;
}

//-----
command SpecialChangeUnitsScript() button "translateCommandChangeScript"
description "translateCommandChangeScriptDescription" hotkey priority 254
{
    //special command - no implementation
}

//-----
command UserOneParam0(int nMode)

```

Creating campaign step by step

How to create your own campaign

Beta version 0.20a

Author: Mirek Dymek

Lame polish-english translation by Batonik

Sorry for any mistakes, but my english isn't perfect ;-)

Players - Maps - Scripts - Campaign script - Base scripts - Mission script

UCS campaign - an example.

Players

There are always three players in a campaign. One of them is controlled by the user, while the others are driven by the AI. Players taking part in a campaign have assigned numbers and colours:

UCS: No 1, red,

ED: No 2, green,

LC: No 3, blue,

This should be considered when planning a campaign, writing scripts and making maps in the editor. Continuity of researching throughout the campaign is maintained for main players. There can be other opponents on some missions, but their researches start from the beginning and such players are deleted after the end of the mission.

Maps

During the campaign battlefield consists of four parts called worlds. There are three smaller worlds on which player's bases are placed and there is a world where main action takes place. Each of them is created in the ingame editor. World where bases are placed must have size of "base" (that's obvious :)). Mission map can have any size smaller than "huge". Loading of "huge" map will crash the game.

Mission map (World #0), allowed sizes: base, small, medium, large

UCS base (World #1) size: base

ED base (World #2) size: base

LC base (World #3) size: base

Basic rules for creating maps:

- Maps can have any name, but it's better to keep them self-explainable.
- Maps should contain starting point for each player that is going to be placed in a campaign.
- Mission map should contain landing zone or a building vessel for player controlled by the user. Otherwise the user couldn't enter nor leave the mission.
- There should be at least one building or unit for a player or else he will be considered dead at the start of the mission.
- Players should be assigned to proper races. Races are changed in the window switched by the [?] button on the "Objects" tab.

Each player (human and AI players) contains one "player" object in every world. Most of this object properties are separated for every world but some (like researches) are common for all worlds. Consult included scripts to see how players are used.

Scripts

Campaigns are driven by scripts written in language called "EarthC". It is similar to object oriented C++. Each script consists of following parts: Header (mission or campaign), constants declarations (consts), states declarations (state), states definitions (state) and events definitions(event).

Here is the example of an empty mission script:

```
mission "mission name"                                B header
{
    consts                                         B constants declarations
    {
    }
    state Initialize;                            B states declarations
    state Working;
```

```
state Initialize
{
    return Working;
}
state Working
{
    return Working;
}
```

B states definitions

Mission described above starts with the "Initialize" state. Then it leaves "Initialize", enters the "Working" state and stays in it. There is no way to end this mission.

Mission script

Now we'll discuss the basic mission script for an UCS campaign. Map for this script should contain following items:

- starting point #1 (UCS),
- some objects for player 1 (red),
- starting point #2,
- some objects for player 2 (green),
- race of player 1 will be set to UCS,
- race of player 2 will be set to ED,

The script performs following tasks: at the start in the Initialize state it creates references to players playing the mission. Then it sets starting amounts of money (20 000 CR for each player). Next state Initialize is paused and scripts goes to Working state (command: return Working). In Working state the script checks if players have any buildings or units. If this is true for player 1, function EndMission(false) is executed. The mission ends and is considered lost. If there are no buildings or units of player 2 (the enemy) script calls EnableEndMissionButton(true) function, which activates the "end mission" button and enters state Nothing. Mission stays in this state waiting for the user to press the "end mission" button.

Here is the script:

```
mission "Misja 1"                                     B header
{
    player playerUCS;                                B declarations of variables
    player playerED;

state Initialize;                                    B declaration of states
state Working;
state Nothing;

state Initialize                                     B starting state definition
{
    playerUCS = GetPlayer(1);
    playerED = GetPlayer(2);                         B assigning player references to variables.

    playerUCS.SetMoney(20000);                      B starting money for each player
    playerED.SetMoney(20000);
    return Working;
}

state Working                                         B definition of the main state of the
mission
{
    if( !playerUCS.GetNumberOfUnits() && !playerUCS.GetNumberOfBuildings() )
    {
        AddBriefing("You lost!");
        EndMission(false);
    }

    if( !playerUCS.GetNumberOfUnits() && !playerUCS.GetNumberOfBuildings() )
    {
        AddBriefing("You won!");
        EnableEndMissionButton(true);
        return Nothing;
    }
    return Working;100;                               B 100 means that this state will be
entered again after 5 seconds.
}

state Nothing                                         B definition of the ending state
{
    return Nothing;1200;                            B this state will be entered one time per
minute (1200/20=60 sec).
}
```

Campaign script

You can see examples of campaign script in files CampaignED.ec, CampaignUCS.ec, CampaignLC.ec. Your campaign script name must begin with CampaignED (CampaignUCS, CampaignLC or TutorialED, TutorialUCS, TutorialLC for tutorial campaigns) f.e.: CampaignEDSomeExtension.eco (or TutorialEDSomeExt.eco for tutorial) and must be placed in directory Scripts\Campaigns\ED (or UCS, LC for other races). If your campaign name have name f.e. "translateCampaignEDMySuperCamp" then you can add string "translateCampaignEDMySuperCampDescription" - with short description of your campaign. This string will be displayed in text box in choose campaign dialog. If you want modify Earth campaign remember to use other names for mission scripts - otherwise your new files will cover standard files with unpredictable results.

Base scripts

Base scripts are normal mission files but exist as log as base world - thru all game. Because of game limitations you must have 3 bases in your campaign - one for player, and two (possible without any units) for AI players. If you want to make this two other bases available for player you can do it, but unfortunately there is a bug in game which causes that flight time of units transporter between two bases is very long.

Creating a mission

- 1.1. Create a map with the editor.
- 1.2. Place starting point for player 1 (red) and for other players - enemies
- 1.3. Place Landing Zone for player 1 or an unit, preferably the builder (Mamut)
- 1.4. Place buildings and units of enemies
- 1.5. Write script for mission

Object identifiers

Object identifiers for scripts

Chasis and weapon: ED UCS LC

Buildings: ED UCS LC

Researches: ED UCS LC Neutral

Artifacts: Normal Special

Tables below contains two columns. Identifier in first column is special name representing some object which you can use as parameter for some functions described below. Name in second column is language-specific name for this object displayed in game. Names here are from English version of Earth 2150. Names in other language versions are the same or very similar. Numbers 1,2,3 at the end of identifiers are for upgrades of chasis/weapons/researches.

Chasis and weapon identifiers

This object identifiers can be used to create units from scripts with function player.CreateUnit, or player.CreateUnitEx.

You can only put valid weapon on valid chasis, if you are not sure which weapon can be put on some chasis check it in "Construction Center".

Identifier Name

ED chasis

EDUBU1 Gruz

EDUFAKE1 Fake TT

EDUFAKE2 Fake ZT

EDUFAKE3 Fake HT

EDUST1 TT 100 Pamir

EDUST2 TT 110 Pamir

EDUST3 TT 120 Pamir

EDUMI1 ZT 200 Minelayer

EDUMI2 ZT 210 Minelayer

EDUOH1 ZK Taiga

EDUOH1 ZT 100 Siberia

EDUMT2 ZT 101 Siberia

EDUMT3 ZT 102 Siberia

EDUSPY1 NEO

EDUMW1 TK 100 Caspian

EDUMW2 TK 110 Caspian

EDUMW3 TK 111 Caspian

EDUHW1 TL 70 Volga

EDUHW2 TL 80 Volga

EDUHT1 HT 400 Kruszhev

EDUHT2 HT 500 Kruszhev

EDUHT3 HT 600 Kruszhev

EDUBT1 HT 800 Ural

EDUBT2 HT 900 Ural

EDUSS1 ESS 30 Irkutsk

EDUSS2 ESS 40 Irkutsk

EDUSS3 ESS 50 Irkutsk

EDUBS1 ESS 200 Leviathan

EDUBS2 ESS 300 Leviathan

EDUA11 MI 106 Cossack

EDUA12 MI 107 Cossack

EDUA21 MI 140 Grozny

EDUA22 MI 150 Grozny

EDUA31 MI 200 Han

EDUA32 MI 210 Han

EDUA41 MI 300 Thor

EDUA42 MI 310 Thor

EDUA51 MI 27 Boyar

EDUMM11 Truck

EDUMM21 Truck

EDUMM31 Heavy Tank

EDUMM41 Tank

ED weapons

EDWMM21 150 mm Cannon (for EDUMM21)

EDWMM31 105 mm Cannon (for EDUMM31)

EDWMM41 105 mm Cannon (for EDUMM41)

EDWMM42 105 mm Cannon (for EDUMM41)

EDWCH1 20 mm Chaingun

EDWCH2 20 mm Double Chaingun

EDWC1 105 mm Cannon

EDWC2 105 mm Double Cannon

EDWSR1AB Rocket Launcher (for Pamir)

EDWSR2AB Double Rocket Launcher

EDWSR1 Rocket Launcher

EDWSR2 Rocket Launcher upg.1

EDWSR3 Rocket Launcher upg.2

EDWSL1 Laser Cannon

EDWSL2 Laser Cannon upg.1

EDWSL3 Laser Cannon upg.2

EDWS1 Ion Cannon

EDWS2 Ion Cannon upg.1

EDWHC1 120 mm Cannon

EDWHC2 120 mm Double Cannon

EDWMR1 Heavy Rocket Launcher

EDWMR2 Heavy Rocket Launcher upg.1

EDWMR3 Heavy Rocket Launcher upg.2

EDWHShip1 120 mm Artillery

EDWHShip2 120 mm Double Artillery

EDWMRShip1 Long Range Rocket Launcher

EDWMRShip2 Long Range Rocket Launcher upg.1

EDWMRShip3 Long Range Rocket Launcher upg.2

EDWR1 Ballistic Rocket Launcher

EDWHL1 Heavy Laser Cannon

EDWHL2 Heavy Laser Cannon upg.1

EDWHL3 Heavy Laser Cannon upg.2

EDWH1 Ion Cannon (heavy tank)

EDWH2 Ion Cannon upg. (heavy tank)

EDWAC1 20 mm Chaingun (air unit)

EDWAC2 20 mm Double Chaingun (air unit)

EDWAR1 Rocket Launcher upg.1 (air unit)

EDWAR2 Rocket Launcher upg.2 (air unit)

EDWAMR1 Heavy Rocket Launcher (air unit)

EDWAMR2 Heavy Rocket Launcher upg.1 (air unit)

EDWAB1 Bomb Bay (3 bombs)

EDWAB2 Bomb Bay (5 bombs)

ED special equipment

EDSCA1 Carrier

EDSACA1 Container Hook

EDSCREAMER1 Noise Generator

EDSCREAMER2 Noise Generator upg.1

EDSCREAMER3 Noise Generator upg.2

EDREPAIR1 Repairer

EDREPAIR2 Repairer upg.1

EDREPAIR3 Repairer upg.2

EDBANNER Banner

UCS chasis

UCSUB1 Mammoth

UCSUM1 Minelayer

UCSUM2 Minelayer II

UCSUL1 Tiger

UCSUL2 Tiger II

UCSUL3 Tiger III

UCSUM1 Spider

UCSUM2 Spider II

UCSUM3 Spider III

UCSUL1 Panther

UCSUL2 Panther II

UCSUL3 Panther III

UCSUBL1 Jaguar

UCSUBL2 Jaguar II

UCSUL41 Grizzly

UCSUL42 Grizzly II

UCSUL43 Grizzly III

UCSSU1 Shark

UCSSU2 Shark II

UCSSU3 Shark III

UCSUS1 Hydra

UCSUS2 Hydra II

UCSUS3 Hydra III

UCSUA11 Gargoyle

UCSUA12 Gargoyle II

UCSUA13 Gargoyle III

UCSUA21 Bat

UCSUA22 Bat II

UCSUA31 Dragon

UCSUA32 Dragon II

UCSUS1 Condor

UCSUO1 Harvester

UCSUO2 Harvester II

UCSUO3 Harvester III

UCSUA1 Harvester IV

UCSUA2 Harvester V

UCSUA3 Harvester VI

UCS weapons

UCSWTC1 20 mm Chaingun (Tiger)

UCSWTC2 20 mm Chaingun upg.1 (Tiger)

UCSWTSP1 Plasma Cannon (Tiger)

UCSWTSP2 Plasma Cannon upg.1 (Tiger)

UCSWTSR1 Rocket Launcher (Tiger)

UCSWTSR2 Rocket Launcher upg.1 (Tiger)

UCSWTSR3 Rocket Launcher upg.2 (Tiger)

UCSWBMR1 Heavy Rocket Launcher (Panter/Jaguar)

UCSWBMR2 Heavy Rocket Launcher upg.1 (Panter/Jaguar)

UCSWBMR3 Heavy Rocket Launcher upg.2 (Panter/Jaguar)

UCSWBSR1 Rocket Launcher (Panter/Jaguar)

UCSWBSR2 Rocket Launcher upg.1 (Panter/Jaguar)

UCSWBSR3 Rocket Launcher upg.2 (Panter/Jaguar)

UCSWBHP1 Heavy Plasma Cannon

UCSWBHP2 Heavy Plasma Cannon upg.1
 UCSWBHP3 Heavy Plasma Cannon upg.2
 UCSWSCH1 20 mm chaingun (Spider)
 UCSWSCH2 20 mm chaingun (Spider)
 UCSWTSG1 Grenade Launcher (Tiger)
 UCSWTSG2 Grenade Launcher (Tiger)
 UCSWBHG1 Grenade Launcher (Panther/Jaguar)
 UCSWBHG2 Grenade Launcher (Panther/Jaguar)
 UCSWSSP1 Plasma Cannon (Spider)
 UCSWSSP2 Plasma Cannon (Spider)
 UCWSRSR1 Rocket Launcher (Spider)
 UCWSRSR2 Rocket Launcher (Spider)
 UCWSRSR3 Rocket Launcher (Spider)
 UCWSMR1 Heavy Rocket Launcher (Spider/Jaguar)
 UCWSMR2 Heavy Rocket Launcher (Spider/Jaguar)
 UCSDSR1 Rocket Launcher (Grizzly/Hydra)
 UCSDSR2 Rocket Launcher (Grizzly/Hydra)
 UCSDSR3 Rocket Launcher (Grizzly/Hydra)
 UCSDWDR1 Heavy Rocket Launcher (Grizzly/Hydra)
 UCSDWDR2 Heavy Rocket Launcher (Grizzly/Hydra)
 UCSDHHP1 Heavy Plasma Cannon (Grizzly/Hydra)
 UCSDHHP2 Heavy Plasma Cannon (Grizzly/Hydra)
 UCSWACH1 20 mm Chaingun (Gargoyle)
 UCSWACH2 20 mm Double Chaingun (Gargoyle)
 UCWSAR1 Rocket Launcher (Gargoyle)
 UCWSAR2 Rocket Launcher upg.1 (Gargoyle)
 UCSWAMR1 Heavy Rocket Launcher (Bat/Dragon)
 UCSWAMR2 Heavy Rocket Launcher upg.1 (Bat/Dragon)
 UCSWAPB1 Bomb Bay (Bat/Dragon)
 UCSWAPB2 Bomb Bay upg.1 (Bat/Dragon)
 UCS special equipment
 UCWSH1 Shadow Generator
 UCWSH2 Shadow Generator
 UCWSH3 Shadow Generator
 UCSRadar1 Radar
 UCSRRepair1 Repairer
 UCSRRepair2 Repairer upg.1
 UCBANNER Banner
 LC chassis
 LCULU1 Lunar m1
 LCULU2 Lunar m2
 LCULU3 Lunar m3
 LCOMO1 Moon m1
 LCOMO2 Moon m2
 LCOMO3 Moon m3
 LCUCR1 Crater m1
 LCUCR2 Crater m2
 LCUCR3 Crater m3
 LCUCU1 Crusher m1
 LCUCU2 Crusher m2
 LCUCU3 Crusher m3
 LCUH1 Cron
 LCUHERO Fang
 LCUOB1 Phobos
 LCOME1 Meteor m1
 LCOME2 Meteor m2
 LCOME3 Meteor m3
 LCUBO1 Thunderer m1
 LCUBO2 Thunderer m2
 LCUS1 Mercury
 LCUIUO Alien craft
 PROTOTYPE PROTOTYPE
 LC weapons
 LCWH1 20 mm Chaingun
 LCWH2 20 mm Double Chaingun
 LCWSR1 Rocket Launcher
 LCWSR2 Rocket Launcher upg.1
 LCWSR3 Rocket Launcher upg.2
 LCWSL1 Electro-Cannon
 LCWSL2 Electro-Cannon upg.1
 LCWS1 Sonic Cannon
 LCWS2 Sonic Cannon upg.1
 LCWMR1 Heavy Rocket Launcher
 LCWMR2 Heavy Rocket Launcher upg.1
 LCWMR3 Heavy Rocket Launcher upg.2
 LCWLH1 Heavy Electro-Cannon
 LCWLH2 Heavy Electro-Cannon upg.1
 LCWSH1 Heavy Sonic Cannon
 LCWSH2 Heavy Sonic Cannon upg.1
 LCWHC1 Plasma Cannon
 LCWHERO Plasma Beam Projector
 LCWAC1 Chaingun (air unit)
 LCWAC2 Chaingun upg.1 (air unit)
 LCWAR1 Rocket Launcher (air unit)
 LCWAR2 Rocket Launcher upg.1 (air unit)
 LCWAMR1 Heavy Rocket Launcher (air unit)

LLOWAMR2 Heavy Rocket Launcher upg.1 (air unit)

LCWAS1 Heavy Sonic Cannon (air unit)

LCWAS2 Heavy Sonic Cannon upg.1 (air unit)

LCWUFO Alien Weapon

LC special equipment

LCSOB1 Detector m1

LCSOB2 Detector m2

LCSSHRR2 Shield Recharger m2

LCSSHRR3 Shield Recharger m3

LCSREG1 Regenerator m1

LCSREG2 Regenerator m2

LCSREG3 Regenerator m3

LCBANNER Banner

Building identifiers

Building identifiers can be used to enable or disable build of some types of buildings in mission or base by function player.EnableBuilding. Note that calling this function for some special buildings (like silos or solar battery) doesn't have effect. Also AI does not takes care of this function so you can only disable buildings for human players not for AI players.

Identifier Name

ED buildings

EDBBA Vehicle Production Center

EDBFA Weapons Production Center

EDBPP Power Plant

EDBMI Mine

EDBRE Refinery

EDBTC Transport Base

EDBRC Research Center

EDBAB Supply Depot

EDBWB Ship Yard

EDBPP Pillbox

EDBST Small Tower

EDBBT Large Tower

EDBBC Missile Control Center

EDBHQ Headquarters

EDBSI Silo

EDBRA Radar

EDBEN1 Tunnel entrance

EDBEN2 Tunnel entrance

EDBSS Space port

EDBLZ Landing Zone

EDBBBA Vehicle Production Center

EDBBFA Weapons Production Center

EDBBP Power Plant

EDBBMI Mine

EDBBRE Refinery

EDBBTC Transport Base

EDBBRC Research Center

EDBAB Supply Depot

EDBBWB Ship yard

EDBBPP Pillbox

EDBBST Small Tower

EDBBBT Large Tower

EDBBC Missile Control Center

EDBBHQ Headquarters

EDBBSI Silo

EDBBRA Radar

EDBBEN1 Tunnel entrance

EDBBEN2 Tunnel entrance

EDBBS Space port

EDBLZ Landing Zone

UCS buildings

UCSBA Vehicle Production Center

UCSBA Weapons Production Center

UCSBP Power Plant

UCSPR Nuclear Reactor

UCSBT Ore Transport Base

UCSBET Energy Transmitter

UCSBR Refinery

UCSRC Research Center

UCSAB Aerial Supply Depot

UCSBWB Ship Yard

UCSBFO Fortress

UCSBST Small Tower

UCSBBT Large Tower

UCSPBP Plasma Control Center

UCSPBC Plasma Cannon

UCSBHQ Headquarters

UCSBD SDI Defense Center

UCSBSH Shadow Tower

UCSBTE Teleport

UCSBEN1 Tunnel Entrance

UCSBEN2 Tunnel Entrance

UCSBSS Space Port

UCSBLZ Landing Zone
LC buildings
LBBF Main Base
LBBP Solar Power Plant
LBBA Solar Battery
LBM Mine
LBSR Ore Transport Refinery
LCBRC Research Center
LCBAB Aerial Supply Center
LCBGA Guardian
LCBDE Defender
LCBHQ Headquarters
LCBSD SDI Defense Center
LCBWC Weather Control Center
LCBSS Space Port
LCBSB Solar Cell
LCBLZ Landing Zone
WLASER Laser Wall

Research identifiers

Research identifiers can be used to enable or disable research production (disabled research can't be researched) with function player.EnableResearch and also to researches for player (research is available without production) with function player.AddResearch. If you are using function AddResearch remember that researches must be add in specific order: from beginning (root) of research tree. So if you want to give player f.e. Kruschev tank (research RES_ED_UHT1) add researches in following order:

rPlayer.AddResearch(RES_ED_UST2);
rPlayer.AddResearch(RES_ED_UST3);
rPlayer.AddResearch(RES_ED_UHT1);

Also you should call AddResearch function only at begin of mission script (in first state), and in campaign use this function combined with EnableResearch (to avoid giving already produced researches).

Neutral researches are available for all races.

Identifier Name

ED researches

RES_ED_UST2 Upg: Pamir

RES_ED_UST3 Upg: Pamir

RES_ED_UMT1 Siberia

RES_ED_UMT2 Upg: Siberia

RES_ED_UMT3 Upg: Siberia

RES_ED_UM1 Mine Layer

RES_ED_UM2 Upg: Mine Layer

RES_ED_UMW1 Caspian

RES_ED_UMW2 Upg: Caspian

RES_ED_UMW3 Upg: Caspian

RES_ED_UHW1 Volga

RES_ED_UHW2 Upg: Volga

RES_ED_UHT1 Kruschev

RES_ED_UHT2 Upg: Kruschev

RES_ED_UHT3 Upg: Kruschev

RES_ED_UTB1 Ural

RES_ED_UTB2 Upg: Ural

RES_ED_USS2 Upg: Irkutsk

RES_ED_USS3 Upg: Irkutsk

RES_ED_UHS1 Leviathan

RES_ED_UHS2 Upg: Leviathan

RES_ED_UA11 Cossack

RES_ED_UA12 Upg: Cossack

RES_ED_UA21 Grozny

RES_ED_UA22 Upg: Grozny

RES_ED_UA31 Han

RES_ED_UA32 Upg: Han

RES_ED_UA41 Thor

RES_ED_UA42 Upg: Thor

RES_ED_WCH2 Upg: Chaingun

RES_ED_ACH2 Upg: Helicopter Chaingun

RES_ED_WCA2 Upg: Cannon

RES_ED_WH1 Heavy Cannon

RES_ED_WH2 Upg: Heavy Cannon

RES_ED_WSR1 Rocket Launcher

RES_ED_WSR2 Upg: Rocket Launcher

RES_ED_WSR3 Upg: Rocket Launcher II

RES_ED_ASR1 Helicopter Rocket Launcher

RES_ED_ASR2 Upg: Helicopter Rocket Launcher

RES_ED_WMR1 Heavy Rocket Launcher

RES_ED_WMR2 Upg: Heavy Rocket Launcher

RES_ED_WMR3 Upg: Heavy Rocket Launcher II

RES_ED_AMR1 Helicopter Heavy Rocket Launcher

RES_ED_AMR2 Upg: Helicopter Heavy Rocket Launcher

RES_ED_WHR1 Ballistic Rocket Launcher

RES_ED_WSL1 Laser Cannon

RES_ED_WSL2 Upg: Laser Cannon

RES_ED_WSL3 Upg: Laser Cannon

RES_ED_WHL1 Heavy Laser Cannon

RES_ED_WHL2 Upg: Heavy Laser Cannon
RES_ED_WHL3 Upg: Heavy Laser Cannon
RES_ED_WSI1 Ion Cannon
RES_ED_WSI2 Upg: Ion Cannon
RES_ED_WH1 Heavy Ion Cannon
RES_ED_WH2 Upg: Heavy Ion Cannon
RES_ED_AB1 Bomb Bay
RES_ED_AB2 Upg: Bomb Bay
RES_ED_BMD Medium Defense Building
RES_ED_BHD Heavy Defense Building
RES_ED_SGen Power Shield Generator 600 PSU
RES_ED_MGen Power Shield Generator 1200 PSU
RES_ED_HGen Power Shield Generator 1800 PSU
RES_ED_Repland Repairer
RES_ED_RepHard2 Upg: Repairer
RES_ED_SCR Screamer (noise generator)
RES_ED_SCR2 Upg: Screamer (noise generator)
RES_ED_SCR3 Upg: Screamer II (noise generator)
RES_ED_MSC2 Upg: 105 mm Bullet
RES_ED_MSC3 Upg: 105 mm Bullet
RES_ED_MSC4 Upg: 105 mm Bullet
RES_ED_MHC2 Upg: 120 mm Bullet
RES_ED_MHC3 Upg: 120 mm Bullet
RES_ED_MH4C Upg: 120 mm Bullet
RES_ED_MHR2 Upg: Ballistic Rocket
RES_ED_MHR3 Upg: Ballistic Rocket
RES_ED_MHR4 Nuclear Ballistic Rocket
RES_ED_MB2 Upg: Bomb
RES_ED_MB3 Upg: Bomb
RES_ED_MB4 Nuclear Bomb
UCS researches
RES_UCS_USL2 Upg: Tiger I
RES_UCS_USL3 Upg: Tiger II
RES_UCS_UML1 Spider (6 legged chassis)
RES_UCS_UML2 Upg: Spider I
RES_UCS_UML3 Upg: Spider II
RES_UCS_UHL1 Panther (2 legged heavy chassis)
RES_UCS_UHL2 Upg: Panther I
RES_UCS_UHL3 Upg: Panther II
RES_UCS_U4L1 Grizzly (3 legged heavy chassis)
RES_UCS_U4L2 Upg: Grizzly
RES_UCS_U4L3 Upg: Grizzly II
RES_UCSUBL1 Jaguar (2-legged heavy chassis)
RES_UCSUBL2 Upg: Jaguar
RES_UCS_UM1 Minelayer
RES_UCS_UM2 Upg: Minelayer
RES_UCS_USS1 Shark (small ship)
RES_UCS_USS2 Upg: Shark
RES_UCS_UBS1 Hydra (ship)
RES_UCS_UBS2 Upg: Hydra
RES_UCS_UBS3 Upg: Hydra II
RES_UCS_UOH2 Upg: Harvester
RES_UCS_UOH3 Upg: Harvester II
RES_UCS_UAH1 Upg: Harvester III
RES_UCS_UAH2 Upg: Harvester IV
RES_UCS_UAH3 Upg: Harvester V
RES_UCS_GARG1 Gargoyle (fighter)
RES_UCS_GARG2 Upg: Gargoyle
RES_UCS_GARG3 Upg: Gargoyle II
RES_UCS_BOMBER1 Bat (bomber)
RES_UCS_BOMBER2 Upg: Bat
RES_UCS_BOMBER3 Dragon (heavy bomber)
RES_UCS_BOMBER4 Upg: Dragon
RES_UCS_WCH2 Double Chaingun
RES_UCS_WCH2 Upg Gargoyle Chaingun
RES_UCS_WSG1 Grenade Launcher
RES_UCS_WSG2 Upg: Grenade Launcher
RES_UCS_WHG1 Heavy Grenade Launcher
RES_UCS_WHG2 Upg: Heavy Grenade Launcher
RES_UCS_WSR1 Small Rocket Launchers
RES_UCS_WSR2 Upg: Small Rocket Launchers I
RES_UCS_WSR3 Upg: Small Rocket Launchers II
RES_UCS_WASR1 Gargoyle Rocket Launcher
RES_UCS_WASR2 Upg: Gargoyle Rocket Launcher
RES_UCS_WMR1 Rocket Launchers
RES_UCS_WMR2 Upg: Rocket Launchers I
RES_UCS_WMR3 Upg: Rocket Launchers II
RES_UCS_WAMR1 Bomber Rocket Launcher
RES_UCS_WAMR2 Upg: Bomber Rocket Launcher
RES_UCS_WSP1 Plasma Cannons
RES_UCS_WSP2 Upg: Plasma Cannons
RES_UCS_WHP1 Heavy Plasma Cannons
RES_UCS_WHP2 Upg: Heavy Plasma Cannons I
RES_UCS_WHP3 Upg: Heavy Plasma Cannons II
RES_UCS_WAP1 Bomb Bay
RES_UCS_WAP2 Upg: Bomb Bay



RES_UCS_WSD SDI Laser
 RES_UCS_BMD Medium Defense Building
 RES_UCS_BHD Heavy Defense Building
 RES_UCS_Rephand Repairer
 RES_UCS_Rephand2 Upg: Repairer
 RES_UCS_SGen Power Shield Generator 600 PSU
 RES_UCS_MGen Power Shield Generator 1200 PSU
 RES_UCS_HGen Power Shield Generator 1800 PSU
 RES_UCS_SHD Shadow Generator
 RES_UCS_SHD2 Upg: Shadow Generator I
 RES_UCS_SHD3 Upg: Shadow Generator II
 RES_UCS_SHD4 Upg: Shadow Generator III
 RES_UCS_MB2 Upg: Plasma Bomb
 RES_UCS_MB3 Upg: Plasma Bomb
 RES_UCS_MB4 Upg: Plasma Bomb
 RES_UCS_MG2 Upg: Grenade
 RES_UCS_MG3 Upg: Grenade
 RES_UCS_MG4 Upg: Grenade
 RES_UCS_PC Offensive Plasma Cannon
 LC researches
 RES_LC_ULU2 Upg: Lunar
 RES_LC_ULU3 Upg: Lunar
 RES_LC_UM02 Upg: Moon (medium tank)
 RES_LC_UM03 Upg: Moon
 RES_LC_UCR1 Crater (heavy tank)
 RES_LC_UCR2 Upg: Crater
 RES_LC_UCR3 Upg: Crater
 RES_LC_UCU1 Crusher (two cannon tank)
 RES_LC_UCU2 Upg: Crusher
 RES_LC_UCU3 Upg: Crusher
 RES_LC_UME1 Meteor (fighter)
 RES_LC_UME2 Upg: Meteor
 RES_LC_UME3 Upg: Meteor
 RES_LC_UBO1 Thunderer (bomber)
 RES_LC_UBO2 Upg: Thunderer
 RES_LC_WCH2 Upg: Chaingun
 RES_LC_ACH2 Upg: Air Chaingun
 RES_LC_WSR1 Rocket Launcher
 RES_LC_WSR2 Upg: Rocket Launcher
 RES_LC_WSR3 Upg: Rocket Launcher
 RES_LC_ASRI Air Rocket Launcher
 RES_LC_ASR2 Upg: Air Rocket Launcher
 RES_LC_WMR1 Heavy Rocket Launcher
 RES_LC_WMR2 Heavy Rocket Launcher upg.1
 RES_LC_WMR3 Heavy Rocket Launcher upg.2
 RES_LC_AMR1 Air Heavy Rocket Launcher
 RES_LC_AMR2 Air Heavy Rocket Launcher upg.1
 RES_LC_WSL1 Electro-Cannon
 RES_LC_WSL2 Upg: Electro-Cannon
 RES_LC_WHL1 Heavy Electro-Cannon
 RES_LC_WHL2 Upg: Heavy Electro-Cannon
 RES_LC_WSS1 Sonic Cannon
 RES_LC_WSS2 Upg: Sonic Cannon
 RES_LC_WHIS1 Heavy Sonic Cannon
 RES_LC_WHIS2 Upg: Heavy Sonic Cannon
 RES_LC_WAS1 Air Sonic Cannon
 RES_LC_WAS2 Upg: Air Sonic Cannon
 RES_LC_WARTILLER Plasma Artillery
 RES_LC_BMD Medium Defense Building
 RES_LC_BHD Heavy Defense Building
 RES_LC_BWC Weather Control Center
 RES_LC_SOBI Detector
 RES_LC_SOBJ Upg: Detector
 RES_LC_SHR1 Shield Recharger
 RES_LC_SHR2 Upg: Shield Recharger m1
 RES_LC_SHR3 Upg: Shield Recharger m2
 RES_LC_SGen Power Shield Generator 1200 PSU
 RES_LC_MGen Power Shield Generator 2400 PSU
 RES_LC_HGen Power Shield Generator 3600 PSU
 RES_LC_REG1 Regenerator
 RES_LC_REG2 Upg: Regenerator m1
 RES_LC_REG3 Upg: Regenerator m2
 RES_LC_SDIDEF SDI Defense
 Neutral researches (for all races)
 RES_MCH2 Upg: 20 mm Bullets
 RES_MCH3 Upg: 20 mm Bullets
 RES_MCH4 Upg: 20 mm Bullets
 RES_MSR2 Upg 1: Rocket (guided: 25%)
 RES_MSR3 Upg 2: Rocket (guided: 50%)
 RES_MSR4 Upg 3: Rocket (guided: 100%)
 RES_MMR2 Upg: Heavy Rocket (guided: 25%)
 RES_MMR3 Upg: Heavy Rocket (guided: 50%)
 RES_MMR4 Upg: Heavy Rocket (guided: 100%)

Artifact identifiers

Artifact identifiers can be used in function mission.CreateArtifact to create normal or special (handled by event mission.Artifact) artifacts.

Identifier Name
Normal artifacts
NEAMAMMO Ammunition
NEAENERGY Energy
NEAGIVESHIELD Shield generator
NEAMAXHP Full repair
NEGIVEMONEY Money 10 000 CR
NEASHOWMAP0 Show surface map
NEASHOWMAP1 Show tunnel map
Special artifacts (script-handled)
NEASPECIAL1 Artifact
NEASPECIAL2 Artifact
NEASPECIAL3 Artifact
NEASPECIAL4 Artifact
NEASPECIAL5 Artifact

EarthC tool

Using EarthC

When you will write your script you must to compile them to binary form. To do that use EarthC.exe tool. Call it as follows:

EarthC.exe sourceScript.eco sourceScript.eco

As you can see binary EarthC file must have extension ".eco". You must copy this script to directory depends on script type. Look at Scripts directory structure to check it. Also note that game type scripts for multiplayer must be placed in "Scripts\Gametypes" directory and for skirmish in "Scripts\Gametypes\single" directory.

WDCreator tool

Using WDCompiler

WDCompiler.exe is a tool which allows you to put many files into single file with '.wd' extension.

First, it's important to understand how game file system is working. Data files can be stored in '*.wd' files (as standard files are) or can be stored in subdirectories in Earth 2150 as separate files (like levels created in editor which are stored in 'Levels' directory). WD files contains directory structure - files of different types are in different directories. Name of *.wd file isn't important although usually standard *.wd files contains files from directory with this name (f.e. Scripts.wd contains files from Scripts directory). If file with the same name exist in the same directory in some *.wd file and under Earth 2150 directory then game uses newer file.

WDCompiler.exe can be used in two ways. If you want just add files from one directory to one *.wd file you can call WDCompiler.exe with specific parameters. For more complicated operations you can call WDCompiler.exe with specific file name as parameter which contains commands for execute. You can add files to *.wd file with or without compression. Script and level files are compressed itself so there should be add without compression.

As you can see in examples below file names contains slashes and backslashes. Backslashes are in the "external" part of path, and slashes are in "internal" part of path - which is part of game file system. You must use slashes properly otherwise game can't handle this files. If some path ends with f.e. 'Scripts/>.eco' than char '>' mean "all files in this directory and its subdirectories". Note that for scripts and levels you must use option 'c' or '<archive'.

Parameters for WDCompiler.exe:

WDCompiler.exe <WDFile> [<command> [command args]]

possible commands:

a - add files to WD without compression

p - add files to WD with compression

c - add files to WD (added files are already compressed)

examples:

```
WDCompiler.exe      MyLevels.wd      c      C:\Game\Earth\Levels/*.Ind  
C:\Game\Earth\Levels/*.mis  
WDCompiler.exe      MyScripts.wd     c      ./Scripts/Units/MyTank.eco  
./Scripts/Gametypes/MyMultiGame.eco  
WDCompiler.exe MyScripts.wd c ./Scripts/Gametypes/Single/*.eco
```

Using configuration files:

WDCompiler @scriptFile [define {define}]

where scriptFile is a file with following commands:

>wdfile - set name of wd and its guid

>pack - set compression for file (default)

>store - unset compression for file

>archive - information that file is already compressed (required for scripts and levels)

>binary - type of file - binary (default)

>text - type of file - text file

>add - add file to WD

examples:

WDCompiler.exe @WDCompiler.dat SCRIPTS

WDCompiler.exe @WDCompiler.dat ALL

with following WDCompiler.dat file:

```
#ifdef ALL  
#define LEVELS  
#define SCRIPTS  
#define LANGUAGE  
#endif  
#ifdef LANGUAGE  
<wdFile Language.wd  
<add <text "c:\Games\Earth\Language/*.txt"  
<add <archive "c:\Games\Earth\Language/*.lan"  
#endif  
#ifdef LEVELS  
<wdFile Levels.wd  
<add <archive "c:\Games\Earth\Levels/*.Ind"  
<add <archive "c:\Games\Earth\Levels/*.mis"  
#endif  
#ifdef SCRIPTS  
<wdFile Scripts.wd  
<add <archive "c:\Games\Earth\Scripts/>.eco"  
#endif
```

LangC tool

Using LangC tool

As you can see in examples included in this documentation scripts doesn't contains plain text but texts beginning with "translateSomething". This strings are "keys" for language-dependent strings which are stored in *.lan files in 'Language' directory ('language.wd' file). Game takes this key finds value for it and uses translated string. If you want to distribute your scripts it's good idea to use the same way of using strings. Also you must use it to display f.e. formatted briefings in campaigns. LangC.exe is the tool used to create *.lan files. LangC requires three types of files: *.idx files with only "translateSomething" list, *.txt files with "translateSomething" keys and its values and *.ini file - configuration file for LangC. *.lan file created by LangC should be copied to Language directory, and can be places in the same *.wd file with scripts (or in separated *.wd file if will be translated later to other language). Don't create language.lan file because it's standard Earth file.

Example of using LangC:

```
LangC.exe C:\Games\EarthLang\MyLang.ini
copy C:\Games\EarthLang\MyLang.lan C:\Games\Earth\Language
MyLang.ini contents:
[LangC] - must be
Output="MyLang" - name of *.lan' file
Index="C:\Games\EarthLang\" - directory which contains *.idx' files for 'Input'
files
Input0="MyLang1" - *.txt' file name
Input1="MyLang2"
MyLang1.idx contents:
translateMyCampaign
translateParams
MyLang2.idx contents:
translateBriefing1
translateGoal1
MyLang1.txt contents:
item translateMyCampaign "My campaign title"
//text below is example of text for functions with multi-parameters like AddBriefing
or AddGoal
//you must pass the same count of parameters (2 here) (strings or numbers) to
this function
item translateParams "First param is <%0>, and second is <%1>"
MyLang2.txt contents:
//you can make comments like this one
//example of preformatted text below:
message formated translateBriefing1
{
    This is multi-line
        preformatted text
    (without any quotas)
}
//multi-spaces below spilted to one space
item translateGoal1 "Earn      <%0>      CR"
```

Tips&Tricks

Tips&Tricks&various notes

If you want to create your own scripts you should read this notes to avoid some surprises, artifacts or game crashes.

EarthC and other tools in this package were designed for developer use only. It means that these tools are not very error-resistant. For example you can write script which is compiled without errors but game will crash when this script is used. Before writing your own scripts you should study included script sources, and if some bugs in your scripts occurs try to compare it with our scripts - some functions must be used in some specific order or with other functions.

Scripts compiled with included EarthC.exe tool are working only with Earth 2150 version 2.8.7 and can crash while used in older game versions.

Test your scripts well before you publish them. We are not responsible for psychical/physical damages made by your buggy scripts :).

Don't be surprised if you will see some strange comments in scripts. It's Polish. (And BTW sorry for poor and buggy English in this documentation)

Don't copy this documentation to Earth 2150 game directory. At startup Earth scans this directory and read headers of all files, so doing this operation with large amount of files takes a lot of time.

Use WDCreator tool - its comfortable because it creates one file instead of many files in many directories.

In multiplayer game all missing files like non-standard levels, user banners or scripts (gametypes, AI players, unit scripts) are downloaded before game starts (in session configuration dialog). If such a file is placed in "*.wd" file then whole "*.wd" file is downloaded. So don't make too big "*.wd" files (f.e. don't put 100 multiplayer levels into single file) because it take too much time to download it.

You don't need to copy "*.wd" files to "WDFiles" directory. It's good idea to copy this files to f.e. "CustomWDFiles" directory.

If you will modify some standard Earth files (like levels or scripts) save it under other name (otherwise it will cover standard file from "*.wd" file). Many EarthC functions uses values called "tick". Tick is one game simulation step. For default game speed game makes 20 ticks per second (5 for minimum, 50 for maximum game speed). But if game have a lot to calculate and a lot to display (f.e. battle displayed in maximum zoom out) than (especially at maximum game speed) game can slow down and make less ticks per second than it should do.

File names for mission levels should start with char '!' (f.e. !234.ind, !234.mis). Such a files are not displayed in skirmish/multiplayer configuration dialog. To load such a file in editor (which normally don't display files beginning with '!') use console command 'editor.singleplayer 1' (in editor press Enter, type: editor.singleplayer 1 and press Enter again). If you are bored doing it all the time you can put this line in file 'autoexec.com' in main Earth 2150 directory. Using 'editor.singleplayer' command you can open all levels from standard Earth campaigns.

Use Lang and "translate" strings in scripts instead of putting plain text. It looks much better, don't cost a lot of time and your scripts can be easily translated to other language.

Scripts of all types are stored in saves, and when save is loaded game uses scripts stored in save file. So if you want for example to test changed mission script you must start this mission again (choose it in choose mission dialog). If you have changed campaign script you must start the whole campaign again.

You can debug your scripts. To do that use script function TraceWith parameter string or number. To see this traces you must turn on display with console command: 'display.show 1'.

Game type scripts (for skirmish and multiplayer), are stored in two directories. Scripts for multiplayer games are stored in 'Scripts\Gametypes' directory and scripts for skirmish are stored in 'Scripts\Gametypes\single' directory.

In unit and mission scripts delete any reference members (of type unit, tank, etc) in command Uninitialize. It's very important, otherwise game can crash after exiting campaign mission.)

Your campaign script name must begin with CampaignED (CampaignnUCS, CampaignLC or TutorialED, TutorialUCS, TutorialLC for tutorial campaigns) f.e.: CampaignEDSomeExtension.eco (or TutorialEDSomeExt.eco for tutorial) and must be placed in directory Scripts\Campaigns\ED (or UCS, LC for other races). If your campaign name (in first line of script) have name f.e. "translateCampaignEDMySuperCamp" then you can add string with "Description" suffix ("translateCampaignEDMySuperCampDescription") - with short description of your campaign. This string will be displayed in the text box in 'Choose campaign' dialog.

If you have added some custom campaign (or tutorial) you should see 'Choose campaign' dialog after pressing 'Start new game' or 'Tutorial' button. If this dialog is not displayed (and game start immediately) it means that your campaign script have bad file name, or is placed in bad directory.

If you are testing your campaign you can use console command 'ai.allmissions 1' - then you can select any mission in 'Choose mission' dialog.

In any campaign mission you can use console command 'world.endmission 1' - then button 'End mission' appears immediately.

You can switch to any AI player in game to see if it works properly and as you expect it to work. To do this type console command 'ai.local n' where n is an 'IFF number' (slot in which player is placed - numbered from 0 to 15). In this case you must first enable cheats with command 'I_wanna_cheat'. Note that you will play as AI player but AI in this player is still functioning and doing its job.

Note that you can't add any custom levels or scripts to Demo version of Earth 2150.



The Moon Project

Notice: All Earth 2150 orders can also be used in the Moon Projekt.
In this section you will only find specific Moon Projekt orders!

MoonC documentation

Version 2.0 BETA for "The Moon Project" version 1.0
Copyright © 1999-2002 ZUXXEZ ENTERTAINMENT AG

This documentation and all tools distributed with it, are used on your own risk and we offer NO support of ANY kind. We do not guarantee that all functions/methods/structures etc. will behave exactly in the way it is described in this documentation. Badly designed custom scripts may cause game crashes or make you unable to play over net.

!!Welcome to the MoonC documentation.!!

EarthC is a developer tool for The Moon Project which allows you to create your own campaigns, tutorials, gametypes, AI players and unit scripts. This documentation contains sources of all scripts used in The Moon Project. You can use it to learn how EarthC works before creating your own scripts, then you can use it as the base for your scripts. You can also create your own versions of the three standard The Moon Project campaigns by changing missions, adding new ones or removing boring ones (we hope there aren't any :-)).

EarthC is a programming language so you need basic programming skills to write your own scripts but included sources should help you a lot.

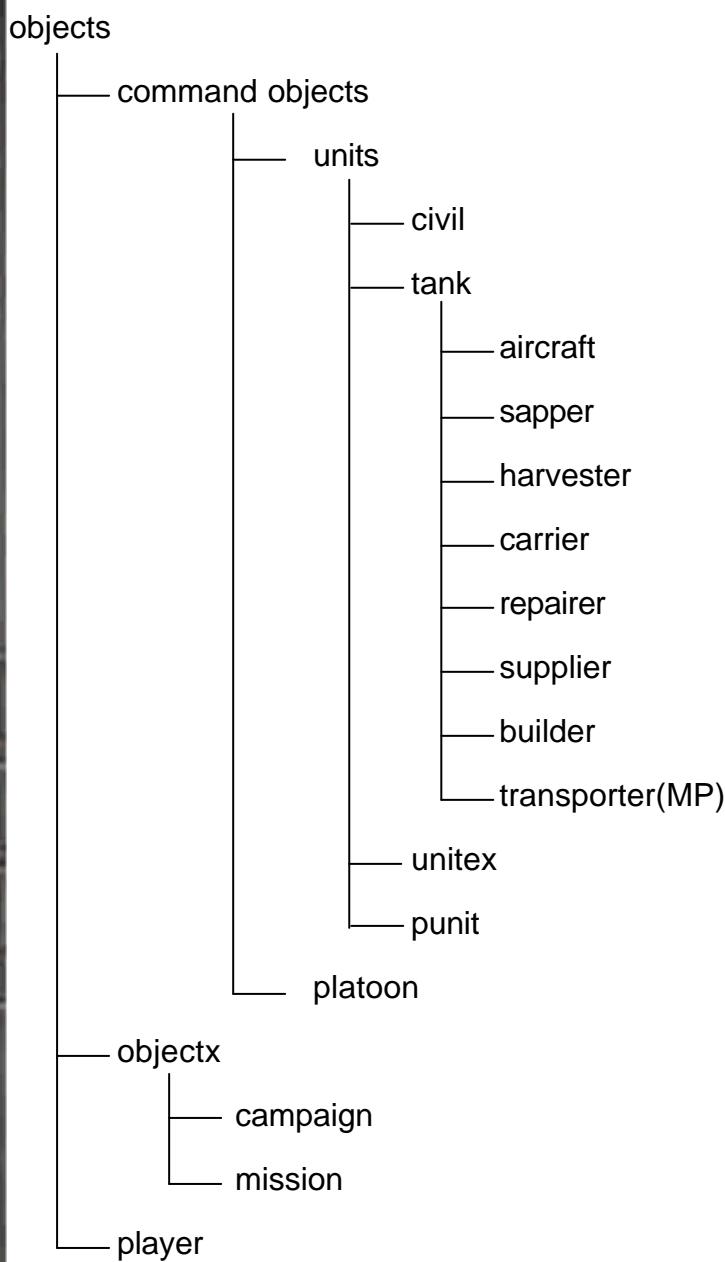
We have included also two other usefull tools: WDCreator for putting files into a single '*.wd' file and LangC which helps you organize and localize strings in scripts. All tools are located in the Tools subdirectory.

Note that new functions and objects from "The Moon Project" included in this documentation are signed with symbol [MP]. Compiled script for "The Moon Project" must have extension 'ecoMP' not 'eco'.

MoonC language description

Same as in Earth!

Objects hierarchy



object transporter

Description

Scripts of type transporter are attached to Transporter units - used to transport other objects.

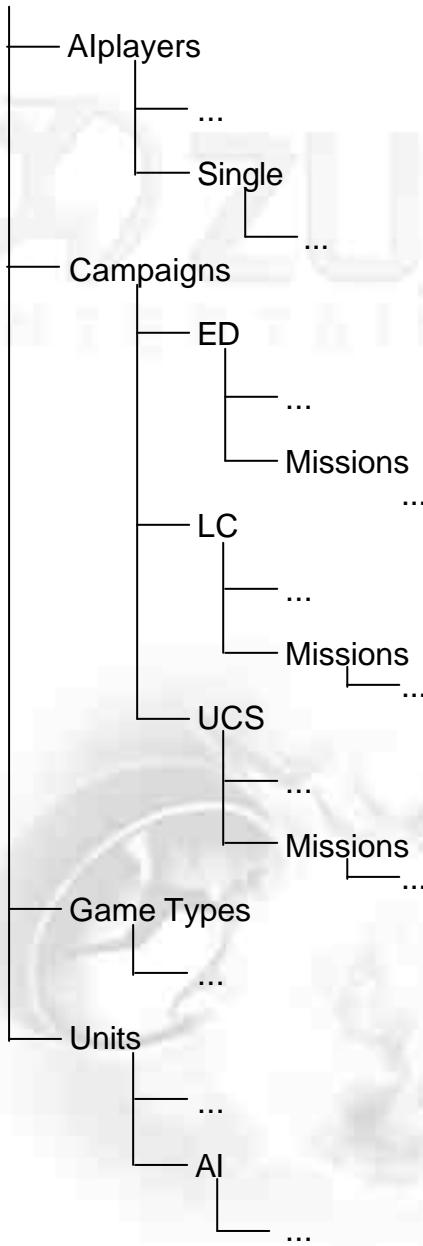
Functions

```
void CallGetUnit(int nX, int nY, int nZ)
void CallPutUnit(int nX, int nY, int nZ)
void CallDropUnit(int nX, int nY, int nZ)
int IsGettingUnit()
int IsPuttingUnit()
int IsDroppingUnit()
int HaveUnitOnHook()
int CanPutUnitFromHookInPoint(int nX, int nY, int nZ)
int IsUnitToGetInPoint(int nX, int nY, int nZ)
int FindFreePlaceToPutUnitFromHook(int nX, int nY, int nZ, int nSearchRange)
int GetFoundFreePlaceToPutUnitX()
int GetFoundFreePlaceToPutUnitY()
int GetFoundFreePlaceToPutUnitZ()
void SetAutoGetPoint(int nX, int nY, int nZ)
void InvalidateAutoGetPoint()
void SetAutoPutPoint(int nX, int nY, int nZ)
void InvalidateAutoPutPoint()
```

Scripts directory structure

Note that some scripts for The Moon Project are located in f.e. 'MP-ED' or 'MP' directory, but compiled scripts must be copied to the same directories as for other scripts (f.e. to 'Campaigns\ED' directory).

Scripts



Single default

```
player "translateAlPlayerMedium"
{
    state Initialize;
    state Nothing;

    state Initialize
    {
        if(GetRace() == 1)//ucs
        {
            SetName("translateAlPlayerNameUCSMedium");
            if(GetFFNumber() == 1 || GetFFNumber() == 6)SetName("translateAlNameMediumUCS1");
            if(GetFFNumber() == 2 || GetFFNumber() == 7)SetName("translateAlNameMediumUCS2");
            if(GetFFNumber() == 3 || GetFFNumber() == 8)SetName("translateAlNameMediumUCS3");
            if(GetFFNumber() == 4 || GetFFNumber() == 9)SetName("translateAlNameMediumUCS4");
        }
        if(GetRace() == 2)//ed
        {
            SetName("translateAlPlayerNameEDMedium");
            if(GetFFNumber() == 1 || GetFFNumber() == 6)SetName("translateAlNameMediumED1");
            if(GetFFNumber() == 2 || GetFFNumber() == 7)SetName("translateAlNameMediumED2");
            if(GetFFNumber() == 3 || GetFFNumber() == 8)SetName("translateAlNameMediumED3");
            if(GetFFNumber() == 4 || GetFFNumber() == 9)SetName("translateAlNameMediumED4");
        }
        if(GetRace() == 3)//lc
        {
            SetName("translateAlPlayerNameLCMedium");
            if(GetFFNumber() == 1 || GetFFNumber() == 6)SetName("translateAlNameMediumLC1");
            if(GetFFNumber() == 2 || GetFFNumber() == 7)SetName("translateAlNameMediumLC2");
            if(GetFFNumber() == 3 || GetFFNumber() == 8)SetName("translateAlNameMediumLC3");
            if(GetFFNumber() == 4 || GetFFNumber() == 9)SetName("translateAlNameMediumLC4");
        }
    }
    return Nothing;
}
state Nothing
{
    return Nothing,10000;
}
```

CampaignEDMP01.ec

```
campaign "translateCampaignEDMP01"
```

```
{
    consts
    {
        raceUCS=1;
        raceED=2;
        raceLC=3;

        mis511 = 50;
        mis512 = 51;
        mis513 = 52;
        mis514 = 53;
        mis515 = 54;
        mis516 = 55;
        mis517 = 56;
        mis518 = 57;
        mis519 = 58;
        mis520 = 59;
        mis521 = 60;//add by JS to compile
        mis522 = 61;//add by JS to compile

        base1 = 65;
        base2 = 66;
        base3 = 67;

        timeFlagWinter = 1;
        timeFlagEarlySpring = 2;
        timeFlagSpring = 4;
        timeFlagSummer = 8;
        timeFlagDesert = 16;
        timeFlagVolcano = 32;
        timeFlagHeavyVolcano = 64;
        timeFlagMoon = 128;
        baseFlag = 255 ;
    }

    int n_TimeFlag;

    state Initialize;
    state Start;
    state Nothing;
    state LoadMission522;
```

```

state Initialize
{
    int i;

    RegisterMission(base1,"IMPEDBase1","ED\\Missions\\EDbase1script","");
    baseFlag, 0, 90,10,10,10);
    RegisterMission(base2,"IMPEDbase2","ED\\Missions\\EDbase2script","");
    baseFlag, 0,-90,10,10,10);
    RegisterMission(base3,"IMPEDbase3","ED\\Missions\\EDbase3script","");
    baseFlag, 0,-90,10,10,10);

    //          n      Ind     script      preBriefing
timeFlags x y d1 d2 d3 next1 next2 next3 next4
    RegisterMission(mis511, "IS11", "ED\\Missions\\Mission511",
"translateBriefingShort511", timeFlagWinter, 90, 65, 10, 10, 10);
    RegisterMission(mis512, "IS12", "ED\\Missions\\Mission512",
"translateBriefingShort512", timeFlagWinter, 60, 75, 10, 10, 10,
mis514,mis513);
    RegisterMission(mis513, "IS13", "ED\\Missions\\Mission513",
"translateBriefingShort513", timeFlagEarlySpring, 105, 45, 10, 10, 10);
    RegisterMission(mis514, "IS14", "ED\\Missions\\Mission514",
"translateBriefingShort514", timeFlagEarlySpring, 65, 40, 10, 10, 10,
mis516,mis515);
    RegisterMission(mis515, "IS15", "ED\\Missions\\Mission515",
"translateBriefingShort515", timeFlagSpring, 70, 30, 10, 10, 10);
    RegisterMission(mis516, "IS16", "ED\\Missions\\Mission516",
"translateBriefingShort516", timeFlagSpring, 30, 50, 10, 10, 10, mis517);
    RegisterMission(mis517, "IS17", "ED\\Missions\\Mission517",
"translateBriefingShort517", timeFlagSummer, -3, 40, 10, 10, 10, mis519);
    //RegisterMission(mis518, "IS18", "ED\\Missions\\Mission518",
"translateBriefingShort518", timeFlagSummer, -50, 10, 10, 10, 10);
    RegisterMission(mis519, "IS19", "ED\\Missions\\Mission519",
"translateBriefingShort519", timeFlagDesert, 30, 0, 10, 10, 10, mis521);
    //RegisterMission(mis520, "IS20", "ED\\Missions\\Mission520",
"translateBriefingShort520", timeFlagDesert, 0, 90, 10, 10, 10);
    RegisterMission(mis521, "IS21", "ED\\Missions\\Mission521",
"translateBriefingShort521", timeFlagVolcano, -82, 28, 10, 10, 10,mis522);
    RegisterMission(mis522, "IS22", "ED\\Missions\\Mission522",
"translateBriefingShort522", timeFlagMoon, 0, 90, 10, 10, 10);

    CreateGamePlayer(1, raceUCS, playerAI, null);
    CreateGamePlayer(2, raceED, playerLocal, null);
    CreateGamePlayer(3, raceLC, playerAI, null);

    LoadBase(1,base1,2);
    LoadBase(2,base2,1);
    LoadBase(3,base3,3);

    SetActivePlayerAndWorld(2,1); //player, world
    SetAvailableWorlds(1 | 2 | 4 | 8); //misja 1 2 bazy ED

    EnableMission(mis511,true);
    EnableMission(mis512,true);

    SetSeason(1);
    n_TimeFlag = timeFlagWinter;
    ActivateMissions(timeFlagWinter,true);

    return Start,221;//10sek
}
//-----
state Start
{
    EnableChooseMissionButton(true);
    return Nothing,50;
}
//-----
state Nothing
{
    return Nothing,0; //ma byc 0 zeby szybko wskoczylo do LoadMission522
}
//-----
state LoadMission522
{
    TraceD("Load 522 \n");
    SetSeason(8);
    LoadMission(0.mis522);
    SetAvailableWorlds(1)//only mission
    SendCustomEvent(0,0,0,128);
    return Nothing;
}
//-----
event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    LoadMission(0,iMission);
}

EnableMission(iMission,false);
}

//-----
event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
    if(bEnable==2)//game loose -show video SUICIDE
    {
        SetAvailableWorlds(1 | 2 | 4 | 8);
        EndGame("Video\\Cutscene16.wd1");
        return;
    }
    if(bEnable==3)//game win -show video
    {
        SetAvailableWorlds(1 | 2 | 4 | 8);
        EndGame("Video\\Cutscene17.wd1");
        return;
    }
    EnableMission(GetNextMission(iMission,iNextNr),bEnable);
}

//-----
event EndingMission(int iMission, int nResult)
{
    if(iMission==mis521)
    {
        //tak dla pewnosci
        SetAvailableWorlds(1 | 2 | 4 | 8);
    }
}
//-----
event EndMission(int iMission, int nResult) //
{
    Traced("End Mission: ");TraceD(iMission);TraceD(" \n");
    if(iMission==mis521)
    {
        TraceD("Ready to load 522 \n");
        ForceKeepBitmapAfterStatistic(); //musi byc tutaj a nie przed
LoadMission
    state LoadMission522;
    return;
    }

    if(nResult==true)
    {
        SetMissionState(iMission,stateAccomplished);
    }
    else
    {
        SetMissionState(iMission,stateFailed);
    }

    if(!AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
    {
        if(n_TimeFlag != timeFlagVolcano)
        {
            n_TimeFlag = n_TimeFlag -2;
        }
        else
            n_TimeFlag = timeFlagMoon;
    }

    if(AreMissionsEnabledEq(n_TimeFlag,n_TimeFlag))
    {
        ActivateMissionsEq(n_TimeFlag,n_TimeFlag,true);
    }
    else
    {
        SendCustomEvent(0,0,0,0);
        return;
    }

    SetSeason(1);
    if(n_TimeFlag==timeFlagEarlySpring)SetSeason(2);
    if(n_TimeFlag==timeFlagSpring) SetSeason(3);
    if(n_TimeFlag==timeFlagSummer) SetSeason(4);
    if(n_TimeFlag==timeFlagDesert) SetSeason(5);
    if(n_TimeFlag==timeFlagVolcano) SetSeason(6);
    if(n_TimeFlag==timeFlagHeavyVolcano) SetSeason(7);
    if(n_TimeFlag==timeFlagMoon) SetSeason(8);

    EnableChooseMissionButton(true);
}
}
*****
```

TutorialED.ec

```
campaign "translateCampaignTutorialED0"
{
    state Initialize;
    state Nothing;

    state Initialize
    {
        CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");
        CreateGamePlayer(2,raceED,playerLocal,"TestGameAI");

        SetTime(100);

        RegisterMission(0,"ITutorialED","ED\\Missions\\TutorialEDmis","","0,0,0,0,0);
        LoadMission(0,0);
        SetAvailableWorlds(1);
        SetActivePlayerAndWorld(2,0);
        return Nothing;
    }

    state Nothing
    {
        return Nothing,200;
    }
}
```

TutorialED1.ec

```
campaign "translateCampaignTutorialED1"
{
    state Initialize;
    state Nothing;

    state Initialize
    {
        CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");
        CreateGamePlayer(2,raceED,playerLocal,"TestGameAI");

        SetTime(100);

        RegisterMission(0,"ITutorialED1","ED\\Missions\\TutorialED1mis","","0,0,0,0,0);
        LoadMission(0,0);
        SetAvailableWorlds(1);
        SetActivePlayerAndWorld(2,0);
        return Nothing;
    }

    state Nothing
    {
        return Nothing,200;
    }
}
```

TutorialED2.ec

```
campaign "translateCampaignTutorialED2"
{
    state Initialize;
    state Nothing;

    state Initialize
    {
        CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");
        CreateGamePlayer(2,raceED,playerLocal,"TestGameAI");
        CreateGamePlayer(3,raceUCS,playerAI,"TestGameAI");
        CreateGamePlayer(4,raceUCS,playerAI,"TestGameAI");

        SetTime(100);

        RegisterMission(0,"ITutorialED2","ED\\Missions\\TutorialED2mis","","0,0,0,0,0);
        LoadMission(0,0);
        SetAvailableWorlds(1);
        SetActivePlayerAndWorld(2,0);
        return Nothing;
    }

    state Nothing
    {
        return Nothing,200;
    }
}
```

TutorialED3.ec

```
campaign "translateCampaignTutorialED3"
{
    consts
    {
        raceUCS=1;
        raceED=2;
    }
```

```
        raceLC=3;
        mis1 = 0;
        base1 = 1;
        base2 = 2;
        base3 = 3;
        timeFlagWinter = 1;
        timeFlagEarlySpring = 2;
        timeFlagSpring = 4;
        timeFlagSummer = 8;
        timeFlagDesert = 16;
        timeFlagVolcano = 32;
        timeFlagHeavyVolcano = 64;
        timeFlagMoon = 128;
        baseFlag = 255;
    }

    int n_TimeFlag;
    state Initialize;
    state Start;
    state Nothing;
    state Initialize
    {
        int i;
        RegisterMission(base1,"ITutorialEDBase1","ED\\Missions\\TutorialED3base1","");
        baseFlag, 0, 90,10,10,10);
        RegisterMission(base2,"ITutorialEDBase2","ED\\Missions\\TutorialED3base2","");
        baseFlag, -0,90,10,10,10);
        RegisterMission(base3,"ITutorialEDBase2","ED\\Missions\\TutorialED3base2","");
        baseFlag, -0,90,10,10,10);
        //-----timeFlags x y nr lnd script preBriefing
        RegisterMission(mis1,"TutorialED3","ED\\Missions\\TutorialED3mis",
        "translateBriefingShortTutorialED3",timeFlagWinter, 90, 65, 1, 1, 1);
        CreateGamePlayer(1,raceUCS, playerAI, null);
        CreateGamePlayer(2, raceED, playerLocal, null);
        CreateGamePlayer(3, raceLC, playerAI, null);

        LoadBase(1,base1,2);
        LoadBase(2,base2,1);
        LoadBase(3,base3,3);

        SetActivePlayerAndWorld(2,1); //player, world
        SetAvailableWorlds(1 | 2); //misja i baza ED
        EnableMission(mis1,true);
        SetSeason(3);
        n_TimeFlag = timeFlagWinter;
        ActivateMissions(timeFlagWinter,true);
        EnableChooseMissionButton(false);
        return Start,100//10sek
    }
    //-----
    state Start
    {
        return Nothing,50;
    }
    //-----
    state Nothing
    {
        return Nothing;
    }
    //-----
    event StartMission(int iMission)
    {
        EnableChooseMissionButton(false);
        LoadMission(iMission);
        EnableMission(iMission,false);
    }
    //-----
    event EnableNextMission(int iMission,int iNextNr,int bEnable)
    {
        EnableChooseMissionButton(true);
    }
}
```

```

//-----
event EndMission(int iMission, int nResult) //-----  

{  

    //-----  

    //----- Missions -----  

    //-----  

    mission "translateEDbase1"  

    {  

        consts  

        {  

            accountMainBase = 1;  

            accountResearchBase = 2;  

            accountCareerPoints = 3;  

            careerStep1 = 20;  

            careerStep2 = 40;  

            careerStep3 = 60;  

            careerStep4 = 80;  

            careerStep5 = 100;  

            careerStep6 = 120;  

            goalCareerStep1=0;  

            goalCareerStep2=1;  

            goalCareerStep3=2;  

            goalCareerStep4=3;  

            goalCareerStep5=4;  

            goalCareerStep6=5;  

        }  

        player p_Player;  

        player p_EnemyUCS;  

        player p_EnemyLC;  

        int i;  

        int n_CareerPoints;  

        int n_CareerStep;  

        state Initialize;  

        state PlayTrackState;  

        state CameraMove;  

        state Nothing;  

        state ShowBriefing;  

        state Initialize  

        {  

            unix uHero;  

            //----- Goals -----  

            RegisterGoal(goalCareerStep1,"translateGoalCareerStep1");  

            RegisterGoal(goalCareerStep2,"translateGoalCareerStep2");  

            RegisterGoal(goalCareerStep3,"translateGoalCareerStep3");  

            RegisterGoal(goalCareerStep4,"translateGoalCareerStep4");  

            RegisterGoal(goalCareerStep5,"translateGoalCareerStep5");  

            RegisterGoal(goalCareerStep6,"translateGoalCareerStep6");  

            EnableGoal(goalCareerStep1,true);  

            EnableGoal(goalCareerStep2,true);  

            EnableGoal(goalCareerStep3,true);  

            EnableGoal(goalCareerStep4,true);  

            EnableGoal(goalCareerStep5,true);  

            EnableGoal(goalCareerStep6,true);  

            //----- Players -----  

            p_Player = GetPlayer(2);  

            p_EnemyUCS = GetPlayer(1);  

            p_EnemyLC = GetPlayer(3);  

            //----- AI -----  

            p_EnemyUCS.EnableAIFeatures(aiEnabled,false);  

            p_EnemyLC.EnableAIFeatures(aiEnabled,false);  

            //----- Money -----  

            p_Player.SetMoney(0);  

            p_EnemyUCS.SetMoney(0);  

            p_EnemyLC.SetMoney(0);  

            //----- Buildings -----  

            p_Player.EnableBuilding("EDBPP",true);  

            p_Player.EnableBuilding("EDBBA",true);  

            p_Player.EnableBuilding("EDBFA",true);  

            p_Player.EnableBuilding("EDWBW",false);  

            p_Player.EnableBuilding("EDBAB",true);  

            // 2nd tab  

            p_Player.EnableBuilding("EDBRE",false);  

            p_Player.EnableBuilding("EDBMBI",false);  

            p_Player.EnableBuilding("EDBTC",false);  

            // 3rd tab  

            p_Player.EnableBuilding("EDBST",true);  

            p_Player.EnableBuilding("EDBHT",true);  

            p_Player.EnableBuilding("EDBHT",true);  

            // 4th tab  

            p_Player.EnableBuilding("EDBUC",false);  

            p_Player.EnableBuilding("EDBRC",false);  

            p_Player.EnableBuilding("EDBHQ",false);  

            p_Player.EnableBuilding("EDBRA",true);  

            p_Player.EnableBuilding("EDBN1",true);  

            p_Player.EnableBuilding("EDBLZ",true);  

            p_Player.EnableBuilding("EDBART",false);  

            p_Player.EnableCommand(commandSoldBuilding,true);  

            //----- Research -----  

            p_Player.AddResearch("RES_MISSION_PACK1_ONLY");  

            p_Player.AddResearch("RES_MSR2");  

            p_Player.AddResearch("RES_MSR3");  

            p_Player.AddResearch("RES_MSR4");  

            p_EnemyUCS.AddResearch("RES_MISSION_PACK1_ONLY");  

            p_EnemyUCS.AddResearch("RES_MSR2");  

            p_EnemyUCS.AddResearch("RES_MSR3");  

            p_EnemyUCS.AddResearch("RES_MSR4");  

            p_EnemyLC.AddResearch("RES_MISSION_PACK1_ONLY");  

            p_EnemyLC.AddResearch("RES_MSR2");  

            p_EnemyLC.AddResearch("RES_MSR3");  

            p_EnemyLC.AddResearch("RES_MSR4");  

            //511  

            p_Player.EnableResearch("RES_ED_UML3",false);  

            p_Player.EnableResearch("RES_ED_UA22",false);  

            p_Player.EnableResearch("RES_ED_MGen",false); //shield gen 2  

            p_Player.EnableResearch("RES_ED_WSR2",false);  

            p_Player.EnableResearch("RES_ED_WAN1",false);  

            p_Player.EnableResearch("RES_ED_WSL2",false);  

            p_Player.EnableResearch("RES_ED_WSL2",false);  

            p_Player.EnableResearch("RES_ED_MSC2",false);  

            p_Player.EnableResearch("RES_ED_MCH2",false);  

            //512  

            p_Player.EnableResearch("RES_ED_UMW2",false);  

            p_Player.EnableResearch("RES_EDUUT",false);  

            p_Player.EnableResearch("RES_ED_ACH2",false);  

            p_Player.EnableResearch("RES_ED_WSL3",false);  

            p_Player.EnableResearch("RES_ED_MSC3",false);  

            p_Player.EnableResearch("RES_MCH3",false);  

            p_Player.EnableResearch("RES_ED_SCR",false);  

            //513  

            p_Player.EnableResearch("RES_ED_HGen",false); //shield gen 3  

            p_Player.EnableResearch("RES_ED_UM11",false);  

            p_Player.EnableResearch("RES_ED_UWW3",false);  

            p_Player.EnableResearch("RES_ED_WSR3",false); //SR3  

            p_Player.EnableResearch("RES_ED_ASR1",false); //helicopter rocket launcher  

            p_Player.EnableResearch("RES_ED_BMD",false); //medium defense building  

            p_Player.EnableResearch("RES_ED_SCR2",false);  

            p_Player.EnableResearch("RES_ED_ART",false); //zby nie mozna go bylo wynalezc bo dostajemy go w misji 513  

            //514  

            p_Player.EnableResearch("RES_ED_ASR2",false); //helicopter rocket launcher  

            p_Player.EnableResearch("RES_ED_AMR1",false); //helicopter rocket launcher  

            p_Player.EnableResearch("RES_ED_SCR3",false);  

            p_Player.EnableResearch("RES_ED_UA41",false);  

            p_Player.EnableResearch("RES_ED_USTEALTH",false);  

            //515  

            p_Player.EnableResearch("RES_ED_BC1",false);  

            p_Player.EnableResearch("RES_ED_UA31",false);  

            p_Player.EnableResearch("RES_ED_AMR2",false); //helicopter rocket launcher  

            p_Player.EnableResearch("RES_ED_UHW1",false);  

            p_Player.EnableResearch("RES_ED_WH1",false);  

            p_Player.EnableResearch("RES_ED_WHL1",false);  

            p_Player.EnableResearch("RES_ED_WMR1",false);

```

```

p_Player.EnableResearch("RES_ED_WH11",false);
p_Player.EnableResearch("RES_MMR2",false);

//516
p_Player.EnableResearch("RES_ED_UHT1",false);
p_Player.EnableResearch("RES_ED_MHC2",false);
p_Player.EnableResearch("RES_ED_WH2",false);
p_Player.EnableResearch("RES_ED_WHL2",false);
p_Player.EnableResearch("RES_ED_WMR1",false);
p_Player.EnableResearch("RES_ED_WH2",false);
p_Player.EnableResearch("RES_EDWEQ1",false);
p_Player.EnableResearch("RES_EDBHQ1",false);

//517
p_Player.EnableResearch("RES_ED_UHT2",false);
p_Player.EnableResearch("RES_ED_MHC3",false);
p_Player.EnableResearch("RES_ED_BHD",false);//medium defense building

//519
p_Player.EnableResearch("RES_ED_WBT1",false);

//521
p_Player.EnableResearch("RES_ED_WBT2",false);

//never used
p_Player.EnableResearch("RES_ED_AB1",false);
p_Player.EnableResearch("RES_ED_WHR1",false);

p_Player.EnableResearch("RES_ED_USS1",false);
p_Player.EnableResearch("RES_ED_USS2",false);
p_Player.EnableResearch("RES_ED_USM1",false);
p_Player.EnableResearch("RES_ED_USM2",false);
p_Player.EnableResearch("RES_ED_WH11",false);

p_Player.EnableResearch("RES_ED_WHC1",false);
p_Player.EnableResearch("RES_ED_WMR1",false);
p_Player.EnableResearch("RES_ED_WHL1",false);

//--- Timers -----
SetTimer(0,10);
//----- Camera -----
CallCamera();
EnableInterface(false);
EnableCameraMovement(false);
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY());
25,128,30,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY())
,10,128,40,0,80,1);
return PlayTrackState,3;
}

state PlayTrackState
{
TraceD("!!!!!!PlayTrack: music\edday_3.mp2\n");
PlayTrack("music\edday_3.mp2");
return CameraMove,77;
}
//-----
state CameraMove
{
}

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY())
,5,60,0,80,1);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
EnableInterface(true);
EnableCameraMovement(true);
AddBriefing("translateStartCampaignEDMP01",p_Player.GetName());
}

Show(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),45,400,5000,800
,10);
return Nothing,50;
}

//-----
state Nothing
{
if(p_Player.GetScriptData(7))//Attack
{
CallCamera();
AddBriefing("translateBriefingBase1a");
}
}

p_Player.SetScriptData(7,0);
p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX(),+5,
0,null,"UCSUA1","UCSWAP1",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+1,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP1",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+2,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP1",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+3,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP1",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+4,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP1",null,null,null,0);
}
if(p_Player.GetScriptData(8))//Attack
{
CallCamera();
AddBriefing("translateBriefingBase1b");
p_Player.SetScriptData(8,0);
p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX(),+5,
0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+1,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+2,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+3,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+4,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);
}
if(p_Player.GetScriptData(9))//Attack
{
CallCamera();
AddBriefing("translateBriefingBase1c");
p_Player.SetScriptData(9,0);
p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX(),+5,
0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+1,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+2,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+3,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+4,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);
}
if(p_Player.GetScriptData(9))//Attack
{
CallCamera();
AddBriefing("translateBriefingBase1c");
p_Player.SetScriptData(9,0);
p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX(),+5,
0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+1,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+2,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+3,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX())+4,p_EnemyUCS.GetStartingPointY(),+5,0,null,"UCSUA1","UCSWAP2",null,null,null,0);
}
if(p_Player.GetScriptData(accountMainBase))//Chash for accomplished mission
{
p_Player.AddMoney(p_Player.GetScriptData(accountMainBase));
p_Player.SetScriptData(accountMainBase,0);
}
if(p_Player.GetScriptData(accountCareerPoints))
{
n_CareerPoints =
n_CareerPoints+p_Player.GetScriptData(accountCareerPoints);
p_Player.SetScriptData(accountCareerPoints,0);
}

if(n_CareerStep==0 &&
n_CareerPoints >= careerStep1)
{
SetGoalState(goalCareerStep1,goalAchieved);
ShowVideo("Cutscene10.wd1");
n_CareerStep=1;
}
if(n_CareerStep==1 && n_CareerPoints >= careerStep2)
{
SetGoalState(goalCareerStep2,goalAchieved);
ShowVideo("Cutscene11.wd1");
n_CareerStep=2;
}
}

```

```

if(n_CareerStep==2 && n_CareerPoints >= careerStep3)
{
    SetGoalState(goalCareerStep3, goalAchieved);
    ShowVideo("Cutscene12.wd1");
    n_CareerStep=3;
}
if(n_CareerStep==3 && n_CareerPoints >= careerStep4)
{
    SetGoalState(goalCareerStep4, goalAchieved);
    ShowVideo("Cutscene13.wd1");
    n_CareerStep=4;
}
if(n_CareerStep==4 && n_CareerPoints >= careerStep5)
{
    SetGoalState(goalCareerStep5, goalAchieved);
    ShowVideo("Cutscene14.wd1");
    n_CareerStep=5;
}
if(n_CareerStep==5 && n_CareerPoints >= careerStep6)
{
    SetGoalState(goalCareerStep6, goalAchieved);
    ShowVideo("Cutscene15.wd1");
    n_CareerStep=6;
}
if(n_CareerStep==0)SetConsoleText("translateMessageCareer",careerStep1
-n_CareerPoints);
if(n_CareerStep==1)SetConsoleText("translateMessageCareer",careerStep2
-n_CareerPoints);
if(n_CareerStep==2)SetConsoleText("translateMessageCareer",careerStep3
-n_CareerPoints);
if(n_CareerStep==3)SetConsoleText("translateMessageCareer",careerStep4
-n_CareerPoints);
if(n_CareerStep==4)SetConsoleText("translateMessageCareer",careerStep5
-n_CareerPoints);
if(n_CareerStep==5)SetConsoleText("translateMessageCareer",careerStep6
-n_CareerPoints);
if(n_CareerStep==6)SetConsoleText(" ");
return Nothing,50;
}
//-----
event Timer0()
{
}

//-----
event CustomEvent0(int k1,int k2,int k3,int k4)
{
    if(k4==128)
    {
        EnableUnitSounds(false);
        EnableBuildingSounds(false);
    }
}

----- Money -----
p_Player.SetMoney(0);
p_EnemyUCS.SetMoney(0);
p_EnemyLC.SetMoney(0);

----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,false);

p_Player.EnableBuilding("EDBPP",true);
p_Player.EnableBuilding("EDBBA",true); //JS
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDWBW",false);
p_Player.EnableBuilding("EDBAB",true); //JS
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBMF",false);
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
p_Player.EnableBuilding("EDBST",true);
p_Player.EnableBuilding("EDBBT",true);
p_Player.EnableBuilding("EDBHT",true);
p_Player.EnableBuilding("EDBART",false);
// 4th tab
p_Player.EnableBuilding("EDBCU",false);
p_Player.EnableBuilding("EDBRC",true);
p_Player.EnableBuilding("EDBHQ",false);
p_Player.EnableBuilding("EDBRA",true);
p_Player.EnableBuilding("EDBEN1",false);
p_Player.EnableBuilding("EDBZL",true);

----- Research -----
//--- Variables for campaign goals ---
/p_Player.SetScriptData(0,0);
/p_Player.SetScriptData(1,0);
/p_Player.SetScriptData(2,0);
/p_Player.SetScriptData(3,0);
/p_Player.SetScriptData(4,0);

----- Camera -----
p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0.4
5,0);
return Nothing,100;
}
----- state Nothing -----
{
    if(p_Player.GetScriptData(10))//Attack
    {
        AddBriefing("translateBriefingBase2a");
        p_Player.SetScriptData(10,0);
    }
}

----- Camera -----
p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStar
tingPointY())+5, 0,null,"UCSU13","UCSWAP1",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX()+1,p_EnemyUCS.GetS
tartingPointY())+5, 0,null,"UCSU13","UCSWAP1",null,null,null,0);

p_EnemyUCS.CreateUnitEx(p_EnemyUCS.GetStartingPointX()+2,p_EnemyUCS.GetS
tartingPointY())+5, 0,null,"UCSU13","UCSWAP1",null,null,null,0);

----- EDBase2script.ec -----
mission "translateEDBase2"
{
    consts
    {
        accountResearchBase = 2;
    }

    player p_Player;
    player p_EnemyUCS;
    player p_EnemyLC;

    int i;

    state Initialize;
    state Nothing;

    state Initialize
    {
        unitez uHero;
        //----- Goals -----
        //----- Players -----
        p_Player = GetPlayer(2);
        p_EnemyUCS = GetPlayer(1);
        p_EnemyLC = GetPlayer(3);
        //----- AI -----
        p_EnemyUCS.EnableAIFeatures(aiEnabled,false);
        p_EnemyLC.EnableAIFeatures(aiEnabled,false);
    }

    if(p_Player.GetScriptData(accountResearchBase))//Hash for accomplis
hed mission
    {
        p_Player.AddMoney(p_Player.GetScriptData(accountResearchBase));
        p_Player.SetScriptData(accountResearchBase,0);
    }
    if(p_Player.GetScriptData(4))//Hash for secondary goals mission
    {
        p_Player.AddMoney(p_Player.GetScriptData(4));
        p_Player.SetScriptData(4,0);
    }
    return Nothing,50;
}

//-----
event CustomEvent0(int k1,int k2,int k3,int k4)
{
    if(k4==128)
    {
        EnableUnitSounds(false);
        EnableBuildingSounds(false);
    }
}

```

```

        }
    }

EDbase3script.ec
mission "translateEDBase3"
{
    player p_Player;
    player p_EnemyUCS;
    player p_EnemyLC;

    int i;

    state Initialize;
    state Nothing;
    state Attack1;

    state Initialize
    {
        unitex uHero;
        //----- Goals -----
        //----- Players -----
        p_Player = GetPlayer(2);
        p_EnemyUCS = GetPlayer(1);
        p_EnemyLC = GetPlayer(3);
        //----- AI -----
        p_EnemyUCS.EnableAIFeatures(aiEnabled,false);
        p_EnemyLC.EnableAIFeatures(aiEnabled,false);
        //----- Money -----
        p_Player.SetMoney(0);
        p_EnemyUCS.SetMoney(0);
        p_EnemyLC.SetMoney(0);

        //----- Buildings -----
        p_Player.EnableCommand(commandSoldBuilding,false);
        p_Player.EnableBuilding("EDBPP",true); //IS
        p_Player.EnableBuilding("EDBBA",true); //IS
        p_Player.EnableBuilding("EDBFA",false);
        p_Player.EnableBuilding("EDBW",false);
        p_Player.EnableBuilding("EDBAB",true); //IS
        // 2nd tab
        p_Player.EnableBuilding("EDBRE",false);
        p_Player.EnableBuilding("EDBM",false);
        p_Player.EnableBuilding("EDBTC",false);
        // 3rd tab
        p_Player.EnableBuilding("EDBST",true); //IS
        p_Player.EnableBuilding("EDBBT",true); //IS
        p_Player.EnableBuilding("EDBHT",true); //IS
        p_Player.EnableBuilding("EDBART",false);
        // 4th tab
        p_Player.EnableBuilding("EDBUC",true); //IS
        p_Player.EnableBuilding("EDBRC",false);
        p_Player.EnableBuilding("EDBHQ",false);
        p_Player.EnableBuilding("EDBRA",true); //IS
        p_Player.EnableBuilding("EDBEN1",false);
        p_Player.EnableBuilding("EDBLZ",true); //IS
        //----- Timers -----
        SetTimer(0,20*60*3);
        //----- Camera -----
        p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0,4
5,0);
        SetConsoleText("translateRecycleMessage",50);
        return Nothing,100;
    }
    //-----
    state Attack1
    {
        AddBriefing(" ",p_Player.GetName()); //help atak na centrum wynalazków
        return Nothing,50;
    }
    //-----
    state Nothing
    {
        return Nothing,50;
    }
    //-----
    event Timer0()
    {
        int nPrecent;
        nPrecent = Rand(60)+20;
        p_Player.SetDefaultUnitsRecyclePercent(nPrecent);
        SetConsoleText("translateRecycleMessage",nPrecent);
    }
}

//-----
event CustomEvent0(int k1,int k2,int k3,int k4)
{
    if(k4==128)
    {
        EnableUnitSounds(false);
        EnableBuildingSounds(false);
    }
}

```



