Prototype Design

G.Affleck

Why bother with design?

Complex tasks are simply too difficult to think through without planning.

Prior Planning Prevents Poor Performance.

Modelling a system:

A representation using abstracted relevant detail while discarding irrelevant detail.

Often **diagram techniques** are used – Why?

Using diagramming conventions plans become easily communicable:

A picture is worth a thousand words.

Modelling With UML

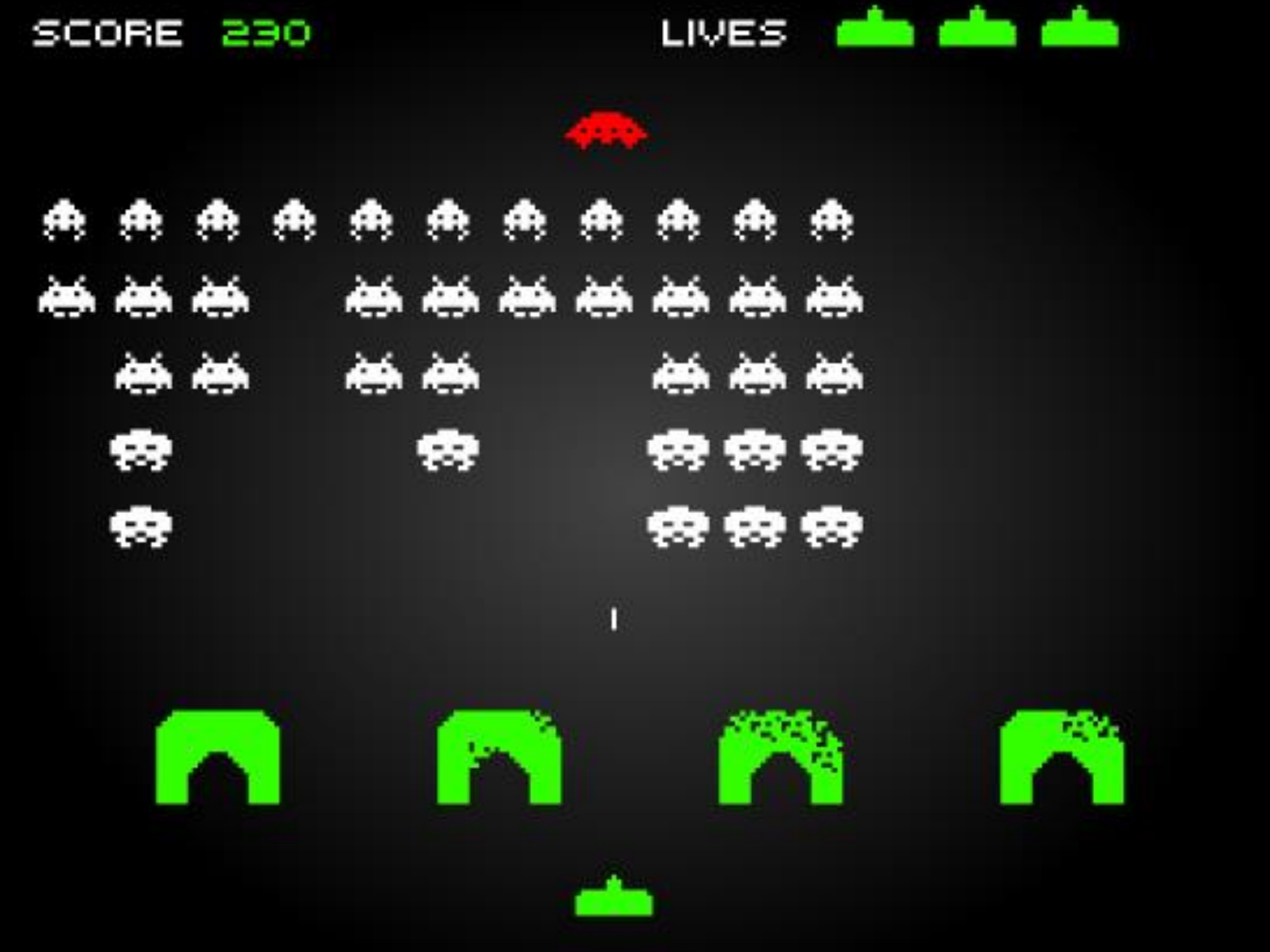
Use case diagrams

Describe the functional behaviour of the system from user view-point.