```

mission511.ec
mission "translateMission511"
{//Escort convoy
const
{
    escortConvoy = 0;
    destroyCBase = 1;
    destroyAICUnits = 2;
    allConvoySurvived = 3;

    primaryGoal = 0;
    secondaryGoal = 1;
    hiddenGoal = 2;
    endMission = 3;

    accountMainBase = 1;
    accountResearchBase = 2;
    accountCareerPoints = 3;
}

player p_Enemy1;
player p_Enemy2;
player p_Neutral;
player p_Player;
unitex p_ConvoyCraft1;
unitex p_ConvoyCraft2;
unitex p_ConvoyCraft3;
unitex p_ConvoyCraft4;
unitex p_ConvoyCraft5;
unitex p_ConvoyCraft6;

int nWayPoint;
int bCheckEndMission;
int m_bSuccess;
int bIAActivated;
int bShowBriefingB;
//-----
function int Transfer(int account, int value)
{
    p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
}
//-----
function int SetPrize(int reason)
{
    if(reason==primaryGoal)
    {
        Transfer(accountMainBase,3000);
        Transfer(accountResearchBase,15000);
        Transfer(accountCareerPoints,2);
    }
    if(reason==secondaryGoal)
    {
        //Transfer(accountMainBase,3000);
        Transfer(accountCareerPoints,5);
    }
    if(reason==hiddenGoal)/2x
    {
        Transfer(accountMainBase,3000);
        Transfer(accountCareerPoints,2);
    }
    if(reason==endMission)
    {
        if (m_bSuccess)
        {
            Transfer(accountMainBase,p_Player.GetMoney()/2);
            Transfer(accountResearchBase,p_Player.GetMoney()/2);
            p_Player.AddMoney(0 - p_Player.GetMoney());
        }
    }
}

```

```

state Initialize;
state ShowBriefing;
state OnTheWay;
state Fight;
state Final;
//-----
state Initialize
{
    player tmpPlayer;

    m_bSuccess = true;
//----- Goals -----
    if(GetDifficultyLevel() == 0)
        RegisterGoal(escortConvoy,"translateGoal511a",2);
    if(GetDifficultyLevel() == 1)
        RegisterGoal(escortConvoy,"translateGoal511a",4);
    if(GetDifficultyLevel() == 2)
        RegisterGoal(escortConvoy,"translateGoal511a",6);

    RegisterGoal(destroyLCBase,"translateGoal511b");
    RegisterGoal(destroyAllCLUnits,"translateGoal511c");
    RegisterGoal(allConvoySurvived,"translateGoal511d");

    EnableGoal(escortConvoy,true);

//----- Temporary players -----
tmpPlayer = GetPlayer(1);
tmpPlayer.EnableStatistics(false);
//----- Players -----
p_Player = GetPlayer(2);
p_Enemy1 = GetPlayer(3);
p_Enemy2 = GetPlayer(5);
p_Neutral = GetPlayer(4);
//----- AI -----
p_Neutral.EnableStatistics(false);
p_Enemy2.EnableStatistics(false);

p_Enemy1.LoadScript("single\singleDefault");

    if(GetDifficultyLevel() == 2)
        p_Enemy1.LoadScript("single\singleHard");

    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);

    p_Neutral.EnableAIFeatures(aiRejectAlliance,false);

    if(GetDifficultyLevel() == 0)
        p_Player.SetAlly(p_Neutral);

p_Neutral.ChooseEnemy(p_Enemy1);

p_Player.EnableAIFeatures(aiEnabled,false);
p_Enemy1.EnableAIFeatures(aiEnabled,false);
p_Enemy2.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableAIFeatures(aiEnabled,false);

//----- Money -----
p_Player.SetMoney(0);
p_Enemy1.SetMoney(20000);
p_Enemy2.SetMoney(0);
    p_Neutral.SetMoney(0);
//----- Researches -----
p_Enemy1.AddResearch("RES_LC_SGen");
p_Enemy1.AddResearch("RES_LC_MGen");
p_Enemy1.AddResearch("RES_LC_HGen");

p_Player.EnableResearch("RES_ED_UML3",true);
p_Player.EnableResearch("RES_ED_UA22",true);
p_Player.EnableResearch("RES_ED_MGen",true);//shield gen 2
p_Player.EnableResearch("RES_ED_WSR2",true);
p_Player.EnableResearch("RES_ED_EDWAN1",true);
p_Player.EnableResearch("RES_ED_WSL2",true);
p_Player.EnableResearch("RES_ED_WSI2",true);
p_Player.EnableResearch("RES_ED_MSC2",true);
p_Player.EnableResearch("RES_MCH2",true);

//----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,true);
//----- Units -----
p_ConvoyCraft1 = GetUnit(GetPointX(0),GetPointY(0),0);
p_ConvoyCraft2 = GetUnit(GetPointX(1),GetPointY(1),0);
p_ConvoyCraft3 = GetUnit(GetPointX(2),GetPointY(2),0);
p_ConvoyCraft4 = GetUnit(GetPointX(3),GetPointY(3),0);
p_ConvoyCraft5 = GetUnit(GetPointX(4),GetPointY(4),0);
}
}

p_ConvoyCraft6 = GetUnit(GetPointX(5),GetPointY(5),0);
//----- Artefacts -----
if(GetDifficultyLevel() == 0)
{
    KillArea(p_Enemy2.GetIFF(), GetPointX(15), GetPointY(15), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(16), GetPointY(16), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(17), GetPointY(17), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(18), GetPointY(18), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(19), GetPointY(19), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(20), GetPointY(20), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(21), GetPointY(21), 0, 1);
}
if(GetDifficultyLevel() == 1)
{
    KillArea(p_Enemy2.GetIFF(), GetPointX(15), GetPointY(15), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(19), GetPointY(19), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(20), GetPointY(20), 0, 1);
}
}

//----- Timers -----
SetTimer(0,100);
//----- Variables -----
nWayPoint=1;
    bCheckEndMission=false;
    bAIActivated=false;
    bShowBriefingB=true;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
}
return ShowBriefing,100;
}

state ShowBriefing
{
    if(GetDifficultyLevel() == 0)
        AddBriefing("translateBriefing511a",2);

    if(GetDifficultyLevel() == 1)
        AddBriefing("translateBriefing511a",4);

    if(GetDifficultyLevel() == 2)
        AddBriefing("translateBriefing511a",6);

    Snow(GetPointX(6),GetPointY(6),45,400,5000,800,10);
    Snow(GetPointX(7),GetPointY(7),45,400,5000,800,10);
return OnTheWay,300;
}

state OnTheWay
{
    int nGoToPoint;
    int bMakeStep;
    int nTrucksFinished;
    int nTrucksAlive;
    int j;

    bMakeStep=false;

    if(bShowBriefingB)
    {
        for(j=24;j<34 && bShowBriefingB==true;j=j+1)
        {
            if(p_Player.IsPointLocated(GetPointX(j),GetPointY(j),0))
            {
                bShowBriefingB=false;
                EnableGoal(destroyLCBase,true);
                AddBriefing("translateBriefing511b");
            }
        }
    }

    if(nWayPoint==1)
    {
        if(p_Player.IsPointLocated(GetPointX(6),GetPointY(6),0))
        {
            nGoToPoint=6;
            bMakeStep=true;
        }
        if(p_Player.IsPointLocated(GetPointX(7),GetPointY(7),0))
        {
            nGoToPoint=7;
            bMakeStep=true;
        }
    }
}

```

```

if(nWayPoint==2)
{
    if(p_Player.IsPointLocated(GetPointX(8),GetPointY(8,0))
    {
        nGoToPoint=8;
        bMakeStep=true;
        Snow(GetPointX(10),GetPointY(10),45,400,5000,800,10);
        Snow(GetPointX(13),GetPointY(13),45,400,5000,800,10);
    }
    if(p_Player.IsPointLocated(GetPointX(9),GetPointY(9,0)))
    {
        nGoToPoint=9;
        bMakeStep=true;
        Snow(GetPointX(10),GetPointY(10),45,400,5000,800,10);
        Snow(GetPointX(13),GetPointY(13),45,400,5000,800,10);
    }
}
if(GetDifficultyLevel()==2 && nWayPoint==3 && !bAIActivated)
{
    bAIActivated=true;
    p_Enemy1.EnableAIFeatures(aiEnabled,true);
}
if(nWayPoint>2)
{
    nGoToPoint=nWayPoint+7;
}
if(p_Player.IsPointLocated(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0))
    bMakeStep=true;
}
if(bMakeStep || nWayPoint>5)
{
    if(nWayPoint<6)nWayPoint=nWayPoint+1;
}

_p_ConvoyCraft1.CommandMove(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0);
_p_ConvoyCraft2.CommandMove(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0);
_p_ConvoyCraft3.CommandMove(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0);
_p_ConvoyCraft4.CommandMove(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0);
_p_ConvoyCraft5.CommandMove(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0);
_p_ConvoyCraft6.CommandMove(GetPointX(nGoToPoint),GetPointY(nGoToPoint),0);
}
if(nWayPoint>5)
{
    nTrucksAlive=0;
    if(p_ConvoyCraft1.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft2.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft3.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft4.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft5.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft6.IsLive())nTrucksAlive=nTrucksAlive+1;

    nTrucksFinished=0;
    if(p_ConvoyCraft1.IsLive()) &&
_p_ConvoyCraft1.DistanceTo(GetPointX(13),GetPointY(13)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft2.IsLive()) &&
_p_ConvoyCraft2.DistanceTo(GetPointX(13),GetPointY(13)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft3.IsLive()) &&
_p_ConvoyCraft3.DistanceTo(GetPointX(13),GetPointY(13)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft4.IsLive()) &&
_p_ConvoyCraft4.DistanceTo(GetPointX(13),GetPointY(13)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft5.IsLive()) &&
_p_ConvoyCraft5.DistanceTo(GetPointX(13),GetPointY(13)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft6.IsLive()) &&
_p_ConvoyCraft6.DistanceTo(GetPointX(13),GetPointY(13)) <
7)nTrucksFinished=nTrucksFinished+1;

    if(
        nTrucksAlive==nTrucksFinished &&(
        (GetDifficultyLevel()==0 && nTrucksFinished>1) ||
        (GetDifficultyLevel()==1 && nTrucksFinished>3) ||
        (GetDifficultyLevel()==2 && nTrucksFinished>5)))
    {
        SetGoalState(escortConvoy, goalAchieved);
        if(nTrucksFinished==6)
        {
}
}
else
{
    AddBriefing("translateAccomplished511a");
    p_Player.CreateUnitEx(GetPointX(22),GetPointY(22),
0,null,"EDUSPECIAL","EDNSPECIAL",null,null,0);
    p_Player.CreateUnitEx(GetPointX(23),GetPointY(23),
0,null,"EDUSPECIAL","EDNSPECIAL",null,null,0);
    SetPrize(primaryGoal);
    m_bSuccess = true;
    EnableEndMissionButton(true);
    return Fight,100;
}
}
return OnTheWay,300;
}
//-----
state Fight
{
    if(lp_Enemy1.GetNumberOfBuildings())
    {
        SetGoalState(destroyLCBase, goalAchieved);
        AddBriefing("translateAccomplished511c");
        SetPrize(secondaryGoal);
        return Final,500;
    }
}
return Fight,200;
}
//-----
state Final
{
    return Final,500;
}
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    int nLostConvoyCrafts;
    if(!bCheckEndMission) return;
    bCheckEndMission=false;
    if(GetGoalState(escortConvoy)==goalFailed &&
GetGoalState(escortConvoy)==goalAchieved )
    {
        if(lp_ConvoyCraft1.IsLive())nLostConvoyCrafts=1;
        if(lp_ConvoyCraft2.IsLive())nLostConvoyCrafts=nLostConvoyCrafts+1;
        if(lp_ConvoyCraft3.IsLive())nLostConvoyCrafts=nLostConvoyCrafts+1;
        if(lp_ConvoyCraft4.IsLive())nLostConvoyCrafts=nLostConvoyCrafts+1;
        if(lp_ConvoyCraft5.IsLive())nLostConvoyCrafts=nLostConvoyCrafts+1;
        if(lp_ConvoyCraft6.IsLive())nLostConvoyCrafts=nLostConvoyCrafts+1;

        if(
            ((GetDifficultyLevel()==0 && nLostConvoyCrafts>4) ||
            ((GetDifficultyLevel()==1 && nLostConvoyCrafts>2) ||
            ((GetDifficultyLevel()==2 && nLostConvoyCrafts>0)))
        {
            SetGoalState(escortConvoy, goalFailed);
            AddBriefing("translateFailed511a");
            m_bSuccess = false;
            EnableEndMissionButton(true,false);
            state Final;
        }
}
if(GetGoalState(destroyAllCUnits)==goalAchieved &&
lp_Enemy2.GetNumberOfUnits()&& lp_Enemy1.GetNumberOfUnits())
{
    EnableGoal(destroyAllCUnits,true);
    SetGoalState(destroyAllCUnits, goalAchieved);
    SetPrize(hiddenGoal);
}
if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
{
    if(GetGoalState(escortConvoy)==goalAchieved)
}
}

```

```

    {
        AddBriefing("translateFailed511b");
        m_bSuccess = false;
        EndMission(false);
    }
    else
    {
        m_bSuccess = true;
        EndMission(true);
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    SetPrize(endMission);
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
}
}

mission512.ec
mission "translateMission512"
{//Rescue mission

// stoi artefakt kolo niego unit (amfibja)
// odnalezienie ich przed upwem 4 godzin (2 goale zaliczone)
// odnalezienie po tym terminie 1 goal zaliczony.
// strata bazy - przegrana
// odnalezienie i zniszczenie wszystkich UCSów - hidden goal.
// po 2 dniach bombowce UCS przyląduj i robi 1 nalot (startuje z markera 7)
// easy 1 med 2 hard 3.

consts
{
    findTransmitter = 0;
    rescueSpy = 1;
    destroyAllUCSUnits = 2;

    primaryGoal = 0;
    secondaryGoal = 1;
    hiddenGoal = 2;
    endMission = 3;

    accountMainBase = 1;
    accountResearchBase = 2;
    accountCareerPoints = 3;
}

player p_Enemy;
player p_Neutral;
player p_Player;
unitex u_Radar1;
unitex u_Radar2;
unitex u_Rescuer;

int nWayPoint;
int bCheckEndMission;
int nRescueTime;
int nAttackTime;
int n_Transmitter_X;
int n_Transmitter_Y;
int m_bSuccess;
int nTimeLeft;

//-----
function int Transfer(int account, int value)
{
    p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
}
//-----
function int SetPrize(int reason)
{
    if(reason==primaryGoal)
    {
        Transfer(accountMainBase,0);
        Transfer(accountResearchBase,15000);
        Transfer(accountCareerPoints,2);
    }
    if(reason==secondaryGoal)
    {
        Transfer(accountMainBase,3000);
        Transfer(accountCareerPoints,7);
    }
    if(reason==hiddenGoal)
    {
        Transfer(accountMainBase,3000);
        Transfer(accountCareerPoints,1);
    }
    if(reason==endMission)
    {
        if (m_bSuccess)
        {
            Transfer(accountMainBase,p_Player.GetMoney()/2);
            Transfer(accountResearchBase,p_Player.GetMoney()/2);
            p_Player.AddMoney(0 - p_Player.GetMoney());
        }
    }
}
//-----

state Initialize;
state PostInitialize;
state ShowBriefing;
state Searching;
state Escaping;
//-----
state Initialize
{
    player tmpPlayer;

    m_bSuccess = true;
    if(GetDifficultyLevel()==0)
    {
        nRescueTime=12;//godzin
        nAttackTime=18;//godzin gry
    }
    if(GetDifficultyLevel()==1)
    {
        nRescueTime=6;//godzin
        nAttackTime=12;//godzin gry
    }
    if(GetDifficultyLevel()==2)
    {
        nRescueTime=3;//godzin
        nAttackTime=9;//godzin gry
    }
    nTimeLeft=nRescueTime;
    //----- Goals -----
    RegisterGoal(findTransmitter,"translateGoal512a");
    RegisterGoal(rescueSpy,"translateGoal512b",nRescueTime);
    RegisterGoal(destroyAllUCSUnits,"translateGoal512c");

    EnableGoal(findTransmitter,true);
    EnableGoal(rescueSpy,true);
    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(1);
    p_Neutral = GetPlayer(4);
    //----- AI -----
    p_Neutral.EnableStatistics(false);
    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);

    p_Enemy.LoadScript("single\singleDefault");

    p_Enemy.SetNeutral(p_Neutral);
    p_Neutral.SetNeutral(p_Enemy);

    p_Neutral.EnableAIFeatures(aiEnabled,false);
    p_Enemy.EnableAIFeatures(aiEnabled,false);

    //----- Money -----
    p_Player.SetMoney(0);
    p_Enemy.SetMoney(0);
    p_Neutral.SetMoney(0);
}
}

```

```

//----- Researches -----
p_Player.EnableResearch("RES_ED_UMW2",true);
p_Player.EnableResearch("RES_EDUUT",true);
p_Player.EnableResearch("RES_ED_ACH2",true);
p_Player.EnableResearch("RES_ED_WSL3",true);
p_Player.EnableResearch("RES_ED_MSC3",true);
p_Player.EnableResearch("RES_MCH3",true);
p_Player.EnableResearch("RES_ED_SCR",true);

//----- Buildings -----
p_Player.EnableBuilding("EDBRC",false);

//----- Units -----
u_Radar2 = GetUnit(GetPointX(0),GetPointY(0),0);
u_Radar1 = GetUnit(GetPointX(1),GetPointY(1),0);
u_Radar1.SetUnitName("translate512scanner1");
u_Radar2.SetUnitName("translate512scanner2");
p_Player.AddUnitToSpecialTab(u_Radar1,true,-1);
p_Player.AddUnitToSpecialTab(u_Radar2,true,-1);

//----- Artifacts -----
//JS - przeniesienie tworzenia artefaktow do drugiego state'a po to
//aby przy restart generowalo za kazdym razem inną pozycje
//(bo sa restart jest po pierwszym stepie AIWorld'a)

    if(GetDifficultyLevel()>0)//medium
    { //jednostki wrogia
        p_Enemy.CreateUnitEx(GetPointX(2)-1,GetPointY(2)-1,
0,null,"UCSUSL3","UCSWTSP2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(3)-1,GetPointY(3)-1,
0,null,"UCSUSL3","UCSWTSP2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(4)-1,GetPointY(4)-1,
0,null,"UCSUSL3","UCSWTSP2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(5)-1,GetPointY(5)-1,
0,null,"UCSUSL3","UCSWTSP2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(6)-1,GetPointY(6)-1,
0,null,"UCSUSL3","UCSWTSP2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(7)-1,GetPointY(7)-1,
0,null,"UCSUSL3","UCSWTSP2",null,null,null,0);
    }
    if(GetDifficultyLevel()==2)//hard
    {
        p_Enemy.CreateUnitEx(GetPointX(2),GetPointY(2)-1,
0,null,"UCSUSL3","UCSWTSR2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(3),GetPointY(3)-1,
0,null,"UCSUSL3","UCSWTSR2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(4),GetPointY(4)-1,
0,null,"UCSUSL3","UCSWTSR2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(5),GetPointY(5)-1,
0,null,"UCSUSL3","UCSWTSR2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(6),GetPointY(6)-1,
0,null,"UCSUSL3","UCSWTSR2",null,null,null,0);
        p_Enemy.CreateUnitEx(GetPointX(7),GetPointY(7)-1,
0,null,"UCSUSL3","UCSWTSR2",null,null,null,0);
    }

    //----- Timers -----
SetTimer(0,100);
//----- Variables -----
nWayPoint=1;
bCheckEndMission=false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return PostInitialize0;
}
//----- PostInitialize -----
state PostInitialize
{
    int n_Rnd;
    //----- Artifacts -----
    n_Rnd = Rand(6)+2;
    TraceD("Rnd = ");TraceD(n_Rnd);TraceD("      \n");
    n_Transmitter_X = GetPointX(n_Rnd);
    n_Transmitter_Y = GetPointY(n_Rnd);
    p_Neutral.CreateUnitEx(n_Transmitter_X+1,n_Transmitter_Y,
0,null,"EDUMW3","EDWCH1",null,null,null,0);

CreateArtifact("NEASPECIAL1",n_Transmitter_X,n_Transmitter_Y,0,0,artefactSpecia
lOther);
return ShowBriefing99;
}
//----- ShowBriefing -----
state ShowBriefing
{
    EnableNextMission(0,true);///513
    AddBriefing("translateBriefing512",nRescueTime);
    Snow(GetPointX(0),GetPointY(0),45,400,5000,800,10);
    return Searching,1;
}
//----- Searching -----
state Searching
{
    int n_Dist1;
    int n_Dist2;
    n_Dist1=u_Radar1.DistanceTo(n_Transmitter_X,n_Transmitter_Y);
    n_Dist2=u_Radar2.DistanceTo(n_Transmitter_X,n_Transmitter_Y);
    SetConsoleText("translateMessage512",n_Dist1*n_Dist2*8,nTimeLeft);
    if(GetGoalState(findTransmitter)==goalAchieved)
    {
        SetConsoleText("");
        return Escaping,20;
    }
    return Searching,20;
}
//----- Escaping -----
state Escaping
{
    if(GetGoalState(rescueSpy)==goalNotAchieved && u_Rescuer!=null)
    {
        if(u_Rescuer.DistanceTo(GetPointX(0),GetPointY(0))<7)
        {
            SetGoalState(rescueSpy,goalAchieved);
            SetPrize(secondaryGoal);
            AddBriefing("translateAccomplished512c");
            u_Rescuer=null;
        }
    }
    return Escapeing,20;
}
//----- Escapeing -----
event Timer0() //wolany co 100 cykli< ustwione funkcja SetTimer w state
Initialize
{
    //clicksPerDay = 16383;
    //clicksPerHour = 683;
    nTimeLeft = nRescueTime*683 - GetMissionTime();
    if(nTimeLeft<0)
    {
        nTimeLeft=0;
        if(GetGoalState(findTransmitter)==goalNotAchieved)
            SetGoalState(rescueSpy,goalFailed);
    }
    else
        nTimeLeft=(nTimeLeft/683)+1;
    RegisterGoal(rescueSpy,"translateGoal512b",nTimeLeft);

    if(nAttackTime && (nAttackTime*683)<GetMissionTime())
    {//atak UCS
        nAttackTime=0;
        p_Enemy.CreateUnitEx(GetPointX(0),GetPointY(0)+8,
0,null,"UCSUA32","UCSWAPB2",null,null,null,0);
        if(GetDifficultyLevel()==1)
            p_Enemy.CreateUnitEx(GetPointX(0)+1,GetPointY(0)+8,
0,null,"UCSUA32","UCSWAPB2",null,null,null,0);
        if(GetDifficultyLevel()==2)
            p_Enemy.CreateUnitEx(GetPointX(0)-1,GetPointY(0)+8,
0,null,"UCSUA32","UCSWAPB2",null,null,null,0);
    }
    if(!bCheckEndMission) return;
    bCheckEndMission=false;
}

```

```

        if(GetGoalState(rescueSpy)==goalNotAchieved && u_Rescuer!=null)
        {
            if(!u_Rescuer.IsLive())
            {
                SetGoalState(rescueSpy,goalFailed);
                u_Rescuer=null;
            }
        }

        if(GetGoalState(destroyAllUCSUnits)==goalAchieved &&
lp_Enemy.GetNumberOfUnits())
        {
            EnableGoal(destroyAllUCSUnits,true);
            SetGoalState(destroyAllUCSUnits, goalAchieved);
            SetPrize(hiddenGoal);
        }

        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            if(GetGoalState(findTransmiser)==goalAchieved)
            {
                AddBriefing("translateFailed512");
                m_bSuccess = false;
                EndMission(false);
            }
            else
            {
                m_bSuccess = true;
                EndMission(true);
            }
        }
    //-----
    event UnitDestroyed(unit u_Unit)
    {
        bCheckEndMission=true;
    }
    //-----
    event BuildingDestroyed(unit u_Unit)
    {
        bCheckEndMission=true;
    }
    //-----
    event EndMission()
    {
        SetPrize(endMission);
        p_Neutral.SetEnemy(p_Player);
        p_Player.SetEnemy(p_Neutral);
        p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
    }
    //-----
    event Artefact(int alD,player piPlayer)
    {
        if(piPlayer!=p_Player) return false;
        EnableNextMission(1,true); //514
        SetGoalState(findTransmiser,goalAchieved);
        if(GetGoalState(rescueSpy)==goalNotAchieved)
        {
            u_Rescuer = GetUnit(n_Transmitter_X,n_Transmitter_Y);
            AddBriefing("translateAccomplished512a");
        }
        else
        {
            AddBriefing("translateAccomplished512b");
        }
        SetPrize(primaryGoal);
        m_bSuccess = true;
        EnableEndMissionButton(true);
        return true; //usuwa sie
    }
}

mission513.ec
mission "translateMission513"
{
    consts
    {
        findTargetForBombers = 0;
        destroyLBase1 = 1;
        findBuilders = 2;
        destroyLBase2 = 3;
        findPositionForArtillery = 4;
        destroyLBase3 = 5;
        destroyAllLCUnits = 6;
        primaryGoal = 0;
        hiddenGoal = 2;
        endMission = 3;
        accountMainBase = 1;
        accountResearchBase = 2;
        accountCareerPoints = 3;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;
    player p_Enemy4;
    player p_Neutral;/builders and bombers
    player p_Ally;/bombers
    player p_Player;
    unitex u_Builder1;
    unitex u_Builder2;

    int bCheckEndMission;
    int bAccomplisedShowed;
    int m_bSuccess;
    //-----
    function int Transfer(int account, int value)
    {
        p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
    }
    //-----
    function int SetPrize(int reason)
    {
        if(reason==primaryGoal)
        {
            Transfer(accountMainBase,10000);
            Transfer(accountResearchBase,15000);
            Transfer(accountCareerPoints,8);
        }
        if(reason==hiddenGoal)
        {
            Transfer(accountMainBase,3000);
            Transfer(accountCareerPoints,3);
        }
        if(reason==endMission)
        {
            if (m_bSuccess)
            {
                Transfer(accountMainBase,p_Player.GetMoney()/2);
                Transfer(accountResearchBase,p_Player.GetMoney()/2);
                p_Player.AddMoney(0 - p_Player.GetMoney());
            }
        }
    }

    state Initialize;
    state ShowBriefing;
    state Searching;
    state FindingGrazes;
    state FinalBattle;
    state AttackOnBase;
    state Nothing;
    //-----
    state Initialize
    {
        player tmpPlayer;
        m_bSuccess = true;
        //----- Goals -----
        RegisterGoal(findTargetForBombers,"translateGoal513a");
        RegisterGoal(destroyLBase1,"translateGoal513b");
        RegisterGoal(findBuilders,"translateGoal513c");
        RegisterGoal(destroyLBase2,"translateGoal513d");
        RegisterGoal(findPositionForArtillery,"translateGoal513e");
        RegisterGoal(destroyLBase3,"translateGoal513f");
        RegisterGoal(destroyAllLCUnits,"translateGoal513g");
        EnableGoal(findTargetForBombers,true);

        //----- Temporary players -----
        tmpPlayer = GetPlayer(1);
        tmpPlayer.EnableStatistics(false);
        //----- Players -----
        p_Player = GetPlayer(2);
        p_Enemy1 = GetPlayer(3);
        p_Enemy2 = GetPlayer(5);
    }
}

```

```

p_Enemy3 = GetPlayer(8);
p_Enemy4 = GetPlayer(9);
p_Neutral = GetPlayer(4);
p_Ally = GetPlayer(6);
//----- AI -----
p_Neutral.EnableStatistics(false);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);
p_Enemy1.SetNeutral(p_Neutral);
p_Enemy2.SetNeutral(p_Neutral);

if(GetDifficultyLevel() == 0)
    p_Enemy1.LoadScript("single\singleEasy");
if(GetDifficultyLevel() == 1)
    p_Enemy1.LoadScript("single\singleMedium");
if(GetDifficultyLevel() == 2)
    p_Enemy1.LoadScript("single\singleHard");

p_Ally.LoadScript("single\singleDefault");
p_Enemy1.LoadScript("single\singleDefault");
p_Enemy2.LoadScript("single\singleDefault");
p_Enemy3.LoadScript("single\singleDefault");
p_Enemy4.LoadScript("single\singleDefault");

p_Ally.EnableAIFeatures(aiEnabled,false);
p_Ally.EnableAIFeatures(aiRejectAlliance,false);
p_Player.SetAlly(p_Ally);

p_Neutral.EnableAIFeatures(aiEnabled,false);

p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);

p_Enemy3.EnableAIFeatures(aiBuildBuildings,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);

p_Enemy4.EnableAIFeatures(aiEnabled,false);

p_Ally.SetAlly(p_Neutral);
p_Enemy1.SetAlly(p_Neutral);
p_Enemy2.SetAlly(p_Neutral);
p_Enemy3.SetAlly(p_Neutral);
p_Enemy4.SetAlly(p_Neutral);

p_Enemy4.SetAlly(p_Enemy1);
p_Enemy4.SetAlly(p_Enemy2);
p_Enemy4.SetAlly(p_Enemy3);
p_Enemy1.SetAlly(p_Enemy2);
p_Enemy1.SetAlly(p_Enemy3);
p_Enemy2.SetAlly(p_Enemy3);

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlOffense,false);

//----- Money -----
p_Player.SetMoney(0);
if(GetDifficultyLevel() == 1)
{
    p_Enemy1.SetMoney(20000);
    p_Enemy2.SetMoney(20000);
    p_Enemy3.SetMoney(20000);
}
if(GetDifficultyLevel() == 2)
{
    p_Enemy1.SetMoney(100000);
    p_Enemy2.SetMoney(120000);
    p_Enemy3.SetMoney(120000);
}
p_Neutral.SetMoney(0);
p_Ally.SetMoney(100000);
//----- Researches -----
p_Player.EnableResearch("RES_ED_ART",true); //zeby nie bylo jakis rese-
chow
p_Player.AddResearch("RES_ED_ART");
p_Player.EnableResearch("RES_ED_ART",false); //zeby nie bylo widac go
wynalezioneego w drzewie
p_Player.EnableResearch("RES_ED_BCI",true);

p_Player.EnableResearch("RES_ED_HGen",true); //shield gen 3
p_Player.EnableResearch("RES_ED_JM1",true);
p_Player.EnableResearch("RES_ED_JM2",true);
p_Player.EnableResearch("RES_ED_WSR3",true); //SR3
p_Player.EnableResearch("RES_ED_ASR1",true); //helicopter rocket laun-
cher

p_Ally.AddResearch("RES_ED_AB1");
p_Ally.AddResearch("RES_ED_AB2");
//----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,true); // 1st
tab
p_Player.EnableBuilding("EDBPP",false);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBMA",false);
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
p_Player.EnableBuilding("EDBBT",false);
p_Player.EnableBuilding("EDBHT",false);
p_Player.EnableBuilding("EDBART",false);
// 4th tab
p_Player.EnableBuilding("EDBUC",false);
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHQ",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBN1",true);
p_Player.EnableBuilding("EDBLZ",true);

//----- Artifacts -----
CreateArtifact("NEACOMPUTER",GetPointX(4),GetPointY(4),GetPointZ(4),0,artefact
SpecialAIOther);

CreateArtifact("NEASWITCH2",GetPointX(5),GetPointY(5),GetPointZ(5),1,artefactS
pecialAIOther);
//----- Units -----
u_Builder1 = GetUnit(GetPointX(2),GetPointY(2),GetPointZ(2));
u_Builder2 = GetUnit(GetPointX(3),GetPointY(3),GetPointZ(3));
//----- Units -----
if(GetDifficultyLevel() < 2) //easy and medium
{
    KillArea(p_Enemy4.GetFFI(), GetPointX(10), GetPointY(10), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(11), GetPointY(11), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(12), GetPointY(12), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(13), GetPointY(13), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(14), GetPointY(14), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(15), GetPointY(15), 0, 2);
}
if(GetDifficultyLevel() > 1) //easy
{
    KillArea(p_Enemy4.GetFFI(), GetPointX(16), GetPointY(16), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(17), GetPointY(17), 0, 2);
    KillArea(p_Enemy4.GetFFI(), GetPointX(18), GetPointY(18), 0, 2);
}

//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission = false;
bAccomplishedShowed = false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing513a");
    Rain(GetPointX(0),GetPointY(0),45,400,5000,800,5);
    return Searching,1;
}
//-----
state Searching
{
    unitex u_Bomber;
    int x;
    int y;
    int n_X;
}

```

```

int n_Y;
if(p_Player.IsPointLocated(GetPointX(0),GetPointY(0),0))
{
    if(GetDifficultyLevel()==2)
        p_Enemy1.EnableAIFeatures(aiControlOffense,true);
    SetGoalState(findTargetForBombers,goalAchieved);
    EnableGoal(destroyLCBase1,true);
    EnableGoal(findBuilders,true);
    SetConsoleText("translateMessage513");//Bomber attack launched
    AddBriefing("translateBriefing513b");
    for(x=-1;x<=x+x+1)
    for(y=-1;y<=y+y+1)
    {
        n_X = GetPointX(1)+x;
        n_Y = GetPointY(1)+y;
        u_Bomber=p_Ally.CreateUnitEx(n_X,n_Y,
0,null,"EDUA32","EDWA2Z",null,null,0);
        u_Bomber.CommandMove(GetPointX(0),GetPointY(0),0);
    }
    return FindingGruzes,1000;
}
return Searching,20;
}
//-----
state FindingGruzes
{
    SetConsoleText("");
    return FindingGruzes;
}
//-----
state FinalBattle
{
    if(GetGoalState(destroyLCBase2)==goalAchieved &&
    p_Player.IsPointLocated(GetPointX(7),GetPointY(7),0))
    {
        p_Player.EnableBuilding("EDBART",true);
        p_Player.EnableBuilding("EDBTP",true);
        p_Player.EnableBuilding("EDBAP",true);
        p_Player.EnableBuilding("EDBAA",true);
        p_Player.AddMoney(25000);
        SetGoalState(findPositionForArtillery,goalAchieved);
        AddBriefing("translateBriefing513c");//build artillery
        p_Player.LookAt(GetPointX(7),GetPointY(7),10,128,20,0);
        p_Player.DelayedLookAt(GetPointX(7),GetPointY(7),10,0,20,0,100,1);

        if (GetDifficultyLevel()==1)
            Storm(GetPointX(7),GetPointY(7),15,300,5000,1000,5,3,1);
        if (GetDifficultyLevel()==2)
            Storm(GetPointX(7),GetPointY(7),15,300,5000,1000,5,3,3);

        return AttackOnBase,1200;
    }
    return FinalBattle;
}
//-----
state AttackOnBase
{
    p_Player.SetScriptData(7,1);
    return Nothing;
}
//-----
state Nothing
{
    return Nothing;
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(!bCheckEndMission) return;
    bCheckEndMission=false;

    if(GetGoalState(destroyAllCUnits)==goalAchieved &&
    lp_Enemy1.GetNumberOfUnits())
    {
        EnableGoal(destroyAllCUnits,true);
        SetGoalState(destroyAllCUnits, goalAchieved);
        SetPrize(hiddenGoal);
    }
    //JS zmiana playerow bo byli nie tacy jak trzeba
    if(lp_Enemy2.GetNumberOfBuildings() &&
    GetGoalState(destroyLCBase1)!=goalAchieved)
    {
        SetGoalState(destroyLCBase1,goalAchieved);
        p_Ally.GiveAllBuildingsTo(p_Player); //xxmd
        //IS wykomentowane bo LC od razu zabijali tunel u gory po prawej
        //p_Neutral.GiveAllBuildingsTo(p_Player);
        if(GetDifficultyLevel()==0)
        {
            p_Neutral.EnableAIFeatures(aiRejectAlliance,false);
            p_Player.SetAlly(p_Neutral);
        }
        if (GetDifficultyLevel()==0)
            Storm(GetPointX(1),GetPointY(1),15,300,5000,1000,5,5,1);
        if (GetDifficultyLevel()==1)
            Storm(GetPointX(1),GetPointY(1),15,300,5000,1000,5,5,5);
        if (GetDifficultyLevel()==2)
            Storm(GetPointX(1),GetPointY(1),15,300,5000,1000,5,5,10);
    }

    if(lp_Enemy3.GetNumberOfBuildings() &&
    GetGoalState(destroyLCBase2)==goalAchieved)
    {
        SetGoalState(destroyLCBase2,goalAchieved);
        EnableGoal(findPositionForArtillery,true);
        EnableGoal(destroyLCBase3,true);
    }

    if(lp_Enemy1.GetNumberOfBuildings() &&
    GetGoalState(destroyLCBase3)==goalAchieved)
    {
        SetGoalState(destroyLCBase3,goalAchieved);
    }

    if(!bAccomplishedShowed &&
    GetGoalState(destroyLCBase1)==goalAchieved &&
    GetGoalState(destroyLCBase2)==goalAchieved &&
    GetGoalState(destroyLCBase3)==goalAchieved)
    {
        bAccomplishedShowed=true;
        SetPrize(primaryGoal);
        AddBriefing("translateAccomplished513");
        m_bSuccess = true;
        EnableEndMissionButton(true);
    }
}

if(lp_Player.GetNumberOfUnits() &&lp_Player.GetNumberOfBuildings())
{
    if(GetGoalState(destroyLCBase1)==goalAchieved || |
    GetGoalState(destroyLCBase2)==goalAchieved || |
    GetGoalState(destroyLCBase3)==goalAchieved)
    {
        AddBriefing("translateFailed513");
        m_bSuccess = false;
        EndMission(false);
    }
    else
    {
        m_bSuccess = true;
        EndMission(true);
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    SetPrize(endMission);
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
}

//-----
event Artefact(int aID,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(aID==0)//builders
    {
        u_Builder1.ChangePlayer(p_Player);
    }
}

```

```

    u_Builder2.ChangePlayer(p_Player);
    p_Player.AddMoney(5000);
    SetGoalState(findBuilders,goalAchieved);
    EnableGoal(destroyLCbase2,true);
    AddBriefing("translateBriefing513GoalGruz");
    statFinalBattle;
    return true; //usuwa sie
}
if(alD==1)//detonator
{
    KillArea(256,GetPointX(6),GetPointY(6),GetPointZ(6),4);
CreateArtefact("NEASWITCH1",GetPointX(5),GetPointY(5),GetPointZ(5),4,artefactSpecialAIother);
    AddBriefing("translateBriefing513GoalDetonator");
    return true; //usuwa sie
}
return false; //usuwa sie
}
}

```

mission514.ec

```

//1 doprowadzic ciezarowke do startingpointa gracza 4 i przejcie jego unitu

```

```
//Artefakty na markerach 1-30 to miny
```

```
//Hard 1-30
```

```
//Medium 11-30
```

```
//Easy 11-20
```

```
//AI dostaje dzia³ka przeciwlotnicze i plazme oraz rakiety naprowadzane
//My'accone ma budowanie budynków.
```

```
//Budowanie budynków zostaje w³acone po przejêciu pojazdu.
```

```
//Na Easy Gracz4 jest Ally i show area elektrowni.
```

```
//Na medium jest show area gracza 4
```

```
//NA hardzie ¯i ma w³acone offense (na medium i easy wy³acone)
```

```
mission "translateMission514"
```

```
{
```

```
consts
```

```
{
    deliverPilot = 0;
    evacuateCargoSalamander = 1;
    destroyAllPowerPlants = 2;
    destroyAllBuildings = 3;
    destroyAllUnits = 4;
}
```

```
primaryGoal = 0;
secondaryGoal = 1;
hiddenGoal = 2;
endMission = 3;
```

```
accountMainBase = 1;
accountResearchBase = 2;
accountCareerPoints = 3;
```

```
}
```

```
player p_Enemy;
player p_Neutral;/CargoSalamander
player p_Player;
```

```
unitex u_Truck;
unitex u_CargoSalamander;
```

```
int bCheckEndMission;
int bAccomplishedShowed;
int m_bSuccess;
```

```
function int Transfer(int account, int value)
```

```
{
    p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
}
```

```
function int SetPrize(int reason)
```

```
{
    if(reason==primaryGoal)
    {
        Transfer(accountMainBase,0);
        Transfer(accountResearchBase,15000);
        Transfer(accountCareerPoints,2);
    }
}
```

```

    if(reason==secondaryGoal)
    {
        Transfer(accountMainBase,3000);
        Transfer(accountCareerPoints,10);
    }
    if(reason==hiddenGoal)
    {
        Transfer(accountMainBase,5000);
        Transfer(accountCareerPoints,3);
    }
    if(reason==endMission)
    {
        if (m_bSuccess)
        {
            Transfer(accountMainBase,p_Player.GetMoney()/2);
            Transfer(accountResearchBase,p_Player.GetMoney()/2);
            p_Player.AddMoney(0 - p_Player.GetMoney());
        }
    }
}

```

```
state Initialize;
```

```
state ShowBriefing;
```

```
state Delivering;
```

```
state ShowVideo;
```

```
state Escaping;
```

```
state Nothing;
```

```
state EndMissionFailed;
```

```
-----
```

```
state Initialize
```

```
{
    player tmpPlayer;
```

```
int i;
```

```
int start;
```

```
int end;
```

```
m_bSuccess = true;
```

```
----- Goals -----
```

```
RegisterGoal(deliverPilot,"translateGoal514a");
RegisterGoal(evacuateCargoSalamander,"translateGoal514b");
RegisterGoal(destroyAllPowerPlants,"translateGoal514c");
RegisterGoal(destroyAllBuildings,"translateGoal514d");
RegisterGoal(destroyAllUnits,"translateGoal514e");
```

```
EnableGoal(deliverPilot,true);
```

```
EnableGoal(destroyAllPowerPlants,true);
```

```
----- Temporary players -----
```

```
tmpPlayer = GetPlayer(3);
```

```
tmpPlayer.EnableStatistics(false);
```

```
----- Players -----
```

```
p_Player = GetPlayer(2);
```

```
p_Enemy = GetPlayer(1);
```

```
p_Neutral = GetPlayer(4);
```

```
----- AI -----
```

```
p_Neutral.EnableStatistics(false);
```

```
p_Neutral.SetNeutral(p_Player);
```

```
p_Player.SetNeutral(p_Neutral);
```

```
p_Enemy.SetNeutral(p_Neutral);
```

```
p_Neutral.SetNeutral(p_Enemy);
```

```
p_Neutral.EnableAIFeatures(aiEnabled,false);
```

```
// Na Easy Gracz4 jest Ally i show area elektrowni.
```

```
// Na medium jest show area gracza 4
```

```
// NA hardzie ¯i ma w³acone offense (na medium i easy wy³acone)
```

```
if(GetDifficultyLevel()==0)
```

```
{
    p_Neutral.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_Neutral);
    p_Enemy.LoadScript("single\singleEasy");
    p_Enemy.EnableAIFeatures(aiControlOffense,false);
}
```

```
if(GetDifficultyLevel()==1)
```

```
{
    p_Enemy.LoadScript("single\singleMedium");
```

```
ShowArea(4,_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),0.3);
```

```
p_Enemy.EnableAIFeatures(aiControlOffense,false);
```

```
}
```

```
if(GetDifficultyLevel()==2)
```

```

    p_Enemy.LoadScript("single\singleHard");

```

```
p_Enemy.EnableAIFeatures(aiBuildBuildings,false);
```

```

        //----- Money -----
p_Player.SetMoney(0);
if(GetDifficultyLevel() == 1)
{
    p_Enemy.SetMoney(20000);
}
if(GetDifficultyLevel() == 2)
{
    p_Enemy.SetMoney(100000);
}
p_Neutral.SetMoney(0);
//----- Researches -----
p_Enemy.AddResearch("RES_UCS_WSP1");
p_Enemy.AddResearch("RES_UCS_WSP2");
p_Enemy.AddResearch("RES_UCS_WSR1");
p_Enemy.AddResearch("RES_UCS_WSR2");
p_Enemy.AddResearch("RES_UCS_WSR3");
p_Enemy.AddResearch("RES_UCSAA1");
p_Enemy.AddResearch("RES_UCSAA2");
p_Enemy.AddResearch("RES_UCS_SGen");
p_Enemy.AddResearch("RES_UCS_MGen");
p_Enemy.AddResearch("RES_UCS_HGen");

p_Player.EnableResearch("RES_ED_ASР2",true); //helicopter rocket launcher
p_Player.EnableResearch("RES_ED_AMR1",true); //helicopter rocket launcher
p_Player.EnableResearch("RES_ED_SCR3",true);
p_Player.EnableResearch("RES_ED_UJA1",true);
p_Player.EnableResearch("RES_EDUSTEALTH",true);

//----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,true);
    // 1st tab
p_Player.EnableBuilding("EDBPP",false);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDWBW",false);
p_Player.EnableBuilding("EDBAB",false);
    // 2nd tab
p_Player.EnableBuilding("EDBST",false);
p_Player.EnableBuilding("EDBBT",false);
p_Player.EnableBuilding("EDBHT",false);
p_Player.EnableBuilding("EDBART",false);
    // 3rd tab
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHO",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBEN1",false);
p_Player.EnableBuilding("EDBLZ",true);

//----- Artefacts -----
//Artefakty na markerach 1-30 to miny
//Easy 11-20
//Medium 11-30
//Hard 1-30

if(GetDifficultyLevel() == 0)
{
    start=11;
    end=20;
}
if(GetDifficultyLevel() == 1)
{
    start=11;
    end=30;
}
if(GetDifficultyLevel() == 2)
{
    start=1;
    end=30;
}
for(i=start;i<=end;i+=1)
{
    CreateArtifact("NEAMINE",GetPointX(i),GetPointY(i),GetPointZ(i),i,artifactSpecialAI
Other);
}

//----- Units -----
u_Truck = GetUnit(GetPointX(0),GetPointY(0),0);
u_CargoSalamander =
GetUnit(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),0);
u_CargoSalamander.LoadScript("Scripts\Units\Tank.ecomp");
p_Player.SetScriptUnit(0,u_CargoSalamander);

if(GetDifficultyLevel() == 0)
{
    KillArea(p_Enemy.GetIFF(), GetPointX(31), GetPointY(31), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(32), GetPointY(32), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(33), GetPointY(33), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(34), GetPointY(34), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(35), GetPointY(35), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(36), GetPointY(36), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(37), GetPointY(37), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(38), GetPointY(38), 0, 0);
}
if(GetDifficultyLevel() == 1)
{
    KillArea(p_Enemy.GetIFF(), GetPointX(33), GetPointY(33), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(35), GetPointY(35), 0, 0);
    KillArea(p_Enemy.GetIFF(), GetPointX(37), GetPointY(37), 0, 0);
}

//----- Campaign Unit -----
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission = false;
bAccomplishedShowed = false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0);
    return ShowBriefing,100;
}
state ShowBriefing
{
    EnableNextMission(0,true); //516
    AddBriefing("translateBriefing5_14a");
    Rain(GetPointX(0),GetPointY(0),45,400,5000,800,5);
    return Delivering,1;
}
state Delivering
{
    if(u_Truck.DistanceTo(p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY()) < 2)
    {
        SetGoalState(deliverPilot.goalAchieved);
        EnableGoal(evacuateCargoSalamander,true);
        AddBriefing("translateBriefing5_14b");
        u_CargoSalamander.ChangePlayer(p_Player);
        if(GetDifficultyLevel() == 2)
            Storm(GetPointX(20),GetPointY(20),300,5000,1000,5,3,3);
        if(GetDifficultyLevel() == 1)
            Storm(GetPointX(20),GetPointY(20),300,5000,1000,5,1,1);
    }
    p_Player.EnableBuilding("EDBPP",true);
    p_Player.EnableBuilding("EDBAB",true);

    return ShowVideo;
}
return Delivering,20;
}
state ShowVideo
{
    ShowVideo("Cutscene18");
    return Escaping;
}
state Escaping
{
    if(u_CargoSalamander.IsLive() &&
u_CargoSalamander.IsInWorld(WorldNum()))
    {
        SetPrize(primaryGoal);
        EnableNextMission(1,true); //515
        SetGoalState(evacuateCargoSalamander.goalAchieved);
    }
}

```

```

        AddBriefing("translateAccomplished514");
        m_bSuccess = true;
        EnableEndMissionButton(true);
        return Nothing;
    }
    return Escaping;
}
//-----
state Nothing
{
    return Nothing;
}
//-----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(!bCheckEndMission) return;
    bCheckEndMission=false;

    if(GetGoalState(destroyAllUnits)!=goalAchieved &&
lp_Enemy.GetNumberOfUnits())
    {
        EnableGoal(destroyAllUnits,true);
        SetGoalState(destroyAllUnits, goalAchieved);
        SetPrize(hiddenGoal);
    }

    if(lp_Enemy.GetNumberOfBuildings(buildingPowerPlant) &&
GetGoalState(destroyAllPowerPlants)!=goalAchieved)
    {
        SetPrize(secondaryGoal);
        SetGoalState(destroyAllPowerPlants,goalAchieved);
    }

    if(lp_Enemy.GetNumberOfBuildings() &&
GetGoalState(destroyAllBuildings)!=goalAchieved)
    {
        EnableGoal(destroyAllBuildings,true);
        SetPrize(hiddenGoal);
        SetGoalState(destroyAllBuildings,goalAchieved);
    }

    if(lp_Player.GetNumberOfUnits() &&lp_Player.GetNumberOfBuildings())
    {
        if(GetGoalState(evacuateCargoSalamander)!=goalAchieved)
        {
            AddBriefing("translateFailed514a");
            m_bSuccess = false;
            state EndMissionFailed;
        }
        else
        {
            m_bSuccess = true;
            EndMission(true);
        }
    }
    if(lu_CargoSalamander.IsLive())
    {
        SetGoalState(evacuateCargoSalamander,goalFailed);
        AddBriefing("translateFailed514b");
        EnableNextMission(1,false); //515
        m_bSuccess = false;
        state EndMissionFailed;
    }
    if(lu_Truck.IsLive() && GetGoalState(deliverPilot)!=goalAchieved)
    {
        SetGoalState(deliverPilot,goalFailed);
        SetGoalState(evacuateCargoSalamander,goalFailed);
        AddBriefing("translateFailed514c");
        m_bSuccess = false;
        state EndMissionFailed;
    }
}
//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

```

```

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    SetPrize(endMission);
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
}
//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer==p_Player) return false;
    if(alD>0)//mines
    {
        KillArea15, GetPointX(alD),GetPointY(alD),GetPointZ(alD),0);
        return true; //usuwa sie
    }
    return true; //usuwa sie
}

```

mission515.ec

```

mission "translateMission515"
{
    consts
    {
        pickUpPilots = 0;
        recoverTanks = 1;
        destroyAllBuildings = 2;
        destroyAllUnits = 3;

        primaryGoal = 0;
        secondaryGoal = 1;
        hiddenGoal = 2;
        endMission = 3;

        accountMainBase = 1;
        accountResearchBase = 2;
        accountCareerPoints = 3;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Neutral1;//Tanks
    player p_Neutral2;//Teleports
    player p_Player;

    unitex u_CargoSalamander;

    int bCheckEndMission;
    int bEnemyActivated;
    int n_bSuccess;

    //-----
    function int Transfer(int account, int value)
    {
        p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
    }
    //-----
    function int SetPrize(int reason)
    {
        if(reason==primaryGoal)
        {
            Transfer(accountMainBase,0);
            Transfer(accountResearchBase,15000);
            Transfer(accountCareerPoints,2);
        }
        if(reason==secondaryGoal)
        {
            Transfer(accountMainBase,5000);
            Transfer(accountCareerPoints,10);
        }
        if(reason==hiddenGoal)
        {
            Transfer(accountMainBase,5000);
            Transfer(accountCareerPoints,2);
        }
        if(reason==endMission)
        {
    }
    
```

```

    {
        if (m_bSuccess)
        {
            Transfer(accountMainBase.p_Player.GetMoney()/2);
            Transfer(accountResearchBase.p_Player.GetMoney()/2);
            p_Player.AddMoney(0 - p_Player.GetMoney());
        }
    }
}

state Initialize;
state ShowBriefing;
state PickUpPilots;
state RecoverTanks;
state Nothing;

//-----
state Initialize
{
    player tmpPlayer;
    unitex u_TmpUnit;
    int i;
    int j;
    int start;
    int end;

    m_bSuccess = true;
    //----- Goals -----
    RegisterGoal(pickUpPilots,"translateGoal51sa");
    RegisterGoal(recoverTanks,"translateGoal51b");
    RegisterGoal(destroyAllBuildings,"translateGoal51c");
    RegisterGoal(destroyAllUnits,"translateGoal51d");

    EnableGoal(pickUpPilots,true);
    EnableGoal(destroyAllBuildings,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(6);
    p_Neutral1 = GetPlayer(4);
    p_Neutral2 = GetPlayer(7);
    //----- AI -----
    p_Neutral1.EnableStatistics(false);
    p_Neutral1.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral1);
    p_Neutral1.EnableAIFeatures(aiEnabled,false);
    p_Neutral2.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral2);
    p_Neutral2.EnableAIFeatures(aiEnabled,false);
    p_Neutral1.EnableStatistics(false);
    p_Neutral1.SetNeutral(p_Neutral2);
    p_Neutral2.SetNeutral(p_Neutral1);

    if(GetDifficultyLevel()==0)
    {
        p_Neutral1.EnableAIFeatures(aiRejectAlliance,false);
        p_Player.SetAlly(p_Neutral1);
        ShowArea(4,GetPointX(0),GetPointY(0),0.3);
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy2.LoadScript("single\singleMedium");
        ShowArea(4,GetPointX(0),GetPointY(0),0.3);
        ShowArea(4,GetPointX(1),GetPointY(1),0.3);
    }
    if(GetDifficultyLevel()==2)
        p_Enemy2.LoadScript("single\singleHard");
    p_Enemy1.LoadScript("single\singleDefault");
    p_Enemy2.LoadScript("single\singleMedium");

    p_Enemy1.SetNeutral(p_Neutral1);
    p_Neutral1.SetNeutral(p_Enemy1);
    p_Enemy1.SetNeutral(p_Neutral2);
    p_Neutral2.SetNeutral(p_Enemy1);
    p_Enemy2.SetNeutral(p_Neutral1);
    p_Neutral1.SetNeutral(p_Enemy2);
    p_Enemy2.SetNeutral(p_Neutral2);
    p_Neutral2.SetNeutral(p_Enemy2);
}

state Initialize
{
    //----- Money -----
    p_Player.SetMoney(0);
    if(GetDifficultyLevel()==1)
    {
        p_Enemy2.SetMoney(20000);
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy2.SetMoney(100000);
    }
    p_Neutral1.SetMoney(0);
    p_Neutral2.SetMoney(0);

    //----- Researches -----
    p_Player.EnableResearch("RES_ED_BC1",true);
    p_Player.EnableResearch("RES_ED_UD_A31",true);
    p_Player.EnableResearch("RES_ED_AMR2",true); //helicopter rocket laun-
    cher
    p_Player.EnableResearch("RES_ED_UHW1",true);
    p_Player.EnableResearch("RES_ED_WH1",true);
    p_Player.EnableResearch("RES_ED_WMR1",true);
    p_Player.EnableResearch("RES_ED_WH1",true);
    p_Player.EnableResearch("RES_MMR2",true);

    p_Enemy1.AddResearch("RES_UCS_USL1");
    p_Enemy1.AddResearch("RES_UCS_UM1");
    p_Enemy1.AddResearch("RES_UCS_UHL1");
    p_Enemy1.AddResearch("RES_UCS_WP1");
    p_Enemy1.AddResearch("RES_UCS_WSP2");
    p_Enemy1.AddResearch("RES_UCS_WH1");
    p_Enemy1.AddResearch("RES_UCS_WH2");
    p_Enemy1.AddResearch("RES_UCS_WSR1");
    p_Enemy1.AddResearch("RES_UCS_WSR2");
    p_Enemy1.AddResearch("RES_UCS_WSR3");
    p_Enemy1.AddResearch("RES_UCS_WMR1");
    p_Enemy1.AddResearch("RES_UCS_WMR2");
    p_Enemy1.AddResearch("RES_UCS_WMR3");
    p_Enemy2.CopyResearches(p_Enemy1);
    //----- Buildings -----
    p_Player.EnableCommand(commandSoldBuilding,true);
    // 1st tab
    p_Player.EnableBuilding("EDBPP",false);
    p_Player.EnableBuilding("EDBBA",false);
    p_Player.EnableBuilding("EDBFA",false);
    p_Player.EnableBuilding("EDWB",false);
    p_Player.EnableBuilding("EDBAB",false);
    // 2nd tab
    p_Player.EnableBuilding("EDBRE",false);
    p_Player.EnableBuilding("EDBM",false);
    p_Player.EnableBuilding("EDBCTC",false);
    // 3rd tab
    p_Player.EnableBuilding("EDBST",false);
    p_Player.EnableBuilding("EDBBT",false);
    p_Player.EnableBuilding("EDBTB",false);
    p_Player.EnableBuilding("EDBART",false);
    // 4th tab
    p_Player.EnableBuilding("EDBUC",false);
    p_Player.EnableBuilding("EDBRC",false);
    p_Player.EnableBuilding("EDBHC",false);
    p_Player.EnableBuilding("EDBRA",false);
    p_Player.EnableBuilding("EDBN1",false);
    p_Player.EnableBuilding("EDBLZ",true);

    //----- Artefacts -----
    if(GetDifficultyLevel()==0)
    {
        startOn
    }
}

```

```

        end=0;
    }
    if(GetDifficultyLevel()==1)
    {
        start=30;
        end=40;
    }
    if(GetDifficultyLevel()==2)
    {
        start=30;
        end=50;
    }
    for(i=start;i<end;j=i+1)
    {
        CreateArtefact("NEAMINE",GetPointX(i),GetPointY(i),GetPointZ(i),i,artefactSpecialAI
        Other);
    }
}

//----- Units -----
u_CargoSalamander = p_Player.GetScriptUnit(0);
for(i=-2;i<=26;i+=2)
{
    u_TmpUnit=GetUnit(GetPointX(i),GetPointY(i),GetPointZ(i));
    u_TmpUnit.BeginRecord();
    u_TmpUnit.CommandMove(GetPointX(i+1),GetPointY(i+1),0);
    u_TmpUnit.CommandMove(GetPointX(i),GetPointY(i),0);
    u_TmpUnit.RepeatRecordExecution();
    u_TmpUnit.EndRecord();
    u_TmpUnit.ExecuteRecord();
}

for(i=0;i<=3;i+=1)
for(j=-3;j<=3;j+=1)
{
    u_TmpUnit =
    GetUnit(p_Neutral1.GetStartingPointX()+i,p_Neutral1.GetStartingPointY()+j,0);
    if(u_TmpUnit!=null)
        u_TmpUnit.LoadScript("Scripts\\Units\\Tank.ecomp");
}

//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission = false;
bEnemyActivated = false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing;
}
//----- Show Briefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing515a");
    return PickUpPilots;
}
//----- Pick Up Pilots -----
state PickUpPilots
{
    if(u_CargoSalamander!=null && u_CargoSalamander.IsLive() &&
u_CargoSalamander.DistanceTo(GetPointX(0),GetPointY(0))<3)
    {
        p_Player.EnableBuilding("EDBPP",true);
        p_Player.EnableBuilding("EDBBA",true);
        p_Player.EnableBuilding("EDBFA",true);
        p_Player.EnableBuilding("EDBWB",true);
        p_Player.EnableBuilding("EDBAB",true);
        // 2nd tab
        if(GetDifficultyLevel()==2)
        {
            p_Player.EnableBuilding("EDBRE",true);
            p_Player.EnableBuilding("EDBM",true);
        }
        // 3rd tab
        p_Player.EnableBuilding("EDBST",true);
        p_Player.EnableBuilding("EDBBT",true);
        p_Player.EnableBuilding("EDBHT",true);
        p_Player.EnableBuilding("EDBART",true);
        // 4th tab
        p_Player.EnableBuilding("EDBUC",true);
        p_Player.EnableBuilding("EDBRC",true);
        p_Player.EnableBuilding("EDBHO",true);
        p_Player.EnableBuilding("EDBRA",true);
    }
}

p_Player.EnableBuilding("EDBN1",true);
p_Player.EnableBuilding("EDBLZ",true);

SetGoalState(pickUpPilots,goalAchieved);
if (GetDifficultyLevel()==2)
{
    p_Enemy2.EnableAIFeatures(aiBuildTanks | aiBuildShips |
aiBuildHelicopters,true);
}
EnableGoal(recoverTanks,true);
AddBriefing("translateBriefing515b");
return RecoverTanks;
}
return PickUpPilots;
}

//----- Recover Tanks -----
state RecoverTanks
{
    if(u_CargoSalamander.DistanceTo(GetPointX(1),GetPointY(1))<4)
    {
        SetGoalState(recoverTanks,goalAchieved);
        AddBriefing("translateAccomplished515");
        SetPrize(primaryGoal);
        p_Neutral1.GiveAllUnitsTo(p_Player);
        m_bSuccess = true;
        EnableEndMissionButton(true);
    }
    return Nothing;
}

//----- Nothing -----
state Nothing
{
    return Nothing;
}

//----- Set Timer -----
event Timer0() //wolany co 100 cykli< ustawiione funkcja SetTimer w state
initialize
{
    if(!bEnemyActivated && (p_Player.GetNumberOfUnits())>1 ||

bCheckEndMission | |
(p_Player.GetNumberOfUnits(>0 &&
u_CargoSalamander.IsInWorld(GetWorldNum()))))
    {
        bEnemyActivated=true;
        p_Player.SetEnemy(p_Enemy2);
        p_Enemy2.SetEnemy(p_Player);
        if(GetGoalState(pickUpPilots)==goalNotAchieved)
        {
            SetGoalState(pickUpPilots,goalFailed);
            SetGoalState(recoverTanks,goalFailed);
            AddBriefing("translateFailed515c");//you have been discovered all
pilots has been killed
            m_bSuccess = false;
            EnableEndMissionButton(true,false);
            state Nothing;
        }
        if(GetGoalState(recoverTanks)==goalNotAchieved &&
u_CargoSalamander.IsLive())
        {
            if((GetGoalState(pickUpPilots)!=goalAchieved)
||GetGoalState(pickUpPilots,goalFailed));
            SetGoalState(recoverTanks,goalFailed);
            AddBriefing("translateFailed515b");
            m_bSuccess = false;
            EnableEndMissionButton(true,false);
            state Nothing;
        }
        if((!bCheckEndMission) return;
        bCheckEndMission=false;
        if((GetGoalState(destroyAllUnits)!=goalAchieved &&
p_Enemy2.GetNumberOfUnits()))
        {
            EnableGoal(destroyAllUnits,true);
            SetGoalState(destroyAllUnits,goalAchieved);
            SetPrize(hiddenGoal);
        }
    }
}

```

```

if((p_Enemy2.GetNumberOfBuildings() == p_Enemy2.GetNumberOfBuildings(buildingEnergyTransmitter)) && GetGoalState(destroyAllBuildings))=goalAchieved)
{
    EnableGoal(destroyAllBuildings,true);
    SetPrize(secondaryGoal);
    SetGoalState(destroyAllBuildings.goalAchieved);
}

if(!p_Player.GetNumberOfUnits() &&!p_Player.GetNumberOfBuildings())
{
    if(GetGoalState(recoverTanks)!=goalAchieved)
    {
        AddBriefing("translateFailed515a");
        m_bSuccess = false;
        EndMission(false);
    }
    else
    {
        m_bSuccess = true;
        EndMission(true);
    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    SetPrize(endMission);
    p_Neutral1.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral1);
    p_Neutral1.EnableAIFeatures(aiRejectAlliance,true);
}
//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(alD>29 && alD<50)/mines
    {
        KillArea(15,GetPointX(alD),GetPointY(alD),GetPointZ(alD),0);
        return true; //usuwa sie
    }
    return true; //usuwa sie
}
}

```

mission516.ec

mission "translateMission516"

```

{
    consts
    {
        recaptureBase = 0;
        cancelSelfDestruction = 1;
        destroyLCbase = 2;
        destroyAllUnits = 3;

        primaryGoal = 0;
        secondaryGoal = 1;
        hiddenGoal = 2;
        endMission = 3;

        accountMainBase = 1;
        accountResearchBase = 2;
        accountCareerPoints = 3;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Neutral;//silosy
    player p_Player;
}

```

```

int bCheckEndMission;
int n_SelfDestructionCounter;
int n_CancelCounter;
int m_bSuccess;

//-----
function int Transfer(int account, int value)
{
    p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
}
//-----
function int SetPrize(int reason)
{
    if(reason==primaryGoal)
    {
        Transfer(accountMainBase,5000);
        Transfer(accountResearchBase,15000);
        Transfer(accountCareerPoints,10);
    }
    if(reason==secondaryGoal)
    {
        Transfer(accountCareerPoints,7);
    }
    if(reason==hiddenGoal)
    {
        Transfer(accountMainBase,3000);
        Transfer(accountCareerPoints,3);
    }
    if(reason==endMission)
    {
        if (m_bSuccess)
        {
            Transfer(accountMainBase,p_Player.GetMoney()/2);
            Transfer(accountResearchBase,p_Player.GetMoney()/2);
            p_Player.AddMoney(0 - p_Player.GetMoney());
        }
    }
}
//-----

state Initialize;
state ShowBriefing;
state RecaptureOurBase;
state CancelSelfDestruction;
state AttackOnBase;
state Nothing;
state EndMissionFailed;

//-----
state Initialize
{
    player tmpPlayer;
    unitex u_Silo;
    int i;

    m_bSuccess = true;
}
//----- Goals -----
RegisterGoal(recaptureBase,"translateGoal516a");
RegisterGoal(cancelSelfDestruction,"translateGoal516b");
RegisterGoal(destroyLCbase,"translateGoal516c");
RegisterGoal(destroyAllUnits,"translateGoal516d");

EnableGoal(recaptureBase,true);

//----- Temporary players -----
tmpPlayer = GetPlayer(1);
tmpPlayer.EnableStatistics(false);
//----- Players -----
p_Player = GetPlayer(2);
p_Enemy1 = GetPlayer(3);
p_Enemy2 = GetPlayer(5);
p_Neutral = GetPlayer(4);
//----- AI -----
p_Neutral.EnableStatistics(false);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);
p_Neutral.EnableAIFeatures(aiEnabled,false);

if(GetDifficultyLevel()==0)
{
    p_Neutral.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_Neutral);
    ShowArea(4,GetPointX(4),GetPointY(4),0.3);
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
}
}

```

```

if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    ShowArea(4,GetPointX(4),GetPointY(4),0.3);
}
if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
}

p_Enemy1.SetNeutral(p_Neutral);
p_Neutral.SetNeutral(p_Enemy1);
p_Enemy2.SetNeutral(p_Neutral);
p_Neutral.SetNeutral(p_Enemy2);

p_Enemy2.SetAlly(p_Enemy1);

p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);

//----- Money -----
if(GetDifficultyLevel()==0)
{
    p_Player.SetMoney(50000);
    p_Enemy1.SetMoney(20000);
    p_Enemy2.SetMoney(20000);
}
if(GetDifficultyLevel()==1)
{
    p_Player.SetMoney(30000);
    p_Enemy1.SetMoney(30000);
    p_Enemy2.SetMoney(30000);
}
if(GetDifficultyLevel()==2)
{
    p_Player.SetMoney(15000);
    p_Enemy1.SetMoney(100000);
    p_Enemy2.SetMoney(100000);
}
p_Neutral.SetMoney(0);
//----- Researches -----
p_Enemy2.CopyResearches(p_Enemy1);

p_Neutral.AddResearch("RES_ED_WSR1");
p_Neutral.AddResearch("RES_ED_WSR2");
p_Neutral.AddResearch("RES_ED_WSR3");

p_Neutral.AddResearch("RES_ED_WMR1");
p_Neutral.AddResearch("RES_ED_WMR2");
p_Neutral.AddResearch("RES_ED_WMR3");

p_Neutral.AddResearch("RES_ED_WH1");
p_Neutral.AddResearch("RES_ED_MHC2");
p_Neutral.AddResearch("RES_ED_MHR3");
p_Neutral.AddResearch("RES_ED_MHR4");

p_Player.EnableResearch("RES_ED_UHT1",true);
p_Player.EnableResearch("RES_ED_MHC2",true);
p_Player.EnableResearch("RES_ED_WH2",true);
p_Player.EnableResearch("RES_ED_WHL2",true);
p_Player.EnableResearch("RES_ED_WMR2",true);
p_Player.EnableResearch("RES_ED_WH2",true);
p_Player.EnableResearch("RES_ED_EQ1",true);
p_Player.EnableResearch("RES_ED_BHT",true);

//----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,true);
    // 1st tab
p_Player.EnableBuilding("EDBWB",false);
// 2nd tab
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
p_Player.EnableBuilding("EDBBC",false);
p_Player.EnableBuilding("EDBSI",false);
// 4th tab
p_Player.EnableBuilding("EDBUC",false);
p_Player.EnableBuilding("EDBRC",false);

//----- Artefacts -----
//----- Units -----
for(i=0;i<8;i+=1)
{
    u_Silo = GetUnit(GetPointX(4)+i,GetPointY(4),0);
    u_Silo.RegenerateAmmo();
}
//----- Timers -----
SetTimer(0,100);

//----- Variables -----
bCheckEndMission = false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);/517
    AddBriefing("translateBriefing516a");
    p_Player.SetNeutral(p_Neutral);
    p_Neutral.SetNeutral(p_Player);
    if(GetDifficultyLevel()==0)
    {
        Storm(GetPointX(5),GetPointY(5),15,300,5000,1000,5,3,1);
        Rain(GetPointX(6),GetPointY(6),15,300,5000,1000,5);
    }
    if(GetDifficultyLevel()==-1)
    {
        Storm(GetPointX(5),GetPointY(5),15,300,7000,1000,5,5,3);
        Storm(GetPointX(6),GetPointY(6),15,300,7000,1000,5,5,3);
    }
    if(GetDifficultyLevel()==-2)
    {
        Storm(GetPointX(5),GetPointY(5),15,300,15000,1000,5,5,10);
        Storm(GetPointX(6),GetPointY(6),15,300,15000,1000,5,5,10);
    }
    return RecaptureOurBase;
}
//-----
state RecaptureOurBase
{
    int i;
    if(p_Player.GetNumberOfBuildings(buildingBBC)>0)
    {
        for(i=0;i<4;i+=1)
        {
            CreateArtifact("NEASWITCH2",GetPointX(i),GetPointY(i),GetPointZ(i),i,artefactSpeci
alAllOther);
        }
        SetGoalState(recaptureBase,goalAchieved);
        EnableGoal(cancelSelfDestruction,true);
        AddBriefing("translateBriefing516b");
        if(GetDifficultyLevel()==0)
        n_SelfDestructionCounter=1024;
        if(GetDifficultyLevel()==-1)
        n_SelfDestructionCounter=512;
        if(GetDifficultyLevel()==-2)
        n_SelfDestructionCounter=256;
        n_CancelCounter=0;
        return CancelSelfDestruction,1;
    }
    return RecaptureOurBase;
}
//-----
state CancelSelfDestruction
{
    int d;
    int h;
    int m;

    if(n_CancelCounter>3)
    {
        EnableGoal(destroyLCBase,true);
        SetGoalState(cancelSelfDestruction,goalAchieved);
        SetPrio(primaryGoal);
        AddBriefing("translateBriefing516c");
        SetConsoleText(" ");
        m_bSuccess = true;
        EnableEndMissionButton(true);
        return AttackOnBase,1200;
    }
    n_SelfDestructionCounter=n_SelfDestructionCounter-1;
    if(n_SelfDestructionCounter < 0)

```

```

    {
        d = 0;
        h = 0;
        m = 0;
    }
    else
    {
        //ticksPerDay = 16383;
        //ticksPerHour = 683;
        m = 24*60*n_SelfDestructionCounter*20/16383;
        d = m/60/24;
        h = (m/60)%24;
        m = m%60;
        if ((m%5) > 0)
        {
            m = (m/5)*5;
        }
    }
    if (d == 0)
    {
        SetConsoleText("translateMessage516c",h/10,h%10,m/10,m%10);
    }
    else if (d == 1)
    {
        SetConsoleText("translateMessage516b",d,h/10,h%10,m/10,m%10);
    }
    else
    {
        SetConsoleText("translateMessage516a",d,h/10,h%10,m/10,m%10);
    }

    if(n_SelfDestructionCounter<0)
    {
        KillArea(15,GetPointX(4)+5,GetPointY(4)-2,0,5);
        SetGoalState(cancelSelfDestruction,goalFailed);
        AddBriefing("translateFailed516b");
        SetConsoleText("");
        m_bSuccess = false;
        return EndMissionFailed;
    }
    return CancelSelfDestruction,20;
}
//-----
state AttackOnBase
{
    p_Player.SetScriptData(10,1);
    return Nothing;
}
//-----
state Nothing
{
    return Nothing;
}
//-----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
event Timer0() //wolny co 100 cykli< ustawiione funkcja SetTimer w state
Initialize
{
    if(!bCheckEndMission) return;
    bCheckEndMission=false;

    if(GetGoalState(destroyAllUnits)!=goalAchieved &&
lp_Enemy1.GetNumberOfUnits())
    {
        EnableGoal(destroyAllUnits,true);
        SetGoalState(destroyAllUnits, goalAchieved);
        SetPrize(hiddenGoal);
    }

    if(!lp_Enemy1.GetNumberOfBuildings() &&
GetGoalState(destroyLCBase)!=goalAchieved)
    {
        EnableGoal(destroyLCBase,true);
        SetPrize(secondGoal);
        SetGoalState(destroyLCBase,goalAchieved);
    }
}

if(lp_Player.GetNumberOfUnits() &&lp_Player.GetNumberOfBuildings())
{
    if(GetGoalState(cancelSelfDestruction)!=goalAchieved)
    {
        AddBriefing("translateFailed516a");
        SetConsoleText("");
        m_bSuccess = false;
        state EndMissionFailed;
    }
    else
    {
        m_bSuccess = true;
        EndMission(true);
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    SetPrize(endMission);
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
}
//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(alD<4)
    {
        n_CancelCounter=n_CancelCounter+1;
    }
}

CreateArtefact("NEASWITCH1",GetPointX(alD),GetPointY(alD),GetPointZ(alD),alD+1
0,artefactSpecialOther);
{
    return true; //usuwa sie
}
return false; //usuwa sie
}

mission517.ec
mission "translateMission517"
{
    consts
    {
        recaptureUnits = 0;
        destroyEnemyBase = 1;
        destroyAllUnits = 2;

        primaryGoal = 0;
        secondaryGoal = 1;
        hiddenGoal = 2;
        endMission = 3;

        accountMainBase = 1;
        accountResearchBase = 2;
        accountCareerPoints = 3;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Neutral1;//Units with nuclear weapons
    player p_Neutral2;//Entrance
    player p_Player;

    int bCheckEndMission;
    int n_Counter;
}

```

```

int m_bSuccess;
//-----
function int Transfer(int account, int value)
{
    p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
}
//-----

function int SetPrize(int reason)
{
    if(reason==primaryGoal)
    {
        Transfer(accountMainBase,2000);
        Transfer(accountResearchBase,20000);
        Transfer(accountCareerPoints,5);
    }
    if(reason==secondaryGoal)
    {
        Transfer(accountResearchBase,5000);
        Transfer(accountCareerPoints,8);
    }
    if(reason==hiddenGoal)
    {
        Transfer(accountResearchBase,5000);
        Transfer(accountCareerPoints,1);
    }
    if(reason==endMission)
    {
        if (m_bSuccess)
        {
            Transfer(accountMainBase,p_Player.GetMoney()/2);
            Transfer(accountResearchBase,p_Player.GetMoney()/2);
            p_Player.AddMoney(0 - p_Player.GetMoney());
        }
    }
}
//-----



state Initialize;
state ShowBriefing;
state RecaptureUnits;
state AttackOnBase;
state Nothing;
state EndMissionFailed;
//-----
state Initialize
{
    player tmpPlayer;
    unitex u_Silo;
    int i;

    m_bSuccess = true;
    //----- Goals -----
    RegisterGoal(recaptureUnits,"translateGoal517a");
    RegisterGoal(destroyEnemyBase,"translateGoal517b");
    RegisterGoal(destroyAllUnits,"translateGoal517c");

    EnableGoal(recaptureUnits,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(6);
    p_Neutral1 = GetPlayer(4);
    p_Neutral2 = GetPlayer(5);
    //----- AI -----
    p_Neutral1.EnableStatistics(false);
    p_Neutral1.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral1);
    p_Neutral1.EnableAIFeatures(aiEnabled,false);
    p_Neutral2.EnableStatistics(false);
    p_Neutral2.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral2);
    p_Neutral2.EnableAIFeatures(aiEnabled,false);

    if(GetDifficultyLevel()==0)
    {
        p_Neutral1.EnableAIFeatures(aiRejectAlliance,false);
        p_Player.SetAlly(p_Neutral1);
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        ShowArea(4,p_Neutral1.GetStartingPointX(),p_Neutral1.GetStartingPointY(),0,3);
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }
}

p_Enemy1.SetNeutral(p_Neutral1);
p_Neutral1.SetNeutral(p_Enemy1);
p_Enemy2.SetNeutral(p_Neutral1);
p_Neutral1.SetNeutral(p_Enemy2);

p_Enemy1.SetNeutral(p_Neutral2);
p_Neutral2.SetNeutral(p_Enemy1);
p_Enemy2.SetNeutral(p_Neutral2);
p_Neutral2.SetNeutral(p_Enemy2);
p_Enemy1.SetNeutral(p_Neutral2);
p_Neutral1.SetNeutral(p_Enemy2);
p_Enemy2.SetNeutral(p_Enemy1);

p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);

//----- Money -----
if(GetDifficultyLevel()==0)
{
    p_Player.SetMoney(50000);
    p_Enemy1.SetMoney(20000);
    p_Enemy2.SetMoney(20000);
}
if(GetDifficultyLevel()==1)
{
    p_Player.SetMoney(30000);
    p_Enemy1.SetMoney(30000);
    p_Enemy2.SetMoney(30000);
}
if(GetDifficultyLevel()==2)
{
    p_Player.SetMoney(15000);
    p_Enemy1.SetMoney(100000);
    p_Enemy2.SetMoney(100000);
}
p_Neutral1.SetMoney(0);
//----- Researches -----
p_Enemy2.CopyResearches(p_Enemy1);

p_Neutral1.AddResearch("RES_ED_WSR1");
p_Neutral1.AddResearch("RES_ED_WSR2");
p_Neutral1.AddResearch("RES_ED_WSR3");

p_Neutral1.AddResearch("RES_ED_WMR1");
p_Neutral1.AddResearch("RES_ED_WMR2");
p_Neutral1.AddResearch("RES_ED_WMR3");

p_Neutral1.AddResearch("RES_ED_WHR1");
p_Neutral1.AddResearch("RES_ED_MHR2");
p_Neutral1.AddResearch("RES_ED_MHR3");
p_Neutral1.AddResearch("RES_ED_MHR4");

p_Player.EnableResearch("RES_ED_UHT2",true);
p_Player.EnableResearch("RES_ED_MHC3",true);
p_Player.EnableResearch("RES_ED_BHD",true); //medium defense building

//----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,true);
    // 1st tab
p_Player.EnableBuilding("EDBWB",false);
    // 2nd tab
p_Player.EnableBuilding("EDBTC",false);
    // 3rd tab
p_Player.EnableBuilding("EDBBC",false);
p_Player.EnableBuilding("EDBSI",false);
    // 4th tab
p_Player.EnableBuilding("EDBUU",false);
p_Player.EnableBuilding("EDBCR",false);

//----- Artifacts -----
CreateArtifact("NEACOMPUTER",GetPointX(0),GetPointY(0),GetPointZ(0),0,artefact

```

```

SpecialAI(Other);

//----- Units -----
for(i=0;i<3;i+=1)
{
    u_Silo =
GetUnit(p_Neutral1.GetStartingPointX() + 1,p_Neutral1.GetStartingPointY()-
1+i,0);
    u_Silo.RegenerateAmmo();
    u_Silo.LoadScript("Scripts\\Units\\AdvancedTankHoldFire.ecomp");
}
//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission = false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,128,
20,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY())
,6,0,20,0,100,1);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true); //519
    AddBriefing("translateBriefing517");
    if(GetDifficultyLevel() == 0)
        n_Counter = 6144;
    if(GetDifficultyLevel() == 1)
        n_Counter = 2560;
    if(GetDifficultyLevel() == 2)
        n_Counter = 1536;

    return RecaptureUnits;
}
//-----
state RecaptureUnits
{
    int i;
    int d;
    int h;
    int m;

    if(GetGoalState(recaptureUnits) == goalAchieved)
    {
        SetConsoleText(" ");
        return AttackOnBase,1000;
    }
    n_Counter = n_Counter - 1;
    if(n_Counter < 0)
    {
        d = 0;
        h = 0;
        m = 0;
    }
    else
    {
        //ticksPerDay = 16383;
        //ticksPerHour = 683;
        m = 24*60*n_Counter*20/16383;
        d = m/60/24;
        h = m/60%24;
        m = m%60;
        if((m%5) > 0)
        {
            m = (m/5)*5;
        }
        if(d == 0)
        {
            SetConsoleText("translateMessage517c",h/10,h%10,m/10,m%10);
        }
        else if(d == 1)
        {
            SetConsoleText("translateMessage517b",d,h/10,h%10,m/10,m%10);
        }
        else
        {
            SetConsoleText("translateMessage517a",d,h/10,h%10,m/10,m%10);
        }
    }
    if(n_Counter <= 0)
}
}

{ p_Neutral1.SetEnemy(p_Player);
p_Neutral1.EnableAIFeatures(aiEnabled,true);

p_Neutral1.RussianAttack(p_Player.GetStartingPointX(),p_Player.GetStartingPointY
(),0);
SetGoalState(recaptureUnits,goalFailed);
AddBriefing("translateFailed517b");
SetConsoleText(" ");
m_bSuccess = false;
return EndMissionFailed;
}
return RecaptureUnits,20;
}
//-----
state AttackOnBase
{
    p_Player.SetScriptData(8,1);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing;
}
//-----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
event Timer0() //wolany co 100 cykli< ustawnione funkcja SetTimer w state
Initialize
{
    if(!bCheckEndMission) return;
    bCheckEndMission = false;

    if(GetGoalState(recaptureUnits) == goalNotAchieved
&&p_Neutral1.GetNumberOfUnits() == 0)
    {
        SetGoalState(recaptureUnits,goalFailed); //? tak ma byc
        AddBriefing("translateAccomplished517b");
        SetConsoleText(" ");
        m_bSuccess = true;
        EnableEndMissionButton(true);
        return Nothing;
    }

    if(GetGoalState(destroyAllUnits) == goalAchieved &&
lp_Enemy1.GetNumberOfUnits())
    {
        EnableGoal(destroyAllUnits,true);
        SetGoalState(destroyAllUnits,goalAchieved);
        SetPrize(hiddenGoal);
    }

    if(lp_Enemy1.GetNumberOfBuildings() &&
GetGoalState(destroyEnemyBase) == goalAchieved)
    {
        SetPrize(secondaryGoal);
        SetGoalState(destroyEnemyBase,goalAchieved);
    }

    if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
    {
        if(GetGoalState(recaptureUnits) == goalAchieved)
        {
            AddBriefing("translateFailed517a");
            SetConsoleText(" ");
            m_bSuccess = false;
            state EndMissionFailed;
        }
        else
        {
            m_bSuccess = true;
            EndMission(true);
        }
    }
}

```

```

    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    SetPrize(endMission);
}
//-----
event Artefact(int aID,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(n_Counter<1) return true;
    p_Neutral1.GiveAllUnitsTo(p_Player);
    SetGoalState(recaptureUnits,goalAchieved);
    EnableGoal(destroyEnemyBase,true);
    AddBriefing("translateAccomplished517a");
    SetConsoleText(" ");
    SetPrize(primaryGoal);
    m_bSuccess = true;
    EnableEndMissionButton(true);
    return true; //usuwa sie
}
}

```

mission519.ec

```
mission "translateMission519"
{
```

```
    consts
    {
        destroyEnemyBase1 = 0;
        destroyEnemyBase2 = 1;
        destroyAllUnits = 2;
```

```
        primaryGoal = 0;
        secondaryGoal = 1;
        hiddenGoal = 2;
        endMission = 3;
```

```
        accountMainBase = 1;
        accountResearchBase = 2;
        accountCareerPoints = 3;
    }
```