The external initiations of internal activity.

What are we going to model as
an example?

Let’s look at a well known
and very successful game!


SCORE 230
LIVES

How would you create - What does the game consist of?

What is the first steps to modelling

Make rough drawings/storyboard what you think the interface will look like

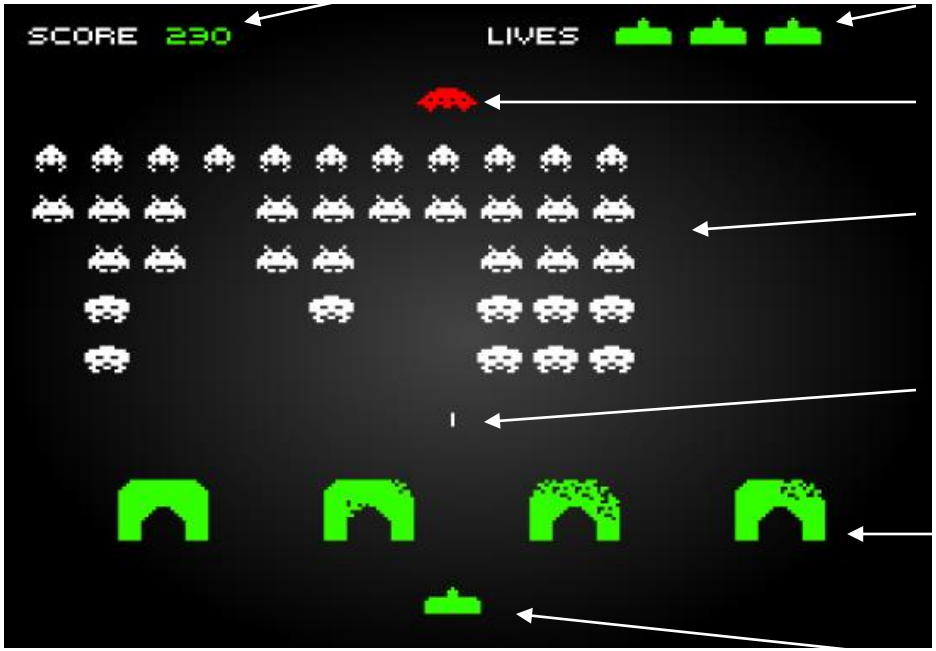
In our example of a game look for:

Game objects/entities/tokens

Description\Requirements

Model Behaviour

Looking for: Game Entities\Objects\Tokens:



I have simply listed everything I see on the screen (and the screen).

Description / Requirements list:

The Alien

Rows of aliens will descend from the top of the screen. They will move across the screen, then drop when they reach the edge of screen (left or right). The speed will increase as numbers decrease and as they get lower.

If the aliens reach the bottom of the screen, the game will end.

Bombs dropped by aliens

The aliens will drop random bombs. Multiple bombs can be dropped at the same time.

If a bomb hits the defender, one life is subtracted from the number of lives left.

Bomb Collision :	hits a defender.
	hits a bullet.
	goes off screen.
	Hits a base.

The Defender

The **defender** can move left to right at the bottom of the screen,

but cannot move off-screen.

The **defender** can fire **bullets**.

Collisions:

If a bullet hits an alien

A bullet may hit a dropped bomb

Hit a **spaceships**
goes off the screen.
Hits base shield

Base

The base consists of sections and as either bombs dropped by aliens or bullets fired by defender hit the **bases** sections are blown away.

The base will lose strength as parts disintegrate and ultimately the base will disappear.

Once destroyed the bases will not reappear with each screen.

base	–	Decrease strength
		Hit by bomb
		Hit by bullet

Spacecraft

At given intervals a spacecraft will appear at one side and travel to the other side. If the player hits the spaceship bonus points are awarded. If the player hits the spaceship then the next bonus points will be greater e.g. 50,100,150, 200. However when the player misses then the points will return to 50.

Spacecraft -	Trigger
	Move
	IsHit
	No of points

The Defender' Lives

The defender starts with 3 lives. At the end of each screen when all the aliens are killed the screen will reset and the no of live is increased by 1.

If the defender is killed the player will lose one life and the game will continue.

If the defender has no lives left , the game will end.

No of lives will simply increase/ decrease

Score –

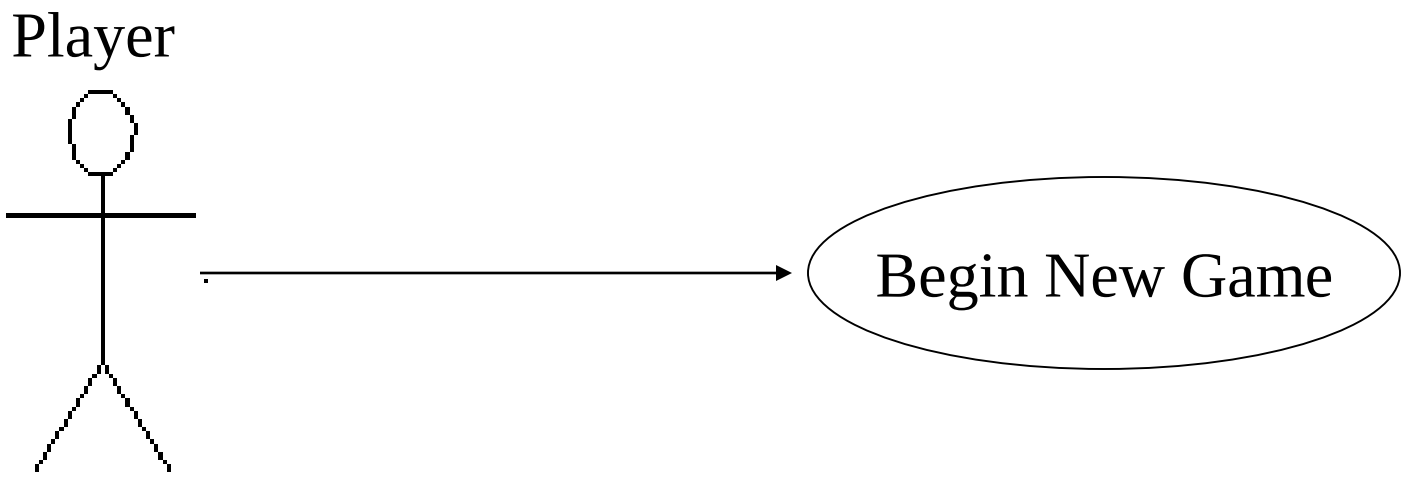
Increase as aliens are killed and as spaceships are hit.

Score and Lives:
Player/Defender attributes?

Modelling Behaviour
UML Use Case Diagrams

A use case describes a sequence of actions that provide a measurable value to an actor.

A use case is drawn as a horizontal ellipse on a UML use case diagram.