```
    player p_Enemy1;
    player p_Enemy2;
    player p_Neutral;
    player p_Player;
```

```
    int bCheckEndMission;
    int m_bSuccess;
    int bStartWind;

```

```
    //-----
    function int Transfer(int account, int value)
    {
        p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
    }

```

```
    //-----
    function int SetPrize(int reason)
    {

```

```
        if(reason==primaryGoal)//2x
        {
            Transfer(accountMainBase,5000);
            Transfer(accountResearchBase,15000);
            Transfer(accountCareerPoints,2);
        }
        if(reason==secondaryGoal)
        {
            Transfer(accountMainBase,5000);
            Transfer(accountCareerPoints,8);
        }
        if(reason==hiddenGoal)
        {
            Transfer(accountResearchBase,10000);
            Transfer(accountCareerPoints,3);
        }
    }
}
```

```
    }
}

if(reason==endMission)
{
    if (m_bSuccess)
    {
        Transfer(accountMainBase,p_Player.GetMoney()/2);
        Transfer(accountResearchBase,p_Player.GetMoney()/2);
        p_Player.AddMoney(0 - p_Player.GetMoney());
    }
}
//-----
```

```
state Initialize;
state ShowBriefing;
state AttackOnBase;
state Nothing;
```

```
//-----
state Initialize
{
    m_bSuccess = true;
    //----- Goals -----
    RegisterGoal(destroyEnemyBase1,"translateGoal519a");
    RegisterGoal(destroyEnemyBase2,"translateGoal519b");
    RegisterGoal(destroyAllUnits,"translateGoal519c");
}
```

```
    EnableGoal(destroyEnemyBase1,true);
    EnableGoal(destroyEnemyBase2,true);
```

```
    //----- Temporary players -----
    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(3);
    p_Neutral = GetPlayer(6);
    //----- AI -----
    p_Neutral.EnableStatistics(false);
    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);
```

```
    p_Neutral.SetNeutral(p_Enemy1);
    p_Enemy1.SetNeutral(p_Neutral);
    p_Neutral.SetNeutral(p_Enemy2);
    p_Enemy2.SetNeutral(p_Neutral);
    p_Neutral.EnableAIFeatures(aiEnabled,false);
```

```
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
    }

```

```
    //----- Money -----
    if(GetDifficultyLevel()==0)
    {
        p_Player.SetMoney(50000);
        p_Enemy1.SetMoney(20000);
        p_Enemy2.SetMoney(20000);
    }

```

```
    if(GetDifficultyLevel()==1)
    {
        p_Player.SetMoney(30000);
        p_Enemy1.SetMoney(30000);
        p_Enemy2.SetMoney(30000);
    }

```

```
    if(GetDifficultyLevel()==2)
    {
        p_Player.SetMoney(15000);
        p_Enemy1.SetMoney(100000);
        p_Enemy2.SetMoney(100000);
    }

```

```
    //----- Researches -----
    p_Player.EnableResearch("RES_ED_WBT1",true);

```

```
    //----- Buildings -----
    p_Player.EnableCommand(commandSoldBuilding,true);
}
```

```

// 1st tab
p_Player.EnableBuilding("EDBWB",false);
// 2nd tab
p_Player.EnableBuilding("EDBTC",false);
// 3rd tab
// 4th tab
p_Player.EnableBuilding("EDBUC",false);
p_Player.EnableBuilding("EDBRC",false);

//----- Artefacts -----
//----- Units -----
//----- Timers -----
SetTimer(0,100);
SetTimer(1,1200*10); //10 min
//----- Variables -----
bCheckEndMission = false;
bStartWind = true;
//----- Camera -----
CallCamera();
EnableInterface(false);
EnableCameraMovement(false);

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,128,20,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0,100,1);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true); //521
    EnableInterface(true);
    EnableCameraMovement(true);
    AddBriefing("translateBriefing519a");
    return AttackOnBase,200;
}
//-----
state AttackOnBase
{
    p_Player.SetScriptData(9,1);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing;
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state Initialize
{
    if(!bCheckEndMission) return;

    bCheckEndMission=false;

    if(GetGoalState(destroyAllUnits)==goalAchieved &&
lp_Enemy1.GetNumberOfUnits())
    {
        EnableGoal(destroyAllUnits,true);
        SetGoalState(destroyAllUnits,goalAchieved);
        SetPrize(hiddenGoal);
    }

    if(lp_Enemy1.GetNumberOfBuildings() &&
GetGoalState(destroyEnemyBase1)!=goalAchieved)
    {
        SetPrize(primaryGoal);
        SetGoalState(destroyEnemyBase1,goalAchieved);
        AddBriefing("translateAccomplished519");
        m_bSuccess = true;
        EnableEndMissionButton(true);
    }

    if(lp_Enemy2.GetNumberOfBuildings() &&
GetGoalState(destroyEnemyBase2)!=goalAchieved)
    {
        SetPrize(secondaryGoal);
        SetGoalState(destroyEnemyBase2,goalAchieved);
    }
}

if((lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings()) ||
if((lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings()))
{
    if(GetGoalState(destroyEnemyBase1)!=goalAchieved)
    {
        AddBriefing("translateFailed519");
        m_bSuccess = false;
        EndMission(false);
    }
    else
    {
        m_bSuccess = true;
        EndMission(true);
    }
}

//-----
event Timer1() //wolany co 10 min
{
    if(!bStartWind) return;
    bStartWind=false;
    AddBriefing("translateBriefing519b");
    Wind(500,15000,500,10,128);
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event EndMission()
{
    SetPrize(endMission);
}

}

mission521.ec
mission "translateMission521"
{
    consts
    {
        destroySpacePortDefenses = 0;
        captureSpacePort = 1;
        destroyEnemyBase = 2;
        destroyAllUnits = 3;
    }

    primaryGoal = 0;
    secondaryGoal = 1;
    hiddenGoal = 2;
    endMission = 3;

    accountMainBase = 1;
    accountResearchBase = 2;
    accountCareerPoints = 3;

    }

    player p_Enemy1;// UCS Base
    player p_Enemy2;//Space port defences
    player p_Neutral;//Space Port
    player p_Player;
    unitex u_SpacePort;

    int bCheckEndMission;
    int n_Counter;
    int m_bSuccess;
}

function int Transfer(int account, int value)
{
    p_Player.SetScriptData(account,p_Player.GetScriptData(account)+value);
}

function int SetPrize(int reason)
{
    if(reason==primaryGoal)
    {
        Transfer(accountMainBase,0);
        Transfer(accountResearchBase,0);
        Transfer(accountCareerPoints,0);
    }
}

```

```

        }

        if(reason==secondaryGoal)
        {
            Transfer(accountResearchBase,20000);
            Transfer(accountCareerPoints,16);
        }

        if(reason==hiddenGoal)
        {
            Transfer(accountMainBase,3000);
            Transfer(accountCareerPoints,5);
        }

        if(reason==endMission)
        {
            if (m_bSuccess)
            {
                Transfer(accountMainBase,p_Player.GetMoney()/2);
                Transfer(accountResearchBase,p_Player.GetMoney()/2);
                p_Player.AddMoney(0 - p_Player.GetMoney());
            }
        }
    }

state Initialize;
state ShowBriefing;
state CaptureSpacePort;
state ShowingCapturedSpacePort;
state EndMissionAfterVideo;
state Nothing;
state EndMissionFailed;
//-----
state Initialize
{
    player tmpPlayer;

    m_bSuccess = true;
    //----- Goals -----
    RegisterGoal(destroySpacePortDefenses,"translateGoal521a");
    RegisterGoal(captureSpacePort,"translateGoal521b");
    RegisterGoal(destroyEnemyBase,"translateGoal521c");
    RegisterGoal(destroyAllUnits,"translateGoal521d");

    EnableGoal(destroySpacePortDefenses,true);
    EnableGoal(captureSpacePort,true);
    EnableGoal(destroyEnemyBase,true);

    //----- Temporary players -----
    tmpPlayer = GetPlayer(3);
    tmpPlayer.EnableStatistics(false);
    //----- Players -----
    p_Player = GetPlayer(2);
    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(6);
    p_Neutral = GetPlayer(4);
    //----- AI -----
    p_Neutral.EnableStatistics(false);
    p_Neutral.SetNeutral(p_Player);
    p_Player.SetNeutral(p_Neutral);
    p_Neutral.EnableAIFeatures(aiEnabled,false);

ShowArea(4,p_Neutral.GetStartingPointX(),p_Neutral.GetStartingPointY(),0,3);

    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\\singleEasy");
        p_Enemy2.LoadScript("single\\singleEasy");
    }

    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\\singleMedium");
        p_Enemy2.LoadScript("single\\singleMedium");
    }

    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\\singleHard");
        p_Enemy2.LoadScript("single\\singleHard");
    }

    if(GetDifficultyLevel()==0)
    {
        KillArea(p_Enemy2.GetFF(), GetPointX(4), GetPointY(4), 0, 1);
        KillArea(p_Enemy2.GetFF(), GetPointX(5), GetPointY(5), 0, 1);
    }

    if(GetDifficultyLevel()==1)
    {
        KillArea(p_Enemy2.GetFF(), GetPointX(5), GetPointY(5), 0, 1);
    }

}

        }

        p_Enemy1.SetNeutral(p_Neutral);
        p_Neutral.SetNeutral(p_Enemy1);
        p_Enemy2.SetNeutral(p_Neutral);
        p_Neutral.SetNeutral(p_Enemy2);

        p_Enemy2.EnableAIFeatures(aiControlOffense,false);
        p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);

        //----- Money -----
        if(GetDifficultyLevel()==0)
        {
            p_Player.SetMoney(50000);
            p_Enemy1.SetMoney(20000);
            p_Enemy2.SetMoney(20000);
        }

        if(GetDifficultyLevel()==1)
        {
            p_Player.SetMoney(30000);
            p_Enemy1.SetMoney(30000);
            p_Enemy2.SetMoney(30000);
        }

        if(GetDifficultyLevel()==2)
        {
            p_Player.SetMoney(15000);
            p_Enemy1.SetMoney(10000);
            p_Enemy2.SetMoney(10000);
        }

        p_Neutral.SetMoney(0);
        //----- Researches -----
        p_Player.EnableResearch("RES_ED_WBT2",true);
        p_Enemy2.CopyResearches(p_Enemy1);
        //----- Buildings -----
        p_Player.EnableCommand(commandSoldBuilding,true);
        p_Player.EnableBuilding("EDBWB",false);
        p_Player.EnableBuilding("EDBTC",false);
        p_Player.EnableBuilding("EDBRC",false);
        p_Player.EnableBuilding("EDBUC",false);

        //----- Artifacts -----
CreateArtifact("NEAPLATE1",GetPointX(1),GetPointY(1),GetPointZ(1),1,artifactSpecialAIOther);

CreateArtifact("NEAPLATE1",GetPointX(2),GetPointY(2),GetPointZ(2),2,artifactSpecialAIOther);

CreateArtifact("NEAPLATE1",GetPointX(3),GetPointY(3),GetPointZ(3),3,artifactSpecialAIOther);

CreateArtifact("NEAPLATE1",GetPointX(4),GetPointY(4),GetPointZ(4),4,artifactSpecialAIOther);

        //----- Units -----
        u_SpacePort=GetUnit(GetPointX(0),GetPointY(0,0));
        //----- Timers -----
        SetTimer(0,100);
        //----- Variables -----
        bCheckEndMission = false;
        n_Counter=0;
        //----- Camera -----
        CallCamera();
        EnableInterface(false);
        EnableCameraMovement(false);

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,128,20,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0,100,1);
        return ShowBriefing,100;
    }

    state ShowBriefing
    {
        EnableNextMission(0,true); //522
        EnableInterface(true);
        EnableCameraMovement(true);
        AddBriefing("translateBriefing521");
        return CaptureSpacePort;
    }

    state CaptureSpacePort
    {

```

```

if(GetGoalState(destroySpacePortDefenses)==goalAchieved &&
GetGoalState(captureSpacePort)==goalAchieved && n_Counter>3)
{
    SetGoalState(captureSpacePort,goalAchieved);
    n_Counter=0;
    AddBriefing("translateAccomplished521");//XXXMD JS zmiana
translateAccomplished521b -> translateAccomplished521
    CallCamera();
    p_Player.LookAt(GetPointX(0),GetPointY(0),6,128,20,0);
    p_Player.DelayedLookAt(GetPointX(0),GetPointY(0),6,0,20,0,100,1);
    SetPrize(primaryGoal);
    return ShowingCapturedSpacePort,100;
}
return CaptureSpacePort;
}

state ShowingCapturedSpacePort
{
    ShowVideo("Cutscene2");
    return EndMissionAfterVideo,0;
}

state EndMissionAfterVideo
{
    m_bSuccess = true;
    EndMission(true);
    return Nothing;
}

state Nothing
{
    return Nothing;
}

state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
event Timer0() //wolny co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(lu_SpacePort.IsLive())
    {
        AddBriefing("translateFailed521b");
        state EndMissionFailed;
    }
    if(!bCheckEndMission) return;

    bCheckEndMission=false;

    if(GetGoalState(destroyAllUnits)==goalAchieved &&
    !p_Enemy1.GetNumberOfUnits()&&
    !p_Enemy2.GetNumberOfUnits())
    {
        EnableGoal(destroyAllUnits,true);
        SetGoalState(destroyAllUnits, goalAchieved);
        SetPrize(hiddenGoal);
    }

    if(!p_Enemy1.GetNumberOfBuildings() &&
    GetGoalState(destroyEnemyBase)!=goalAchieved)
    {
        SetPrize(secondaryGoal);
        SetGoalState(destroyEnemyBase,goalAchieved);
    }

    if(!p_Enemy2.GetNumberOfBuildings() &&
    GetGoalState(destroySpacePortDefenses)!=goalAchieved)
    {
        SetGoalState(destroySpacePortDefenses,goalAchieved);
    }

    if(!p_Player.GetNumberOfUnits() && !p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateFailed521a");
        m_bSuccess = false;
        state EndMissionFailed;
    }
}

event UnitDestroyed(unit u_Unit)
{
}

```

```

bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event EndMission()
{
    SetPrize(endMission);
    p_Neutral.SetEnemy(p_Player);
    p_Player.SetEnemy(p_Neutral);
    p_Neutral.EnableAIFeatures(aiRejectAlliance,true);
}

//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    if(alD>4) return false;
}

CreateArtefact("NEAPLATE2",GetPointX(alD),GetPointY(alD),GetPointZ(alD),alD+1
0,artefactSpecialAIOther);
n_Counter=n_Counter+1;
return true; //usuwa sie
}

}

mission522.ec
mission "translateMission522"
{
    consts
    {
        destroySunLight = 0;
        fiodorowSurvived = 1;
    }

    player p_Enemy;
    player p_Ally;
    player p_Neutral;//TunellEntrance
    player p_Player;

    unitex u_Fiodorow;

    int bCheckEndMission;
    int n_AnimationStep;
    int b_Rain1;
    int b_Rain2;
    int b_Rain3;

    state Initialize;
    state ShowBriefing;
    state ShowBriefingAnimation;
    state DestroySunLights;
    state ShowDestroy;
    state Nothing;

//-----
state Initialize
{
    int i;
    int start;
    int end;
    int nShield;

    //----- Extra -----
    EnableGameFeature(lockResearchDialog,true);
    EnableGameFeature(lockConstructionDialog,true);
    EnableGameFeature(lockUpgradeWeaponDialog,true);

    //----- Goals -----
    RegisterGoal(destroySunLight,"translateGoal522a");
    RegisterGoal(fiodorowSurvived,"translateGoal522b");

    EnableGoal(destroySunLight,true);
    EnableGoal(fiodorowSurvived,true);

    //----- Players -----
    p_Player = GetPlayer(2);
    p_Enemy = GetPlayer(3);
    p_Neutral = GetPlayer(1);
    //----- AI -----
    p_Neutral.EnableStatistics(false);
    p_Neutral.SetNeutral(p_Player);
}

```



```
p_Player.SetNeutral(p_Neutral);
p_Neutral.EnableAIFeatures(aiEnabled,false);

if(GetDifficultyLevel()==0)
    p_Enemy.LoadScript("single\\singleEasy");
if(GetDifficultyLevel()==1)
    p_Enemy.LoadScript("single\\singleMedium");
if(GetDifficultyLevel()==2)
    p_Enemy.LoadScript("single\\singleHard");

p_Enemy.SetNeutral(p_Neutral);
p_Neutral.SetNeutral(p_Enemy);

p_Enemy.EnableAIFeatures(aiControlOffense,false);
p_Enemy.EnableAIFeatures(aiControlDefense,false);
p_Enemy.EnableAIFeatures(aiBuildBuildings,false);

//----- Money -----
p_Player.SetMoney(0);
if(GetDifficultyLevel()==0)
    p_Enemy.SetMoney(1000);
if(GetDifficultyLevel()==1)
    p_Enemy.SetMoney(2000);
if(GetDifficultyLevel()==2)
    p_Enemy.SetMoney(10000);
p_Neutral.SetMoney(0);

//----- Researches -----
//----- Buildings -----
p_Player.EnableCommand(commandSoldBuilding,true);
if(GetDifficultyLevel()==0)
{
    KillArea(p_Enemy.GetIFF(),GetPointX(21),GetPointY(21),0,1);
    KillArea(p_Enemy.GetIFF(),GetPointX(22),GetPointY(22),0,1);
    KillArea(p_Enemy.GetIFF(),GetPointX(23),GetPointY(23),0,1);
    KillArea(p_Enemy.GetIFF(),GetPointX(24),GetPointY(24),0,1);
    KillArea(p_Enemy.GetIFF(),GetPointX(25),GetPointY(25),0,1);
}
if(GetDifficultyLevel()==1)
{
    KillArea(p_Enemy.GetIFF(),GetPointX(22),GetPointY(22),0,1);
    KillArea(p_Enemy.GetIFF(),GetPointX(24),GetPointY(24),0,1);
}
//----- Artifacts -----
start=0;
end=0;
if(GetDifficultyLevel()==1)
{
    start=30;
    end=41;
}
if(GetDifficultyLevel()==2)
{
    start=30;
    end=61;
}
for(i=start;i<end;i+=1)
{
    CreateArtifact("NEAMINE",GetPointX(i),GetPointY(i),GetPointZ(i),i,artefactSpecialAIOther);
}

//----- Units -----
p_Player.CreateUnitEx(GetPointX(0),GetPointY(0),
0,null,"EDUSPECIAL","EDWSPECIAL",null,null,0);
u_Fiodorow = GetUnit((GetPointX(0),GetPointY(0));
u_Fiodorow.SetUnitName("translate522Fiodorow");
p_Player.AddUnitToSpecialTab(u_Fiodorow,true,-1);

if(GetDifficultyLevel()==0)
{
    end=9;
    nShield=0;
}
if(GetDifficultyLevel()==1)
{
    end=6;
    nShield=-1;
}
if(GetDifficultyLevel()==2)
{
    end=4;
    nShield=-1;
}
for (i=0;i<=end;i+=1)

    { p_Player.CreateUnitEx((GetPointX(70+i),GetPointY(70+i),0,null,"LCUU-
FO","LCWUFO",null,null,nShield);
        u_Fiodorow = GetUnit((GetPointX(70+i),GetPointY(70+i),0);
        if (i == 1) u_Fiodorow.SetUnitName("translate522Unit1");
        else if (i == 1) u_Fiodorow.SetUnitName("translate522Unit2");
        else if (i == 2) u_Fiodorow.SetUnitName("translate522Unit3");
        else if (i == 3) u_Fiodorow.SetUnitName("translate522Unit4");
        else if (i == 4) u_Fiodorow.SetUnitName("translate522Unit5");
        else if (i == 5) u_Fiodorow.SetUnitName("translate522Unit6");
        else if (i == 6) u_Fiodorow.SetUnitName("translate522Unit7");
        else if (i == 7) u_Fiodorow.SetUnitName("translate522Unit8");
        else if (i == 8) u_Fiodorow.SetUnitName("translate522Unit9");
        else if (i == 9) u_Fiodorow.SetUnitName("translate522Unit10");
        p_Player.AddUnitToSpecialTab(u_Fiodorow,true,-1);
    }
    u_Fiodorow = GetUnit((GetPointX(0),GetPointY(0));

//----- Timers -----
SetTimer(0,100);
//----- Variables -----
bCheckEndMission = false;
n_AnimationStep=0;
//----- Camera -----
CallCamera();
EnableInterface(false);
EnableCameraMovement(false);

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,128,
20,0);

p_Player.DelayedLookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
6,20,0,20,100,1);
return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    int nIntensity;
    if(GetDifficultyLevel()==0)
        nIntensity = 1;
    else if(GetDifficultyLevel()==1)
        nIntensity = 2;
    else
        nIntensity = 3;
    MeteorRain(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(),20,500,30000,100,nIntensity,1);
    EnableInterface(true); //zby mozna bylo nacisnac OK w briefingu
    AddBriefing("translateBriefing522");
    n_AnimationStep=0;
    ShowArea(4,GetPointX(10),GetPointY(10),0,10,showAreaBuildings); //cen-
    trum sterowania SunLight
    return ShowBriefingAnimation,1;
}
//-----
state ShowBriefingAnimation
{
    if(n_AnimationStep==0)
    {
        EnableInterface(false);
        p_Player.LookAt((GetPointX(10),GetPointY(10),10,128,20,0);
        p_Player.DelayedLookAt((GetPointX(10),GetPointY(10),10,0,20,0,100,1);
        n_AnimationStep=1;
        return ShowBriefingAnimation,101;
    }
    if(n_AnimationStep==1)
    {
        p_Player.DelayedLookAt((GetPointX(0),GetPointY(0),8,0,20,0,50,1);
        n_AnimationStep=2;
        return ShowBriefingAnimation,50;
    }
    if(n_AnimationStep==2)
    {
        n_AnimationStep=0;
        EnableInterface(true);
        EnableCameraMovement(true);
        return DestroySunLights;
    }
}
//-----
state DestroySunLights
{
    if(u_Fiodorow.DistanceTo((GetPointX(1),GetPointY(1))<2 &&
        u_Fiodorow.GetLocationZ() == 1)
    {
```

```

SetGoalState(destroySunLight.goalAchieved);
SetGoalState(fedorowSurvived.goalAchieved);
AddBriefing("translateAccomplished522");
n_AnimationStep = 0;
return ShowDestroy,1;
}
return DestroySunLights;
}
//-----
state ShowDestroy
{
    int n_Time;
    n_Time=20;

    if(n_AnimationStep==0)//move to PP1
    {
        TraceD("move to PP1\n");
    }

ShowArea(4,GetPointX(10),GetPointY(10),0,20,showAreaBuildings);//centrum sterowania SunLight
    EnableInterface(false);
    p_Player.LookAt(GetPointX(10),GetPointY(10),15,127,40,0);

p_Player.DelayedLookAt(GetPointX(10),GetPointY(10),15,128,40,0,80,0);
    n_AnimationStep=2;
    return ShowDestroy,15//n_Time
}
if(n_AnimationStep>1 && n_AnimationStep<6)//destroy PP1
{
    TraceD("destroy PP1\n");

KillArea(15,GetPointX(n_AnimationStep),GetPointY(n_AnimationStep),0,0.7);
    n_AnimationStep=n_AnimationStep+1;
    if(n_AnimationStep==6)
        return ShowDestroy,3;
    return ShowDestroy,20;
}
if(n_AnimationStep==6)//destroy everything
{
    n_AnimationStep = 7;
}

p_Player.DelayedLookAt(GetPointX(10),GetPointY(10),15,135,40,0,120,0);
    return ShowDestroy,15;
}
if(n_AnimationStep==7)//destroy everything
{
    TraceD("destroy everything\n");
    KillArea(15,GetPointX(10),GetPointY(10),0,15);
    n_AnimationStep = 8;
    return ShowDestroy,63;
}
if(n_AnimationStep==8)//destroy everything
{
    n_AnimationStep = 9;
}

p_Player.DelayedLookAt(GetPointX(10),GetPointY(10),15,140,40,0,120,0);
    return ShowDestroy,35;
}
if(n_AnimationStep==9)
{
    EnableInterface(true);
    EnableNextMission(0,3)// Win
}
/*if(n_AnimationStep==0)//move to PP1
{
    TraceD("move to PP1\n");
    EnableInterface(false);
    p_Player.LookAt(GetPointX(10),GetPointY(10),10,0,20,0);
    p_Player.DelayedLookAt(GetPointX(2),GetPointY(2),10,0,20,0,n_Time,1);
    n_AnimationStep=2;
    return ShowDestroy,15//n_Time
}
if(n_AnimationStep==2)//destroy PP1
{
    EnableInterface(true);
    TraceD("destroy PP1\n");
    KillArea(15,GetPointX(2),GetPointY(2),0,0.7);
    n_AnimationStep = 3;
    EnableInterface(false);
    return ShowDestroy,20;
}
if(n_AnimationStep==3)//move to PP2
{
    TraceD("move to PP2\n");
    n_AnimationStep = 4;
}

p_Player.DelayedLookAt(GetPointX(3),GetPointY(3),10,0,20,0,n_Time,1);
    return ShowDestroy,15;
}
if(n_AnimationStep==4)//destroy PP2
{
    TraceD("destroy PP2\n");
    KillArea(15,GetPointX(3),GetPointY(3),0,0.7);
    n_AnimationStep = 5;
    return ShowDestroy,20;
}
}
if(n_AnimationStep==5)//move to PP3
{
    TraceD("move to PP3\n");
    n_AnimationStep = 6;
    p_Player.DelayedLookAt(GetPointX(4),GetPointY(4),10,0,20,0,n_Time,1);
    return ShowDestroy,15;
}
if(n_AnimationStep==6)//destroy PP3
{
    TraceD("destroy PP3\n");
    KillArea(15,GetPointX(4),GetPointY(4),0,0.7);
    n_AnimationStep = 7;
    return ShowDestroy,20;
}
}
if(n_AnimationStep==7)//move to PP4
{
    TraceD("move to PP4\n");
    n_AnimationStep = 8;
    p_Player.DelayedLookAt(GetPointX(5),GetPointY(5),10,0,20,0,n_Time,1);
    return ShowDestroy,15;
}
if(n_AnimationStep==8)//destroy PP4
{
    TraceD("destroy PP4\n");
    KillArea(15,GetPointX(5),GetPointY(5),0,0.7);
    n_AnimationStep = 9;
    return ShowDestroy,15;
}
}
if(n_AnimationStep==9)//move to center
{
    TraceD("move to center\n");
}

p_Player.DelayedLookAt(GetPointX(10),GetPointY(10),30,0,20,0,n_Time,1);
    n_AnimationStep=10;
    return ShowDestroy,50;
}
if(n_AnimationStep==10)//destroy everything
{
    TraceD("destroy everything\n");
    KillArea(15,GetPointX(10),GetPointY(10),0,15);
    n_AnimationStep = 11;
    return ShowDestroy,100;
}
}
if(n_AnimationStep==11)
{
    EnableInterface(true);
    EnableNextMission(0,3)// Win
}
*/
return Nothing;
}
//-----
state Nothing
{
    return Nothing;
}
//-----
state EndGame
{
    EnableNextMission(0,2)// loose
}

//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    int nIntensity;
    ShowArea(4,GetPointX(10),GetPointY(10),0,0.8,showAreaBuildings);//centrum sterowania SunLight
    if(!b_Rain1 && p_Player.IsPointLocated(GetPointX(11),GetPointY(11),0))
    {
        b_Rain1=true;
    }
}

```

```

    if(GetDifficultyLevel() == 0)
        nIntensity = 1;
    else if(GetDifficultyLevel() == 1)
        nIntensity = 2;
    else
        nIntensity = 3;

    MeteorRain(GetPointX(11),GetPointY(11),20,500,30000,100,nIntensity,1);
}
if(!b_Rain2 && p_Player.IsPointLocated(GetPointX(12),GetPointY(12),0))
{
    b_Rain2=true;
    if(GetDifficultyLevel() == 0)
        nIntensity = 1;
    else if(GetDifficultyLevel() == 1)
        nIntensity = 2;
    else
        nIntensity = 3;

    MeteorRain(GetPointX(12),GetPointY(12),20,500,30000,100,nIntensity,1);
}
if(!b_Rain3 && p_Player.IsPointLocated(GetPointX(13),GetPointY(13),0))
{
    b_Rain3=true;
    if(GetDifficultyLevel() == 0)
        nIntensity = 1;
    else if(GetDifficultyLevel() == 1)
        nIntensity = 2;
    else
        nIntensity = 3;

    MeteorRain(GetPointX(13),GetPointY(13),20,500,30000,100,nIntensity,1);
}

if(!bCheckEndMission) return;

bCheckEndMission=false;

if(!u_Fiodorow.IsLive())
{
    AddBriefing("translateFailed522");
    state EndGame;
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
}

//-----
event EndMission()
{
}

//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer==p_Player) return false;
    if(alD>29 && alD<=60)//mines
    {
        KillArea(15, GetPointX(alD),GetPointY(alD),GetPointZ(alD),0);
        return true; //usuwa sie
    }
    return true; //usuwa sie
}

}



## TutorialEDmis.ec


mission "translateTutorialED"
{
    consts
    {
        buildPowerPlant = 0;
        buildVechProdCent = 1;
        buildMine = 2;
        buildRefinery = 3;
        buildTransporters = 4;
        harvestResources = 5;
        buildReapProdCent = 6;
        buildArmy = 7;
        findEnemy = 8;
        destroyEnemy = 9;
    }
}

player p_EnemyUCS;
player p_Player;

int nUCSUnitsCount;
int nBuildingCount;
int nMoney;
int nStartX;
int nStartY;

//*****
state Initialize;
state Start;
state MoveCamera;
state BuildPowerPlant;
state BuildVechProdCent;
state BuildMine;
state BuildRefinery;
state BuildTransporters;
state HarvestResources;
state BuildWeapProdCent;
state BuildArmy;
state More;
state EndState;

state Initialize
{
    RegisterGoal(buildPowerPlant,"translateGoalTutorialED_PP");
    RegisterGoal(buildVechProdCent,"translateGoalTutorialED_BA");
    RegisterGoal(buildMine,"translateGoalTutorialED_MI");
    RegisterGoal(buildRefinery,"translateGoalTutorialED_RE");
    RegisterGoal(buildTransporters,"translateGoalTutorialED_OH");
    RegisterGoal(harvestResources,"translateGoalTutorialED_Mining");
    RegisterGoal(buildWeapProdCent,"translateGoalTutorialED_FA");
    RegisterGoal(buildArmy,"translateGoalTutorialED_tanks");
    RegisterGoal(findEnemy,"translateGoalTutorialED_findEnemy");
    RegisterGoal(destroyEnemy,"translateGoalTutorialED_destroyEnemy");
}

p_EnemyUCS=GetPlayer(1);
p_Player=GetPlayer(2);

p_EnemyUCS.SetMoney(0);
p_Player.SetMoney(20000);

p_EnemyUCS.EnableAI(false);
p_Player.EnableAI(false);

//weapons
p_Player.EnableResearch("RES_ED_ACH2",false);
p_Player.EnableResearch("RES_ED_WSL1",false);
p_Player.EnableResearch("RES_ED_WSL1*",false);
p_Player.EnableResearch("RES_ED_WSR2",false);
p_Player.EnableResearch("RES_ED_WMR1",false);
p_Player.EnableResearch("RES_ED_WH1",false);
p_Player.EnableResearch("RES_ED_WH1*",false);
p_Player.EnableResearch("RES_ED_WH1L",false);
p_Player.EnableResearch("RES_ED_WH1R",false);
p_Player.EnableResearch("RES_ED_AB1",false);
//ammo
p_Player.EnableResearch("RES_ED_MHC2",false);
p_Player.EnableResearch("RES_ED_MB2",false);
p_Player.EnableResearch("RES_ED_MSC3",false);
p_Player.EnableResearch("RES_MM2",false);
p_Player.EnableResearch("RES_MS3",false);
p_Player.EnableResearch("RES_ED_MHR2",false);
p_Player.EnableResearch("RES_ED_MHR4",false);
//chassis
p_Player.EnableResearch("RES_ED_UT3",false);
p_Player.EnableResearch("RES_ED_UT2",false);
p_Player.EnableResearch("RES_ED_UW1",false);
p_Player.EnableResearch("RES_ED_UW1*",false);
p_Player.EnableResearch("RES_ED_UHT1",false);
p_Player.EnableResearch("RES_ED_UHW1",false);
p_Player.EnableResearch("RES_ED_UTB1",false);
p_Player.EnableResearch("RES_ED_UA1",false);
p_Player.EnableResearch("RES_ED_UA2",false);
p_Player.EnableResearch("RES_ED_UA3",false);
p_Player.EnableResearch("RES_ED_UA4",false);
p_Player.EnableResearch("RES_ED_US2",false);
p_Player.EnableResearch("RES_ED_UHS1",false);
//special
p_Player.EnableResearch("RES_ED_SCR",false);
p_Player.EnableResearch("RES_ED_S6n",false);
p_Player.EnableResearch("RES_ED_BMD",false);
p_Player.EnableResearch("RES_ED_BHD",false);
p_Player.EnableResearch("RES_ED_ReHand2",false);
}

```

```

// 1st tab
p_Player.EnableBuilding("EDBPP",false);
p_Player.EnableBuilding("EDBBA",false);
p_Player.EnableBuilding("EDBFA",false);
p_Player.EnableBuilding("EDBWB",false);
p_Player.EnableBuilding("EDBAB",false);
// 2nd tab
p_Player.EnableBuilding("EDBRE",false);
p_Player.EnableBuilding("EDBMA",false);
p_Player.EnableBuilding("EDBCT",false);
// 3rd tab
p_Player.EnableBuilding("EDBST",false);
// 4th tab
p_Player.EnableBuilding("EDBRC",false);
p_Player.EnableBuilding("EDBHO",false);
p_Player.EnableBuilding("EDBRA",false);
p_Player.EnableBuilding("EDBEN1",false);
p_Player.EnableBuilding("EDBLZ",false);

nStartX = p_Player.GetStartingPointX();
nStartY = p_Player.GetStartingPointY();

LookAt(nStartX,nStartY,6.0,20.0,0);
SetTimer(0,6000);
SetTimer(1,100);
nBuildingCount=0;
return Start,100;
}

state Start
{
    AddBriefing("translateTutorialED_moveCamera");
    nUCUnitsCount=p_EnemyUCS.GetNumberOfUnits()/2;
    return MoveCamera,600;
}

state MoveCamera
{
    p_Player.EnableBuilding("EDBPP",true);
    EnableGoal(buildPowerPlant,true);
    AddBriefing("translateTutorialED_PP");
    return BuildPowerPlant,100;
}

state BuildPowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingPowerPlant))
    {
        p_Player.EnableBuilding("EDBBA",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildPowerPlant.goalAchieved);
        EnableGoal(buildVechProdCent,true);
        if(p_Player.GetNumberOfBuildings(buildingBase))
            AddBriefing("translateTutorialED_BA");
        return BuildVechProdCent,100;
    }
    if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
    {
        p_Player.CreateUnitEx(nStartX,nStartY,
        0,null,"EDUBU1",null,null,null,null);
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildPowerPlant,60;
}

state BuildVechProdCent
{
    if(p_Player.GetNumberOfBuildings(buildingBase))
    {
        p_Player.EnableBuilding("EDBMA",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildVechProdCent.goalAchieved);
        EnableGoal(buildMine,true);
        if(p_Player.GetNumberOfBuildings(buildingMine))
            AddBriefing("translateTutorialED_MI");
        return BuildMine,100;
    }
    if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
    {
        p_Player.CreateUnitEx(nStartX,nStartY,
        0,null,"EDUBU1",null,null,null,null);
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildVechProdCent,60;
}

state BuildMine
{
    if(p_Player.GetNumberOfBuildings(buildingMine))
    {
        p_Player.EnableBuilding("EDBRE",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildMine.goalAchieved);
        EnableGoal(buildRefinery,true);
        if(p_Player.GetNumberOfBuildings(buildingRefinery))
            AddBriefing("translateTutorialED_RE");
        return BuildRefinery,100;
    }
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialED_BA_Fool");
    }
    return BuildVechProdCent,60;
}

state BuildRefinery
{
    if(p_Player.GetNumberOfBuildings(buildingRefinery))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildRefinery.goalAchieved);
        EnableGoal(buildTransporters,true);
        if(p_Player.GetNumberOfUnits(chassisTank | unitCarrier)<2)
            AddBriefing("translateTutorialED_OH");
        return BuildTransporters,100;
    }
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialED_RE_Fool");
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildRefinery,100;
}

state BuildTransporters
{
    if(p_Player.GetNumberOfUnits(chassisTank | unitCarrier)>=2)
    {
        SetGoalState(buildTransporters.goalAchieved);
        EnableGoal(harvestResources,true);
        AddBriefing("translateTutorialED_Mining");
        nMoney=p_Player.GetMoney();
        return HarvestResources,100;
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildTransporters,100;
}

state HarvestResources
{
    if(nMoney < p_Player.GetMoney())
    {
        p_Player.EnableBuilding("EDBFA",true);
        SetGoalState(harvestResources.goalAchieved);
        EnableGoal(buildWeapProdCent,true);
        if(p_Player.GetNumberOfBuildings(buildingFactory))
            AddBriefing("translateTutorialED_FA");
        return BuildWeapProdCent,100;
    }
    return HarvestResources,100;
}

```

```

//-----
state BuildWeapProdCent
{
    if(p_Player.GetNumberOfBuildings(buildingFactory))
    {
        SetGoalState(buildWeapProdCent.goalAchieved);
        EnableGoal(buildArmy,true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        AddBriefing("translateTutorialED_tanks");
        return BuildArmy;
    }
    if(p_Player.GetNumberOfBuildings()>nBuildingCount)
    {
        nBuildingCount=nBuildingCount+1;
    }
    return BuildWeapProdCent,100;
}
//-----

state BuildArmy
{
    if(p_Player.GetNumberOfUnits(chassisTank | unitArmed)>=5)
    {
        SetGoalState(buildArmy.goalAchieved);
        EnableGoal(findEnemy,true);
        AddBriefing("translateTutorialED_findEnemy");
        return More,600;
    }
    return BuildArmy,200;
}
//-----

state More
{
    p_Player.EnableBuilding("EDBAP",true);
    p_Player.EnableBuilding("EDBST",true);
    p_Player.EnableBuilding("EDBRC",true);
    p_Player.EnableBuilding("EDBEN1",true);
    AddBriefing("translateTutorialED_more");
    return EndState,100;
}
//-----

//-----
state EndState
{
    return EndState,500;
}
//-----

event Timer0()
{
    Snow(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),40,400,5000,800
,8);
}
//-----

event Timer1()
{
    if(GetGoalState(findEnemy)==goalAchieved &&
p_Player.IsPointLocated(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStartin
gPointY(),0,0))
    {
        //LookAt(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStartingPointY(),6,0,2
,0,0);
        SetGoalState(findEnemy,goalAchieved);
        EnableGoal(destroyEnemy,true);
        AddBriefing("translateTutorialED_destroyEnemy");
    }
    if(GetGoalState(destroyEnemy)!=goalAchieved &&
lp_EnemyUCS.GetNumberOfUnits())
    {
        SetGoalState(destroyEnemy,goalAchieved);
        AddBriefing("translateTutorialED_Victory");
    }
}
//-----

TutorialED1mis.ec
mission "translateTutorialED 1"
{
    consts
    {
        buildBridge = 0;
        buildTunnelEntrance1 = 1;
        buildTunnel = 2;
        buildTunnelEntrance2 = 3;
        useUnitTransporter = 4;
    }
    player p_Player;
    unitex_u_Builder;
    //*****
    state Initialize;
    state Start;
    state BuildBridge;
    state BuildTunnelEntrance1;
    state BuildTunnel;
    state BuildTunnelEntrance2;
    state UseUnitTransporter;
    state EndState;
    state Initialize
    {
        RegisterGoal(buildBridge,"translateGoalTutorialED1_1");
        RegisterGoal(buildTunnelEntrance1,"translateGoalTutorialED1_2");
        RegisterGoal(buildTunnel,"translateGoalTutorialED1_3");
        RegisterGoal(buildTunnelEntrance2,"translateGoalTutorialED1_4");
        RegisterGoal(useUnitTransporter,"translateGoalTutorialED1_5");
    }
    p_Player=GetPlayer(1);
    if(p_Player==null)p_Player.EnableStatistics(false);
    p_Player=GetPlayer(3);
    if(p_Player==null)p_Player.EnableStatistics(false);
    p_Player=GetPlayer(2);
    if(p_Player==null)p_Player.EnableStatistics(false);
    p_Player.SetMoney(20000);
    p_Player.EnableAI(false);
    EnableGameFeature(lockResearchDialog,true);
    EnableGameFeature(lockConstructionDialog,true);
    EnableGameFeature(lockUpgradeWeaponDialog,true);
    // 1st tab
    p_Player.EnableBuilding("EDBPP",false);
    p_Player.EnableBuilding("EDBBA",false);
    p_Player.EnableBuilding("EDBFA",false);
    p_Player.EnableBuilding("EDBWB",false);
    p_Player.EnableBuilding("EDBAB",false);
    // 2nd tab
    p_Player.EnableBuilding("EDBRE",false);
    p_Player.EnableBuilding("EDBML",false);
    p_Player.EnableBuilding("EDBTC",false);
    // 3rd tab
    p_Player.EnableBuilding("EDBST",false);
    // 4th tab
    p_Player.EnableBuilding("EDBUC",false);
    p_Player.EnableBuilding("EDBRC",false);
    p_Player.EnableBuilding("EDBHQ",false);
    p_Player.EnableBuilding("EDBRA",false);
    p_Player.EnableBuilding("EDBEN1",false);
    p_Player.EnableBuilding("EDBLZ",false);
    p_Player.EnableCommand(commandBuildTrench,false);
    p_Player.EnableCommand(commandBuildFlatTerrain,false);
    p_Player.EnableCommand(commandBuildWall,false);
    p_Player.EnableCommand(commandBuildRoad,false);
    p_Player.EnableCommand(commandBuildWideTunnel,false);
    p_Player.EnableCommand(commandBuildNarrowTunnel,false);
    p_Player.EnableCommand(commandBuildWideBridge,false);
    p_Player.EnableCommand(commandBuildNarrowBridge,true);
    u_Builder=GetUnit(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),0);
    p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,128,
20,0);
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),40,400,5000,800,
8);
    return Start,100;
}
//-----

state Start
{
    EnableGoal(buildBridge,true);
}

```

```

AddBriefing("translateTutorialED1_1");
CreateArtefact("NEABANNED", GetPointX(0),
GetPointY(0),0,0,artefactSpecialAIOther);
return BuildBridge;
}
//-----

state BuildBridge
{
if(u_Builder.GetLocationY()>GetPointY(1))
{
SelectUnit(null,false);
p_Player.EnableCommand(commandBuildNarrowTunnel,true);
p_Player.EnableBuilding("EDBEN1",true);
SelectUnit(u_Builder,false);
}

EnableGoal(buildTunnelEntrance1,true);
SetGoalState(buildBridge,goalAchieved);

AddBriefing("translateTutorialED1_2");
CreateArtefact("NEABANNED", GetPointX(2),
GetPointY(2),0,0,artefactSpecialAIOther);
return BuildTunnelEntrance1,20;
}
return BuildBridge;
}
//-----
```

```

state BuildTunnelEntrance1
{
if(u_Builder.GetLocationZ()!=0)
{
EnableGoal(buildTunnel,true);
SetGoalState(buildTunnelEntrance1,goalAchieved);
AddBriefing("translateTutorialED1_3");
return BuildTunnel;
}
return BuildTunnelEntrance1;
}
//-----
```

```

state BuildTunnel
{
if(u_Builder.DistanceTo(GetPointX(3),GetPointY(3))<5)
{
EnableGoal(buildTunnelEntrance2,true);
SetGoalState(buildTunnel,goalAchieved);
AddBriefing("translateTutorialED1_4");
return BuildTunnelEntrance2;
}
return BuildTunnel;
}
//-----
```

```

state BuildTunnelEntrance2
{
if(u_Builder.GetLocationZ()==0
&&u_Builder.DistanceTo(GetPointX(3),GetPointY(3))<7)
{
EnableGoal(useUnitTransporter,true);
SetGoalState(buildTunnelEntrance2,goalAchieved);
AddBriefing("translateTutorialED1_5");
p_Player.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "EDEUT",
"EDSUT", null, null, null);
CreateArtefact("NEABANNED", GetPointX(4),
GetPointY(4),0,0,artefactSpecialAIOther);
return useUnitTransporter;
}
return BuildTunnelEntrance2;
}
//-----
```

```

state useUnitTransporter
{
if(u_Builder.GetLocationZ()==0 &&
!u_Builder.IsTransported() &&
u_Builder.DistanceTo(GetPointX(4),GetPointY(4))<3)
{
SetGoalState(useUnitTransporter,goalAchieved);
AddBriefing("translateTutorialED1_6");
return EndState;
}
return useUnitTransporter;
}
//-----
```

```

state EndState
{
return EndState,500;
}
//-----
```

```

event Artefact(int aiD,player piPlayer)
{
return false;
}
}

TutorialED2mis.ec
mission "translateTutorialED2"
{
consts
{
sellUnits = 0;
sellBuildings = 1;
captureUnits = 2;
captureBuilding1 = 3;
captureBuilding2 = 4;
}
player p_Player;
player p_Dummy1;
player p_Dummy2;
player p_Enemy;

//*****
state Initialize;
state Start;
state SellUnits;
state SellBuilding;
state Wait1;
state CaptureUnits;
state Wait2;
state CaptureBuilding;
state EndState;

state Initialize
{
RegisterGoal(sellUnits,"translateGoalTutorialED2_1");
RegisterGoal(sellBuildings,"translateGoalTutorialED2_2");
RegisterGoal(captureUnits,"translateGoalTutorialED2_3");
RegisterGoal(captureBuilding1,"translateGoalTutorialED2_4");
RegisterGoal(captureBuilding2,"translateGoalTutorialED2_5");

p_Player = GetPlayer(2);
p_Dummy1 = GetPlayer(3);
p_Dummy2 = GetPlayer(4);
p_Enemy = GetPlayer(1);

p_Player.EnableStatistics(false);
p_Dummy1.EnableStatistics(false);
p_Dummy2.EnableStatistics(false);
p_Enemy.EnableStatistics(false);

p_Dummy1.SetNeutral(p_Enemy);
p_Dummy2.SetNeutral(p_Enemy);

p_Player.SetMoney(0);

p_Player.EnableAI(false);
p_Dummy1.EnableAI(false);
p_Dummy2.EnableAI(false);
p_Enemy.EnableAI(false);

EnableGameFeature(lockResearchDialog,true);
EnableGameFeature(lockConstructionDialog,true);
EnableGameFeature(lockUpgradeWeaponDialog,true);
p_Player.EnableCommand(commandAutodestruction,false);

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return Start,100;
}
//-----
```

```

state Start
{
EnableGoal(sellUnits,true);
AddBriefing("translateTutorialED2_1");
return SellUnits;
}
//-----
```

```

-----  

state SellUnits  

{  

    if(p_Player.GetNumberOfUnits()  

    {  

        EnableGoal(sellBuildings,true);  

        SetGoalState(sellUnits,goalAchieved);  

        p_Player.EnableCommand(commandSoldBuilding,true);  

        AddBriefing("translateTutorialED2_2");  

        return SellBuilding;  

    }  

    return SellUnits;  

}  

-----  

state SellBuilding  

{  

    unitex u_Tmp;  

    if(p_Player.GetNumberOfBuildings()  

    {  

        SetGoalState(sellBuildings,goalAchieved);  

        return Wait1,40;  

    }  

    return SellBuilding;  

}  

-----  

state Wait1  

{  

    unitex u_Tmp;  

    EnableGoal(captureUnits,true);  

    EnableGoal(captureBuilding1,true);  

    AddBriefing("translateTutorialED2_3");  

    p_Player.EnableCommand(commandSoldBuilding,false);  

    p_Dummy1.GiveAllUnitsTo(p_Player);  

    u_Tmp=GetUnit(GetPointX(0),GetPointY(0));  

    u_Tmp.LoadScript("Scripts\Units\Tank.ecomp");  

    u_Tmp=GetUnit(GetPointX(0),GetPointY(0));  

    u_Tmp.LoadScript("Scripts\Units\Repairer.ecomp");  

    LookAt(GetPointX(0),GetPointY(0),6,0,20,0.0);  

    return CaptureUnits;  

}  

-----  

state CaptureUnits  

{  

    if(p_Player.GetNumberOfUnits()==5)  

    {  

        SetGoalState(captureUnits,goalAchieved);  

    }  

    if(p_Player.GetNumberOfBuildings()>1)  

    {  

        SetGoalState(captureBuilding1,goalAchieved);  

    }  

    if(GetGoalState(captureUnits)==goalAchieved &&  

    GetGoalState(captureBuilding1)==goalAchieved)  

    {  

        return Wait2,40;  

    }  

    return CaptureUnits;  

}  

-----  

state Wait2  

{  

    unitex u_Tmp;  

    p_Player.GiveAllUnitsTo(p_Dummy1);  

    p_Player.GiveAllBuildingsTo(p_Dummy1);  

    p_Dummy2.GiveAllUnitsTo(p_Player);  

    u_Tmp=GetUnit(GetPointX(1),GetPointY(1),0);  

    u_Tmp.LoadScript("Scripts\Units\Repairer.ecomp");  

    LookAt(GetPointX(1),GetPointY(1),6,0,20,0.0);  

    EnableGoal(captureBuilding2,true);  

    AddBriefing("translateTutorialED2_4");  

    return CaptureBuilding;  

}  

-----  

state CaptureBuilding  

{  

    if(p_Player.GetNumberOfBuildings()>1)

```

```

        SetGoalState(captureBuilding2,goalAchieved);
        AddBriefing("translateTutorialED2_5");
        return EndState,500;
    }
    return CaptureBuilding;
}  

-----  

state EndState  

{
    return EndState,500;
}  

-----  

event Artefact(int alD,player piPlayer)
{
    return false;
}
}

TutorialED3base1.ec
mission "translateTutorialED3"
{
    consts
    {
        goalBuildLandingZone = 0;
        goalStartMission = 1;
        goalFinishMission = 2;
    }
    player p_Player;

    state Initialize;
    state PlayTrackState;
    state ShowBriefing;
    state BuildLandingZone;
    state FinishMission;
    state Nothing;

    state Initialize
    {
        //----- Goals -----
        RegisterGoal(goalBuildLandingZone,"translateGoalTutorialED3_1");
        RegisterGoal(goalStartMission,"translateGoalTutorialED3_2");
        RegisterGoal(goalFinishMission,"translateGoalTutorialED3_3");

        EnableGoal(goalBuildLandingZone,true);

        //----- Players -----
        p_Player = GetPlayer(2);
        //----- AI -----
        p_Player.EnableAIFeatures(aiEnabled,false);
        //----- Money -----
        p_Player.SetMoney(10000);

        //----- Buildings -----
        p_Player.EnableBuilding("EDBPP",false);
        p_Player.EnableBuilding("EDBBA",false);
        p_Player.EnableBuilding("EDBFA",false);
        p_Player.EnableBuilding("EDBWB",false);
        p_Player.EnableBuilding("EDBAB",false);
        // 2nd tab
        p_Player.EnableBuilding("EDBRE",false);
        p_Player.EnableBuilding("EDBML",false);
        p_Player.EnableBuilding("EDBTC",false);
        // 3rd tab
        p_Player.EnableBuilding("EDBST",false);
        p_Player.EnableBuilding("EDBBT",false);
        p_Player.EnableBuilding("EDBHT",false);
        // 4th tab
        p_Player.EnableBuilding("EDBUC",false);
        p_Player.EnableBuilding("EDBRC",false);
        p_Player.EnableBuilding("EDBHQ",false);
        p_Player.EnableBuilding("EDBRA",false);
        p_Player.EnableBuilding("EDBN1",false);
        p_Player.EnableBuilding("EDBLZ",true);
        p_Player.EnableBuilding("EDBART",false);

        EnableGameFeature(lockResearchDialog,true);
        EnableGameFeature(lockConstructionDialog,true);
        EnableGameFeature(lockUpgradeWeaponDialog,true);

        p_Player.EnableCommand(commandAutodestruction,false);
    }
}

```

```

p_Player.EnableCommand(commandBuildOldBuilding,false);
p_Player.EnableCommand(commandBuildTrench,false);
p_Player.EnableCommand(commandBuildFlatTerrain,false);
p_Player.EnableCommand(commandBuildWall,false);
p_Player.EnableCommand(commandBuildRoad,false);
p_Player.EnableCommand(commandBuildWideBridge,false);
p_Player.EnableCommand(commandBuildNarrowBridge,false);
p_Player.EnableCommand(commandBuildWideTunnel,false);
p_Player.EnableCommand(commandBuildNarrowTunnel,false);

//----- Camera -----
CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return PlayTrackState,3;
}

state PlayTrackState
{
PlayTrack("music\edday_3.mp2");
return ShowBriefing,50;
}
//-----
state ShowBriefing
{
AddBriefing("translateTutorialED3_1",p_Player.GetName());
return BuildLandingZone;
}
//-----
state BuildLandingZone
{
if(p_Player.GetNumberOfBuildings()>0)
{
EnableNextMission(0,1);
SetGoalState(goalBuildLandingZone.goalAchieved);
EnableGoal(goalStartMission,true);
AddBriefing("translateTutorialED3_2",p_Player.GetName());
return FinishMission;
}
return BuildLandingZone;
}
//-----
state FinishMission
{
if(p_Player.GetScriptData(1))
{
p_Player.SetScriptData(1,0);
EnableGoal(goalFinishMission,true);
SetGoalState(goalStartMission,goalAchieved);
}
if(p_Player.GetScriptData(2))
{
p_Player.SetScriptData(2,0);
SetGoalState(goalFinishMission,goalAchieved);
AddBriefing("translateTutorialED3_3",p_Player.GetName());
return Nothing,50;
}
return FinishMission;
}
//-----
state Nothing
{
return Nothing,50;
}
}

}
player p_Player;
//*****
state Initialize;
state Start;
state Start2;
state Working;
state EndState;

state Initialize
{
RegisterGoal(goalBringGruz,"translateGoalTutorialED3_4");
RegisterGoal(goalBringCash,"translateGoalTutorialED3_5");

p_Player=GetPlayer(1);
if(p_Player!=null)p_Player.EnableStatistics(false);
p_Player=GetPlayer(3);
if(p_Player!=null)p_Player.EnableStatistics(false);

p_Player = GetPlayer(2);
p_Player.SetMoney(0);
p_Player.EnableAI(false);

EnableGameFeature(lockResearchDialog,true);
EnableGameFeature(lockConstructionDialog,true);
EnableGameFeature(lockUpgradeWeaponDialog,true);
p_Player.EnableCommand(commandAutodestruction,false);
p_Player.EnableCommand(commandSellBuilding,false);

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
p_Player.SetScriptData(1,1);
return Start,100;
}
//-----
state Start
{
AddBriefing("translateTutorialED3_4");
return Start2,400;
}
//-----
state Start2
{
EnableGoal(goalBringGruz,true);
EnableGoal(goalBringCash,true);
AddBriefing("translateTutorialED3_5");
return Working;
}
//-----
state Working
{
if(p_Player.GetNumberOfUnits()>0)
SetGoalState(goalBringGruz,goalAchieved);
if(p_Player.GetMoney()>=5000)
SetGoalState(goalBringCash,goalAchieved);

if(GetGoalState(goalBringGruz)==goalAchieved &&
GetGoalState(goalBringCash)==goalAchieved)
{
EnableEndMissionButton(true);
}
}

```

TutorialED3base2.ec

```
mission "translateTutorialED31"
{
    state Nothing
    {
        return Nothing,50;
    }
}
```

TutorialED3mis.ec

```
mission "translateLesson4"
{
    consts
    {
        goalBringGruz = 0;
        goalBringCash = 1;
```

MP-LC

CampaignLCMP01.ec

campaign "translateCampaignLCMP01"

```
{  
    consts  
    {  
        raceUCS=1;  
        raceLC=3;  
  
        mis611 = 0;  
        mis612 = 1;  
        mis613 = 2;  
        mis614 = 3;  
        mis615 = 4;  
        mis616 = 5;  
        mis617 = 6;  
        mis618 = 7;  
        mis619 = 8;  
        mis620 = 9;  
        mis621 = 10;  
        mis622 = 11;  
  
        base1 = 20;  
        base2 = 21;  
        base3 = 22;  
  
        timeFlagWinter = 1;  
        baseFlag = 255;  
    }  
  
    state Initialize;  
    state Start;  
    state Nothing;  
  
    state Initialize  
    {  
        int i;  
  
        RegisterMission(base1,"!LCbase1","LC\\Missions\\LCbase1Script","","  
        baseFlag,0,-90,0,0);  
        RegisterMission(base2,"!LCbase2","LC\\Missions\\LCbase2Script","","  
        baseFlag,0,90,0,0);  
        RegisterMission(base3,"!LCbase3","LC\\Missions\\LCbase3Script","","  
        baseFlag,0,90,0,0);  
  
        //-----  
        // nr Ind script preBriefing  
        timeFlags x y d1 d2 d3 next1 next2 next3 next4  
        RegisterMission(mis611, "l611", "LC\\Missions\\Mission611",  
        "translateBriefingShort611", timeFlagWinter, -170, 80, 60, 60, 60,  
        mis621,mis622);  
        RegisterMission(mis612, "l612", "LC\\Missions\\Mission612",  
        "translateBriefingShort612", timeFlagWinter, -20, 40, 60, 60, mis614);  
        RegisterMission(mis613, "l613", "LC\\Missions\\Mission613",  
        "translateBriefingShort613", timeFlagWinter, -40, 20, 20, mis615);  
        RegisterMission(mis614, "l614", "LC\\Missions\\Mission614",  
        "translateBriefingShort614", timeFlagWinter, -160, 20, 20, 20, mis616);  
        RegisterMission(mis615, "l615", "LC\\Missions\\Mission615",  
        "translateBriefingShort615", timeFlagWinter, -136, 0, 20, 20, mis617);  
        RegisterMission(mis616, "l616", "LC\\Missions\\Mission616",  
        "translateBriefingShort616", timeFlagWinter, -125, -10, 20, 20, mis618);  
        RegisterMission(mis617, "l617", "LC\\Missions\\Mission617",  
        "translateBriefingShort617", timeFlagWinter, -224, 0, 20, 20, 20, mis619);  
        RegisterMission(mis618, "l618", "LC\\Missions\\Mission618",  
        "translateBriefingShort618", timeFlagWinter, -50, 10, 20, 20, mis620);  
        RegisterMission(mis619, "l619", "LC\\Missions\\Mission619",  
        "translateBriefingShort619", timeFlagWinter, -316, 0, 20, 20, 20, mis620);  
        RegisterMission(mis620, "l620", "LC\\Missions\\Mission620",  
        "translateBriefingShort620", timeFlagWinter, 0, 90, 20, 20, 20);  
        RegisterMission(mis621, "l621", "LC\\Missions\\Mission621",  
        "translateBriefingShort621", timeFlagWinter, 0, 70, 20, 20, 20, mis612);  
        RegisterMission(mis622, "l622", "LC\\Missions\\Mission622",  
        "translateBriefingShort622", timeFlagWinter, -130, 50, 20, 20, mis613);  
  
        CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");  
        CreateGamePlayer(3,raceLC,playerLocal,"TestGameAI");  
  
        LoadBase(1,base3,3);  
        LoadBase(2,base2,1);  
        LoadBase(3,base1,1);  
    }
```

```
SetActivePlayerAndWorld(3,1)//player, world  
SetAvailableWorlds(2)///baza LC(swiat nr 1)
```

```
EnableMission(mis611,true);
```

```
SetSeason(8);
```

```
ActivateMissions(timeFlagWinter,true);
```

```
return Start,10;
```

```
}//-----
```

```
state Start
```

```
{  
    SetAvailableWorlds(1|2|4|8)//misja i baza LC(swiat nr 3)  
    EnableChooseMissionButton(true);  
    return Nothing,50;
```

```
}//-----
```

```
state Nothing
```

```
{
```

```
}//-----
```

```
event StartMission(int iMission)
```

```
{  
    EnableChooseMissionButton(false);
```

```
LoadMission(0,iMission);
```

```
EnableMission(iMission,false);
```

```
}
```

```
//-----
```

```
event EnableNextMission(int iMission,int iNextNr,int bEnable)
```

```
{  
    //if(iMission==mis613) return; //XXX DEMO  
    if(bEnable<2)  
        EnableMission(GetNextMission(iMission,iNextNr),bEnable);
```

```
//---gra przygaraana - jakis odpowiedni filmik---  
if(bEnable==2)  
    EndGame("Video\\Cutscene5.wd1");
```

```
//---gra wygrana - jakis odpowiedni filmik----  
if(bEnable==3)  
    EndGame("Video\\Cutscene6.wd1");
```

```
}
```

```
//-----
```

```
event EndMission(int iMission, int nResult) //
```

```
{  
    if(nResult==true)  
    {  
        SetMissionState(iMission,stateAccomplished);  
    }  
    else  
    {  
        SetMissionState(iMission,stateFailed);  
    }  
    EnableChooseMissionButton(true);  
}
```

```
*****
```

TutorialLC.ec

campaign "translateTutorialLC"

```
{  
    state Initialize;  
    state Nothing;  
  
    state Initialize  
    {  
        CreateGamePlayer(1,raceUCS,playerAI,"TestGameAI");  
        CreateGamePlayer(3,raceLC,playerLocal,"TestGameAI");  
  
        RegisterMission(0,"!TutorialLC","LC\\Missions\\TutorialLCmis","","0,0,0,0,0,0);  
        LoadMission(0,0);  
        SetAvailableWorlds(1);  
        SetActivePlayerAndWorld(3,0);  
        SetTime(100);  
        return Nothing;  
    }
```

```

state Nothing
{
    return Nothing;
}
}

Lcbase1Script.ec
mission "translateLCBase1"
{
    player p_Player;

    state Initialize;
    state Nothing;
    state EndGameState;

    state Initialize
    {
        p_Player=GetPlayer(3);
        return Nothing;
    }
    //-----
    state Nothing
    {
        if(p_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateCampaignLCBaseDestroyed",p_Player.GetName());
            return EndGameState,5;
        }
        return Nothing,50;
    }
    //-----
    state EndGameState
    {
        EnableNextMission(0,2); //end campaign
        return EndGameState,600;
    }
}

Lcbase2Script.ec
mission "translateLCBase2"
{
    consts
    {
        scriptFieldMoney=9;
    }

    player p_Enemy;
    player p_Player;

    state Initialize;
    state Nothing;
    state EndGameState;

    state Initialize
    {
        p_Player = GetPlayer(3);
        p_Enemy = GetPlayer(1);
        p_Enemy.SetMoney(10000);

        p_Enemy.EnableAIFeatures(aiBuildTanks,false);
        p_Enemy.EnableAIFeatures(aiBuildShips,false);
        p_Enemy.EnableAIFeatures(aiBuildHelicopters,false);
        p_Enemy.EnableAIFeatures(aiBuildSpecialUnits,false);

        p_Enemy.EnableResearch("RES_UCS_WMR1",false); //615
        p_Enemy.EnableResearch("RES_UCS_WAMR1",false); //615
        p_Enemy.EnableResearch("RES_UCS_WHPI",false); //615
        p_Enemy.EnableResearch("RES_UCS_UHL1",false); //615
        p_Enemy.EnableResearch("RES_UCS_BOMBER2",false); //615
        p_Enemy.EnableResearch("RES_UCS_BOMBER3",false); //615

        p_Enemy.EnableResearch("RES_UCS邬BL1",false); //617

        p_Enemy.EnableResearch("RES_UCS_WAPB1",true); //618
        p_Enemy.EnableResearch("RES_UCS_MB2",true); //618

        p_Enemy.EnableResearch("RES_UCS邬MI1",false); //Miner
        p_Enemy.EnableResearch("RES_UCS_USSI1",false); //Shark
        p_Enemy.EnableResearch("RES_UCS_UBSI1",false); //Hydra
        p_Enemy.EnableResearch("RES_UCS_UJUT",false); //Unit Transporter
        p_Enemy.EnableResearch("RES_UCS_PC",false); //Stacjonarne dzia³o plaz-
        p_Enemy.EnableResearch("RES_UCS_WSD",false); //Laser antykatetowy
    }
}

p_Player.EnableBuilding("LCBRC",false);

return Nothing;
}
//-----
state Nothing
{
    if(p_Player.GetScriptData(scriptFieldMoney)) //Chash from missions after the
    end.
    {
        p_Player.AddMoney(p_Player.GetScriptData(scriptFieldMoney));
        p_Player.SetScriptData(scriptFieldMoney,0);
    }

    if(p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateCampaignLCBaseDestroyed",p_Player.GetName());
        return EndGameState,5;
    }
    return Nothing,50;
}
//-----
state EndGameState
{
    EnableNextMission(0,2); //end campaign
    return EndGameState,600;
}

Lcbase3Script.ec
mission "translateLCBase3"
{
    player p_Player;

    int i;

    state Initialize;
    state Nothing;
    state ShowBriefing;
    state EndGameState;

    state Initialize
    {
        unitex uHero;
        //----- Goals -----
        RegisterGoal(0,"translateGoalBuildALPHA");
        RegisterGoal(1,"translateGoalBuildBETA");
        RegisterGoal(2,"translateGoalBuildGAMMA");
        RegisterGoal(3,"translateGoalBuildDELTA");
        RegisterGoal(4,"translateGoalBuildControlStation");

        EnableGoal(0,true);
        EnableGoal(1,true);
        EnableGoal(2,true);
        EnableGoal(3,true);
        EnableGoal(4,true);

        //----- Players -----
        p_Player = GetPlayer(3);
        //----- AI -----
        p_Player.SetMilitaryUnitsLimit(50000);

        //----- Money -----
        p_Player.SetMoney(0);

        //----- Buildings -----
        p_Player.EnableBuilding("LCBSR",false); //Centrum wydobywczo-transporto-
        we

        //----- Research -----
        p_Player.AddResearch("RES_MISSION_PACK1_ONLY");

        //----- Researches -----
        p_Player.EnableResearch("RES_LC_SGen",false); //611
        p_Player.EnableResearch("RES_LC_UМО2",false); //611
        p_Player.EnableResearch("RES_LCUF1",false); //611 add
        p_Player.EnableResearch("RES_LCUF2",false); //611
        p_Player.EnableResearch("RES_LC_WSL1",false); //611

        p_Player.EnableResearch("RES_LCAA1",false); //612 add
        p_Player.EnableResearch("RES_LCUSF1",false); //612 add

        p_Player.EnableResearch("RES_LCBPP2",false); //613 add w drugiej
        po³owie misji
        p_Player.EnableResearch("RES_LCBNE",false); //613
    }
}

```

```

p_Player.EnableResearch("RES_LC_BMD",false); //613
p_Player.EnableResearch("RES_LC_MGen",false); //613
p_Player.EnableResearch("RES_LC_SOBI",false); //613
p_Player.EnableResearch("RES_LC_REGI",false); //613
p_Player.EnableResearch("RES_LC_UUME",false); //613//612

p_Player.EnableResearch("RES_LC_WMR1",false); //615
p_Player.EnableResearch("RES_LC_UCR1",false); //615
p_Player.EnableResearch("RES_LC_HGen",false); //615
p_Player.EnableResearch("RES_LC_WH1",false); //615
p_Player.EnableResearch("RES_LC_MM2",false); //615

p_Player.EnableResearch("RES_LC_UUH",false); //616 add

p_Player.EnableResearch("RES_LC_UB01",false); //617
p_Player.EnableResearch("RES_LC_AMR1",false); //617
p_Player.EnableResearch("RES_LC_WHL1",false); //617

p_Player.EnableResearch("RES_LC_ART",false); //618
p_Player.EnableResearch("RES_LC_BWC",false); //618

p_Player.EnableResearch("RES_LC_WARTILLERY",false); //619
p_Player.EnableResearch("RES_LC_UCU1",false); //619

p_Player.EnableResearch("RES_LCUU",false); //621 add

p_Player.EnableResearch("RES_LC_SHR1",false); //622

p_Player.EnableResearch("RES_LC_WSS1",false);
p_Player.EnableResearch("RES_LC_WH51",false);
p_Player.EnableResearch("RES_LC_WAS1",false);
p_Player.EnableResearch("RES_LC_SDIF",false); //Laser antyrakitowy
p_Player.EnableBuilding("LCBSD",false); //Obrona antyrakitowa
p_Player.EnableBuilding("LCBSR",false);
p_Player.EnableBuilding("LCBRC",false);

//--- Variables for campaign goals ---
p_Player.SetScriptData(0,0);
p_Player.SetScriptData(1,0);
p_Player.SetScriptData(2,0);
p_Player.SetScriptData(3,0);
p_Player.SetScriptData(4,0);

//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0,4
5,0);
return ShowBriefing,1;
}
//-----
state ShowBriefing
{
AddBriefing("translateStartCampaignLC1MP01",p_Player.GetName());
return Nothing,50;
}

//-----
state Nothing
{
//---checking campaign goals---
for(i=0; i<5; i=i+1)
{
if(p_Player.GetScriptData(i)==2)
{
SetGoalState(i, goalAchieved);
}
}
if((p_Player.GetNumberOfBuildings())
{
AddBriefing("translateCampaignLCBaseDestroyed",p_Player.GetName());
return EndGameState,5;
}
return Nothing,50;
}
//-----
state EndGameState
{
EnableNextMission(0,2); //end campaign
return EndGameState,600;
}
//-----
```

mission611.ec

// ucs wyladowa³ na ksiezyku. Uzywa specjalistycznych pojazdów obardzo duzej sie rzenia.
//zaatakowali nasze centrum badania nowych technologii.

// fat girl jest dopiero prototypem ale musisz go uzyc aby zniszczyc wroga.
//Pojazdy wroga zosta³y rozdzielone wiec spróbuj je zniszczyc pojedynczo.

mission "translateMission611"

```

{
const
{
scriptFieldMoney=9;
goalDestroyUCSUnits = 0;
goalDestroyUCSBuildings = 1;
}

int bCheckEndMission;
player p_Enemy;
player p_Player;

state Initialize;
state ShowBriefing;
state EndMissionFailed;
state Working;
state Nothing;
//-----
state Initialize
{
//----- Goals -----
RegisterGoal(goalDestroyUCSUnits,"translateGoal611a");
RegisterGoal(goalDestroyUCSBuildings,"translateGoal611b");

EnableGoal(goalDestroyUCSUnits,true);
EnableGoal(goalDestroyUCSBuildings,true);

//----- Players -----
p_Player = GetPlayer(3);
p_Enemy = GetPlayer(1);

//----- AI -----
if(GetDifficultyLevel()==0)
p_Enemy.LoadScript("single\singleEasy");
if(GetDifficultyLevel()==1)
p_Enemy.LoadScript("single\singleMedium");
if(GetDifficultyLevel()==2)
{
p_Enemy.LoadScript("single\singleHard");
p_Enemy.SetNumberOffensiveTankPlatoons(2);
p_Enemy.SetNumberOffensiveShipPlatoons(0);
p_Enemy.SetNumberOffensiveHelicopterPlatoons(2);
}
//----- Money -----
if(GetDifficultyLevel()==0)
{
p_Player.SetMoney(30000);
p_Enemy.SetMoney(10000);
}
if(GetDifficultyLevel()==1)
{
p_Player.SetMoney(20000);
p_Enemy.SetMoney(20000);
}
if(GetDifficultyLevel()==2)
{
p_Player.SetMoney(10000);
p_Enemy.SetMoney(30000);
}

if(GetDifficultyLevel()==0)
{
KillArea(2,GetPointX(0), GetPointY(0), 0,1);
KillArea(2,GetPointX(1), GetPointY(1), 0,1);
}
if(GetDifficultyLevel()==1)
{
KillArea(2,GetPointX(1), GetPointY(1), 0,1);
}

bCheckEndMission=false;
```

```

p_Player.EnableResearch("RES_LC_SGen",true);/:/611
p_Player.EnableResearch("RES_LC_LM02",true);/:/611
p_Player.EnableResearch("RES_LC_WSL1",true);/:/611

//----- Buildings -----
p_Player.EnableBuilding("LCBPP2",false);
p_Player.EnableBuilding("LCBHQ",false);
p_Player.EnableBuilding("LCBRC",false);
p_Player.EnableBuilding("LCBEN1",false);
p_Player.EnableBuilding("LCBSR",false);
p_Player.EnableBuilding("LCBSD",false);
p_Player.EnableBuilding("LCBSS",false);
p_Player.EnableBuilding("LCBART",false);

p_Player.AddResearch("RES_LCUFG1");
p_Player.AddResearch("RES_LC_WSL1");

p_Enemy.AddResearch("RES_MISSION_PACK1_ONLY");
p_Enemy.AddResearch("RES_MSR2");
p_Enemy.AddResearch("RES_MSR3");
p_Enemy.AddResearch("RES_MSR4");

p_Player.EnableResearch("RES_LCUFG1",true);
p_Player.EnableResearch("RES_LC_WSL1",true);

p_Enemy.EnableResearch("RES_MSR2",true);
p_Enemy.EnableResearch("RES_MSR3",true);
p_Enemy.EnableResearch("RES_MSR4",true);

SetTimer(0,100);
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0,4
5,0);
return ShowBriefing,1;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing611a", p_Player.GetName());
    if(GetDifficultyLevel()==0)
        MeteorRain(GetPointX(2), GetPointY(2),40,100,30000,100,1,1);
    if(GetDifficultyLevel()==1)
        MeteorRain(GetPointX(2), GetPointY(2),40,100,30000,100,3,2);
    if(GetDifficultyLevel()==2)
        MeteorRain(GetPointX(2), GetPointY(2),40,100,30000,100,6,3);
    return Working, 20;
}
//-----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}
//-----
state Working
{
    //--- all goals achieved ---
    if(GetGoalState(goalDestroyJCSUnits)==goalAchieved &&
       GetGoalState(goalDestroyJCSBuildings)==goalAchieved)
    {
        EnableNextMission(0,true);
        EnableNextMission(1,true);
        p_Player.AddResearch("RES_LCUFG1");/:/611 add
        p_Player.EnableResearch("RES_LCUFG2",true);/:/611
        AddBriefing("translateAccomplished611", p_Player.GetName());
        EnableEndMissionButton(true);
        return Nothing;
    }
}
//-----
state Nothing
{
    return Nothing, 500;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    if(!bCheckEndMission) return;
    bCheckEndMission=false;
}

if(p_Player.GetNumberOfUnits() && p_Player.GetNumberOfBuildings())
{
    AddBriefing("translateFailed611", p_Player.GetName());
    state EndMissionFailed;
}

if(GetGoalState(goalDestroyJCSUnits)==goalAchieved &&
   lp_Enemy.GetNumberOfUnits())
{
    SetGoalState(goalDestroyJCSUnits, goalAchieved);
    AddBriefing("translateBriefing611b", p_Player.GetName());
}

if(GetGoalState(goalDestroyJCSBuildings)==goalAchieved &&
   lp_Enemy.GetNumberOfBuildings())
{
    SetGoalState(goalDestroyJCSBuildings, goalAchieved);
    AddBriefing("translateBriefing611c", p_Player.GetName());
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    if(GetGoalState(goalDestroyJCSUnits)==goalAchieved &&
       GetGoalState(goalDestroyJCSBuildings)==goalAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission612.ec
mission "translateMission612"
{()

const
{
    scriptFieldMoney=9;
    goalDestroyEnemy1 = 0;
    goalDestroyEnemy2 = 1;
    goalFangHaveToSurvive = 2;
}

player p_TmpPlayer;
player p_Player;
player p_Enemy1;
player p_Enemy2;
player p_Neutral;

int nMoney;
int iCheckEndMission;
int iMaxMeteorCounter;
int iMeteorCounter;
int iMeteorX;
int iMeteorY;
int b_FireMetors;
int iMeteorIntensity;
int iMeteorPower;
int iMeteorRange;

unitex uFang;

state Initialize;
state ShowBriefing;
state Working;
state MissionFailed;
state Nothing;

//-----
state Initialize

```

```

{ //----- Goals -----
RegisterGoal(goalDestroyEnemy1, "translateGoal612a");
RegisterGoal(goalDestroyEnemy2, "translateGoal612b");
RegisterGoal(goalFangHaveToSurvive, "translateGoal612c");

//---Show goals on list---
EnableGoal(goalDestroyEnemy1, true);
EnableGoal(goalDestroyEnemy2, true);
EnableGoal(goalFangHaveToSurvive, true);

//----- Players -----
p_Player = GetPlayer(3); //LC

p_Enemy1 = GetPlayer(1); //UCS
p_Enemy2 = GetPlayer(5); //UCS

p_Neutral = GetPlayer(7); //UCS
p_Neutral.EnableStatistics(false);
p_Neutral.SetAlly(p_Enemy1);
p_Neutral.SetAlly(p_Enemy2);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);

//----- AI -----
if(GetDifficultyLevel() == 0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
}

if(GetDifficultyLevel() == 1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
}

if(GetDifficultyLevel() == 2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
}

//wyłącz inteligencje graczy
p_Enemy1.EnableAIFeatures(aiControlOffense, false);
p_Enemy1.EnableAIFeatures(aiBuildBuildings, false);
p_Enemy1.EnableAIFeatures(aiBuildBuilders, false);

p_Enemy2.EnableAIFeatures(aiControlOffense, false);
p_Enemy2.EnableAIFeatures(aiBuildBuildings, false);
p_Enemy2.EnableAIFeatures(aiBuildBuilders, false);

p_Enemy1.SetAlly(p_Enemy2);

//----- Money -----
p_Player.SetMoney(0);
p_Enemy1.SetMoney(20000);
p_Enemy2.SetMoney(20000);

//----- Buildings -----
p_Player.EnableBuilding("LCBPF", false);
p_Player.EnableBuilding("LCBPP", false);
p_Player.EnableBuilding("LCBP2", false);
p_Player.EnableBuilding("LCBNE", false);
p_Player.EnableBuilding("LCBSB", false);
p_Player.EnableBuilding("LCBBA", false);
p_Player.EnableBuilding("LCBMR", false);
p_Player.EnableBuilding("LCBSR", false);
p_Player.EnableBuilding("LCBC", false);
p_Player.EnableBuilding("LCBAP", false);
p_Player.EnableBuilding("LCBGA", false);
p_Player.EnableBuilding("LCBDE", false);
p_Player.EnableBuilding("LCBHQ", false);
p_Player.EnableBuilding("LCBSD", false);
p_Player.EnableBuilding("LCBWC", false);
p_Player.EnableBuilding("LCBSS", false);
p_Player.EnableBuilding("LCBLZ", false);

p_Player.EnableBuilding("LCBUC", false);
p_Player.EnableBuilding("LCBN1", false);
p_Player.EnableBuilding("LCLASERWALL", false);
p_Player.EnableCommand(commandSoldBuilding, true);

p_Player.EnableResearch("RES_LC_UIME1", true); //613/612
p_Player.EnableResearch("RES_LCAA1", true); //612
p_Player.EnableResearch("RES_LCUSF1", true); //612
p_Player.AddResearch("RES_LCAA1"); //612 add
p_Player.AddResearch("RES_LCUSF1"); //612 add
//----- Timers -----
SetTimer(0, 20);

//----- Variables -----
bCheckEndMission = false;
if(GetDifficultyLevel() == 0)
{
    nMaxMeteorCounter = 60 * 1;
    nMeteorIntensity = 3;
    nMeteorPower = 10;
    nMeteorRange = 12;
}
if(GetDifficultyLevel() == 1)
{
    nMaxMeteorCounter = 60 * 2;
    nMeteorIntensity = 2;
    nMeteorPower = 7;
    nMeteorRange = 10;
}
if(GetDifficultyLevel() == 2)
{
    nMaxMeteorCounter = 60 * 3;
    nMeteorIntensity = 2;
    nMeteorPower = 6;
    nMeteorRange = 7;
}
nMeteorCounter = nMaxMeteorCounter;
b_FireMeteors = true;
uFang = GetUnit((GetPointX(12), GetPointY(12)), 0);
p_Player.AddUnitToSpecialTab(uFang, true, -1);
//--- Creating & destroying additional units ---
if(GetDifficultyLevel() == 0)
{
    KillArea(p_Enemy1.GetIFF(), GetPointX(0), GetPointY(0), 0, 1);
    KillArea(p_Enemy1.GetIFF(), GetPointX(6), GetPointY(6), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(8), GetPointY(8), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(10), GetPointY(10), 0, 1);
}
if(GetDifficultyLevel() == 1)
{
    KillArea(p_Enemy1.GetIFF(), GetPointX(6), GetPointY(6), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(10), GetPointY(10), 0, 1);
}
p_Enemy1.CreateUnitEx((GetPointX(1), GetPointY(1), 0, null, "UCSUM1", "UCWSMSR1"), null, null, null); // Spider + hR
p_Enemy1.CreateUnitEx((GetPointX(2), GetPointY(2), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(3), GetPointY(3), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(4), GetPointY(4), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(5), GetPointY(5), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy2.CreateUnitEx((GetPointX(9), GetPointY(9), 0, null, "UCSUSL1", "UCWSMSR1"), null, null, null); // Spider + hR
p_Enemy1.CreateUnitEx((GetPointX(2), GetPointY(2), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(3), GetPointY(3), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(4), GetPointY(4), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(5), GetPointY(5), 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(2), GetPointY(2) + 1, 0, null, "UCSUSL1", "UCWSMSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(3), GetPointY(3) + 1, 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R
p_Enemy1.CreateUnitEx((GetPointX(4), GetPointY(4) + 1, 0, null, "UCSUSL1", "UCSWTSR1"), null, null, null); // Tiger + R

```

```

    p_Enemy1.CreateUnitEx(GetPointX(5), GetPointY(5) + 1, 0, null,
    "UCSUSL1", "UCSWTSR1", null, null, null); // Tiger + R

    p_Enemy2.CreateUnitEx(GetPointX(9), GetPointY(9), 0, null, "UCSUML1",
    "UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy2.CreateUnitEx(GetPointX(11), GetPointY(11), 0, null,
    "UCSUML1", "UCSWSMR1", null, null, null); // Spider + hR
}

//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 12, 0, 4.5, 0);
return ShowBriefing, 100;
}

//----- ShowBriefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing612", p_Player.GetName());
    return Working, 100;
}

//----- state Working -----
state Working
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(!uFang.IsLive())
        {
            SetGoalState(goalFangHaveToSurvive,goalFailed);
            AddBriefing("translateFailed612", p_Player.GetName());
            return MissionFailed, 20;
        }

        if(p_Enemy1.GetNumberOfBuildings())
        {
            SetGoalState(goalDestroyEnemy1,goalAchieved);
        }
        if(p_Enemy2.GetNumberOfBuildings())
        {
            SetGoalState(goalDestroyEnemy2,goalAchieved);
        }
    }

    if(GetGoalState(goalDestroyEnemy1) == goalAchieved &
    GetGoalState(goalDestroyEnemy2) == goalAchieved)
    {
        b_FireMeteors=false;
        SetGoalState(goalFangHaveToSurvive,goalAchieved);
        SetConsoleText("");
        AddBriefing("translateAccomplished612", p_Player.GetName());
        EnableNextMission(0,true);
        EnableEndMissionButton(true);
        return Nothing;
    }
    return Working, 100;
}

//----- state MissionFailed -----
state MissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//----- state Nothing -----
state Nothing
{
    if(!uFang.IsLive())
    {
        SetGoalState(goalFangHaveToSurvive,goalFailed);
        AddBriefing("translateFailed612", p_Player.GetName());
        return MissionFailed, 20;
    }
    return Nothing, 100;
}

//----- event UnitDestroyed(unit u_Unit) -----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