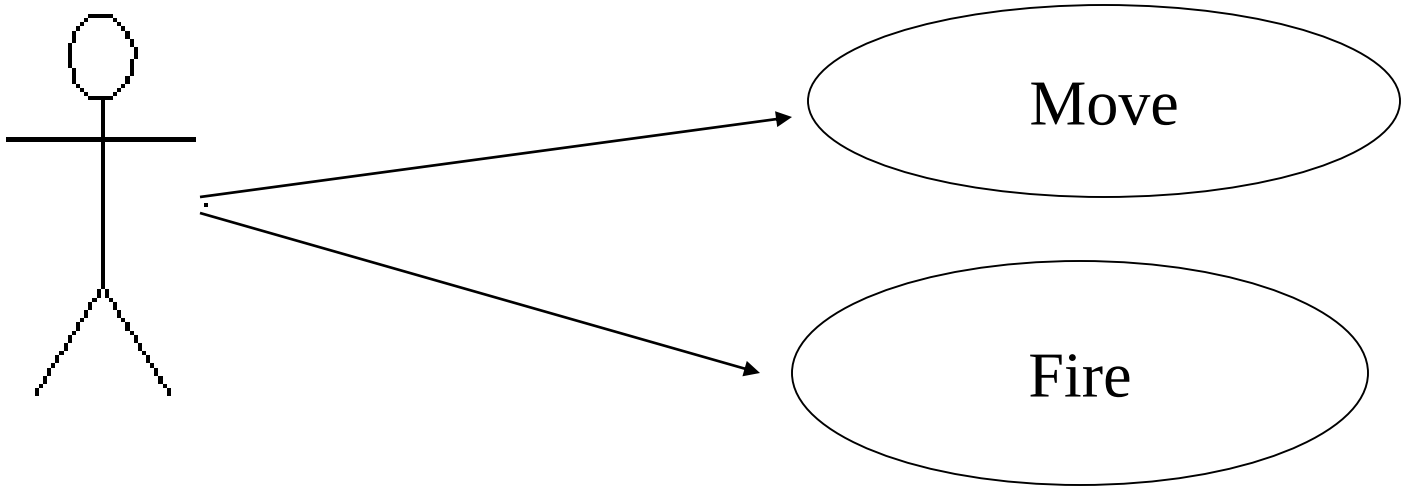


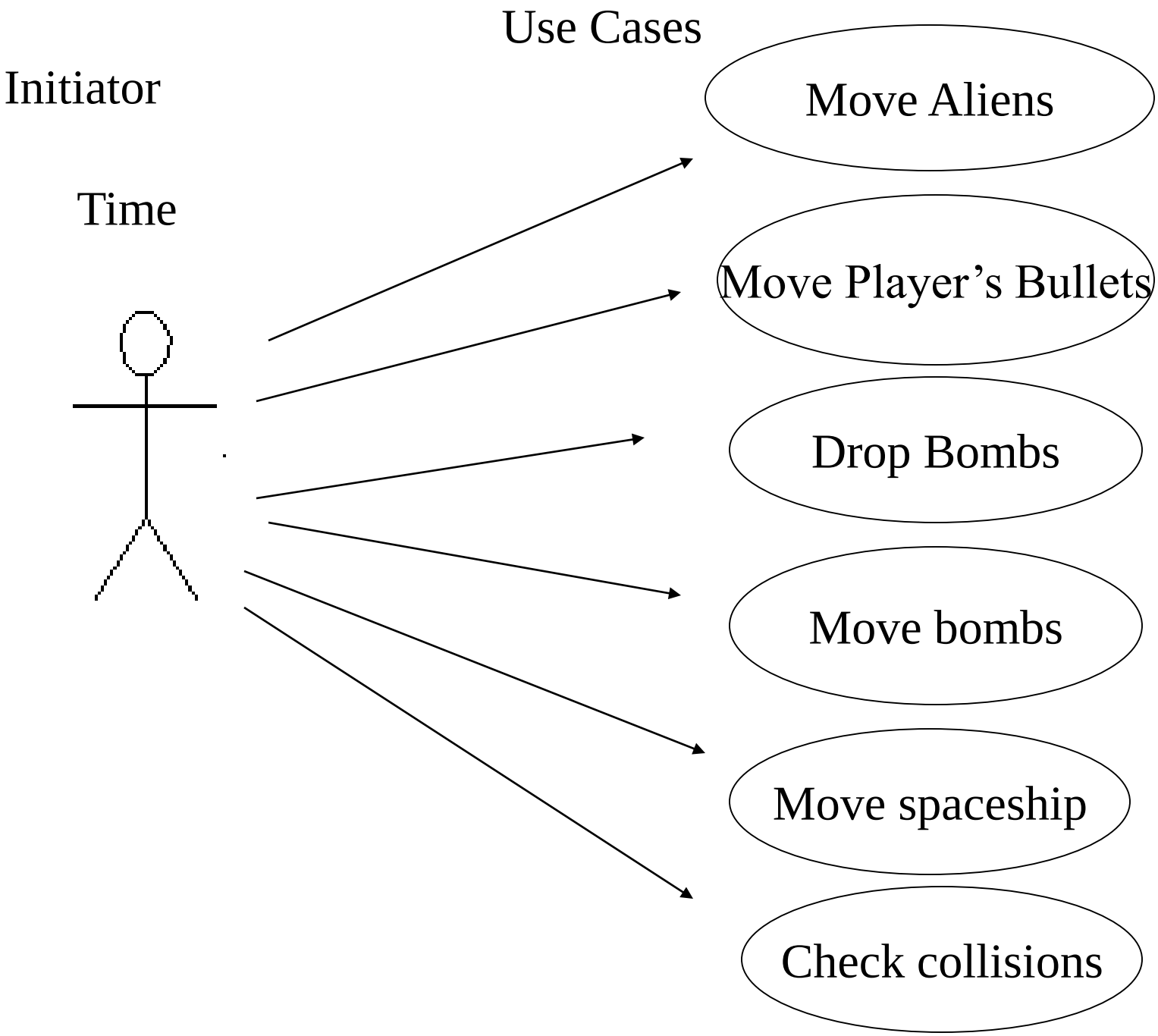
Example Diagrams from Space Invaders

Initiator

Use Cases

Player




Use Cases
Initiator
Time
Move Aliens
Move Player's Bullets
Drop Bombs
Move bombs
Move spaceship
Check collisions

Use Case are then detailed with:

Who or what initiates

Pre-conditions

Flow of events

Post-conditions

Use Case Name
Id: Use Case Identifier
Actors: who or what initiates
Preconditions: What must be true before this use case will be triggered.
Flow of events: What happens – the actual steps in the use case.
Post Conditions: What will be true after the use case has executed.

Example:

Player firing
Id: Use Case 5
Actors: Player
Preconditions: Game is executing and no other firing is in progress.