//----- event BuildingDestroyed(unit u_Unit) -----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//----- event Timer0() //wolany co sekunde -----
event Timer0() //wolany co sekunde
{
    if(b_FireMeteors)
    {
        nMeteorCounter=nMeteorCounter-1;
        SetConsoleText("translateMessage612a",nMeteorCounter);
        if(!nMeteorCounter)
        {
            SetConsoleText("translateMessage612b");//FIRE!!!! blinking
            nMeteorCounter=nMaxMeteorCounter;
            if(uFang.IsLive() & uFang.IsInWorld(GetWorldNum()) &&
            uFang.GetLocationZ()==0)
            {
                nMeteorX=uFang.GetLocationX();
                nMeteorY=uFang.GetLocationY();
            }
        }
    }
}

//----- event MeteorRain(nMeteorX,nMeteorY,nMeteorRange,500,1000,500,nMeteorIntensity,nMeteorPower) -----
event MeteorRain(nMeteorX,nMeteorY,nMeteorRange,500,1000,500,nMeteorIntensity,nMeteorPower)
{
}

//----- event EndMission() -----
event EndMission()
{
    if(GetGoalState(goalDestroyEnemy1) == goalAchieved &&
    GetGoalState(goalDestroyEnemy2) == goalAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)+p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission613.ec
mission "translateMission613"
{
    consts
    {
        scriptFieldMoney=9;
        goalDestroyUCSNorth = 0;
        goalDestroyUCSSouth = 1;
        goalBuildNorth = 2;
        goalBuildSouth = 3;
        FirstOffenseAfter = 15; // minut
        OffenseFrequency = 3; // minut
        OffenseTime = 30; // sekundy
        FirstOffensePlayer = 1;
        LastOffensePlayer = 5;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Player;
    player p_Neutral;

    int nTimer0Counter;
    int nTimer1Counter;
    int nOffensePlayerNumber;
    player p_OffensePlayer;

    int DestroyUCS1Achieved;
    int DestroyUCS2Achieved;
    int BuildKartilleryAndPPlantsAchieved;

    int BriefingBShowed;
    int CheckBase;

    int BuildRP1;
    int BuildRP2;

    int MinX1;
}

```

```

int MinY1;
int MaxX1;
int MaxY1;

int MinX2;
int MinY2;
int MaxX2;
int MaxY2;

int BuildingType;
int IPP1;
int IPP2;
int IR1;
int IR2;
unitex CurUnit;
unitex FirstUnit;
unitex FirstUnitUCS2;

int i;
int x;
int y;
int UCS1IsMining;
int UCS2IsMining;

int bMiningChecked;

int MinStateOfBuildings1;
int MinStateOfBuildings2;

int CurrentStateOfBuildings1;
int CurrentStateOfBuildings2;

state Initialize;
state ShowBriefing;
state Working;
state EndMissionFailed;
state Nothing;
//-----
state Initialize
{
    //----- Timers -----
    SetTimer(0, 1200);
    SetTimer(1, 20);
    SetTimer(2, 151);
    //----- Variables -----
    nTimer0Counter = FirstOffenseAfter;
    nTimer1Counter = 0;
    nOffensePlayerNumber = FirstOffensePlayer - 1;

    BriefingShowed = true;
    DestroyUCS1Achieved = false;
    DestroyUCS2Achieved = false;
    BuildRefineryAndPPPlantsAchieved = false;
    BuildRP1 = false;
    BuildRP2 = false;
    CheckBase = 0;
    UCS1sMining = true;
    UCS2sMining = true;
    IPP1 = 0;
    IPP2 = 0;
    IR1 = 0;
    IR2 = 0;

    MinX1 = GetPointX(0);
    MinY1 = GetPointY(0);
    MaxX1 = GetPointX(1);
    MaxY1 = GetPointY(1);

    MinX2 = GetPointX(5);
    MinY2 = GetPointY(5);
    MaxX2 = GetPointX(6);
    MaxY2 = GetPointY(6);

    //----- Goals -----
    RegisterGoal(goalDestroyUCSNorth, "translateGoal613A");
    RegisterGoal(goalDestroyUCSSouth, "translateGoal613B");

    RegisterGoal(goalBuildNorth, "translateGoal613C");
    RegisterGoal(goalBuildSouth, "translateGoal613D");

    EnableGoal(goalDestroyUCSNorth, true);
    EnableGoal(goalDestroyUCSSouth, true);

    //----- Players -----
}

p_Player = GetPlayer(3);
p_Enemy1 = GetPlayer(1);
p_Enemy2 = GetPlayer(5);

p_Neutral = GetPlayer(7); //UCS
p_Neutral.EnableStatistics(false);
p_Neutral.SetAll(p_Enemy1);
p_Neutral.SetAll(p_Enemy2);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);

//----- AI -----
if(GetDifficultyLevel() == 0)
{
    p_Enemy1.LoadScript("single\\singleEasy");
    p_Enemy2.LoadScript("single\\singleEasy");
}

if(GetDifficultyLevel() == 1)
{
    p_Enemy1.LoadScript("single\\singleMedium");
    p_Enemy2.LoadScript("single\\singleMedium");
}

if(GetDifficultyLevel() == 2)
{
    p_Enemy1.LoadScript("single\\singleHard");
    p_Enemy2.LoadScript("single\\singleHard");
}

p_Enemy1.EnableAIFeatures(aiControlOffense, false);
p_Enemy1.EnableAIFeatures(aiControlDefense, false);

p_Enemy2.EnableAIFeatures(aiControlOffense, false);
p_Enemy2.EnableAIFeatures(aiControlDefense, false);

//----- Money -----
p_Player.EnableCommand(commandSoldBuilding, true); // 1st
tab p_Player.SetMoney(10000);
p_Player.AddResearch("RES_MISSION_PACK1_ONLY");

p_Enemy1.AddResearch("RES_MISSION_PACK1_ONLY");
p_Enemy2.AddResearch("RES_MISSION_PACK1_ONLY");

p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(10000);

//-----easy-----
if(GetDifficultyLevel() == 0)
{
    //Enemy1
    KillArea(p_Enemy1.GetIFF(), GetPointX(2), GetPointY(2), 0, 1);
    KillArea(p_Enemy1.GetIFF(), GetPointX(10), GetPointY(10), 0, 1);

    //Enemy2
    KillArea(p_Enemy2.GetIFF(), GetPointX(7), GetPointY(7), 0, 1);
}

//-----Normal-----
if(GetDifficultyLevel() == 1)
{
    //Enemy1
    KillArea(p_Enemy1.GetIFF(), GetPointX(10), GetPointY(10), 0, 1);

    //-----add to p_Enemy-----
    /Spider+R : UCSSUM1 # UCSWSSR1
    p_Enemy1.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "UCSUM1",
    "UCSWSSR1", null, null, null);
    p_Enemy1.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "UCSUM1",
    "UCSWSSR1", null, null, null);

    //Enemy2
    /Spider+R : UCSSUM1 # UCSWSSR1
    p_Enemy2.CreateUnitEx(GetPointX(8), GetPointY(8), 0, null, "UCSUM1",
    "UCSWSSR1", null, null, null);
    p_Enemy2.CreateUnitEx(GetPointX(9), GetPointY(9), 0, null, "UCSUM1",
    "UCSWSSR1", null, null, null);
}

```

```

        }

        //----Hard-----
        if(GetDifficultyLevel() == 2)
        {
            //----add to p_Enemy-----
            //Enemy1
            //Panther + R: UCSUHL1 # UCSWBSR1
            p_Enemy1.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);
            p_Enemy1.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);

            //Enemy2
            //Panther + R: UCSUHL1 # UCSWBSR1
            p_Enemy2.CreateUnitEx(GetPointX(8), GetPointY(8), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);
            p_Enemy2.CreateUnitEx(GetPointX(9), GetPointY(9), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);
        }

        //----- Buildings -----
        p_Player.EnableBuilding("LCBSR",false); //Centrum wydobyczo-transportowe
        p_Player.EnableBuilding("LCBRC",false);

        p_Enemy2.SetAlly(p_Enemy1);
        p_Enemy2.CopyResearches(p_Enemy1);

        p_Player.EnableResearch("RES_LCBNE",true);//613
        p_Player.EnableResearch("RES_LC_BMD",true);//613
        p_Player.EnableResearch("RES_LC_MGen",true);//613
        p_Player.EnableResearch("RES_LC_SOBI",true);//613
        p_Player.EnableResearch("RES_LC_REG1",true);//613
        p_Player.EnableResearch("RES_LC_UIME1",true);//613

        MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
        MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;

        //----- Camera -----
        CallCamera();
    }

    p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0,4
5,0);
    return ShowBriefing,60;
}

//-----state ShowBriefing
{
    AddBriefing("translateBriefing613a", p_Player.GetName());
    BriefingShowed = false;
    return Working, 20;
}
//-----state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}
//-----state Working
{
    //-----mission failed-----
    if(p_Player.GetNumberOfUnits() && !p_Player.GetNumberOfBuildings() )
    {
        AddBriefing("translateFailed613", p_Player.GetName());
        return EndMissionFailed;
    }

    //--- check 1 goal - destroy UCS ---
    if(!DestroyUCS1Achieved)
    {
        CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();
        if(CurrentStateOfBuildings1 < MinStateOfBuildings1)
        {
            p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy1.SetMoney(0);

            if(!CurrentStateOfBuildings1)
            {
                SetGoalState(goalDestroyUCSNorth, goalAchieved);
            }
        }
    }

    DestroyUCS1Achieved = true;
    p_Player.EnableResearch("RES_LCBPP2",true);
    p_Player.AddResearch("RES_LCBPP2");//613 add w drugiej
    po3owie misji
    }
}

if(!DestroyUCS2Achieved)
{
    CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();

    if(CurrentStateOfBuildings2 < MinStateOfBuildings2)
    {
        p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy2.SetMoney(0);

        if(!CurrentStateOfBuildings2)
        {
            SetGoalState(goalDestroyUCSSouth, goalAchieved);
            DestroyUCS2Achieved = true;
            p_Player.EnableResearch("RES_LCBPP2",true);
            p_Player.AddResearch("RES_LCBPP2");//613 add w drugiej
            po3owie misji
        }
    }
}

//--- all goals achieved ---
if(BuildRP1 && BuildRP2 && DestroyUCS1Achieved &&
DestroyUCS2Achieved)
{
    p_Player.SetScriptData(0,2);

    EnableNextMission(0,true);
    AddBriefing("translateAccomplished613", p_Player.GetName());
    EnableEndMissionButton(true);
    return Nothing;
}
//-----state Nothing
{
    return Nothing, 500;
}
//-----event BuildingDestroyed(unit u_Unit)
{
    //czy zniszczono budynek wroga??
    if(!BriefingBShowed)
    {
        if(u_Unit.IsBuilding())
        {
            if(u_Unit.GetFFNumber() == p_Enemy1.GetFFNumber() ||

u_Unit.GetFFNumber() == p_Enemy2.GetFFNumber())
            {
                EnableGoal(goalBuildNorth,true);
                EnableGoal(goalBuildSouth,true);
                AddBriefing("translateBriefing613b", p_Player.GetName());
                BriefingBShowed = true;
            }
        }
    }
}

//-----event Timer0() //wolny co minute
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofensyw
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber >= LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        if(nOffensePlayerNumber == 2)
            nOffensePlayerNumber = 5;

        p_OffensePlayer = GetPlayer(nOffensePlayerNumber);

        if(p_OffensePlayer.GetNumberOfUnits())
        {

```

```

    p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
    p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);

    nTimerOCounter = OffenseFrequency;
    nTimer1Counter = OffenseTime;

    if(!p_OffensePlayer.GetNumberOfUnits())
    {
        nTimerOCounter = 1;
    }
}

//-----
event Timer1() //wolny co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofenswy
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, false);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, false);
        }
    }
}

//-----check UCS bases-----
event Timer2()
{
    //sprawdzanie "na zmiane"
    if(CheckBase == 0)
    {
        CheckBase = 1;
        bMiningChecked = false;

        //czy wybudowano w bazie 1
        if(!BuildRP1)
        {
            //TraceD("1:");
            TraceD(MinX1);TraceD(":");TraceD(MinY1);TraceD(":");

            //TraceD(MaxX1);TraceD(":");TraceD(MaxY1);TraceD(":");

            bMiningChecked = true;

            for(y=MinY1; y<MaxY1+1; y=y+1)
                for(x=MinX1; x<MaxX1+1; x=x+1)
                {
                    CurrUnit = GetUnit(x, y, 0);
                    if(CurrUnit != null)
                    {
                        if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
                            CurrUnit.IsBuilding()) //czy nasz budynek
                        {
                            //TraceD("o");
                            BuildingType = CurrUnit.GetBuildingType();
                            //TraceD(BuildingType);

                            if(IPP1 < 2) //jeszcze nie wykryl elektrowni
                            {
                                //TraceD("<2");
                                if(BuildingType == buildingPowerPlant) //OK - wykryl
                                    elektrownie
                                    nia
                                    nia
                                    {
                                        //TraceD("!");
                                        if(IPP1 == 0) //to pierwsza kostka z elektrownia
                                        {
                                            IPP1 = 1;
                                            FirstUnit = CurrUnit;
                                            //TraceD(" Wybudowano elektr.1 w UCS1 ");
                                        }
                                        else //nie pierwsza kostka ale czy inna elektrownia
                                        {
                                            if(FirstUnit != CurrUnit) //inna
                                            {
                                                IPP1 = 2;
                                                //TraceD(" Wybudowano elektr.2 w UCS1 ");
                                            }
                                        }
                                    }
                                if(!IR1 && BuildingType == buildingMiningRefinery)
                                    {
                                        IR1 = 1; //wybudowano kopalnie
                                        //TraceD(" Wybudowano kopalnie w UCS1 ");
                                    }
                                if(IPR1 && IPP1 == 2)
                                {
                                    BuildRP1 = true;
                                    SetGoalState(goalBuildNorth, goalAchieved);
                                    //TraceD(" Wybudowano to co trzeba w UCS1 ");
                                }
                                else
                                {
                                    //TraceD(".");
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

//czy wybudowano w bazie 2
if(!BuildRP2)
{
    //TraceD("2:");
    TraceD(MinX2);TraceD(":");TraceD(MinY2);TraceD(":");

    //TraceD(MaxX2);TraceD(":");TraceD(MaxY2);TraceD(":");

    for(y=MinY2; y<MaxY2+1; y=y+1)
        for(x=MinX2; x<MaxX2+1; x=x+1)
        {
            CurrUnit = GetUnit(x, y, 0);
            if(CurrUnit != null)
            {
                if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
                    CurrUnit.IsBuilding()) //czy nasz budynek
                {
                    BuildingType = CurrUnit.GetBuildingType();
                    //TraceD(BuildingType);
                    if(IPP2 < 2) //jeszcze nie wykryl elektrowni
                    {
                        //TraceD("<2");
                        if(BuildingType == buildingPowerPlant) //OK - wykryl
                            elektrownie
                            nia
                            nia
                            {
                                if(IPP2 == 0) //to pierwsza kostka z elektrownia
                                {
                                    IPP2 = 1;
                                    FirstUnitUCS2 = CurrUnit;
                                    //TraceD(" Wybudowano elektr.1 w UCS2 ");
                                }
                                else //nie pierwsza kostka ale czy inna elektrownia
                                {
                                    if(FirstUnitUCS2 != CurrUnit) //inna
                                    {
                                        IPP2 = 2;
                                        //TraceD(" Wybudowano elektr.2 w UCS2 ");
                                    }
                                }
                            }
                        if(IR2 && BuildingType == buildingMiningRefinery)
                        {
                            IR2 = 1; //wybudowano kopalnie
                            //TraceD(" Wybudowano kopalnie w UCS2 ");
                        }
                    }
                    if(IPR2 && IPP2 == 2 && !BuildRP2)
                    {
                        BuildRP2 = true;
                        SetGoalState(goalBuildSouth, goalAchieved);
                        //TraceD(" Wybudowano to co trzeba w UCS2 ");
                    }
                    else
                    {
                        //TraceD(".");
                    }
                }
            }
        }
    }
}

```

```

        }
        //TraceD("      \n");
    }
}

//-----
event EndMission()
{
    if(BuildRP1 && BuildRP2 && DestroyUCS1Achieved &&
DestroyUCS2Achieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission614.ec
mission "translateMission614"
{
    consts
    {
        scriptFieldMoney=9;
        goalDestroyEnemy1 = 0;
        goalDestroyEnemy2 = 1;
        goalDestroyEnemy3 = 2;
        goalActivateUnitTransporters = 3;
        goalFangHaveToSurvive = 4;
    }

    player p_Player;
    player p_Enemy0;
    player p_Enemy1;
    player p_Enemy2;
    player p_Neutral;
    player p_TmpPlayer;

    unitex uFang;

    int bCheckEndMission;
    int bUnitTransportersActivated;

    state Initialize;
    state ShowBriefing;
    state Working;
    state MissionFailed;
    state Nothing;

    //-----
    state Initialize
    {
        //----- Goals -----
        RegisterGoal(goalDestroyEnemy1, "translateGoal614a");
        RegisterGoal(goalDestroyEnemy2, "translateGoal614b");
        RegisterGoal(goalDestroyEnemy3, "translateGoal614c");
        RegisterGoal(goalActivateUnitTransporters, "translateGoal614d");
        RegisterGoal(goalFangHaveToSurvive, "translateGoal614e");

        EnableGoal(goalDestroyEnemy1, true);
        EnableGoal(goalDestroyEnemy2, true);
        EnableGoal(goalDestroyEnemy3, true);
        EnableGoal(goalActivateUnitTransporters, true);
        EnableGoal(goalFangHaveToSurvive, true);

        //----- Players -----
        p_Player = GetPlayer(3); //LC

        p_Enemy0 = GetPlayer(0); //UCS
        p_Enemy1 = GetPlayer(1); //UCS
        p_Enemy2 = GetPlayer(5); //UCS

        p_TmpPlayer = GetPlayer(2);
        p_TmpPlayer.EnableStatistics(false);

        p_Neutral = GetPlayer(6); //UCS
        p_Neutral.EnableStatistics(false);
        p_Neutral.SetAlly(p_Enemy0);
    }

    //----- AI -----
    if(GetDifficultyLevel() == 0)
    {
        p_Neutral.SetAlly(p_Enemy2);
        p_Neutral.SetNeutral(p_Player);
        p_Player.SetNeutral(p_Neutral);
    }

    if(GetDifficultyLevel() == 1)
    {
        p_Enemy0.LoadScript("single\singleEasy");
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy0.SetName("translateAINameEasyUCS1");
        p_Enemy1.SetName("translateAINameEasyUCS2");
        p_Enemy2.SetName("translateAINameEasyUCS3");
    }

    if(GetDifficultyLevel() == 2)
    {
        p_Enemy0.LoadScript("single\singleMedium");
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy0.SetName("translateAINameMediumUCS1");
        p_Enemy1.SetName("translateAINameMediumUCS2");
        p_Enemy2.SetName("translateAINameMediumUCS3");
    }

    if(GetDifficultyLevel() == 3)
    {
        p_Enemy0.LoadScript("single\singleHard");
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy0.SetName("translateAINameHardUCS1");
        p_Enemy1.SetName("translateAINameHardUCS2");
        p_Enemy2.SetName("translateAINameHardUCS3");
    }

    //wyłącz inteligencje graczy
    p_Enemy0.EnableAIFeatures(aiEnabled,false);
    p_Enemy1.EnableAIFeatures(aiControlOffense,false);
    p_Enemy2.EnableAIFeatures(aiControlOffense,false);
    p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);

    p_Enemy1.SetAlly(p_Enemy2);
    p_Enemy1.SetAlly(p_Enemy0);

    p_Player.SetNeutral(p_Neutral);
    p_Enemy0.SetNeutral(p_Neutral);
    p_Enemy1.SetNeutral(p_Neutral);
    p_Enemy2.SetNeutral(p_Neutral);

    //----- Money -----
    p_Player.SetMoney(10000);
    p_Enemy0.SetMoney(0);
    p_Enemy1.SetMoney(10000);
    p_Enemy2.SetMoney(30000);
    p_Neutral.SetMoney(0);

    //----- Timers -----
    SetTimer(0, 40);
    //----- Variables -----
    bCheckEndMission = false;
    bUnitTransportersActivated = false;

    uFang = GetUnit(GetPointX(13),GetPointY(13),0);
    p_Player.AddUnitToSpecialTab(uFang,true,-1);

    p_Player.EnableBuilding("LCBLZ",false);
    p_Player.EnableBuilding("LCBRC",false);
    p_Player.EnableBuilding("LCBSR",false); //Centrum wydobyczo-transportowe
    p_Player.EnableCommand(commandSoldBuilding,true);
    p_Player.SetMaxDistance(20);
    //--- Creating & destroying additional units ----
    if(GetDifficultyLevel() == 0)
    {
        KillArea(p_Enemy1.GetFF(), GetPointX(2), GetPointY(2), 0, 1);
        KillArea(p_Enemy1.GetFF(), GetPointX(6), GetPointY(6), 0, 1);
        p_Player.CreateUnitEx(GetPointX(10), GetPointY(10), 0, null, "LCUR3",
"LCWHL2", null, null, null);
        p_Player.CreateUnitEx(GetPointX(11), GetPointY(11), 0, null, "LCUR3",
"LCWHL2", null, null, null);
        p_Player.CreateUnitEx(GetPointX(12), GetPointY(12), 0, null, "LCUR3",
"LCWHL2", null, null, null);
    }

    if(GetDifficultyLevel() == 1)

```

```

    {
        p_Player.CreateUnitEx(GetPointX(10), GetPointY(10), 0, null, "LCUCR3",
        "LCWHL2", null, null, null);

        KillArea(p_Enemy1.GetIFF(), GetPointX(6), GetPointY(6), 0, 1);

        p_Enemy1.CreateUnitEx(GetPointX(3) - 1, GetPointY(3), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy1.CreateUnitEx(GetPointX(3) + 1, GetPointY(3), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy1.CreateUnitEx(GetPointX(4) - 1, GetPointY(4), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy1.CreateUnitEx(GetPointX(4) + 1, GetPointY(4), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR

        p_Enemy2.CreateUnitEx(GetPointX(8) - 1, GetPointY(8), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy2.CreateUnitEx(GetPointX(8) + 1, GetPointY(8), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR

    }
    if(GetDifficultyLevel() == 2)
    {
        p_Enemy1.CreateUnitEx(GetPointX(3) - 1, GetPointY(3), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy1.CreateUnitEx(GetPointX(3) + 1, GetPointY(3), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy1.CreateUnitEx(GetPointX(4) - 1, GetPointY(4), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy1.CreateUnitEx(GetPointX(4) + 1, GetPointY(4), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR

        p_Enemy1.CreateUnitEx(GetPointX(5) - 1, GetPointY(5), 0, null,
        "UCSUL1", "UCSWBSR1", null, null, null); // Panther + R
        p_Enemy1.CreateUnitEx(GetPointX(5) + 1, GetPointY(5), 0, null,
        "UCSUL1", "UCSWBSR1", null, null, null); // Panther + R
        p_Enemy1.CreateUnitEx(GetPointX(7) - 1, GetPointY(7), 0, null,
        "UCSUL1", "UCSWBSR1", null, null, null); // Panther + R
        p_Enemy1.CreateUnitEx(GetPointX(7) + 1, GetPointY(7), 0, null,
        "UCSUL1", "UCSWBSR1", null, null, null); // Panther + R

        p_Enemy2.CreateUnitEx(GetPointX(8) - 1, GetPointY(8), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR
        p_Enemy2.CreateUnitEx(GetPointX(8) + 1, GetPointY(8), 0, null,
        "UCSUL1", "UCSWSMR1", null, null, null); // Spider + hR

        p_Enemy2.CreateUnitEx(GetPointX(8), GetPointY(8) - 1, 0, null,
        "UCSUL1", "UCSWBHP1", null, null, null); // Panther + hP
        p_Enemy2.CreateUnitEx(GetPointX(8), GetPointY(8) + 1, 0, null,
        "UCSUL1", "UCSWBHP1", null, null, null); // Panther + hP
    }

    //----- Camera -----
    CallCamera();

    p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 12, 0.4
    5.0);
    return ShowBriefing, 100;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing614a", p_Player.GetName());
    return Working, 100;
}

//-----
state Working
{
    unitex uTransporter;
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(uFang!=null && uFang.IsLive())
        {
            SetGoalState(goalFangHaveToSurvive, goalFailed);
            AddBriefing("translateFailed614", p_Player.GetName());
            return MissionFailed, 20;
        }

        if(GetGoalState(goalDestroyEnemy1)!= goalAchieved &&
        lp_Enemy0.GetNumberOfUnits())
            SetGoalState(goalDestroyEnemy1, goalAchieved);

        if(GetGoalState(goalDestroyEnemy2)!= goalAchieved &&
        lp_Enemy1.GetNumberOfBuildings())
    }

    SetGoalState(goalDestroyEnemy2, goalAchieved);
    if(GetGoalState(goalDestroyEnemy3)!= goalAchieved &&
    lp_Enemy2.GetNumberOfBuildings())
    {
        SetGoalState(goalDestroyEnemy3, goalAchieved);
    }

    if(!bUnitTransportersActivated &&
    uFang!=null && uFang.IsLive() && uFang.GetLocationZ()==0 &&
    uFang.DistanceTo(GetPointX(14), GetPointY(14))<5)
    {
        bUnitTransportersActivated=true;
        p_Neutral.GiveAllUnitsTo(p_Player);
    }

    uTransporter = GetUnit(GetPointX(14)-1,GetPointY(14),0);
    uTransporter.LoadSceneScript("Scripts\\Units\\Transporter.ecomp");
    uTransporter = GetUnit(GetPointX(14)+1,GetPointY(14),0);
    uTransporter.LoadSceneScript("Scripts\\Units\\Transporter.ecomp");
    uTransporter = GetUnit(GetPointX(14).GetPointY(14)+1,0);
    uTransporter.LoadSceneScript("Scripts\\Units\\Transporter.ecomp");

    SetGoalState(goalActivateUnitTransporters,goalAchieved);
    AddBriefing("translateBriefing614b", p_Player.GetName());
}

if(GetGoalState(goalDestroyEnemy1) == goalAchieved &&
GetGoalState(goalDestroyEnemy2) == goalAchieved &&
GetGoalState(goalDestroyEnemy3) == goalAchieved &&
GetGoalState(goalActivateUnitTransporters) == goalAchieved)
{
    SetGoalState(goalFangHaveToSurvive, goalAchieved);
    AddBriefing("translateAccomplished614", p_Player.GetName());
    EnableNextMission(0, true);
    EnableEndMissionButton(true);
    return Nothing;
}

return Working, 100;
}

//-----
state MissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state Nothing
{
    if(uFang==null && !uFang.IsLive())
    {
        SetGoalState(goalFangHaveToSurvive, goalFailed);
        AddBriefing("translateFailed614", p_Player.GetName());
        return MissionFailed, 20;
    }
    return Nothing, 500;
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event Timer0() //wolany co minute
{
}

//-----
event EndMission()
{
    if(GetGoalState(goalDestroyEnemy1)== goalAchieved &&
    GetGoalState(goalDestroyEnemy2)== goalAchieved &&
    GetGoalState(goalDestroyEnemy3)== goalAchieved &&
    GetGoalState(goalActivateUnitTransporters)== goalAchieved)
    {

        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
}

```

```
+p_Player.GetMoney());  
    p_Player.SetMoney(0);  
}  
}  
}
```

mission615.ec

```
mission "translateMission615"
```

```
{  
    consts  
    {  
        scriptFieldMoney=9;  
        goalDestroyUCSSouth = 0;  
        goalDestroyUCSSouthEast = 1;  
        goalDestroyUCSSouthWest = 2;  
        goalCollectMoney = 3;  
        goalBuildSouth = 4;  
        goalBuildSouthEast = 5;  
        goalBuildSouthWest = 6;  
    }
```

```
    NeededMoney = 40000;
```

```
    FirstOffenseAfter = 15; // minuty  
    OffenseFrequency = 3; // minuty  
    OffenseTime = 30; // sekundy  
    FirstOffensePlayer = 1;  
    LastOffensePlayer = 7;
```

```
    fieldsCoveredByPP2 = 17;  
}
```

```
player p_Enemy1;  
player p_Enemy2;  
player p_Enemy3;  
player p_Player;  
player p_Neutral;
```

```
int nTimer0Counter;  
int nTimer1Counter;  
int nOffensePlayerNumber;  
player p_OffensePlayer;
```

```
int MinStateOfBuildings1;  
int MinStateOfBuildings2;  
int MinStateOfBuildings3;
```

```
int CurrentStateOfBuildings1;  
int CurrentStateOfBuildings2;  
int CurrentStateOfBuildings3;
```

```
int DestroyUCS1Achieved;  
int DestroyUCS2Achieved;  
int DestroyUCS3Achieved;
```

```
int DestroyAllUCSAchieved;  
int CollectMoneyAchieved;
```

```
int CheckBase;  
int BuildAllRP;
```

```
int BuildRP1;  
int BuildRP2;  
int BuildRP3;
```

```
int MinX1;  
int MinY1;  
int MaxX1;  
int MaxY1;
```

```
int MinX2;  
int MinY2;  
int MaxX2;  
int MaxY2;
```

```
int MinX3;  
int MinY3;  
int MaxX3;  
int MaxY3;
```

```
int BuildingType;  
int IPP1;  
int IPP2;  
int IPP3;
```

```
int IR1;  
int IR2;  
int IR3;  
int nMoney;
```

```
unitex CurrUnit;
```

```
int i;  
int x;  
int y1;  
int y2;  
int y3;
```

```
int UCS1IsMining;  
int UCS2IsMining;  
int UCS3IsMining;
```

```
int bMiningChecked;
```

```
state Initialize;  
state ShowBriefing;  
state Working;  
state EndMissionFailed;  
state Nothing;  
//-----
```

```
state Initialize
```

```
{  
    //----- Timers -----  
    SetTimer(0, 1200);  
    SetTimer(1, 20);  
  
    //----- Variables -----  
    nTimer0Counter = FirstOffenseAfter;  
    nTimer1Counter = 0;  
    nOffensePlayerNumber = FirstOffensePlayer - 1;
```

```
    DestroyUCS1Achieved = false;  
    DestroyUCS2Achieved = false;  
    DestroyUCS3Achieved = false;  
    DestroyAllUCSAchieved = false;  
    BuildAllRP = false;  
    CollectMoneyAchieved = false;
```

```
    BuildRP1 = false;  
    BuildRP2 = false;  
    BuildRP3 = false;  
    CheckBase = 0;  
    UCS1IsMining = true;  
    UCS2IsMining = true;  
    UCS3IsMining = true;  
    IPP1 = 0;  
    IPP2 = 0;  
    IPP3 = 0;  
    IR1 = 0;  
    IR2 = 0;  
    IR3 = 0;
```

```
    MinX1 = GetPointX(0);  
    MinY1 = GetPointY(0);  
    MaxX1 = GetPointX(1);  
    MaxY1 = GetPointY(1);
```

```
    if((MinX1 > MaxX1){=MinX1;MinX1=MaxX1;MaxX1=i;}  
    if((MinY1 > MaxY1){=MinY1;MinY1=MaxY1;MaxY1=i;})
```

```
    MinX2 = GetPointX(10);  
    MinY2 = GetPointY(10);  
    MaxX2 = GetPointX(11);  
    MaxY2 = GetPointY(11);
```

```
    if((MinX2 > MaxX2){=MinX2;MinX2=MaxX2;MaxX2=i;}  
    if((MinY2 > MaxY2){=MinY2;MinY2=MaxY2;MaxY2=i;})
```

```
    MinX3 = GetPointX(20);  
    MinY3 = GetPointY(20);  
    MaxX3 = GetPointX(21);  
    MaxY3 = GetPointY(21);
```

```
    if((MinX3 > MaxX3){=MinX3;MinX3=MaxX3;MaxX3=i;}  
    if((MinY3 > MaxY3){=MinY3;MinY3=MaxY3;MaxY3=i;})
```

```
    y1 = MinY1 + 1;  
    y2 = MinY2 + 1;
```

```

y3 = MinY3 + 1;

//----- Goals -----
RegisterGoal(goalDestroyUCSSouth,"translateGoal615A");
RegisterGoal(goalDestroyUCSSouthEast,"translateGoal615B");
RegisterGoal(goalDestroyUCSSouthWest,"translateGoal615C");
RegisterGoal(goalCollectMoney,"translateGoal615D");
RegisterGoal(goalBuildSouth,"translateGoal615E");
RegisterGoal(goalBuildSouthEast,"translateGoal615F");
RegisterGoal(goalBuildSouthWest,"translateGoal615G");

EnableGoal(goalDestroyUCSSouth,true);
EnableGoal(goalDestroyUCSSouthEast,true);
EnableGoal(goalDestroyUCSSouthWest,true);
EnableGoal(goalCollectMoney,true);
EnableGoal(goalBuildSouth,true);
EnableGoal(goalBuildSouthEast,true);
EnableGoal(goalBuildSouthWest,true);

//----- Players -----
p_Player = GetPlayer(3);
p_Enemy1 = GetPlayer(1);
p_Enemy2 = GetPlayer(5);
p_Enemy3 = GetPlayer(7);

p_Neutral = GetPlayer(9); //UCS
p_Neutral.EnableStatistics(false);
p_Neutral.SetAll(p_Enemy1);
p_Neutral.SetAll(p_Enemy2);
p_Neutral.SetAll(p_Enemy3);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);

//----- AI -----
if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScene("single\singleEasy");
    p_Enemy2.LoadScene("single\singleEasy");
    p_Enemy3.LoadScene("single\singleEasy");
}

if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScene("single\singleMedium");
    p_Enemy2.LoadScene("single\singleMedium");
    p_Enemy3.LoadScene("single\singleMedium");
}

if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScene("single\singleHard");
    p_Enemy2.LoadScene("single\singleHard");
    p_Enemy3.LoadScene("single\singleHard");
}

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy1.EnableAIFeatures(aiControlDefense,false);

p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlDefense,false);

p_Enemy3.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlDefense,false);

p_Enemy1.EnableResearch("RES_UCS_WMR1",true); //615
p_Enemy1.EnableResearch("RES_UCS_WAMR1",true); //615
p_Enemy1.EnableResearch("RES_UCS_WHPI",true); //615
p_Enemy1.EnableResearch("RES_UCS_UHL1",true); //615
p_Enemy1.EnableResearch("RES_UCS_BOMBER21",true); //615
p_Enemy1.EnableResearch("RES_UCS_BOMBER31",true); //615

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);
//----- Money -----
p_Player.EnableCommand(commandSoldBuilding,true); // 1st

tab
p_Player.SetMoney(10000);

p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(12000);
p_Enemy3.SetMoney(12000);

//-----easy-----
if(GetDifficultyLevel()==0)

{
    //-----Normal-----
    if(GetDifficultyLevel()==1)
    {
        //Enemy1
        KillArea(p_Enemy1.GetIFF(), GetPointX(2), GetPointY(2), 0, 1);

        //Enemy2
        KillArea(p_Enemy2.GetIFF(), GetPointX(12), GetPointY(12), 0, 1);
        KillArea(p_Enemy2.GetIFF(), GetPointX(15), GetPointY(15), 0, 1);

        //Enemy3
        KillArea(p_Enemy3.GetIFF(), GetPointX(22), GetPointY(22), 0, 1);
        KillArea(p_Enemy3.GetIFF(), GetPointX(25), GetPointY(25), 0, 1);
    }

    //-----Normal-----
    if(GetDifficultyLevel()==1)
    {
        //Enemy1
        //Panther + hP: UCSUHL1 # UCSWBHP1 - ok
        p_Enemy1.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "UCSUHL1",
"UCSWBHP1", null, null, null);
        p_Enemy1.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "UCSUHL1",
"UCSWBHP1", null, null, null);
        p_Enemy1.CreateUnitEx(GetPointX(5), GetPointY(5), 0, null, "UCSUHL1",
"UCSWBHP1", null, null, null);

        //Enemy2
        KillArea(p_Enemy2.GetIFF(), GetPointX(15), GetPointY(15), 0, 1);
        //Spider + hR: UCSCML1 # UCSWSMR1 - ok
        p_Enemy2.CreateUnitEx(GetPointX(13), GetPointY(13), 0, null,
"UCSCML1", null, null, null);
        p_Enemy2.CreateUnitEx(GetPointX(14), GetPointY(14), 0, null,
"UCSCML1", null, null, null);

        //Enemy3
        KillArea(p_Enemy3.GetIFF(), GetPointX(25), GetPointY(25), 0, 1);
        //Panther + hP: UCSUHL1 # UCSWBHP1 - ok
        p_Enemy3.CreateUnitEx(GetPointX(23), GetPointY(23), 0, null,
"UCSUHL1", "UCSWBHP1", null, null, null);
        p_Enemy3.CreateUnitEx(GetPointX(24), GetPointY(24), 0, null,
"UCSUHL1", "UCSWBHP1", null, null, null);
    }

    //-----Hard-----
    if(GetDifficultyLevel()==2)
    {
        //----add to p_Enemy-----
        //Enemy1
        //Panther + R: UCSUHL1 # UCSWBSR1
        p_Enemy1.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);
        p_Enemy1.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);
        p_Enemy1.CreateUnitEx(GetPointX(5), GetPointY(5), 0, null, "UCSUHL1",
"UCSWBSR1", null, null, null);

        p_Enemy1.CreateUnitEx(GetPointX(3)+1, 0, null,
"UCSUHL1", "UCSWBSR1", null, null, null);
        p_Enemy1.CreateUnitEx(GetPointX(4)+1, 0, null,
"UCSUHL1", "UCSWBSR1", null, null, null);
        p_Enemy1.CreateUnitEx(GetPointX(5)+1, 0, null,
"UCSUHL1", "UCSWBSR1", null, null, null);

        //Enemy2
        //Panther + hP: UCSUHL1 # UCSWBHP1 - ok
        p_Enemy2.CreateUnitEx(GetPointX(13), GetPointY(13), 0, null,
"UCSUHL1", "UCSWBHP1", null, null, null);
        p_Enemy2.CreateUnitEx(GetPointX(14), GetPointY(14), 0, null,
"UCSUHL1", "UCSWBHP1", null, null, null);

        p_Enemy2.CreateUnitEx(GetPointX(13)+1, 0, null,
"UCSUHL1", "UCSWBHP1", null, null, null);
        p_Enemy2.CreateUnitEx(GetPointX(14)+1, 0, null,
"UCSUHL1", "UCSWBHP1", null, null, null);

        //Enemy3
        //Panther + hR: UCSUHL1 # UCSWBSMR1
        p_Enemy3.CreateUnitEx(GetPointX(23), GetPointY(23), 0, null,
"UCSUHL1", "UCSWBSMR1", null, null, null);
        p_Enemy3.CreateUnitEx(GetPointX(24), GetPointY(24), 0, null,
"UCSUHL1", "UCSWBSMR1", null, null, null);
    }
}

```

```

//----- Buildings -----
p_Player.EnableBuilding("LCBSR",false); //Centrum wydobywco-transporto-
we

SetTimer(2, 1);

p_Enemy2.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy2);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
MinStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings()/5;

p_Player.EnableResearch("RES_LC_WMR1",true);//615
p_Player.EnableResearch("RES_LC_UCR1",true);//615
p_Player.EnableResearch("RES_LC_HGn",true);//615
p_Player.EnableResearch("RES_LC_BHD",true);//615
p_Player.EnableResearch("RES_MMR2",true);//615
p_Player.EnableBuilding("LCBRC",false);
//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
12, 0, 45, 0);
return ShowBriefing, 60;
}
//----- state ShowBriefing -----
{
    AddBriefing("translateBriefing615", p_Player.GetName());
    return Working, 20;
}
//----- state EndMissionFailed -----
{
    EnableNextMission(0,2);
    return Nothing;
}

state Working
{
//-----mission failed-----
if(!p_Player.GetNumberOfUnits())
{
    AddBriefing("translateFailed615", p_Player.GetName());
    return EndMissionFailed;
}

//---- check 1 goal - destroy UCS ---
if(!DestroyUCS1Achieved)
{
    CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();

    if(CurrentStateOfBuildings1 < MinStateOfBuildings1)
    {
        p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy1.SetMoney(0);

        if(!CurrentStateOfBuildings1)
        {
            SetGoalState(goalDestroyUCSSouth, goalAchieved);
            DestroyUCS1Achieved = true;
        }
    }
}

if(!DestroyUCS2Achieved)
{
    CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
    if(CurrentStateOfBuildings2 < MinStateOfBuildings2)
    {
        p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy2.SetMoney(0);

        if(!CurrentStateOfBuildings2)
        {
            SetGoalState(goalDestroyUCSSouthEast, goalAchieved);
            DestroyUCS2Achieved = true;
        }
    }
}

}
}

}
}

if(!DestroyUCS3Achieved)
{
    CurrentStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings();
    if(CurrentStateOfBuildings3 < MinStateOfBuildings3)
    {
        p_Enemy3.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy3.SetMoney(0);

        if(!CurrentStateOfBuildings3)
        {
            SetGoalState(goalDestroyUCSSouthWest, goalAchieved);
            DestroyUCS3Achieved = true;
        }
    }
}

if(!DestroyAllUCSAchieved)
{
    if(DestroyUCS1Achieved && DestroyUCS2Achieved &&
DestroyUCS3Achieved)
    {
        DestroyAllUCSAchieved = true;
    }
}

//-- Buildings ready ---
if(BuildAllRP)
{
    if(BuildRP1 && BuildRP2 && BuildRP3)
    {
        BuildAllRP = true;
    }
}

//is enough money
if(!CollectMoneyAchieved)
{
    nMoney = p_Player.GetMoney();

    if(nMoney >= NeededMoney)
    {
        nMoney = nMoney - NeededMoney;
        p_Player.SetMoney(nMoney);
        SetGoalState(goalCollectMoney, goalAchieved);
        CollectMoneyAchieved = true;
    }
}

//---- All goals achieved ----
if(DestroyAllUCSAchieved && CollectMoneyAchieved && BuildAllRP)
{
    p_Player.SetScriptData(1,2);

    EnableNextMission(0,true);
    AddBriefing("translateAccomplished615");
    EnableEndMissionButton(true);
    return Nothing;
}

}
}

state Nothing
{
    return Nothing, 500;
}

//----- event Timer0() //wolany co minute -----
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofensywy
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber >= LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        if(nOffensePlayerNumber == 2)
            nOffensePlayerNumber = 5;

        if(nOffensePlayerNumber == 6 || nOffensePlayerNumber == 8)
            nOffensePlayerNumber = nOffensePlayerNumber + 1;
    }
}

```

```

p_OffensePlayer = GetPlayer(offensePlayerNumber);

if(p_OffensePlayer.GetNumberOfUnits())
{
    p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
    p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);

    nTimerOCounter = OffenseFrequency;
    nTimer1Counter = OffenseTime;
}
if(p_OffensePlayer.GetNumberOfUnits())
{
    nTimerOCounter = 1;
}

}

//-----
event Timer1() //wolny co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofensywy
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, false);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, false);
        }
    }
}

//-----check UCS bases-----
event Timer2()
{
    //sprawdzanie "na zmiane"
    //----- Base UCS1 -----
    if(checkBase == 0)
    {
        //petla y1
        for(x=MinX1+1; x<MaxX1; x=x+1)
        {
            CurrUnit = GetUnit(x, y1, 0);

            //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
            terenie - jesli tak to niech nie wydobywa
            if(UCS1IsMining || UCS2IsMining || UCS3IsMining)
            {
                if(CurrUnit.IsHarvester())
                {
                    CurrUnit.CommandMove(MinX1, MinY1, 0);

                    if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                    {
                        UCS1IsMining = false;
                        p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS1 nie wydobywa ");
                    }

                    if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                    {
                        UCS2IsMining = false;
                        p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS2 nie wydobywa ");
                    }

                    if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                    {
                        UCS3IsMining = false;
                        p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS3 nie wydobywa ");
                    }
                }
            }
        }
    }
}

//czy wybudowano w bazie 1
if(!BuildRP1)
{
    rafinerie
    //sprawdzenie czy odpowiednie budynki na tym terenie
    if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding()) //czy nasz budynek
    {
        BuildingType = CurrUnit.GetBuildingType();

        if(BuildingType == buildingPowerPlant) //wykryl elektrownie
        {
            IPP1 = IPP1 + 1;
            //TraceD(" El. w UCS1 ");
            //TraceD("IPP1");
            //TraceD(" ");
        }
    }

    if(!IR1 && BuildingType == buildingMiningRefinery) //wykryl
rafinerie
    {
        IR1 = 1; //wybudowano kopalnie
        //TraceD(" Wybudowano kopalnie w UCS1 ");
    }

    //jest juz rafineria i co najmniej 3 elektrownie
    if(IR1 && IPP1>fieldsCoveredByPP2 && !BuildRP1)
    {
        SetGoalState(goalBuildSouth, goalAchieved);
        BuildRP1 = true;
        TraceD(" Wybudowano to co trzeba w UCS1 ");
    }
}

y1 = y1 + 1;
if(y1 >= MaxY1) //koniec sprawdzania obszaru
{
    y1 = MinY1 + 1;
    IPP1 = 0;

    //next time check next base
    CheckBase = 1;
}

//----- Base UCS2 -----
if(checkBase == 1)
{
    //petla y2
    for(x=MinX2+1; x<MaxX2; x=x+1)
    {
        CurrUnit = GetUnit(x, y2, 0);

        //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
        terenie - jesli tak to niech nie wydobywa
        if(UCS1IsMining || UCS2IsMining || UCS3IsMining)
        {
            if(CurrUnit.IsHarvester())
            {
                CurrUnit.CommandMove(MinX2, MinY2, 0);

                if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                {
                    UCS1IsMining = false;
                    p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                    TraceD("UCS1 nie wydobywa ");
                }

                if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                {
                    UCS2IsMining = false;
                    p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                    TraceD("UCS2 nie wydobywa ");
                }

                if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                {
                    UCS3IsMining = false;
                    p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                    TraceD("UCS3 nie wydobywa ");
                }
            }
        }
    }
}

```

```

        }

    //czy wybudowano w bazie 2
    if(!BuildRP2)
    {
        //sprawdzenie czy odpowiednie budynki na tym terenie
        if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
        CurrUnit.IsBuilding()) //czy nasz budynek
        {
            BuildingType = CurrUnit.GetBuildingType();

            if(BuildingType == buildingPowerPlant) //wykryl elektrownie
            {
                IPP2 = IPP2 + 1;
                //TraceD(" El. w UCS2 ");
                //TraceD(IPP2);
                //TraceD(" ");
            }

            if(IRR2 && BuildingType == buildingMiningRefinery) //wykryl
            {
                IR2 = 1; //wybudowano kopalnie
                //TraceD(" Wybudowano kopalnie w UCS2 ");

                //jest juz rafineria i co najmniej 3 elektrownie
                if(IR2 && IPP2>fieldsCoveredByPP2 && !BuildRP2)
                {
                    SetGoalState(goalBuildSouthEast, goalAchieved);
                    BuildRP2 = true;
                    TraceD(" Wybudowano to co trzeba w UCS2 ");
                }
            }
        }

        y2 = y2 + 1;
        if(y2 >= MaxY2) //koniec sprawdzania obszaru
        {
            y2 = MinY2+1;
            IPP2 = 0;

            //next time check next base
            CheckBase = 2;
        }
    }

    //----- Base UCS3 -----
    if(CheckBase == 2)
    {
        //petla y3

        for(x=MinX3+1; x<MaxX3; x=x+1)
        {
            CurrUnit = GetUnit(x, y3, 0);

            //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
            terenie - jesli tak to niech nie wydobywa
            if(UCS1IsMining || UCS2IsMining || UCS3IsMining)
            {
                if(CurrUnit.IsHarvester())
                {
                    CurrUnit.CommandMove(MinX3, MinY3, 0);

                    if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
                    p_Enemy1.GetIFFNumber())
                    {
                        UCS1IsMining = false;
                        p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
                        aiBuildMiningUnits | aiControlMiningUnits,false);
                        TraceD(" UCS1 nie wydobywa ");
                    }

                    if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
                    p_Enemy2.GetIFFNumber())
                    {
                        UCS2IsMining = false;
                        p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
                        aiBuildMiningUnits | aiControlMiningUnits,false);
                        TraceD(" UCS2 nie wydobywa ");
                    }
                }
            }
        }
    }

    if(UCS3IsMining & CurrUnit.GetIFFNumber() ==
    p_Enemy3.GetIFFNumber())
    {
        UCS3IsMining = false;
        p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
        aiBuildMiningUnits | aiControlMiningUnits,false);
        TraceD(" UCS3 nie wydobywa ");
    }
}

//czy wybudowano w bazie 3
if(!BuildRP3)
{
    //sprawdzenie czy odpowiednie budynki na tym terenie
    if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
    CurrUnit.IsBuilding()) //czy nasz budynek
    {
        BuildingType = CurrUnit.GetBuildingType();

        if(BuildingType == buildingPowerPlant) //wykryl elektrownie
        {
            IPP3 = IPP3 + 1;
            //TraceD(" El. w UCS3 ");
            //TraceD(IPP3);
            //TraceD(" ");
        }

        if(IRR3 && BuildingType == buildingMiningRefinery) //wykryl
        {
            IR3 = 1; //wybudowano kopalnie
            //TraceD(" Wybudowano kopalnie w UCS3 ");

            //jest juz rafineria i co najmniej 3 elektrownie
            if(IR3 && IPP3>fieldsCoveredByPP2 && !BuildRP3)
            {
                SetGoalState(goalBuildSouthWest, goalAchieved);
                BuildRP3 = true;
                TraceD(" Wybudowano to co trzeba w UCS3 ");
            }
        }
    }

    y3 = y3 + 1;
    if(y3 >= MaxY3) //koniec sprawdzania obszaru
    {
        y3 = MinY3+1;
        IPP3 = 0;

        //next time check next base
        CheckBase = 0;
    }
}

event EndMission()
{
    if(DestroyAllUCSAchieved && CollectMoneyAchieved && BuildAllRP)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
        +p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission "translateMission616"
{
    consts
    {
        scriptFieldMoney=9;
        goalEscortConvoy = 0;
        goalFangHaveToSurvived = 1;
    }

    player p_Player;
    player p_Enemy;
}

```

```

unitex uFang;
unitex p_ConvoyCraft1;
unitex p_ConvoyCraft2;
unitex p_ConvoyCraft3;
unitex p_ConvoyCraft4;
unitex p_ConvoyCraft5;
unitex p_ConvoyCraft6;

int bCheckEndMission;
int nDestX;
int nDestY;

state Initialize;
state ShowBriefing;
state Working;
state MissionFailed;
state MissionAccomplished;
state Nothing;

//-----
state Initialize
{
    int i;
//----- Goals -----
    if(GetDifficultyLevel()==0)
        RegisterGoal(goalEscortConvoy,"translateGoal616a",2);
    if(GetDifficultyLevel()==1)
        RegisterGoal(goalEscortConvoy,"translateGoal616a",4);
    if(GetDifficultyLevel()==2)
        RegisterGoal(goalEscortConvoy,"translateGoal616a",6);

    RegisterGoal(goalFangHaveToSurvived,"translateGoal616b");

//---Show goals on list---
    EnableGoal(goalEscortConvoy, true);
    EnableGoal(goalFangHaveToSurvived, true);

//----- Players -----
    p_Player = GetPlayer(3); //LC
    p_Enemy = GetPlayer(1); //UCS

//----- AI -----
//wylacz inteligencje graczy
    p_Enemy.EnableAIfeatures(aiEnabled, false);

//----- Money -----
    p_Player.SetMoney(0);
    p_Enemy.SetMoney(0);

//----- Variables -----
uFang = GetUnit(GetPointX(30),GetPointY(30), 0);
p_ConvoyCraft1 = GetUnit(GetPointX(31),GetPointY(31), 0);
p_ConvoyCraft2 = GetUnit(GetPointX(32),GetPointY(32), 0);
p_ConvoyCraft3 = GetUnit(GetPointX(33),GetPointY(33), 0);
p_ConvoyCraft4 = GetUnit(GetPointX(34),GetPointY(34), 0);
p_ConvoyCraft5 = GetUnit(GetPointX(35),GetPointY(35), 0);
p_ConvoyCraft6 = GetUnit(GetPointX(36),GetPointY(36), 0);
    p_Player.AddUnitToSpecialTab(uFang,true,-1);

    p_Player.AddUnitToSpecialTab(p_ConvoyCraft1,true,-1);
    p_Player.AddUnitToSpecialTab(p_ConvoyCraft2,true,-1);
    p_Player.AddUnitToSpecialTab(p_ConvoyCraft3,true,-1);
    p_Player.AddUnitToSpecialTab(p_ConvoyCraft4,true,-1);
    p_Player.AddUnitToSpecialTab(p_ConvoyCraft5,true,-1);
    p_Player.AddUnitToSpecialTab(p_ConvoyCraft6,true,-1);

nDestX=GetPointX(13);
nDestY=GetPointY(13);

bCheckEndMission=false;

//----- Artefacts -----
for(i=20;i<=27;i=i+1)
{
    CreateArtifact("NEAMINE",GetPointX(i),GetPointY(i),GetPointZ(i),i,artefactSpecialAI
    Other);
}

//--- Creating additional units -----
if(GetDifficultyLevel()==0)
{
    p_Player.CreateUnitEx(GetPointX(0),GetPointY(0), 0, null, "LCUCR3",
    "LCWHL2", "LCWSL2", null, null,1); // Crater m3 + E + E
    p_Player.CreateUnitEx(GetPointX(1),GetPointY(1), 0, null, "LCUCR3",
    "LCWHL2", "LCWSL2", null, null,1); // Crater m3 + E + E
    p_Player.CreateUnitEx(GetPointX(2),GetPointY(2), 0, null, "LCUCU3",
    "LCWMR3", "LCWMR3", null, null,1); // Crusher m3 + R + R
}
    if((GetDifficultyLevel()==1)
    {
        p_Player.CreateUnitEx(GetPointX(0),GetPointY(0), 0, null, "LCUCR3",
    "LCWHL2", "LCWSL2", null, null); // Crater m3 + E + E

    p_Enemy.CreateUnitEx(GetPointX(3),GetPointY(3), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(4),GetPointY(4), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(5),GetPointY(5), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(6),GetPointY(6), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(7),GetPointY(7), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(8),GetPointY(8), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(9),GetPointY(9), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(10),GetPointY(10), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(11),GetPointY(11), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(12),GetPointY(12), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
}
    if((GetDifficultyLevel()==2)
    {
        p_Enemy.CreateUnitEx(GetPointX(3),GetPointY(3), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(4),GetPointY(4), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(5),GetPointY(5), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(6),GetPointY(6), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(7),GetPointY(7), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(8),GetPointY(8), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(9),GetPointY(9), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(10),GetPointY(10), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(11),GetPointY(11), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
    p_Enemy.CreateUnitEx(GetPointX(12),GetPointY(12), 0, null, "UCSUML1",
    "UCSWSSP1", null, null, null); // Spider + P
}
    p_Enemy.CreateUnitEx(GetPointX(3)+1,GetPointY(3), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(4)+1,GetPointY(4), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(5)+1,GetPointY(5), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(6)+1,GetPointY(6), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(7)+1,GetPointY(7), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(8)+1,GetPointY(8), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(9)+1,GetPointY(9), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(10)+1,GetPointY(10), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(11)+1,GetPointY(11), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(12)+1,GetPointY(12), 0, null,
    "UCSUML1", "UCWSMR1", null, null, null); // Spider + hR
}

//----- Building -----
    p_Player.EnableCommand(commandSoldBuilding,false);
    p_Player.EnableBuilding("LCBB",false);
    p_Player.EnableBuilding("LCBPP",false);
    p_Player.EnableBuilding("LCBPP2",false);
    p_Player.EnableBuilding("LCBSB",false);
    p_Player.EnableBuilding("LCBMR",false);
    p_Player.EnableBuilding("LCBSR",false);
}

```

```

p_Player.EnableBuilding("LCBRC",false);
p_Player.EnableBuilding("LCBAB",false);
p_Player.EnableBuilding("LCBGA",false);
p_Player.EnableBuilding("LCBNE",false);
p_Player.EnableBuilding("LCBDE",false);
p_Player.EnableBuilding("LCBHQ",false);
p_Player.EnableBuilding("LCBART",false);
p_Player.EnableBuilding("LCBUC",false);
p_Player.EnableBuilding("LCBSD",false);
p_Player.EnableBuilding("LCBLZ",false);
p_Player.EnableBuilding("LCBSS",false);
p_Player.EnableBuilding("LCBEN",false);
p_Player.EnableBuilding("LCBEN1",false);
//----- Camera -----
CallCamera();
```

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0,4,5,0);
 return ShowBriefing, 100;

//-----
 state ShowBriefing
{
 if(GetDifficultyLevel()==0)
 AddBriefing("translateBriefing616", p_Player.GetName(),2);
 if(GetDifficultyLevel()==1)
 AddBriefing("translateBriefing616", p_Player.GetName(),4);
 if(GetDifficultyLevel()==2)
 AddBriefing("translateBriefing616", p_Player.GetName(),6);
 return Working, 100;
}

//-----
 state Working
{
 int nTrucksAlive;
 int nTrucksFinished;
 nTrucksAlive=0;
 if(p_ConvoyCraft1.IsLive())nTrucksAlive=nTrucksAlive+1;
 if(p_ConvoyCraft2.IsLive())nTrucksAlive=nTrucksAlive+1;
 if(p_ConvoyCraft3.IsLive())nTrucksAlive=nTrucksAlive+1;
 if(p_ConvoyCraft4.IsLive())nTrucksAlive=nTrucksAlive+1;
 if(p_ConvoyCraft5.IsLive())nTrucksAlive=nTrucksAlive+1;
 if(p_ConvoyCraft6.IsLive())nTrucksAlive=nTrucksAlive+1;
 if(bCheckEndMission)
 {
 bCheckEndMission=false;
 if(!luFang.IsLive())
 {
 SetGoalState(goalFangHaveToSurvived, goalFailed);
 AddBriefing("translateFailed616a", p_Player.GetName());
 return MissionFailed, 20;
 }
 if((nTrucksAlive<6 && GetDifficultyLevel()==2) ||
 (nTrucksAlive<4 && GetDifficultyLevel()==1) ||
 (nTrucksAlive<2 && GetDifficultyLevel()==0))
 {
 SetGoalState(goalEscortConvoy,goalFailed);
 AddBriefing("translateFailed616b", p_Player.GetName());
 return MissionFailed, 20;
 }
 }
 nTrucksFinished=0;
 if(p_ConvoyCraft1.IsLive())
 && p_ConvoyCraft1.DistanceTo(nDestX,nDestY) <
 7)nTrucksFinished=nTrucksFinished+1;
 if(p_ConvoyCraft2.IsLive() && p_ConvoyCraft2.DistanceTo(nDestX,nDestY) <
 7)nTrucksFinished=nTrucksFinished+1;
 if(p_ConvoyCraft3.IsLive() && p_ConvoyCraft3.DistanceTo(nDestX,nDestY) <
 7)nTrucksFinished=nTrucksFinished+1;
 if(p_ConvoyCraft4.IsLive() && p_ConvoyCraft4.DistanceTo(nDestX,nDestY) <
 7)nTrucksFinished=nTrucksFinished+1;
 if(p_ConvoyCraft5.IsLive() && p_ConvoyCraft5.DistanceTo(nDestX,nDestY) <
 7)nTrucksFinished=nTrucksFinished+1;
 if(p_ConvoyCraft6.IsLive() && p_ConvoyCraft6.DistanceTo(nDestX,nDestY) <
 7)nTrucksFinished=nTrucksFinished+1;

```

if( nTrucksAlive==nTrucksFinished &&
    ((GetDifficultyLevel()==0 && nTrucksFinished>1) ||
     ((GetDifficultyLevel()==1)&& nTrucksFinished>3) ||
     ((GetDifficultyLevel()==2)&& nTrucksFinished>5)))
{
    SetGoalState(goalEscortConvoy, goalAchieved);
    p_Player.EnableResearch("RES_LCUNH",true); //616
    p_Player.AddResearch("RES_LCUNH"); //616
    SetGoalState(goalFangHaveToSurvived, goalAchieved);
    AddBriefing("translateAccomplished616", p_Player.GetName());
    return MissionAccomplished, 20;
}

return Working, 100;
}

//-----
state MissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state MissionAccomplished
{
    EnableNextMission(0, true);
    EnableEndMissionButton(true);
    return Nothing;
}

//-----
state Nothing
{
    if(luFang.IsLive())
    {
        SetGoalState(goalFangHaveToSurvived, goalFailed);
        AddBriefing("translateFailed616a", p_Player.GetName());
        return MissionFailed, 20;
    }
    return Nothing, 100;
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    KillArea(255, GetPointX(alD),GetPointY(alD),GetPointZ(alD),1);
    return true; //nie usuwa sie
}

//-----
event EndMission()
{
    if(GetGoalState(goalEscortConvoy)==goalAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
        +p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission617.ec
mission "translateMission617"
{
    consts
    {
        scriptFieldMoney=9;
        goalDestroyUCSWest = 0;
        goalDestroyUCCSouthWest = 1;
        goalDestroyUCCSouth = 2;
        goalBuildWest = 3;
        goalBuildSouthWest = 4;
        goalBuildSouth = 5;
    }
}
```

```

FirstOffenseAfter = 17; // minuty
OffenseFrequency = 3; // minuty
OffenseTime = 30; // sekundy
FirstOffensePlayer = 1;
LastOffensePlayer = 7;
fieldsCoveredByPP2 = 17;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Player;
player p_Neutral;

int nTimer0Counter;
int nTimer1Counter;
int nOffensePlayerNumber;
player p_OffensePlayer;

int MinStateOfBuildings1;
int MinStateOfBuildings2;
int MinStateOfBuildings3;

int CurrentStateOfBuildings1;
int CurrentStateOfBuildings2;
int CurrentStateOfBuildings3;

int DestroyUCS1Achieved;
int DestroyUCS2Achieved;
int DestroyUCS3Achieved;

int DestroyAllUCSAchieved;

int CheckBase;
int BuildAllRP;

int BuildRP1;
int BuildRP2;
int BuildRP3;

int MinX1;
int MinY1;
int MaxX1;
int MaxY1;

int MinX2;
int MinY2;
int MaxX2;
int MaxY2;

int MinX3;
int MinY3;
int MaxX3;
int MaxY3;

int BuildingType;
int IPP1;
int IPP2;
int IPP3;
int IR1;
int IR2;
int IR3;

int nMoney;

unitex CurrUnit;

int i;
int x;
int y1;
int y2;
int y3;

int UCS1IsMining;
int UCS2IsMining;
int UCS3IsMining;

state Initialize;
state ShowBriefing;
state Working;
state EndMissionFailed;
}

state Nothing;
//-----
state Initialize
{
    //----- Timers -----
    SetTimer(0, 1200);
    SetTimer(1, 20);

    //----- Variables -----
    nTimer0Counter = FirstOffenseAfter;
    nTimer1Counter = 0;
    nOffensePlayerNumber = FirstOffensePlayer - 1;

    DestroyUCS1Achieved = false;
    DestroyUCS2Achieved = false;
    DestroyUCS3Achieved = false;
    DestroyAllUCSAchieved = false;
    BuildAllRP = false;

    BuildRP1 = false;
    BuildRP2 = false;
    BuildRP3 = false;
    CheckBase = 0;
    UCS1IsMining = true;
    UCS2IsMining = true;
    UCS3IsMining = true;
    IPP1 = 0;
    IPP2 = 0;
    IPP3 = 0;
    IR1 = 0;
    IR2 = 0;
    IR3 = 0;

    MinX1 = GetPointX(0);
    MinY1 = GetPointY(0);
    MaxX1 = GetPointX(1);
    MaxY1 = GetPointY(1);

    if(MinX1>MaxX1){=MinX1;MinX1=MaxX1;MaxX1=i;}
    if(MinY1>MaxY1){=MinY1;MinY1=MaxY1;MaxY1=i;}

    MinX2 = GetPointX(10);
    MinY2 = GetPointY(10);
    MaxX2 = GetPointX(11);
    MaxY2 = GetPointY(11);

    if(MinX2>MaxX2){=MinX2;MinX2=MaxX2;MaxX2=i;}
    if(MinY2>MaxY2){=MinY2;MinY2=MaxY2;MaxY2=i;}

    MinX3 = GetPointX(20);
    MinY3 = GetPointY(20);
    MaxX3 = GetPointX(21);
    MaxY3 = GetPointY(21);

    if(MinX3>MaxX3){=MinX3;MinX3=MaxX3;MaxX3=i;}
    if(MinY3>MaxY3){=MinY3;MinY3=MaxY3;MaxY3=i;}

    y1 = MinY1 + 1;
    y2 = MinY2 + 1;
    y3 = MinY3 + 1;

    //----- Goals -----
    RegisterGoal(goalDestroyUCSWest,"translateGoal617A");
    RegisterGoal(goalDestroyUCSSouthWest,"translateGoal617B");
    RegisterGoal(goalDestroyUCSSouth,"translateGoal617C");
    RegisterGoal(goalBuildWest,"translateGoal617D");
    RegisterGoal(goalBuildSouthWest,"translateGoal617E");
    RegisterGoal(goalBuildSouth,"translateGoal617F");

    EnableGoal(goalDestroyUCSWest,true);
    EnableGoal(goalDestroyUCSSouthWest,true);
    EnableGoal(goalDestroyUCSSouth,true);
    EnableGoal(goalBuildWest,true);
    EnableGoal(goalBuildSouthWest,true);
    EnableGoal(goalBuildSouth,true);

    //----- Players -----
    p_Player = GetPlayer(3);
    p_Enemy1 = GetPlayer(1);

    p_Enemy2 = GetPlayer(5);
}

```

```

p_Enemy3 = GetPlayer(7);

p_Neutral = GetPlayer(9); //UCS
p_Neutral.EnableStatistics(false);
p_Neutral.SetAlly(p_Enemy1);
p_Neutral.SetAlly(p_Enemy2);
p_Neutral.SetAlly(p_Enemy3);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);

//----- AI -----
if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
    p_Enemy3.LoadScript("single\singleEasy");
}

if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    p_Enemy3.LoadScript("single\singleMedium");
}

if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
    p_Enemy3.LoadScript("single\singleHard");
}

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy1.EnableAIFeatures(aiControlDefense,false);

p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlDefense,false);

p_Enemy3.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlDefense,false);

p_Enemy1.EnableResearch("RES_UCS_UBL1",true); //617
p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

//----- Money -----
p_Player.EnableCommand(commandSoldBuilding,true); // 1st
we
p_Player.SetMoney(10000);

p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(12000);
p_Enemy3.SetMoney(12000);

//----- Buildings -----
p_Player.EnableBuilding("LCBSR",false); //Centrum wydobywczo-transporto-
we
SetTimer(2, 1);

p_Enemy2.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy2);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
MinStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings()/5;

p_Player.EnableResearch("RES_LC_UBO1",true); //617
p_Player.EnableResearch("RES_LC_AMR1",true); //617
p_Player.EnableResearch("RES_LC_WHL1",true); //617
p_Player.EnableBuilding("LCBRC",false);

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
12, 0, 45, 0);
return ShowBriefing, 60;
}
//----- Show Briefing -----
state ShowBriefing
{
    AddBriefing("translateBriefing617", p_Player.GetName());
    return Working, 20;
}

//----- End Mission -----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

state Working
{
    //-----mission failed-----
    if(p_Player.GetNumberOfUnits())
    {
        AddBriefing("translateFailed617", p_Player.GetName());
        return EndMissionFailed;
    }

    //--- check 1 goal - destroy UCS ---
    if(!DestroyUCSAchieved)
    {
        CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();

        if(CurrentStateOfBuildings1 < MinStateOfBuildings1)
        {
            p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy1.SetMoney(0);

            if(!CurrentStateOfBuildings1)
            {
                SetGoalState(goalDestroyUCSWest, goalAchieved);
                DestroyUCS1Achieved = true;
            }
        }

        if(!DestroyUCS2Achieved)
        {
            CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
            if(CurrentStateOfBuildings2 < MinStateOfBuildings2)
            {
                p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
                p_Enemy2.SetMoney(0);

                if(!CurrentStateOfBuildings2)
                {
                    SetGoalState(goalDestroyUCSSouthWest, goalAchieved);
                    DestroyUCS2Achieved = true;
                }
            }

            if(!DestroyUCS3Achieved)
            {
                CurrentStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings();
                if(CurrentStateOfBuildings3 < MinStateOfBuildings3)
                {
                    p_Enemy3.EnableAIFeatures(aiBuildBuildings,false);
                    p_Enemy3.SetMoney(0);

                    if(!CurrentStateOfBuildings3)
                    {
                        SetGoalState(goalDestroyUCSSouth, goalAchieved);
                        DestroyUCS3Achieved = true;
                    }
                }

                if(!DestroyAllUCSAchieved)
                {
                    if(DestroyUCS1Achieved && DestroyUCS2Achieved &&
DestroyUCS3Achieved)
                    {
                        DestroyAllUCSAchieved = true;
                    }
                }

                //-- Buildings ready --
                if(!BuildAllRP)
                {
                    if(BuildRP1 && BuildRP2 && BuildRP3)
                    {
                        BuildAllRP = true;
                    }
                }
            }
        }
    }
}

```

```

        }

//---- All goals achieved ----
if(DestroyAllUCSAchieved && BuildAIIRP)
{
    p_Player.SetScriptData(2,2);

    EnableNextMission(0,true);
    AddBriefing("translateAccomplished617");
    EnableEndMissionButton(true);
    return Nothing;
}

state Nothing
{
    return Nothing, 500;
}

//-----
event Timer0() //wolany co minute
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofenswy
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber >= LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        if(nOffensePlayerNumber == 2)
            nOffensePlayerNumber = 5;

        if(nOffensePlayerNumber == 6 || nOffensePlayerNumber == 8)
            nOffensePlayerNumber = nOffensePlayerNumber + 1;

        p_OffensePlayer = GetPlayer(nOffensePlayerNumber);

        if(p_OffensePlayer.GetNumberOfUnits())
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);

            nTimer0Counter = OffenseFrequency;
            nTimer1Counter = OffenseTime;
        }
        if(ip_OffensePlayer.GetNumberOfUnits())
        {
            nTimer0Counter = 1;
        }
    }
}

//-----
event Timer1() //wolany co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofenswy
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, false);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, false);
        }
    }
}

//-----check UCS bases-----
event Timer2()
{
    //sprawdzanie "na zmiane"
    //------------------ Base UCS1 -----
    if(CheckBase == 0)
    {
        //petla y1

        for(x=MinX1+1; x<MaxX1; x=x+1)
        {
            CurrUnit = GetUnit(x, y1, 0);

            //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
            terenie - jesli tak to niech nie wydobywa
            if(UCS1IsMining || UCS2IsMining || UCS3IsMining)
            {
                if(CurrUnit.IsHarvester())
                {
                    CurrUnit.CommandMove(MinX1, MinY1, 0);

                    if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                    {
                        UCS1IsMining = false;
                        p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD(" UCS1 nie wydobywa ");
                    }
                }
                if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                {
                    UCS2IsMining = false;
                    p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                    TraceD(" UCS2 nie wydobywa ");
                }
                if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                {
                    UCS3IsMining = false;
                    p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits, false);
                    TraceD(" UCS3 nie wydobywa ");
                }
            }
            //czy wybudowano w bazie 1
            if(BuildRP1)
            {
                //sprawdzenie czy odpowiednie budynki na tym terenie
                if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding())
                //czy nasz budynek
                {
                    BuildingType = CurrUnit.GetBuildingType();

                    if(BuildingType == buildingPowerPlant) //wykryl elektrownie
                    {
                        IPP1 = IPP1 + 1;
                        //TraceD(" El. w UCS1 ");
                        //TraceD("IPP1");
                        //TraceD(" ");
                    }
                    if(!IR1 && BuildingType == buildingMiningRefinery) //wykryl
rafinerie
                    {
                        IR1 = 1; //wybudowano kopalnie
                        //TraceD(" Wybudowano kopalnie w UCS1 ");
                    }
                    //jest juz rafineria i co najmniej 3 elektrownie
                    if(IR1 && IPP1 > fieldsCoveredByPP2 && !BuildRP1)
                    {
                        BuildRP1 = true;
                        SetGoalState(goalBuildWest, goalAchieved);
                        TraceD(" Wybudowano to co trzeba w UCS1 ");
                    }
                }
            }
            y1 = y1 + 1;
            if(y1 >= MaxY1) //koniec sprawdzania obszaru
            {
                y1 = MinY1 + 1;
                IPP1 = 0;

                //next time check next base
                CheckBase = 1;
            }
        }
        //------------------ Base UCS2 -----
        if(CheckBase == 1)
        {
    }
}

```

```

//petla y2
for(x=MinX2+1; x<MaxX2; x=x+1)
{
    CurrUnit = GetUnit(x, y2, 0);

    //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
    terenie - jesli tak to niech nie wydobywa
    if(UCS1IsMining || UCS2IsMining || UCS3IsMining)
    {
        if(CurrUnit.IsHarvester())
        {
            CurrUnit.CommandMove(MinX2, MinY2, 0);

            if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
            {
                UCS1IsMining = false;
                p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                TraceD("UCS1 nie wydobywa ");
            }

            if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
            {
                UCS2IsMining = false;
                p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                TraceD("UCS2 nie wydobywa ");
            }

            if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
            {
                UCS3IsMining = false;
                p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                TraceD("UCS3 nie wydobywa ");
            }
        }
    }

    //czy wybudowano w bazie 2
    if(!BuildRP2)
    {
        //sprawdzenie czy odpowiednie budynki na tym terenie
        if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding()) //czy nasz budynek
        {
            BuildingType = CurrUnit.GetBuildingType();

            if(BuildingType == buildingPowerPlant) //wykryl elektrownie
            {
                IPP2 = IPP2 + 1;
                //TraceD("El. w UCS2 ");
                //TraceD("IPP2");
                //TraceD(" ");
            }

            if(!IR2 && BuildingType == buildingMiningRefinery) //wykryl
            {
                IR2 = 1; //wybudowano kopalnie
                //TraceD(" Wybudowano kopalnie w UCS2 ");
            }

            //jest juz rafineria i co najmniej 3 elektrownie
            if(IR2 && IPP2>fieldsCoveredByPP2 && !BuildRP2)
            {
                SetGoalState(goalBuildSouthWest, goalAchieved);
                BuildRP2 = true;
                TraceD(" Wybudowano to co trzeba w UCS2 ");
            }
        }
    }

    y2 = y2 + 1;
    if(y2 >= MaxY2) //koniec sprawdzania obszaru
    {
        y2 = MinY2+1;
        IPP2 = 0;
    }
}

//next time check next base
CheckBase = 2;
}
}

//-----
if(CheckBase == 2)
{
    //petla y3
    for(x=MinX3+1; x<MaxX3; x=x+1)
    {
        CurrUnit = GetUnit(x, y3, 0);

        //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
        terenie - jesli tak to niech nie wydobywa
        if(UCS1IsMining || UCS2IsMining || UCS3IsMining)
        {
            if(CurrUnit.IsHarvester())
            {
                CurrUnit.CommandMove(MinX3, MinY3, 0);

                if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                {
                    UCS1IsMining = false;
                    p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS1 nie wydobywa ");
                }

                if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                {
                    UCS2IsMining = false;
                    p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS2 nie wydobywa ");
                }

                if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                {
                    UCS3IsMining = false;
                    p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS3 nie wydobywa ");
                }
            }
        }

        //czy wybudowano w bazie 3
        if(!BuildRP3)
        {
            //sprawdzenie czy odpowiednie budynki na tym terenie
            if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding()) //czy nasz budynek
            {
                BuildingType = CurrUnit.GetBuildingType();

                if(BuildingType == buildingPowerPlant) //wykryl elektrownie
                {
                    IPP3 = IPP3 + 1;
                    //TraceD("El. w UCS3 ");
                    //TraceD("IPP3");
                    //TraceD(" ");
                }

                if(!IR3 && BuildingType == buildingMiningRefinery) //wykryl
                {
                    IR3 = 1; //wybudowano kopalnie
                    //TraceD(" Wybudowano kopalnie w UCS3 ");
                }

                //jest juz rafineria i co najmniej 3 elektrownie
                if(IR3 && IPP3>fieldsCoveredByPP2 && !BuildRP3)
                {
                    SetGoalState(goalBuildSouth, goalAchieved);
                    BuildRP3 = true;
                    TraceD(" Wybudowano to co trzeba w UCS3 ");
                }
            }
        }
    }
}
}

```

```

y3 = y3 + 1;
if(y3 >= MaxY3) //koniec sprawdzania obszaru
{
    y3 = MinY3+1;
    IPP3 = 0;

    //next time check next base
    CheckBase = 0;
}
}

//-----
event EndMission()
{
    if(DestroyAllUCSAchieved && BuildAllRP)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}
}

mission618.ec
mission "translateMission618"
{
consts
{
    scriptFieldMoney=9;
    goalDestroyNorthCannon = 0;
    goalDestroyWestCannon = 1;
    goalDestroySouthCannon = 2;
    goalFangHaveToSurvive = 3;
}

player p_Player;
player p_Enemy;
player p_TmpPlayer;

int bCheckEndMission;

unitex Cannon1;
unitex Cannon2;
unitex Cannon3;
unitex Cannon4;
unitex Cannon5;
unitex Cannon6;
unitex CannonHQ1;
unitex CannonHQ2;
unitex CannonHQ3;
unitex uFang;

state Initialize;
state ShowBriefing;
state Working;
state MissionFailed;
state Nothing;

//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(goalDestroyNorthCannon, "translateGoal618a");
    RegisterGoal(goalDestroyWestCannon, "translateGoal618b");
    RegisterGoal(goalDestroySouthCannon, "translateGoal618c");
    RegisterGoal(goalFangHaveToSurvive, "translateGoal618d");
    //Show goals on list-
    EnableGoal(goalDestroyNorthCannon, true);
    EnableGoal(goalDestroyWestCannon, true);
    EnableGoal(goalDestroySouthCannon, true);
    EnableGoal(goalFangHaveToSurvive, true);

    //----- Players -----
    p_Player = GetPlayer(3); //LC
    p_Enemy = GetPlayer(1); //UCS

    p_TmpPlayer = GetPlayer(2);
    p_TmpPlayer.EnableStatistics(false);

    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy.LoadScript("single\singleEasy");
        if(GetDifficultyLevel()==1)
        {
            p_Enemy.LoadScript("single\singleMedium");
        }
        if(GetDifficultyLevel()==2)
        {
            p_Enemy.LoadScript("single\singleHard");
        }

        //wylacz inteligencje graczy
        p_Enemy.EnableAIFeatures(aiControlOffense,false);
        p_Enemy.EnableAIFeatures(aiControlDefense,false);
        p_Enemy.EnableAIFeatures(aiBuildBuildings,false);

        //----- Money -----
        p_Player.SetMoney(0);
        p_Enemy.SetMoney(0);

        //----- Timers -----
        //----- Variables -----
        bCheckEndMission = false;

        Cannon1 = GetUnit(GetPointX(8), GetPointY(8), 0);
        Cannon2 = GetUnit(GetPointX(9), GetPointY(9), 0);
        Cannon3 = GetUnit(GetPointX(10), GetPointY(10), 0);
        Cannon4 = GetUnit(GetPointX(11), GetPointY(11), 0);
        Cannon5 = GetUnit(GetPointX(12), GetPointY(12), 0);
        Cannon6 = GetUnit(GetPointX(13), GetPointY(13), 0);

        CannonHQ1 = GetUnit(GetPointX(14), GetPointY(14), 0);
        CannonHQ2 = GetUnit(GetPointX(15), GetPointY(15), 0);
        CannonHQ3 = GetUnit(GetPointX(16), GetPointY(16), 0);

        uFang = GetUnit(GetPointX(31), GetPointY(31), 0);
        p_Player.AddUnitToSpecialTab(uFang,true,-1);

        //--- Creating & destroying additional units ---
        if(GetDifficultyLevel()==0)
        {
            p_Player.CreateUnitEx(GetPointX(0), GetPointY(0), 0, null, "LCUCR3",
"LCWHL2", null, "LCWSL2", null, 1); // Crater m3 + E + E
            p_Player.CreateUnitEx(GetPointX(1), GetPointY(1), 0, null, "LCUCU3",
"LCWMR3", "LCWMR3", null, null, 1); // Crusher m3 + R + R
            p_Player.CreateUnitEx(GetPointX(2), GetPointY(2), 0, null, "LCUBO2",
"LCWAMR2", null, null, null, 1); // Thunder m2 + HR
            p_Player.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "LCUBO2",
"LCWAMR2", null, null, null, 1); // Meteor m3 + 20mm

            p_Player.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "LCUCR3",
"LCWHL2", null, "LCWSL2", null, 1); // Crater m3 + E + E
            p_Player.CreateUnitEx(GetPointX(5), GetPointY(5), 0, null, "LCUCU3",
"LCWMR3", "LCWMR3", null, null, 1); // Crusher m3 + R + R
            p_Player.CreateUnitEx(GetPointX(6), GetPointY(6), 0, null, "LCUBO2",
"LCWAMR2", null, null, null, 1); // Thunder m2 + HR
            p_Player.CreateUnitEx(GetPointX(7), GetPointY(7), 0, null, "LCUBO2",
"LCWAMR2", null, null, null, 1); // Meteor m3 + 20mm

            KillArea(p_Enemy.GetIFF(), GetPointX(17), GetPointY(17), 0, 1);
            KillArea(p_Enemy.GetIFF(), GetPointX(18), GetPointY(18), 0, 1);
            KillArea(p_Enemy.GetIFF(), GetPointX(19), GetPointY(19), 0, 1);
            KillArea(p_Enemy.GetIFF(), GetPointX(20), GetPointY(20), 0, 1);
            KillArea(p_Enemy.GetIFF(), GetPointX(21), GetPointY(21), 0, 1);
            KillArea(p_Enemy.GetIFF(), GetPointX(22), GetPointY(22), 0, 1);

            if(GetDifficultyLevel()==1)
            {
                p_Player.CreateUnitEx(GetPointX(0), GetPointY(0), 0, null, "LCUCR3",
"LCWHL2", null, "LCWSL2", null, 1); // Crater m3 + E + E
                p_Player.CreateUnitEx(GetPointX(1), GetPointY(1), 0, null, "LCUCU3",
"LCWMR3", "LCWMR3", null, null, 1); // Crusher m3 + R + R
                p_Player.CreateUnitEx(GetPointX(2), GetPointY(2), 0, null, "LCUBO2",
"LCWAMR2", null, null, null, 1); // Thunder m2 + HR
                p_Player.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "LCUBO2",
"LCWAMR2", null, null, null, 1); // Meteor m3 + 20mm

                KillArea(p_Enemy.GetIFF(), GetPointX(17), GetPointY(17), 0, 1);
                KillArea(p_Enemy.GetIFF(), GetPointX(19), GetPointY(19), 0, 1);
                KillArea(p_Enemy.GetIFF(), GetPointX(21), GetPointY(21), 0, 1);

                p_Enemy.CreateUnitEx(GetPointX(23), GetPointY(23), 0, null, "UCSUML1",
"UCSWSMR1", null, null); // Spider + HR
                p_Enemy.CreateUnitEx(GetPointX(24), GetPointY(24), 0, null, "UCSUML1",
"UCSWSMR1", null, null); // Spider + HR
                p_Enemy.CreateUnitEx(GetPointX(25), GetPointY(25), 0, null, "UCSUML1",

```

```

"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(26), GetPointY(26), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(27), GetPointY(27), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(28), GetPointY(28), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(29), GetPointY(29), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(30), GetPointY(30), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
}
if(GetDifficultyLevel() == 2)
{
    p_Enemy.CreateUnitEx(GetPointX(23), GetPointY(23), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(24), GetPointY(24), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(25), GetPointY(25), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(26), GetPointY(26), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(27), GetPointY(27), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(28), GetPointY(28), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(29), GetPointY(29), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
    p_Enemy.CreateUnitEx(GetPointX(30), GetPointY(30), 0, null, "UCSUML1",
"UCSWSMR1", null, null, null); // Spider + hR
}
&&
    p_Enemy.CreateUnitEx(GetPointX(23), GetPointY(23) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(24), GetPointY(24) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(25), GetPointY(25) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(26), GetPointY(26) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(27), GetPointY(27) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(28), GetPointY(28) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(29), GetPointY(29) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
    p_Enemy.CreateUnitEx(GetPointX(30), GetPointY(30) + 1, 0, null,
"UCSUBL1", "UCSWBMR3", "UCWSRR3", null, null); // Jaguar + HR + R
}

//----- Buildings -----
p_Player.EnableBuilding("LCBBF", false);
p_Player.EnableBuilding("LCBPP", false);
p_Player.EnableBuilding("LCBPP2", false);
p_Player.EnableBuilding("LCBMR", false);
p_Player.EnableBuilding("LCBSB", false);
p_Player.EnableBuilding("LCBA", false);
p_Player.EnableBuilding("LCBMR", false);
p_Player.EnableBuilding("LCBSR", false);
p_Player.EnableBuilding("LCBC", false);
p_Player.EnableBuilding("LCBAB", false);
p_Player.EnableBuilding("LCBGA", false);
p_Player.EnableBuilding("LCBDE", false);
p_Player.EnableBuilding("LCBHQ", false);
p_Player.EnableBuilding("LCBSD", false);
p_Player.EnableBuilding("LCBCW", false);
p_Player.EnableBuilding("LCBSS", false);
p_Player.EnableBuilding("LCBLZ", true);

p_Player.EnableBuilding("LCBUC", false);
p_Player.EnableBuilding("LCBN1", false);
p_Player.EnableBuilding("LCLASERWALL", false);

p_Enemy.EnableResearch("RES_UCS_WAPB1", true);
p_Enemy.EnableResearch("RES_UCS_MB2", true);

p_Player.EnableResearch("RES_LC_ART", true); //618
p_Player.EnableResearch("RES_LC_BWC", true); //618
p_Player.EnableCommand(commandOldBuilding, true);
p_Player.EnableBuilding("LCBRC", false);

//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 12.0, 4.50);
return ShowBriefing, 100;
}

```

```

    {
        if(uFang!=null && !uFang.IsLive())
        {
            SetGoalState(goalFangHaveToSurvive, goalFailed);
            AddBriefing("translateFailed618a", p_Player.GetName());
            return MissionFailed, 20;
        }
        return Nothing, 100;
    }

    //-----
    event UnitDestroyed(unit u_Unit)
    {
        bCheckEndMission=true;
    }

    //-----
    event BuildingDestroyed(unit u_Unit)
    {
        bCheckEndMission=true;
    }

    //-----
    event EndMission()
    {
        if(GetGoalState(goalDestroyNorthCannon) == goalAchieved &&
           GetGoalState(goalDestroyWestCannon) == goalAchieved &&
           GetGoalState(goalDestroySouthCannon) == goalAchieved)
        {
            p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
            p_Player.SetMoney(0);
        }
    }
}

mission619.ec
mission "translateMission619"
{
    consts
    {
        scriptFieldMoney=9;
        goalDestroyUCSNorthWest = 0;
        goalDestroyUCSNorthEast = 1;
        goalDestroyUCSSouthEast = 2;
        goalDestroyUCSSouthWest = 3;
        goalBuildNorthWest = 4;
        goalBuildNorthEast = 5;
        goalBuildSouthEast = 6;
        goalBuildSouthWest = 7;

        goalCollectMoney = 8;
        NeededMoney = 60000;
        FirstOffenseAfter = 17; // minut
        OffenseFrequency = 3; // minut
        OffenseTime = 30; // sekundy
        FirstOffensePlayer = 1;
        LastOffensePlayer = 9;
        fieldsCoveredByPP2 = 3;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;
    player p_Enemy4;

    player p_Player;
    player p_Neutral;

    int nTimer0Counter;
    int nTimer1Counter;
    int nOffensePlayerNumber;
    player p_OffensePlayer;

    int MinStateOfBuildings1;
    int MinStateOfBuildings2;
    int MinStateOfBuildings3;
    int MinStateOfBuildings4;
    int CurrentStateOfBuildings1;

    int CurrentStateOfBuildings2;
    int CurrentStateOfBuildings3;
    int CurrentStateOfBuildings4;

    int DestroyUCS1Achieved;
    int DestroyUCS2Achieved;
    int DestroyUCS3Achieved;
    int DestroyUCS4Achieved;

    int DestroyAllUCSAchieved;
    int CollectMoneyAchieved;
    int BuildAllRP;

    int CheckBase;

    int BuildRP1;
    int BuildRP2;
    int BuildRP3;
    int BuildRP4;

    int MinX1;
    int MinY1;
    int MaxX1;
    int MaxY1;

    int MinX2;
    int MinY2;
    int MaxX2;
    int MaxY2;

    int MinX3;
    int MinY3;
    int MaxX3;
    int MaxY3;

    int MinX4;
    int MinY4;
    int MaxX4;
    int MaxY4;

    int BuildingType;
    int IPP1;
    int IPP2;
    int IPP3;
    int IPP4;
    int IR1;
    int IR2;
    int IR3;
    int IR4;

    int nMoney;
    unitex CurrUnit;

    int i;
    int x;
    int y1;
    int y2;
    int y3;
    int y4;

    int UCS1IsMining;
    int UCS2IsMining;
    int UCS3IsMining;
    int UCS4IsMining;

    state Initialize;
    state ShowBriefing;
    state Working;
    state EndMissionFailed;
    state Nothing;
    //-----
    state Initialize
    {
        //----- Timers -----
        SetTimer(0, 1200);
        SetTimer(1, 20);

        //----- Variables -----
        nTimer0Counter = FirstOffenseAfter;
        nTimer1Counter = 0;
        nOffensePlayerNumber = FirstOffensePlayer - 1;

        DestroyUCS1Achieved = false;
        DestroyUCS2Achieved = false;
    }
}

```

```

DestroyUCS3Achieved = false;
DestroyUCS4Achieved = false;

DestroyAllUCSAchieved = false;
BuildAllRP = false;

CollectMoneyAchieved = false;

BuildRP1 = false;
BuildRP2 = false;
BuildRP3 = false;
BuildRP4 = false;

CheckBase = 0;
UCS1IsMining = true;
UCS2IsMining = true;
UCS3IsMining = true;
UCS4IsMining = true;

IPP1 = 0;
IPP2 = 0;
IPP3 = 0;
IPP4 = 0;
IR1 = 0;
IR2 = 0;
IR3 = 0;
IR4 = 0;

MinX1 = GetPointX(0);
MinY1 = GetPointY(0);
MaxX1 = GetPointX(1);
MaxY1 = GetPointY(1);

if(MinX1>MaxX1){=MinX1;MinX1=MaxX1;MaxX1=;}
if(MinY1>MaxY1){=MinY1;MinY1=MaxY1;MaxY1=;}

MinX2 = GetPointX(10);
MinY2 = GetPointY(10);
MaxX2 = GetPointX(11);
MaxY2 = GetPointY(11);

if(MinX2>MaxX2){=MinX2;MinX2=MaxX2;MaxX2=;}
if(MinY2>MaxY2){=MinY2;MinY2=MaxY2;MaxY2=;}

MinX3 = GetPointX(20);
MinY3 = GetPointY(20);
MaxX3 = GetPointX(21);
MaxY3 = GetPointY(21);

if(MinX3>MaxX3){=MinX3;MinX3=MaxX3;MaxX3=;}
if(MinY3>MaxY3){=MinY3;MinY3=MaxY3;MaxY3=;}

MinX4 = GetPointX(30);
MinY4 = GetPointY(30);
MaxX4 = GetPointX(31);
MaxY4 = GetPointY(31);

if(MinX4>MaxX4){=MinX4;MinX4=MaxX4;MaxX4=;}
if(MinY4>MaxY4){=MinY4;MinY4=MaxY4;MaxY4=;}

y1 = MinY1 + 1;
y2 = MinY2 + 1;
y3 = MinY3 + 1;
y4 = MinY4 + 1;

//----- Goals -----
RegisterGoal(goalDestroyUCSNorthWest,"translateGoal619A");
RegisterGoal(goalDestroyUCSNorthEast,"translateGoal619B");
RegisterGoal(goalDestroyUCSSouthEast,"translateGoal619C");
RegisterGoal(goalDestroyUCSSouthWest,"translateGoal619D");
RegisterGoal(goalBuildNorthWest,"translateGoal19E");
RegisterGoal(goalBuildNorthEast,"translateGoal19F");
RegisterGoal(goalBuildSouthEast,"translateGoal19G");
RegisterGoal(goalBuildSouthWest,"translateGoal19H");
RegisterGoal(goalCollectMoney,"translateGoal619I");

EnableGoal(goalDestroyUCSNorthWest,true);
EnableGoal(goalDestroyUCSNorthEast,true);
EnableGoal(goalDestroyUCSSouthEast,true);
EnableGoal(goalDestroyUCSSouthWest,true);
EnableGoal(goalBuildNorthWest,true);
EnableGoal(goalBuildNorthEast,true);
EnableGoal(goalBuildSouthEast,true);
EnableGoal(goalBuildSouthWest,true);
EnableGoal(goalCollectMoney,true);

//----- Players -----
p_Player = GetPlayer(3);

p_Enemy1 = GetPlayer(1);
p_Enemy2 = GetPlayer(5);
p_Enemy3 = GetPlayer(7);
p_Enemy4 = GetPlayer(9);

p_Neutral = GetPlayer(11); //UCS
p_Neutral.EnableStatistics(false);
p_Neutral.SetAlly(p_Enemy1);
p_Neutral.SetAlly(p_Enemy2);
p_Neutral.SetAlly(p_Enemy3);
p_Neutral.SetAlly(p_Enemy4);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);

//----- AI -----
if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
    p_Enemy3.LoadScript("single\singleEasy");
    p_Enemy4.LoadScript("single\singleEasy");
}

if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    p_Enemy3.LoadScript("single\singleMedium");
    p_Enemy4.LoadScript("single\singleMedium");
}

if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
    p_Enemy3.LoadScript("single\singleHard");
    p_Enemy4.LoadScript("single\singleHard");
}

p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy1.EnableAIFeatures(aiControlDefense,false);

p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlDefense,false);

p_Enemy3.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlDefense,false);

p_Enemy4.EnableAIFeatures(aiControlOffense,false);
p_Enemy4.EnableAIFeatures(aiControlDefense,false);

//----- Money -----
p_Player.SetMoney(10000);

p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(10000);
p_Enemy3.SetMoney(10000);
p_Enemy4.SetMoney(10000);

p_Player.EnableCommand(commandSoldBuilding,true); // 1st tab

//----- Buildings -----
p_Player.EnableBuilding("LCBSR",false); //Centrum wydobywczysto-transportowe
we

p_Player.SetMilitaryUnitsLimit(100000);

SetTimer(2, 1);

p_Enemy2.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy1);
p_Enemy4.SetAlly(p_Enemy1);

p_Enemy3.SetAlly(p_Enemy2);

```

```

p_Enemy4.SetAlly(p_Enemy2);
p_Enemy4.SetAlly(p_Enemy3);

p_Enemy1.EnableResearch("RES_UCS_MB2",true);
p_Enemy1.EnableResearch("RES_UCS_WAPB1", true);
p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);
p_Enemy4.CopyResearches(p_Enemy1);

MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
MinStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings()/5;
MinStateOfBuildings4 = p_Enemy4.GetNumberOfBuildings()/5;

wa p_Player.EnableResearch("RES_LC_WARTILLERY",true); //Artyleria plazmo-
p_Player.EnableResearch("RES_LC_UCU1",true)/;619
p_Player.EnableBuilding("LCBRC",false);

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
12, 0, 45, 0);
return ShowBriefing, 60;
}
//-----
state ShowBriefing
{
    AddBriefing("translateBriefing619", p_Player.GetName());
    return Working, 20;
}
//-----

state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

state Working
{
    //-----mission failed-----
if(!p_Player.GetNumberOfUnits())
{
    AddBriefing("translateFailed619", p_Player.GetName());
    return EndMissionFailed;
}

//---- check 1 goal - destroy UCS ----
if(!DestroyUCS1Achieved)
{
    CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();

    if(CurrentStateOfBuildings1 < MinStateOfBuildings1)
    {
        p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy1.SetMoney(0);

        if(CurrentStateOfBuildings1)
        {
            SetGoalState(goalDestroyUCSNorthWest, goalAchieved);
            DestroyUCS1Achieved = true;
        }
    }
}

if(!DestroyUCS2Achieved)
{
    CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
    if(CurrentStateOfBuildings2 < MinStateOfBuildings2)
    {
        p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy2.SetMoney(0);

        if(!CurrentStateOfBuildings2)
        {
            SetGoalState(goalDestroyUCSNorthEast, goalAchieved);
            DestroyUCS2Achieved = true;
        }
    }
}

if(!DestroyUCS3Achieved)
{
    CurrentStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings();
    if(CurrentStateOfBuildings3 < MinStateOfBuildings3)
    {
        p_Enemy3.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy3.SetMoney(0);

        if(!CurrentStateOfBuildings3)
        {
            SetGoalState(goalDestroyUCSSouthEast, goalAchieved);
            DestroyUCS3Achieved = true;
        }
    }
}

if(!DestroyUCS4Achieved)
{
    CurrentStateOfBuildings4 = p_Enemy4.GetNumberOfBuildings();
    if(CurrentStateOfBuildings4 < MinStateOfBuildings4)
    {
        p_Enemy4.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy4.SetMoney(0);

        if(!CurrentStateOfBuildings4)
        {
            SetGoalState(goalDestroyUCSSouthWest, goalAchieved);
            DestroyUCS4Achieved = true;
        }
    }
}

if(!DestroyAllUCSAchieved)
{
    if(DestroyUCS1Achieved && DestroyUCS2Achieved &&
DestroyUCS3Achieved && DestroyUCS4Achieved)
    {
        DestroyAllUCSAchieved = true;
    }
}

//--- Buildings ready ---
if(BuildAllIRP)
{
    if(BuildRP1 && BuildRP2 && BuildRP3 && BuildRP4)
    {
        BuildAllIRP = true;
    }
}

//is enough money
if(CollectMoneyAchieved)
{
    nMoney = p_Player.GetMoney();

    if(nMoney >= NeededMoney)
    {
        nMoney = nMoney - NeededMoney;
        p_Player.SetMoney(nMoney);
        SetGoalState(goalCollectMoney, goalAchieved);
        CollectMoneyAchieved = true;
    }
}

//---- All goals achieved ----
if(DestroyAllUCSAchieved && BuildAllIRP && CollectMoneyAchieved)
{
    p_Player.SetScriptData(3,2);

    if(p_Player.GetScriptData(11)!=11)//bo musi sie zakończyć misja 618
        p_Player.SetScriptData(11,11);
    else
        EnableNextMission(0, true);

    AddBriefing("translateAccomplished619");
    EnableEndMissionButton(true);
    return Nothing;
}

state Nothing

```

```

    {
        return Nothing, 500;
    }

//-----
event Timer0() //wolany co minute
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofensywy
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber >= LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        if(nOffensePlayerNumber == 2)
            nOffensePlayerNumber = 5;

        if(nOffensePlayerNumber == 6 || nOffensePlayerNumber == 8)
            nOffensePlayerNumber = nOffensePlayerNumber + 1;

        p_OffensePlayer = GetPlayer(nOffensePlayerNumber);

        if(p_OffensePlayer.GetNumberOfUnits())
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);

            nTimer0Counter = OffenseFrequency;
            nTimer1Counter = OffenseTime;
        }
        if(lp_OffensePlayer.GetNumberOfUnits())
        {
            nTimer0Counter = 1;
        }
    }
}

//-----
event Timer1() //wolany co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofensywy
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, false);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, false);
        }
    }
}

//-----check UCS bases-----
event Timer2()
{
    //sprawdzanie "na zmiane"
    //----- Base UCS1 -----
    if(CheckBase == 0)
    {
        //petla y1

        for(x=MinX1+1; x<MaxX1; x=x+1)
        {
            CurrUnit = GetUnit(x, y1, 0);

            //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
            //terenie - jesli tak to niech nie wydobywa
            if(UCS1IsMining || UCS2IsMining || UCS3IsMining || |
                UCS4IsMining)
            {
                if(CurrUnit.IsHarvester())
                {
                    CurrUnit.CommandMove(MinX1, MinY1, 0);

                    if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
                        p_Enemy1.GetIFFNumber())
                    {
                        UCS1IsMining = false;
                        p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
                            aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS1 nie wydobywa ");
                    }
                    if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
                        p_Enemy2.GetIFFNumber())
                    {
                        UCS2IsMining = false;
                        p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
                            aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS2 nie wydobywa ");
                    }
                    if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
                        p_Enemy3.GetIFFNumber())
                    {
                        UCS3IsMining = false;
                        p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
                            aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS3 nie wydobywa ");
                    }
                    if(UCS4IsMining && CurrUnit.GetIFFNumber() ==
                        p_Enemy4.GetIFFNumber())
                    {
                        UCS4IsMining = false;
                        p_Enemy4.EnableAIFeatures(aiBuildMiningBuildings |
                            aiBuildMiningUnits | aiControlMiningUnits, false);
                        TraceD("UCS4 nie wydobywa ");
                    }
                }
            }
        }
    }
}

rafinerie
if(IR1 && BuildingType == buildingMiningRefinery) //wykryl elektrownie
{
    IPP1 = IPP1 + 1;
    //TraceD(" El. w UCS1 ");
    //TraceD("IPP1");
    //TraceD(" ");
}

if(IR1 && BuildingType == buildingMiningRefinery) //wykryl elektrownie
{
    IR1 = 1; //wybudowano kopalnie
    //TraceD(" Wybudowano kopalnie w UCS1 ");
}

//jest juz rafineria i co najmniej 3 elektrownie
if(IR1 && IPP1>fieldsCoveredByPP2 && !BuildRP1)
{
    SetGoalState(goalBuildNorthWest, goalAchieved);
    BuildRP1 = true;
    TraceD(" Wybudowano to co trzeba w UCS1 ");
}

y1 = y1 + 1;
if(y1 >= MaxY1) //koniec sprawdzania obszaru
{
    y1 = MinY1+1;
    IPP1 = 0;

    //next time check next base
    CheckBase = 1;
}

//----- Base UCS2 -----
if(CheckBase == 1)
{
    //petla y2

    for(x=MinX2+1; x<MaxX2; x=x+1)
    {
        CurrUnit = GetUnit(x, y2, 0);
    }
}

```

```

        //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
        terenie - jesli tak to niech nie wydobywa
        if(UCS1IsMining || UCS2IsMining || UCS3IsMining ||

UCS4IsMining)
        {
            if(CurrUnit.IsHarvester())
            {
                CurrUnit.CommandMove(MinX2, MinY2, 0);

                if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                {
                    UCS1IsMining = false;
                    p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS1 nie wydobywa ");
                }

                if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                {
                    UCS2IsMining = false;
                    p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS2 nie wydobywa ");
                }

                if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                {
                    UCS3IsMining = false;
                    p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS3 nie wydobywa ");
                }

                if(UCS4IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy4.GetIFFNumber())
                {
                    UCS4IsMining = false;
                    p_Enemy4.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS4 nie wydobywa ");
                }
            }

            //czy wybudowano w bazie 2
            if(!BuildRP2)
            {
                //sprawdzenie czy odpowiednie budynki na tym terenie
                if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding())
                {
                    BuildingType = CurrUnit.GetBuildingType();

                    if(BuildingType == buildingPowerPlant) //wykryl elektrownie
                    {
                        IPP2 = IPP2 + 1;
                        //TraceD(" El. w UCS2 ");
                        //TraceD(IPP2);
                        //TraceD(" ");
                    }

                    if(IIR2 && BuildingType == buildingMiningRefinery) //wykryl
rafinerie
                    {
                        IIR2 = 1; //wybudowano kopalnie
                        //TraceD(" Wybudowano kopalnie w UCS2 ");
                    }

                    //jest juz rafineria i co najmniej 3 elektrownie
                    if(IIR2 && IPP2>fieldsCoveredByPP2 && !BuildRP2)
                    {
                        BuildRP2 = true;
                        SetGoalState(goalBuildNorthEast, goalAchieved);
                        TraceD(" Wybudowano to co trzeba w UCS2 ");
                    }
                }
            }

            y2 = y2 + 1;
            if(y2 >= MaxY2) //koniec sprawdzania obszaru
            {
                y2 = MinY2+1;
            }
        }

        IPP2 = 0;
        //next time check next base
        CheckBase = 2;
    }
}

//----- Base UCS3 -----
if(CheckBase == 2)
{
    //petla y3
    for(x=MinX3+1; x<MaxX3; x=x+1)
    {
        CurrUnit = GetUnit(x, y3, 0);

        //sprawdzenie jest to konieczne czy UCS nie wydobywa na tym
        terenie - jesli tak to niech nie wydobywa
        if(UCS1IsMining || UCS2IsMining || UCS3IsMining ||

UCS4IsMining)
        {
            if(CurrUnit.IsHarvester())
            {
                CurrUnit.CommandMove(MinX3, MinY3, 0);

                if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                {
                    UCS1IsMining = false;
                    p_Enemy1.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS1 nie wydobywa ");
                }

                if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                {
                    UCS2IsMining = false;
                    p_Enemy2.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS2 nie wydobywa ");
                }

                if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                {
                    UCS3IsMining = false;
                    p_Enemy3.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS3 nie wydobywa ");
                }

                if(UCS4IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy4.GetIFFNumber())
                {
                    UCS4IsMining = false;
                    p_Enemy4.EnableAIFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD("UCS4 nie wydobywa ");
                }
            }

            //czy wybudowano w bazie 3
            if(!BuildRP3)
            {
                //sprawdzenie czy odpowiednie budynki na tym terenie
                if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding())
                {
                    BuildingType = CurrUnit.GetBuildingType();

                    if(BuildingType == buildingPowerPlant) //wykryl elektrownie
                    {
                        IPP3 = IPP3 + 1;
                        //TraceD(" El. w UCS3 ");
                        //TraceD(IPP3);
                        //TraceD(" ");
                    }

                    if(IIR3 && BuildingType == buildingMiningRefinery) //wykryl
rafinerie
                    {
                        IIR3 = 1; //wybudowano kopalnie
                    }
                }
            }
        }
    }
}

```

```

        //TraceD(" Wybudowano kopalnie w UCS3 ");
    }

    //jest juz rafineria i co najmniej 3 elektrownie
    if(IRS && IPP3>fieldsCoveredByPP2 && !BuildRP3)
    {
        BuildRP3 = true;
        SetGoalState(goalBuildSouthEast, goalAchieved);
        TraceD(" Wybudowano to co trzeba w UCS3 ");
    }
}

y3 = y3 + 1;
if(y3 >= MaxY3) //koniec sprawdzania obszaru
{
    y3 = MinY3+1;
    IPP3 = 0;

    //next time check next base
    CheckBase = 3;
}

//----- Base UCS4 -----
if(CheckBase == 3)
{
    //petla y4

    for(x=MinX4+1; x<MaxX4; x=x+1)
    {
        CurrUnit = GetUnit(x, y4, 0);

        //sprawdzenie jest to koniecne czy UCS nie wydobywa na tym
        terenie - jesli tak to niech nie wydobywa
        if(UCS1IsMining || UCS2IsMining || UCS3IsMining ||

        UCS4IsMining)
        {
            if(CurrUnit.IsHarvester())
            {
                CurrUnit.CommandMove(MinX4, MinY4, 0);

                if(UCS1IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy1.GetIFFNumber())
                {
                    UCS1IsMining = false;
                    p_Enemy1.EnableAllFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD(" UCS1 nie wydobywa ");
                }

                if(UCS2IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy2.GetIFFNumber())
                {
                    UCS2IsMining = false;
                    p_Enemy2.EnableAllFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD(" UCS2 nie wydobywa ");
                }

                if(UCS3IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy3.GetIFFNumber())
                {
                    UCS3IsMining = false;
                    p_Enemy3.EnableAllFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD(" UCS3 nie wydobywa ");
                }

                if(UCS4IsMining && CurrUnit.GetIFFNumber() ==
p_Enemy4.GetIFFNumber())
                {
                    UCS4IsMining = false;
                    p_Enemy4.EnableAllFeatures(aiBuildMiningBuildings |
aiBuildMiningUnits | aiControlMiningUnits,false);
                    TraceD(" UCS4 nie wydobywa ");
                }
            }
        }
    }
}

//czy wybudowano w bazie 4
if(!BuildRP4)
{
    //sprawdzenie czy odpowiednie budynki na tym terenie
}
}

if(CurrUnit.GetIFFNumber() == p_Player.GetIFFNumber() &&
CurrUnit.IsBuilding()) //czy nasz budynek
{
    BuildingType = CurrUnit.GetBuildingType();

    if(BuildingType == buildingPowerPlant) //wykryl elektrownie
    {
        IPP4 = IPP4 + 1;
        //TraceD(" El. w UCS4 ");
        //TraceD("IPP4");
        //TraceD(" ");
    }
}

if(!IR4 && BuildingType == buildingMiningRefinery) //wykryl
rafinerie
{
    IR4 = 1; //wybudowano kopalnie
    //TraceD(" Wybudowano kopalnie w UCS4 ");

}

//jest juz rafineria i co najmniej 3 elektrownie
if(IR4 && IPP4>fieldsCoveredByPP2 && !BuildRP4)
{
    BuildRP4 = true;
    SetGoalState(goalBuildSouthWest, goalAchieved);
    TraceD(" Wybudowano to co trzeba w UCS4 ");
}

y4 = y4 + 1;
if(y4 >= MaxY4) //koniec sprawdzania obszaru
{
    y4 = MinY4+1;
    IPP4 = 0;

    //next time check next base
    CheckBase = 0;
}

//----- End checking bases -----
}

event EndMission()
{
    if(DestroyAliUCAchieved && BuildAllRP && CollectMoneyAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney) +
p_Player.GetMoney());
        p_Player.SetMoney();
    }
}

mission620.ec
mission "translateMission620"
{
    consts
    {
        goalDestroyUCS1 = 0;
        goalDestroyUCS2 = 1;
        goalDestroyUCS3 = 2;
        goalDestroyUCS4 = 3;
        goalDefendCC = 4;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;
    player p_Enemy4;

    player p_Player;

    unitex uCC1;
    unitex uCC2;
    unitex uCC3;
    unitex uCC4;

    int bCheckEndMission;

    state Initialize;
}

```

```

state ShowBriefing;
state Working;
state EndMissionFailed;
state EndGameSuccess;
state Nothing;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(goalDestroyUCS1,"translateGoal620a");
    RegisterGoal(goalDestroyUCS2,"translateGoal620b");
    RegisterGoal(goalDestroyUCS3,"translateGoal620c");
    RegisterGoal(goalDestroyUCS4,"translateGoal620d");
    RegisterGoal(goalDefendCC,"translateGoal620e");

    EnableGoal(goalDestroyUCS1,true);
    EnableGoal(goalDestroyUCS2,true);
    EnableGoal(goalDestroyUCS3,true);
    EnableGoal(goalDestroyUCS4,true);
    EnableGoal(goalDefendCC,true);

    //----- Players -----
    p_Player = GetPlayer(3);

    p_Enemy1 = GetPlayer(1);
    p_Enemy2 = GetPlayer(5);
    p_Enemy3 = GetPlayer(7);
    p_Enemy4 = GetPlayer(9);

    //----- AI -----
    if(GetDifficultyLevel() == 0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy3.LoadScript("single\singleEasy");
        p_Enemy4.LoadScript("single\singleEasy");
    }

    if(GetDifficultyLevel() == 1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy3.LoadScript("single\singleMedium");
        p_Enemy4.LoadScript("single\singleMedium");
    }

    if(GetDifficultyLevel() == 2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy3.LoadScript("single\singleHard");
        p_Enemy4.LoadScript("single\singleHard");
    }

    p_Enemy1.EnableAIFeatures(aiRush,false);
    p_Enemy2.EnableAIFeatures(aiRush,false);
    p_Enemy3.EnableAIFeatures(aiRush,false);
    p_Enemy4.EnableAIFeatures(aiRush,false);

    p_Enemy2.SetAlly(p_Enemy1);
    p_Enemy3.SetAlly(p_Enemy1);
    p_Enemy4.SetAlly(p_Enemy1);

    p_Enemy3.SetAlly(p_Enemy2);
    p_Enemy4.SetAlly(p_Enemy2);

    p_Enemy4.SetAlly(p_Enemy3);

    p_Enemy1.EnableAIFeatures(aiBuildSuperAttack, true);
    p_Enemy2.EnableAIFeatures(aiBuildSuperAttack, true);
    p_Enemy3.EnableAIFeatures(aiBuildSuperAttack, true);
    p_Enemy4.EnableAIFeatures(aiBuildSuperAttack, true);

    p_Enemy2.CopyResearches(p_Enemy1);
    p_Enemy3.CopyResearches(p_Enemy1);
    p_Enemy4.CopyResearches(p_Enemy1);

    //----- Money -----
    p_Player.SetMoney(40000);

    p_Enemy1.SetMoney(10000);
    p_Enemy2.SetMoney(10000);
    p_Enemy3.SetMoney(10000);
    p_Enemy4.SetMoney(10000);
}

p_Player.EnableCommand(commandSoldBuilding,true); // 1st
tab
//----- Timers -----
SetTimer(0, 20);
SetTimer(1, 20);
SetTimer(2, 1)/xxmd
//----- Variables -----
uCC1 = GetUnit(GetPointX(0),GetPointY(0),0);
uCC2 = GetUnit(GetPointX(1),GetPointY(1),0);
uCC3 = GetUnit(GetPointX(2),GetPointY(2),0);
uCC4 = GetUnit(GetPointX(3),GetPointY(3),0);
//----- Buildings -----
p_Player.EnableBuilding("LCBSR",false); //Centrum wydobywczo-transportowe
we
p_Player.EnableBuilding("LCBRC",false);
p_Player.EnableResearch("RES_LC_WARTILLERY",true); //Artyleria plazmowa

p_Enemy1.EnableResearch("RES_UCS_MB2",true);
p_Enemy1.EnableResearch("RES_UCS_WAPB1",true);

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
12, 0, 45, 0);
return ShowBriefing, 60;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing620", p_Player.GetName());
    return Working, 20;
}

//-----
state Working
{
    if(!bCheckEndMission)
        return Working;

    bCheckEndMission=false;
    //-----mission failed-----
    if(!uCC1.IsLive() || !uCC2.IsLive() || !uCC3.IsLive() || !uCC4.IsLive())
    {
        SetGoalState(goalDefendCC,goalFailed);
        AddBriefing("translateFailed620", p_Player.GetName());
        return EndMissionFailed;
    }

    //---- check 1 goal - destroy UCS ----
    if(GetGoalState(goalDestroyUCS1)==goalAchieved && 
    lp_Enemy1.GetNumberOfBuildings()==
    {
        SetGoalState(goalDestroyUCS1,goalAchieved);
        p_Enemy1.EnableAIFeatures(aiEnabled,false);
    }
    if(GetGoalState(goalDestroyUCS2)==goalAchieved && 
    lp_Enemy2.GetNumberOfBuildings()==
    {
        SetGoalState(goalDestroyUCS2,goalAchieved);
        p_Enemy2.EnableAIFeatures(aiEnabled,false);
    }
    if(GetGoalState(goalDestroyUCS3)==goalAchieved && 
    lp_Enemy3.GetNumberOfBuildings()==
    {
        SetGoalState(goalDestroyUCS3,goalAchieved);
        p_Enemy3.EnableAIFeatures(aiEnabled,false);
    }
    if(GetGoalState(goalDestroyUCS4)==goalAchieved && 
    lp_Enemy4.GetNumberOfBuildings()==
    {
        SetGoalState(goalDestroyUCS4,goalAchieved);
        p_Enemy4.EnableAIFeatures(aiEnabled,false);
    }

    //---- All goals achieved -----
    if(GetGoalState(goalDestroyUCS1)==goalAchieved && 
    GetGoalState(goalDestroyUCS2)==goalAchieved && 
    GetGoalState(goalDestroyUCS3)==goalAchieved && 
    GetGoalState(goalDestroyUCS4)==goalAchieved)
    {
        SetGoalState(goalDefendCC,goalAchieved);
        p_Player.SetScriptData(4,2);
        AddBriefing("translateAccomplished620");
    }
}

```

```

        return EndGameSuccess;
    }
    return Working;
}

//-----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state EndGameSuccess
{
    EnableNextMission(0,3);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
}

```

mission621.ec

```

mission "translateMission621"
{
    consts
    {
        scriptFieldMoney=9;
        goalDestroyUCSHarvesters = 0;
        goalDestroyUCSBase = 1;
    }

    player p_Enemy;
    player p_Player;

    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Working;
    state Nothing;
}

//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(goalDestroyUCSHarvesters,"translateGoal621a");
    RegisterGoal(goalDestroyUCSBase,"translateGoal621b");

    EnableGoal(goalDestroyUCSHarvesters,true);
    EnableGoal(goalDestroyUCSBase,true);

    //----- Players -----
    p_Player = GetPlayer(3);
    p_Enemy = GetPlayer(1);

    //----- AI -----
    if(GetDifficultyLevel()==0)
        p_Enemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        p_Enemy.LoadScript("single\singleHard");

    p_Enemy.EnableAIFeatures(aiControlOffense,false);
    p_Enemy.EnableAIFeatures(aiBuildMiningUnits,false);
}

//----- Money -----
p_Player.SetMoney(10000);

```

```

p_Enemy.SetMoney(0);

if(GetDifficultyLevel()==0)
{
    KillArea(2,GetPointX(0), GetPointY(0), 0,1);
    KillArea(2,GetPointX(1), GetPointY(1), 0,1);
    KillArea(2,GetPointX(2), GetPointY(2), 0,1);
    KillArea(2,GetPointX(3), GetPointY(3), 0,1);
}
if(GetDifficultyLevel()==1)
{
    KillArea(2,GetPointX(1), GetPointY(1), 0,1);
    KillArea(2,GetPointX(2), GetPointY(2), 0,1);
}

//----- Buildings -----
p_Player.EnableBuilding("LCBFB",false);
p_Player.EnableBuilding("LCBPP",false);
p_Player.EnableBuilding("LCBPP2",false);
p_Player.EnableBuilding("LCBSB",false);
p_Player.EnableBuilding("LCBBA",false);
p_Player.EnableBuilding("LCBMP",false);
p_Player.EnableBuilding("LCBSR",false);
p_Player.EnableBuilding("LCBRC",false);
p_Player.EnableBuilding("LCBAB",false);
p_Player.EnableBuilding("LCBGA",false);
p_Player.EnableBuilding("LCBDE",false);
p_Player.EnableBuilding("LCBHQ",false);
p_Player.EnableBuilding("LCBNE",false);
p_Player.EnableBuilding("LCBSD",false);
p_Player.EnableBuilding("LCBWC",false);
p_Player.EnableBuilding("LCBSST",false);
p_Player.EnableBuilding("LCBLZ",false);

p_Player.EnableBuilding("LCBUJ",false);
p_Player.EnableBuilding("LCBN1",false);
p_Player.EnableBuilding("CLASERWALL",false);
p_Player.EnableCommand(commandGoldBuilding,true);

p_Player.EnableResearch("RES_LCUUT",true); //621 add
p_Player.AddResearch("RES_LCUUT"); //621 add

bCheckEndMission=false;

SetTimer(0,100);
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),12,0,4
5,0);
return ShowBriefing,1;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing621a", p_Player.GetName());
    EnableNextMission(0,true);
    return Working, 20;
}

//-----
state Working
{
    if(GetGoalState(goalDestroyUCSHarvesters)==goalNotAchieved)

SetConsoleText("translateMessage621",p_Enemy.GetNumberOfUnits(unitHarvester
| chassisAny));
    if(GetGoalState(goalDestroyUCSHarvesters)==goalAchieved &&
    GetGoalState(goalDestroyUCSBase)==goalAchieved)
    {
        AddBriefing("translateAccomplished621", p_Player.GetName());
        EnableEndMissionButton(true);
        return Nothing;
    }
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event Timer0()
{
    if(bCheckEndMission)
    {
        bCheckEndMission=false;
    }
}

```

```

        if(GetGoalState(goalDestroyUCSHarvesters)==goalNotAchieved &&
lp_Enemy.GetNumberOfUnits(unitHarvester (chassisAny))
        {
            SetConsoleText("");
            SetGoalState(goalDestroyUCSHarvesters,goalAchieved);
            AddBriefing("translateBriefing621b", p_Player.GetName());
            p_Player.EnableBuilding("LCBLZ",true);
            p_Player.SetMoney(10000);
            p_Enemy.SetMoney(20000);
        }

        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            AddBriefing("translateFailed621", p_Player.GetName());
            EndMission(false);
        }

        if(GetGoalState(goalDestroyUCSBase)==goalNotAchieved &&
lp_Enemy.GetNumberOfBuildings())
        {
            SetGoalState(goalDestroyUCSBase,goalAchieved);
            AddBriefing("translateBriefing621c", p_Player.GetName());
        }
    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    if(GetGoalState(goalDestroyUCSHarvesters)==goalAchieved &&
GetGoalState(goalDestroyUCSBase)==goalAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission622.ec
//XXX tens skrypt nie dzia³a
mission "translateMission622"
//Prepare way for convoy
consts
{
    scriptFieldMoney=9;
    goalEscortConvoy = 0;
    goalAllConvoySurvived=1;
}

player p_Player;
player p_Convoy;

unitex p_ConvoyCraft1;
unitex p_ConvoyCraft2;
unitex p_ConvoyCraft3;
unitex p_ConvoyCraft4;
unitex p_ConvoyCraft5;
unitex p_ConvoyCraft6;

int bCheckEndMission;
int nMaxTimer;

//-----
state Initialize;
state ShowBriefing;
state BuildingWay;
state OnTheWay;
state Final1 //explanation
state Final2;//video
state Final3;
//-----
state Initialize
{



    int i;
    //----- Goals -----
    if(GetDifficultyLevel()==0)
        RegisterGoal(goalEscortConvoy,"translateGoal622a",2);
    if(GetDifficultyLevel()==1)
        RegisterGoal(goalEscortConvoy,"translateGoal622a",4);
    if(GetDifficultyLevel()==2)
        RegisterGoal(goalEscortConvoy,"translateGoal622a",6);

    RegisterGoal(goalAllConvoySurvived,"translateGoal622b");

    EnableGoal(goalEscortConvoy,true);

    //----- Players -----
    p_Player = GetPlayer(3);

    p_Convoy = GetPlayer(1);
    p_Convoy.EnableStatistics(false);
    p_Convoy = GetPlayer(5);
    p_Convoy.LoadScript("single/singleMedium");
    //----- AI -----
    p_Convoy.EnableAIFeatures(aiRejectAlliance,false);
    p_Player.SetAlly(p_Convoy);
    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Convoy.EnableAIFeatures(aiEnabled,false);

    //----- Money -----
    p_Player.SetMoney(20000);

    //----- Researches -----
    p_Player.EnableResearch("RES_LC_SHR1",true); //622

    //----- Buildings -----
    p_Player.EnableCommand(commandSoldBuilding,false);
    p_Player.EnableCommand(commandAutodestruction,false);
    p_Player.EnableBuilding("LCBF",false);
    p_Player.EnableBuilding("LCBP",false);
    p_Player.EnableBuilding("LCBP2",false);
    p_Player.EnableBuilding("LCBS",false);
    p_Player.EnableBuilding("LCBA",false);
    p_Player.EnableBuilding("LCBR",false);
    p_Player.EnableBuilding("LCBSR",false);
    p_Player.EnableBuilding("LCBR",false);
    p_Player.EnableBuilding("LCBABA",false);
    p_Player.EnableBuilding("LCBGA",false);
    p_Player.EnableBuilding("LCBDE",false);
    p_Player.EnableBuilding("LCBNE",false);
    p_Player.EnableBuilding("LCBHQ",false);
    p_Player.EnableBuilding("LCBART",false);
    p_Player.EnableBuilding("LCBUC",false);
    p_Player.EnableBuilding("LCBSD",false);
    p_Player.EnableBuilding("LCBWC",false);
    p_Player.EnableBuilding("LCBSS",false);
    p_Player.EnableBuilding("LCBLZ",false);
    p_Player.EnableBuilding("LCBEN1",false);

    //----- Units -----
    p_ConvoyCraft1 = GetUnit(GetPointX(1),GetPointY(1),0);
    p_ConvoyCraft2 = GetUnit(GetPointX(2),GetPointY(2),0);
    p_ConvoyCraft3 = GetUnit(GetPointX(3),GetPointY(3),0);
    p_ConvoyCraft4 = GetUnit(GetPointX(4),GetPointY(4),0);
    p_ConvoyCraft5 = GetUnit(GetPointX(5),GetPointY(5),0);
    p_ConvoyCraft6 = GetUnit(GetPointX(6),GetPointY(6),0);
    //----- Artefacts -----
    for(i=10;i<=50;i=i+1)
    {
        CreateArtefact("NEAMINE",GetPointX(i),GetPointY(i),GetPointZ(i),i,artefactSpecialAI
Other);
    }
    //----- Timers -----
    SetTimer(0,100);
    //----- Camera -----
    CallCamera();

    p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    EnableNextMission(0,true);
    if(GetDifficultyLevel()==0)
    {

```

```

AddBriefing("translateBriefing622a",p_Player.GetName(),2);
nMaxTimer=60*7;
}

if(GetDifficultyLevel()==1)
{
    AddBriefing("translateBriefing622a",p_Player.GetName(),4);
    nMaxTimer=60*5;
}

if(GetDifficultyLevel()==2)
{
    AddBriefing("translateBriefing622a",p_Player.GetName(),6);
    nMaxTimer=60*3;
}

if(GetDifficultyLevel()!=2)
    ShowArea(8,GetPointX(0),GetPointY(0),0,128);
return BuildingWay,20;
}
//-----
state BuildingWay
{
    SetConsoleText("translateMessage622",nMaxTimer);
    nMaxTimer=nMaxTimer-1;
    if(!nMaxTimer)
    {
        SetConsoleText("");
        AddBriefing("translateBriefing622b",p_Player.GetName());
        return OnTheWay;
    }
    return BuildingWay,20;
}
//-----
state OnTheWay
{
    int nGoToPoint;
    int bMakeStep;
    int nTrucksFinished;
    int nTrucksAlive;
    int j;

    bMakeStep=false;

    p_ConvoyCraft1.CommandMove(GetPointX(0),GetPointY(0));
    p_ConvoyCraft2.CommandMove(GetPointX(0),GetPointY(0));
    p_ConvoyCraft3.CommandMove(GetPointX(0),GetPointY(0));
    p_ConvoyCraft4.CommandMove(GetPointX(0),GetPointY(0));
    p_ConvoyCraft5.CommandMove(GetPointX(0),GetPointY(0));
    p_ConvoyCraft6.CommandMove(GetPointX(0),GetPointY(0));

    nTrucksAlive=0;

    if(p_ConvoyCraft1.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft2.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft3.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft4.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft5.IsLive())nTrucksAlive=nTrucksAlive+1;
    if(p_ConvoyCraft6.IsLive())nTrucksAlive=nTrucksAlive+1;

    nTrucksFinished=0;

    if(p_ConvoyCraft1.IsLive() &&
p_ConvoyCraft1.DistanceTo(GetPointX(0),GetPointY(0)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft2.IsLive() &&
p_ConvoyCraft2.DistanceTo(GetPointX(0),GetPointY(0)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft3.IsLive() &&
p_ConvoyCraft3.DistanceTo(GetPointX(0),GetPointY(0)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft4.IsLive() &&
p_ConvoyCraft4.DistanceTo(GetPointX(0),GetPointY(0)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft5.IsLive() &&
p_ConvoyCraft5.DistanceTo(GetPointX(0),GetPointY(0)) <
7)nTrucksFinished=nTrucksFinished+1;
    if(p_ConvoyCraft6.IsLive() &&
p_ConvoyCraft6.DistanceTo(GetPointX(0),GetPointY(0)) <
7)nTrucksFinished=nTrucksFinished+1;

    if(
        nTrucksAlive==nTrucksFinished &&
        ((GetDifficultyLevel()==0 && nTrucksFinished>1) || |
        ((GetDifficultyLevel()==1 && nTrucksFinished>3) || |
        ((GetDifficultyLevel()==2 && nTrucksFinished>5))) )
    {
        SetGoalState(goalEscortConvoy, goalAchieved);

        if(nTrucksFinished==6)
        {
            p_Player.EnableBuilding("LCBLZ",true);
            p_Player.SetMoney(p_Player.GetMoney()+2000);
            p_Convoy.GiveAllUnitsTo(p_Player);
            EnableGoal(goalAllConvoySurvived,true);
            SetGoalState(goalAllConvoySurvived, goalAchieved);
            AddBriefing("translateAccomplished622b",p_Player.GetName());
        }
        else
            AddBriefing("translateAccomplished622a",p_Player.GetName());

        return Final1,100;
    }
    return OnTheWay,200;
}
//-----
state Final1
{
    AddBriefing("translateExplanation622",p_Player.GetName());
    return Final2,1;
}
//-----
state Final2
{
    ShowVideo("cutscene3");
    return Final3,20;
}
//-----
state Final3
{
    EnableEndMissionButton(true);
    return Final3,10000;
}
//-----
event Timer0() //wolany co 100 cykli< ustawione funkcja SetTimer w state
Initialize
{
    int nLostConvoyCrafts;

    if(!bCheckEndMission) return;

    bCheckEndMission=false;

    if(GetGoalState(goalEscortConvoy)==goalNotAchieved)
    {
        if((p_ConvoyCraft1.IsLive())&|nLostConvoyCrafts==1;
        if((p_ConvoyCraft2.IsLive())&|nLostConvoyCrafts==1;
        if((p_ConvoyCraft3.IsLive())&|nLostConvoyCrafts==1;
        if((p_ConvoyCraft4.IsLive())&|nLostConvoyCrafts==1;
        if((p_ConvoyCraft5.IsLive())&|nLostConvoyCrafts==1;
        if((p_ConvoyCraft6.IsLive())&|nLostConvoyCrafts==1;

        if(
            ((GetDifficultyLevel()==0 && nLostConvoyCrafts>4) || |
            ((GetDifficultyLevel()==1 && nLostConvoyCrafts>2) || |
            ((GetDifficultyLevel()==2 && nLostConvoyCrafts>0)))
        {
            SetGoalState(goalEscortConvoy, goalFailed);
            AddBriefing("translateFailed622a",p_Player.GetName());
            EnableEndMissionButton(true,false);
            state Final1;
        }
    }

    if((p_Player.GetNumberOfUnits() &&p_Player.GetNumberOfBuildings())
    {
        if(GetGoalState(goalEscortConvoy)==goalAchieved)
        {
            AddBriefing("translateFailed622b",p_Player.GetName());
            EndMission(false);
        }
        else
        {
            EndMission(true);
        }
    }
}
//-----
event UnitDestroyed(unit_u_Unit)
{
    bCheckEndMission=true;
}

```

```

    }
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}
//-----
event EndMission()
{
    if(GetGoalState(goalEscortConvoy)==goalAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}
//-----
event Artefact(int alD,player piPlayer)
{
    KillArea(255,GetPointX(alD),GetPointY(alD),GetPointZ(alD),1);
    return false; //nie usuwa sie
}
}

TutorialLCmis.ec
mission "translateTutorialLC"
{
    consts
    {
        buildPowerPlant = 0;
        upgradePowerPlant = 1;
        buildBattery = 2;
        buildMainBase = 3;
        buildMine = 4;
        buildArmy = 5;
        findEnemy = 6;
        destroyEnemy = 7;
    }
    player p_EnemyUCS;
    player p_Player;

    int nUCSUnitsCount;
    int nBuildingCount;
    int tmpCounter;
    int nUnitCount;
    //*****
    state Initialize;
    state Start;
    state MoveCamera;
    state BuildPowerPlant;
    state UpgradePowerPlant;
    state BuildBattery;
    state BuildMainBase;
    state BuildMine;
    state BuildArmy;
    state More;
    state EndState;
    state Initialize
    {

        RegisterGoal(buildPowerPlant,"translateGoalTutorialLC_PP");
        RegisterGoal(upgradePowerPlant,"translateGoalTutorialLC_PP_UPG");
        RegisterGoal(buildBattery,"translateGoalTutorialLC_BA");
        RegisterGoal(buildMainBase,"translateGoalTutorialLC_BE");
        RegisterGoal(buildMine,"translateGoalTutorialLC_MI");
        RegisterGoal(buildArmy,"translateGoalTutorialLC_tanks");
        RegisterGoal(findEnemy,"translateGoalTutorialLC_findEnemy");
        RegisterGoal(destroyEnemy,"translateGoalTutorialLC_destroyEnemy");
        nUnitCount=0;
        p_EnemyUCS=GetPlayer(1);
        p_Player=GetPlayer(3);

        p_EnemyUCS.SetMoney(10000);
        p_Player.SetMoney(20000);

        p_EnemyUCS.EnableAIFeatures(aiEnabled,false);
        p_Player.EnableAIFeatures(aiEnabled,false);

        p_Player.EnableResearch("RES_MM2",false);
        //p_Player.EnableResearch("RES_LC_WCH2",false);
        p_Player.EnableResearch("RES_LC_ACH2",false);
        //p_Player.EnableResearch("RES_LC_WSR1",false);
    }

    p_Player.EnableResearch("RES_LC_WSR2",false);
    p_Player.EnableResearch("RES_LC_AS1",false);
    //p_Player.EnableResearch("RES_LC_WSL1",false);
    p_Player.EnableResearch("RES_LC_WSL2",false);
    p_Player.EnableResearch("RES_LC_WSS1",false);
    p_Player.EnableResearch("RES_LC_WMR1",false);
    p_Player.EnableResearch("RES_LC_WHL1",false);
    p_Player.EnableResearch("RES_LC_WHS1",false);

    p_Player.EnableResearch("RES_LC_WAS1",false);
    //p_Player.EnableResearch("RES_LC_UMO2",false);
    p_Player.EnableResearch("RES_LC_UCR1",false);
    p_Player.EnableResearch("RES_LC_UCU1",false);

    p_Player.EnableResearch("RES_LC_ULU3",false);
    p_Player.EnableResearch("RES_LC_UMO3",false);
    p_Player.EnableResearch("RES_LC_UME1",false);
    p_Player.EnableResearch("RES_LC_UBO1",false);

    p_Player.EnableResearch("RES_LC_BMD",false);
    p_Player.EnableResearch("RES_LC_BHD",false);
    p_Player.EnableResearch("RES_LC_SDIDEF",false);
    //p_Player.EnableResearch("RES_LC_SGen",false);
    p_Player.EnableResearch("RES_LC_MGen",false);
    p_Player.EnableResearch("RES_LC_MGen",false);
    p_Player.EnableResearch("RES_LC_SHR1",false);
    p_Player.EnableResearch("RES_LC_R61",false);
    p_Player.EnableResearch("RES_LC_SO1",false);
    p_Player.EnableResearch("RES_LC_SDIDEF",false);
    p_Player.EnableResearch("RES_LC_BWC",false);

    p_Player.EnableBuilding("LCBPP",false);
    p_Player.EnableBuilding("LCBBF",false);
    p_Player.EnableBuilding("LCBBA",false);
    p_Player.EnableBuilding("LCBMR",false);
    p_Player.EnableBuilding("LCBSP",false);
    p_Player.EnableBuilding("LCBRC",false);
    p_Player.EnableBuilding("LCBAB",false);
    p_Player.EnableBuilding("LCBGA",false);
    p_Player.EnableBuilding("LCBDE",false);
    p_Player.EnableBuilding("LCBHQ",false);
    p_Player.EnableBuilding("LCBSD",false);
    p_Player.EnableBuilding("LCBWC",false);
    p_Player.EnableBuilding("LCBLZ",false);
    p_Player.EnableBuilding("LCLASERWALL",false);

    SetTimer(0,100);

    LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,0,0);
    nBuildingCount=0;
    tmpCounter=0;
    return Start,100;
}
//-----

state Start
{
    AddBriefing("translateTutorialLC_moveCamera");
    nUCSUnitsCount=p_EnemyUCS.GetNumberOfUnits()/2;
    return MoveCamera,500;
}
//-----

state MoveCamera
{
    EnableGoal(buildPowerPlant,true);
    p_Player.EnableBuilding("LCBPP",true);
    AddBriefing("translateTutorialLC_PP");
    return BuildPowerPlant,100;
}
//-----

state BuildPowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingSolarPower))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildPowerPlant,goalAchieved);
        EnableGoal(upgradePowerPlant,true);
        AddBriefing("translateTutorialLC_PP_UPG");
        return UpgradePowerPlant,100;
    }
    return BuildPowerPlant,100;
}
//-----
```

```

state UpgradePowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingSolarBattery)>3)
    {
        p_Player.EnableBuilding("LCBBA",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(upgradePowerPlant,goalAchieved);
        EnableGoal(buildBattery,true);
        if(p_Player.GetNumberOfBuildings(buildingEnergyBattery))
            AddBriefing("translateTutorialLC_BA");
        if(p_Player.GetNumberOfUnits())
    }

    p_Player.CreateUnitEx(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
    0,null,"LCULU1","LWCH1",null,null,null,0);
    return BuildBattery,100;
}
//-----

state BuildBattery
{
    if(p_Player.GetNumberOfBuildings(buildingEnergyBattery))
    {
        p_Player.EnableBuilding("LCBBF",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildBattery,goalAchieved);
        EnableGoal(buildMainBase,true);
        if(p_Player.GetNumberOfBuildings(buildingBaseFactory))
            AddBriefing("translateTutorialLC_BF");
        return BuildMainBase,100;
    }
    if(p_Player.GetNumberOfUnits())
}

p_Player.CreateUnitEx(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
0,null,"LCULU1","LWCH1",null,null,null,0);
if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=p_Player.GetNumberOfBuildings();
    AddBriefing("translateTutorialLC_BF_Fool");
}

return BuildBattery,50;
}
//-----

state BuildMainBase
{
    if(p_Player.GetNumberOfBuildings(buildingBaseFactory))
    {
        p_Player.EnableBuilding("LCBMR",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildMainBase,goalAchieved);
        EnableGoal(buildMine,true);
        p_EnemyUCS.LoadScript("single|singleEasy");
        p_EnemyUCS.EnableAIFeatures(aiEnabled,true);
        p_EnemyUCS.EnableAIFeatures(aiBuildBuildings,false);
        p_EnemyUCS.EnableAIFeatures(aiControlOffense,false);

        p_EnemyUCS.EnableAIFeatures(aiBuildTanks | aiBuildShips | aiBuildHelicopters,false);
    }

    if(p_Player.GetNumberOfBuildings(buildingMine))
        AddBriefing("translateTutorialLC_MI");
    return BuildMine,100;
}
if(p_Player.GetNumberOfUnits())
}

p_Player.CreateUnitEx(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
0,null,"LCULU1","LWCH1",null,null,null,0);
/* if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=p_Player.GetNumberOfBuildings();
    AddBriefing("translateTutorialLC_BF_Fool");
} */
return BuildMainBase,50;
}
//-----
```

```

state BuildMine
{
    if(p_Player.GetNumberOfBuildings(buildingMinningRefinery))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildMine,goalAchieved);

        EnableGoal(buildArmy,true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        AddBriefing("translateTutorialLC_tanks");
}

nUnitCount=p_Player.GetNumberOfUnits(chassisAmphibianTank | unitArmed);
if(nUnitCount>3)nUnitCount=3;
return BuildArmy,50;
}
if(p_Player.GetNumberOfUnits())
}

p_Player.CreateUnitEx(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),
0,null,"LCULU1","LWCH1",null,null,null,0);
if(p_Player.GetNumberOfBuildings(>nBuildingCount)
{
    nBuildingCount=nBuildingCount+1;
    AddBriefing("translateTutorialLC_MI_Fool");
}
return BuildMine,50;
}
//-----

state BuildArmy
{
}

if(p_Player.GetNumberOfUnits(chassisAmphibianTank | unitArmed)>=nUnitCount+
4)
{
    SetGoalState(buildArmy,goalAchieved);
    EnableGoal(findEnemy,true);
    AddBriefing("translateTutorialLC_findEnemy");
    return More,600;
}
return BuildArmy,100;
}
//-----

state More
{
    p_Player.EnableBuilding("LCBAB",true);
    p_Player.EnableBuilding("LCBGA",true);
    p_Player.EnableBuilding("LCBRC",true);
    p_Player.EnableBuilding("LCLASERWALL",true);
    AddBriefing("translateTutorialLC_more");
    return EndState,100;
}
//-----

state EndState
{
    return EndState,500;
}
//-----

event Timer0()
{
    if(GetGoalState(findEnemy)!=goalAchieved &&
p_Player.IsPointLocated(p_EnemyUCS.GetStartingPointX(),p_EnemyUCS.GetStartin
gPointY(),0))
    {
        SetGoalState(findEnemy,goalAchieved);
        EnableGoal(destroyEnemy,true);
        AddBriefing("translateTutorialLC_destroyEnemy");
    }
    if(GetGoalState(destroyEnemy)!=goalAchieved &&
lp_EnemyUCS.GetNumberOfUnits())
    {
        SetGoalState(destroyEnemy,goalAchieved);
        AddBriefing("translateTutorialLC_Victory");
    }
}
}
```

MP-UCS

CampaignUCSMP01.ec

campaign "translateCampaignUCSMP01"

```

{
    const
    {
        raceUCS=1;
        raceLC=3;
        mis410 = 30;
```

```

mis411 = 31;
mis412 = 32;
mis413 = 33;
mis414 = 34;
mis415 = 35;
mis416 = 36;
mis417 = 37;
mis418 = 38;
mis419 = 39;
mis420 = 40;

base1 = 45;
base2 = 46;
base3 = 47;

timeFlagWinter = 1;
baseFlag = 255;

}

int n_TimeFlag;
int n_MissionToLoad;

state Initialize;
state Start;
state Nothing;
state LoadNextMission;

state Initialize
{
    int i;
}

RegisterMission(base1,"IUCSbaseUCSMP01","UCS\\Missions\\UCSbase1script","","",
baseFlag, 0,90,0,0);
    RegisterMission(base2,
"UCSbaseLCMP01","UCS\\Missions\\UCSbase2script","","",baseFlag, 0, 90,0,0,0);
    RegisterMission(base3,
"UCSbaseLCMP01","UCS\\Missions\\UCSbase3script","","",baseFlag, 0, 90,0,0,0);
    // nr Ind script preBriefing
timeFlags x y d1 d2 d3 next1 next2 next3 next4
    RegisterMission(mis410, "I410", "UCS\\Missions\\Mission410",
"translateBriefingShort410", timeFlagWinter, 10,-80, 60, 60, mis412);
    RegisterMission(mis411, "I411", "UCS\\Missions\\Mission411",
"translateBriefingShort411", timeFlagWinter, 10,-80, 60, 60, mis412);
    RegisterMission(mis412, "I412", "UCS\\Missions\\Mission412",
"translateBriefingShort412", timeFlagWinter, 30,-70, 60, 60, mis413);
    RegisterMission(mis413, "I413", "UCS\\Missions\\Mission413",
"translateBriefingShort413", timeFlagWinter, 44, 0, 60, 60, mis415);

    RegisterMission(mis414, "I414", "UCS\\Missions\\Mission414",
"translateBriefingShort414", timeFlagWinter, 113,-45, 90, 90, 90);
    RegisterMission(mis415, "I415", "UCS\\Missions\\Mission415",
"translateBriefingShort415", timeFlagWinter, 136, 0, 120,120,120, mis416);

    RegisterMission(mis416, "I416", "UCS\\Missions\\Mission416",
"translateBriefingShort416", timeFlagWinter, 203,-45, 90, 90, mis417);
    RegisterMission(mis417, "I417", "UCS\\Missions\\Mission417",
"translateBriefingShort417", timeFlagWinter, 224, 0, 120,120,120, mis418);

    RegisterMission(mis418, "I418", "UCS\\Missions\\Mission418",
"translateBriefingShort418", timeFlagWinter, 316,-15, 90, 90, mis419);
    RegisterMission(mis419, "I419", "UCS\\Missions\\Mission419",
"translateBriefingShort419", timeFlagWinter, 316, 0, 120,120,120, mis420);
    RegisterMission(mis420, "I420", "UCS\\Missions\\Mission420",
"translateBriefingShort420", timeFlagWinter, 0, 90, 240,240,240);

n_TimeFlag = timeFlagWinter;

CreateGamePlayer(1,raceUCS,playerLocal,null);
CreateGamePlayer(3,raceLcplayerAI,null);

ForceKeepLoadProgress(4);
LoadBase(1,base1,1);
LoadBase(2,base2,3);/// ?????? dwa razy 3
LoadBase(3,base3,3);/// ?????? dwa razy 3
ForceCloseLoadProgress();
LoadMission(0,mis410);

SetActivePlayerAndWorld(1,0);//player, world
SetAvailableWorlds(1)//misja i baza UCS(swiat nr 1)
    EnableMission(mis413,true);
ActivateMissions(timeFlagWinter,true);
SetSeason(8);

}
return Start,10;
}
//-----
state Start
{
    //EnableChooseMissionButton(true);
    return Nothing,50;
}
//-----
state Nothing
{
    return Nothing, 0;
}
//-----
state LoadNextMission
{
    LoadMission(0, n_MissionToLoad);
    SetAvailableWorlds(1);
    return Nothing;
}
//-----
event StartMission(int iMission)
{
    EnableChooseMissionButton(false);
    TraceD(Mission);
    LoadMission(0,Mission);
    EnableMission(iMission,false);
}
//-----
event EnableNextMission(int iMission,int iNextNr,int bEnable)
{
    //if(iMission==mis413)return;///XXX DEMO
    if(bEnable<2)
        EnableMission(GetNextMission(iMission,iNextNr),bEnable);
    if(bEnable==2)//failed
        EndGame("Video\\Cutscene6.wd1");//video failed
    if(bEnable==3)//victory
        EndGame("Video\\Cutscene5.wd1");//video victory
}
//-----
event EndingMission(int iMission, int nResult)
{
    if(iMission==mis410 || iMission==mis411 || iMission==mis412)
    {
        SetAvailableWorlds(1|2);
    }
    if(iMission==mis412)
    {
        SendCustomEvent(0,1,0,0,0);//enable show briefing in Moon Base
    }
}
//-----
event EndMission(int iMission, int nResult) //
{
    if(nResult==true)
    {
        SetMissionState(iMission,stateAccomplished);
    }
    else
    {
        SetMissionState(iMission,stateFailed);
    }
    if(iMission==mis410)
    {
        n_MissionToLoad = mis411;
        ForceKeepBitmapAfterStatistic();//musi byc tutaj a nie przed LoadMission
        state LoadNextMission;
    }
    else if(iMission==mis411)
    {
        n_MissionToLoad = mis412;
        ForceKeepBitmapAfterStatistic();//musi byc tutaj a nie przed LoadMission
        state LoadNextMission;
    }
    else
        EnableChooseMissionButton(true);
}
//*****
```

TutorialUCS.ec

campaign "translateTutorialUCS"

```
{  
    state Initialize;  
    state Nothing;  
  
    state Initialize  
    {  
        CreateGamePlayer(1,raceUCS,playerLocal,null);  
        CreateGamePlayer(2,raceED,playerAI,null);  
  
        SetTime(100);  
  
        RegisterMission(0,!TutorialUCS,"UCS\\Missions\\TutorialUCSmis","","0,0,0,0,0,0);  
        LoadMission(0);  
        SetAvailableWorlds(1);  
        SetActivePlayerAndWorld(1,0);  
        return Nothing;  
    }  
    state Nothing  
    {  
        return Nothing,200;  
    }  
}
```

Missions

mission410.ec

mission "translateMission410"

```
{  
    consts  
    {  
        bringGrizzli1 = 0;  
        bringGrizzli2 = 1;  
        harvest50000 = 2;  
        DestroyLC = 3;  
        DestroyED = 4;  
    }  
  
    player p_Enemy1;/ED  
    player p_Enemy2;/LC  
    player p_Player;  
    player p_Ally1;/4  
    player p_Ally2;/6  
  
    unitex u_Grizzli1;  
    unitex u_Grizzli2;  
    unitex u_SpacePort;  
  
    int bCheckEndMission;  
    int i;  
  
    state Initialize;  
    state PlayTrackState;  
    state ShowBriefing;  
    state EnemyLocated;  
    state Working;  
    state Harvest;  
    state EndMissionTrue;  
    state EndMissionFailed;  
    state Nothing;  
  
    //-----  
    state Initialize  
    {  
  
        //----- Goals -----  
        RegisterGoal(bringGrizzli1,"translateGoal410a");  
        RegisterGoal(bringGrizzli2,"translateGoal410b");  
        RegisterGoal(harvest50000,"translateGoal410c");  
        RegisterGoal(DestroyLC, "translateGoal410d");//reward 25000  
        RegisterGoal(DestroyED, "translateGoal410e");//reward 25000  
  
        //---show goals on list---  
        EnableGoal(bringGrizzli1,true);  
        EnableGoal(bringGrizzli2,true);  
        EnableGoal(harvest50000,true);  
        EnableGoal(DestroyLC,false);  
        EnableGoal(DestroyED,false);  
  
        //----- Players -----  
    }
```

```
p_Player = GetPlayer(1); //ja czyli UCS  
p_Enemy1 = GetPlayer(2); //ED  
p_Enemy2 = GetPlayer(3); //LC  
p_Ally1 = GetPlayer(4); //gracz do przejeca  
p_Ally2 = GetPlayer(6); //gracz do przejeca  
  
//----- AI -----  
p_Enemy1.LoadScript("single\\singleMedium");  
p_Enemy2.LoadScript("single\\singleMedium");  
p_Ally1.LoadScript("single\\singleMedium");  
p_Ally2.LoadScript("single\\singleMedium");  
  
p_Ally1.EnableAIFeatures(aiRejectAlliance, false);  
p_Ally2.EnableAIFeatures(aiRejectAlliance, false);  
p_Player.SetAlly(p_Ally1);  
p_Player.SetAlly(p_Ally2);  
p_Ally1.SetAlly(p_Ally2);  
p_Ally1.SetEnemy(p_Enemy1);  
p_Ally2.SetEnemy(p_Enemy2);  
p_Ally1.EnableAIFeatures(aiControlOffense, false);  
p_Ally2.EnableAIFeatures(aiControlOffense, false);  
//--- Grizzlies -----  
u_Grizzli1 = GetUnit(GetPointX(1),GetPointY(1),GetPointZ(1));  
u_Grizzli2 = GetUnit(GetPointX(2),GetPointY(2),GetPointZ(2));  
  
u_Grizzli1.SetUnitName("translateGrizzli1");  
u_Grizzli2.SetUnitName("translateGrizzli2");  
  
p_Player.AddUnitToSpecialTab(u_Grizzli1,true,-1);  
p_Player.AddUnitToSpecialTab(u_Grizzli2,true,-1);  
  
u_SpacePort = GetUnit(GetPointX(0),GetPointY(0),0);
```

```
/*CreateArtifact("NEACOMPUTER", GetPointX(1)+1, GetPointY(1),  
1,0,artefactSpecialAIother);  
CreateArtifact("NEASWITCH1", GetPointX(1)+2, GetPointY(1),  
1,0,artefactSpecialAIother);  
CreateArtifact("NEASWITCH2", GetPointX(1)+3, GetPointY(1),  
1,0,artefactSpecialAIother);  
CreateArtifact("NEAPLATE1", GetPointX(2)+1, GetPointY(2),  
1,0,artefactSpecialAIother);  
CreateArtifact("NEAPLATE2", GetPointX(2)+2, GetPointY(2),  
1,0,artefactSpecialAIother);*/
```

```
//----- Variables -----  
p_Player.EnableBuilding("UCSBWB", false); //Stocznia  
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu rudy  
p_Player.EnableBuilding("UCSCSD", false); //Laser antykatietowy
```

```
//----- Money -----  
p_Enemy1.SetMoney(25000);  
p_Enemy2.SetMoney(25000);  
p_Ally1.SetMoney(25000);  
p_Ally2.SetMoney(25000);  
p_Player.SetMoney(20000);
```

```
SetTimer(0, 100);
```

```
//----- Camera -----  
CallCamera();  
ShowArea(2,GetPointX(0),GetPointY(0)+10,0,10);  
ShowArea(2,GetPointX(0),GetPointY(0)+20,0,10);  
ShowArea(2,GetPointX(0),GetPointY(0)+30,0,10);  
ShowArea(2,GetPointX(0),GetPointY(0)+40,0,10);
```

```
p_Player.LookAt(GetPointX(1),GetPointY(1), 6, 0, 20, 1);  
p_Player.DelayedLookAt(GetPointX(1),GetPointY(1),6,1,20,1,100,0);  
//p_Player.LookAt(GetPointX(0), GetPointY(0)+40, 6, 0, 20, 0);  
//p_Player.DelayedLookAt(GetPointX(0),GetPointY(0),6,0,20,0,90,1);  
return PlayTrackState;3;
```

```
}
```

```
//-----  
state PlayTrackState  
{  
    PlayTrack("music\\ucsday_3.mp2");  
    return ShowBriefing,100;  
}
```

```
//-----  
state ShowBriefing  
{  
    AddBriefing("translateBriefing410a", p_Player.GetName()); //D:+name  
    //p_Player.LookAt(GetPointX(1),GetPointY(1), 6, 0, 20, 1);  
    //EnableEndMissionButton(true);/xmd usunac
```

284

```

        return EnemyLocated,1200://15s
    }

//-----
state EnemyLocated
{
    ShowArea(2,GetPointX(3),GetPointY(3),0,6,showAreaBuildings);//centrum
sterowania SunLight
    ShowArea(2,GetPointX(4),GetPointY(4),0,6,showAreaBuildings);//centrum
sterowania SunLight
    //p_Player.LookAt(GetPointX(3), GetPointY(3), 6, 0, 20, 0);
    EnableGoal(DestroyLC,true);
    EnableGoal(DestroyED,true);
    AddBriefing("translateBriefing410b", p_Player.GetName()); //D:+name
    return Working;
}

//-----
state Working
{
    if(u_Grizzly1.DistanceTo(GetPointX(0),GetPointY(0))<3 &&
    u_Grizzly1.GetLocationZ() == 1)
    {
        SetGoalState(bringGrizzly1,goalAchieved);
        u_Grizzly1.ChangePlayer(p_Ally);
    }

    if(u_Grizzly2.DistanceTo(GetPointX(0),GetPointY(0))<3 &&
    u_Grizzly2.GetLocationZ() == 1)
    {
        SetGoalState(bringGrizzly2,goalAchieved);
        u_Grizzly2.ChangePlayer(p_Ally);
    }

    if(p_Player.GetMoney()>=50000)
        SetGoalState(harvest50000,goalAchieved);
    else
        SetGoalState(harvest50000,goalNotAchieved);

    if(GetGoalState(bringGrizzly1)==goalAchieved &&
    GetGoalState(bringGrizzly2)==goalAchieved)
    {
        if(p_Player.GetMoney()>=50000)
            SetGoalState(harvest50000,goalAchieved);

        AddBriefing("translateAccomplished410",p_Player.GetName());//doprowadziles
grizzlich i uzbera' es forse
        p_Player.SetMoney(p_Player.GetMoney()-50000);
        return EndMissionTrue;
    }
    else
    {
        AddBriefing("translateBriefing410c",p_Player.GetName()); //uzniera
forse
        return Harvest;
    }
}

return Working;
}

//-----
state Harvest
{
    if(p_Player.GetMoney()>=50000)
    {
        SetGoalState(harvest50000,goalAchieved);
        AddBriefing("translateAccomplished410",p_Player.GetName());//uzbie
ra3ee forse
        p_Player.SetMoney(p_Player.GetMoney()-50000);
        return EndMissionTrue;
    }
    return Harvest;
}

//-----
state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state EndMissionTrue
{
    EndMission(true);
}

return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}
}

event Timer0()
{
    if(lbCheckEndMission) return;
    bCheckEndMission=false;

    if(p_Enemy2.GetNumberOfBuildings()<4)
    {
        p_Enemy2.EnableAIFeatures(aiEnabled, false);
    }
    if(lu_Grizzly1.IsLive())
    {
        AddBriefing("translateGrizzly1Killed",p_Player.GetName());
        state EndMissionFailed;
    }
    if(lu_Grizzly2.IsLive())
    {
        AddBriefing("translateGrizzly2Killed",p_Player.GetName());
        state EndMissionFailed;
    }
    if(lu_SpacePort.IsLive())
    {
        AddBriefing("translateFailed410", p_Player.GetName());
        state EndMissionFailed;
    }

    if(GetGoalState(DestroyLC)!=goalAchieved &&
    lp_Enemy2.GetNumberOfBuildings())
    {
        SetGoalState(DestroyLC, goalAchieved);
        p_Enemy2.EnableAIFeatures(aiEnabled, false);
        AddBriefing("translateBriefing410d", p_Player.GetName());
        p_Player.AddMoney(25000);
    }
    if(GetGoalState(DestroyED)!=goalAchieved &&
    lp_Enemy1.GetNumberOfBuildings())
    {
        SetGoalState(DestroyED, goalAchieved);
        AddBriefing("translateBriefing410e", p_Player.GetName());
        p_Player.AddMoney(25000);
    }
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

}

mission411.ec

mission "translateMission411"
{
    consts
    {
        RescueGrizzly1 = 0;
        RescueGrizzly2 = 1;
        LocateLCbase = 2;
        DestroyLC = 3;
        CaptureCPU = 4;
    }

    player p_Enemy;
    player p_Escort1;
    player p_Escort2;
    player p_Player;
    player p_Grizzly1;
    player p_Grizzly2;
    player p_Eleport;
}

```

```

unitex u_Grizzly1;
unitex u_Grizzly2;

int DestroyLCAchieved;
int RescueGrizzlyAchieved1;
int RescueGrizzlyAchieved2;
int CaptureCPUAchieved;
int LCBaseLocated;

int xLCBase;
int yLCBase;

int i;

state Initialize;
state ShowVideo;
state ShowBriefing;
state Working;
state Nothing;
state EndMissionFailed;
state EndMissionState;
//-----
state Initialize
{
    //----- Extra -----
    EnableGameFeature(lockResearchDialog,true);
    EnableGameFeature(lockConstructionDialog,true);
    EnableGameFeature(lockUpgradeWeaponDialog,true);

    DestroyLCAchieved = false;
    RescueGrizzlyAchieved1 = false;
    RescueGrizzlyAchieved2 = false;
    CaptureCPUAchieved = false;
    LCBaseLocated = false;

    //----- Goals -----
    RegisterGoal(RescueGrizzly1, "translateGoalRescueGrizzly1");
    RegisterGoal(RescueGrizzly2, "translateGoalRescueGrizzly2");
    RegisterGoal(LocateLCBase, "translateGoalLocateLCBase");
    RegisterGoal(DestroyLC, "translateGoalDestroyLC");
    RegisterGoal(CaptureCPU, "translateGoalCaptureCPU");

    //---show goals on list---
    EnableGoal(DestroyLCfalse);
    EnableGoal(RescueGrizzly1,true);
    EnableGoal(RescueGrizzly2,true);
    EnableGoal(LocateLCBase,true);
    EnableGoal(CaptureCPU,false);
}

//----- Players -----
p_Player = GetPlayer(1); //ja czyl UCS
p_Enemy = GetPlayer(3); //LC
p_Grizzly1 = GetPlayer(5); //gracz do przejcia
p_Grizzly2 = GetPlayer(4); //gracz do przejcia
p_Teleport = GetPlayer(6);
p_Escort1 = GetPlayer(8);
p_Escort2 = GetPlayer(7);

p_Grizzly1.EnableStatistics(false);
p_Grizzly2.EnableStatistics(false);
p_Escort1.EnableStatistics(false);
p_Escort2.EnableStatistics(false);
p_Teleport.EnableStatistics(false);

//----- AI -----
p_Enemy.LoadScript("single\\singleMedium");
p_Escort1.LoadScript("single\\singleMedium");
p_Escort2.LoadScript("single\\singleMedium");

//---Grizzly as neutral---
p_Grizzly1.EnableAIFeatures(aiControlOffense, false);

p_Grizzly1.SetNeutral(p_Player);
p_Player.SetNeutral(p_Grizzly1);

p_Grizzly1.SetNeutral(p_Escort1);
p_Escort1.SetNeutral(p_Grizzly1);

p_Grizzly2.EnableAIFeatures(aiControlOffense, false);

p_Grizzly1.SetNeutral(p_Escort2);
p_Escort2.SetNeutral(p_Grizzly1);

p_Grizzly2.SetNeutral(p_Escort2);
p_Escort2.SetNeutral(p_Grizzly2);

//--- don't attack Teleport please ---
p_Enemy.SetNeutral(p_Teleport);
p_Player.SetNeutral(p_Teleport);

//----- Variables -----
xLCBase = GetPointX(0);
yLCBase = GetPointY(0);

//---AI off-----
p_Enemy.EnableAIFeatures(aiControlOffense, false);
p_Enemy.EnableAIFeatures(aiControlDefense, false);
p_Enemy.EnableAIFeatures(aiBuildBuildings, false);

p_Escort1.EnableAIFeatures(aiControlOffense, false);
p_Escort1.EnableAIFeatures(aiControlDefense, false);

p_Escort2.EnableAIFeatures(aiControlOffense, false);
p_Escort2.EnableAIFeatures(aiControlDefense, false);

//----- Money -----
p_Enemy.SetMoney(5000);

p_Enemy.EnableResearch("RES_LC_SGen",true);
p_Enemy.EnableResearch("RES_LC_BMD",true);
p_Enemy.EnableResearch("RES_MCH2",true);

//---Artefacts : computers to capture ---
CreateArtifact("NEACOMPUTER", GetPointX(3), GetPointY(3),
GetPointZ(3), 0, artifactSpecialAIOther);

p_Escort1.SetAlly(p_Enemy);
p_Escort2.SetAlly(p_Enemy);

//---units---
u_Grizzly1 = GetUnit(GetPointX(1),GetPointY(1),GetPointZ(1));
u_Grizzly2 = GetUnit(GetPointX(2),GetPointY(2),GetPointZ(2));
u_Grizzly1.SetUnitName("translateGrizzly1");
u_Grizzly2.SetUnitName("translateGrizzly2");

u_Grizzly1.LoadSceneScript("Scripts\\Units\\Tank.ecomp");
u_Grizzly2.LoadSceneScript("Scripts\\Units\\Tank.ecomp");
//----Timers-
SetTimer(0, 100);

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
6, 0, 20, 0);
return ShowVideo;
}

//----- ShowVideo -----
state ShowVideo
{
    ShowVideo("Cutscene1");
    return ShowBriefing,100;
}

state ShowBriefing
{
    ShowArea(p_Player.GetIFF(), GetPointX(1), GetPointY(1), 0, 2);
    ShowArea(p_Player.GetIFF(), GetPointX(2), GetPointY(2), 0, 2);
    AddBriefing("translateBriefing411a", p_Player.GetName()); //D:+name
    MeteorRain(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(), 20, 500, 30000, 100, 1, 1);
    //EnableEndMissionButton(true); //XXXMD
    return Working;
}

//----- Working -----
state Working
{
    if(!p_Player.GetNumberOfUnits())
    {
        AddBriefing("translateFailed411", p_Player.GetName());
    }
}

```

```

        return EndMissionFailed;
    }

    //----- check goals -----
    //---capture LC base---

    if(GetGoalState(DestroyLC)==goalAchieved &&
    lp_Enemy.GetNumberOfBuildings())
    {
        SetGoalState(DestroyLC, goalAchieved);
        AddBriefing("translateBriefing411e", p_Player.GetName());//LC base
        has been destroyed
        DestroyLCAchieved = true;
    }

    //---free Grizzli1---
    if(GetGoalState(RescueGrizzli1)!= goalAchieved &&
    lp_Escort1.GetNumberOfUnits())
    {
        MeteorRain(p_Grizzli1.GetStartingPointX(),
        p_Grizzli1.GetStartingPointY(),20,100,30000,100,1,1);
        p_Grizzli1.GiveAllUnitsTo(p_Player);
        SetGoalState(RescueGrizzli1, goalAchieved);
        AddBriefing("translateBriefing411c", p_Player.GetName());//grizzli 1
    recovered
        p_Player.AddUnitToSpecialTab(u_Grizzli1,true, -1);
    }

    //---free Grizzli2---
    if(GetGoalState(RescueGrizzli2)!= goalAchieved &&
    lp_Escort2.GetNumberOfUnits())
    {
        MeteorRain(p_Grizzli2.GetStartingPointX(),
        p_Grizzli2.GetStartingPointY(),20,100,30000,100,1,1);
        p_Grizzli2.GiveAllUnitsTo(p_Player);
        SetGoalState(RescueGrizzli2, goalAchieved);
        AddBriefing("translateBriefing411d", p_Player.GetName()); //grizzli 2
    recovered
        p_Player.AddUnitToSpecialTab(u_Grizzli2,true, -1);
    }

    //---LC base located---
    if(!lCBaseLocated && p_Player.IsPointLocated(xLCBase, yLCBase))
    {
        SetGoalState(LocateLCBase, goalAchieved);
        EnableGoal(CaptureCPU,true);
        EnableGoal(DestroyLC,true);
        AddBriefing("translateBriefing411b", p_Player.GetName());
    }

//D+name
    LCBBaseLocated=true;
}

//-----check the goal-----
//--- misja wykonała całkowicie ---
if(DestroyLCAchieved && CaptureCPUAchieved&&
    GetGoalState(RescueGrizzli1)==goalAchieved&&
    GetGoalState(RescueGrizzli2)==goalAchieved)
{
    EnableNextMission(0,true);
    AddBriefing("translateAccomplished411",
    p_Player.GetName())//D+name

    return EndMissionState;
}

return Working;
}

state EndMissionFailed
{
    EnableNextMission(0,2);
    ShowVideo("");
    return Nothing;
}

state EndMissionState
{
    EndMission(true);
}

//----- Nothing -----
state Nothing
{
    return Nothing, 500;
}
//-----

event Timer0()
{
    // TraceD(p_Enemy.GetMoney());
    // TraceD("\n");
    if(!u_Grizzli1.IsLive())
    {
        AddBriefing("translateGrizzli1Killed",p_Player.GetName());
        state EndMissionFailed;
    }
    if(!u_Grizzli2.IsLive())
    {
        AddBriefing("translateGrizzli2Killed",p_Player.GetName());
        state EndMissionFailed;
    }
}

//---capture CPU-----
event Artefact(int alD,player piPlayer)
{
    if(piPlayer==p_Player) return false;
    CaptureCPUAchieved = true;
    SetGoalState(CaptureCPU, goalAchieved);
    ShowArea(p_Player.GetIFF(), GetPointX(3), GetPointY(3), 1, 200);
    AddBriefing("translateBriefing411f", p_Player.GetName());
    return true; //usuwa sie
}

}

mission412.ec
mission "translateMission412"
{
    consts
    {
        EscortVehicle = 0;
    }

    player p_TmpPlayer;
    player p_Enemy;
    player p_Neutral;
    player p_Player;

    unitex u_EscortedVehicle;

    int bCheckEndMission;
    int DestinationX;
    int DestinationY;
    int bStartRain1;
    int bStartRain2;
    int bStartRain3;
    int bStartRain4;

    unitex u_Grizzli1;
    unitex u_Grizzli2;

    state Initialize;
    state ShowBriefing;
    state ShowVideo;
    state Working;
    state MissionFailed;
    state MissionAccomplished;
    state Nothing;

    //-----
    state Initialize
    {

        //----- Goals -----
        RegisterGoal(EscortVehicle, "translateGoal412EscortVehicle");
        //---Show goals on list---
        EnableGoal(EscortVehicle, true);

        //----- Players -----
        p_Player = GetPlayer(1); //UCS
        p_Enemy = GetPlayer(3); //LC

        p_TmpPlayer = GetPlayer(2);
        p_TmpPlayer.EnableStatistics(false);

        p_Neutral = GetPlayer(4);
    }
}

```

```

p_Neutral.EnableStatistics(false);

//----- AI -----
p_Enemy.LoadScript("single\\singleMedium");

//wylacz inteligencje graczy
p_Enemy.EnableAIFeatures(aiControlOffense,false);
p_Enemy.EnableAIFeatures(aiControlDefense,false);

p_Enemy.SetNeutral(p_Neutral);
p_Player.SetNeutral(p_Neutral);

//----- Money -----
p_Player.SetMoney(0);
p_Neutral.SetMoney(0);
p_Enemy.SetMoney(5000);

p_Enemy.EnableResearch("RES_LC_WCH2",true);
p_Enemy.EnableResearch("RES_LC_WSR2",true);
p_Enemy.EnableResearch("RES_MSR2",true);
p_Enemy.EnableResearch("RES_MCH3",true);
//----- Variables -----

DestinationX=GetPointX(10);
DestinationY=GetPointY(10);

bCheckEndMission=false;

//---units-----
u_EscortedVehicle = GetUnit(GetPointX(0),GetPointY(0),0);
p_Player.AddUnitToSpecialTab(u_EscortedVehicle,true, -1);
u_Grizzly1 = GetUnit(GetPointX(1),GetPointY(1),GetPointZ(1));
u_Grizzly2 = GetUnit(GetPointX(2),GetPointY(2),GetPointZ(2));
u_Grizzly1.SetUnitName("translateGrizzly1");
u_Grizzly2.SetUnitName("translateGrizzly2");
p_Player.AddUnitToSpecialTab(u_Grizzly1,true, -1);
p_Player.AddUnitToSpecialTab(u_Grizzly2,true, -1);

p_Player.EnableBuilding("UCSBWB", false); //Szczoznia
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu rudy
p_Player.EnableBuilding("UCSCSD", false); //Laser antykatietowy

bStartRain1=false;
bStartRain2=false;
bStartRain3=false;
bStartRain4=false;
//----- Camera -----
CallCamera();

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);
return ShowVideo();
}

//----- state ShowVideo
{
ShowVideo("Cutscene19");
return ShowBriefing,100;
}
//----- state ShowBriefing
{
AddBriefing("translateBriefing412", p_Player.GetName());
MeteorRain(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(),20,100,30000,100,2,1);
return Working, 100;
}

//----- state Working
{
if(!bStartRain1 && p_Player.IsPointLocated(GetPointX(3),GetPointY(3),0))
{
bStartRain1=true;
MeteorRain(GetPointX(3),GetPointY(3),10,400,10000,100,3,1);
}
if(!bStartRain2 && p_Player.IsPointLocated(GetPointX(4),GetPointY(4),0))
{
bStartRain2=true;
MeteorRain(GetPointX(4),GetPointY(4),10,400,10000,100,3,1);
}

if(!bStartRain3 && p_Player.IsPointLocated(GetPointX(5),GetPointY(5),0))
{
bStartRain3=true;
MeteorRain(GetPointX(5),GetPointY(5),10,400,10000,100,3,1);
}
if(!bStartRain4 && p_Player.IsPointLocated(GetPointX(6),GetPointY(6),0))
{
bStartRain4=true;
MeteorRain(GetPointX(6),GetPointY(6),10,400,10000,100,3,1);
}

if(bCheckEndMission)
{
bCheckEndMission=false;

if(!u_EscortedVehicle.IsLive())
{
SetGoalState(EscortVehicle, goalFailed);
AddBriefing("translateFailed412", p_Player.GetName());
return MissionFailed, 20;
}
if(!u_Grizzly1.IsLive())
{
AddBriefing("translateGrizzly1Killed", p_Player.GetName());
return MissionFailed;
}
if(!u_Grizzly2.IsLive())
{
AddBriefing("translateGrizzly2Killed", p_Player.GetName());
return MissionFailed;
}
}

if(u_EscortedVehicle.DistanceTo(DestinationX, DestinationY) < 5)
{
SetGoalState(EscortVehicle, goalAchieved);
AddBriefing("translateAccomplished412", p_Player.GetName());
return MissionAccomplished, 20;
}

return Working, 100;
}

//----- state MissionFailed
{
EnableNextMission(0,2);
return Nothing;
}

//----- state MissionAccomplished
{
EnableNextMission(0, true);
EndMission(true);
return Nothing;
}

//----- state Nothing
{
return Nothing, 500;
}

//----- event UnitDestroyed(unit u_Unit)
{
bCheckEndMission=true;
}

}

mission413.ec
mission "translateMission413"
{
consts
{
scriptFieldGoal1=0;
scriptFieldGoal2=1;
scriptFieldGoal3=2;
scriptFieldGoal4=3;
scriptFieldGoal5=4;
scriptFieldGoal6=5;
scriptfieldMoney=9;
scriptFieldMeteors=10;
}

```

```

DestroyALPHA = 0;
}

player p_Enemy1;
player p_Enemy2;
player p_Enemy3;
player p_Player;
player p_Neutral;

int DestroyALPHA1Achieved;
int DestroyALPHA2Achieved;

int MinStateOfBuildings1;
int CurrentStateOfBuildings1;

int MinStateOfBuildings2;
int CurrentStateOfBuildings2;
int ShowBriefingB;
int i;

int bStartRain1;
int bStartRain2;
int bStartRain3;
int bStartRain4;

int nTimer0Counter;
int nTimer1Counter;
int nOffensePlayerNumber;
player p_OffensePlayer;

state Initialize;
state ShowVideo;
state ShowBriefing;
state Working;
state EndMissionFailed;
state Nothing;
//-----
state Initialize
{
    DestroyALPHA1Achieved = false;
    DestroyALPHA2Achieved = false;

    //----- Timers -----
    SetTimer(0, 1200);
    SetTimer(1, 20);
    SetTimer(2, 20);

    //----- Variables -----
    //----- Goals -----
    RegisterGoal(DestroyALPHA, "translateGoalDestroyBaseALPHA");
    EnableGoal(DestroyALPHA,true);

    //----- Players -----
    p_Player = GetPlayer(1); //ja czyl UCS
    p_Enemy1 = GetPlayer(3); //LC
    p_Enemy2 = GetPlayer(4); //LC
    p_Enemy3 = GetPlayer(6); //LC
    p_Neutral = GetPlayer(5); //UCS

    p_Neutral.EnableStatistics(false);
    p_Enemy3.EnableStatistics(false);
    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\\singleEasy");
        p_Enemy2.LoadScript("single\\singleEasy");
    }

    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\\singleMedium");
        p_Enemy2.LoadScript("single\\singleMedium");
    }

    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\\singleHard");
        p_Enemy2.LoadScript("single\\singleHard");
    }

    p_Enemy1.EnableAIFeatures(aiRush,false);
    p_Enemy2.EnableAIFeatures(aiRush,false);
}

//----AI off-----
p_Enemy3.EnableAIFeatures(aiEnabled,false);
p_Enemy3.SetAlly(p_Enemy2);
p_Enemy3.SetAlly(p_Enemy1);

p_Neutral.EnableAIFeatures(aiEnabled,false);
p_Neutral.SetAlly(p_Enemy2);
p_Neutral.SetNeutral(p_Player);
p_Player.SetNeutral(p_Neutral);
//----- Money -----
p_Player.EnableCommand(commandSoldBuilding,true);
p_Player.SetMoney(10000);
p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(10000);
p_Player.SetMilitaryUnitsLimit(30000);

p_Player.AddResearch("RES_MISSION_PACK1_ONLY");
p_Player.EnableResearch("RES_UCS_USL2",true);
p_Player.EnableResearch("RES_UCS_GARG1",true);
p_Player.EnableResearch("RES_UCS_WCH2",true);
p_Player.EnableResearch("RES_UCS_WSP1",true);
p_Player.EnableResearch("RES_UCS_WSR1",true);
p_Player.EnableResearch("RES_UCS_WSG2",true);
p_Player.EnableResearch("RES_MCH2",true);
p_Player.EnableResearch("RES_MSR2",true);
p_Player.EnableResearch("RES_UCS_MG2",true);
p_Player.EnableResearch("RES_UCS_Rephand",true);
p_Player.EnableResearch("RES_UCS_Sgen",true);
p_Player.AddResearch("RES_UCSUCS"); //Cargo Salamander

p_Enemy1.AddResearch("RES_MISSION_PACK1_ONLY");
p_Enemy1.EnableResearch("RES_LC_UME1",true);
p_Enemy1.EnableResearch("RES_LC_ULU2",true);
p_Enemy1.EnableResearch("RES_LC_UMO2",true);
p_Enemy1.EnableResearch("RES_LC_ACH2",true);
p_Enemy1.EnableResearch("RES_LC_ASR1",true);
p_Enemy1.EnableResearch("RES_LC_WSL1",true);
p_Enemy1.EnableResearch("RES_MSR2",true);

p_Enemy2.CopyResearches(p_Enemy1);
//----- Easy -----
if(GetDifficultyLevel()==0)
{
    //kill needless buildings
    KillArea(p_Enemy1.GetFF(), GetPointX(0), GetPointY(0), 0, 2);
    KillArea(p_Enemy1.GetFF(), GetPointX(1), GetPointY(1), 0, 2);

    KillArea(p_Enemy2.GetFF(), GetPointX(2), GetPointY(2), 0, 2);
    KillArea(p_Enemy2.GetFF(), GetPointX(3), GetPointY(3), 0, 2);
}

//----- Normal -----
if(GetDifficultyLevel()==1)
{
    //kill needless buildings
    KillArea(p_Enemy1.GetFF(), GetPointX(0), GetPointY(0), 0, 2);
    KillArea(p_Enemy2.GetFF(), GetPointX(2), GetPointY(2), 0, 2);

    for(i=4; i<=8; i+=1)
    {
        p_Enemy3.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCUMO3","LWSL2",null,null,null);
        p_Enemy3.CreateUnitEx(GetPointX(i)+1, GetPointY(i),
0,null,"LCUMO3","LWSR2",null,null,null);
    }
}

//----- Hard -----
if(GetDifficultyLevel()==2)
{
    for(i=4; i<=13; i+=1)
    {
        p_Enemy3.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCUCU3","LWMR3","LWHL2",null,null,1);
        p_Enemy3.CreateUnitEx(GetPointX(i)+1, GetPointY(i),
0,null,"LCUCU3","LCWMR3","LCWHL2",null,null,1);
    }
}

p_Enemy1.SetAlly(p_Enemy2);

p_Enemy1.EnableBuilding("LCCSD", false); //Laser antyrakietowy
p_Enemy2.EnableBuilding("LCCSD", false); //Laser antyrakietowy

```

```

p_Player.EnableBuilding("UCSBWB", false); //Stocznia
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu rudy
p_Player.EnableBuilding("UCSCSD", false); //Laser antyrakietowy

MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
bStartRain1=false;
bStartRain2=false;
bStartRain3=false;
bStartRain4=false;
ShowBriefingB=true;
//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 8, 0, 45, 0);
return ShowVideo;
}

//-----
state ShowVideo
{
    ShowVideo("Cutscene4");
    return ShowBriefing,100;
}
//-----
state ShowBriefing
{
    //check campaign goal 0
    p_Player.SetScriptData(scriptFieldGoal1, 2);

    //show campaign goal 1
    p_Player.SetScriptData(scriptFieldGoal2, 1);
    p_Player.SetScriptData(scriptFieldGoal3, 1);

    //show mission briefing
    AddBriefing("translateBriefing413a", p_Player.GetName()); //D+
    ShowArea(2,GetPointX(36),GetPointY(36),0,4,showAreaBuildings); //return Working;
}

//-----
state Working
{
    if(!bStartRain1 && p_Player.IsPointLocated(GetPointX(4),GetPointY(4),0))
    {
        bStartRain1=true;
        MeteorRain(GetPointX(4),GetPointY(4),10,400,10000,100,3,1);
    }
    if(!bStartRain2 && p_Player.IsPointLocated(GetPointX(5),GetPointY(5),0))
    {
        bStartRain2=true;
        MeteorRain(GetPointX(5),GetPointY(5),10,400,10000,100,3,1);
    }
    if(!bStartRain3 && p_Player.IsPointLocated(GetPointX(8),GetPointY(8),0))
    {
        bStartRain3=true;
        MeteorRain(GetPointX(8),GetPointY(8),10,400,10000,100,3,1);
    }
    if(!bStartRain4 && p_Player.IsPointLocated(GetPointX(13),GetPointY(13),0))
    {
        bStartRain4=true;
        MeteorRain(GetPointX(13),GetPointY(13),10,400,10000,100,3,1);
    }
    if(p_Player.GetNumberOfUnits())
    {
        AddBriefing("translateFailed413", p_Player.GetName());
        return EndMissionFailed;
    }
}

//----- check goals -----
//---capture LC base---

if(!DestroyALPHA1Achieved)
{
    CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();
    if(CurrentStateOfBuildings1 <= MinStateOfBuildings1)
    {
        p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy1.SetMoney(0);

        if(CurrentStateOfBuildings1)
        {
            DestroyALPHA1Achieved = true;
            MeteorRain(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(),20,500,100,500,2,1);
        }
    }
}

if(!DestroyALPHA2Achieved)
{
    CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
    if(CurrentStateOfBuildings2 <= MinStateOfBuildings2)
    {
        p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
        p_Enemy2.SetMoney(0);

        if(CurrentStateOfBuildings2)
        {
            DestroyALPHA2Achieved = true;
            MeteorRain(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(),20,500,100,500,2,1);
        }
    }
}

//--- misja wykonana calkowicie ---
if(DestroyALPHA1Achieved && DestroyALPHA2Achieved)
{
    //check campaign goal 1
    p_Player.SetScriptData(scriptFieldGoal1, 1);

    //check mission goal
    SetGoalState(DestroyALPHA,goalAchieved);

    EnableNextMission(0,true);
    EnableNextMission(1,true);
    AddBriefing("translateAccomplished413", p_Player.GetName());
    EnableEndMissionButton(true);
    return Nothing;
}

return Working;
}

state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}
//-----

//----- event Timer0() //wolany co minute
{
}

//-----
event Timer2()
{
    if>ShowBriefingB && p_Player.IsPointLocated(GetPointX(34),GetPointY(34),0)
    {
        ShowBriefingB=false;
        CallCamera();
        p_Player.LookAt(GetPointX(34),GetPointY(34), 15, 0, 20, 0);
        AddBriefing("translateBriefing413b", p_Player.GetName());
    }
    if>ShowBriefingB && p_Player.IsPointLocated(GetPointX(35),GetPointY(35),0)
    {
        ShowBriefingB=false;
        CallCamera();
        p_Player.LookAt(GetPointX(35),GetPointY(35), 15, 0, 20, 0);
        AddBriefing("translateBriefing413b", p_Player.GetName());
    }
    DamageArea(p_Player.GetIFF(),GetPointX(34),GetPointY(34),0,9,5);
    DamageArea(p_Player.GetIFF(),GetPointX(35),GetPointY(35),0,9,5);
}

event EndMission()
{
}

```

```

        if(DestroyALPHA1Achieved && DestroyALPHA2Achieved)
        {
            p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+>p_Player.GetMoney());
            p_Player.SetMoney(0);
        }
    }

mission414.ec
mission "translateMission414"
{
    consts
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;

        DestroyEnemy = 0;
    }

    player p_TmpPlayer;
    player p_Player;
    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;

    int nMoney;
    int bCheckEndMission;

    state Initialize;
    state ShowBriefing;
    state Working;
    state MissionFailed;
    state Nothing;

//-----
state Initialize
{
//----- Goals -----
    RegisterGoal(DestroyEnemy, "translateGoal414DestroyEnemy");

    //---Show goals on list---
    EnableGoal(DestroyEnemy, true);

//----- Players -----
    p_Player = GetPlayer(1); //UCS
    p_Enemy1 = GetPlayer(3); //LC
    p_Enemy2 = GetPlayer(4); //LC
    p_Enemy3 = GetPlayer(5); //LC

    p_TmpPlayer = GetPlayer(2);
    p_TmpPlayer.EnableStatistics(false);

//----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\\singleEasy");
        p_Enemy2.LoadScript("single\\singleEasy");
        p_Enemy3.LoadScript("single\\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\\singleMedium");
        p_Enemy2.LoadScript("single\\singleMedium");
        p_Enemy3.LoadScript("single\\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\\singleHard");
        p_Enemy2.LoadScript("single\\singleHard");
        p_Enemy3.LoadScript("single\\singleHard");
    }
    p_Enemy1.AddResearch("RES_LCUME1");
    p_Enemy1.AddResearch("RES_LCUME2");
    p_Enemy1.AddResearch("RES_LCUME3");
    p_Enemy1.AddResearch("RES_LCUSF1");
}

    p_Enemy1.AddResearch("RES_LCUSF2");
    p_Enemy1.AddResearch("RES_LC_ASRI");
    p_Enemy1.AddResearch("RES_LC_ASZR");

    p_Enemy1.SetNumberOfOffensiveTankPlatoons(1);
    p_Enemy1.SetNumberOfOffensiveShipPlatoons(0);
    p_Enemy1.SetNumberOfOffensiveHelicopterPlatoons(3);
    p_Enemy1.SetNumberOfDefensiveTankPlatoons(1);
    p_Enemy1.SetNumberOfDefensiveShipPlatoons(0);
    p_Enemy1.SetNumberOfDefensiveHelicopterPlatoons(3);

    p_Enemy2.SetNumberOfOffensiveTankPlatoons(1);
    p_Enemy2.SetNumberOfOffensiveShipPlatoons(0);
    p_Enemy2.SetNumberOfOffensiveHelicopterPlatoons(3);
    p_Enemy2.SetNumberOfDefensiveTankPlatoons(1);
    p_Enemy2.SetNumberOfDefensiveShipPlatoons(0);
    p_Enemy2.SetNumberOfDefensiveHelicopterPlatoons(3);

    p_Enemy3.SetNumberOfOffensiveTankPlatoons(1);
    p_Enemy3.SetNumberOfOffensiveShipPlatoons(0);
    p_Enemy3.SetNumberOfOffensiveHelicopterPlatoons(3);
    p_Enemy3.SetNumberOfDefensiveTankPlatoons(1);
    p_Enemy3.SetNumberOfDefensiveShipPlatoons(0);
    p_Enemy3.SetNumberOfDefensiveHelicopterPlatoons(3);

    } //wylacz inteligencje gracyj
    p_Enemy1.EnableAIFeatures(aiUpgradeCannons,true);
    p_Enemy2.EnableAIFeatures(aiUpgradeCannons,true);
    p_Enemy3.EnableAIFeatures(aiUpgradeCannons,true);

    p_Enemy1.EnableAIFeatures(aiRush,false);
    p_Enemy2.EnableAIFeatures(aiRush,false);
    p_Enemy3.EnableAIFeatures(aiRush,false);

    p_Enemy1.SetAlly(p_Enemy2);
    p_Enemy1.SetAlly(p_Enemy3);
    p_Enemy2.SetAlly(p_Enemy3);

    p_Enemy.AddResearch("RES_UCS_ART");
    p_Player.EnableResearch("RES_UCS_UML1",true);
    p_Player.EnableResearch("RES_MCH3",true);
    p_Player.EnableResearch("RES_MSR3",true);
    p_Player.EnableResearch("RES_UCS_MG3",true);
    p_Player.EnableResearch("RES_UCS_MGen",true);

    p_Enemy1.EnableResearch("RES_LC_WSL1",true);
    p_Enemy1.EnableResearch("RES_LC_WSR3",true);
    p_Enemy1.EnableResearch("RES_LC_MGen",true);

    p_Enemy2.CopyResearches(p_Enemy1);
    p_Enemy3.CopyResearches(p_Enemy1);

//----- Money -----
    p_Player.EnableCommand(commandSoldBuilding,true);
    if(GetDifficultyLevel()==0)
        p_Player.SetMoney(50000);
    if(GetDifficultyLevel()==1)
        p_Player.SetMoney(30000);
    if(GetDifficultyLevel()==2)
        p_Player.SetMoney(20000);

    p_Enemy1.SetMoney(3000);
    p_Enemy2.SetMoney(5000);
    p_Enemy3.SetMoney(10000);

//----- Timers -----
//----- Variables -----
    bCheckEndMission=false;

//--- Creating additional units ---
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.CreateUnitEx(GetPointX(0),GetPointY(0),0,null,"LCUCU3",
"LCWMR3","LCWNR3",null,null); // Cruiser m3 + hR
        p_Enemy2.CreateUnitEx(GetPointX(2),GetPointY(2),0,null,"LCUCU3",
"LCWMR3","LCWNR3",null,null); // Cruiser m3 + hR
        p_Enemy3.CreateUnitEx(GetPointX(3),GetPointY(3),0,null,"LCUCU3",
"LCWMR3","LCWNR3",null,null); // Cruiser m3 + hR
        p_Enemy3.CreateUnitEx(GetPointX(6),GetPointY(6),0,null,"LCUCU3",
"LCWMR3","LCWNR3",null,null); // Cruiser m3 + hR
    }
}

```

```

        p_Enemy3.CreateUnitEx(GetPointX(7), GetPointY(7), 0, null, "LCUCU3",
    "LCWMP3", "LCNMR3", null, null); // Crusher m3 + HR
    }
    if(GetDifficultyLevel() == 2)
    {
        p_Enemy1.CreateUnitEx(GetPointX(0), GetPointY(0), 0, null, "LCUCU3",
    "LCWMP3", "LCNMR3", null, null); // Crusher m3 + HR
        p_Enemy1.CreateUnitEx(GetPointX(1), GetPointY(1), 0, null, "LCUCR3",
    "LCWMP3", null, null, null); // Crater m3 + R
        p_Enemy2.CreateUnitEx(GetPointX(2), GetPointY(2), 0, null, "LCUCU3",
    "LCWMP3", "LCNMR3", null, null); // Crusher m3 + HR
        p_Enemy2.CreateUnitEx(GetPointX(3), GetPointY(3), 0, null, "LCUCU3",
    "LCWMP3", "LCNMR3", null, null); // Crusher m3 + HR
        p_Enemy2.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "LCUCR3",
    "LCWMP3", null, null, null); // Crater m3 + R
        p_Enemy2.CreateUnitEx(GetPointX(5), GetPointY(5), 0, null, "LCUCR3",
    "LCWMP3", null, null, null); // Crater m3 + R
        p_Enemy3.CreateUnitEx(GetPointX(6), GetPointY(6), 0, null, "LCUCU3",
    "LCWMP3", "LCNMR3", null, null); // Crusher m3 + HR
        p_Enemy3.CreateUnitEx(GetPointX(7), GetPointY(7), 0, null, "LCUCU3",
    "LCWMP3", "LCNMR3", null, null); // Crusher m3 + HR
        p_Enemy3.CreateUnitEx(GetPointX(9), GetPointY(9), 0, null, "LCUCR3",
    "LCWMP3", null, null, null); // Crater m3 + R
        p_Enemy3.CreateUnitEx(GetPointX(10), GetPointY(10), 0, null, "LCUCR3",
    "LCWMP3", null, null, null); // Crater m3 + R
    }

    p_Player.EnableBuilding("UCSBW", false); // Stocznia
    p_Player.EnableBuilding("UCSBT", false); // Centrum transportu rudy
    p_Player.EnableBuilding("UCSBD", false); // Laser antykatowy
    p_Player.EnableBuilding("UCSBL", false); // Landing Zone

    //----- Camera -----
    CallCamera();
}

p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 6.0, 20,
0);
return ShowBriefing, 100;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing414", p_Player.GetName());
    return Working, 100;
}

//-----
state Working
{
    if(bCheckEndMission)
    {
        bCheckEndMission = false;
        if(p_Player.GetNumberOfUnits() &&
p_Player.GetNumberOfBuildings())
        {
            SetGoalState(DestroyEnemy, goalFailed);
            AddBriefing("translateFailed414", p_Player.GetName());
            return MissionFailed, 20;
        }

        if(p_Enemy1.GetNumberOfBuildings() &&
p_Enemy2.GetNumberOfBuildings() && p_Enemy3.GetNumberOfBuildings())
        {
            SetGoalState(DestroyEnemy, goalAchieved);
        }
    }

    if(GetGoalState(DestroyEnemy) == goalAchieved)
    {
        AddBriefing("translateAccomplished414", p_Player.GetName());
        p_Player.EnableBuilding("UCSBL", true); // Landing Zone
        EnableEndMissionButton(true);
        return Nothing;
    }
}

return Working, 100;
}

//-----
state MissionFailed
{
    EnableNextMission(0, 2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission = true;
}

//-----
event EndMission()
{
    if(GetGoalState(DestroyEnemy) == goalAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney, p_Player.GetScriptData(scriptFieldMoney)
+p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}
}

mission415.ec
//mission starts from meteor rain in base(completely destroy)
//then first goal - destroy weather control buildings if not meteor rain in main
base.

mission "translateMission415"
{
    consts
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;

        DestroyBETA = 0;
        goalDestroyWC = 1;

        FirstOffenseAfter = 30; // minut
        OffenseFrequency = 10; // minut
        OffenseTime = 30; // sekundy
        FirstOffensePlayer = 3;
        LastOffensePlayer = 5;
    }

    int nTimer0Counter;
    int nTimer1Counter;
    int nMeteorShowerCounter;
    int nOffensePlayerNumber;
    int nFireMeteor;

    player p_OffensePlayer;

    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;
    player p_Player;
    player p_Neutral;

    int DestroyBETA1Achieved;
    int DestroyBETA2Achieved;
    int DestroyBETA3Achieved;

    int MinStateOfBuildings1;
    int MinStateOfBuildings2;
}

```

```

int MinStateOfBuildings3;
int currentStateOfBuildings1;
int currentStateOfBuildings2;
int currentStateOfBuildings3;

int i;

state Initialize;
state ShowBriefing;
state ShowBriefing2;
state Working;
state EndMissionFailed;
state Nothing;
//-----
state Initialize
{
    DestroyBETA1Achieved = false;
    DestroyBETA2Achieved = false;
    DestroyBETA3Achieved = false;
}

//----- Timers -----
SetTimer(0, 1200);
SetTimer(1, 20);
SetTimer(2, 20);

//----- Variables -----
nTimer0Counter = FirstOffenseAfter;
nTimer1Counter = 0;
nOffensePlayerNumber = FirstOffensePlayer - 1;
bFireMeteor = false;
//----- Goals -----
RegisterGoal(DestroyBETA, "translateGoalDestroyBaseBETA");
RegisterGoal(goalDestroyWC, "translateGoalDestroyWeatherControls");
//--show goals on list--
EnableGoal(DestroyBETA,true);

//----- Players -----
p_Player = GetPlayer(1); //ja czyl UCS
p_Enemy1 = GetPlayer(3); //LC
p_Enemy2 = GetPlayer(4); //LC
p_Enemy3 = GetPlayer(5); //LC

p_Neutral = GetPlayer(6);
p_Neutral.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableStatistics(false);
p_Player.SetNeutral(p_Neutral);
p_Enemy1.SetNeutral(p_Neutral);
p_Enemy2.SetNeutral(p_Neutral);
p_Enemy3.SetNeutral(p_Neutral);

//----- AI -----
if(GetDifficultyLevel() == 0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
    p_Enemy3.LoadScript("single\singleEasy");
}

if(GetDifficultyLevel() == 1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    p_Enemy3.LoadScript("single\singleMedium");
}

if(GetDifficultyLevel() == 2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
    p_Enemy3.LoadScript("single\singleHard");
}

//--- AI off -----
p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);
p_Enemy1.EnableAIFeatures(aiControlDefense,false);
p_Enemy2.EnableAIFeatures(aiControlDefense,false);
p_Enemy3.EnableAIFeatures(aiControlDefense,false);

p_Enemy1.EnableAIFeatures(aiRush,false);
p_Enemy2.EnableAIFeatures(aiRush,false);

p_Enemy3.EnableAIFeatures(aiRush,false);
p_Enemy1.EnableCommand(aiRush, false);
p_Enemy2.EnableCommand(aiRush, false);
p_Enemy3.EnableCommand(aiRush, false);

//----- Money -----
p_Player.EnableCommand(commandGoldBuilding,true);
p_Player.SetMoney(10000);
p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(10000);
p_Enemy3.SetMoney(10000);

p_Player.SetMilitaryUnitsLimit(40000);

p_Player.EnableResearch("RES_UCS_GARG1", true);
p_Player.EnableResearch("RES_UCS_UHL1", true);
p_Player.EnableResearch("RES_UCS_WHG1", true);
p_Player.EnableResearch("RES_UCS_WHG2", true);
p_Player.EnableResearch("RES_UCS_WSR1", true);
p_Player.EnableResearch("RES_UCS_WMR1", true);
p_Player.EnableResearch("RES_MMRC", true);
p_Player.EnableResearch("RES_UCS_RphHand2", true);
p_Player.EnableResearch("RES_UCS_HGcn", true);
p_Player.EnableResearch("RES_UCS_BMD", true);
p_Player.EnableResearch("RES_UCS_SHD", true);
p_Player.EnableResearch("RES_UCSUCS", true);
p_Player.EnableResearch("RES_UCSWEQ1", true);
p_Player.EnableResearch("RES_UCSWBHD1", true);

p_Player.AddResearch("RES_MISSION_PACK1_ONLY");

p_Enemy1.EnableResearch("RES_LC_UCR1",true);
p_Enemy1.EnableResearch("RES_LC_WHL1",true);
p_Enemy1.EnableResearch("RES_LC_WMR1",true);
p_Enemy1.EnableResearch("RES_MMRC2",true);
p_Enemy1.EnableResearch("RES_LC_BHD",true);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);
//----- Easy -----
if(GetDifficultyLevel() == 0)
{
    //kill needless buildings
    KillArea(p_Enemy1.GetIFF(), GetPointX(0), GetPointY(0), 0, 1);
    KillArea(p_Enemy1.GetIFF(), GetPointX(7), GetPointY(7), 0, 1);

    KillArea(p_Enemy2.GetIFF(), GetPointX(14), GetPointY(14), 0, 1);
    KillArea(p_Enemy2.GetIFF(), GetPointX(22), GetPointY(22), 0, 1);

    KillArea(p_Enemy3.GetIFF(), GetPointX(29), GetPointY(29), 0, 1);
    KillArea(p_Enemy3.GetIFF(), GetPointX(36), GetPointY(36), 0, 1);
}

//----- Normal -----
if(GetDifficultyLevel() == 1)
{
    ///////////////////LC1/////////////////
    //kill needless buildings
    KillArea(p_Enemy1.GetIFF(), GetPointX(7), GetPointY(7), 0, 1);

    //crusher : LCUCU1
    p_Enemy1.CreateUnitEx(GetPointX(1),
    GetPointY(1),0,null,"LCUCU3","LCWMR3",null,null);

    //lunar: LCULU1 + LCWCH1
    for(i=2; i<5; i=i+1)
    {
        p_Enemy1.CreateUnitEx(GetPointX(i), GetPointY(i),
        0,null,"LCULU1","LCWCH1",null,null);
    }

    //moon: LCUMO1 - under ground
    p_Enemy1.CreateUnitEx(GetPointX(5), GetPointY(5),
    GetPointZ(5),null,"LCUMO1","LCWCH1",null,null);
    p_Enemy1.CreateUnitEx(GetPointX(6), GetPointY(6),
    GetPointZ(6),null,"LCUMO1","LCWCH1",null,null);

    ///////////////////LC2/////////////////
    KillArea(p_Enemy2.GetIFF(), GetPointX(22), GetPointY(22), 0, 1);

    //crusher : LCUCU1
    p_Enemy2.CreateUnitEx(GetPointX(15),
    GetPointY(15),0,null,"LCUCU3","LCWMR3",null,null);
}

```

```

//lunar: LCULU1 + LCWCH1
for(i=16; i<19; i+=1)
{
    p_Enemy2.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCULU1","LCWCH1",null,null,null);
}

//moon : LCUMO1 - under ground
p_Enemy2.CreateUnitEx(GetPointX(19), GetPointY(19),
GetPointZ(19),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy2.CreateUnitEx(GetPointX(20), GetPointY(20),
GetPointZ(20),null,"LCUMO1","LCWCH1",null,null,null);

/////////////////LC3/////////////
KillArea(p_Enemy3.GetIFF()), GetPointX(36), GetPointY(36), 0, 1);

//thunder: LCUBO2 + LCWAMR1
p_Enemy3.CreateUnitEx(GetPointX(30),
GetPointY(30),0,null,"LCUBO2","LCWAMR1",null,null,null);

/////////////////LC1/////////////
//lunar: LCULU1 + LCWCH1
for(i=31; i<34; i+=1)
{
    p_Enemy3.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCULU1","LCWCH1",null,null,null);
}

//moon : LCUMO1 - under ground
p_Enemy3.CreateUnitEx(GetPointX(34), GetPointY(34),
GetPointZ(34),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy3.CreateUnitEx(GetPointX(35), GetPointY(35),
GetPointZ(35),null,"LCUMO1","LCWCH1",null,null,null);

}

//----Hard-----
if(GetDifficultyLevel() == 2)
{
 //////////////////LC1/////////////
 //crusher : LCUCU1
p_Enemy1.CreateUnitEx(GetPointX(1),
GetPointY(1),0,null,"LCUCU3","LCWMR3",null,null,null);
p_Enemy1.CreateUnitEx(GetPointX(8),
GetPointY(8),0,null,"LCUCU3","LCWMR3",null,null,null);

//lunar: LOULU1 + LCWCH1
for(i=2; i<5; i+=1)
{
    p_Enemy1.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCULU1","LCWCH1",null,null,null);
}

//moon : LCUMO1
for(i=9; i<12; i+=1)
{
    p_Enemy1.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCUMO1","LCWCH1",null,null,null);
}

//moon : LCUMO1 - under ground
p_Enemy1.CreateUnitEx(GetPointX(5), GetPointY(5),
GetPointZ(5),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy1.CreateUnitEx(GetPointX(6), GetPointY(6),
GetPointZ(6),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy1.CreateUnitEx(GetPointX(12), GetPointY(12),
GetPointZ(12),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy1.CreateUnitEx(GetPointX(13), GetPointY(13),
GetPointZ(13),null,"LCUMO1","LCWCH1",null,null,null);

/////////////////LC2/////////////
//crusher : LCUCU1
p_Enemy2.CreateUnitEx(GetPointX(15),
GetPointY(15),0,null,"LCUCU3","LCWMR3",null,null,null);
p_Enemy2.CreateUnitEx(GetPointX(23),
GetPointY(23),0,null,"LCUCU3","LCWMR3",null,null,null);

//lunar: LOULU1 + LCWCH1
for(i=16; i<19; i+=1)
{
    p_Enemy2.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCULU1","LCWCH1",null,null,null);
}

//moon : LCUMO1
for(i=24; i<27; i+=1)
{
    p_Enemy1.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCUMO1","LCWCH1",null,null,null);
}

//moon : LCUMO1 - under ground
p_Enemy2.CreateUnitEx(GetPointX(19), GetPointY(19),
GetPointZ(19),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy2.CreateUnitEx(GetPointX(20), GetPointY(20),
GetPointZ(20),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy2.CreateUnitEx(GetPointX(27), GetPointY(27),
GetPointZ(27),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy2.CreateUnitEx(GetPointX(28), GetPointY(28),
GetPointZ(28),null,"LCUMO1","LCWCH1",null,null,null);

/////////////////LC3/////////////
//thunder: LCUBO2 + LCWAMR1
p_Enemy3.CreateUnitEx(GetPointX(30),
GetPointY(30),0,null,"LCUBO2","LCWAMR1",null,null,null);
p_Enemy3.CreateUnitEx(GetPointX(37),
GetPointY(37),0,null,"LCUBO2","LCWAMR1",null,null,null);

//lunar: LCULU1 + LCWCH1
for(i=31; i<34; i+=1)
{
    p_Enemy3.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCULU1","LCWCH1",null,null,null);
}

//moon : LCUMO1
for(i=38; i<41; i+=1)
{
    p_Enemy1.CreateUnitEx(GetPointX(i), GetPointY(i),
0,null,"LCUMO1","LCWCH1",null,null,null);
}

//moon : LCUMO1 - under ground
p_Enemy3.CreateUnitEx(GetPointX(34), GetPointY(34),
GetPointZ(34),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy3.CreateUnitEx(GetPointX(35), GetPointY(35),
GetPointZ(35),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy3.CreateUnitEx(GetPointX(41), GetPointY(41),
GetPointZ(41),null,"LCUMO1","LCWCH1",null,null,null);
p_Enemy3.CreateUnitEx(GetPointX(42), GetPointY(42),
GetPointZ(42),null,"LCUMO1","LCWCH1",null,null,null);

}

p_Enemy2.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy1);

p_Enemy1.EnableBuilding("LCCSD", false); //Laser antykatietowy
p_Enemy2.EnableBuilding("LCCSD", false); //Laser antykatietowy
p_Enemy3.EnableBuilding("LCCSD", false); //Laser antykatietowy

p_Player.EnableBuilding("UCSBWB", false); //Stocznia
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu
p_Player.EnableBuilding("UCSCSD", false); //Laser antykatietowy

rudy
MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
MinStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings()/5;

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(), 15, 128, 20, 0);
p_Player.DelayedLookAt(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(), 6, 0, 20, 0.50, 1);
return ShowBriefing,55;
}

//-----
state ShowBriefing
{

```

```

//show campaign goal 3
p_Player.SetScriptData(scriptFieldGoal4, 1);
//show mission briefing
AddBriefing("translateBriefing415a", p_Player.GetName());
MeteorRain(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 15, 200, 1200, 200, 10, 10);
return ShowBriefing2,1200;
}
//-----
state ShowBriefing2
{
    //show mission briefing2
    EnableGoal(goalDestroyWC,true);
    AddBriefing("translateBriefing415b", p_Player.GetName());
    ShowArea(2,GetPointX(47),GetPointY(47),0,4,showAreaBuildings);
    ShowArea(2,GetPointX(48),GetPointY(48),0,4,showAreaBuildings);
    p_Player.LookAt(GetPointX(47),GetPointY(47), 15, 128, 20, 0);
    if(GetDifficultyLevel() == 0) nMeteorShowerCounter = 30*60;
    if(GetDifficultyLevel() == 1) nMeteorShowerCounter = 20*60;
    if(GetDifficultyLevel() == 2) nMeteorShowerCounter = 10*60;
    return Working;
}
//-----
state Working
{
    if(!p_Player.GetNumberOfUnits())
    {
        AddBriefing("translateFailed415", p_Player.GetName());
        return EndMissionFailed;
    }

    if(!p_Enemy1.GetNumberOfBuildings(buildingWeatherControl))
    {

        SetConsoleText("");
        if(nMeteorShowerCounter > 0)
        {
            SetGoalState(goalDestroyWC,goalAchieved);
            AddBriefing("translateBriefing415c", p_Player.GetName());
        }
        nMeteorShowerCounter = 0;
    }
    //----- check goals -----
    //---capture LC base---

    if(!DestroyBETA1Achieved)
    {
        CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();
        if(CurrentStateOfBuildings1 < MinStateOfBuildings1)
        {
            p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy1.SetMoney(0);

            if(!CurrentStateOfBuildings1)
                DestroyBETA1Achieved = true;
        }
    }

    if(!DestroyBETA2Achieved)
    {
        CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
        if(CurrentStateOfBuildings2 < MinStateOfBuildings2)
        {
            p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy2.SetMoney(0);

            if(!CurrentStateOfBuildings2)
                DestroyBETA2Achieved = true;
        }
    }

    if(!DestroyBETA3Achieved)
    {
        CurrentStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings();
        if(CurrentStateOfBuildings3 < MinStateOfBuildings3)
        {
            p_Enemy3.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy3.SetMoney(0);

            if(!CurrentStateOfBuildings3)
                DestroyBETA3Achieved = true;
        }
    }
}

//mission achieved
if(DestroyBETA1Achieved && DestroyBETA2Achieved && DestroyBETA3Achieved)
{
    //check campaign goal 1
    p_Player.SetScriptData(scriptFieldGoal4, 2);
    //check mission goal
    SetGoalState(DestroyBETA, goalAchieved);
    EnableNextMission(0,true);
    AddBriefing("translateAccomplished415", p_Player.GetName());
    EnableEndMissionButton(true);
    return Nothing;
}

return Working;
}

state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event Timer0() //wolany co minute
{
    unitex u_Grizzly1;
    u_Grizzly1 = p_Player.GetScriptUnit(1);
    if(bFireMeteor && nMeteorShowerCounter > 0 && u_Grizzly1 != null && u_Grizzly1.IsLive() && u_Grizzly1.IsInWorld(GetWorldNum()))
    {
        if(GetDifficultyLevel() == 1)
        Meteor(u_Grizzly1.GetLocationX(),u_Grizzly1.GetLocationY(),1);
        if(GetDifficultyLevel() == 2)
        Meteor(u_Grizzly1.GetLocationX(),u_Grizzly1.GetLocationY(),2);
    }

    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofensyw
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber == LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        p_OffensePlayer = GetPlayer(nOffensePlayerNumber);

        if(p_OffensePlayer.GetNumberOfUnits())
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);

            nTimer0Counter = OffenseFrequency;
            nTimer1Counter = OffenseTime;
        }
        if(!p_OffensePlayer.GetNumberOfUnits())
        {
            nTimer0Counter = 1;
        }
    }
}

//-----
//-----
event Timer1() //wolany co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofensyw
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, false);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, false);
        }
    }
    if(nMeteorShowerCounter > 0)

```

```

    {
        nMeteorShowerCounter = nMeteorShowerCounter - 1;
        SetConsoleText("translateMessage415",nMeteorShowerCounter);
        if(nMeteorShowerCounter == 0) // Meteor rain.
        {
            SetGoalState(goalDestroyWC,goalFailed);
            SetConsoleText("");
            p_Player.SetScriptData(scriptFieldMeteors,10);
            MeteorRain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),15,500,30000,100,10,10);
        }
    }
}

//-----
event Timer2()
{
    DamageArea(p_Player.GetIFF(),GetPointX(43),GetPointY(43),0,6,5);
    DamageArea(p_Player.GetIFF(),GetPointX(44),GetPointY(44),0,10,5);
    DamageArea(p_Player.GetIFF(),GetPointX(45),GetPointY(45),0,5,5);
    DamageArea(p_Player.GetIFF(),GetPointX(46),GetPointY(46),0,7,5);
}

event BuildingDestroyed(unit uBuilding)
{
    unitex u_Grizzly1;

    if(lbFireMeteor && nMeteorShowerCounter>0 )
    {
        u_Grizzly1 = p_Player.GetScriptUnit(1);
        if(uBuilding.GetAttacker() == u_Grizzly1)// gdy Grizzly1 rozwali pierwszy
budynek na misji to zaczynamy w niego walki.
        {
            bFireMeteor=true;
        }
    }
}

//-----
event EndMission()
{
    if(DestroyBETA1Achieved && DestroyBETA2Achieved &&
DestroyBETA3Achieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}
}

```

mission416.ec

mission "translateMission416"

```

{
    const
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;

        goalTakeOverEnemyPos = 0;
        goalDestroyEnemy = 1;

        FirstOffenseAfter = 25; // minut
        OffenseFrequency = 10; // minut
        OffenseTime = 30; // sekundy
    }

    player p_TmpPlayer;
    player p_Player;
    player p_Enemy;

    int bCheckEndMission;

    int nTimerOCounter;
    int nTimer1Counter;

    int nMinNoOfBuildings;
    int nCurrNoOfBuildings;
}

unitex Most1Wieza1;
unitex Most1Wieza2;
unitex Most1Wieza3;
unitex Most2Wieza1;
unitex Most2Wieza2;
unitex Most2Wieza3;
unitex Most3Wieza1;
unitex Most3Wieza2;
unitex Most3Wieza3;

state Initialize;
state ShowBriefing;
state Working;
state MissionFailed;
state Nothing;

//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(goalTakeOverEnemyPos,
    "translateGoal416TakeOverEnemyPos");
    RegisterGoal(goalDestroyEnemy, "translateGoal416DestroyEnemy");

    //---Show goals on list---
    EnableGoal(goalTakeOverEnemyPos, true);
    EnableGoal(goalDestroyEnemy, true);

    //----- Players -----
    p_Player = GetPlayer(1); //UCS
    p_Enemy = GetPlayer(3); //LC

    p_TmpPlayer = GetPlayer(2);
    p_TmpPlayer.EnableStatistics(false);

    //----- AI -----
    if(GetDifficultyLevel()==0)
        p_Enemy.LoadScript("single\singleEasy");
    if(GetDifficultyLevel()==1)
        p_Enemy.LoadScript("single\singleMedium");
    if(GetDifficultyLevel()==2)
        p_Enemy.LoadScript("single\singleHard");

    p_Player.SetMilitaryUnitsLimit(45000);

    //wyłącz inteligencję graczy
    p_Enemy.EnableAIFeatures(aiControlOffense,false);
    p_Enemy.EnableAIFeatures(aiControlDefense,false);

    p_Enemy.EnableResearch("RES_LC_UCU1",true);
    p_Enemy.EnableResearch("RES_LC_UBO1",true);
    p_Enemy.EnableResearch("RES_LC_AMR1",true);
    p_Enemy.EnableResearch("RES_LC_HGn",true);
    p_Enemy.AddResearch("RES_LC_WARTILLERY");

    //----- Money -----
    p_Player.EnableCommand(commandSoldBuilding,true);
    p_Player.SetMoney(10000);
    p_Enemy.SetMoney(10000);

    //----- Timers -----
    SetTimer(0, 1200);
    SetTimer(1, 20);

    //----- Variables -----
    bCheckEndMission=false;

    nTimerOCounter = FirstOffenseAfter;
    nTimer1Counter = 0;

    Most1Wieza1 = GetUnit(GetPointX(25),GetPointY(25),0);
    Most1Wieza2 = GetUnit(GetPointX(26),GetPointY(26),0);
    Most1Wieza3 = GetUnit(GetPointX(27),GetPointY(27),0);
    Most2Wieza1 = GetUnit(GetPointX(28),GetPointY(28),0);
    Most2Wieza2 = GetUnit(GetPointX(29),GetPointY(29),0);
    Most2Wieza3 = GetUnit(GetPointX(30),GetPointY(30),0);
    Most3Wieza1 = GetUnit(GetPointX(31),GetPointY(31),0);
    Most3Wieza2 = GetUnit(GetPointX(32),GetPointY(32),0);
    Most3Wieza3 = GetUnit(GetPointX(33),GetPointY(33),0);

    //--- Creating & destroying additional units -----
    if(GetDifficultyLevel()==0)
    {
        KillArea(p_Enemy.GetIFF(), GetPointX(0), GetPointY(0), 0, 1);
    }
}

```

```

    KillArea(p_Enemy.GetIFF(), GetPointX(3), GetPointY(3), 0, 1);

    if(GetDifficultyLevel()==1)
    {
        KillArea(p_Enemy.GetIFF(), GetPointX(3), GetPointY(3), 0, 1);

        p_Enemy.CreateUnitEx(GetPointX(1), GetPointY(1), 0, null, "LCUCU3",
        "LCWMR3", "LCWMR3", null, null,1); // Crusher m3 + hR
        p_Enemy.CreateUnitEx(GetPointX(2), GetPointY(2), 0, null, "LCUCU3",
        "LCWMR3", "LCWMR3", null, null,1); // Crusher m3 + hR

        p_Enemy.CreateUnitEx(GetPointX(10), GetPointY(10), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(11), GetPointY(11), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(12), GetPointY(12), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR

        p_Enemy.CreateUnitEx(GetPointX(6), GetPointY(6), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(8), GetPointY(8), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(14), GetPointY(14), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR

        p_Enemy.CreateUnitEx(GetPointX(7), GetPointY(7), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2); // Lunar m3 + 20mm
        p_Enemy.CreateUnitEx(GetPointX(9), GetPointY(9), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2); // Lunar m3 + 20mm
        p_Enemy.CreateUnitEx(GetPointX(15), GetPointY(15), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2); // Lunar m3 + 20mm
    }

    if(GetDifficultyLevel()==2)
    {
        p_Enemy.CreateUnitEx(GetPointX(1), GetPointY(1), 0, null, "LCUCU3",
        "LCWMR3", "LCWMR3", null, null,1); // Crusher m3 + hR
        p_Enemy.CreateUnitEx(GetPointX(2), GetPointY(2), 0, null, "LCUCU3",
        "LCWMR3", "LCWMR3", null, null,1); // Crusher m3 + hR
        p_Enemy.CreateUnitEx(GetPointX(4), GetPointY(4), 0, null, "LCUCR3",
        "LCWMR3", null, null, null,2); // Crater m3 + R
        p_Enemy.CreateUnitEx(GetPointX(5), GetPointY(5), 0, null, "LCUCR3",
        "LCWMR3", null, null, null,2); // Crater m3 + R

        p_Enemy.CreateUnitEx(GetPointX(10), GetPointY(10), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(11), GetPointY(11), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(12), GetPointY(12), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(22), GetPointY(22), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
        p_Enemy.CreateUnitEx(GetPointX(23), GetPointY(23), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
        p_Enemy.CreateUnitEx(GetPointX(24), GetPointY(24), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);

        p_Enemy.CreateUnitEx(GetPointX(6), GetPointY(6), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(8), GetPointY(8), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(14), GetPointY(14), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(16), GetPointY(16), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(17), GetPointY(17), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR
        p_Enemy.CreateUnitEx(GetPointX(18), GetPointY(18), 0, null, "LCUBO2",
        "LCWAMR2", null, null, null,1); // Thunder m2 + hR

        p_Enemy.CreateUnitEx(GetPointX(7), GetPointY(7), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
        p_Enemy.CreateUnitEx(GetPointX(9), GetPointY(9), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
        p_Enemy.CreateUnitEx(GetPointX(15), GetPointY(15), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
        p_Enemy.CreateUnitEx(GetPointX(20), GetPointY(20), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
        p_Enemy.CreateUnitEx(GetPointX(21), GetPointY(21), 0, null, "LCUCR3",
        "LCWHL2", null, null, null,2);
    }

    nMinNumberOfBuildings = p_Enemy.GetNumberOfBuildings() / 5;
}

p_Player.EnableBuilding("UCSBWB", false); // Stocznia
p_Player.EnableBuilding("UCSCTB", false); // Centrum transportu rudy
p_Player.EnableBuilding("UCSCSD", false); // Lase antykatietowy

p_Player.EnableResearch("RES_UCS_UBL1", true);
p_Player.EnableResearch("RES_UCS_UMI1", true);
p_Player.EnableResearch("RES_UCS_BOMBER21", true);
p_Player.EnableResearch("RES_UCS_WHP3", true);
p_Player.EnableResearch("RES_UCS_WMR2", true);
p_Player.EnableResearch("RES_UCS_WAMR1", true);
p_Player.EnableResearch("RES_MMR3", true);

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), 0.0, 20.0);
p_Player.GetStartingPointY(), 100;
return ShowBriefing;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing416", p_Player.GetName());
    return Working, 100;
}

//-----
state Working
{
    if(GetGoalState(goalTakeOverEnemyPos) == goalAchieved &&
    GetGoalState(goalDestroyEnemy) == goalAchieved)
    {
        AddBriefing("translateAccomplished416", p_Player.GetName());
        EnableNextMission(0, true);
        EnableEndMissionButton(true);
        return Nothing;
    }

    if(bCheckEndMission)
    {
        bCheckEndMission=false;
        if(lp_Player.GetNumberOfUnits() && lp_Player.GetNumberOfBuildings())
        {
            if(GetGoalState(goalTakeOverEnemyPos) != goalAchieved)
                SetGoalState(goalTakeOverEnemyPos, goalFailed);
            if(GetGoalState(goalDestroyEnemy) != goalAchieved)
                SetGoalState(goalDestroyEnemy, goalFailed);

            AddBriefing("translateFailed416", p_Player.GetName());
            return MissionFailed, 20;
        }
    }

    nCurrNumberOfBuildings = p_Enemy.GetNumberOfBuildings();

    if(nCurrNumberOfBuildings < nMinNumberOfBuildings)
    {
        p_Enemy.EnableAIFeatures(aiBuildBuildings, false);
        p_Enemy.SetMoney(0);

        if(!nCurrNumberOfBuildings)
            SetGoalState(goalTakeOverEnemyPos, goalAchieved);
        if(lp_Player.GetNumberOfUnits())
            SetGoalState(goalDestroyEnemy, goalAchieved);
    }

    if(lp_Player.GetNumberOfUnits())
        SetGoalState(goalDestroyEnemy, goalAchieved);
}

return Working, 100;
}

//-----
state MissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}

```

```

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit _Unit)
{
    bCheckEndMission=true;
}

//-----
event Timer0() //wolany co minute
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofensywy
    {
        p_Enemy.EnableAIFeatures(aiControlOffense, true);
        p_Enemy.EnableAIFeatures(aiControlDefense, true);

        nTimer0Counter = OffenseFrequency;
        nTimer1Counter = OffenseTime;
    }
}

//-----
event Timer1() //wolany co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofensywy
        {
            p_Enemy.EnableAIFeatures(aiControlOffense, false);
            p_Enemy.EnableAIFeatures(aiControlDefense, false);
        }
    }
}

//-----
event EndMission()
{
    if(GetGoalState(goalTakeOverEnemyPos) == goalAchieved &&
    GetGoalState(goalDestroyEnemy) == goalAchieved)
    {

        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
        +p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

```

mission417.ec

mission "translateMission417"

```

{
    consts
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;

        DestroyGAMMA = 0;
        CopyData = 1;

        FirstOffenseAfter = 3; // minuty
        OffenseFrequency = 10; // minuty
        OffenseTime = 30; // sekundy
        FirstOffensePlayer = 3;
        LastOffensePlayer = 6;
    }

    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;
    player p_Enemy4;
    player p_Neutral;

```

```

player p_Player;
int nTimer0Counter;
int nTimer1Counter;
int nOffensePlayerNumber;
player p_OffensePlayer;

int MinStateOfBuildings1;
int MinStateOfBuildings2;
int MinStateOfBuildings3;
int MinStateOfBuildings4;

int CurrentStateOfBuildings1;
int CurrentStateOfBuildings2;
int CurrentStateOfBuildings3;
int CurrentStateOfBuildings4;

int DestroyGAMMA1Achieved;
int DestroyGAMMA2Achieved;
int DestroyGAMMA3Achieved;
int DestroyGAMMA4Achieved;
int CopyDataAchieved;
int Briefing2Showed;

int xBaseLC1;
int yBaseLC1;

int xBaseLC2;
int yBaseLC2;

int xBaseLC3;
int yBaseLC3;

int xBaseLC4;
int yBaseLC4;

int i;

state Initialize;
state ShowBriefing;
state Working;
state EndMissionFailed;
state Nothing;
//-----
state Initialize
{
    DestroyGAMMA1Achieved = false;
    DestroyGAMMA2Achieved = false;
    DestroyGAMMA3Achieved = false;
    DestroyGAMMA4Achieved = false;
    CopyDataAchieved = false;
    Briefing2Showed = false;
}

//----- Timers -----
SetTimer(0, 1200);
SetTimer(1, 20);
SetTimer(2, 20);

//----- Variables -----
nTimer0Counter = FirstOffenseAfter;
nTimer1Counter = 0;
nOffensePlayerNumber = FirstOffensePlayer - 1;

//----- Goals -----
RegisterGoal(DestroyGAMMA, "translateGoalDestroyBaseGAMMA");
RegisterGoal(CopyData, "translateGoalCopyData");

//---show goals on list---
EnableGoal(DestroyGAMMA,true);
EnableGoal(CopyData,true);

//----- Players -----
p_Player = GetPlayer(1); //ja czyl UCS
p_Enemy1 = GetPlayer(3); //LC
p_Enemy2 = GetPlayer(4); //LC
p_Enemy3 = GetPlayer(5); //LC
p_Enemy4 = GetPlayer(6); //LC

p_Neutral = GetPlayer(7);
p_Neutral.EnableAIFeatures(aiEnabled,false);
p_Neutral.EnableStatistics(false);
p_Player.SetNeutral(p_Neutral);

```

```

p_Enemy1.SetNeutral(p_Neutral);
p_Enemy2.SetNeutral(p_Neutral);
p_Enemy3.SetNeutral(p_Neutral);
p_Enemy4.SetNeutral(p_Neutral);
//----- AI -----
if(GetDifficultyLevel()==0)
{
    p_Enemy1.LoadScript("single\singleEasy");
    p_Enemy2.LoadScript("single\singleEasy");
    p_Enemy3.LoadScript("single\singleEasy");
    p_Enemy4.LoadScript("single\singleEasy");
}

if(GetDifficultyLevel()==1)
{
    p_Enemy1.LoadScript("single\singleMedium");
    p_Enemy2.LoadScript("single\singleMedium");
    p_Enemy3.LoadScript("single\singleMedium");
    p_Enemy4.LoadScript("single\singleMedium");
}

if(GetDifficultyLevel()==2)
{
    p_Enemy1.LoadScript("single\singleHard");
    p_Enemy2.LoadScript("single\singleHard");
    p_Enemy3.LoadScript("single\singleHard");
    p_Enemy4.LoadScript("single\singleHard");
}

    //---AI off-----
p_Enemy1.EnableAIFeatures(aiControlOffense,false);
p_Enemy2.EnableAIFeatures(aiControlOffense,false);
p_Enemy3.EnableAIFeatures(aiControlOffense,false);
p_Enemy4.EnableAIFeatures(aiControlOffense,false);

p_Enemy1.EnableAIFeatures(aiControlDefense,false);
p_Enemy2.EnableAIFeatures(aiControlDefense,false);
p_Enemy3.EnableAIFeatures(aiControlDefense,false);
p_Enemy4.EnableAIFeatures(aiControlDefense,false);

p_Enemy1.EnableAIFeatures(aiRush,false);
p_Enemy2.EnableAIFeatures(aiRush,false);
p_Enemy3.EnableAIFeatures(aiRush,false);

//----- Money -----
p_Player.EnableCommand(commandSoldBuilding,true);
p_Player.SetMoney(15000);
p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(10000);
p_Enemy3.SetMoney(10000);
p_Enemy4.SetMoney(10000);

p_Player.SetMilitaryUnitsLimit(50000);

//---Variables---
xBaseLC1 = p_Enemy1.GetStartingPointX();
yBaseLC1 = p_Enemy1.GetStartingPointY();

xBaseLC2 = p_Enemy2.GetStartingPointX();
yBaseLC2 = p_Enemy2.GetStartingPointY();

xBaseLC3 = p_Enemy3.GetStartingPointX();
yBaseLC3 = p_Enemy3.GetStartingPointY();

xBaseLC4 = p_Enemy4.GetStartingPointX();
yBaseLC4 = p_Enemy4.GetStartingPointY();

//---Artefacts : computers to capture ---
CreateArtefact("NEACOMPUTER", GetPointX(0),GetPointY(0),GetPointZ(0),0,artefactSpecialAIOther);

p_Enemy2.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy1);
p_Enemy4.SetAlly(p_Enemy1);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);
p_Enemy4.CopyResearches(p_Enemy1);

p_Enemy1.EnableBuilding("LCCSD", false); //Laser antyrakietowy
p_Enemy2.EnableBuilding("LCCSD", false); //Laser antyrakietowy
p_Enemy3.EnableBuilding("LCCSD", false); //Laser antyrakietowy
p_Enemy4.EnableBuilding("LCCSD", false); //Laser antyrakietowy

p_Player.EnableBuilding("UCSBWB", false); //Stocznia
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu rudy
p_Player.EnableBuilding("UCCSD", false); //Laser antyrakietowy

p_Player.EnableResearch("RES_UCS_BHD", true);
p_Player.EnableResearch("RES_UCS_BOMBER22", true);
p_Player.EnableResearch("RES_UCS_SHD2", true);

MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
MinStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings()/5;
MinStateOfBuildings4 = p_Enemy4.GetNumberOfBuildings()/5;

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 6, 0, 20, 0);
return ShowBriefing;
}

//----- Show Briefing -----
state ShowBriefing
{
    //show campaign goal 2
    p_Player.SetScriptData(scriptFieldGoal5, 1);

    //show mission briefing
    AddBriefing("translateBriefing417a", p_Player.GetName());

    return Working;
}

//----- Working -----
state Working
{
    if(p_Player.GetNumberOfUnits())
    {
        AddBriefing("translateFailed417", p_Player.GetName());
        return EndMissionFailed;
    }

    //----- check goals -----
    //---capture LC base---

    if(!DestroyGAMMA1Achieved)
    {
        CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();
        if(CurrentStateOfBuildings1 < 3)
        {
            p_Enemy1.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy1.SetMoney(0);
            DestroyGAMMA1Achieved = true;
        }
    }

    if(!DestroyGAMMA2Achieved)
    {
        CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
        if(CurrentStateOfBuildings2 < 3)
        {
            p_Enemy2.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy2.SetMoney(0);
            DestroyGAMMA2Achieved = true;
        }
    }

    if(!DestroyGAMMA3Achieved)
    {
        CurrentStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings();
        if(CurrentStateOfBuildings3 < 3)
        {
            p_Enemy3.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy3.SetMoney(0);
            DestroyGAMMA3Achieved = true;
        }
    }

    if(!DestroyGAMMA4Achieved)
    {
        CurrentStateOfBuildings4 = p_Enemy4.GetNumberOfBuildings();
        if(CurrentStateOfBuildings4 < 3)
        {
            p_Enemy4.EnableAIFeatures(aiBuildBuildings,false);
            p_Enemy4.SetMoney(0);
            DestroyGAMMA4Achieved = true;
        }
    }
}

```

```

        }

    if(!Briefing2Showed)
    {
        if(p_Player.IsPointLocated(xBaseLC1, yBaseLC1) ||
           p_Player.IsPointLocated(xBaseLC2, yBaseLC2) ||
           p_Player.IsPointLocated(xBaseLC3, yBaseLC3) ||
           p_Player.IsPointLocated(xBaseLC4, yBaseLC4) ||
           p_Player.IsPointLocated(GetPointX(0), GetPointY(0), GetPointZ(0)))
        {
            Briefing2Showed = true;
            AddBriefing("translateBriefing417b", p_Player.GetName());
        }
    }

    if(DestroyGAMMA1Achieved && DestroyGAMMA2Achieved &&
DestroyGAMMA3Achieved && DestroyGAMMA4Achieved &&
GetGoalState(DestroyGAMMA) != goalAchieved)
    {
        //check mission goal
        SetGoalState(DestroyGAMMA, goalAchieved);
    }

    //mission achieved
    if(DestroyGAMMA1Achieved && DestroyGAMMA2Achieved &&
DestroyGAMMA3Achieved && DestroyGAMMA4Achieved && CopyDataAchieved)
    {
        //check campaign goal 4
        p_Player.SetScriptData(scriptFieldGoal5, 2);
        EnableNextMission(0,true);
        AddBriefing("translateAccomplished417", p_Player.GetName());
        EnableEndMissionButton(true);
        return Nothing;
    }

    return Working;
}

state EndMissionFailed
{
    EnableNextMission(0.2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}
//-----

//--capture CPU--
event Artefact(int alld,player piPlayer)
{
    if(piPlayer!=p_Player) return false;
    CopyDataAchieved = true;
    SetGoalState(CopyData, goalAchieved);
    return true;
}

//-----
event Timer0() //wolany co minute
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofenswy
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber == LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        p_OffensePlayer = GetPlayer(nOffensePlayerNumber);

        if(p_OffensePlayer.GetNumberOfUnits())
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);
        }
        nTimer0Counter = OffenseFrequency;
    }
}

nTimer1Counter = OffenseTime;
}
if(p_OffensePlayer.GetNumberOfUnits())
{
    nTimer0Counter = 1;
}
}

//-----
event Timer1() //wolany co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofenswy
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, false);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, false);
        }
    }
}

//-----
event Timer2()
{
    DamageArea(p_Player.GetIFF(),GetPointX(4),GetPointY(4),0.7,5);
    DamageArea(p_Player.GetIFF(),GetPointX(5),GetPointY(5),0.8,5);
    DamageArea(p_Player.GetIFF(),GetPointX(6),GetPointY(6),0.7,5);
    DamageArea(p_Player.GetIFF(),GetPointX(7),GetPointY(7),0.6,5);
}

//-----
event EndMission()
{
    if(DestroyGAMMA1Achieved && DestroyGAMMA2Achieved &&
DestroyGAMMA3Achieved && DestroyGAMMA4Achieved && CopyDataAchieved)
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+ p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}

mission418.ec
mission "translateMission418"
{
    consts
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;
        DestroyPowerPlants = 0;
    }

    player p_Player;
    player p_Enemy1;
    player p_Enemy2;
    player p_Neutral;

    int bCheckEndMission;
    int bReinforcementsOnTheWay;
    int nCallReinforcements;

    unitex Elektr1;
    unitex Elektr2;
    unitex Elektr3;
    unitex Elektr4;
    unitex Elektr5;
    unitex Elektr6;

    state Initialize;
    state ShowBriefing;
    state Working;
}

```

```

state MissionFailed;
state Nothing;

//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(DestroyPowerPlants, "translateGoal418DestroyPowerPlants");

    //---Show goals on list---
    EnableGoal(DestroyPowerPlants, true);

    //----- Players -----
    p_Player = GetPlayer(1); //UCS
    p_Enemy1 = GetPlayer(3); //LC
    p_Enemy2 = GetPlayer(5); //LC
    p_Enemy2.EnableStatistics(false);

    p_Enemy1.SetAlly(p_Enemy2);
    p_Enemy2.SetAlly(p_Enemy1);

    p_Neutral = GetPlayer(4);
    p_Neutral.EnableStatistics(false);

    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\\singleEasy");
        p_Enemy2.LoadScript("single\\singleEasy");
    }
    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\\singleMedium");
        p_Enemy2.LoadScript("single\\singleMedium");
    }
    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\\singleHard");
        p_Enemy2.LoadScript("single\\singleHard");
    }

    //wyłącz inteligencje graczy
    p_Enemy1.EnableAIFeatures(aiControlOffense, false);
    p_Enemy1.EnableAIFeatures(aiControlDefense, false);
    p_Enemy2.EnableAIFeatures(aiEnabled, false);
    p_Enemy2.EnableAIFeatures(aiControlOffense, false);
    p_Enemy2.EnableAIFeatures(aiControlDefense, false);

    p_Enemy1.SetNeutral(p_Neutral);
    p_Enemy2.SetNeutral(p_Neutral);
    p_Player.SetNeutral(p_Neutral);

    //----- Money -----
    p_Player.EnableCommand(commandSoldBuilding,true);
    p_Player.SetMoney(0);
    p_Neutral.SetMoney(0);
    p_Enemy1.SetMoney(0);
    p_Enemy2.SetMoney(0);

    //----- Variables -----
    bCheckEndMission = false;
    bReinforcementsOnTheWay = false;
    nCallReinforcements = 0;

    Elektr1 = GetUnit(GetPointX(1), GetPointY(1), 0);
    Elektr2 = GetUnit(GetPointX(2), GetPointY(2), 0);
    Elektr3 = GetUnit(GetPointX(3), GetPointY(3), 0);
    Elektr4 = GetUnit(GetPointX(4), GetPointY(4), 0);
    Elektr5 = GetUnit(GetPointX(5), GetPointY(5), 0);
    Elektr6 = GetUnit(GetPointX(6), GetPointY(6), 0);

    //---- Creating & destroying additional units ----
    if(GetDifficultyLevel()==0)
    {
        p_Player.CreateUnitEx(GetPointX(0) + 1, GetPointY(0) - 2, 0, null,
        "UCSUL1", "UCSWTSP1", null, null, null); // Tiger + P
        p_Player.CreateUnitEx(GetPointX(0) + 2, GetPointY(0) - 1, 0, null,
        "UCSUL1", "UCSWMR1", null, null, null); // Panther + R
        p_Player.CreateUnitEx(GetPointX(0) - 2, GetPointY(0) - 1, 0, null,
        "UCSUSL1", "UCSWTSR1", null, null, null); // Tiger + R
        p_Player.CreateUnitEx(GetPointX(0) + 2, GetPointY(0) - 1, 0, null,
        "UCSUSL1", "UCSWTSP1", null, null, null); // Tiger + P
        p_Player.CreateUnitEx(GetPointX(0) + 2, GetPointY(0), 0, null,
        "UCSUSL1", "UCSWTSP1", null, null, null); // Tiger + P
    }

    //----- Buildings -----
    p_Player.EnableBuilding("UCSBWB", false); //Stocznia
    p_Player.EnableBuilding("UCSMB", false); //Centrum transportu
    p_Player.EnableBuilding("UCSCSD", false); //Laser antykatietowy

    p_Player.EnableResearch("RES_UCS_BOMBER31", true);
    p_Player.EnableResearch("RES_UCS_WAPB1", true);
    p_Player.EnableResearch("RES_UCS_SHD3", true);

    p_Enemy2.CopyResearches(p_Enemy1);

    //----- Camera -----
    CallCamera();
    p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(), 6.0, 20.0);
    return ShowBriefing, 100;
}

//-----
state ShowBriefing
{
    AddBriefing("translateBriefing418", p_Player.GetName());
    return Working, 100;
}

//-----
state Working
{
    if(!bReinforcementsOnTheWay)
    {
        if(Elektr1.IsLive() || !Elektr2.IsLive() || !Elektr3.IsLive() ||
        Elektr4.IsLive() || !Elektr5.IsLive() || !Elektr6.IsLive())
        {
            nCallReinforcements = nCallReinforcements + 1;
        }

        if(nCallReinforcements > 60) // po 5 minutach
        {
            p_Enemy2.EnableAIFeatures(aiEnabled, true);
            p_Enemy2.EnableAIFeatures(aiControlOffense, true);
            p_Enemy2.EnableAIFeatures(aiControlDefense, true);
            p_Enemy2.RussianAttack(p_Enemy1.GetStartingPointX(),
            p_Enemy1.GetStartingPointY(), 0);
            bReinforcementsOnTheWay = true;
        }
    }

    if(bCheckEndMission)
    {
        bCheckEndMission = false;
        if(p_Player.GetNumberOfUnits() == 0)
        {
            SetGoalState(DestroyPowerPlants, goalFailed);
            AddBriefing("translateFailed418", p_Player.GetName());
            return MissionFailed, 20;
        }

        if(!Elektr1.IsLive() && !Elektr2.IsLive() && !Elektr3.IsLive() &&
        Elektr4.IsLive() && !Elektr5.IsLive() && !Elektr6.IsLive())
        {
            SetGoalState(DestroyPowerPlants, goalAchieved);
            AddBriefing("translateAccomplished418", p_Player.GetName());
            EnableNextMission(O, true);
            EnableEndMissionButton(true);
            return Nothing;
        }
    }
    return Working, 100;
}

```

```

//-----
state MissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

//-----
event EndMission()
{
    if(!Elektr1.IsLive() && !Elektr2.IsLive() && !Elektr3.IsLive() &&
Elektr4.IsLive()
        && !Elektr5.IsLive() && !Elektr6.IsLive())
    {
        p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}
}

mission419.ec
mission "translateMission419"
{
    consts
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=1;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;

        DestroyDELTA = 0;

        FirstOffenseAfter = 15; // minut
        OffenseFrequency = 3; // minut
        OffenseTime = 30; // sekundy
        FirstOffensePlayer = 3;
        LastOffensePlayer = 5;
    }

    int nTimer0Counter;
    int nTimer1Counter;
    int nOffensePlayerNumber;
    player p_OffensePlayer;

    player p_Enemy1;
    player p_Enemy2;
    player p_Enemy3;
    player p_Player;
    player p_PlayerLC;
    player p_Neutral;

    int DestroyDELTA1Achieved;
    int DestroyDELTA2Achieved;
    int DestroyDELTA3Achieved;

    int MinStateOfBuildings1;
    int MinStateOfBuildings2;
    int MinStateOfBuildings3;

    int CurrentStateOfBuildings1;
    int CurrentStateOfBuildings2;
    int CurrentStateOfBuildings3;

    int i;

    state Initialize;
    state ShowBriefing;
    state Working;
    state EndMissionFailed;
    state Nothing;
//-----
state Initialize
{
    DestroyDELTA1Achieved = false;
    DestroyDELTA2Achieved = false;
    DestroyDELTA3Achieved = false;

    //----- Timers -----
    SetTimer(0, 1200);
    SetTimer(1, 120);
    SetTimer(2, 20);

    //----- Variables -----
    nTimer0Counter = FirstOffenseAfter;
    nTimer1Counter = 0;
    nOffensePlayerNumber = FirstOffensePlayer - 1;

    //----- Goals -----
    RegisterGoal(DestroyDELTA, "translateGoalDestroyBaseDELTA");

    //--- show goals on list --
    EnableGoal(DestroyDELTA,true);

    //----- Players -----
    p_Player = GetPlayer(1); //ja czyl UCS
    p_Enemy1 = GetPlayer(3); //LC
    p_Enemy2 = GetPlayer(4); //LC
    p_Enemy3 = GetPlayer(5); //LC

    p_Neutral = GetPlayer(6);
    p_Neutral.EnableAIFeatures(aiEnabled,false);
    p_Neutral.EnableStatistics(false);
    p_Player.SetNeutral(p_Neutral);
    p_Enemy1.SetNeutral(p_Neutral);
    p_Enemy2.SetNeutral(p_Neutral);
    p_Enemy3.SetNeutral(p_Neutral);

    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy3.LoadScript("single\singleEasy");
    }

    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy3.LoadScript("single\singleMedium");
    }

    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy3.LoadScript("single\singleHard");
    }

    //--- AI off -----
    p_Enemy1.EnableAIFeatures(aiControlOffense,false);
    p_Enemy2.EnableAIFeatures(aiControlOffense,false);
    p_Enemy3.EnableAIFeatures(aiControlOffense,false);

    p_Enemy1.EnableAIFeatures(aiControlDefense,false);
    p_Enemy2.EnableAIFeatures(aiControlDefense,false);
    p_Enemy3.EnableAIFeatures(aiControlDefense,false);

    p_Enemy1.EnableAIFeatures(aiRush,false);
    p_Enemy2.EnableAIFeatures(aiRush,false);
    p_Enemy3.EnableAIFeatures(aiRush,false);

    //----- Money -----
    p_Player.EnableCommand(commandSellBuilding,true);
    p_Player.SetMoney(20000);
}

```

```

p_Enemy1.SetMoney(10000);
p_Enemy2.SetMoney(10000);
p_Enemy3.SetMoney(10000);

p_Player.SetMilitaryUnitsLimit(60000);

p_Enemy2.SetAlly(p_Enemy1);
p_Enemy3.SetAlly(p_Enemy1);

p_Enemy1.EnableBuilding("LCCSD", false); //Laser antyrafikietowy
p_Enemy2.EnableBuilding("LCCD", false); //Laser antyrafikietowy
p_Enemy3.EnableBuilding("LCCD", false); //Laser antyrafikietowy

p_Player.EnableBuilding("UCSBWB", false); //Stocznia
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu
p_Player.EnableBuilding("UCSCSD", false); //Laser antyrafikietowy

MinStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings()/5;
MinStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings()/5;
MinStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings()/5;

//----- Researches -----
p_Player.EnableResearch("RES_UCS_WAPB2", true);
p_Player.EnableResearch("RES_UCS_MB2", true);
p_Player.EnableResearch("RES_UCS_SHD4", true);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);

//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
6, 0, 20, 0);
return ShowBriefing;
}

//-----
state ShowBriefing
{
    //show campaign goal 2
    p_Player.SetScriptData(scriptFieldGoal6, 1);

    //show mission briefing
    AddBriefing("translateBriefing419", p_Player.GetName());

    return Working;
}

//-----
state Working
{
    if(lp_Player.GetNumberOfUnits())
    {
        AddBriefing("translateFailed419", p_Player.GetName());
        return EndMissionFailed;
    }

    //----- check goals -----
    //----capture LC base---

    if(!DestroyDELTAAchieved)
    {
        CurrentStateOfBuildings1 = p_Enemy1.GetNumberOfBuildings();

        if(CurrentStateOfBuildings1 < MinStateOfBuildings1)
        {
            p_Enemy1.EnableAIFeatures(aiBuildBuildings, false);
            p_Enemy1.SetMoney(0);

            if(CurrentStateOfBuildings1)
                DestroyDELTAAchieved = true;
        }
    }

    if(!DestroyDELTABAchieved)
    {
        CurrentStateOfBuildings2 = p_Enemy2.GetNumberOfBuildings();
        if(CurrentStateOfBuildings2 < MinStateOfBuildings2)
        {
            p_Enemy2.EnableAIFeatures(aiBuildBuildings, false);
            p_Enemy2.SetMoney(0);

            if(CurrentStateOfBuildings2)
                DestroyDELTABAchieved = true;
        }
    }

    if(!DestroyDELTACAchieved)
    {
        CurrentStateOfBuildings3 = p_Enemy3.GetNumberOfBuildings();
        if(CurrentStateOfBuildings3 < MinStateOfBuildings3)
        {
            p_Enemy3.EnableAIFeatures(aiBuildBuildings, false);
            p_Enemy3.SetMoney(0);

            if(CurrentStateOfBuildings3)
                DestroyDELTACAchieved = true;
        }
    }

    //mission achieved
    if(DestroyDELTAAchieved && DestroyDELTABAchieved &&
    DestroyDELTACAchieved)
    {
        //check campaign goal 4
        p_Player.SetScriptData(scriptFieldGoal6, 2);
        //check mission goal
        SetGoalState(DestroyDELTAA, goalAchieved);

        EnableNextMission(0,true);
        AddBriefing("translateAccomplished419", p_Player.GetName());
        EnableEndMissionButton(true);
        return Nothing;
    }

    return Working;
}

state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//-----
state Nothing
{
    return Nothing, 500;
}

//-----
event Timer0() //wolany co minute
{
    nTimer0Counter = nTimer0Counter - 1;

    if(nTimer0Counter == 0) // start ofensyw
    {
        nOffensePlayerNumber = nOffensePlayerNumber + 1;

        if(nOffensePlayerNumber == LastOffensePlayer + 1)
            nOffensePlayerNumber = FirstOffensePlayer;

        p_OffensePlayer = GetPlayer(nOffensePlayerNumber);

        if(p_OffensePlayer.GetNumberOfUnits())
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense, true);
            p_OffensePlayer.EnableAIFeatures(aiControlDefense, true);

            nTimer0Counter = OffenseFrequency;
            nTimer1Counter = OffenseTime;
        }

        if(lp_OffensePlayer.GetNumberOfUnits())
        {
            nTimer0Counter = 1;
        }
    }
}

```

```

//-----
event Timer1() //wolany co sekunde
{
    if(nTimer1Counter > 0)
    {
        nTimer1Counter = nTimer1Counter - 1;

        if(nTimer1Counter == 0) // stop ofenswy
        {
            p_OffensePlayer.EnableAIFeatures(aiControlOffense,false);
            p_OffensePlayer.EnableAIFeatures(aiControlOffense,false);
        }
    }
}
//-----
event Timer2()
{
    DamageArea(p_Player.GetIFF(),GetPointX(0),GetPointY(0),0.4,5);
    DamageArea(p_Player.GetIFF(),GetPointX(1),GetPointY(1),0.4,5);
    DamageArea(p_Player.GetIFF(),GetPointX(2),GetPointY(2),0.6,5);
}
//-----
event EndMission()
{
    if(DestroyDELTAAchieved && DestroyDELTABAchieved &&
DestroyDELTACAchieved)
    {

p_Player.SetScriptData(scriptFieldMoney,p_Player.GetScriptData(scriptFieldMoney)
+p_Player.GetMoney());
        p_Player.SetMoney(0);
    }
}
mission "translateMission420"
{
    consts
    {
        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;

        goalDeactivePP1 = 0;
        goalDeactivePP2 = 1;
        goalDeactivePP3 = 2;
        goalDeactivePP4 = 3;
        goalDestroyControlCenter1 = 4;
        goalDestroyControlCenter2 = 5;
        goalDestroyControlCenter3 = 6;
        goalDestroyControlCenter4 = 7;
    }

    player p_Enemy1://3
    player p_Enemy2://4
    player p_Enemy3://5
    player p_Enemy4://6 - tunelach
    player p_Neutral;//7
    player p_Player;

    int bCheckEndMission;

    unitex uC1;
    unitex uC2;
    unitex uC3;
    unitex uC4;
    unitex uP1;
    unitex uP2;
    unitex uP3;
    unitex uP4;
}

int bPP1active;
int bPP2active;
int bPP3active;
int bPP4active;

state Initialize;
state ShowBriefing;
state Working;
state EndMissionFailed;
state Victory;
state Nothing;
//-----
state Initialize
{
    //----- Goals -----
    RegisterGoal(goalDestroyControlCenter1, "translateGoal420a");
    RegisterGoal(goalDestroyControlCenter2, "translateGoal420b");
    RegisterGoal(goalDestroyControlCenter3, "translateGoal420c");
    RegisterGoal(goalDestroyControlCenter4, "translateGoal420d");

    RegisterGoal(goalDeactivePP1, "translateGoal420e");
    RegisterGoal(goalDeactivePP2, "translateGoal420f");
    RegisterGoal(goalDeactivePP3, "translateGoal420g");
    RegisterGoal(goalDeactivePP4, "translateGoal420h");

    EnableGoal(goalDestroyControlCenter1,true);
    EnableGoal(goalDestroyControlCenter2,true);
    EnableGoal(goalDestroyControlCenter3,true);
    EnableGoal(goalDestroyControlCenter4,true);

    //----- Players -----
    p_Player = GetPlayer(1); //ja czyli UCS
    p_Enemy1 = GetPlayer(3); //LC
    p_Enemy2 = GetPlayer(4); //LC
    p_Enemy3 = GetPlayer(5); //LC
    p_Enemy4 = GetPlayer(6); //LC

    p_Neutral = GetPlayer(7);
    p_Neutral.EnableAIFeatures(aiEnabled,false);
    p_Neutral.EnableStatistics(false);

    //----- AI -----
    if(GetDifficultyLevel()==0)
    {
        p_Enemy1.LoadScript("single\singleEasy");
        p_Enemy2.LoadScript("single\singleEasy");
        p_Enemy3.LoadScript("single\singleEasy");
        p_Enemy4.LoadScript("single\singleEasy");
    }

    if(GetDifficultyLevel()==1)
    {
        p_Enemy1.LoadScript("single\singleMedium");
        p_Enemy2.LoadScript("single\singleMedium");
        p_Enemy3.LoadScript("single\singleMedium");
        p_Enemy4.LoadScript("single\singleMedium");
    }

    if(GetDifficultyLevel()==2)
    {
        p_Enemy1.LoadScript("single\singleHard");
        p_Enemy2.LoadScript("single\singleHard");
        p_Enemy3.LoadScript("single\singleHard");
        p_Enemy4.LoadScript("single\singleHard");
    }

    //----- Money -----
    p_Player.EnableCommand(commandSoldBuilding,true);
    p_Player.SetMoney(30000);
    p_Player.SetMilitaryUnitsLimit(70000);

    p_Enemy1.SetMoney(10000);
    p_Enemy2.SetMoney(10000);
    p_Enemy3.SetMoney(10000);
    p_Enemy4.SetMoney(10000);

    p_Player.SetNeutral(p_Neutral);
    p_Enemy1.SetNeutral(p_Neutral);
    p_Enemy2.SetNeutral(p_Neutral);
    p_Enemy3.SetNeutral(p_Neutral);
    p_Enemy4.SetNeutral(p_Neutral);

    p_Enemy1.SetAlly(p_Enemy2);
    p_Enemy1.SetAlly(p_Enemy3);
    p_Enemy1.SetAlly(p_Enemy4);
}

```

```

p_Enemy2.SetAlly(p_Enemy3);
p_Enemy2.SetAlly(p_Enemy4);
p_Enemy3.SetAlly(p_Enemy4);

p_Enemy2.CopyResearches(p_Enemy1);
p_Enemy3.CopyResearches(p_Enemy1);
p_Enemy4.CopyResearches(p_Enemy1);
p_Enemy1.EnableBuilding("LCCSD", false); //Laser antykatietowy
p_Enemy2.EnableBuilding("LCCSD", false); //Laser antykatietowy
p_Enemy3.EnableBuilding("LCCSD", false); //Laser antykatietowy
p_Enemy4.EnableBuilding("LCCSD", false); //Laser antykatietowy

p_Player.EnableBuilding("UCSBWB", false); //Stocznia
p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu rudy
p_Player.EnableBuilding("UCSCSD", false); //Laser antykatietowy

//----- Timers -----
SetTimer(0, 1);
//----- Variables -----
bPP1active=true;
bPP2active=true;
bPP3active=true;
bPP4active=true;

//----- Units -----
uCC1 = GetUnit(GetPointX(1), GetPointY(1,0));
uCC2 = GetUnit(GetPointX(2), GetPointY(2,0));
uCC3 = GetUnit(GetPointX(3), GetPointY(3,0));
uCC4 = GetUnit(GetPointX(4), GetPointY(4,0));

uPP1 = GetUnit(GetPointX(5), GetPointY(5,0));
uPP2 = GetUnit(GetPointX(6), GetPointY(6,0));
uPP3 = GetUnit(GetPointX(7), GetPointY(7,0));
uPP4 = GetUnit(GetPointX(8), GetPointY(8,0));
//----- Artifacts -----
CreateArtifact("NEAPLATE1", GetPointX(9), GetPointY(9), GetPointZ(9),
9, artefactSpecialAIOther);
CreateArtifact("NEAPLATE1", GetPointX(10), GetPointY(10),
GetPointZ(10),10, artefactSpecialAIOther);
CreateArtifact("NEAPLATE1", GetPointX(11), GetPointY(11),
GetPointZ(11),11, artefactSpecialAIOther);
CreateArtifact("NEAPLATE1", GetPointX(12), GetPointY(12),
GetPointZ(12),12, artefactSpecialAIOther);
//----- Camera -----
CallCamera();
p_Player.LookAt(p_Player.GetStartingPointX(), p_Player.GetStartingPointY(),
6, 0, 20, 0);
return ShowBriefing;
}

//----- state ShowBriefing
{
    //show mission briefing
    AddBriefing("translateBriefing420a", p_Player.GetName());
    return Working;
}

//----- state Working
{
    if(!bCheckEndMission) return Working;
    bCheckEndMission=false;

    if((p_Player.GetNumberOfUnits()&&p_Player.GetNumberOfBuildings()))
    {
        AddBriefing("translateFailed420", p_Player.GetName());
        return EndMissionFailed;
    }

    if(GetGoalState(goalDestroyControlCenter1)!=goalAchieved &&
lUCC1.IsLive())
    {
        SetGoalState(goalDestroyControlCenter1,goalAchieved);
    }
    if(GetGoalState(goalDestroyControlCenter2)!=goalAchieved &&
lUCC2.IsLive())
    {
        SetGoalState(goalDestroyControlCenter2,goalAchieved);
    }
    if(GetGoalState(goalDestroyControlCenter3)!=goalAchieved &&
lUCC3.IsLive())
    {
        SetGoalState(goalDestroyControlCenter3,goalAchieved);
    }
}

//----- state EndMissionFailed
{
    EnableNextMission(0,2);
    return Nothing;
}

//----- state Victory
{
    EnableNextMission(0,3);
    return Victory, 500;
}

//----- state Nothing
{
    return Nothing, 500;
}

event Timer0() //wolany co minute
{
    //regeneration of HPs
    if(bPP1.IsActive())
    {
        if(bPP1active) uPP1.RegenerateHP();
        uCC1.RegenerateHP();
    }
    if(bPP2.IsActive())
    {
        if(bPP2active) uPP2.RegenerateHP();
        uCC2.RegenerateHP();
    }
    if(bPP3.IsActive())
    {
        if(bPP3active) uPP3.RegenerateHP();
        uCC3.RegenerateHP();
    }
    if(bPP4.IsActive())
    {
        if(bPP4active) uPP4.RegenerateHP();
        uCC4.RegenerateHP();
    }
}

event UnitDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event BuildingDestroyed(unit u_Unit)
{
    bCheckEndMission=true;
}

event Artefact(int aID,player piPlayer)
{
    if(piPlayer!= p_Player) return false;
    if(aID>12) return false;
    if(aID==9)
    {
        bPP1active=false;
        CreateArtifact("NEAPLATE1", GetPointX(aID), GetPointY(aID),
GetPointZ(aID), aID+10, artefactSpecialAIOther);
        SetGoalState(goalDeactivePP1,goalAchieved);
        AddBriefing("translateBriefing420b", p_Player.GetName());
    }
}

```

```

        }
        if(alD==10)
        {
            bPP2active=false;
            CreateArtifact("NEAPLATE1", GetPointX(alD), GetPointY(alD),
GetPointZ(alD), alD+10,artifactSpecialAIOther);
            SetGoalState(goalDeactivePP2,goalAchieved);
            AddBriefing("translateBriefing420c",p_Player.GetName());
        }
        if(alD==11)
        {
            bPP3active=false;
            CreateArtifact("NEAPLATE1", GetPointX(alD), GetPointY(alD),
GetPointZ(alD), alD+10,artifactSpecialAIOther);
            SetGoalState(goalDeactivePP3,goalAchieved);
            AddBriefing("translateBriefing420d",p_Player.GetName());
        }
        if(alD==12)
        {
            bPP4active=false;
            CreateArtifact("NEAPLATE1", GetPointX(alD), GetPointY(alD),
GetPointZ(alD), alD+10,artifactSpecialAIOther);
            SetGoalState(goalDeactivePP4,goalAchieved);
            AddBriefing("translateBriefing420e",p_Player.GetName());
        }
    }

    return true; //usawa sie
}

}

TutorialUCSmis.ec
mission "translateTutorialUCS"
{
    consts
    {
        buildPowerPlant = 0;
        buildVechProdCent = 1;
        buildRefinery = 3;
        buildTransporters = 4;
        harvestResources = 5;
        buildWeapProdCent = 6;
        buildArmy = 7;
        findEnemy = 8;
        destroyEnemy = 9;
    }

    player p_Player;
    player p_Enemy;

    int nDelayCount;
    int nBuildingCount;
    int nMoney;
    int nStartX;
    int nStartY;

    //*****
    state Initialize;
    state Start;
    state MoveCamera;
    state BuildPowerPlant;
    state BuildVechProdCent;
    state BuildRefinery;
    state BuildHarvesters;
    state HarvestResources;
    state BuildWeapProdCent;
    state BuildArmy;
    state More;
    state EndState;

    state Initialize
    {
        RegisterGoal(buildPowerPlant,"translateGoalTutorialUCS_PP");
        RegisterGoal(buildVechProdCent,"translateGoalTutorialUCS_BA");
        RegisterGoal(buildRefinery,"translateGoalTutorialUCS_RF");
        RegisterGoal(buildTransporters,"translateGoalTutorialUCS_OH");
        RegisterGoal(harvestResources,"translateGoalTutorialUCS_Mining");
        RegisterGoal(buildWeapProdCent,"translateGoalTutorialUCS_FA");
        RegisterGoal(buildArmy,"translateGoalTutorialUCS_tanks");
        RegisterGoal(findEnemy,"translateGoalTutorialUCS_findEnemy");
        RegisterGoal(destroyEnemy,"translateGoalTutorialUCS_destroyEnemy");

        p_Player=GetPlayer(1);
        p_Enemy=GetPlayer(2);

        p_Player.SetMoney(20000);
    }

    p_Enemy.SetMoney(20000);
    p_Player.EnableAIFeatures(aiEnabled,false);
    p_Enemy.EnableAIFeatures(aiEnabled,true);
    p_Enemy.EnableAIFeatures(aiControlOffense,false);

    //weapons
    p_Player.EnableResearch("RES_UCS_WACH2",false);
    p_Player.EnableResearch("RES_UCS_WSR2",false);
    p_Player.EnableResearch("RES_UCS_WASR1",false);
    p_Player.EnableResearch("RES_UCS_WSP2",false);
    p_Player.EnableResearch("RES_UCS_WHG1",false);
    p_Player.EnableResearch("RES_UCS_WSD",false);
    //ammo
    p_Player.EnableResearch("RES_UCS_WAPB1",false);
    p_Player.EnableResearch("RES_UCS_MMRC2",false);
    p_Player.EnableResearch("RES_UCS_MB2",false);
    p_Player.EnableResearch("RES_UCS_MG2",false);
    //chassis
    p_Player.EnableResearch("RES_UCS_GARG1",false);
    p_Player.EnableResearch("RES_UCS_USL3",false);
    p_Player.EnableResearch("RES_UCS_USS2",false);
    p_Player.EnableResearch("RES_UCS_UML3",false);
    p_Player.EnableResearch("RES_UCS_UHL1",false);
    p_Player.EnableResearch("RES_UCS_UNI1",false);
    p_Player.EnableResearch("RES_UCS_US1",false);
    p_Player.EnableResearch("RES_UCS_UBS1",false);
    p_Player.EnableResearch("RES_UCS_USM1",false);
    p_Player.EnableResearch("RES_UCS_UAH1",false);
    p_Player.EnableResearch("RES_UCS_BOMBER2",false);
    //special
    p_Player.EnableResearch("RES_UCS_BMD",false);
    p_Player.EnableResearch("RES_UCS_BHD",false);

    p_Player.EnableResearch("RES_UCS_RePhand2",false);
    p_Player.EnableResearch("RES_UCS_Sgen",false);
    p_Player.EnableResearch("RES_UCS_SHD",false);

    p_Player.EnableResearch("RES_UCS_WSD",false);
    p_Player.EnableResearch("RES_UCS_PC",false);

    // 1st tab
    p_Player.EnableBuilding("UCSBPP",false);
    p_Player.EnableBuilding("UCSBET",false);
    p_Player.EnableBuilding("UCSBA",false);
    p_Player.EnableBuilding("UCSBAF",false);
    p_Player.EnableBuilding("UCSBWB",false);
    p_Player.EnableBuilding("UCSAB",false);
    // 2nd tab
    p_Player.EnableBuilding("UCSBRF",false);
    p_Player.EnableBuilding("UCSBTB",false);
    // 3rd tab
    p_Player.EnableBuilding("UCSBST",false);
    // 4th tab
    p_Player.EnableBuilding("UCSBC",false);
    p_Player.EnableBuilding("UCSBTE",false);
    p_Player.EnableBuilding("UCSBHQ",false);
    p_Player.EnableBuilding("UCSBEN1",false);
    p_Player.EnableBuilding("UCSBLZ",false);

    nStartX = p_Player.GetStartingPointX();
    nStartY = p_Player.GetStartingPointY();

    LookAt(nStartX,nStartY,6,0,20,0);

    SetTimer(0,6000); //5min
    SetTimer(1,100); //5sec
    nBuildingCount=0;

    return Start;
}
//-----
state Start
{
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,2500,800,
10);
    AddBriefing("translateTutorialUCS_moveCamera");
    return MoveCamera,600;
}
//-----
state MoveCamera

```

```

    {
        EnableGoal(buildPowerPlant,true);
        p_Player.EnableBuilding("UCSBPP",true);
        AddBriefing("translateTutorialUCS_PP");
        return BuildPowerPlant,100;
    }
}

state BuildPowerPlant
{
    if(p_Player.GetNumberOfBuildings(buildingPowerPlant))
    {
        p_Player.EnableBuilding("UCSBA",true);
        p_Player.EnableBuilding("UCSBET",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildPowerPlant,goalAchieved);
        EnableGoal(buildVechProdCent,true);
        if(p_Player.GetNumberOfBuildings(buildingBase))
            AddBriefing("translateTutorialUCS_BA");
        return BuildVechProdCent,100;
    }
    if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
    {
        p_Player.CreateUnitEx(nStartX,nStartY,
0,null,"UCSUB1",null,null,null);
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildPowerPlant,50;
}
}

state BuildVechProdCent
{
    if(p_Player.GetNumberOfBuildings(buildingBase))
    {
        p_Player.EnableBuilding("UCSBRF",true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildVechProdCent,goalAchieved);
        EnableGoal(buildRefinery,true);
        if(p_Player.GetNumberOfBuildings(buildingRefinery))
            AddBriefing("translateTutorialUCS_RF");
        return BuildRefinery,100;
    }
    if(p_Player.GetNumberOfBuildings(>nBuildingCount))
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialUCS_BA_Fool");
    }
    if(p_Player.GetNumberOfUnits(chassisTank | unitBuilder))
    {
        p_Player.CreateUnitEx(nStartX,nStartY,
0,null,"UCSUB1",null,null,null);
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildVechProdCent,50;
}
}

state BuildRefinery
{
    if(p_Player.GetNumberOfBuildings(buildingRefinery))
    {
        nBuildingCount=p_Player.GetNumberOfBuildings();
        SetGoalState(buildRefinery,goalAchieved);
        EnableGoal(buildTransporters,true);
        if(p_Player.GetNumberOfUnits(chassisTank | unitCarrier)<2)
            AddBriefing("translateTutorialUCS_OH");
        return BuildHarvesters,100;
    }
    if(p_Player.GetNumberOfBuildings(>nBuildingCount))
    {
        nBuildingCount=nBuildingCount+1;
        AddBriefing("translateTutorialUCS_RF_Fool");
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
    return BuildRefinery,50;
}
}

state BuildHarvesters
{
    if(p_Player.GetNumberOfUnits(unitHarvester | chassisAny)>=2)
    {
        SetGoalState(buildTransporters,goalAchieved);
        EnableGoal(harvestResources,true);
        AddBriefing("translateTutorialUCS_Mining");
        nMoney=p_Player.GetMoney();
        return HarvestResources,100;
    }
    if(p_Player.GetMoney()<2000) p_Player.AddMoney(2000);
}
}

state HarvestResources
{
    if(nMoney < p_Player.GetMoney())
    {
        p_Player.EnableBuilding("UCSBA",true);
        SetGoalState(harvestResources,goalAchieved);
        EnableGoal(buildWeapProdCent,true);
        if(p_Player.GetNumberOfBuildings(buildingFactory))
            AddBriefing("translateTutorialUCS_FA");
        return BuildWeapProdCent,100;
    }
    return HarvestResources,50;
}
}

state BuildWeapProdCent
{
    if(p_Player.GetNumberOfBuildings(buildingFactory))
    {
        SetGoalState(buildWeapProdCent,goalAchieved);
        EnableGoal(buildArmy,true);
        nBuildingCount=p_Player.GetNumberOfBuildings();
        AddBriefing("translateTutorialUCS_tanks");
        return BuildArmy;
    }
    if(p_Player.GetNumberOfBuildings(>nBuildingCount))
    {
        nBuildingCount=nBuildingCount+1;
    }
    return BuildWeapProdCent,100;
}
}

state BuildArmy
{
    if(p_Player.GetNumberOfUnits(chassisTank | unitArmed)>=5)
    {
        SetGoalState(buildArmy,goalAchieved);
        EnableGoal(findEnemy,true);
        if(GetGoalState(findEnemy)==goalAchieved)
            AddBriefing("translateTutorialUCS_findEnemy");
        return More,600;
    }
    return BuildArmy,200;
}
}

state More
{
    p_Player.EnableBuilding("UCSBET",true);
    p_Player.EnableBuilding("UCSBA",true);
    p_Player.EnableBuilding("UCSBST",true);
    p_Player.EnableBuilding("UCSBC",true);
    p_Player.EnableBuilding("UCSBEN1",true);
    AddBriefing("translateTutorialUCS_more");
    return EndState,100;
}
}

state EndState
{
    return EndState,500;
}
}

event Timer0()
{
    Rain(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),30,400,2500,800,
10);
}
}

event Timer1()
{
    if(GetGoalState(findEnemy)==goalAchieved &&
p_Player.IsPointLocated(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY())
}

```

```

).0))
    {
    //LookAt(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),6,0,20,0,);
    SetGoalState(!finEnemy.goalAchieved);
    EnableGoal(destroyEnemy,true);
    AddBriefing("translateTutorialUCS_destroyEnemy");
    }
    if(getGoalState(destroyEnemy)!=goalAchieved &&
    !p_Enemy.GetNumberOfUnits())
    {
        SetGoalState(destroyEnemy.goalAchieved);
        AddBriefing("translateTutorialUCS_Victory");
    }
}
}

UCSbase1script.ec
mission "translateMissionUCSBaseUCSMP01"
{
    consts
    {

        scriptFieldGoal1=0;
        scriptFieldGoal2=1;
        scriptFieldGoal3=2;
        scriptFieldGoal4=3;
        scriptFieldGoal5=4;
        scriptFieldGoal6=5;
        scriptFieldMoney=9;
        scriptFieldMeteors=10;
    }
    player p_Player;
    int i;

    state Initialize;
    state Waiting;
    state PrepareShowBriefing;
    state ShowBriefing;
    state Nothing;
    state EndGameState;

    state Initialize
    {
        unitex u_Grizzly1;
        unitex u_Grizzly2;

        //----- Goals -----
        RegisterGoal(0,"translateGoalCaptureInfo");
        RegisterGoal(1,"translateGoalSunLightFail");
        RegisterGoal(2,"translateGoalDestroyALPHA");
        RegisterGoal(3,"translateGoalDestroyBETA");
        RegisterGoal(4,"translateGoalDestroyGAMMA");
        RegisterGoal(5,"translateGoalDestroyDELTA");
        RegisterGoal(6,"translateGrizzly1HaveToSurvive");
        RegisterGoal(7,"translateGrizzly2HaveToSurvive");

        EnableGoal(0,true);
        EnableGoal(6,true);
        EnableGoal(7,true);

        //----- Players -----
        p_Player = GetPlayer(1);

        p_Player.EnableBuilding("UCSBWB", false); //Stocznia
        p_Player.EnableBuilding("UCSBTB", false); //Centrum transportu rudy
        p_Player.EnableBuilding("UCCSO", false); //Laser antyrakietowy

        //p_Player.AddResearch("RES_UCS_WSR1"); //Wyrzutnia rakiet
        //p_Player.AddResearch("RES_UCS_UML1"); //Spider
        //p_Player.AddResearch("RES_UCS_UHL1"); //Panther

        p_Player.AddResearch("RES_MISSION_PACK1_ONLY");
        p_Player.EnableResearch("RES_UCS_US1",false); //Shark - 3dY
        p_Player.EnableResearch("RES_UCS_US2",false); //Shark - 3dY
        p_Player.EnableResearch("RES_UCS_USM1",false); //Orca - 3dY podwod-
na
        p_Player.EnableResearch("RES_UCS_USM2",false); //Orca - 3dY podwod-
na
    }

    //LookAt(p_Enemy.GetStartingPointX(),p_Enemy.GetStartingPointY(),6,0,20,0,);
    SetGoalState(!finEnemy.goalAchieved);
    EnableGoal(destroyEnemy,true);
    AddBriefing("translateTutorialUCS_destroyEnemy");
    }
    if(getGoalState(destroyEnemy)!=goalAchieved &&
    !p_Enemy.GetNumberOfUnits())
    {
        SetGoalState(destroyEnemy.goalAchieved);
        AddBriefing("translateTutorialUCS_Victory");
    }
}

plazmowe

p_Player.EnableResearch("RES_UCS_IBS1",false); //Hydra - 3dY
p_Player.EnableResearch("RES_UCS_WSD",false); //laser antyrakietowy
p_Player.EnableResearch("RES_UCS_PC",false); //Stacjonarne dzia³o

p_Player.EnableResearch("RES_UCS_ART",false);
//----- 413
p_Player.EnableResearch("RES_UCS_USL2",false);
p_Player.EnableResearch("RES_UCS_GARG1",false);
p_Player.EnableResearch("RES_UCS_WH2",false);
p_Player.EnableResearch("RES_UCS_WSP1",false);
p_Player.EnableResearch("RES_UCS_WSR1",false);
p_Player.EnableResearch("RES_UCS_WSG2",false);
p_Player.EnableResearch("RES_MCH2",false);
p_Player.EnableResearch("RES_MSR2",false);
p_Player.EnableResearch("RES_UCS_MG2",false);
p_Player.EnableResearch("RES_UCS_Rephand",false);
p_Player.EnableResearch("RES_UCS_SGen",false);
//----- 414
p_Player.EnableResearch("RES_UCS_UML1",false);
p_Player.EnableResearch("RES_MCH3",false);
p_Player.EnableResearch("RES_MS3",false);
p_Player.EnableResearch("RES_UCS_MG3",false);
p_Player.EnableResearch("RES_UCS_MGen",false);
//----- 415
p_Player.EnableResearch("RES_UCS_GARG1",false);
p_Player.EnableResearch("RES_UCS_UHL1",false);
p_Player.EnableResearch("RES_UCS_WHG1",false);
p_Player.EnableResearch("RES_UCS_WHP1",false);
p_Player.EnableResearch("RES_UCS_WHP2",false);
p_Player.EnableResearch("RES_UCS_WSR3",false);
p_Player.EnableResearch("RES_UCS_WASR1",false);
p_Player.EnableResearch("RES_UCS_WMR1",false);
p_Player.EnableResearch("RES_MMR2",false);
p_Player.EnableResearch("RES_UCS_Rephand2",false);
p_Player.EnableResearch("RES_UCS_HGen",false);
p_Player.EnableResearch("RES_UCS_BMD",false);
p_Player.EnableResearch("RES_UCS_SHD",false);
//----- 416
p_Player.EnableResearch("RES_UCS_UBL1",false);
p_Player.EnableResearch("RES_UCS_UML1",false);
p_Player.EnableResearch("RES_UCS_BOMBER21",false);
p_Player.EnableResearch("RES_UCS_WHP3",false);
p_Player.EnableResearch("RES_UCS_WMR2",false);
p_Player.EnableResearch("RES_UCS_WAMR1",false);
p_Player.EnableResearch("RES_MMR3",false);
//----- 417
p_Player.EnableResearch("RES_UCS_BHD",false);
p_Player.EnableResearch("RES_UCS_BOMBER22",false);
p_Player.EnableResearch("RES_UCS_SHD2",false);
//----- 418
p_Player.EnableResearch("RES_UCS_BOMBER31",false);
p_Player.EnableResearch("RES_UCS_WPB1",false);
p_Player.EnableResearch("RES_UCS_SHD3",false);
//----- 419
p_Player.EnableResearch("RES_UCS_WAPB2",false);
p_Player.EnableResearch("RES_UCS_MB2",false);
p_Player.EnableResearch("RES_UCS_SHD4",false);

p_Player.EnableResearch("RES_UCSUCS",false);
p_Player.EnableResearch("RES_UCSWE1",false);
p_Player.EnableResearch("RES_UCSBH1",false);
//-- Variables for campaign goals --
p_Player.SetScriptData(0,0);
p_Player.SetScriptData(1,0);
p_Player.SetScriptData(2,0);
p_Player.SetScriptData(3,0);
p_Player.SetScriptData(4,0);
p_Player.SetScriptData(5,0);

p_Player.EnableCommand(commandSoldBuilding,true);
//----- Units -----
u_Grizzly1 = GetUnit((GetPointX(1),GetPointY(1),GetPointZ(1)));
u_Grizzly2 = GetUnit((GetPointX(2),GetPointY(2),GetPointZ(2));
u_Grizzly1.SetUnitName("translateGrizzly1");
u_Grizzly2.SetUnitName("translateGrizzly2");

p_Player.SetScriptUnit(1,u_Grizzly1);
p_Player.SetScriptUnit(2,u_Grizzly2);
//----- Money -----
p_Player.SetMoney(0);
p_Player.SetMilitaryUnitsLimit(15000);
//----- Camera -----
CallCamera();

```

```

p_Player.LookAt(p_Player.GetStartingPointX(),p_Player.GetStartingPointY(),6,0,20,
0);

    EnableUnitSounds(false);
    EnableBuildingSounds(false);

    return Waiting;
}

//-----
state Waiting
{
}

state PrepareShowBriefing
{
    return ShowBriefing,100;
}
state ShowBriefing
{
    unitex u_Grizzly1;
    unitex u_Grizzly2;
    AddBriefing("translateStartCampaignUCSMP01",p_Player.GetName());
    u_Grizzly1 = p_Player.GetScriptUnit(1);
    p_Player.AddUnitToSpecialTab(u_Grizzly1,true,-1);
    u_Grizzly2 = p_Player.GetScriptUnit(2);
    p_Player.AddUnitToSpecialTab(u_Grizzly2,true,-1);

    EnableUnitSounds(true);
    EnableBuildingSounds(true);

    TraceD("EnableGameFeature(lockResearchDialog,fase); \n");
    EnableGameFeature(lockResearchDialog,fase);
    EnableGameFeature(lockConstructionDialog,fase);
    EnableGameFeature(lockUpgradeWeaponDialog,fase);

    return Nothing,50;
}
//-----

state Nothing
{
    unitex u_Grizzly1;
    unitex u_Grizzly2;

    if(p_Player.GetScriptData(scriptFieldMeteors)==10)
    {
        p_Player.SetScriptData(scriptFieldMeteors,0);
        MeteorRain(p_Player.GetStartingPointX(),
p_Player.GetStartingPointY(),20,500,1200,500,10,10);
    }
    if(p_Player.GetScriptData(scriptFieldMoney))//Hash from missions after the
end.
    {
        p_Player.AddMoney(p_Player.GetScriptData(scriptFieldMoney));
        p_Player.SetScriptData(scriptFieldMoney,0);
    }
    //---checking campaign goals---
    for(i=0; i<6; i+=1)
    {
        if(p_Player.GetScriptData(i)==1)
        {
            EnableGoal(i,true);
        }

        if(p_Player.GetScriptData(i)==2)
        {
            SetGoalState(i,goalAchieved);
        }
    }
    u_Grizzly1 = p_Player.GetScriptUnit(1);
    u_Grizzly2 = p_Player.GetScriptUnit(2);

    if((p_Player.GetNumberOfBuildings())
    {
        AddBriefing("translateCampaignUCSBaseDestroyed",p_Player.GetName());
        return EndGameState,5;
    }
    if((u_Grizzly1==null || !u_Grizzly1.IsLive())&&
(u_Grizzly2==null || !u_Grizzly2.IsLive()))
    {
        SetGoalState(6,goalFailed);
        SetGoalState(7,goalFailed);
    }
}
}

AddBriefing("translateBothGrizzliesKilled",p_Player.GetName());
return EndGameState,5;
}

if(u_Grizzly1==null || !u_Grizzly1.IsLive())
{
    SetGoalState(6,goalFailed);
    AddBriefing("translateGrizzly1Killed",p_Player.GetName());
    return EndGameState,5;
}

if(u_Grizzly2==null || !u_Grizzly2.IsLive())
{
    SetGoalState(7,goalFailed);
    AddBriefing("translateGrizzly2Killed",p_Player.GetName());
    return EndGameState,5;
}
return Nothing,50;
}

//-----
state EndGameState
{
    EnableNextMission(0,2);//end campaign
    return EndGameState,600;
}

//-----
event CustomEvent0(int k1,int k2,int k3,int k4) //XXXMD
{
    if(k1==1)
    {
        state ShowBriefing;
    }
}
}



### UCSbase2script.ec


mission "translateMissionUCSBaseLCMP01"
{
    player p_PlayerLC;

    state Initialize;
    state Nothing;
    state Initialize
    {
        p_PlayerLC=GetPlayer(3);
        p_PlayerLC.SetMoney(10000);

        p_PlayerLC.EnableAIFeatures(aiBuildTanks,fase);
        p_PlayerLC.EnableAIFeatures(aiBuildShips,fase);
        p_PlayerLC.EnableAIFeatures(aiBuildHelicopters,fase);

        p_PlayerLC.EnableAIFeatures(aiBuildSpecialUnits,fase);

        //----- Researches -----
        /* /413
        p_PlayerLC.EnableResearch("RES_LC_WCH2",false); //Sprzętowe dzia³ka 20
mm
        p_PlayerLC.EnableResearch("RES_LC_ACH2",false); //Lotnicze dzia³ka 20
mm
        p_PlayerLC.EnableResearch("RES_LC_WSR1",false); //Wyrzutnia rakiet R1
(byle "RES_LC_WSR2")
        p_PlayerLC.EnableResearch("RES_LC_ASР1",false); //Wyrzutnia rakiet R1a
        p_PlayerLC.EnableResearch("RES_LC_WSL1",false); //Dzia³o elektroczne E1
        */

        //414
        p_PlayerLC.EnableResearch("RES_LC_WMR1",false); //Wyrzutnia
ciê¿kich rakiet R12
        p_PlayerLC.EnableResearch("RES_LC_AMR1",false); //Wyrzutnia ciê¿kich
rakiet R1a
        p_PlayerLC.EnableResearch("RES_LC_WHL1",false); //Dzia³o elektroczne E3
        p_PlayerLC.EnableResearch("RES_LC_UCR1",false); //Crater m1

        //415
        p_PlayerLC.EnableResearch("RES_LC_WSS1",false); //miotacz dzwiekowy D1
        p_PlayerLC.EnableResearch("RES_LC_WHS1",false); //-. D3
        p_PlayerLC.EnableResearch("RES_LC_UCU1",false); //Crusher m1

        //416
        p_PlayerLC.EnableResearch("RES_LC_WAS1",false); //-. D5
        p_PlayerLC.EnableResearch("RES_LC_UBO1",false); //Thunder m1
}

```

```

//417
p_PlayerLC.EnableResearch("RES_LC_WARTILLERY",false);
//Artyleria plazmowa
*/
p_PlayerLC.AddResearch("RES_MISSION_PACK1_ONLY");
p_PlayerLC.EnableResearch("RES_LC_WSS1",false); //miotacz dzwie-
kowy D1
p_PlayerLC.EnableResearch("RES_LC_WHS1",false); // -.- D3
//-----411
p_PlayerLC.EnableResearch("RES_LC_SGen",false);
p_PlayerLC.EnableResearch("RES_LC_BMD",false);
p_PlayerLC.EnableResearch("RES_MCH2",false);
//-----412
p_PlayerLC.EnableResearch("RES_LC_WCH2",false);
p_PlayerLC.EnableResearch("RES_LC_WSR2",false);
p_PlayerLC.EnableResearch("RES_MSR2",false);
p_PlayerLC.EnableResearch("RES_MCH3",false);
//-----413
p_PlayerLC.EnableResearch("RES_LC_UME1",false);
p_PlayerLC.EnableResearch("RES_LC_ULL2",false);
p_PlayerLC.EnableResearch("RES_LC_UMO2",false);
p_PlayerLC.EnableResearch("RES_LC_ACH2",false);
p_PlayerLC.EnableResearch("RES_LC_ASRI",false);
p_PlayerLC.EnableResearch("RES_LC_WSL1",false);
p_PlayerLC.EnableResearch("RES_MSR2",false);
//-----414
p_PlayerLC.EnableResearch("RES_LC_WSL1",false);
p_PlayerLC.EnableResearch("RES_LC_WSR3",false);
p_PlayerLC.EnableResearch("RES_LC_MGen",false);
//-----415
p_PlayerLC.EnableResearch("RES_LC_UCR1",false);
p_PlayerLC.EnableResearch("RES_LC_WHL1",false);
p_PlayerLC.EnableResearch("RES_LC_WMR1",false);
p_PlayerLC.EnableResearch("RES_MMR2",false);
p_PlayerLC.EnableResearch("RES_LC_BHD",false);
//-----416
p_PlayerLC.EnableResearch("RES_LC_UCU1",false);
p_PlayerLC.EnableResearch("RES_LC_UBO1",false);
p_PlayerLC.EnableResearch("RES_LC_AMR1",false);
p_PlayerLC.EnableResearch("RES_LC_HGen",false);

//SPECIAL
p_PlayerLC.EnableResearch("RES_LC_BWC",false); //Centrum kontroli
pody
p_PlayerLC.EnableResearch("RES_LC_SDIDEF",false); //Laser antyrakieto-
wy
p_PlayerLC.EnableBuilding("LCCSD", false); //Laser antyrakietowy
p_PlayerLC.EnableResearch("RES_LC_WSS1",false);

EnableUnitSounds(false);
EnableBuildingSounds(false);
return Nothing;
}
state Nothing
{
}
event CustomEvent0(int k1,int k2,int k3,int k4) //XXXMD
{
    if(k1==1)
    {
        EnableUnitSounds(true);
        EnableBuildingSounds(true);
    }
}
}

```

UCSbase3script.ec

```

mission "translateMissionUCSBaseLCMPO1"
{
    player p_PlayerLC;

    state Initialize;
    state Nothing;

    state Initialize
    {
        p_PlayerLC=GetPlayer(3);
        p_PlayerLC.SetMoney(10000);
        p_PlayerLC.EnableAIFeatures(aiBuildTanks,false);
        p_PlayerLC.EnableAIFeatures(aiBuildShips,false);
        p_PlayerLC.EnableAIFeatures(aiBuildHelicopters,false);
        p_PlayerLC.EnableAIFeatures(aiBuildSpecialUnits,false);
    }
}

```

```

EnableUnitSounds(false);
EnableBuildingSounds(false);

return Nothing;
}
state Nothing
{
}
event CustomEvent0(int k1,int k2,int k3,int k4) //XXXMD
{
    if(k1==1)
    {
        EnableUnitSounds(true);
        EnableBuildingSounds(true);
    }
}
}

```

GameTypes MP

UncleSamMP.ec

```

mission "translateGameTypeUncleSamMP"
{

```

```

int nCashRate;
int nCashTime;

int bGameEnded;

enum comboMoney
{
    "translateScript10000CR",
    "translateScript15000CR",
    "translateScript20000CR",
    "translateScript30000CR",
    "translateScript40000CR",
    "translateScript50000CR",
}
multi:
    "translateGameMenuStartingMoney"
}

```

```

enum comboCashRate
{
    "translateScript1000CR",
    "translateScript12500CR",
    "translateScript15000CR",
    "translateScript10000CR",
    "translateScript15000CR",
    "translateScript20000CR",
}
multi:
    "translateGameMenuCashRate"
}

```

```

enum comboCashTime
{
    "translateGameMenuRateFrequency1min",
    "translateGameMenuRateFrequency3min",
    "translateGameMenuRateFrequency5min",
    "translateGameMenuRateFrequency10min",
    "translateGameMenuRateFrequency15min",
}
multi:
    "translateGameMenuRateFrequency"
}

```

```

enum comboStartingUnits
{
    "translateGameMenuStartingUnitsDefault",
    "translateGameMenuStartingUnitsBuilderOnly",
}
multi:
    "translateGameMenuStartingUnits"
}

```

```

enum comboUnitsLimit
{
    "translateGameMenuUnitsLimitNoLimit",
    "translateGameMenuUnitsLimit10000CR",
    "translateGameMenuUnitsLimit20000CR",
    "translateGameMenuUnitsLimit30000CR",
    "translateGameMenuUnitsLimit50000CR",
}
multi:
    "translateGameMenuUnitsLimit"
}

```

```

enum comboAlliedVictory
{
    "translateGameMenuAlliedVictoryNo",
    "translateGameMenuAlliedVictoryYes",
    multi:
    "translateGameMenuAlliedVictory"
}

state Initialize;
state Nothing;

state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

    if(comboMoney==0)nStartingMoney=10000;
    if(comboMoney==1)nStartingMoney=15000;
    if(comboMoney==2)nStartingMoney=20000;
    if(comboMoney==3)nStartingMoney=30000;
    if(comboMoney==4)nStartingMoney=40000;
    if(comboMoney==5)nStartingMoney=50000;

    if(comboCashRate==0)nCashRate = 1000;
    if(comboCashRate==1)nCashRate = 2500;
    if(comboCashRate==2)nCashRate = 5000;
    if(comboCashRate==3)nCashRate = 10000;
    if(comboCashRate==4)nCashRate = 15000;
    if(comboCashRate==5)nCashRate = 20000;

    if(comboCashTime==0)nCashTime=1*60*20;
    if(comboCashTime==1)nCashTime=3*60*20;
    if(comboCashTime==2)nCashTime=5*60*20;
    if(comboCashTime==3)nCashTime=10*60*20;
    if(comboCashTime==4)nCashTime=15*60*20;

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if (rPlayer==null)
        {
            if(rPlayer.GetRace()==raceUCS)
            {
                rPlayer.EnableBuilding("UCSBLZ", false);
                rPlayer.EnableBuilding("UCSBTB", false);
            }
            if(rPlayer.GetRace()==raceED)
            {
                rPlayer.EnableBuilding("EDBLZ", false);
                rPlayer.EnableBuilding("EDBTC", false);
            }
            if(rPlayer.GetRace()==raceLC)
            {
                rPlayer.EnableBuilding("LCBLZ", false);
                rPlayer.EnableBuilding("LCBSR", false);
            }
        }
    }

    bGameEnded=false;
    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);

        if(rPlayer!=null)
        {
            rPlayer.SetMaxDistance(30);
            if(comboAlliedVictory)
                rPlayer.EnableAllFeatures2(ai2BNSendResult,false); //nie wysylac
        }
    }
    resultatow do EARTH Netu

    rPlayer.SetMoney(nStartingMoney);

    rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMiningUnits,false);

    rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);

    if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
    if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
    if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
    if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
    if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);
}

if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
{
    rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0,0);
    rPlayer.EnableCommand(commandSoldBuilding,true);
}

rPlayer.AddResearch("RES_MISSION_PACK1_ONLY");//Enable mission pack units
and buildings

rPlayer.EnableBuilding("EDBML",false);
rPlayer.EnableBuilding("EDBRE",false);
rPlayer.EnableBuilding("UCSBRF",false);
rPlayer.EnableBuilding("LCBMR",false);

rPlayer.EnableResearch("RES_UCS_UOH2",false);
rPlayer.EnableResearch("RES_UCS_UOH3",false);
rPlayer.EnableResearch("RES_UCS_UAH1",false);
rPlayer.EnableResearch("RES_UCS_UAH2",false);
rPlayer.EnableResearch("RES_UCS_UAH3",false);

}

SetTimer(0,100);
SetTimer(1,nCashTime);
SetTimer(2,200);
SetTimer(5,200);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    true;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenDestroyed;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    rLastPlayer=null;
    iAlivePlayers=0;
    if(comboAlliedVictory)//-----
    {
        bOneHasBeenDestroyed=false;
        for(i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if ((iCountBuilding==0){rPlayer.Defeat();}

                if(rPlayer!=null && !rPlayer.IsAlive())
                    bOneHasBeenDestroyed=true;
            }
        }
        bActiveEnemies=false;
        for(i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                for(j=i+1;j<15;j=j+1)

```

```

    {
        rPlayer2 = GetPlayer(i);
        if((rPlayer2!=null && rPlayer2.IsAlive()) &&
        !rPlayer.IsAlly(rPlayer2))
        {
            bActiveEnemies=true;
        }
    }
    if(bActiveEnemies) return;
    if(!bOneHasBeenDestroyed) return;

    for(i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            rPlayer.Victory();
        }
    }
    else//-----
    {
        for(i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if (iCountBuilding>0)
                {
                    iAlivePlayers=iAlivePlayers+1;
                    rLastPlayer=rPlayer;
                }
                else
                {
                    rPlayer.Defeat();
                }
            }
            if (iAlivePlayers==1 && rLastPlayer!=null)
            {
                rLastPlayer.Victory();
            }
        }
    }
}

event Timer1()
{
    int i;
    int money;
    player rPlayer;

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMoney()<100000)
                rPlayer.AddMoney(nCashRate);
        }
    }
}

event Timer2()
{
    int i;
    player rPlayer;

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMaxDistance()<255)
                rPlayer.SetMaxDistance(rPlayer.GetMaxDistance()+1);
        }
    }
}

event Timer5()
{
    int i;
    int j;
}

player rPlayer;
player rPlayer2;
//sprawdzanie zenablowania szpiegowanie
//(po 10 minutach - warunek player ma 2 lub mniej budynki inne niz wie-
yczki
//i 4 lub mniej wiezyczki i 6 lub mniej unitow)
//jesli warunek nie jest spełniony to z powrotem disableujemy szpiegowanie
if (GetMissionTime() > 10*60*20)
{
    for (i = 0; i < 15; i = i + 1)
    {
        rPlayer = GetPlayer(i);
        if ((rPlayer != null) && rPlayer.IsAlive())
        {
            if ((rPlayer.GetNumberOfBuildings(buildingNormal) <= 4) &&
((rPlayer.GetNumberOfBuildings() -
rPlayer.GetNumberOfBuildings(buildingNormal)) <= 2) &&
(rPlayer.GetNumberOfUnits() <= 6))
            {
                for (j = 0; j < 15; j = j + 1)
                {
                    rPlayer2 = GetPlayer(j);
                    if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive() &&
rPlayer2.IsAlly(rPlayer))
                    {
                        if
(rPlayer2.IsCommandDisabled(commandBuildingSpyPlayer0 + i))
                            rPlayer2.EnableCommand(commandBuildingSpyPlayer0
+ i, true);
                    }
                }
            }
            else
            {
                for (j = 0; j < 15; j = j + 1)
                {
                    rPlayer2 = GetPlayer(j);
                    if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive())
                    {
                        if
(rPlayer2.IsCommandEnabled(commandBuildingSpyPlayer0 + i))
                            rPlayer2.EnableCommand(commandBuildingSpyPlayer0
+ i, false);
                    }
                }
            }
        }
    }
}

command Initialize()
{
    nCashRate = 5000;
    comboCashRate = 2;
    nCashTime = 60*20;
    comboStartingUnits=1;
    comboAlliedVictory=1;
    comboUnitsLimit=2;
}
command Combo1(int nMode) button comboMoney
{
    comboMoney=nMode;
}
command Combo2(int nMode) button comboCashRate
{
    comboCashRate = nMode;
}
command Combo3(int nMode) button comboCashTime
{
    comboCashTime = nMode;
}
command Combo4(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}
command Combo5(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}
command Combo6(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}
}

```

ArenaMP.ec

mission "translateGameTypeArena"

```
{  
    int bDemo;  
    int nTimeLimit;  
  
    int bCheckBuilding;  
    int nMeteor;  
    int nPlayerInConsole;  
    enum comboTechLevel  
    {  
        "translateGameMenuTechLevelLow",  
        "translateGameMenuTechLevelMedium",  
        "translateGameMenuTechLevelHigh",  
        multi:  
            "translateGameMenuTechLevel"  
        }  
        enum comboStartingUnits  
        {  
            "1",  
            "2",  
            "3",  
        multi:  
            "translateGameMenuStartingUnits"  
        }  
        enum comboTime  
        {  
            "translateGameMenuTimeLimitNoLimit",  
            "translateGameMenuTimeLimit15min",  
            "translateGameMenuTimeLimit30min",  
            "translateGameMenuTimeLimit145min",  
            "translateGameMenuTimeLimit1h",  
            "translateGameMenuTimeLimit15h",  
        multi:  
            "translateGameMenuTimeLimit"  
        }  
    }  
    //***** F U N C T I O N S *****  
    //***** F U N C T I O N S *****  
    function int CreateArtefacts(player rPlayer)  
    {  
        int i;  
        int x;  
        int y;  
        if(!rPlayer) return false;  
        x = rPlayer.GetStartingPointX();  
        y = rPlayer.GetStartingPointY();  
  
        CreateArtefact("NEAAMMO",x, y ,0,-1,-1);  
        CreateArtefact("NEAAMMO",x-1,y ,0,-1,-1);  
        CreateArtefact("NEAAMMO",x+1,y ,0,-1,-1);  
  
        CreateArtefact("NEAENERGY",x, y ,0,-1,-1);  
        CreateArtefact("NEAENERGY",x-1,y+1,0,-1,-1);  
        CreateArtefact("NEAENERGY",x+1,y+1,0,-1,-1);  
  
        CreateArtefact("NEAMAXHP",x, y,-1,0,-1,-1);  
        CreateArtefact("NEAMAXHP",x-1,y+1,0,-1,-1);  
        CreateArtefact("NEAMAXHP",x+1,y-1,0,-1,-1);  
  
        CreateArtefact("NEASHOWMAP",x, y+2,0,-1,-1);  
        return true;  
    }  
  
    function int CreateStartingUnits(player rPlayer,player rPlayer2)  
    {  
        int x;  
        int y;  
        if(!rPlayer) return false;  
        if(!rPlayer2) return false;  
        x = rPlayer2.GetStartingPointX();  
        y = rPlayer2.GetStartingPointY();  
  
        if(comboTechLevel==0)  
        {  
            if (rPlayer.GetRace() == raceED)  
            {  
                rPlayer.CreateUnitEx(x,y,  
0,null,"EDUMT3","EDWSL1",null,null,0);  
                if(comboStartingUnits>0)  
                    rPlayer.CreateUnitEx(x+1,y,  
0,null,"EDUST3","EDWCA2",null,null,0);  
                if(comboStartingUnits>1)  
                    rPlayer.CreateUnitEx(x,y+1,y,  
0,null,"EDUMT3","EDWSR3",null,null,0);  
                if(comboStartingUnits>2)  
                    rPlayer.CreateUnitEx(x,y+2,y,  
0,null,"EDUMT3","EDWSL2",null,null,0);  
            }  
            if (rPlayer.GetRace() == raceLC)  
            {  
                rPlayer.CreateUnitEx(x,y, 0,null,"LCUMO1","LCWLS1",null,null,0);  
                if(comboStartingUnits>0)  
                    rPlayer.CreateUnitEx(x+1,y,  
0,null,"LCUMO3","LCWSR3",null,null,0);  
                if(comboStartingUnits>1)  
                    rPlayer.CreateUnitEx(x,y+1,y,  
0,null,"LCUMO2","LCWS2",null,null,0);  
                if (rPlayer.GetRace() == raceUCS)  
                {  
                    rPlayer.CreateUnitEx(x,y,0,null,"UCSUML3","UCSWSSP2",null,null,2);  
                    if(comboStartingUnits>0)  
                        rPlayer.CreateUnitEx(x+1,y,0,null,"UCSUSL3","UCSWTSR3",null,null,0);  
                    if(comboStartingUnits>1)  
                        rPlayer.CreateUnitEx(x-1,y,  
0,null,"UCSUML3","UCSWSMR1",null,null,0);  
                }  
            }  
            if(comboTechLevel==1)  
            {  
                if (rPlayer.GetRace() == raceED)  
                {  
                    rPlayer.CreateUnitEx(x,y, 0,null,"EDUHT3","EDWMR3",null,null,1);  
                    if(comboStartingUnits>0)  
                        rPlayer.CreateUnitEx(x+1,y,  
0,null,"EDUHL2","EDWSL2",null,2);  
                    if(comboStartingUnits>1)  
                        rPlayer.CreateUnitEx(x-1,y,  
0,null,"EDUHT3","EDWHC2",null,EDWSR3,0);  
                    if (rPlayer.GetRace() == raceLC)  
                    {  
                        rPlayer.CreateUnitEx(x,y, 0,null,"LCUCR1","LCWMR3",null,null,2);  
                        if(comboStartingUnits>0)  
                            rPlayer.CreateUnitEx(x+1,y,  
y,0,null,"LCUCR1","LCWHL1",null,"LCWSL1",null,2);  
                        if(comboStartingUnits>1)  
                            rPlayer.CreateUnitEx(x-1,y,0,null,"LCUCR1","LCWHS2",null,null,2);  
                    }  
                    if (rPlayer.GetRace() == raceUCS)  
                    {  
                        rPlayer.CreateUnitEx(x,y,  
0,null,"UCSUH3","UCSWBMR3",null,null,0);  
                        if(comboStartingUnits>0)  
                            rPlayer.CreateUnitEx(x+1,y,0,null,"UCSUH3","UCSWBSR3",null,2);  
                    }  
                }  
                if(comboTechLevel==2)  
                {  
                    if (rPlayer.GetRace() == raceED)  
                    {  
                        rPlayer.CreateUnitEx(x,y,  
0,null,"EDUHW2","EDWHL3",null,"EDWSL3",null,2);  
                        if(comboStartingUnits>0)  
                            rPlayer.CreateUnitEx(x+1,y,  
0,null,"EDUBT2","EDWMR3","EDWMR3",null,null,1);  
                        if(comboStartingUnits>1)  
                            rPlayer.CreateUnitEx(x-1,y,  
0,null,"EDUHT3","EDWHC2",null,EDWSR3,0);  
                        if (rPlayer.GetRace() == raceLC)  
                        {  
                            rPlayer.CreateUnitEx(x,y,0,null,"LCUCR3","LCWHL2",null,"LCWSL2",null,2);  
                            if(comboStartingUnits>0)  
                                rPlayer.CreateUnitEx(x+1,y,  
0,null,"LCUCU3","LCWMR3",null,null,0);  
                            if(comboStartingUnits>1)  
                                rPlayer.CreateUnitEx(x,y+1,y,  
1,y,0,null,"LCUCU3","LCWHL2","LCWHL2","LCWS2","LCWS2",1);  
                            if (rPlayer.GetRace() == raceUCS)  
                            {  
                                rPlayer.CreateUnitEx(x,y,0,null,"UCSUH3","UCSWBPH3",null,"UCSWSSP2",null,2);  
                                if(comboStartingUnits>0)  
                                    rPlayer.CreateUnitEx(x+1,y,0,null,"UCSUH3","UCSWBSR3",null,2);  
                                if(comboStartingUnits>1)  
                                    rPlayer.CreateUnitEx(x-1,y,0,null,"UCSUH3","UCSWBSR3",null,2);  
                            }  
                        }  
                    }  
                }  
            }  
        }  
    }
```



```

int i;
    int nPlayersCount;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    nPlayersCount=0;
for(i=0;i<15;i+=1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        nPlayersCount=nPlayersCount+1;
        rLastPlayer=rPlayer;
        if(!rPlayer.GetScriptData(1))
        {
            rPlayer2 = GetPlayer(nPlayerInConsole);
            if(rPlayer2==null)rPlayer2 = rPlayer;
        }
    }
}

rPlayer.LookAt(rPlayer2.GetStartingPointX(),rPlayer2.GetStartingPointY(),-1,-1,-1,0);
    CreateStartingUnits(rPlayer,rPlayer2);
}

if(nPlayersCount==1)
    rLastPlayer.Victory();

}

event Timer1()
{
    int i;
    player rPlayer;
    player rBestPlayer;
    int bestScore;

    bCheckBuilding=true;
    if(nTimeLimit)
    {
        if(GetMissionTime()>nTimeLimit)
        {
            for(i=0;i<15;i+=1)
            {
                rPlayer=GetPlayer(i);
                if(rPlayer==null)
                {
                    if(rPlayer.GetScriptData(0)>bestScore)
                        bestScore = rPlayer.GetScriptData(0);
                }
            }
            for(i=0;i<15;i+=1)
            {
                rPlayer=GetPlayer(i);
                if(rPlayer==null && rPlayer.IsAlive())
                {
                    if(rPlayer.GetScriptData(0)==bestScore)
                        rPlayer.Victory();
                    else
                        rPlayer.Defeat();
                }
            }
        }
    }
}

event UnitDestroyed(unit uUnit)
{
    unit uAttacker;
    player pAttacker;
    player pDestroyed;
    int tmp;

    pAttacker = GetPlayer(uUnit.GetIFFNumber());
    pAttacker.SetScriptData(1,pAttacker.GetScriptData(1)-1);
    uAttacker = uUnit.GetAttacker();
    if(uAttacker==null) return false;
    pAttacker = GetPlayer(uAttacker.GetIFFNumber());
    if(uAttacker.GetIFFNumber()==uUnit.GetIFFNumber())
        tmp = pAttacker.GetScriptData(0)-1;
    else
        tmp = pAttacker.GetScriptData(0)+1;
    pAttacker.SetScriptData(0,tmp);

    return true;
}

command Initialize()
{
    comboStartingUnits=1;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboTechLevel
{
    comboTechLevel = nMode;
}

command Combo2(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}

command Combo3(int nMode) button comboTime
{
    comboTime = nMode;
}

}

CTF_MP.ec
mission "translateGameTypeCTFMP"
{
    int nCashRate;
    int m_nD;

    enum comboCTFType
    {
        "translateScriptCTFTypeCapturedDies",
        "translateScriptCTFTypeCaptureAll",
        multi:
            "translateScriptCTFType"
    }

    enum comboCashType
    {
        "translateScriptMineForMoney",
        "translateScript2500CRmin",
        "translateScript5000CRmin",
        "translateScript10000CRmin",
        "translateScript20000CRmin",
        multi:
            "translateScriptGainingMoney"
    }

    enum comboResearchTime
    {
        "translateScriptNormalTime",
        "translateScript2xfaster",
        "translateScript4xfaster",
        "translateScript8xfaster",
        multi:
            "translateScriptResearchTime"
    }

    enum comboResearchLimit
    {
        "translateScriptAllResearches",
        "translateScriptNoBombs",
        "translateScriptNoMassDestructionWeapons",
        "translateScriptNoBombsAndMDW",
        multi:
            "translateScriptAvailableResearches"
    }

    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit10000CR",
        "translateGameMenuUnitsLimit20000CR",
        "translateGameMenuUnitsLimit30000CR",
        "translateGameMenuUnitsLimit50000CR",
        multi:
            "translateGameMenuUnitsLimit"
    }

    enum comboAlliedVictory
    {
        "translateGameMenuAlliedVictoryNo",
        "translateGameMenuAlliedVictoryYes",
    }
}

```

```

multi:
    "translateGameMenuAlliedVictory"
}

//sprawdza czy rPlayer ma flagę w swojej bazie
function int HaveFlagInBase(player rPlayer)
{
    unitex flagEx;

    flagEx = rPlayer.GetScriptUnit(0);
    if ((flagEx != null) && flagEx.IsLive() && !flagEx.IsTransported() &&
        (Distance(flagEx.GetLocationX()), flagEx.GetLocationY()),
        rPlayer.GetStartingPointX(), rPlayer.GetStartingPointY()) <= m_nD) &&
        (flagEx.GetLocationZ() == 0))
    {
        return true;
    }
    return false;
}

//sprawdza czy rPlayer i ew. jego przyjaciele mają zcapturowane wszystkie
//obce flagi
function int HaveAllFlagsCapturedByAllies(player rPlayer)
{
    int i;
    int j;
    player rAlly;
    player rEnemy;
    unitex flagExAlly;
    unitex flagExEnemy;
    int bCapturedEnemy;

    //najpierw sprawdzamy czy my i wszyscy nasi sprzymiezency mają flagi w
    bazach
    for (j=0;j<15;j=j+1)
    {
        rAlly = GetPlayer(j);
        if ((rAlly != null) && rAlly.IsAlive() && ((rAlly == rPlayer) ||
        rPlayer.IsAlly(rAlly)))
        {
            if (!HaveFlagInBase(rAlly))
            {
                return false;
            }
        }
    }
    for (i=0;i<15;i=i+1)
    {
        rEnemy = GetPlayer(i);
        if ((rEnemy != null) && (rEnemy != rPlayer) && !rPlayer.IsAlly(rEnemy))
        {
            flagExEnemy = rEnemy.GetScriptUnit(0);
            bCapturedEnemy = false;
            if ((flagExEnemy == null) || !flagExEnemy.IsLive() ||
            flagExEnemy.IsTransported())
            {
                return false;
            }
            for (j=0;j<15;j=j+1)
            {
                rAlly = GetPlayer(j);
                if ((rAlly != null) && rAlly.IsAlive() && ((rAlly == rPlayer) ||
                rPlayer.IsAlly(rAlly)))
                {
                    //HaveFlagInBase(rAlly) spelnone
                    flagExAlly = rAlly.GetScriptUnit(0);
                    if ((Distance(flagExEnemy.GetLocationX()),
                    flagExAlly.GetLocationY(), flagExAlly.GetLocationZ(),
                    flagExAlly.GetLocationZ()) <= m_nD) &&
                    (flagExEnemy.GetLocationZ() == flagExAlly.GetLocationZ()))
                    {
                        bCapturedEnemy = true;
                    }
                }
            }
            if (!bCapturedEnemy)
            {
                return false;
            }
        }
    }
    return true;
}

state Initialize;
state DelayedCommands;

```

```

state Nothing;
state Initialize
{
    player rPlayer;
    int i;
    int nPlayers;
    unitex flagEx;

    if(comboCashType==0) nCashRate = 0;
    if(comboCashType==1) nCashRate = 2500;
    if(comboCashType==2) nCashRate = 5000;
    if(comboCashType==3) nCashRate = 10000;
    if(comboCashType==4) nCashRate = 20000;

    if(comboResearchTime==1) SetTimeDivider(2);
    if(comboResearchTime==2) SetTimeDivider(4);
    if(comboResearchTime==3) SetTimeDivider(8);

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if (rPlayer!=null)
        {
            if(rPlayer.GetRace()==raceUCS)
            {
                rPlayer.EnableBuilding("UCSBLZ", false);
                rPlayer.EnableBuilding("UCSBTB", false);
            }
            if(rPlayer.GetRace()==raceED)
            {
                rPlayer.EnableBuilding("EDBLZ", false);
                rPlayer.EnableBuilding("EDBTC", false);
            }
            if(rPlayer.GetRace()==raceLC)
            {
                rPlayer.EnableBuilding("LCBLZ", false);
                rPlayer.EnableBuilding("LCBSR", false);
            }
        }
    }

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            rPlayer.SetMaxDistance(30);
            rPlayer.EnableCommand(commandSoldBuilding,true);
            rPlayer.AddResearch("RES_MISSION_PACK1_ONLY");

            if(comboCashType>0)
            {
                rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMiningUnits,false);

                rPlayer.EnableBuilding("EDBML",false);
                rPlayer.EnableBuilding("EDBRE",false);
                rPlayer.EnableBuilding("UCSBR",false);
                rPlayer.EnableBuilding("LCBM",false);

                rPlayer.EnableResearch("RES_UCS_UOH2",false);
                rPlayer.EnableResearch("RES_UCS_UOH3",false);
                rPlayer.EnableResearch("RES_UCS_UAH1",false);
                rPlayer.EnableResearch("RES_UCS_UAH2",false);
                rPlayer.EnableResearch("RES_UCS_UAH3",false);
            }
        }
    }

    if(comboResearchLimit==1 || comboResearchLimit==3)
    {
        //no bombs
        rPlayer.EnableResearch("RES_ED_AB1",false);
        rPlayer.EnableResearch("RES_ED_MB2",false);
        rPlayer.EnableResearch("RES_UCS_WAPB1",false);
        rPlayer.EnableResearch("RES_UCS_MB2",false);
        //Moon Project
        rPlayer.EnableResearch("RES_UCS_ART",false);
        rPlayer.EnableResearch("RES_ED_ART",false);
        rPlayer.EnableResearch("RES_LC_ART",false);
    }
    if(comboResearchLimit==2 || comboResearchLimit==3)
    {
        //no UW
    }
}

```

```

rPlayer.EnableResearch("RES_ED_WHR1",false);
rPlayer.EnableResearch("RES_LC_BWC",false);
rPlayer.EnableResearch("RES_LC_SDIEF",false);
rPlayer.EnableResearch("RES_UCS_PC",false);
rPlayer.EnableResearch("RES_UCS_WSD",false);
//Moon project
rPlayer.EnableResearch("RES_UCS_USM1");
rPlayer.EnableResearch("RES_UCS_USM2",false);
rPlayer.EnableResearch("RES_ED_USM1",false);
rPlayer.EnableResearch("RES_ED_USM2",false);
}

if(comboAlliedVictory)
    rPlayer.EnableAIFeatures2(ai2BNSSendResult,false);//nie wysylac
resultatow do EARTH NETU

if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

rPlayer.SetMoney(30000);

rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);
if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
    rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX()+1,rPlayer.GetStartingPointY()
+1,0);

    if(rPlayer.GetRace()==1)
        flagEx =
rPlayer.CreateUnitEx(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),
0,"AI\(\CivilEmpty\)" "UCSUCTF",null,null,null);
        if(rPlayer.GetRace()==2)
            flagEx =
rPlayer.CreateUnitEx(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),
0,"AI\(\CivilEmpty\)" "EDUCTF",null,null,null);
            if(rPlayer.GetRace()==3)
                flagEx =
rPlayer.CreateUnitEx(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),
0,"AI\(\CivilEmpty\)" "LUCUTE",null,null,null);
                rPlayer.SetScriptUnit(0, flagEx, true);
                rPlayer.AddUnitIfSpecialTab(flagEx, true, 0);
                rPlayer.SetScriptData(0, 1);/jeśli 1 to flaga zmartwychwstaje niezależnie od tego czy player zjedzie
                //zaznaczenie miejsca gdzie lezy flaga

//CreateArtefact("NEASPECIAL2",rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0,0,artefactSpecialAIOther,0);

CreateArtefact("NEASPECIAL3",rPlayer.GetStartingPointX()+1,rPlayer.GetStartingPointY(),0,0,artefactSpecialAIOther,1);
    CreateArtefact("NEASPECIAL3",rPlayer.GetStartingPointX()-1,rPlayer.GetStartingPointY(),0,0,artefactSpecialAIOther,1);

CreateArtefact("NEASPECIAL3",rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY()+1,0,0,artefactSpecialAIOther,1);

CreateArtefact("NEASPECIAL3",rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY()-1,0,0,artefactSpecialAIOther,1);

    }

    //polecenie ilu jest playerow (potrzebne do obliczenia odleglosci flag)
nPlayers = 0;
for(i=0;i<15;i=i+1)
{
    if (GetPlayer(i))
    {
        nPlayers = nPlayers + 1;
    }
}
if (nPlayers <= 9) m_nD = 1;
else m_nD = 3;
SetTimer(0,300);
SetTimer(1,1200);
SetTimer(2,1);
SetTimer(3,200);
SetTimer(4,200);
SetTimer(5,200);

return DelayedCommands20;
}
state DelayedCommands
{



int i;
player rPlayer;
for(i=0;i<15;i=i+1)
{
    rPlayer=GetPlayer(i);
    if(rPlayer!=null)
    {
        rPlayer.EnableAIFeatures2(ai2CaptureSpecialTarget,true);
    }
}
return Nothing;
}
state Nothing
{
    return Nothing;
}
event RemoveResources()
{
    if(comboCashType==0)
        false;
    else
        true;
}
event RemoveUnits()
{
    true;
}
event Timer0()
{
    int i;
    int j;
    int iCountBuilding;
    player rPlayer;
    player rAlly;
    int bHaveAlly;
    for(i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
            if (iCountBuilding<=1)//1 bo flaga jest unitem
            {
                //jesli to jest tryb 1 (capture all to win) i ten gracz ma jakich przyjaciel to nie daje majecie
                ///po to zeby ktos mogl mu przekazac buildera lub unity)
                bHaveAlly = false;
                if (comboCTFType == 1)
                {
                    for (j=0;j<15;j=j+1)
                    {
                        rAlly = GetPlayer(j);
                        if ((rAlly != null) && (rAlly != rPlayer) && rAlly.IsAlive() &&
rPlayer.IsAlly(rAlly))
                        {
                            bHaveAlly = true;
                        }
                    }
                }
                if (!bHaveAlly)
                {
                    rPlayer.Defeat();
                }
                //nie ma SetScriptData
            }
        }
    }
}
event Timer1()
{
    int i;
    player rPlayer;
    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
}
}
}

```

```

        if(rPlayer.GetMoney()<100000)
            rPlayer.AddMoney(nCashRate);
    }

    event Timer2()
    {
        int i;
        player rPlayer;
        unitex flagEx;
        int flagX;
        int flagY;
        int flagZ;

        //sprawdzenie zabicia flag - ew. utworzenie ich od nowa w ostatnim miejscu
        ich pobytu
        for (i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if ((rPlayer!=null) && (rPlayer.GetScriptData(0) == 1))
            {
                flagEx = rPlayer.GetScriptUnit(0);
                if ((flagEx == null) || !flagEx.IsLive())
                {
                    if (flagEx == null)
                    {
                        flagX = rPlayer.GetScriptData(5);
                        flagY = rPlayer.GetScriptData(6);
                        flagZ = rPlayer.GetScriptData(7);
                    }
                    else
                    {
                        flagX = flagEx.GetLocationX();
                        flagY = flagEx.GetLocationY();
                        flagZ = flagEx.GetLocationZ();
                    }
                    rPlayer.AddUnitToSpecialTab(flagEx, false, -1); //wyrzucenie z
                }
            }
            //tworzymy nowa flagę
            if(rPlayer.GetRace() == 1)
                flagEx = rPlayer.CreateUnitEx(flagX, flagY,
flagZ, "AI\\CivilEmpty", "UCSUCTF", null,null,null,null);
            if(rPlayer.GetRace() == 2)
                flagEx = rPlayer.CreateUnitEx(flagX, flagY,
flagZ, "AI\\CivilEmpty", "EUDUCTF", null,null,null,null);
            if(rPlayer.GetRace() == 3)
                flagEx = rPlayer.CreateUnitEx(flagX, flagY,
flagZ, "AI\\CivilEmpty", "LCUCTF", null,null,null,null);
            rPlayer.SetScriptUnit(0, flagEx, true);
            rPlayer.AddUnitToSpecialTab(flagEx, true, 0);
        }
        else
        {
            flagEx.RegenerateHP();
            flagEx.RegenerateElectronics();
            rPlayer.SetScriptData(5, flagEx.GetLocationX());
            rPlayer.SetScriptData(6, flagEx.GetLocationY());
            rPlayer.SetScriptData(7, flagEx.GetLocationZ());
        }
    }

    event Timer3()
    {
        int i;
        int j;
        int k;
        player rPlayer;
        player rEnemy;
        unitex flagEx;
        unitex flagExEnemy;
        int iAlivePlayers;
        int iCountBuilding;
        int bActiveEnemies;
        int bOneHasBeenCalled;
        player rPlayer2;
        player rLastPlayer;
        int blsVictoryAlliance;
        int bHaveSomeEnemy;

        if (comboCTFType == 0)
        //captured flag defeats
        //sprawdzenie czy jakas flaga stoi w poblizu wrociej flagi ktora stoi w
        poblizu swojego punktu startowego
        for (i=0; i<15; i=i+1)
        {
            rPlayer = GetPlayer(i);
            //nie sprawdzamy IsLive bo ktos moze nie byc zywy ale jego flaga nie
            zostala jeszcze zkaputowana
            if ((rPlayer!=null && rPlayer.IsAlive()))
            {
                flagEx = rPlayer.GetScriptUnit(0);
                if ((flagEx != null) && flagEx.IsTransported())
                {
                    for (j=0; j<15; j=j+1)
                    {
                        rEnemy = GetPlayer(j);
                        if ((j!=i) && (rEnemy!=null) && rEnemy.IsAlive() &&
                        !rPlayer.IsAlly(rEnemy))
                        {
                            flagExEnemy = rEnemy.GetScriptUnit(0);
                            if ((flagExEnemy != null) &&
                            !flagExEnemy.IsTransported())
                            {
                                if (HaveFlagInBase(rEnemy) &&
                                (Distance(flagEx.GetLocationX(),
flagEx.GetLocationY(), flagExEnemy.GetLocationX(),
flagExEnemy.GetLocationY()) <= m_nD) &&
                                flagExEnemy.GetLocationZ() ==
flagEx.GetLocationZ()))
                                {
                                    //captured flag
                                    rPlayer.SetScriptData(0, 0);
                                    rPlayer.Defeat();
                                }
                            }
                        }
                    }
                }
            }
        }
        KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,200); //tym samym zabijamy
        te flagi
        KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,1,200);
        break;
    }
}

if(comboAlliedVictory)
{
    bOneHasBeenDestroyed=false;
    for(i=0;i<15;j=i+1)
    {
        rPlayer = GetPlayer(i);
        if((rPlayer!=null && !rPlayer.IsAlive()))
        bOneHasBeenDestroyed=true;
    }
    bActiveEnemies=false;
    for(i=0;i<15;j=i+1)
    {
        rPlayer = GetPlayer(i);
        if((rPlayer!=null && rPlayer.IsAlive()))
        {
            for(j=0;j<15;j=j+1)
            {
                rPlayer2 = GetPlayer(j);
                if((rPlayer2!=null && rPlayer2.IsAlive()) &&
                !rPlayer.IsAlly(rPlayer2))
                {
                    bActiveEnemies=true;
                }
            }
        }
    }
    if(bActiveEnemies) return;
    if(!bOneHasBeenDestroyed) return;
    //sprawdzamy czy nie ma flag "niezywych" playerow ktore jeszcze nie
    zostaly zkaputowane
    //oraz czy zyjacy maja swoje flagi w bazie
    for (i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null)
        {
            flagEx = rPlayer.GetScriptUnit(0);
            if (!rPlayer.IsAlive())
            {
                if ((flagEx != null) && flagEx.IsLive())
                {
                    return;
                }
            }
        }
    }
}

```

```

        }
    }
}

for(i=0;i<15;j=i+1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        rPlayer.Victory();
    }
}
else
{
    for(i=0;i<15;j=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberofUnits();
            if(iCountBuilding>0)
            {
                iAlivePlayers=iAlivePlayers+1;
                rLastPlayer=rPlayer;
            }
        }
    }
    if(iAlivePlayers==1 && rLastPlayer!=null)
    {
        //sprawdzamy czy nie ma flag "niezywych" playerow ktore jeszcze
nie zostały zkaputowane
        for (i=0;i<15;j=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && !rPlayer.IsAlive())
            {
                flagEx = rPlayer.GetScriptUnit(0);
                if ((flagEx != null) && flagEx.IsLive())
                {
                    return;
                }
            }
            //sprawdzenie czy rLastPlayer ma flagę w bazie
            if (!HaveFlagInBase(rLastPlayer))
            {
                return;
            }
            rLastPlayer.Victory();
        }
    }
    else
    {
        if (comboAlliedVictory)
        {
            //player (i jego alianci) wygrywa gdy on (lub jego alianci) mają swoje
flagi w bazach
            //oraz wszystkie niealianskie flagi obok nich
            //oraz ci alianci powinni mieć przynajmniej jednego wroga
            for(i=0;i<15;j=i+1)
            {
                rPlayer = GetPlayer(i);
                bIsVictoryAlliance = true;
                bHaveSomeEnemy = false;
                if ((rPlayer != null) && rPlayer.IsAlive())
                {
                    //sprawdzenie czy dla niego i jego playerow sa spełnione
warunki zwycięstwa
                    for (j=0;j<15;j=j+1)
                    {
                        rPlayer2=GetPlayer(j);
                        if ((j==i) || (rPlayer2!=null) && rPlayer2.IsAlive() &&
rPlayer.IsAlly(rPlayer2))
                        {
                            if (!HaveAllFlagsCapturedByAllies(rPlayer2))
                            {

```

```

        for (j=0;j<15;j=j+1)
        {
            rPlayer2 = GetPlayer(j);
            if ((j!=i) && (rPlayer2!=null)) //nie sprawdzamy IsAlive bo
potrzebujemy wszystkich flag
            {
                flagExEnemy = rPlayer2.GetScriptUnit(0);
                if ((flagExEnemy == null) || !flagExEnemy.IsLive() || !
flagExEnemy.IsTransported() ||
                (Distance(flagExEnemy.GetLocationX(),
flagExEnemy.GetLocationY(), flagEx.GetLocationX(), flagEx.GetLocationY()) >
m_nD) ||
                (flagExEnemy.GetLocationZ())!=
flagEx.GetLocationZ()))
                {
                    return;
                }
            }
        //rPlayer ma wszystkie flagi - zwyciezyl
        rLastPlayer = rPlayer;
    }
}
if (rLastPlayer!=null)
{
    for(i=0;j<15;j=i+1)
    {
        rPlayer = GetPlayer(i);
        if((rPlayer!=null) && (rPlayer!=rLastPlayer))
        {
            rPlayer.SetScriptData(0, 0);
            rPlayer.Defeat();
        }
    }
}
KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,200); //tym samym zabijamy
tez flagi
    KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,1,200);
}
rLastPlayer.Victory();
}
}
event Timer4()
{
    int i;
    player rPlayer;
    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMaxDistance()<255)
                rPlayer.SetMaxDistance(rPlayer.GetMaxDistance()+1);
        }
    }
}
event Timer5()
{
    int i;
    int j;
    player rPlayer;
    player rPlayer2;
    //sprawdzanie zanoblowania szpiegowania
    //po 10 minutach - warunek player ma 2 lub mniej budynki inne niz wie-
zycki
    //i 4 lub mniej wiezyczki i 6 lub mniej unitow)
    //jesli warunek nie jest spełniony to z powrotem disableujemy szpiegowanie
    if (GetMissionTime() > 10*60*20)
    {
        for (i = 0; i < 15; i = i + 1)
        {
            rPlayer = GetPlayer(i);
            if ((rPlayer != null) && rPlayer.IsAlive())
            {
                if ((rPlayer.GetNumberOfBuildings(buildingNormal) <= 4) &&
((rPlayer.GetNumberOfBuildings() -
rPlayer.GetNumberOfBuildings(buildingNormal)) <= 2) &&
                (rPlayer.GetNumberOfUnits() <= 6))
                {
                    for (j = 0; j < 15; j = j + 1)
                    {
                        rPlayer2 = GetPlayer(j);
                        if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive() &&
rPlayer2.IsAlly(rPlayer))
                        {
                            if
(rPlayer2.CommandDisabled(commandBuildingSpyPlayer0 + i))
                                rPlayer2.EnableCommand(commandBuildingSpyPlayer0 +
i, true);
                        }
                    }
                }
            }
        }
    }
}
command Initialize()
{
    comboCTFType=0;
    comboResearchTime=0;
    comboCashType=1;
    comboAlliedVictory=1;
    comboUnitsLimit=2;
}
command Uninitialize()
{
    ResourcesPerContainer(8);
}
command Combo1(int nMode) button comboCTFType
{
    comboCTFType = nMode;
}
command Combo2(int nMode) button comboCashType
{
    comboCashType = nMode;
}
command Combo3(int nMode) button comboResearchTime
{
    comboResearchTime = nMode;
}
command Combo4(int nMode) button comboResearchLimit
{
    comboResearchLimit = nMode;
}
command Combo5(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}
command Combo6(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}
}



## DestroyEnemyStructuresMP.ec


mission "translateGameTypeDestroyStructuresMP"
{
    int nTimeLimit;
    int nCashRate;
    int nBuildingType;

    enum comboCashType
    {
        "translateScriptMineForMoney",
        "translateScript2500Rmin",
        "translateScript5000Rmin",
    }
}

```

```

        "translateScript10000CRmin",
        "translateScript20000CRmin",
        multi:
        "translateScriptGainingMoney"
    }
    enum comboStartingUnits
    {
        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
        multi:
        "translateGameMenuStartingUnits"
    }/
    enum comboAlliedVictory
    {
        "translateGameMenuAlliedVictoryNo",
        "translateGameMenuAlliedVictoryYes",
        multi:
        "translateGameMenuAlliedVictory"
    }
    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit10000CR",
        "translateGameMenuUnitsLimit20000CR",
        "translateGameMenuUnitsLimit30000CR",
        "translateGameMenuUnitsLimit50000CR",
        multi:
        "translateGameMenuUnitsLimit"
    }
    enum comboWinCondition
    {
        "translateScriptDestroyAllStructures",
        "translateScriptDestroyMainStructures",
        "translateScriptDestroyPowerPlants",
        "translateScriptDestroyFactories",
        multi:
        "translateScriptVictoryCondition"
    }
    enum comboResearchTime
    {
        "translateScriptNormalTime",
        "translateScript2xFaster",
        "translateScript4xFaster",
        "translateScript8xFaster",
        multi:
        "translateScriptResearchTime"
    }
    enum comboResearchLimit
    {
        "translateScriptAllResearches",
        "translateScriptNoBombs",
        "translateScriptNoMassDestructionWeapons",
        "translateScriptNoBombsAndMDW",
        multi:
        "translateScriptAvailableResearches"
    }
    state Initialize;
    state Nothing;

    state Initialize
    {
        player rPlayer;
        int i;
        int nStartingMoney;

        nStartingMoney = 25000;

        if(comboCashType==0) nCashRate = 0;
        if(comboCashType==1) nCashRate = 2500;
        if(comboCashType==2) nCashRate = 5000;
        if(comboCashType==3) nCashRate = 10000;
        if(comboCashType==4) nCashRate = 20000;

        if(comboResearchTime==1) SetTimeDivider(2);
        if(comboResearchTime==2) SetTimeDivider(4);
        if(comboResearchTime==3) SetTimeDivider(8);

        ResourcesPerContainer(2);

        for(i=0;i<15;i=i+1)
        {
            rPlayer=GetPlayer(i);
            if (rPlayer!=null)
            {
                if(rPlayer.GetRace()==raceUCS)

```

```

                {
                    rPlayer.EnableBuilding("UCSBLZ", false);
                    rPlayer.EnableBuilding("UCSBTB", false);
                }
                if(rPlayer.GetRace()==raceED)
                {
                    rPlayer.EnableBuilding("EDBLZ", false);
                    rPlayer.EnableBuilding("EDBTC", false);
                }
                if(rPlayer.GetRace()==raceLC)
                {
                    rPlayer.EnableBuilding("LCBLZ", false);
                    rPlayer.EnableBuilding("LCBSR", false);
                }
            }
        }

        for(i=0;i<15;i=i+1)
        {
            rPlayer=GetPlayer(i);
            if (rPlayer!=null)
            {
                rPlayer.SetMaxDistance(30);
                if(comboAlliedVictory)
                    rPlayer.EnableAIFeatures(ai2BNSSendResult,false); //nie wysylac
            }
        }
    }
    rezultatow do EARTH Netu

    rPlayer.SetScriptData(0,0);
    if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
    if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
    if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
    if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
    if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

    rPlayer.SetMoney(nStartingMoney);

    rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);
    if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())

    rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0);
    ;

    rPlayer.EnableCommand(commandSoldBuilding,true);

    rPlayer.AddResearch("RES_MISSION_PACK1_ONLY"); //Enable mission pack units
    and buildings

    if(comboCashType>0)
    {

        rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMiningUnits,false);

        rPlayer.EnableBuilding("EDBML",false);
        rPlayer.EnableBuilding("EDBRE",false);
        rPlayer.EnableBuilding("UCSRF",false);
        rPlayer.EnableBuilding("LCMR",false);

        rPlayer.EnableResearch("RES_UCS_UOH2",false);
        rPlayer.EnableResearch("RES_UCS_UOH3",false);
        rPlayer.EnableResearch("RES_LCS_UAH1",false);
        rPlayer.EnableResearch("RES_LCS_UAH2",false);
        rPlayer.EnableResearch("RES_LCS_UAH3",false);

        if(comboResearchLimit==1 || comboResearchLimit==3)
        {
            //no bombs
            rPlayer.EnableResearch("RES_ED_AB1",false);
            rPlayer.EnableResearch("RES_ED_MB2",false);
            rPlayer.EnableResearch("RES_UCS_WAPB1",false);
            rPlayer.EnableResearch("RES_UCS_MB2",false);
            //Moon Project
            rPlayer.EnableResearch("RES_UCS_ART",false);
            rPlayer.EnableResearch("RES_ED_ART",false);
            rPlayer.EnableResearch("RES_LC_ART",false);
        }
        if(comboResearchLimit==2 || comboResearchLimit==3)
        {
            //no UW
            rPlayer.EnableResearch("RES_ED_WHR1",false);
            rPlayer.EnableResearch("RES_LC_BWC",false);
            rPlayer.EnableResearch("RES_LC_SDIEF",false);
            rPlayer.EnableResearch("RES_UCS_PC",false);
            rPlayer.EnableResearch("RES_UCS_WSD",false);
            //Moon project
            rPlayer.EnableResearch("RES_UCS_USM1",false);

```

```

        rPlayer.EnableResearch("RES_UCS_USM2",false);
        rPlayer.EnableResearch("RES_ED_USM1",false);
        rPlayer.EnableResearch("RES_ED_USM2",false);
    }

}

SetTimer(0,100);
SetTimer(2,1200);
SetTimer(3,200);
SetTimer(1,400);
SetTimer(5,200);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    if(comboCashType==0)
        false;
    else
        true;
}

event RemoveUnits()
{
/* if(comboStartingUnits)
    true;
else
    false; */
true;
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenCalled;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    rLastPlayer=null;
    iAlivePlayers=0;
    bOneHasBeenCalled=false;
    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.IsAlive())
                if(comboWinCondition==0)
                    iCountBuilding = rPlayer.GetNumberOfBuildings();

                if(comboWinCondition==1)//main structures
                {
                    iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery) +
rPlayer.GetNumberOfBuildings(buildingSupplyCenter) +
rPlayer.GetNumberOfBuildings(buildingMine) +
rPlayer.GetNumberOfBuildings(buildingRefinery) +
rPlayer.GetNumberOfBuildings(buildingResearchCenter) +
rPlayer.GetNumberOfBuildings(buildingHeadquarter) +
rPlayer.GetNumberOfBuildings(buildingBBC) +
rPlayer.GetNumberOfBuildings(buildingPlasmaControl) +
rPlayer.GetNumberOfBuildings(buildingWeatherControl) +
rPlayer.GetNumberOfBuildings(buildingMinningRefinery) +
rPlayer.GetNumberOfBuildings(buildingBaseFactory) +
}
                if(comboWinCondition==2)//power plants
                    iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery);

                if(comboWinCondition==3)//factories
                {
                    iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingBase) +
rPlayer.GetNumberOfBuildings(buildingFactory) +
rPlayer.GetNumberOfBuildings(buildingWaterBase) +
rPlayer.GetNumberOfBuildings(buildingBaseFactory);
}
                if(iCountBuilding > 0)
                    rPlayer.SetScriptData(0,1);
                else
                {
                    if(rPlayer.GetScriptData(0)==1)
                        rPlayer.Defeat();
                }
            }
        }
    }
}

KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/-2,128);
else
{
    if((rPlayer.GetNumberOfUnits()==0) &&
(rPlayer.GetNumberOfBuildings() == 0))
        rPlayer.Defeat();
}
if(!rPlayer.IsAlive())
    bOneBeenCalled=true;
}

bActiveEnemies=false;
for(i=0;i<15;i+=1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        for(j=i+1;j<15;j+=1)
        {
            rPlayer2 = GetPlayer(j);
            if(rPlayer2!=null && rPlayer2.IsAlive() && (!rPlayer.IsAlly(rPlayer2) && comboAllyVictory))
            {
                bActiveEnemies=true;
            }
        }
    }
}
if(bActiveEnemies) return;
if(!bOneBeenCalled) return;

for(i=0;i<15;i+=1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        rPlayer.Victory();
    }
}

event Timer1()
{
    SetConsoleText("");
}

event Timer2()
{
    int i;
    player rPlayer;
    if(comboWinCondition==0)

```

```

SetConsoleText("translateScriptDestroyAllStructures");
if(comboWinCondition==1)
SetConsoleText("translateScriptDestroyMainStructures");
if(comboWinCondition==2)
SetConsoleText("translateScriptDestroyPowerPlants");
if(comboWinCondition==3)
SetConsoleText("translateScriptDestroyFactories");

for(i=0;i<15;i=i+1)
{
    rPlayer=GetPlayer(i);
    if(rPlayer==null && rPlayer.IsAlive())
    {
        if(rPlayer.GetMoney()<100000)
        rPlayer.AddMoney(nCashRate);
    }
}
event Timer3()
{
    int i;
    player rPlayer;
    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer==null)
        {
            if(rPlayer.GetMaxDistance()<255)
            rPlayer.SetMaxDistance(rPlayer.GetMaxDistance()+1);
        }
    }
}

event Timer5()
{
    int i;
    int j;
    player rPlayer;
    player rPlayer2;
    //sprawdzanie zenablowania szpiegowania
    ///[po 10 minutach - warunek player ma 2 lub mniej budynki inne niz wie-
zyczki
///i 4 lub mniej wiezyczki i 6 lub mniej unitow)
//jesli warunek nie jest spełniony to z powrotem disabluujemy szpiegowanie
if (GetMissionTime() > 10*60*20)
{
    for (i = 0; i < 15; i = i + 1)
    {
        rPlayer = GetPlayer(i);
        if ((rPlayer != null) && rPlayer.IsAlive())
        {
            if ((rPlayer.GetNumberOfBuildings(buildingNormal) <= 4) &&
                ((rPlayer.GetNumberOfBuildings() -
                (rPlayer.GetNumberOfBuildings(buildingNormal)) <= 2) &&
                (rPlayer.GetNumberOfUnits() <= 6))
            {
                for (j = 0; j < 15; j = j + 1)
                {
                    rPlayer2 = GetPlayer(j);
                    if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive() &&
                        !rPlayer2.IsAlly(rPlayer))
                    {
                        if
(rPlayer2.IsCommandDisabled(commandBuildingSpyPlayer0 + i))
                        rPlayer2.EnableCommand(commandBuildingSpyPlayer0 +
i, true);
                    }
                }
            }
        else
        {
            for (j = 0; j < 15; j = j + 1)
            {
                rPlayer2 = GetPlayer(j);
                if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive())
                {
                    if
(rPlayer2.IsCommandEnabled(commandBuildingSpyPlayer0 + i))
                    rPlayer2.EnableCommand(commandBuildingSpyPlayer0 +
i, false);
                }
            }
        }
    }
}
}

command Initialize()
{
    comboCashType=0;/mine for money
    //comboStartingUnits=1;
    //comboAlliedVictory=1;
    comboWinCondition=0;/destroy all structures
    comboUnitsLimit=2;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboCashType
{
    comboCashType = nMode;
}

command Combo2(int nMode) button comboWinCondition
{
    comboWinCondition = nMode;
}

command Combo3(int nMode) button comboResearchTime
{
    comboResearchTime = nMode;
}

command Combo4(int nMode) button comboResearchLimit
{
    comboResearchLimit = nMode;
}

command Combo5(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}

command Combo6(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}

/*
command Combo6(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}*/



}

DieHard4MP.ec
mission "translateGameTypeDieHard4"
{
    int bGameEnded;

    enum comboResearchLimit
    {
        "translateScriptAllResearches",
        "translateScriptNoBombs",
        "translateScriptNoMassDestructionWeapons",
        "translateScriptNoBombsAndMDW",
        multi:
        "translateScriptAvailableResearches"
    }

    enum comboWinCondition
    {
        "translateScriptDestroyEverything",
        "translateScriptDestroyAllStructures",
        "translateScriptDestroyMainStructures",
        "translateScriptDestroyPowerPlants",
        "translateScriptDestroyFactories",
        multi:
        "translateScriptVictoryCondition"
    }

    enum comboStartingUnits
    {
    }
}

```

```

        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
multi:
        "translateGameMenuStartingUnits"
    }

enum comboUnitsLimit
{
    "translateGameMenuUnitsLimitNoLimit",
    "translateGameMenuUnitsLimit20000CR",
    "translateGameMenuUnitsLimit50000CR",
    "100000 CR",
multi:
    "translateGameMenuUnitsLimit"
}

enum comboAlliedVictory
{
    "translateGameMenuAlliedVictoryNo",
    "translateGameMenuAlliedVictoryYes",
multi:
    "translateGameMenuAlliedVictory"
}

state Initialize;
state Nothing;

state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

    for(i=0;i<15;i+=1)
    {
        rPlayer=GetPlayer(i);
        if (rPlayer!=null)
        {
            if(rPlayer.GetRace()==raceUCS)
            {
                rPlayer.EnableBuilding("UCSBIZ", false);
                rPlayer.EnableBuilding("UCSBTB", false);
            }
            if(rPlayer.GetRace()==raceED)
            {
                rPlayer.EnableBuilding("EDBLZ", false);
                rPlayer.EnableBuilding("EDBTCE", false);
            }
            if(rPlayer.GetRace()==raceLC)
            {
                rPlayer.EnableBuilding("LCBLZ", false);
                rPlayer.EnableBuilding("LCBSR", false);
            }
        }
    }
    bGameEnded=false;
    for(i=0;i<15;i+=1)
    {
        rPlayer=GetPlayer(i);

        if(rPlayer!=null)
        {
            rPlayer.SetMaxDistance(30);
            if(comboAlliedVictory)
                rPlayer.EnableAIFeatures(2,ai2BNSendResult,false); //nie wysylac
            rezultatow do EARTH NETU

            rPlayer.SetMoney(100000);

            rPlayer.EnableAIFeatures(aiBuildMiningBuildings | aiBuildMiningUnits | aiControlMiningUnits, false);

            rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);

            if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
            if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(20000);
            if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(50000);
            if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(100000);

            if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
            ;
        }
        CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0)
    }
    rPlayer.EnableBuilding("EDBMR", false);
    rPlayer.EnableBuilding("EDBRE", false);
    rPlayer.EnableBuilding("UCSBR", false);
    rPlayer.EnableBuilding("LCSR", false);

    rPlayer.EnableResearch("RES_UCS_UOH2", false);
    rPlayer.EnableResearch("RES_UCS_UAH1", false);
    rPlayer.EnableResearch("RES_UCS_UAH2", false);
    rPlayer.EnableResearch("RES_UCS_UAH3", false);

    if(comboResearchLimit==1 || comboResearchLimit==3)
    {
        //no bombs
        rPlayer.EnableResearch("RES_ED_AB1", false);
        rPlayer.EnableResearch("RES_ED_MB2", false);
        rPlayer.EnableResearch("RES_UCS_WAPB1", false);
        rPlayer.EnableResearch("RES_UCS_MB2", false);
    }
    if(comboResearchLimit==2 || comboResearchLimit==3)
    {
        //no UW
        rPlayer.EnableResearch("RES_ED_WHR1", false);
        rPlayer.EnableResearch("RES_LC_BWC", false);
        rPlayer.EnableResearch("RES_LC_SDIDEF", false);
        rPlayer.EnableResearch("RES_UCS_PC", false);
        rPlayer.EnableResearch("RES_UCS_WSD", false);
    }
    //neutral stuff
    rPlayer.AddResearch("RES_MCH2");
    rPlayer.AddResearch("RES_MCH3");
    rPlayer.AddResearch("RES_MCH4");
    rPlayer.AddResearch("RES_MSR2");
    rPlayer.AddResearch("RES_MSR3");
    rPlayer.AddResearch("RES_MSR4");
    rPlayer.AddResearch("RES_MMRC2");
    rPlayer.AddResearch("RES_MMRC3");
    rPlayer.AddResearch("RES_MMRC4");

    if(rPlayer.GetRace()==1)//UCS
    {
        rPlayer.AddResearch("RES_UCS_USL2");
        rPlayer.AddResearch("RES_UCS_USL3");
        rPlayer.AddResearch("RES_UCS_UML1");
        rPlayer.AddResearch("RES_UCS_UML2");
        rPlayer.AddResearch("RES_UCS_UML3");
        rPlayer.AddResearch("RES_UCS_UHL1");
        rPlayer.AddResearch("RES_UCS_UHL2");
        rPlayer.AddResearch("RES_UCS_UHL3");
        rPlayer.AddResearch("RES_UCS_U4L1");
        rPlayer.AddResearch("RES_UCS_U4L2");
        rPlayer.AddResearch("RES_UCS_U4L3");
        rPlayer.AddResearch("RES_UCSUBL1");
        rPlayer.AddResearch("RES_UCSUBL2");
        rPlayer.AddResearch("RES_UCSM11");
        rPlayer.AddResearch("RES_UCSM12");
        rPlayer.AddResearch("RES_UCSS1");
        rPlayer.AddResearch("RES_UCSS2");
        rPlayer.AddResearch("RES_UCSB1");
        rPlayer.AddResearch("RES_UCSB2");
        rPlayer.AddResearch("RES_UCSB3");
        rPlayer.AddResearch("RES_UCSGAR1");
        rPlayer.AddResearch("RES_UCSGAR2");
        rPlayer.AddResearch("RES_UCSGAR3");
        rPlayer.AddResearch("RES_LCS_BOMBER21");
        rPlayer.AddResearch("RES_LCS_BOMBER22");
        rPlayer.AddResearch("RES_LCS_BOMBER31");
        rPlayer.AddResearch("RES_UCSBOMER32");
        rPlayer.AddResearch("RES_UCSBMD");
        rPlayer.AddResearch("RES_UCSBHD");
        rPlayer.AddResearch("RES_UCSWH2");
        rPlayer.AddResearch("RES_UCSWH3");
        rPlayer.AddResearch("RES_LCS_WSR1");
        rPlayer.AddResearch("RES_LCS_WSR2");
        rPlayer.AddResearch("RES_UCSWSR3");
        rPlayer.AddResearch("RES_UCSWSR1");
        rPlayer.AddResearch("RES_UCSWSR2");
        rPlayer.AddResearch("RES_LCS_WSG1");
        rPlayer.AddResearch("RES_LCS_WSG2");
        rPlayer.AddResearch("RES_LCS_WHG1");
        rPlayer.AddResearch("RES_UCSWHG2");
        rPlayer.AddResearch("RES_UCSWMR1");
        rPlayer.AddResearch("RES_UCSWMR2");
    }
}

```

```

rPlayer.AddResearch("RES_UCS_WMR3");
rPlayer.AddResearch("RES_UCS_WMR1");
rPlayer.AddResearch("RES_UCS_WMR2");
rPlayer.AddResearch("RES_UCS_WSP1");
rPlayer.AddResearch("RES_UCS_WSP2");
rPlayer.AddResearch("RES_UCS_WHP1");
rPlayer.AddResearch("RES_UCS_WHP2");
rPlayer.AddResearch("RES_UCS_WHP3");
if (comboResearchLimit!=2&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_UCS_WAPB1");
    rPlayer.AddResearch("RES_UCS_WAPB2");
}
if (comboResearchLimit!=2&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_UCS_WSD");
    rPlayer.AddResearch("RES_UCS_ReHand");
    rPlayer.AddResearch("RES_UCS_ReHand2");
    rPlayer.AddResearch("RES_UCS_SGen");
    rPlayer.AddResearch("RES_UCS_MGen");
    rPlayer.AddResearch("RES_UCS_HGen");
    rPlayer.AddResearch("RES_UCS_SHD");
    rPlayer.AddResearch("RES_UCS_SHD2");
    rPlayer.AddResearch("RES_UCS_SHD3");
    rPlayer.AddResearch("RES_UCS_SHD4");
    if (comboResearchLimit!=1&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_UCS_MB2");
        rPlayer.AddResearch("RES_UCS_MB3");
        rPlayer.AddResearch("RES_UCS_MB4");
    }
    rPlayer.AddResearch("RES_UCS_MG2");
    rPlayer.AddResearch("RES_UCS_MG3");
    rPlayer.AddResearch("RES_UCS_MG4");
}
if(rPlayer.GetRace()==2)//ed
{
    rPlayer.AddResearch("RES_ED_USTA2");
    rPlayer.AddResearch("RES_ED_UST3");
    rPlayer.AddResearch("RES_ED_UHT1");
    rPlayer.AddResearch("RES_ED_UHT2");
    rPlayer.AddResearch("RES_ED_UHT3");
    rPlayer.AddResearch("RES_ED_UTB1");
    rPlayer.AddResearch("RES_ED_UTB2");
    rPlayer.AddResearch("RES_ED_UOH2");
    rPlayer.AddResearch("RES_ED_UTM1");
    rPlayer.AddResearch("RES_ED_UTM2");
    rPlayer.AddResearch("RES_ED_UTM3");
    rPlayer.AddResearch("RES_ED_UIM1");
    rPlayer.AddResearch("RES_ED_UIM2");
    rPlayer.AddResearch("RES_ED_UMW1");
    rPlayer.AddResearch("RES_ED_UMW2");
    rPlayer.AddResearch("RES_ED_UMW3");
    rPlayer.AddResearch("RES_ED_UHW1");
    rPlayer.AddResearch("RES_ED_UHW2");
    rPlayer.AddResearch("RES_ED_USS2");
    rPlayer.AddResearch("RES_ED_USS3");
    rPlayer.AddResearch("RES_ED_UHS1");
    rPlayer.AddResearch("RES_ED_UHS2");
    rPlayer.AddResearch("RES_ED_UA11");
    rPlayer.AddResearch("RES_ED_UA12");
    rPlayer.AddResearch("RES_ED_UA21");
    rPlayer.AddResearch("RES_ED_UA22");
    rPlayer.AddResearch("RES_ED_UA41");
    rPlayer.AddResearch("RES_ED_UA42");
    rPlayer.AddResearch("RES_ED_UA31");
    rPlayer.AddResearch("RES_ED_UA32");
    rPlayer.AddResearch("RES_ED_BMD");
    rPlayer.AddResearch("RES_ED_BHD");
    rPlayer.AddResearch("RES_ED_WCH2");
    rPlayer.AddResearch("RES_ED_ACH2");
    rPlayer.AddResearch("RES_ED_WCA2");
    rPlayer.AddResearch("RES_ED_WHC1");
    rPlayer.AddResearch("RES_ED_WHC2");
    rPlayer.AddResearch("RES_ED_WSR1");
    rPlayer.AddResearch("RES_ED_WSR2");
    rPlayer.AddResearch("RES_ED_WSR3");
    rPlayer.AddResearch("RES_ED_ASR1");
    rPlayer.AddResearch("RES_ED_ASR2");
    rPlayer.AddResearch("RES_ED_WNR");
    rPlayer.AddResearch("RES_ED_WMR1");
    rPlayer.AddResearch("RES_ED_WMR2");
    rPlayer.AddResearch("RES_ED_WMR3");
    rPlayer.AddResearch("RES_ED_WLS1");
    rPlayer.AddResearch("RES_ED_WLS2");
    rPlayer.AddResearch("RES_ED_WLS3");
    rPlayer.AddResearch("RES_ED_WHL1");
    rPlayer.AddResearch("RES_ED_WHL2");
    rPlayer.AddResearch("RES_ED_WHL3");
    rPlayer.AddResearch("RES_ED_WSI1");
    rPlayer.AddResearch("RES_ED_WSI2");
    rPlayer.AddResearch("RES_ED_WHH1");
    rPlayer.AddResearch("RES_ED_WHH2");
    if (comboResearchLimit!=1&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_ED_AB1");
        rPlayer.AddResearch("RES_ED_AB2");
    }
    rPlayer.AddResearch("RES_ED_ReHand");
    rPlayer.AddResearch("RES_ED_ReHand2");
    rPlayer.AddResearch("RES_ED_SGen");
    rPlayer.AddResearch("RES_ED_MGen");
    rPlayer.AddResearch("RES_ED_HGen");
    rPlayer.AddResearch("RES_ED_SCR");
    rPlayer.AddResearch("RES_ED_SCR2");
    rPlayer.AddResearch("RES_ED_SCR3");
    rPlayer.AddResearch("RES_ED_MHR2");
    rPlayer.AddResearch("RES_ED_MHR3");
    rPlayer.AddResearch("RES_ED_MHR4");
}
if (comboResearchLimit!=1&&comboResearchLimit!=3)
{
    rPlayer.AddResearch("RES_ED_MB2");
    rPlayer.AddResearch("RES_ED_MB3");
    rPlayer.AddResearch("RES_ED_MB4");
}
}
if(rPlayer.GetRace()==3)//lc
{
    rPlayer.AddResearch("RES_LC_ULU2");
    rPlayer.AddResearch("RES_LC_ULU3");
    rPlayer.AddResearch("RES_LC_UMO2");
    rPlayer.AddResearch("RES_LC_LUMO3");
    rPlayer.AddResearch("RES_LC_UCR1");
    rPlayer.AddResearch("RES_LC_UCR2");
    rPlayer.AddResearch("RES_LC_UCR3");
    rPlayer.AddResearch("RES_LC_UCU1");
    rPlayer.AddResearch("RES_LC_UCU2");
    rPlayer.AddResearch("RES_LC_UCU3");
    rPlayer.AddResearch("RES_LC_LUME1");
    rPlayer.AddResearch("RES_LC_LUME2");
    rPlayer.AddResearch("RES_LC_UME3");
    rPlayer.AddResearch("RES_LC_UCB01");
    rPlayer.AddResearch("RES_LC_UCB02");
    rPlayer.AddResearch("RES_LC_BMD");
    rPlayer.AddResearch("RES_LC_BHD");
    if (comboResearchLimit!=2&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_LC_BWC");
        if (comboResearchLimit!=2&&comboResearchLimit!=3)
        {
            rPlayer.AddResearch("RES_LC_SDIEF");
            rPlayer.AddResearch("RES_LC_WCH2");
            rPlayer.AddResearch("RES_LC_ACH2");
            rPlayer.AddResearch("RES_LC_WSR2");
            rPlayer.AddResearch("RES_LC_WSR3");
            rPlayer.AddResearch("RES_LC_ASR1");
            rPlayer.AddResearch("RES_LC_ASR2");
            rPlayer.AddResearch("RES_LC_WMR1");
            rPlayer.AddResearch("RES_LC_WMR2");
            rPlayer.AddResearch("RES_LC_WMR3");
            rPlayer.AddResearch("RES_LC_AMR1");
            rPlayer.AddResearch("RES_LC_AMR2");
            rPlayer.AddResearch("RES_LC_WLS1");
            rPlayer.AddResearch("RES_LC_WSL2");
            rPlayer.AddResearch("RES_LC_WHL1");
            rPlayer.AddResearch("RES_LC_WHL2");
            rPlayer.AddResearch("RES_LC_WSS1");
            rPlayer.AddResearch("RES_LC_WSS2");
            rPlayer.AddResearch("RES_LC_WHS1");
            rPlayer.AddResearch("RES_LC_WHS2");
        }
    }
}

```

```

rPlayer.AddResearch("RES_LC_WAS1");
rPlayer.AddResearch("RES_LC_WAS2");
rPlayer.AddResearch("RES_LC_WARTILLERY");
rPlayer.AddResearch("RES_LC_Sgen");
rPlayer.AddResearch("RES_LC_Mgen");
rPlayer.AddResearch("RES_LC_Hgen");
rPlayer.AddResearch("RES_LC_SHR1");
rPlayer.AddResearch("RES_LC_SHR2");
rPlayer.AddResearch("RES_LC_SHR3");
rPlayer.AddResearch("RES_LC_REG1");
rPlayer.AddResearch("RES_LC_REG2");
rPlayer.AddResearch("RES_LC_REG3");
rPlayer.AddResearch("RES_LC_SOB1");
rPlayer.AddResearch("RES_LC_SOB2");
}

//MOON Project
rPlayer.EnableCommand(commandSoldBuilding,true);
rPlayer.AddResearch("RES_MISSION_PACK1_ONLY");

if(comboResearchLimit==1 || comboResearchLimit==3)
{
    //no bombs
    rPlayer.EnableResearch("RES_UCS_ART",false);
    rPlayer.EnableResearch("RES_ED_ART",false);
    rPlayer.EnableResearch("RES_LC_ART",false);
}

if(comboResearchLimit==2 || comboResearchLimit==3)
{
    //no MDM
    rPlayer.EnableResearch("RES_UCS_USM1",false);
    rPlayer.EnableResearch("RES_UCS_USM2",false);
    rPlayer.EnableResearch("RES_ED_USM1",false);
    rPlayer.EnableResearch("RES_ED_USM2",false);
}

if (rPlayer.GetRace() == raceUCS)
{
    rPlayer.AddResearch("RES_UCSUCS");
    rPlayer.AddResearch("RES_UCSUUT");
    rPlayer.AddResearch("RES_UCSBHT");
    rPlayer.AddResearch("RES_UCSWTSC1");
    rPlayer.AddResearch("RES_UCSWTSC2");
    rPlayer.AddResearch("RES_UCSWBHC1");
    rPlayer.AddResearch("RES_UCSWBHC2");
    rPlayer.AddResearch("RES_UCS_MSC2");
    rPlayer.AddResearch("RES_UCS_MSC3");
    rPlayer.AddResearch("RES_UCS_MSC4");
    rPlayer.AddResearch("RES_UCS_MHC2");
    rPlayer.AddResearch("RES_UCS_MHC3");
    rPlayer.AddResearch("RES_UCS_MHC4");
    if ((comboResearchLimit==1&&comboResearchLimit!=3)
        rPlayer.AddResearch("RES_UCS_ART");
    rPlayer.AddResearch("RES_UCSCAA1");
    rPlayer.AddResearch("RES_UCSCAA2");
    rPlayer.AddResearch("RES_UCSWAP1");
    rPlayer.AddResearch("RES_UCSWAP2");
    rPlayer.AddResearch("RES_UCSWAN1");
    rPlayer.AddResearch("RES_UCSWEQ1");
    rPlayer.AddResearch("RES_UCSWEQ2");
    rPlayer.AddResearch("RES_UCS_BC1");
    if ((comboResearchLimit==2&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_UCS_USM1");
        rPlayer.AddResearch("RES_UCS_USM2");
    }
}

if (rPlayer.GetRace() == raceED)
{
    if
}

(comboResearchLimit!=1&&comboResearchLimit!=3)
    rPlayer.AddResearch("RES_ED_ART");
    rPlayer.AddResearch("RES_EDBHT");
    rPlayer.AddResearch("RES_EDWA1");
    rPlayer.AddResearch("RES_EDWAN1");
    rPlayer.AddResearch("RES_EDWE01");
    rPlayer.AddResearch("RES_EDWE02");
    rPlayer.AddResearch("RES_ED_BC1");
        rPlayer.AddResearch("RES_EDUSTEALTH");
        rPlayer.AddResearch("RES_EDUUT");
    if ((comboResearchLimit==2&&comboResearchLimit!=3)
    {
        rPlayer.AddResearch("RES_ED_USM1");
        rPlayer.AddResearch("RES_ED_USM2");
    }
}

if (rPlayer.GetRace() == raceLC)
    {
        rPlayer.AddResearch("RES_LCBPP2");
        rPlayer.AddResearch("RES_LCUFG1");
        rPlayer.AddResearch("RES_LCUFG2");
        rPlayer.AddResearch("RES_LCUFG3");
        rPlayer.AddResearch("RES_LCNE");
        rPlayer.AddResearch("RES_LCNUH");
            rPlayer.AddResearch("RES_LCIUT");
        if ((comboResearchLimit==1&&comboResearchLimit!=3)
            rPlayer.AddResearch("RES_LC_ART");
        rPlayer.AddResearch("RES_LCOWAN1");
        rPlayer.AddResearch("RES_LCWEQ1");
        rPlayer.AddResearch("RES_LCWEQ2");
        rPlayer.AddResearch("RES_LCUSE1");
        rPlayer.AddResearch("RES_LCUSE2");
        rPlayer.AddResearch("RES_LCCA1");
        rPlayer.AddResearch("RES_LC_BCI");
    }
}

SetTimer(0,100);
SetTimer(1, 20);
SetTimer(2,1200);
SetTimer(3,200);
SetTimer(5,200);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    true;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenDestroyed;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;

    rLastPlayer=null;
    iAlivePlayers=0;
    bOneHasBeenDestroyed=false;
    for(i=0;i<15;i+=1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null)
            {
                if(rPlayer.IsAlive())
                    if((comboWinCondition==0)
                        iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits());
                    if(comboWinCondition==1)
                        iCountBuilding = rPlayer.GetNumberOfBuildings();

                    if(comboWinCondition==2)//main structures
                    {
                        iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +

```

```

rPlayer.GetNumberOfBuildings(buildingEnergyBattery) +
    rPlayer.GetNumberOfBuildings(buildingBase) +
    rPlayer.GetNumberOfBuildings(buildingFactory) +
    rPlayer.GetNumberOfBuildings(buildingWaterBase) +
    rPlayer.GetNumberOfBuildings(buildingSupplyCenter) +
        rPlayer.GetNumberOfBuildings(buildingMine) +
        rPlayer.GetNumberOfBuildings(buildingRefinery) +
        rPlayer.GetNumberOfBuildings(buildingResearchCenter) +
        rPlayer.GetNumberOfBuildings(buildingHeadquater) +
            rPlayer.GetNumberOfBuildings(buildingBBC) +
            rPlayer.GetNumberOfBuildings(buildingPlasmaControl) +
            rPlayer.GetNumberOfBuildings(buildingWeatherControl) +
            rPlayer.GetNumberOfBuildings(buildingMiningRefinery) +
                rPlayer.GetNumberOfBuildings(buildingBaseFactory);
            }
        if(comboWinCondition==3)//power plants
        iCountBuilding =
rPlayer.GetNumberOfBuildings(buildingPowerPlant) +
rPlayer.GetNumberOfBuildings(buildingSolarPower) +
rPlayer.GetNumberOfBuildings(buildingEnergyBattery);

if(comboWinCondition==4)//factories
{
    iCountBuilding =
        rPlayer.GetNumberOfBuildings(buildingBase) +
        rPlayer.GetNumberOfBuildings(buildingFactory) +
        rPlayer.GetNumberOfBuildings(buildingWaterBase) +
        rPlayer.GetNumberOfBuildings(buildingBaseFactory);
}

if(iCountBuilding) rPlayer.SetScriptData(0,1);

if (iCountBuilding==0)
{
    if(rPlayer.GetScriptData(0)==1)
    {
        rPlayer.Defeat();
    }
}

KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
}
else
{
    if((rPlayer.GetNumberOfUnits()==0) &
(rPlayer.GetNumberOfBuildings() == 0))
    rPlayer.Defeat();
}

if(!lrPlayer.IsAlive()) bOneHasBeenDestroyed=true;
}

bActiveEnemies=false;
for(i=0;i<15;i=i+1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        for(j=i+1;j<15;j=j+1)
        {
            rPlayer2 = GetPlayer(j);
            if((rPlayer2!=null && rPlayer2.IsAlive() && !(rPlayer.IsAlly(rPlayer2)&& comboAlliedVictory)))
            {
                bActiveEnemies=true;
            }
        }
    }
}

if(bActiveEnemies) return;
if(!bOneHasBeenDestroyed) return;

for(i=0;i<15;i=i+1)
{
    rPlayer = GetPlayer(i);
    if(rPlayer!=null && rPlayer.IsAlive())
    {
        rPlayer.Victory();
    }
}

event Timer1()
{
    int i;
    player rPlayer;

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if((rPlayer!=null)
        {
            if(rPlayer.GetMoney()<100000)
            rPlayer.AddMoney(100000 - rPlayer.GetMoney());
        }
    }
}

event Timer2()
{
    if(comboWinCondition==0)
        SetConsoleText("translateScriptDestroyEverything");
    if(comboWinCondition==1)
        SetConsoleText("translateScriptDestroyAllStructures");
    if(comboWinCondition==2)
        SetConsoleText("translateScriptDestroyMainStructures");
    if(comboWinCondition==3)
        SetConsoleText("translateScriptDestroyPowerPlants");
    if(comboWinCondition==4)
        SetConsoleText("translateScriptDestroyFactories");
}

event Timer3()
{
    int i;
    player rPlayer;

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMaxDistance()<255)
            rPlayer.SetMaxDistance(rPlayer.GetMaxDistance()+1);
        }
    }
}

event Timer5()
{
    int i;
    int j;
    player rPlayer;
    player rPlayer2;
    //sprawdzanie zenablowania szpiegowania
    //(po 10 minutach - warunek player ma 2 lub mniej budynki inne niz wie-
zyczki
    //i 4 lub mniej wiezyczki i 6 lub mniej unitow)
    //jesli warunek nie jest spełniony to z powrotem disableujemy szpiegowanie
    if ((GetMissionTime() > 10*60*20)
    {
        for (i = 0; i < 15; i = i + 1)
        {
            rPlayer = GetPlayer(i);
            if ((rPlayer != null) && rPlayer.IsAlive())
            {
                if ((rPlayer.GetNumberOfBuildings(buildingNormal) <= 4) &&
((rPlayer.GetNumberOfBuildings(buildingNormal)) -
rPlayer.GetNumberOfBuildings(buildingNormal)) <= 2) &&
(rPlayer.GetNumberOfUnits() <= 6))
                {
                    for (j = 0; j < 15; j = j + 1)
                    {
                        rPlayer2 = GetPlayer(j);
                        if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive() &&
lrPlayer.IsAlly(rPlayer2))
                        {
                            if
(rPlayer2.IsCommandDisabled(commandBuildingSpyPlayer0 + i))
                            rPlayer2.EnableCommand(commandBuildingSpyPlayer0
+ i, true);
                        }
                    }
                }
            }
        }
    }
}

```

```

        }
    }
}

for (j = 0; j < 15; j = j + 1)
{
    rPlayer2 = GetPlayer(j);
    if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive())
    {
        if
(rPlayer2.IsCommandEnabled(commandBuildingSpyPlayer0 + i))
            rPlayer2.EnableCommand(commandBuildingSpyPlayer0
+ i, false);
    }
}
}

command Initialize()
{
    comboResearchLimit=0;
    comboWinCondition=0;
    comboStartingUnits=1;
    comboAlliedVictory=1;
    comboUnitsLimit=2;
}

command Combo1(int nMode) button comboResearchLimit
{
    comboResearchLimit = nMode;
}
command Combo2(int nMode) button comboWinCondition
{
    comboWinCondition = nMode;
}
command Combo3(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}
command Combo4(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}
command Combo5(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}
}

state Initialize;
state Nothing;

state Initialize
{
    player rPlayer;
    int i;
    int nStartingMoney;

    if(comboMoney==0) nStartingMoney=20000;
    if(comboMoney==1) nStartingMoney=30000;
    if(comboMoney==2) nStartingMoney=40000;

    nMoneyToEarn=25000;
    if(comboMoneyToEarn==1) nMoneyToEarn=40000;
    if(comboMoneyToEarn==2) nMoneyToEarn=50000;
    if(comboMoneyToEarn==3) nMoneyToEarn=75000;
    if(comboMoneyToEarn==4) nMoneyToEarn=100000;

    if(comboTime==0)nTimeLimit=0;
    if(comboTime==1)nTimeLimit=15*60*20;
    if(comboTime==2)nTimeLimit=30*60*20;
    if(comboTime==3)nTimeLimit=45*60*20;
    if(comboTime==4)nTimeLimit=60*60*20;
    if(comboTime==5)nTimeLimit=90*60*20;

    if(comboResources==0)resourcesPerContainer(16);
    if(comboResources==1)resourcesPerContainer(8);
    if(comboResources==2)resourcesPerContainer(4);
    if(comboResources==3)resourcesPerContainer(2);

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if (rPlayer!=null)
        {
            if(rPlayer.GetRace() == raceUCS)
            {
                rPlayer.EnableBuilding("UCSBLZ", false);
                rPlayer.EnableBuilding("UCSCTB", false);
            }
            if(rPlayer.GetRace() == raceED)
            {
                rPlayer.EnableBuilding("EDBLZ", false);
                rPlayer.EnableBuilding("EDBTC", false);
            }
            if(rPlayer.GetRace() == raceLC)
            {
                rPlayer.EnableBuilding("LCLBLZ", false);
                rPlayer.EnableBuilding("LCLCSR", false);
            }
        }
    }
}

```

```

for(i=0;i<15;i=i+1)
{
    rPlayer=GetPlayer(i);
    if(rPlayer!=null)
    {
        rPlayer.SetMaxDistance(30);
        rPlayer.EnableCommand(commandSoldBuilding,true);
        rPlayer.AddResearch("RES_MISSION_PACK1_ONLY");

        rPlayer.SetAllowGiveMoney(false);
        rPlayer.SetMoney(nStartingMoney);

        rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6,0,20,0);

        if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
        if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
        if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
        if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
        if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

        if (!rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings())
    }
}
SetTimer(0,100);
SetTimer(1,1200);
SetTimer(4,200);
SetTimer(5,200);
return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    false;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int bOthersDefeat;
    int iCountBuilding;
    player rPlayer;
    player rLastPlayer;

    rLastPlayer=null;
    iAlivePlayers=0;
    bOthersDefeat=false;
    for(i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
            if ((iCountBuilding>0)
            {
                iAlivePlayers=iAlivePlayers+1;
                rLastPlayer=rPlayer;
                if(rPlayer.GetMoney()>=nMoneyToEarn)
                {
                    rPlayer.Victory();
                    bOthersDefeat=true;
                }
            }
            else
            {
                rPlayer.Defeat();
            }
        }
    }
}
}

if (iAlivePlayers==1 && rLastPlayer==null && rPlayer.IsAlive())
{
    rLastPlayer.Victory();
}

if(bOthersDefeat==true)
{
    for(i=0;i<15;i=i+1)
    {
        rPlayer = GetPlayer(i);
        if(rPlayer!=null && rPlayer.IsAlive())
        {
            if(rPlayer.GetMoney()<=nMoneyToEarn)
            {
                rPlayer.Defeat();
            }
        }
    }
}

KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
}

event Timer1()
{
    int i;
    int minLeft;
    player rPlayer;

    if(nTimeLimit)
    {
        minLeft=(nTimeLimit - GetMissionTime())/1200;

        if(minLeft<0)minLeft=0;
        SetConsoleText("translateScriptTimeLeft",minLeft);
        if(minLeft<1)
        {
            for(i=0;i<15;i=i+1)
            {
                rPlayer=GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    rPlayer.Defeat();
                }
            }
        }
        nTimeLimit=0;
    }
}

event Timer4()
{
    int i;
    player rPlayer;

    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMaxDistance()<255)
                rPlayer.SetMaxDistance(rPlayer.GetMaxDistance()+1);
        }
    }
}

event Timer5()
{
    int i;
    int j;
    player rPlayer;
    player rPlayer2;
    //sprawdzanie zenablowania szpiegowania
    //(po 10 minutach - warunek player ma 2 lub mniej budynki inne niz wie-
zyczki
    //i 4 lub mniej wiezyczki i 6 lub mniej unitow)
    //jesli warunek nie jest spełniony z powrotem disablujemy szpiegowanie
    if (GetMissionTime() > 10*60*20)
    {
        for (i = 0; i < 15; i = i + 1)
        {
            rPlayer = GetPlayer(i);
            if ((rPlayer != null) && rPlayer.IsAlive())
            {

```

```

        if ((rPlayer.GetNumberOfBuildings(buildingNormal) <= 4) &&
            ((rPlayer.GetNumberOfBuildings() -
            rPlayer.GetNumberOfBuildings(buildingNormal)) <= 2) &&
            (rPlayer.GetNumberOfUnits() <= 6))
        {
            for (j = 0; j < 15; j = j + 1)
            {
                rPlayer2 = GetPlayer(j);
                if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive() &&
                    !rPlayer2.IsAlly(rPlayer))
                {
                    if
                }
            }
        }
    }
    else
    {
        for (j = 0; j < 15; j = j + 1)
        {
            rPlayer2 = GetPlayer(j);
            if ((j != i) && (rPlayer2 != null) && rPlayer2.IsAlive())
            {
                if
            }
        }
    }
}
(rPlayer2.IsCommandEnabled(commandBuildingSpyPlayer0 + i))
rPlayer2.EnableCommand(commandBuildingSpyPlayer0
+ i, true);
}
}
}
}
}

command Initialize()
{
    nTimeLimit=0;
    comboResources=1;
    comboStartingUnits=1;
    comboUnitsLimit=2;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboMoney
{
    comboMoney=nMode;
    if(comboMoneyToEarn < comboMoney) comboMoneyToEarn =
    comboMoney;
}

command Combo2(int nMode) button comboMoneyToEarn
{
    comboMoneyToEarn=nMode;
    if(comboMoneyToEarn < comboMoney) comboMoney =
    comboMoneyToEarn;
}

command Combo3(int nMode) button comboTime
{
    comboTime = nMode;
}

command Combo4(int nMode) button comboResources
{
    comboResources=nMode;
}

command Combo5(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}

command Combo6(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}
}

```

HarvestAndKillIMP.ec

```

mission "translateGameTypeHarvestAndKillIMP"
{
    int nTimeLimit;
    int bCheckBuilding;
    int nMeteor;

    enum comboMoney
    {
        "translateScript20000CR",
        "translateScript30000CR",
        "translateScript40000CR",
        "translateScript50000CR",
        multi:
        "translateGameMenuStartingMoney"
    }

    enum comboTime
    {
        "translateGameMenuTimeLimitNoLimit",
        "translateGameMenuTimeLimit15min",
        "translateGameMenuTimeLimit30min",
        "translateGameMenuTimeLimit45min",
        "translateGameMenuTimeLimit1h",
        "translateGameMenuTimeLimit15h",
        multi:
        "translateGameMenuTimeLimit"
    }

    enum comboResources
    {
        "translateGameMenuResourcesLow",
        "translateGameMenuResourcesNormal",
        "translateGameMenuResourcesHigh",
        "translateGameMenuResourcesVeryHigh",
        multi:
        "translateGameMenuResources"
    }

    enum comboStartingUnits
    {
        "translateGameMenuStartingUnitsDefault",
        "translateGameMenuStartingUnitsBuilderOnly",
        multi:
        "translateGameMenuStartingUnits"
    }

    enum comboUnitsLimit
    {
        "translateGameMenuUnitsLimitNoLimit",
        "translateGameMenuUnitsLimit10000CR",
        "translateGameMenuUnitsLimit20000CR",
        "translateGameMenuUnitsLimit30000CR",
        "translateGameMenuUnitsLimit50000CR",
        multi:
        "translateGameMenuUnitsLimit"
    }

    enum comboAlliedVictory
    {
        "translateGameMenuAlliedVictoryNo",
        "translateGameMenuAlliedVictoryYes",
        multi:
        "translateGameMenuAlliedVictory"
    }

    state Initialize;
    state Nothing;

    state Initialize
    {
        player rPlayer;
        int i;
        int nStartingMoney;
        if(comboMoney==0)nStartingMoney = 20000;
        if(comboMoney==1)nStartingMoney = 30000;
        if(comboMoney==2)nStartingMoney = 40000;
        if(comboMoney==3)nStartingMoney = 50000;
        if(comboTime==0)nTimeLimit=0;
        if(comboTime==1)nTimeLimit=15*60*20;
        if(comboTime==2)nTimeLimit=30*60*20;
        if(comboTime==3)nTimeLimit=45*60*20;
        if(comboTime==4)nTimeLimit=60*60*20;
        if(comboTime==5)nTimeLimit=90*60*20;
        if(comboResources==0) ResourcesPerContainer(16);
        if(comboResources==1) ResourcesPerContainer(8);
    }
}

```

```

if(comboResources==2) ResourcesPerContainer(4);
if(comboResources==3) ResourcesPerContainer(2);

for(i=0;i<15;i=i+1)
{
    rPlayer=GetPlayer(i);
    if (rPlayer!=null)
    {
        //BYTE look here
        if((rPlayer.GetRight()-rPlayer.GetLeft())<128) //width of the world
            rPlayer.SetMaxDistance(20);
        else
            rPlayer.SetMaxDistance(30);

        if(rPlayer.GetRace()==raceUCS)
        {
            rPlayer.EnableBuilding("UCSBLZ", false);
            rPlayer.EnableBuilding("UCSBTB", false);
        }
        if(rPlayer.GetRace()==raceED)
        {
            rPlayer.EnableBuilding("EDBLZ", false);
            rPlayer.EnableBuilding("EDBT", false);
        }
        if(rPlayer.GetRace()==raceLC)
        {
            rPlayer.EnableBuilding("LCBLZ", false);
            rPlayer.EnableBuilding("LCBSR", false);
        }
        rPlayer.EnableCommand(commandSoldBuilding,true);
    }

    rPlayer.AddResearch("RES_MISSION_PACK1_ONLY");//Enable mission pack units
    and buildings
}

bCheckBuilding=false;
for(i=0;i<15;i=i+1)
{
    rPlayer=GetPlayer(i);

    if(rPlayer!=null)
    {
        if(comboAlliedVictory)
            rPlayer.EnableAIFeatures2(ai2BNSendResult,false);//nie
wysylac rezultatow do EARTH NETU

        if(comboUnitsLimit==0) rPlayer.EnableMilitaryUnitsLimit(false);
        if(comboUnitsLimit==1) rPlayer.SetMilitaryUnitsLimit(10000);
        if(comboUnitsLimit==2) rPlayer.SetMilitaryUnitsLimit(20000);
        if(comboUnitsLimit==3) rPlayer.SetMilitaryUnitsLimit(30000);
        if(comboUnitsLimit==4) rPlayer.SetMilitaryUnitsLimit(50000);

        rPlayer.SetMoney(nStartingMoney);

        rPlayer.LookAt(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),6.0,20,0);
        if ((rPlayer.GetNumberOfUnits() && !rPlayer.GetNumberOfBuildings()))
            rPlayer.CreateDefaultUnit(rPlayer.GetStartingPointX(),rPlayer.GetStartingPointY(),0);
    }

    SetTimer(0,100);
    SetTimer(3,200);
    SetTimer(1,1200);

    nMeteor=1;
    return Nothing;
}

state Nothing
{
    return Nothing;
}

event RemoveResources()
{
    false;
}

event RemoveUnits()
{
    if(comboStartingUnits)
        true;
    else
        false;
}

}
}

event Timer0()
{
    int iAlivePlayers;
    int i;
    int j;
    int iCountBuilding;
    int bActiveEnemies;
    int bOneHasBeenDestroyed;
    player rPlayer;
    player rPlayer2;
    player rLastPlayer;
    //---meteors
    //rPlayer = GetPlayer(1);
    //Meteor(rPlayer.GetStartingPointX());
    nMeteor=2,(Player.GetStartingPointY(),nMeteor);
    //nMeteor=(nMeteor%10)+1;
    //end of meteors

    if (bCheckBuilding==false)
        return 0;
    rLastPlayer=null;
    iAlivePlayers=0;
    if(comboAlliedVictory)//-----
    {
        bOneHasBeenDestroyed=false;
        for(i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if (iCountBuilding==0)rPlayer.Defeat();
            }
            if(rPlayer!=null && !rPlayer.IsAlive())
                bOneHasBeenDestroyed=true;
        }
        bActiveEnemies=false;
        for(i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                for(j=i+1;j<15;j=j+1)
                {
                    rPlayer2 = GetPlayer(j);
                    if(rPlayer2!=null && rPlayer2.IsAlive() &&
!rPlayer.IsAlly(rPlayer2))
                    {
                        bActiveEnemies=true;
                    }
                }
            }
            if(bActiveEnemies) return;
            if(bOneHasBeenDestroyed) return;

            for(i=0;i<15;i=i+1)
            {
                rPlayer = GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    rPlayer.Victory();
                }
            }
        }
    else//-----
    {
        for(i=0;i<15;i=i+1)
        {
            rPlayer = GetPlayer(i);
            if(rPlayer!=null && rPlayer.IsAlive())
            {
                iCountBuilding = rPlayer.GetNumberOfBuildings() +
rPlayer.GetNumberOfUnits();
                if (iCountBuilding>0)
                {
                    iAlivePlayers=iAlivePlayers+1;
                    rLastPlayer=rPlayer;
                }
                else
                {
                    rPlayer.Defeat();
                }
            }
        }
    }
}

```

```

        }
    }
    if (iAlivePlayers==1 && rLastPlayer!=null)
    {
        rLastPlayer.Victory();
    }
}

event Timer1()
{
    int i;
    int minLeft;
    player rPlayer;
    bCheckBuilding=true;
    if(nTimeLimit)
    {
        minLeft=(nTimeLimit - GetMissionTime())/1200;

        if(minLeft<0)minLeft=0;
        SetConsoleText("translateScriptTimeLeft",minLeft);
        if(minLeft<1)
        {
            for(i=0;i<15;i=i+1)
            {
                rPlayer=GetPlayer(i);
                if(rPlayer!=null && rPlayer.IsAlive())
                {
                    rPlayer.Defeat();

                    KillArea(rPlayer.GetIFF(),GetRight()/2,GetBottom()/2,0,128);
                }
            }
            nTimeLimit=0;
        }
    }
}
event Timer3()
{
    int i;
    player rPlayer;
    for(i=0;i<15;i=i+1)
    {
        rPlayer=GetPlayer(i);
        if(rPlayer!=null)
        {
            if(rPlayer.GetMaxDistance()<255)
                rPlayer.SetMaxDistance(rPlayer.GetMaxDistance()+1);
        }
    }
}

command Initialize()
{
    comboMoney=0;
    comboResources=1;
    comboStartingUnits=1;
    comboAlliedVictory=1;
    comboUnitsLimit=2;
}

command Uninitialize()
{
    ResourcesPerContainer(8);
}

command Combo1(int nMode) button comboMoney
{
    comboMoney = nMode;
}

command Combo2(int nMode) button comboTime
{
    comboTime = nMode;
}

command Combo3(int nMode) button comboResources
{
    comboResources = nMode;
}

command Combo4(int nMode) button comboStartingUnits
{
    comboStartingUnits = nMode;
}

command Combo5(int nMode) button comboUnitsLimit
{
    comboUnitsLimit = nMode;
}

command Combo6(int nMode) button comboAlliedVictory
{
    comboAlliedVictory = nMode;
}
}

```

Creating campaign step by step

Same as in Earth!



DZUXXEZ
THE KINGDOMS OF THE EARTH

Object identifiers

Object identifiers for scripts

Chasis and weapon: ED UCS LC

Buildings: ED UCS LC

Researches: ED UCS LC Neutral

Artifacts: Normal Special

Tables below contains two columns. Identifier in first column is special name representing some object which you can use as parameter for some functions described below. Name in second column is language-specific name for this object displayed in game. Names here are from English version of The Moon Project. Names in other language versions are the same or very similar. Numbers 1,2,3 at the end of identifiers are for upgrades of chasis/weapons/researches. In The Moon Project there is a special research "RES_MISSION_PACK1_ONLY" used as a preceding research for all "MP" researches. This research must be added at begin of game for each player to enable new "MP" researches.

Chasis and weapon identifiers

This object identifiers can be used to create units from scripts with function player.CreateUnit, or player.CreateUnitEx.

You can only put valid weapon on valid chasis, if you are not sure which weapon can be put on some chasis check it in "Construction Center".

Identifier Name

ED chasis

EDUBU1 Gruz

EDUFAKE1 Fake TT

EDUFAKE2 Fake ZT

EDUFAKE3 Fake HT

EDUST1 TT 100 Pamir

EDUST2 TT 110 Pamir

EDUST3 TT 120 Pamir

EDUMI1 ZT 200 Minelayer

EDUMI2 ZT 210 Minelayer

EDUOH1 ZK Taiga

EDUMT1 ZT 100 Siberia

EDUMT2 ZT 101 Siberia

EDUMT3 ZT 102 Siberia

EDUSPY1 NEO

EDUMW1 TK 100 Caspian

EDUMW2 TK 110 Caspian

EDUMW3 TK 111 Caspian

EDUHW1 TL 70 Volga

EDUHW2 TL 80 Volga

EDUHT1 HT 400 Kruszhev

EDUHT2 HT 500 Kruszhev

EDUHT3 HT 600 Kruszhev

EDUBT1 HT 800 Ural

EDUBT2 HT 900 Ural

EDUSS1 ESS 30 Irkutsk

EDUSS2 ESS 40 Irkutsk

EDUSS3 ESS 50 Irkutsk

EDUBS1 ESS 200 Leviathan

EDUBS2 ESS 300 Leviathan

EDUA11 MI 106 Cossack

EDUA12 MI 107 Cossack

EDUA21 MI 140 Grozny

EDUA22 MI 150 Grozny

EDUA31 MI 200 Han

EDUA32 MI 210 Han

EDUA41 MI 300 Thor

EDUA42 MI 310 Thor

EDUA51 MI 27 Boyar

EDUMM11 Truck

EDUMM21 Truck

EDUMM31 Heavy Tank

EDUMM41 Tank

EDOUT Unit Transporter [MP]

EDOUSEALTH STEALTH [MP]

EDUSPECIAL Ruslan [MP]

EDUSM1 Kiev [MP]

EDUSM2 Kiev II [MP]

EDUCTF CTF Flag [MP]

ED weapons

EDWMM21 150 mm Cannon (for EDUMM21)

EDWMM31 105 mm Cannon (for EDUMM31)

EDWMM41 105 mm Cannon (for EDUMM41)

EDWMM42 105 mm Cannon (for EDUMM41)

EDWCH1 20 mm Chaingun

EDWCH2 20 mm Double Chaingun

EDWCA1 105 mm Cannon

EDWC2 105 mm Double Cannon

EDWSR1AB Rocket Launcher (for Pamir)

EDWSR2AB Double Rocket Launcher

EDWSR1 Rocket Launcher

EDWSR2 Rocket Launcher upg.1

EDWSR3 Rocket Launcher upg.2

EDWSL1 Laser Cannon

EDWSL2 Laser Cannon upg.1

EDWSL3 Laser Cannon upg.2

EDWSI1 Ion Cannon

EDWSI2 Ion Cannon upg.1

EDWHC1 120 mm Cannon

EDWHC2 120 mm Double Cannon

EDWMR1 Heavy Rocket Launcher

EDWMR2 Heavy Rocket Launcher upg.1

EDWMR3 Heavy Rocket Launcher upg.2

EDWHShip1 120 mm Artillery

EDWHShip2 120 mm Double Artillery

EDWMRShip1 Long Range Rocket Launcher

EDWMRShip2 Long Range Rocket Launcher upg.1

EDWMRShip3 Long Range Rocket Launcher upg.2

EDWHR1 Ballistic Rocket Launcher

EDWHL1 Heavy Laser Cannon

EDWHL2 Heavy Laser Cannon upg.1

EDWHL3 Heavy Laser Cannon upg.2

EDWHI1 Ion Cannon (heavy tank)

EDWHI2 Ion Cannon upg. (heavy tank)

EDWAC1 20 mm Chaingun (air unit)

EDWAC2 20 mm Double Chaingun (air unit)

EDWAR1 Rocket Launcher upg.1 (air unit)

EDWAR2 Rocket Launcher upg.2 (air unit)

EDWAMR1 Heavy Rocket Launcher (air unit)

EDWAB1 Bomb Bay (3 bombs)

EDWAB2 Bomb Bay (5 bombs)

EDCA1 AA Gun [MP]

EDWSPECIAL AAR [MP]

EDWAIA1 AA Gun [MP]

EDWEQ1 Earthquake gen. [MP]

EDWEQ2 Earthquake gen.upg.1 [MP]

EDWSH1 Nuclear warheads [MP]

EDWSH2 Nuclear warheads upg.1 [MP]

EDCART Artillery (building) [MP]

EDWART Artillery (ship) [MP]

ED special equipment

EDSCA1 Carrier

EDSAC1 Container Hook

EDSCREAMER1 Noise Generator

EDSCREAMER2 Noise Generator upg.1

EDSCREAMER3 Noise Generator upg.2

EDREPAIR1 Repairer

EDREPAIR2 Repairer upg.1

EDREPAIR3 Repairer upg.2

EDBANNER Banner

EDSUT Transport hook [MP]

EDSBC1 Buildings capturer [MP]

UCS chasis

EDSUB1 Mammoth

EDSUM1 Minelayer

EDSUM2 Minelayer II

EDUSL1 Tiger

EDUSL2 Tiger II

EDUSL3 Tiger III

EDSUM1 Spider

EDSUM2 Spider II

EDSUM3 Spider III

EDSUL1 Panther

EDSUL2 Panther II

EDSUL3 Panther III

EDSBL1 Jaguar

EDSBL2 Jaguar II

EDSUL1 Grizzly

EDSUL2 Grizzly II

EDSUL3 Grizzly III

EDSUS1 Shark

EDSUS2 Shark II

EDSUS3 Shark III

EDSUS1 Hydra

EDSUS2 Hydra II

EDSUS3 Hydra III

EDSUA1 Gargoyle

EDSUA2 Gargoyle II

EDSUA3 Gargoyle III

EDSUA1 Bat

EDSUA2 Bat II

EDSUA3 Dragon

EDSUA2 Dragon II

EDSUA1 Condor

UCSUOH1	Harvester
UCSUOH2	Harvester II
UCSUOH3	Harvester III
UCSUAH1	Harvester IV
UCSUAH2	Harvester V
UCSUAH3	Harvester VI
UCSUCS	Cargo Salamander [MP]
UCSUIT	Unit Transporter [MP]
UCSUSM1	Orca [MP]
UCSUSM2	Orca II [MP]
UCSUCTF	CTF Flag [MP]
UCS weapons	
UCSWTCH1	20 mm Chaingun (Tiger)
UCSWTCH2	20 mm Chaingun upg.1 (Tiger)
UCSWTSP1	Plasma Cannon (Tiger)
UCSWTSP2	Plasma Cannon upg.1 (Tiger)
UCSWTSR1	Rocket Launcher (Tiger)
UCSWTSR2	Rocket Launcher upg.1 (Tiger)
UCSWTSR3	Rocket Launcher upg.2 (Tiger)
UCSWBMR1	Heavy Rocket Launcher (Panter/Jaguar)
UCSWBMR2	Heavy Rocket Launcher upg.1 (Panter/Jaguar)
UCSWBMR3	Heavy Rocket Launcher upg.2 (Panter/Jaguar)
UCSWBSR1	Rocket Launcher (Panter/Jaguar)
UCSWBSR2	Rocket Launcher upg.1 (Panter/Jaguar)
UCSWBSR3	Rocket Launcher upg.2 (Panter/Jaguar)
UCSWBHP1	Heavy Plasma Cannon
UCSWBHP2	Heavy Plasma Cannon upg.1
UCSWBHP3	Heavy Plasma Cannon upg.2
UCWSISCH1	20 mm chaingun (Spider)
UCWSISCH2	20 mm changun (Spider)
UCSWTSG1	Grenade Launcher (Tiger)
UCSWTSG2	Grenade Launcher (Tiger)
UCSWBHG1	Grenade Launcher (Panther/Jaguar)
UCSWBHG2	Grenade Launcher (Panther/Jaguar)
UCSWSSP1	Plasma Cannon (Spider)
UCSWSSP2	Plasma Cannon (Spider)
UCSWSSR1	Rocket Launcher (Spider)
UCSWSSR2	Rocket Launcher (Spider)
UCSWSSR3	Rocket Launcher (Spider)
UCSWSMR1	Heavy Rocket Launcher (Spider/Jaguar)
UCSWSMR2	Heavy Rocket Launcher (Spider/Jaguar)
UCSWDSR1	Rocket Launcher (Grizzly/Hydra)
UCSWDSR2	Rocket Launcher (Grizzly/Hydra)
UCSWDSR3	Rocket Launcher (Grizzly/Hydra)
UCSWDMR1	Heavy Rocket Launcher (Grizzly/Hydra)
UCSWDMR2	Heavy Rocket Launcher (Grizzly/Hydra)
UCSWDHP1	Heavy Plasma Cannon (Grizzly/Hydra)
UCSWDHP2	Heavy Plasma Cannon (Grizzly/Hydra)
UCSWACH1	20 mm Chaingun (Gargoyle)
UCSWACH2	20 mm Double Chaingun (Gargoyle)
UCSWASR1	Rocket Launcher (Gargoyle)
UCSWASR2	Rocket Launcher upg.1 (Gargoyle)
UCSWAMR1	Heavy Rocket Launcher (Bat/Dragon)
UCSWAMR2	Heavy Rocket Launcher upg.1 (Bat/Dragon)
UCSWAPB1	Bomb Bay (Bat/Dragon)
UCSWAPB2	Bomb Bay upg.1 (Bat/Dragon)
UCSWTSC1	Cannon 105mm [MP]
UCSWTSC2	Double Cannon 105mm [MP]
UCSWBHC1	Double Cannon 120mm [MP]
UCSWBHC2	Quad Cannon 120mm [MP]
UCSCAA1	AA PlasmaGun [MP]
UCSCAA2	AA PlasmaGun II [MP]
UCSWAP1	Gargoil Plasma Cannon [MP]
UCSWAP2	Gargoil Plasma Cannon m2 [MP]
UCSWAN1	Antirocket [MP]
UCSWEQ1	Earthquake gen. [MP]
UCSWEQ2	Earthquake gen.upg.1 [MP]
UCSWSHR1	Plasma warheads [MP]
UCSWSHR2	Plasma warheads upg.1 [MP]
UCSCART	Artillery [MP]
ULS	special equipment
UCSWSH1	Shadow Generator
UCSWSH2	Shadow Generator
UCSWSH3	Shadow Generator
UCSRADAR1	Radar
UCSREPAIR1	Repairer
UCSREPAIR2	Repairer upg.1
UCSBANNER	Banner
UCSSUT	Transport hook [MP]
UCSSBC1	Buildings capturer [MP]
LC chassis	
LCUU1	Lunar m1
LCUU2	Lunar m2
LCUU3	Lunar m3
LCUM01	Moon m1
LCUM02	Moon m2
LCUM03	Moon m3
LCUCR1	Crater m1
LCUCR2	Crater m2
LCUCR3	Crater m3
LCUCU1	Crusher m1
LCUCU2	Crusher m2
LCUCU3	Crusher m3
LCUHT1	Crión
LCUHERO	Fang
LCUOB1	Phobos
LCUME1	Meteoro m1
LCUME2	Meteoro m2
LCUME3	Meteoro m3
LCUBO1	Thunderer m1
LCUBO2	Thunderer m2
LCUAS1	Mercury
LCUFO	Alien craft
PROTOTYPE	PROTOTYPE
LCUGF1	Fat Girl c2 [MP]
LCUGF2	Fat Girl c3 [MP]
LCUGF3	Fat Girl c4 [MP]
LCUSF1	Super Fighter [MP]
LCUSF2	Super Fighter m2 [MP]
LCUUT	Unit Transporter [MP]
LCUDT	Tunnel Digger [MP]
LCUNH	New Hope [MP]
LCUCTF	CTF Flag [MP]
LC weapons	
LCWCH1	20 mm Changun
LCWCH2	20 mm Double Changun
LCWSR1	Rocket Launcher
LCWSR2	Rocket Launcher upg.1
LCWSR3	Rocket Launcher upg.2
LCWSL1	Electro-Cannon
LCWSL2	Electro-Cannon upg.1
LCWSS1	Sonic Cannon
LCWSS2	Sonic Cannon upg.1
LCWMR1	Heavy Rocket Launcher
LCWMR2	Heavy Rocket Launcher upg.1
LCWMR3	Heavy Rocket Launcher upg.2
LCWHL1	Heavy Electro-Cannon
LCWHL2	Heavy Electro-Cannon upg.1
LCWHS1	Heavy Sonic Cannon
LCWHS2	Heavy Sonic Cannon upg.1
LCWHC1	Plasma Cannon
LCWHERO	Plasma Beam Projector
LCWAC1	Changun (air unit)
LCWAC2	Changun upg.1 (air unit)
LCWAR1	Rocket Launcher (air unit)
LCWAR2	Rocket Launcher upg.1 (air unit)
LCWAMR1	Heavy Rocket Launcher (air unit)
LCWAMR2	Heavy Rocket Launcher upg.1 (air unit)
LCWAS1	Heavy Sonic Cannon (air unit)
LCWAS2	Heavy Sonic Cannon upg.1 (air unit)
LCWFU0	Alien Weapon
LCCA1	AA Rocket Launcher [MP]
LCWA1	AA Rocket Launcher [MP]
LCWAN1	Antirocket [MP]
LCWNH	Plasma Beam [MP]
LCWEQ1	Earthquake gen. [MP]
LCWEQ2	Earthquake gen.upg.1 [MP]
LC CART	Artillery [MP]
LC	special equipment
LCSOB1	Detector m1
LCSOB2	Detector m2
LCSSH2	Shield Recharger m2
LCSSH3	Shield Recharger m3
LCREG1	Regenerator m1
LCREG2	Regenerator m2
LCREG3	Regenerator m3
LCBANNER	Banner
LCSTU	Transport hook [MP]
LCSC1	Buildings capturer [MP]

Building identifiers

Building identifiers can be used to enable or disable build of some types of buildings in mission or base by function player.EnableBuilding. Note that calling this function for some special buildings (like silos or solar battery) doesn't have effect. Also AI does not takes care of this function so you can only disable buildings for human players not for AI players.

Identifier Name

ED buildings
EDBBA Vehicle Production Center
EDBFA Weapons Production Center
EDBPP Power Plant

EDBMI Mine
 EDBRE Refinery
 EDBTC Transport Base
 EDBRC Research Center
 EDBAB Supply Depot
 EDBWB Ship Yard
 EDBPB Pillbox
 EDBST Small Tower
 EDBBT Large Tower
 EDBBC Missile Control Center
 EDBHQ Headquarters
 EDBSI Silo
 EDBRA Radar
 EDBEN1 Tunnel entrance
 EDBEN2 Tunnel entrance
 EDBSS Space port
 EDBLZ Landing Zone
 EDBBA Vehicle Production Center
 EDBBF Weapons Production Center
 EDBPP Power Plant
 EDBMI Mine
 EDBRE Refinery
 EDBBT Transport Base
 EDBRC Research Center
 EDBAB Supply Depot
 EDBWB Ship yard
 EDBBP Pillbox
 EDBST Small Tower
 EDBBT Large Tower
 EDBBC Missile Control Center
 EDBHQ Headquarters
 EDBSI Silo
 EDBRA Radar
 EDBEN1 Tunnel entrance (surface)
 EDBEN2 Tunnel entrance (tunnel)
 EDBSS Space port
 EDBLZ Landing Zone
 EDBUC Recycler [MP]
 EDBHT Heavy Tower [MP]
 EDBART Artillery [MP]
 UCS buildings
 UCSBVA Vehicle Production Center
 UCSBFA Weapons Production Center
 UCSPP Power Plant
 UCSBPR Nuclear Reactor
 UCSBTB Ore Transport Base
 UCSBET Energy Transmitter
 UCSRF Refinery
 UCSRC Research Center
 UCSBAB Aerial Supply Depot
 UCSWB Ship Yard
 UCSFO Fortress
 UCSBT Small Tower
 UCSBBT Large Tower
 UCSPB Plasma Control Center
 UCSPPC Plasma Cannon
 UCSBHQ Headquarters
 UCSBSD SDI Defense Center
 UCSBSH Shadow Tower
 UCSBTE Teleport
 EDBEN1 Tunnel Entrance (surface)
 EDBEN2 Tunnel Entrance (tunnel)
 EDBSS Space Port
 EDBLZ Landing Zone
 EDBUC Recycler [MP]
 EDBHT Heavy Tower [MP]
 EDBART Artillery [MP]
 LC buildings
 LCBBF Main Base
 LCBPP Solar Power Plant
 LCBBA Solar Battery
 LCBMR Mine
 LCBSR Ore Transport Refinery
 LCBRC Research Center
 LCBAB Aerial Supply Center
 LCBGA Guardian
 LCBDE Defender
 LCBHQ Headquarters
 LCBSD SDI Defense Center
 LCBWC Weather Control Center
 LCBSS Space Port
 LCBSB Solar Cell
 LCBLZ Landing Zone
 WLASER Laser Wall
 LCBP2 Plantium Power Plant [MP]
 LCBCC Control Center [MP]

LCBUC Recycler [MP]
 LCBNE NEST [MP]
 LCBART Artillery [MP]
 LCBEN1 Tunnel entrance (surface) [MP]
 LCBEN2 Tunnel entrance (tunnel) [MP]

Research identifiers

Research identifiers can be used to enable or disable research production (disabled research can't be researched) with function `player.EnableResearch` and also to and researches for player (research is available without production) with function `player.AddResearch`. If you are using function `AddResearch` remember that researches must be add in specific order: from beginning (root) of research tree. So if you want to give player f.e. Kruszhev tank (research `RES_ED_UHT1`) add researches in following order:

`rPlayer.AddResearch(RES_ED_UST2);
rPlayer.AddResearch(RES_ED_UST3);
rPlayer.AddResearch(RES_ED_UHT1);`

Also you should call `AddResearch` function only at begin of mission script (in first state), and in campaign use this function combined with `EnableResearch` (to avoid giving already produced researches).

Neutral researches are available for all races.

Identifier Name

ED researches	RES_ED_UST2 Upg: Pamir
	RES_ED_UST3 Upg: Pamir
	RES_ED_UMT1 Siberia
	RES_ED_UMT2 Upg: Siberia
	RES_ED_UMT3 Upg: Siberia
	RES_ED_UMI1 Mine Layer
	RES_ED_UMI2 Upg: Mine Layer
	RES_ED_UMW1 Caspian
	RES_ED_UMW2 Upg: Caspian
	RES_ED_UMW3 Upg: Caspian
	RES_ED_UHW1 Volga
	RES_ED_UHW2 Upg: Volga
	RES_ED_UHT1 Kruszhev
	RES_ED_UHT2 Upg: Kruszhev
	RES_ED_UHT3 Upg: Kruszhev
	RES_ED_UT1 Ural
	RES_ED_UT2 Upg: Ural
	RES_ED_USS2 Upg: Irkutsk
	RES_ED_US3 Upg: Irkutsk
	RES_ED_UHS1 Leviathan
	RES_ED_UHS2 Upg: Leviathan
	RES_ED_UA11 Cossack
	RES_ED_UA12 Upg: Cossack
	RES_ED_UA21 Grozny
	RES_ED_UA22 Upg: Grozny
	RES_ED_UA31 Hani
	RES_ED_UA32 Upg: Han
	RES_ED_UA41 Thor
	RES_ED_UA42 Upg: Thor
	RES_ED_WCH2 Upg: Chaingun
	RES_ED_ACH2 Upg: Helicopter Chaingun
	RES_ED_WCA2 Upg: Cannon
	RES_ED_WH1 Heavy Cannon
	RES_ED_WH2 Upg: Heavy Cannon
	RES_ED_WSR1 Rocket Launcher
	RES_ED_WSR2 Upg: Rocket Launcher
	RES_ED_WSR3 Upg: Rocket Launcher II
	RES_ED_ASR1 Helicopter Rocket Launcher
	RES_ED_ASR2 Upg: Helicopter Rocket Launcher
	RES_ED_WMR1 Heavy Rocket Launcher
	RES_ED_WMR2 Upg: Heavy Rocket Launcher
	RES_ED_WMR3 Upg: Heavy Rocket Launcher II
	RES_ED_AMR1 Helicopter Heavy Rocket Launcher
	RES_ED_AMR2 Upg: Helicopter Heavy Rocket Launcher
	RES_ED_WH11 Ballistic Rocket Launcher
	RES_ED_WSL1 Laser Cannon
	RES_ED_WSL2 Upg: Laser Cannon
	RES_ED_WSL3 Upg: Laser Cannon
	RES_ED_WH11 Heavy Ion Cannon
	RES_ED_WH12 Upg: Heavy Ion Cannon
	RES_ED_AB1 Bomb Bay
	RES_ED_AB2 Upg: Bomb Bay
	RES_ED_BMD Medium Defense Building
	RES_ED_BHD Heavy Defense Building
	RES_ED_SGen Power Shield Generator 600 PSU
	RES_ED_MGen Power Shield Generator 1200 PSU
	RES_ED_HGen Power Shield Generator 1800 PSU

RES_ED_RepHand Repairer
RES_ED_RepHand2 Upg: Repairer
RES_ED_SCR Screamer (noise generator)
RES_ED_SCR2 Upg: Screamer (noise generator)
RES_ED_SCR3 Upg: Screamer II (noise generator)
RES_ED_MSC2 Upg: 105 mm Bullet
RES_ED_MSC3 Upg: 105 mm Bullet
RES_ED_MSC4 Upg: 105 mm Bullet
RES_ED_MHC2 Upg: 120 mm Bullet
RES_ED_MHC3 Upg: 120 mm Bullet
RES_ED_MHC4 Upg: 120 mm Bullet
RES_ED_MHR2 Upg: Ballistic Rocket
RES_ED_MHR3 Upg: Ballistic Rocket
RES_ED_MHR4 Nuclear Ballistic Rocket
RES_ED_MB2 Upg: Bomb
RES_ED_MB3 Upg: Bomb
RES_ED_MB4 Nuclear Bomb
RES_EDUT Unit Transporter [MP]
RES_EDUTEATH Stealth Unit [MP]
RES_ED_USM1 Kiev (submarine) [MP]
RES_ED_USM2 Upg: Kiev [MP]
RES_EDWAN1 Antirocket [MP]
RES_EDWA1 AA Gun [MP]
RES_EDWEQ1 Earthquake gen. [MP]
RES_EDWEQ2 Upg.Earthquake gen. [MP]
RES_EDBHT Heavy Tower [MP]
RES_ED_ART Artillery [MP]
RES_ED_BC1 Buildings capturer [MP]
UCS researches
RES_UCS_USL2 Upg: Tiger I
RES_UCS_USL3 Upg: Tiger II
RES_UCS_UML1 Spider (6 legged chassis)
RES_UCS_UML2 Upg: Spider I
RES_UCS_UML3 Upg: Spider II
RES_UCS_UHL1 Panther (2 legged heavy chassis)
RES_UCS_UHL2 Upg: Panther I
RES_UCS_UHL3 Upg: Panther II
RES_UCS_UA1 Grizzly (3 legged heavy chassis)
RES_UCS_UA2 Upg: Grizzly
RES_UCS_UA3 Upg: Grizzly II
RES_UCSUBL1 jaguar (2-legged heavy chassis)
RES_UCSUBL2 Upg: jaguar
RES_UCS_UM1 Minelayer
RES_UCS_UM2 Upg: Minelayer
RES_UCS_USS1 Shark (small ship)
RES_UCS_USS2 Upg: Shark
RES_UCS_UBS1 Hydra (ship)
RES_UCS_UBS2 Upg: Hydra
RES_UCS_UBS3 Upg: Hydra II
RES_UCS_UOH1 Upg: Harvester
RES_UCS_UOH3 Upg: Harvester II
RES_UCS_UAH1 Upg: Harvester III
RES_UCS_UAH2 Upg: Harvester IV
RES_UCS_UAH3 Upg: Harvester V
RES_UCS_GARG1 Gargoyle (fighter)
RES_UCS_GARG2 Upg: Gargoyle
RES_UCS_GARG3 Upg: Gargoyle II
RES_UCS_BOMBER21 Bat (bomber)
RES_UCS_BOMBER22 Upg: Bat
RES_UCS_BOMBER31 Dragon (heavy bomber)
RES_UCS_BOMBER32 Upg: Dragon
RES_UCS_WCH2 Double Chaingun
RES_UCS_IWACH2 Upg Gargoyle Chaingun
RES_UCS_WSG1 Grenade Launcher
RES_UCS_WSG2 Upg: Grenade Launcher
RES_UCS_WHG1 Heavy Grenade Launcher
RES_UCS_WSR1 Small Rocket Launchers
RES_UCS_WSR2 Upg: Small Rocket Launchers I
RES_UCS_WSR3 Upg: Small Rocket Launchers II
RES_UCS_WASR1 Gargoyle Rocket Launcher
RES_UCS_WASR2 Upg: Gargoyle Rocket Launcher
RES_UCS_WMR1 Rocket Launchers I
RES_UCS_WMR3 Upg: Rocket Launchers II
RES_UCS_WAMR1 Bomber Rocket Launcher
RES_UCS_WAMR2 Upg: Bomber Rocket Launcher
RES_UCS_WSP1 Plasma Cannons
RES_UCS_WSP2 Upg: Plasma Cannons
RES_UCS_WHPI Heavy Plasma Cannons
RES_UCS_WHPI2 Upg: Heavy Plasma Cannons I
RES_UCS_WHPI3 Upg: Heavy Plasma Cannons II
RES_UCS_WAPB1 Bomb Bay
RES_UCS_WAPB2 Upg: Bomb Bay
RES_UCS_WSD SDI Laser
RES_UCS_BMD Medium Defense Building
RES_UCS_BHD Heavy Defense Building
RES_UCS_RepHand Repairer
RES_UCS_RepHand2 Upg: Repairer
RES_UCS_SGen Power Shield Generator 600 PSU
RES_UCS_MGen Power Shield Generator 1200 PSU
RES_UCS_HGen Power Shield Generator 1800 PSU
RES_UCS_SHD Shadow Generator
RES_UCS_SHD2 Upg: Shadow Generator I
RES_UCS_SHD3 Upg: Shadow Generator II
RES_UCS_SHD4 Upg: Shadow Generator III
RES_UCS_MB2 Upg: Plasma Bomb
RES_UCS_MB3 Upg: Plasma Bomb
RES_UCS_MB4 Upg: Plasma Bomb
RES_UCS_MG2 Upg: Grenade
RES_UCS_MG3 Upg: Grenade
RES_UCS_MG4 Upg: Grenade
RES_UCS_PC Offensive Plasma Cannon
RES_UCSUCS Cargo Salamander [MP]
RES_UCSUU Unit Transporter [MP]
RES_UCS_USM1 Orca (submarine) [MP]
RES_UCS_USM2 Upg: Orca [MP]
RES_UCSWAP1 Gargoil Plasma Cannon [MP]
RES_UCSWAP2 Gargoil Plasma Cannon m2 [MP]
RES_UCSAA1 AA PlasmaGun [MP]
RES_UCSAA2 AA PlasmaGun II [MP]
RES_UCSWTSC1 Cannon 105mm [MP]
RES_UCSWTSC2 Double Cannon 105mm [MP]
RES_UCSBWHC1 Double Cannon 120mm [MP]
RES_UCSBWHC2 Quad Cannon 120mm [MP]
RES_UCSWAN1 Antirocket [MP]
RES_UCSWEQ1 Earthquake gen. [MP]
RES_UCSWEQ2 Upg.Earthquake gen. [MP]
RES_UCSBHT Heavy Tower [MP]
RES_UCS_ART Artillery [MP]
RES_UCS_BC1 Buildings capturer [MP]
RES_UCS_MSC2 Upg: 105 mm Bullet [MP]
RES_UCS_MSC3 Upg: 105 mm Bullet [MP]
RES_UCS_MSC4 Upg: 105 mm Bullet [MP]
RES_UCS_MHC2 Upg: 120 mm Bullet [MP]
RES_UCS_MHC3 Upg: 120 mm Bullet [MP]
RES_UCS_MHC4 Upg: 120 mm Bullet [MP]
LC researches
RES_LC_UU2 Upg: Lunar
RES_LC_UU3 Upg: Lunar
RES_LC_UU02 Upg: Moon (medium tank)
RES_LC_UU03 Upg: Moon
RES_LC_UCR1 Crater (heavy tank)
RES_LC_UCR2 Upg: Crater
RES_LC_UCR3 Upg: Crater
RES_LC_UU1 Crusher (two cannon tank)
RES_LC_UU2 Upg: Crusher
RES_LC_UU3 Upg: Crusher
RES_LC_UME1 Meteor (fighter)
RES_LC_UME2 Upg: Meteor
RES_LC_UME3 Upg: Meteor
RES_LC_UBO1 Thunderer (bomber)
RES_LC_UBO2 Upg: Thunderer
RES_LC_WCH2 Upg: Chaingun
RES_LC_ACH2 Upg: Air Chaingun
RES_LC_WSR1 Rocket Launcher
RES_LC_WSR2 Upg: Rocket Launcher
RES_LC_WSR3 Upg: Rocket Launcher
RES_LC_ASRI Air Rocket Launcher
RES_LC_ASRI2 Upg: Air Rocket Launcher
RES_LC_WMR1 Heavy Rocket Launcher
RES_LC_WMR2 Heavy Rocket Launcher upg.1
RES_LC_WMR3 Heavy Rocket Launcher upg.2
RES_LC_AMR1 Air Heavy Rocket Launcher
RES_LC_AMR2 Air Heavy Rocket Launcher upg.1
RES_LC_WSL1 Electro-Cannon
RES_LC_WSL2 Upg: Electro-Cannon
RES_LC_WHL1 Heavy Electro-Cannon
RES_LC_WHL2 Upg: Heavy Electro-Cannon
RES_LC_WSS1 Sonic Cannon
RES_LC_WSS2 Upg: Sonic Cannon
RES_LC_WHIS1 Heavy Sonic Cannon
RES_LC_WHIS2 Upg: Heavy Sonic Cannon
RES_LC_WAS1 Air Sonic Cannon
RES_LC_WAS2 Upg: Air Sonic Cannon
RES_LC_WARTILLERY Plasma Artillery
RES_LC_BMD Medium Defense Building
RES_LC_BHD Heavy Defense Building
RES_LC_BWC Weather Control Center
RES_LC_SOBI Detector
RES_LC_SOBU Upg: Detector
RES_LC SHR1 Shield Recharger

RES_LC_SHR2 Upg. Shield Recharger m1
RES_LC_SHR3 Upg. Shield Recharger m2
RES_LC_SGen Power Shield Generator 1200 PSU
RES_LC_MGen Power Shield Generator 2400 PSU
RES_LC_HGen Power Shield Generator 3600 PSU
RES_LC_REG1 Regenerator
RES_LC_REG2 Upg. Regenerator m1
RES_LC_REG3 Upg. Regenerator m2
RES_LC_SDIDEF SDI Defense
RES_LCUFG1 Fat Girl c2 [MP]
RES_LCUFG2 Fat Girl c3 [MP]
RES_LCUFG3 Fat Girl c4 [MP]
RES_LCUSF1 Super Fighter [MP]
RES_LCUSF2 Super Fighter m2 [MP]
RES_LCUNH New Hope [MP]
RES_LCUIT Unit Transporter [MP]
RES_LCAA1 AA Rocket Launcher [MP]
RES_LCWAN1 Antrocket [MP]
RES_LCWA1 AA Rocket Launcher (building) [MP]
RES_LCWEQ1 Earthquake gen. [MP]
RES_LCWEQ2 Upg.Earthquake gen. [MP]
RES_LCBP2 Plantium Power Plant [MP]
RES_LCBNE NEST [MP]
RES_LC_ART Artillery [MP]
RES_LC_BCI Buildings capturer [MP]
Neutral researches (for all races)
RES_MCH2 Upg: 20 mm Bullets
RES_MCH3 Upg: 20 mm Bullets
RES_MCH4 Upg: 20 mm Bullets
RES_MSR2 Upg 1: Rocket (guided: 25%)
RES_MSR3 Upg 2: Rocket (guided: 50%)
RES_MSR4 Upg 3: Rocket (guided: 100%)
RES_MMRA Upg: Heavy Rocket (guided: 25%)
RES_MMRI Upg: Heavy Rocket (guided: 50%)
RES_MMRA Upg: Heavy Rocket (guided: 100%)

Artifact identifiers

Artifact identifiers can be used in function mission.CreateArtifact to create normal or special (handled by event mission.Artifact) artifacts.

Identifier Name
Normal artifacts
NEAMAMMO Ammunition
NEAENERGY Energy
NEAGIVESHIELD Shield generator
NEAMAXHP Full repair
NEAGIVEMONEY Money 10 000 CR
NEASHOWMAP0 Show surface map
NEASHOWMAP1 Show tunnel map
Special artifacts (script-handled)
NEASPECIAL1 Artifact
NEASPECIAL2 Artifact
NEASPECIAL3 Artifact
NEASPECIAL4 Artifact
NEASPECIAL5 Artifact
NEASPECIAL6 Artifact [MP]
NEASPECIAL7 Artifact [MP]
NEASPECIAL8 Artifact [MP]
NEASPECIAL9 Artifact [MP]
NEASPECIAL10 Artifact [MP]
NEASPECIAL11 Artifact [MP]
NEASPECIAL12 Artifact [MP]
NEASPECIAL13 Artifact [MP]
NEABANNERUS Artifact [MP]
NEABANNED Artifact [MP]
NEABANNERLC Artifact [MP]
NEACOMPUTER Artifact [MP]
NEASWITCH1 Artifact [MP]
NEASWITCH2 Artifact [MP]
NEAPLATE1 Artifact [MP]
NEAPLATE2 Artifact [MP]
NEAMONEY Artifact [MP]
NEAMINE Artifact [MP]

MoonC tool

Using MoonC

When you will write your script you must to compile them to binary form. To do that use EarthCMP.exe tool. Call it as follows:

EarthCMP.exe sourceScript.ec sourceScript.ecMP

As you can see binary EarthC file must have extension "*.ecoMP" (*.ecoMP for "The Moon Project" scripts). You must copy this script to directory depends on script type. Look at Scripts directory structure to check it. Also note that game type scripts for multiplayer must be placed in 'Scripts\Gametypes' directory and for skirmish in 'Scripts\Gametypes\single' directory.

Scripts compiled with included EarthCMP.exe tool are working only with The Moon Project version 1.0 and can crash while used in older game versions.

WDCreator tool

Same as in Earth!

LangC tool

Same as in Earth!

DZUXXEZ
THE GUITAR INSTRUMENTS & E



Tips&Tricks

Tips&Tricks&various notes

If you want to create your own scripts you should read this notes to avoid some surprises, artifacts or game crashes.

"The Moon Project":

New functions and objects from "The Moon Project" included in this documentation are signed with symbol [MP].

Compiled scripts for "The Moon Project" must have extension 'ecoMP' not 'eco'.

In The Moon Project there is a special research "RES_MISSION_PACK1_ONLY" used as a preceding research for all "MP" researches. This research must be added at begin of game for each player to enable new "MP" researches.

Some scripts for The Moon Project are located in f.e. "MP-ED" or "MP" directory, but compiled scripts must be copied to the same directories as for other scripts.

EarthC and other tools in this package were designed for developer use only. It means that these tools are not very error-resistant. For example you can write script which is compiled without errors but game will crash when this script is used. Before writing your own scripts you should study included script sources, and if some bugs in your scripts occurs try to compare it with our scripts - some functions must be used in some specific order or with other functions.

Scripts compiled with included EarthCMP.exe tool are working only with The Moon Project version 1.0 and can crash while used in older game versions.

Test your scripts well before you publish them. We are not responsible for psychical/physical damages made by your buggy scripts :).

Don't be surprised if you will see some strange comments in scripts. It's Polish. (And BTW sorry for poor and buggy English in this documentation)

Don't copy this documentation to The Moon Project game directory. At startup Earth scans this directory and read headers of all files, so doing this operation with large amount of files takes a lot of time.

Use WDCreator tool - it's comfortable because it creates one file instead of many files in many directories.

In multiplayer game all missing files like non-standard levels, user banners or scripts (gametypes, AI players, unit scripts) are downloaded before game starts (in session configuration dialog). If such a file is placed in "*.wd" file then whole "*.wd" file is downloaded. So don't make too big "*.wd" files (f.e. don't put 100 multiplayer levels into single file) because it take too much time to download it.

You don't need to copy ".wd" files to "WDFiles" directory. It's good idea to copy this files to f.e. "CustomWDFiles" directory.

If you will modify some standard Earth files (like levels or scripts) save it under other name (otherwise it will cover standard file from "*.wd" file).

Many EarthC functions uses values called "tick". Tick is one game simulation step. For default game speed game makes 20 ticks per second (5 for minimum, 50 for maximum game speed). But if game have a lot to calculate and a lot to display (f.e. battle displayed in maximum zoom out) than (especially at maximum game speed) game can slow down and make less ticks per second than it should do.

File names for mission levels should start with char '!' (f.e. l234.lnd, l234.mis). Such a files are not displayed in skirmish/multiplayer configuration dialog. To load such a file in editor (which normally don't display files beginning with '!') use console command 'editor.singleplayer 1' (in editor press Enter, type: editor.singleplayer 1 and press Enter again). If you are bored doing it all the time you can put this line in file 'autoexec.con' in main The Moon Project directory. Using 'editor.singleplayer' command you can open all levels from standard Earth campaigns.

Use LangC and "translate**" strings in scripts instead of putting plain text. It looks much better, don't cost a lot of time and your scripts can be easily translated to other language.

Scripts of all types are stored in saves, and when save is loaded game uses scripts stored in save file. So if you want for example to test changed mission script you must start this mission again (choose it in choose mission dialog). If you have changed campaign script you must start the whole campaign again.

You can debug your scripts. To do that use script function TraceD with parameter string or number. To see this traces you must turn on display with console command: 'display.show 1'.

Game type scripts (for skirmish and multiplayer), are stored in two directories. Scripts for multiplayer games are stored in 'Scripts\Gametypes' directory and scripts for skirmish are stored in 'Scripts\Gametypes\single' directory.

In unit and mission scripts delete any reference members (of type unit, tank, etc) in command UnitInitialize. It's very important, otherwise game can crash after exiting campaign mission.)

Your campaign script name must begin with CampaignED (CampaignUCS, CampaignLC or TutorialED, TutorialUCS, TutorialLC for tutorial campaigns) f.e.:

CampaignEDSomeExtension.ecoMP (or TutorialEDSomeExt.ecoMP for tutorial) and must be placed in directory Scripts\Campaigns\ED (or UCS, LC for other races). If your campaign name (in first line of script) have name f.e. "translateCampaignEDMySuperCamp" then you can add string with "Description" suffix ("translateCampaignEDMySuperCampDescription") - with short description of your campaign. This string will be displayed in the text box in 'Choose campaign' dialog.

If you have added some custom campaign (or tutorial) you should see 'Choose campaign' dialog after pressing 'Start new game' or 'Tutorial' button. If this dialog is not displayed (and game start immediately) it means that your campaign script have bad file name, or is placed in bad directory.

If you are testing your campaign you can use console command 'ai.allmissions 1' - then you can select any mission in 'Choose mission' dialog.

In any campaign mission you can use console command 'world.endmission 1' - then button 'End mission' appears immediately.

You can switch to any AI player in game to see if it works properly and as you expect it to work. To do this type console command 'ai.local n' where n is an IFF number' (slot in which player is placed - numbered from 0 to 15). In this case you must first enable cheats with command '_wanna_cheat'. Note that you will play as AI player but AI in this player is still functioning and doing its job.

Note that you can't add any custom levels or scripts to Demo version of The Moon Project.

**THE
TOMORROW
SHOW**

COLLECTOR'S EDITION

JEUNISM SIGA