Flow of events: 1 User presses the firing key. 2 The system creates a new upward moving missile at the ship's location. 3 Use case ends.
Post Conditions: A player firing is in progress.

Few tasks will be a simple sequence to model.

Within a flow of events We will need to model such as:

Selection	Selection of use case
Repetition	Repetition of use cases
Nesting : sub tasks	Use case with a use case

In use case diagrams we use **relationships** to model these.

Relationships

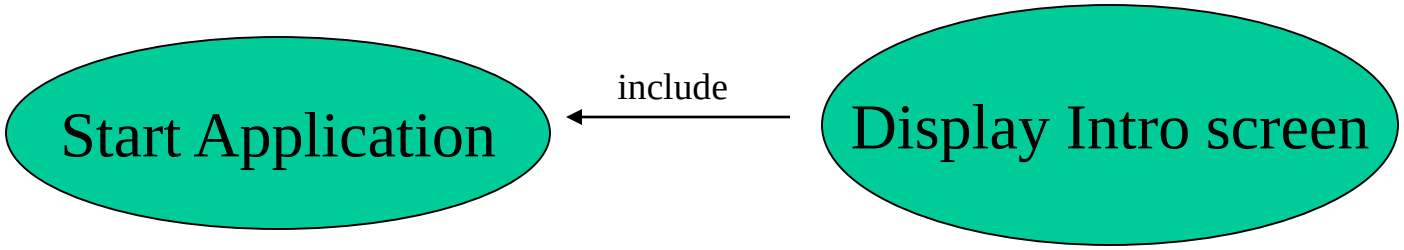
Relationships shown on diagrams help model such flow.

Apply <<**include**>> when you know exactly when to invoke a use case

Place the <<**include**>> use case to the right of the invoking use case (as seen previously).

Note the use case invokes – not the actor.

In diagram:



In flow:

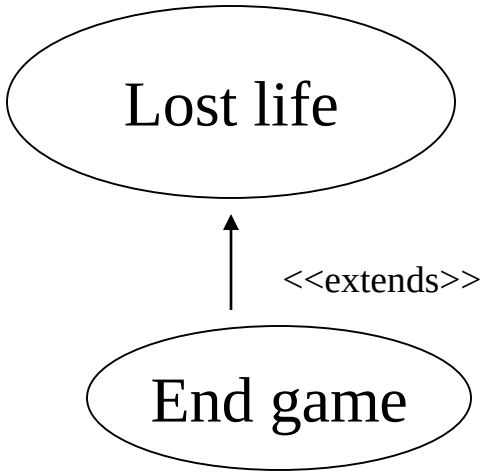
Start Application Use Case

1. Perform initialisation
2. **<<include>> Display Intro screen use case**
3. End Use case

Note the numbering

Apply <<**extend**>> when there is an exceptional or seldom invoked case.

In diagram:



Place the extending use case below the parent use case.

In flow:

If no more lives left

<<extend>> using End Game case study

Examples of Flow of events:

Selection/Branching

1 If player firing

1.1 check bullet collision with base

1.2 check bullet collision with alien

1.3 check bullet collision with screen boundaries

Repetition

Using keyword For

2. For each bomb being dropped

2.1 Check collision with base

2.2 Check collision with player ship

2.3 Check bomb collision with screen boundaries

2.4 if player firing

2.4.1 check collision with bullet

Repetition

Using keyword While

1 while ship and aliens are present (e.g., not destroyed) do

1.1 «include» Move Aliens Use Case.

1.2 «include» Move Missiles Use Case.

1.3 «include» Drop Bomb Use Case.

1.4 «include» Move Spaceship Use Case.

1.5 «include» Check Collisions Use Case.

Example Space Invaders Use Case detail

Use Case: Launch Application

Id: Use Case 1

Actors: Player

Preconditions:
Application installed.

Flow of events:
1 Player double clicks on the Space Invaders program icon.
2 The system initializes
3 «include» Display Intro Screen use case.
4 Use case ends.

Post Conditions: The introduction screen is displayed.

Display Intro Screen
Id: Use Case 2
Actors: Player
Preconditions: Application launched.
Flow of events: 1 System displays the introduction screen. 2 Use case ends.
Post Conditions: The user is allowed to start a new game.

Start New Game

Id: Use Case 3

Actors: Player

Preconditions:
Introduction screen is being displayed.

Flow of events:
1 User presses begin game button.
2 The system displays the game screen
3 «include» Cycle use case.
4 Use case ends.

Post Conditions: The user is playing a new game

Move Ship

Id: Use Case 4

Actors: Player

Preconditions:
Game is executing.

Flow of events:
1 User press the right or left arrow key.
2 The system moves the ship and displays the ship in its new position.
3 Use case ends.

Post Conditions: The ships position has changed.

Player firing

Id: Use Case 5

Actors: Player

Preconditions:
Game is executing and no other firing is in progress.

Flow of events:
1 User presses either the firing key.
2 The system creates a new upward moving missile at the ship's location.
3 Use case ends.

Post Conditions: A player firing is in progress.

Game Cycle

Id: Use Case 6

Actors: Time

Preconditions:
The Game is Executing.

Flow of events:
1 while ship and aliens are present (e.g., not destroyed) do
1.1 «include» Move Aliens Use Case.
1.2 «include» Move Missiles Use Case.
1.3 «include» Drop Bomb Use Case.
1.4 «include» Move Spaceship Use Case.
1.5 «include» Check Collisions Use Case.
3. Use case ends.

…further use cases would then provide further detail

Post Conditions:

End Game

Id: UC13

Actors: Time

Preconditions:

Flow of events:
1 If the score for the current game is > minimal score in the history list
1.1 «include» Record High Score use case..
2 Use case ends.

Post Conditions: