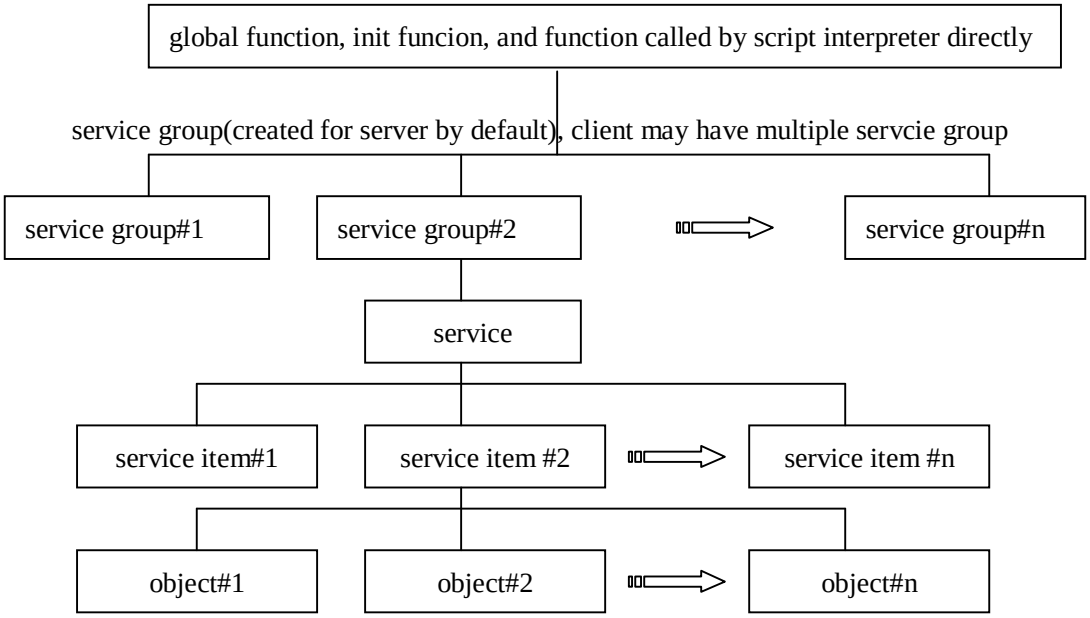
Common Language Extension Interface for script

Contents

1 Introduction

1.1 Hierarchy of objects and functions



1.2 Syntax of different script languages

Different script may have different syntax. The document defines attributes and functons of objects. But how to write applications depends on specific script language. As example:
for lua:
obj = Service:_New()
obj.p1 = 123
for python:
obj = Service._New()
obj.p1 = 123
for php:
$obj = $Service._New()
$obj->p1 = 123
and so on.

In static languages such as java or c#, application uses _Get and _set functions to access object attributes:

```
_GetInt(String Name);
_GetBool(String Name);
_GetDouble(String Name);
_GetStr(String Name);
Object _Get(String Name);
```

```
_Set(String Name,Object Para);
```

Examples in the document, mostly use python language. If you want use other languages, they should be adjusted.

For callback functions are used in applications:
for lua/python: callback directly uses script function object
for PHP: callback function are global, uses script function name
for JAVA/c#: callback function defined in extend class, and uses function name

1.3 Public variable type

for C++,lua,python,php,java, c#, the following types are common, and may be transfered between different languages.
simple types:

bool type(bool).
integer type(integer)
float type(double)
string type(string)

Tuple(tuple,(lua table[index start from 1],Python Tuple, php array,java Object[],c# Object[])). Tuple only supports one-dimensional map. If there are tuples included in tuple, the map may be incorrect.

map table :

from \ to	lua table	python tuple	php array	java Object[]	c# Object[]	Object Property	C/C++ ParaPkg Tuple
lua table		O	O	O	O	O	O
python tuple	O		O	O	O	O	O
php array	O	O		O	O	O	O
java Object[]	O	O	O		O	O	O
c# Object[]	O	O	O	O			O
Object Property	Struct table	Struct object	Strtuct class				
C++ ParaPkg Tuple	O	O	O	O	O		

Dictionary(dict,lua table,python Dict,php array,java corresponds to Hashtable,c# corresponds to Hashtable) may be accessed by C/C++ ParaPkg's function FromDict/ToDict

Agreement for return type:

Single return value uses simple type.
For multiple return value:
lua : return multiple value directly
Python : return tuple
php : return array
java : return Object[]
c# : return Object[]

If attribute is struct defined by C/C++, then use array or tuple to set its value, and struct object for get its value.

1.4 Gloal object

For each language, there are predefined global variables. By them, you can create other objects, or get predefined constants.

lua:
require "libstarcore"
libstarcore is global variable;

python:
import starpy
starp is global variable;

php:
$starcore = new StarCoreFactory()
starcore is global variable;

java: StarCoreFactory is single instance class.
StarCoreFactory starcore= StarCoreFactory.GetFactory();
starcore is global variable;

c#: StarCoreFactory is single instance class.
StarCoreFactory starcore= StarCoreFactory.GetFactory();
starcore is global variable;.

1.5 Notes for java/c#

java/c# are static languages, which should defined class before compile, and then create object.
Among objects defined in StarCore, besides the following three simple types:
StarTimeClass
StarRectClass
StarFontClass
Attributes of instance of other class should access by _Get. Input parameter is attribute name, and

return value is Object type. For convenient, also provides _GetInt,GetBool,GetDouble,_GetStr to return corresponding type of object, and provides _Getint,_Getbool,Getdouble function to return value of simple type, and provides _Toint,_Tobool,_Todouble function to change object to simple type conveniently.

Using _Set function to set attribute value: _Set(attribute name, value)

Pre-defined class in starcore:

StarBinBufClass
StarCommInterfaceClass
StarCoreFactory
StarFunctionParaClass
StarObjectClass
StarParaPkgClass
StarQueryRecordClass
StarServiceClass
StarServiceItemClass
StarSrvGroupClass
StarStructClass
StarSXmlClass

Of which, StarCoreFactory is class factory. Instance of other class should be created by its interface function other than new method. StarCoreFactory is single instance class, which can be obtained by static function GetFacroty, for example:

```
StarCoreFactory starcore= StarCoreFactory.GetFactory();
StarSrvGroupClass SrvGroup = starcore._GetSrvGroup(0);
```

For StarServiceItemClass's function _ClientToSync, StarCommInterfaceClass's function _MsgProc and _WebServerProc,object's system event function, object's event process function, _OnNameValueChange, _OnChange, you should define extend class to overwrite these functions. For example, StarCommInterfaceClass defines function _MsgProc, if you want to overwrite it, should do as follow:

```
class WebServer_CommInterface extends StarCommInterfaceClass{
    public void _MsgProc(int uMes,Object[] Msg){
        …
    }
    public WebServer_CommInterface(StarCommInterfaceClass srcobj){
        super(srcobj);
    }
}
```

Using the following format to create instance :

```
WebServer_CommInterface CommInterface = new WebServer_CommInterface (SrvGroup._NewCommInterface());
```

Another method:

```
StarCommInterfaceClass CommInterface = SrvGroup._NewCommInterface();
CommInterface = CommInterface._Assign( new StarCommInterfaceClass (){
      public void _MsgProc(int uMes,Object[] Msg){
           …
      }
)
```

CLE will find the defined functions by reflect mechanism.

For StarObjectClass, besides callback function listed above, you can define functions which may be called by other language, or attributes which can be access by other language.
These function and attributes should be declared with public. For attributes, their type should be boolean,int,double,String.
For function, parameter and return value type supported may be double, bool, integer, string, StarParaPkgClass,StarBinBufClass and StarObjectClass, and the first parameter must be StarObjectClass. When called, the first parameter is destination object attached by CLE. For example:

```
class MyObjectClass extends StarObjectClass{
    public int GetNumber( StarObjectClass self,int para )
    {
       System.out.println("Remote Call Number "+para);
       return para + 1;
    }
     public MyObjectClass(StarObjectClass srcobj){
          super(srcobj);
     }
}

StarObjectClass Obj = Service._New();
Obj = Obj._Assign(new StarObjectClass(){
    public int GetNumber( StarObjectClass self,int para )
    {
       System.out.println("Remote Call Number "+para);
       return para + 1;
    }
 }
)
```

2 Constant definition

Constant can be accessed through global object and service object

2.1 Sync status

SYNC_NOT
SYNC
SYNC_IN

2.2 Alarm value

FAULT_IND
FAULT_WARN
FAULT_NORMAL
FAULT_CRITICAL
FAULT_SYSTEM

2.3 Object alloc type

ALLOC_STATIC
ALLOC_GLOBAL
ALLOC_CLIENT
ALLOC_LOCAL

2.4 Object save type

SAVE_SAVE
SAVE_LOCAL
SAVE_GLOBAL
SAVE_NONE

2.5 Object active command

ACTIVE_ALONE
ACTIVE_FOLLOW
ACTIVE_ACTIVE
ACTIVE_DEACTIVE

2.6 Remotecall result

RCALL_OK
RCALL_COMMERROR
RCALL_OBJNOTEXIST
RCALL_FUNCNOTEXIST
RCALL_PARAERROR
RCALL_SYSERROR

RCALL_INVALIDUSR
RCALL_OVERTIME
RCALL_UNKNOWN

2.7 Remotecall source type

RCALLSRC_C
RCALLSRC_SCRIPT
RCALLSRC_WEBSERVICE

2.8 Attribute type

TYPE_BOOL
TYPE_INT8
TYPE_UINT8
TYPE_INT16
TYPE_UINT16
TYPE_INT32
TYPE_UINT32
TYPE_FLOAT
TYPE_LONG
TYPE_ULONG
TYPE_LONGHEX
TYPE_ULONGHEX
TYPE_VSTRING
TYPE_PTR
TYPE_MEMORY
TYPE_STRUCT
TYPE_COLOR
TYPE_RECT
TYPE_FONT
TYPE_TIME
TYPE_CHAR
TYPE_UUID
TYPE_STATICID
TYPE_CHARPTR
TYPE_PARAPKGPTR
TYPE_BINBUFPTR

TYPE_INT8PTR
TYPE_UINT8PTR
TYPE_INT16PTR
TYPE_UINT16PTR
TYPE_INT32PTR

TYPE_UINT32PTR
TYPE_FLOATPTR
TYPE_ULONGPTR
TYPE_LONGPTR
TYPE_STRUCTPTR
TYPE_COLORPTR
TYPE_RECTPTR
TYPE_FONTPTR
TYPE_TIMEPTR
TYPE_UUIDPTR
TYPE_VOID

TYPE_OBJPTR
TYPE_TABLE

2.9 UUID

INVALID_UUID

2.10 Virtual key code[win32]

Refer to function _GetKeyState.

2.11 Program run type

VS_SERVER
VS_CLIENT
VS_DEBUG
VS_TOOLS
VS_SERVER_SERVER
VS_SERVER_USER
VS_CLIENT_USER
VS_CLIENT_COOPERATOR
VS_CLIENT_CALLER

2.12 Module type

VSMODULE_SERVER_SERVER
VSMODULE_SERVER_USER
VSMODULE_CLIENT_USER
VSMODULE_CLIENT_COOPERATOR
VSMODULE_CLIENT_CALLER
VSMODULE_DEBUG

2.13 OS type

VSOS_WIN32
VSOS_LINUX
VSOS_ANDROID
VSOS_ANDROIDV7A

2.14 Ssytem object

System object is only one, should not create its instances.
System object can be get by **ServiceGroup._SysObject**. it defines the following events:
OnWndAdjust,OnWndResize,OnWndCanBeResize,OnEditSelect,OnSetFocus,OnWinMsg
Whether to create these events depends on application.

```
def SrvGroup_SysObject_OnWndCanBeResize(self, Event ) :
   print(Event)
   return True,1 // return 1 means forbid to change manager window size, and 0 means allow.
SrvGroup._SysObject.OnWndCanBeResize = SrvGroup_SysObject_OnWndCanBeResize
```

3 Struct definition

3.1 COLOR

Decimal number.

3.2 RECT

python: { 'Value':((INT)Left,(INT)Top,(INT)Right,(INT)Bottom), 'Type':'Rect'}
lua : { Value = { (INT)Left,(INT)Top,(INT)Right,(INT)Bottom }, Type = "Rect"}

```
java:
public class StarRectClass{
      int left;
      int top;
      int right;
      int bottom;
};

c#:
 public ref struct StarRectClass{
        Int32 left;
        Int32 top;
        Int32 right;
        Int32 bottom;
 };
```

3.3 FONT

python: {‘Value’:((INT)Height,(INT)Size,(BYTE)CharSet,(BYTE)Style,(String)Name), ‘Type’:’Font’}
lua: {Value = {(INT)Height,(INT)Size,(BYTE)CharSet,(BYTE)Style,(String)Name }, Type = “Font”}

```
java:
class StarFontClass{
    int Height;
    int Size;
    byte CharSet;
    byte Style;
    String Name;
};

c#:
 public ref struct StarFontClass{
      Int32 Height;
      Int32 Size;
      byte CharSet;
      byte Style;
      String^ Name;
 };
```

Height is negative.
CharSet = 134 is GB2312_CHARSET
Size may be 0, which represents default value.
Style may be 0, or the sum of
LOCALFONT_BOLD : 1
LOCALFONT_ITALIC : 2
LOCALFONT_UNDERLINE : 4
LOCALFONT_STRIKEOUT : 8
.
Name: font name.
for example
1. XX: {‘Value’:(-29,0,134,0,”XX_GB2312”), ‘Type’ : ‘Font’}

3.4 TIME

python: {‘Value’: (WORD wYear,WORD wMonth,WORD wDay,WORD wHour,WORD wMinute,WORD wSecond,WORD wMilliseconds), ‘Type’:‘Time’ }
lua: {Value = {WORD wYear,WORD wMonth,WORD wDay,WORD wHour,WORD wMinute,WORD wSecond,WORD wMilliseconds} ,Type=“Time” }
php: array {‘Value’ = {WORD wYear,WORD wMonth,WORD wDay,WORD wHour,WORD wMinute,WORD wSecond,WORD wMilliseconds} ,’Type’=“Time” }
java:StarTimeClass.

```
class StarTimeClass{
    short wYear;
    short wMonth;
```

```
        short wDayOfWeek;
        short wDay;
        short wHour;
        short wMinute;
        short wSecond;
        short wMilliseconds;
    };

    c#:
     public ref struct StarTimeClass
     {
            unsigned short wYear;
            unsigned short wMonth;
            unsigned short wDayOfWeek;
            unsigned short wDay;
            unsigned short wHour;
            unsigned short wMinute;
            unsigned short wSecond;
            unsigned short wMilliseconds;
     };
```

3.5 Active set

python: (number, number, number,…)
lua: { number, number, number,…}
lua uses table, python uses tuple, PHP uses array, JAVA/c# uses Object[].

3.6 SRP version

((BYTE)MainVersion,(BYTE)SubVersion,(SHORT)BuildVersion)
lua uses table,python uses tuple,PHP uses array,JAVA/c# uses Object[].

4 Object system event

Note: if handler is defined by C++, then script event functions will not be called.
Event:
lua -> table
python -> dict
php -> array
java/c# -> HashTable

Exmaple:

```
def Object_OnDestroy(self,Event) :
    Print( self, Event )
Return True, 0 #---The first return value is True, means the event has been processed, and followed by return values.
Object._OnDestroy = Object_OnDestroy    #Set object event handler
```

```
function Object:_OnDestroy (Event)
	print( self, Event )
	return true, 0 --- The first return value is true, means the event has been processed, and followed by return values.
end
```

```
public Object[] _OnRemoteSend(Hashtable Ev)
{
	return null;	or
	return new Object[]{true,XX};
}
```

4.1 _OnCreate(self, Event)

```
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self==False instance
//--OUT	none
```

```
//--IN	Event._DesObject =Object
//--IN	Event._ThisObject =true self ==false instance
//--OUT	none
```

4.2 _OnDestroy(self, Event)

```
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT	none
```

```
//--IN	Event._DesObject =Object
//--IN	Event._ThisObject =true self ==false instance
//--OUT	none
```

4.3 _OnCreateChild(self, Event)

```
//--IN Event['_Arg0'] = ChildObject
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT	none
```

```
//--IN	Event._Arg0 = ChildObject
//--IN	Event._DesObject =Object
//--IN	Event._ThisObject =true self ==false instance
//--OUT	none
```

4.4 _OnDestroyChild(self, Event)

```
//--IN Event['_Arg0'] = ChildObject
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._Arg0 = ChildObject
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.5 _OnActivating(self, Event)

```
//--IN Event['_Arg0'] = 0 normal activated 1 re-activated for load
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    number= 0 success, otherwise fail
```

```
//--IN      Event._Arg0 = 0    normal activated 1 re-activated for load
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    number= 0 success, otherwise fail
```

The event is triggered only when object is in sync status.

4.6 _OnDeactivating(self, Event)

```
//--IN Event['_Arg0'] = 0 normal deactivated 1 re-deactivated for load
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._Arg0 = 0    normal deactivated 1 re-deactivated for load
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.7 _OnActivate(self, Event)

```
//--IN Event['_Arg0'] = 0 normal activated 1 re-activated for load
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._Arg0 = 0    normal activated 1 re-activated for load
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT     none
```

The event is triggered only when object is in sync status.

4.8 _OnDeactivate(self, Event)

```
//--IN Event['_Arg0'] = 0 normal deactivated 1 re-deactivated for load
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT     none
```

```
//--IN      Event._Arg0 = 0    normal deactivated 1 re-deactivated for load
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT     none
```

4.9 _OnActiveChild(self, Event)

```
//--IN Event['_Arg0'] = ChildObject
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT     none
```

```
//--IN      Event._Arg0 = ChildObject
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT     none
```

4.10 _OnDeactiveChild(self, Event)

```
//--IN Event['_Arg0'] = ChildObject
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT     none
```

```
//--IN      Event._Arg0 = ChildObject
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT     none
```

4.11 _OnParentBeforeChange(self, Event)

```
//--IN Event[‘_ Arg0’] = NewParentObject
//--IN Event[‘_DesObject’] =Object
//--IN Event[‘_ThisObject’] =True self ==False instance
//--OUT    number== 0 permit change, otherwise not allow and is the error code.
The event is valid for object is under local control.
```

```
//--IN      Event._ Arg0 = NewParentObject
//--IN      Event._SrcObject = nil
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    number== 0 permit change, otherwise not allow and is the error code.
The event is valid for object is under local control.
```

4.12 _OnParentChange(self, Event)

```
//--IN Event[‘_DesObject’] =Object
//--IN Event[‘_ThisObject’] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.13 _OnStaticChange (self, Event)

```
//--IN Event[‘_ Arg0’] = AttributeNamex
//--IN Event[‘_DesObject’] =Object
//--IN Event[‘_ThisObject’] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._ Arg0 = AttributeName
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.14 _OnScriptChange(self, Event)

```
//--IN Event[‘_ Arg0’] = ScriptName
//--IN Event[‘_ Arg1’] = Operation 0 change 1 create
//--IN Event[‘_DesObject’] =Object
//--IN Event[‘_ThisObject’] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._ Arg0 = ScriptName
//--IN      Event._ Arg1 = Operation    0 change    1 create
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.15 _OnSyncGroupChange(self, Event)

```
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.16 _OnChildSyncGroupChange(self, Event)

```
//--IN Event['_Arg0'] = ChildObject
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._Arg0 = ChildObject
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.17 _OnActiveSetChange (self, Event)

```
//--IN Event['_Arg0'] = SysRootItem
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    none
```

```
//--IN      Event._Arg0 = SysRootItem
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.18 _OnLoadMask (self, Event)

```
//--IN Event['_DesObject'] =Object
```

```
//--IN Event['_ThisObject'] =True self ==False instance
//--OUT    True,AttributeName1,..      attribute name list which should be not changed by load
```

```
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    true,AttributeName1,..      attribute name list which should be not changed by load
```

4.19 _OnLoadFinish(self, Event)

```
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self==False instance
//--OUT    none
```

```
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--OUT    none
```

4.20 _OnTicket(self, Event) only triggered to self

```
//--IN Event['_Arg0'] = Ticket
//--IN Event['_DesObject'] =Object
//--OUT    none
```

```
//--IN      Event._Arg0 = Ticket
//--IN      Event._DesObject =Object
//--OUT    none
```

4.21 _OnFrameTicket(self, Event) only triggered to self

```
//--IN Event['_Arg0'] = Ticket
//--IN Event['_Arg1'] = FrameTimer , maintained by server
//--IN Event['_DesObject'] =Object
//--OUT    none
```

```
//--IN      Event._Arg0 = Ticket
//--IN      Event._Arg1 = FrameTimer, maintained by server
//--IN      Event._DesObject = Object
//--OUT    none
```

4.22 _OnIdle(self, Event) : only triggered to self

```
//--IN Event['_Arg0'] = Ticket
//--IN Event['_DesObject'] =Object
//--OUT    number== 0 wait next cycle , otherwise continue create the s=event
```

```
//--IN      Event._Arg0 = Ticket
//--IN      Event._DesObject = Object
//--OUT     number == 0 wait next cycle , otherwise continue create the s=event
```

The event is dispatched to current active service.

4.23 _OnAppActive(self, Event) only triggered to self

```
//--IN Event['_DesObject'] =Object
//--OUT     none
```

```
//--IN      Event._DesObject = Object
//--OUT     none
```

The event is dispatched to current active service.

4.24 _OnAppDeactive(self, Event) only triggered to self

```
//--IN Event['_DesObject'] =Object
//--OUT     none
```

```
//--IN      Event._DesObject = Object
//--OUT     none
```

The event is dispatched to current active service.

4.25 _OnServiceActive(self, Event) only triggered to self

```
//--IN Event['_DesObject'] =Object
//--OUT     none
```

```
//--IN      Event._DesObject = Object
//--OUT     none
```

The event is dispatched to current active service.

4.26 _OnServiceDeactive(self, Event) only triggered to self

```
//--IN Event['_DesObject'] =Object
//--OUT     none
```

```
//--IN      Event._DesObject = Object
//--OUT     none
```

The event is dispatched to current active service.

4.27 _OnRemoteSend(self, Event)

```
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self==False instance
//--IN Event['_Arg0'] = ParaPkg(message)
//--OUT    none
```

```
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self==false instance
//--IN      Event._Arg0 = ParaPkg(message)
//--OUT    none
```

4.28 _OnCall(Event)

```
//--IN Event['_DesObject'] =Object
//--IN Event['_ThisObject'] =True self==False instance
//--IN      Event['_Arg0'] = FunctionPara
//--IN      Event['_Arg1'] = FunctionName
//--IN      Event['_Arg2'] = FunctionID
//--OUT    True,RetValue
```

```
//--IN      Event._DesObject =Object
//--IN      Event._ThisObject =true self ==false instance
//--IN      Event._Arg0 = FunctionPara
//--IN      Event._Arg1 = FunctionName
//--IN      Event._Arg2 = FunctionID
//--OUT    true,RetValue
```

5 Interface function

5.1 Init CLE

1. **Integer = _InitCore(bool ServerFlag,bool ShowMenuFlag,bool howClientWndFlag,bool SRPPrintFlag,DebugInterface,DebugPort,ClientInterface,ClientPort);**

```
java:  int  _InitCore(boolean  ServerFlag,boolean  ShowMenuFlag,boolean ShowOutWndFlag,boolean  SRPPrintFlag,String  DebugInterface,int  DebugPortNumber,String ClientInterface,int ClientPortNumber);
```

ServerFlag True : run as server (default), False : run as client

ShowMenuFlag, whether to show menu

ShowClientWndFlag, Whether to show client window, note: it is reserved, no effect.

SRPPrintFlag, whether to print CLE internal information
DebugInterface, interface for debugserver, may be ““
DebugPort, port number for debugserver.
ClientInterface, interface for client,may be ““.
ClientPort, port number for client
return 0 for success,-1for failure;1for have been inited before.

[note: parameter ShowMenuFlag, ShowOutWndFlag are reserved. Examples:

```
import starpy
starpy._InitCore
(ServerFlag,ShowMenuFlag,ShowOutWndFlag,SRPPrintFlag,”“,DebugPort,”“,ClientPort)
```

```
require “libstarcore”
libstarcore._InitCore
(ServerFlag,ShowMenuFlag,ShowOutWndFlag,SRPPrintFlag,”“,DebugPort,”“,ClientPort)
```

]

2. SrvGroup = _InitSimpleEx(ClientPort,WebPort,DependService,…);

java: StarSrvGroupClass _InitSimpleEx(int ClientPortNumber,int WebPortNumber,String...DependService);

A simplified init method.
Concurrent connections of Webclient is 100, and max upload size is 2MB.
[

```
import starpy
SrvGroup= starpy._InitSimpleEx(0,0)
```

```
require “libstarcore”
SrvGroup=libstarcore._InitSimpleEx(0,0)
```

]

3. Service = _InitSimple(ServiceName,Pass,ClientPort,WebPort,DependService,…);
Service=_InitSimple1(ServiceName,ServiceID,Pass,ClientPort,WebPort,DependService,…);

java: StarServiceClass _InitSimple(String ServiceName,String ServicePass,int ClientPortNumber,int WebPortNumber,String...DependService);
java: StarServiceClass _InitSimple1(String ServiceName,String ServiceID,String ServicePass,int ClientPortNumber,int WebPortNumber,String...DependService);

A simplified init method. Pass is the password of root user.
Concurrent connections of Webclient is 100, and max upload size is 2MB.
[

```
import starpy
Service= starpy._InitSimple(“test”,”123”,0,0)
```

```
require “libstarcore”
Service=libstarcore._InitSimple(“test”,”123”,0,0)
```

]

5.2 Callback and control functions

These functions should be called in thread which inits the CLE. If is called in other threads, should acquire lock using function _SRPLock first, and release lock using function _SRPUnLock after finished.

0. **_ModuleExit** () : exit function

```
java: void _ModuleExit();
```

Before exit the progam, the function should be called to clear resources occupied by CLE.

1. **_ModuleClear** ()

```
java: void _ModuleClear();
```

The function deletes all CLE objects. After the function is called, application can re-init service or load new service.

2. **_RegMsgCallBack** (CallBack):

```
java: void _RegMsgCallBack(StarCallBackClass Obj,String CallBackName);
```

register callback function to handle global event from CLE.

For static script language, such as c#, java, the prototype of the function is **_RegMsgCallBack** (StarCallBackClass Obj,CallBack)

```
def CallBack( ServiceGroupID, uMes, wParam, lParam ) :
  …
return True/False,arg1,…        The first argument is True, means the following args is valid.
```

```
function CallBack( ServiceGroupID, uMes, wParam, lParam )
…
return true/false, arg11,…        The first argument is true, means the following args is valid.
```

java/c#: should extend StarCallBackClass, In it define callback, and register with callbackfunction name.

```
Object[] CallBack( (int)ServiceGroupID, (int)uMes, Object, Object )
…
return new Object[] {true/false, arg11,…}        The first argument is true, means the following args is valid.
```

example:

```
class MyStarCallBackClass extends StarCallBackClass{
        public Object[] CallBack( int ServiceGroupID, int uMes, Object wParam, Object lParam )
        {
                return null;
        }
        MyStarCallBackClass(StarCoreFactory
starcore){super(starcore);starcore._RegMsgCallBack(this,"CallBack");}
}

        StarCoreFactory starcore= StarCoreFactory.GetFactory();
        MyStarCallBackClass CallBack = new MyStarCallBackClass(starcore);
```

uMes :
MSG_VSDISPMSG :
MSG_VSDISPLUAMSG :
MSG_DISPMSG :
MSG_DISPLUAMSG: display information, wParam is content string, lParam is warining level.
Return value is ignore.

MSG_MESSAGEBOX : display messagebox, wParam is the caption, lParam is content
Return value is ignore.

MSG_EXIT : request to exit program, wParam is ErrorInfo,lParam is invalid,
Return value is ignore.

MSG_GETWNDHANDLE : get client window handle,wParam is invalid,lParamis invalid,
Return value:True,handle(uint)

MSG_SETWNDSIZE : set client window size,wParam is width,lParam is height,
Return value is ignore.
MSG_GETWNDSIZE : get client window size,wParam is invalid,lParamis invalid,
return value:True,width(uint) ,height(uint)

MSG_CLEARWND :clear client window,wParam is invalid,lParamis invalid,
Return value is ignore.
MSG_HIDEWND :hide client window,wParam is invalid,lParamis invalid,
Return value is ignore.
MSG_SHOWWND :display client window,wParam is invalid,lParamis invalid,
Return value is ignore.
MSG_SETWNDBK:set client window background color,wParam is color,lParamis invalid,
Return value is ignore.
MSG_SETFOCUS: set client window focus,wParam is window handle,lParam 0/1, which 0 represents notification, and 1 means the message should be processed.
Return value is ignore.

MSG_ISAPPACTIVE : whether the program is active or not,wParam is invalid,lParamis invalid,
return value:True,True/False.
MSG_SETIDLEACTIVE :wParam is True/False,lParamis invalid.
Return value is ignore.

MSG_SETINFOCOLOR :set text color,wParam is (NormalTextColor, ExplaneColor, ObjectNameColor, AttributeTypeColor, NumberColor, ErrorColor),lParamis invalid,
For java/c#,wPara is set as int array.
Return value is ignore.
MSG_SETINFOBK: set text background color,wParam is color,lParamis invalid,
Return value is ignore.

MSG_KILLFOCUS: kill client window focus,wParam is window handle. lParam 0/1 ,0 means notification, and 1 should be processed.
Return value is ignore.

MSG_ONBEFORESTOPSERVICE : before stop service, wParam is string of service id.,lParamis invalid,
Return value is ignore.
MSG_ONSTOPSERVICE: service is stopped, wParam is string of service id,lParamis invalid,
Return value is ignore.
MSG_ONACTIVESERVICE:service is activated,wParam is string of service id,lParamis invalid,
Return value is ignore. [note:valid for client service group]
MSG_SAVESERVICE: whether to save service,wParam is invalid,lParam is invalid
return value:True,True/False.[note:valid for server service group]

MSG_HYPERLINK: active hyper connection. wParam is URL,lParam CreateNewWindow,
Return value is ignore.
MSG_APPEVENT: trigger app event, wParam EventID lParam VS_CHAR *EventInfo
Return value is ignore.

MSG_SERVERTERM: server close connection,wParam is invalid,lParamis invalid,

Return value is ignore. [note:valid for client service group]

manager window message:

MSG_ISMANAGERVISIBLE :whether manage window is visible. wParam is invalid,lParamis invalid,
return value:True,True/False.
MSG_HIDEMANAGER :hide manager window,wParam is invalid,lParamis invalid,
Return value is ignore.
MSG_SHOWMANAGER :show manager window,wParam is invalid,lParamis invalid,
Return value is ignore.
MSG_SETMANAGERCAPTION:set manager window caption,wParam is caption,lParamis invalid,
Return value is ignore.
MSG_GETMANAGERSIZE :get manager window size,wParam is invalid,lParamis invalid,
return value:True,width(uint) ,height(uint)
MSG_GETMANAGERHANDLE :get manager window handle,wParam is invalid,lParamis invalid,
return value:True,handle(uint)
MSG_SHOWMANAGERSTATUSMENU:whether to show manager window menu or status bar,wParamis menu status True/False,lParam is status of status bar, True/False,
Return value is ignore.
MSG_SETMANAGERSTYLE :set manager window parameters,wParam is (SystemMenuFlag, MinimizeFlag, MaximizeFlag, ShowBorderFlag, SizeableFlag),lParamis invalid,
for java/c#,wParam is set as bool array
Return value is ignore.
MSG_MOVEMANAGER :move window,wParam is (X,Y,nWidth,nHeight),lParam RepaintFlag,
Return value is ignore.
for java/c#,wPara is set as int array
MSG_GETMANAGERPOS :ge window position,wParam is invalid,lParamis invalid,
return value:true,X,Y,nWidth,nHeight
MSG_SETMANAGERSTATUS: wParam Status lParam 0 //--0 normal 1 minimize other maximize
Return value is ignore.
MSG_REDIRECTTOURLREQUEST:wParam is Url string, lParam is WorkDirectory string.
Return: false(allow),true(forbid)
MSG_REDIRECTTOURLINFO: wParam is Url string, lParam is ParaPkg. return value is ignore
MSG_GETURLREQUEST:wParam is invalid,lParamis invalid,return value:true,url string
MSG_SETPROGRAMTYPE:wParam Type[VS_SERVER_SERVER/VS_SERVER_USER], lParam 0, return value: false(permit),true(forbid)
MSG_ISWINDOWLESSSITE: wParam 0 lParam 0,return: false(is),true(not)

3. **bool = _SRPDispatch** (bool WaitFalg) : message dispatch function

```
java: boolean _SRPDispatch(boolean WaitFalg);
```

One message will be processed for each call. If return value is True, then there is a message being processed, otherwise the message queue is empty.

4. **_SRPIdle** () : Notify CLE to create IDLE system event.

```
java: boolean _SRPIdle();
```

5. **_MsgLoop** (bool ExitFunc = None / LoopTimes = times) : message loop function.

```
java: boolean _MsgLoop(StarCallBackClass Obj,Object ExitFuncName_LoopTimes );
        public boolean ExitFunc()
        {
            return true/false;
        }
```

For static language, such as c#, java, its prototype is **_MsgLoop** (StarCallBackClass Obj,bool ExitFunc = None / LoopTimes = times) :

It the ExitFunc returns False, then the loop is continue, or else is break.

If Looptimes is not 0, the loop is ended after the times. Each loop will process all the message in the queue.

Please refer to C++ code:

```
while(1){
        if(PeekMessage(&msg, NULL, 0, 0, PM_NOREMOVE)){
                if(!GetMessage(&msg, NULL, 0, 0))
                        break;
                TranslateMessage(&msg);
                        DispatchMessage(&msg);
        }else{
                    if( SRPDispatch( VS_FALSE ) == VS_FALSE )
                        SRPIdle procedure
            }
      }
```

```
function ExitFunc()
      return true/false;
end
libstarcore._MsgLoop(ExitFunc)
```

```
def ExitFunc()
     return true/false;
starpy._MsgLoop(ExitFunc)
```

```
function ExitFunc()    //php
{
      return true/false;
}
```

```
class MyStarCallBackClass extends StarCallBackClass{
         MyStarCallBackClass(StarCoreFactory starcore){super(starcore);}
         public boolean ExitProc()
        {
          if(StarCore._KeyPress() == 27){
               return true;
          }
          return false;
        }
}


         StarCoreFactory starcore= StarCoreFactory.GetFactory();
         MyStarCallBackClass CallBack = new MyStarCallBackClass(starcore);

        starcore._MsgLoop(CallBack,"ExitProc");
```

```
class MyStarCallBackClass : StarCallBackClass{        c#
         public MyStarCallBackClass(StarCoreFactory starcore):base(starcore){}
        public Boolean ExitProc()
        {
          if(StarCore._KeyPress() == 27){
               return true;
          }
          return false;
        }
}


         StarCoreFactory starcore= StarCoreFactory.GetFactory();
         MyStarCallBackClass CallBack = new MyStarCallBackClass(starcore);

         starcore._MsgLoop(CallBack,"ExitProc");
```

6. **bool = _WinMsgDispatch** () : dispatch win32 message

```
java: boolean _WinMsgDispatch();
```

Returns FALSE means WM_QUIT message is detected.

7. **code = _KeyPress** () : return key code, for example: ESC=27

```
java: int _KeyPress();
```

8. **_SRPLock** ()/_SRPUnLock, lock and unlock

```
java: void _SRPLock();
java: void _SRPUnLock();
```

5.3 Global function

1. **ServiceGroup = _GetSrvGroup(ServiceGroupID or Service name);**

```
java: StarSrvGroupClass _GetSrvGroup(Object ServiceName_GroupID);
```

Get service group object. for server service group, ServiceGroupID is fixed set to 0, and for client service group, the function should be called after the service group has been created .

2. **ServiceGroup = _CreateSrvGroup (ServiceGroupID,int Type);**

```
java: StarSrvGroupClass _CreateSrvGroup(int ServiceGroupID,int Type );
```

Create service group and return the service group object. Type should be set to VS_CLIENT_USER, which means to create client service group.
if ServiceGroupID==0, the id will be allocated by CLE.
if the service group has existed, None/null will be returned.

_DeleteSrvGroup (ServiceGroupID);

```
java: void _DeleteSrvGroup(int ServiceGroupID);
```

Delete service group. Group 0 can not be deleted.

3. **_SrvGroupInfo ();**

```
java: void _SrvGroupInfo();
```

print current service infomation

4. **ServiceGroupID = _FirstSrvGroup ();**

```
java: int _FirstSrvGroup();
```

Return ID of first service group. Return 0xFFFFFFFF means no more service group.

5. **ServiceGroupID = _NextSrvGroup ();**

```
java: int _NextSrvGroup();
```

Return ID of next service group. Return 0xFFFFFFFF means no more service group

5.3.1 system type and time

1. **Type=_GetOsType();**

```
java: int _GetOsType();
```

System type is 0 Win32, 1 Linux.

2. **_Time();**

```
java: StarTimeClass _Time();
```

Get current system time.

5.3.2 registry query function

1. **Str = _GetRegStr (SubKey, ValueName,** char *DefaultValue**);**

```
java: String _GetRegStr(String SubKey,String ValueName,String DefaultValue);
```

The format of subkey is as “Software\\SRPLab\\SRPServer"
When the SubKey does not exist, DefaultValue is returned.

2. **Int = _GetRegInt (SubKey, ValueName,** int DefaultValue**);**
   java: int _GetRegInt(String SubKey,String ValueName,int DefaultValue);
   The format of subkey is as “Software\\SRPLab\\SRPServer"
   When the SubKey does not exist, DefaultValue is returned.

5.3.3 32bit operation

1. **int = _luashl32(a,b); a << b**
   java: int _shl32(int a,int b);
2. **int = _luashr32(a,b); a >> b**
   java: int _shr32(int a,int b);
3. **int = _luaand32(a,b); a & b**
   java: int _and32(int a,int b);
4. **int = _luaor32(a,b); a | b**
   java: int _or32(int a,int b);
5. **int = _luaxor32(a,b); a ^ b**
   java: int _xor32(int a,int b);

5.3.4 file find function

1. **Exist,Handle,FileName,IsDir(true/false) = _FindFirstFile (FileName);**
   java: Object[] _FindFirstFile(String FileName);
2. **Exist, FileName,IsDir(true/false)= _FindNextFile (Handle);**
   java: Object[] _FindNextFile(Object Handle);
3. **_FindClose (Handle)**
   java: void _FindClose(Object Handle);
   _FindClose should be called after find process.

5.3.5 byte order change

1. **_htonl(long)**
   java: int _htonl(int a);
2. **_htons(short)**
   java: short _htons(short a);
3. **_ntohl(long)**
   java: int _ntohl(int a);
4. **_ntohs(short)**
   java: short _ntohs(short a);

5.3.6 Shell execute command[Windows,do not support C#]

1. **_ShellExecute (“operation”,FileName)**

java: void _ShellExecute(String Op,String FileName);

Open ie uses _ShellExecute("open",URL)
C# does not support this function.

5.3.7 virtual key status[Windows, ,do not support C#]

1. bool = _GetKeyState (Key) :

java: boolean _GetKeyState(int Key);

If the key is pressed, then True is returned. Key defines as follow:

VS_LBUTTON	0x01
VS_RBUTTON	0x02
VS_MBUTTON	0x04
VS_ESCAPE	0x1B
VS_BACKSPACE	0x08
VS_TAB	0x09
VS_ENTER	0x0D
VS_SPACE	0x20
VS_SHIFT	0x10
VS_CTRL	0x11
VS_ALT	0x12
VS_LWIN	0x5B
VS_RWIN	0x5C
VS_APPS	0x5D
VS_PAUSE	0x13
VS_CAPSLOCK	0x14
VS_NUMLOCK	0x90
VS_SCROLLLOCK	0x91
VS_PGUP	0x21
VS_PGDN	0x22
VS_HOME	0x24
VS_END	0x23
VS_INSERT	0x2D
VS_DELETE	0x2E
VS_LEFT	0x25
VS_UP	0x26
VS_RIGHT	0x27
VS_DOWN	0x28
VS_0	0x30
VS_1	0x31
VS_2	0x32
VS_3	0x33
VS_4	0x34
VS_5	0x35
VS_6	0x36
VS_7	0x37
VS_8	0x38
VS_9	0x39
VS_A	0x41
VS_B	0x42
VS_C	0x43
VS_D	0x44
VS_E	0x45
VS_F	0x46
VS_G	0x47
VS_H	0x48
VS_I	0x49

VS_J 0x4A
VS_K 0x4B
VS_L 0x4C
VS_M 0x4D
VS_N 0x4E
VS_O 0x4F
VS_P 0x50
VS_Q 0x51
VS_R 0x52
VS_S 0x53
VS_T 0x54
VS_U 0x55
VS_V 0x56
VS_W 0x57
VS_X 0x58
VS_Y 0x59
VS_Z 0x5A

VS_GRAVE 0xC0
VS_MINUS 0xBD
VS_EQUALS 0xBB
VS_BACKSLASH 0xDC
VS_LBRACKET 0xDB
VS_RBRACKET 0xDD
VS_SEMICOLON 0xBA
VS_APOSTROPHE 0xDE
VS_COMMA 0xBC
VS_PERIOD 0xBE
VS_SLASH 0xBF

VS_NUMPAD0 0x60
VS_NUMPAD1 0x61
VS_NUMPAD2 0x62
VS_NUMPAD3 0x63
VS_NUMPAD4 0x64
VS_NUMPAD5 0x65
VS_NUMPAD6 0x66
VS_NUMPAD7 0x67
VS_NUMPAD8 0x68
VS_NUMPAD9 0x69

VS_MULTIPLY 0x6A
VS_DIVIDE 0x6F
VS_ADD 0x6B
VS_SUBTRACT 0x6D
VS_DECIMAL 0x6E

VS_F1 0x70
VS_F2 0x71
VS_F3 0x72
VS_F4 0x73
VS_F5 0x74
VS_F6 0x75
VS_F7 0x76
VS_F8 0x77
VS_F9 0x78
VS_F10 0x79
VS_F11 0x7A
VS_F12 0x7B

5.3.8 intger compare

1. **int = _IntComp(a,b); -1: a<b 0 a==b 1 a>b**

   java: int _IntComp(int a,int b);

1. **int = _UIntComp(a,b); -1: a<b 0 a==b 1 a>b**

   java: int _UIntComp(int a,int b);

5.3.9 get UUID

1. **Uuid=_UuidCreate()**
   java: String _UuidCreate();

5.3.10 get current Url

1. **Url=_GetUrl ()**
   java: String _GetUrl();
   Example: for http://127.0.0.1/test/test.html, the return value is http:\\127.0.0.1\test
2. **Url=_GetRootUrl ()**
   java: String _GetRootUrl();
   Example: for http://127.0.0.1/test/test.html, the return value is http:\\127.0.0.1

5.3.11 set program run type [valid at server]

If type is different from current, current service which has been loaded will be unloaded.

1. **_SetProgramType (Type) [Reserved]**
   java: void _SetProgramType(int Type);
   Type takes value from: VS_SERVER_SERVER/VS_SERVER_USER
2. **Type= _GetProgramType()**
   java: int _GetProgramType();

5.3.12 find char position

1. **Int=_strchr (string,char)**
   java: int _strchr(String Str,String Ch);
2. **Int=_strrchr(string,char)**
   java: int _strrchr(String Str,String Ch);

For lua, Return index starts from 1, and -1 means not exist.

For other script, Return index starts from 0, and -1 means not exist.
Do not distinguish '/' and '\\'

5.3.13 ID to MD5 string

1. **MD5Str =_IDToMD5 (IDStr)**
   java: String _IDToMD5(String IDStr);

5.3.14 get system path

1. **Path =_GetSysPath()**
   java: String _GetSysPath();

On Windows, the return is X:\windows\system32

On Linux, the return is /usr/lib

5.3.15 set log file

1. **_SetLogFile (FileName,bool LogAll)**

```
java: void _SetLogFile(String FileName,boolean LogAll);
```

If FileName is ““, current log will be canceled.

LogAll:true, log all print information, otherwise only log warning and error information.

5.3.16 authorize

1. **string= _GetSystemRegCode()**

```
java: String _GetSystemRegCode();
```

get local serial number

2. **bool=_SetRegisterCode(CodeStr,Single)**

```
java: boolean _SetRegisterCode(String CodeString,boolean Single);
```

The function should be called after service is created. Authorization code may be obtained from http://www.srplab.com

Single=True, for single computer service, =False for service.

3. **bool= _IsRegistered ()**

```
java: boolean _IsRegistered();
```

Whether CLE is registered.

4. **bool _PreAuthorize(string ServiceName,string ServiceID,string CodeStr,bool Single)**

```
java: boolean _PreAuthorize(String ServiceName,String ServiceID,String CodeStr,boolean Single);
```

Called before creating service, authorizes for service to be dynamicly imported.

For example:

```
require "libstarcore"
libstarcore._PreAuthorize("AddFunctionService","CD752291-5874-4f82-B477-F136070A1FCE","XXXX",false);
Service = libstarcore._InitSimple1("AddFunctionService","CD752291-5874-4f82-B477-F136070A1FCE","123",0,0,"ChildAddFunction.py?script=python");
 libstarcore._SetRegisterCode(XXXX,false);
```

5.3.17 set locale of cle(android)

1. **void _SetLocale(String Lang);**

```
void _SetLocale(String Lang);
```

2. **string _GetLocale();**

```
String _GetLocale();
```

5.3.18 find window handle[win32]

3. **int hWnd= _FindWindow(String Caption);**

```
int _FindWindow(String Caption);
```

find window handle.

5.4 Service group attribute

4. **SysObject=ServiceGroup._SysObject**
   get system defined object
5. **Int=ServiceGroup. _EnvStartType**
   service start type
   Return value lists bellow :
   0 start from directory contains multiple files.
   1 start from single file package.
6. **ParaPkg=ServiceGroup. _EnvPara**
   current service parameter
7. **ParaPkg =ServiceGroup. _EnvInputPara**
   current service input parameter
8. **Url =ServiceGroup. _EnvParentUrl**
   parent service Url
9. **SysDocClass=ServiceGroup. _SysDocClass**
   get system doc class
10. **ServiceGroup. _ProgramType**
    refer to macro definitions of program run type.
11. **ServiceGroup. _IsRootService**
    If the service is dynamic service which is loaded by import function, the return value is False, or else is true.

5.5 Service group function

5.5.1 public functions

12. **Service= _GetService(username, userpassword) _GetServiceEx(ServiceName,username, userpassword);**
    java: StarServiceClass _GetService(String username, String userpassword)
    java: StarServiceClass _GetServiceEx(String ServiceName,String username, String userpassword)
    Get current service object, the input parameter is user name and pasword.
    User name and password is ignored at client service group.
13. **_MessageBox(Caption,Info) :**
    java: void _MessageBox(String Caption,String Info)
    Display message box, it is processed by application.
14. **_Print(...) /_PrintError(AlarmLevel,string):**
    java: void _Print(String Args)
    java: void _PrintError(int AlarmLevel,String Args)
    For static language, only support _Print(String)/_PrintError(AlarmLevel,string).
15. **_ID();**

java: int _ID()

Returns service group id

16. **_IsServer();**

java: boolean _IsServer()

Returns True, means is server service group.

17. **_IsClient();**

java: boolean _IsClient()

Returns True, means is client service group.

18. **_IsDebug();**

java: boolean _IsDebug()

Returns True, means is debug service group.

19. **_IsServerClient();**

java: boolean _IsServerClient()

Returns True, means is server client service group.

20. **_IsObject(Object);**

java: boolean _IsObject(Object Args)

Returns True or False

21. **_IsParaPkg (ParaPkg);**

java: boolean _IsParaPkg(Object Args)

Returns True or False

22. **_IsQueryRecord (QueryRecord);**

java: boolean _IsQueryRecord(Object Args)

Returns True or False

23. **_IsBinBuf (BinBuf);**

java: boolean _IsBinBuf(Object Args)

Returns True or False

24. **_IsSXml(SXml);**

java: boolean _IsSXml(Object Args)

Returns True or False

25. **_IsFunctionPara(FunctionPara);**

java: boolean _IsFunctionPara(Object Args)

Returns True or False

26. **_IsCommInterface(CommInterface);**

java: boolean _IsCommInterface(Object Args)

Returns True or False

27. **_TickCount ();**

java: int _TickCount()

If high precision is supported, then it will return high precision timer.

28. **_MD5(Str);**

java: String _MD5(String Args)

Returns MD5 of the string.

29. **Bool = _SetDataServerAddr (DirectConnect,DataServerInterface, DataServerName, DataServerPort, LocalDataServerInterface,LocalDataServerPort);**

java: boolean _SetDataServerAddr(boolean DirectConnect,String DataServerInterface, String

DataServerName, int DataServerPort, String LocalDataServerInterface,int LocalDataServerPort)

valid at server service group;

DirectConnect: ==True, when DataServerPort is valid, client will connect to it directly, otherwise, will be relayed by server.

DataServerInterface: DataServer Interface, may be ““

DataServerName: DataServer Url.

DataServerPort: DataServer port number

LocalDataServerInterface: Local DataServer Interface, may be”“

LocalDataServerPort: Local DataServer port, 0 is invalid.

30. _SetServerPara (MaxClientNumber, MaxDataServerConnectionNumber,DataServerOverTime);

java: void _SetServerPara(int MaxClientNumber,int MaxDataServerConnectionNumber,int DataServerOverTime)

Set server parameter

31. _QuyeryStatistic(ClientID);

java: Object[] _QuyeryStatistic(int ClientID)

Get statistic info of CLE when ClientID is 0, or else returns statistic of special client. Return value is

int ClientConnectionNumber;
int DebugConnectionNumber;
int ServerConnectionNumber;
int DataConnectionNumber;
int ReceiveMsgItemNumber;
int ReceiveMsgItemBytes;
int SendMsgItemNumber;
int SendMsgItemBytes;
int SysSendQueueOccupyRate;
int ObjSendQueueOccupyRate;

int PeerDelayTicket;

32. int = _Hash(String);

java: int _Hash(String Args)

Get string hash value.

33. _GetModulePath ();

java: String _GetModulePath()

Get module path.

34. _GetServicePath ();

java: String _GetServicePath()

Get service default path

35. _SetServicePath (Path);

java: void _SetServicePath(String Args)

Set service default path

36. **_GetCurrentPath ();**

java: String _GetCurrentPath()

Get current path.

37. **_SetCurrentPath (Path);**

java: void _SetCurrentPath(String Args)

Set current path.

38. **Bool= _ServicePathIsSet();**

java: boolean _ServicePathIsSet()

Service default path is set or not.

39. **_GetSRPTempPath ();**

java: String _GetSRPTempPath()

Get temporary path

40. **_GetSRPConfigPath ();**

java: String _GetSRPConfigPath()

Get path of config file.

5.5.2 service sync function(valid at client)

1. **Bool = _IsInSync ();**

java: boolean _IsInSync()

Returns Ture means the service is in sync procedure.

2. **Bool = _IsServiceSync ();**

java: boolean _IsServiceSync()

Returns Ture means the service is in sync status.

3. **Bool = _WaitServiceSync(WaitTimeMs);**

java: boolean _WaitServiceSync(int WaitTimeMs)

Wait service becomes to sync. If success, return value is True . WaitTimeMs is the maximum time (ms) to wait. 0 means wait forever.

5.5.3 service manager function[called at server]

1. **Bool = ImportServiceEx (ServiceID,LoadRunModule=True);**

java: boolean _ImportServiceEx(String ServiceID,boolean LoadRunModule)

Import a service,ServiceID is UUID of service,for example :"66efc37a-a16c-442b-bc2a-3df5b03b3294". Server should not load or create any service before the function is called. ServiceID should be registered at registry under key SOFTWARE\SRPLab\SRPServer.

2. **Bool = _ImportServiceWithPath (ServicePath,ServiceName,LoadRunModule=True);**

java: boolean _ImportServiceWithPath(String ServicePath,String ServiceName,boolean LoadRunModule)

Import a service, input parameters are service path and service name. Server should not load or create any service before the function is called. Under service path, directory which name

is same with the service name must exist, where service files are located.

ServicePath is local path or network path, such as:http://www.XXX.com. If it equals to "http://srplab", then service is located at default srplab website.

If service name contains ".", then it is dynamic service, the function is same as ImportDynaService.

If service name starts with char "@", then the service file locates at local disk.

3. **Bool = _ImportService (ServiceName,LoadRunModule=True);**

   java: boolean _ImportService(String ServiceName,boolean LoadRunModule)

   Import a service, input parameter is service name. Server should not load or create any service before the function is called. ServiceID and ServiceName must be registeded at registry under key SOFTWARE\SRPLab\SRPServer.

If service name contains ".", then it is dynamic service, the function is same as ImportDynaService.

If service name starts with char "@", then the service file locates at local disk.

4. **servicename = _ImportDynaService (Url);**

   java: String _ImportDynaService(String Url)

   Import dynamic service, which may be local file or network file. Return value is service name.

   If Url starts with char "@", then the service file locates at local disk.

5. **Service= _CreateService/_CreateServiceEx (ServicePath,ServiceName, RootPass, FrameInterval, NetPkgSize, UploadPkgSize, DownloadPkgSize , DataUpPkgSize, DataDownPkgSize ,ServiceID="");**

   java: StarServiceClass _CreateService(String ServicePath,String ServiceName,String RootPass,int FrameInterval,int NetPkgSize,int UploadPkgSize,int DownloadPkgSize,int DataUpPkgSize,int DataDownPkgSize,String ServiceID)

   java: StarServiceClass _CreateServiceEx(String ServicePath,String ServiceName,String RootPass,int FrameInterval,int NetPkgSize,int UploadPkgSize,int DownloadPkgSize,int DataUpPkgSize,int DataDownPkgSize,String ServiceID)

   Server should not load or create any service before the function is called. RootPass is password of root user. FrameInterval is interval(10ms) between frame of the service. NetPkgSize is size of network package, which default is 10240(bytes); UploadPkgSize is size of data uploaded per frame by client, unit is byte and default is 2048;DownloadPkgSize is size of data downloaded per frame by client, unit is byte, default is 2048. Range of FrameIntervale is [2,100], NetPkgSize is [1024,100*1024]; UploadPkgSize/ DataUpPkgSize is [1024,100*1024]; DownloadPkgSize/DataDownPkgSize is [1024,100*1024].

DataUpPkgSize, DataDownPkgSize is valid only for independent data port.

ServicePath is path of new service, may be "". In this case, default path is used..

For example, suppose the new created service will be saved at directory:

C:AAA\BBB\service name\service files, then ServicePath should be set to "C:AAA\BBB"

CreateServiceEx: if BIN type files exist in the service path, then they are not deleted. Service should use ClearStatic function to clear static data.

CreateService: if BIN type files exist in the service path, then they will be deleted.

6. **Service = _LoadServiceEx (ServiceID,UserName,UserPass, LoadRunModule=True);**

   java: StarServiceClass _LoadServiceEx(String ServiceID,String UserName,String UserPass,boolean LoadRunModule)

   Load a service. Server should not load or create any service before the function is called. ServiceID should be registered in registry under key SOFTWARE\SRPLab\SRPServer.

7. **Service = _LoadServiceWithPath (ServicePath, ServiceName,UserName,UserPass,LoadRunModule=True);**

   java: StarServiceClass _LoadServiceWithPath(String ServicePath,String ServiceName,String UserName,String UserPass,boolean LoadRunModule)

   Server should not load or create any service before the function is called.

8. **Service = _LoadService (ServiceName,UserName,UserPass,LoadRunModule=True);**

   java: StarServiceClass _LoadService(String ServiceName,String UserName,String UserPass,boolean LoadRunModule)

   Server should not load or create any service before the function is called.

9. **_ClearService ();**

   void _ClearService()

   For server service group, service will be unloaded.

   For client service group, connection will be closed.

10. **_ClearServiceEx();**

    java: void _ClearServiceEx()

    For server service group, service will be unloaded.

    For client service group, connection will be closed.

    The function will be processed for all service groups existed.

11. **_ClearLuaGlobal ();**

    java: void _ClearLuaGlobal()

    Clear Lua global variable.

12. **Bool = _ExportServiceHeader(ServiceName,Path=““);**

    java: boolean _ExportServiceHeader(String ServiceName,String Path)

    Export service header file for C++. Path may be ““, in this case, output will be written to default service path.

13. **Bool = _ XmlToService (SXml,DataPath,SegmentName,PrintFunc=None);**

    java: boolean _XmlToService(StarSXmlClass SXml,String DataPath,String SegmentName,String PrintFuncName)

    Another function for creating service, which uses xml file as input. Its syntax follows other documents.

14. **Bool =_ XmlToServiceEx (FileName,PrintFunc=None);**

    java: boolean _XmlToServiceEx(String FileName,String PrintFuncName)

    Another function for creating service, which uses xml file as input. Its syntax follows other documents.

```
def InfoFunc(Info ) :
    print( Info)
```

java/c#:
public void PrintFuncName(String Info)
{
}

5.5.4 client connect to server

1. **_Connect (ServerInterface,ServerName,PortNumber,RetrySecond, UserName, UserPassword, ParaPkg=None, ServiceGroup_ClientConnectProc=None);**
   java: int _Connect(String ServerInterface,String ServerName,int PortNumber,int RetrySecond,String UserName,String UserPassword,StarParaPkgClass ParaPkg,String ServiceGroup_ClientConnectProc)
2. **ServiceGroup._ConnectEx (ServiceName,RetrySecond, UserName,UserPassword, ParaPkg=None, ServiceGroup_ClientConnectProc=None);**
   java: int _ConnectEx(String ServiceName,int RetrySecond,String UserName,String UserPassword,StarParaPkgClass ParaPkg,String ServiceGroup_ClientConnectProc)

Valid at client service group. Client connects to server,. Return value is ConnectionID, and 0 means failure. Prototype of the function is:

```
def ServiceGroup_ClientConnectProc(self,uMes, ConnectionID,
LinkInterfaceStatus,ServerName, PortNumber ) :
```

```
function ServiceGroup:ClientConnectProc(uMes, ConnectionID,
LinkInterfaceStatus,ServerName, PortNumber )
```

```
PHP:
function ServiceGroup_ClientConnectProc(self,uMes, ConnectionID,
LinkInterfaceStatus,ServerName, PortNumber )
{
}
```

```
JAVA/c#:
public void ClientConnectProc(int uMes,int ConnectionID,int
LinkInterfaceStatus,String ServerName,int PortNumber)
{
}
```

uMes:
0 //---after the event ,connection has been setup successfully, and will be followed by service initialization message.
1 //---if ConnectionID is zero, indicates the connection is released and noallback will be created latter. Otherwise CLE will retry to connect.
2 //---service initialization is failed at client side.
3 //---service initialization is successful at client side.
4 //---service finish sync at client side.

5 //---connection is closed.

Interface is string, which format is :

“Host=interface address;if=share lib file name(include extension);site=share lib download address, ftp/http address ;para=parameter string”

for example:" host=192.168.0.1;if=SRPTcpLinkInterface1.dll;site=ftp://127.0.0.1"

1. **ConnectionID = _SConnect (ServerInterface,ServerName,PortNumber, UserName, UserPassword,ParaPkg = nil/None);**

   java: int _SConnect(String ServerInterface,String ServerName,int PortNumber,String UserName,String UserPassword,StarParaPkgClass ParaPkg)

2. **ConnectionID =_SConnectEx(ServiceName,UserName, UserPassword, ParaPkg = nil/None);**

   java: int _SConnectEx(String ServiceName,String UserName,String UserPassword,StarParaPkgClass ParaPkg)

   Setup connection to server, wait, and returns ID of the connection after service is loaded.

Interface is string, which format is :

“Host=interface address;if=share lib file name(include extension);site=share lib download address, ftp/http address ;para=parameter string”

for example:" host=192.168.0.1;if=SRPTcpLinkInterface1.dll;site=ftp://127.0.0.1"

3. **Service =_Connect2 (ServerInterface,ServerName,PortNumber, UserName, UserPassword,SysRootItemName,ParaPkg = None);**

   java: StarServiceClass _Connect2(String ServerInterface,String ServerName,int PortNumber,String UserName,String UserPassword,String SysRootItemName,StarParaPkgClass ParaPkg)

4. **Service =_ConnectEx2(ServiceName,UserName, UserPassword, SysRootItemName,ParaPkg = None);**

   java: StarServiceClass _Connect2Ex(String ServiceName,String UserName,String UserPassword,String SysRootItemName,StarParaPkgClass ParaPkg)

   Setup connection to server,and wait service becomes sync.The return value is service object.

5. **_DisConnect ();**

   java: void _DisConnect()

Close connection.

6. **Bool =_IsConnect ();**

   java: boolean _IsConnect()

Valid at client service group, which is used to determine whether the connection is established.

5.5.5 trigger hyper connection

1. **_HyperLink (URL,bool = CreateNewWindow);**

java: void _HyperLink(String URLArg,boolean CreateNewWindow)

Valid at client and server, trigger a new hyper connection.

5.5.6 trigger app event

1. **_AppEvent(VS_ULONG EventID,VS_CHAR *EventInfo);**

   java: void _AppEvent(int EventID,String EventInfo)

Valid ar client and server, create callback message:MSG_APPEVENT

5.5.7 QueryRecord object

1. **_NewQueryRecord ();**

   java: StarQueryRecordClass _NewQueryRecord()

5.5.8 parapkg object

1. **_NewParaPkg ();**

   java: StarParaPkgClass _NewParaPkg()

5.5.9 binbuf object

1. **_NewBinBuf();**

   java: StarBinBufClass _NewBinBuf()

5.5.10 SXML object

1. **XML =_NewSXml()**

   java: StarSXmlClass _NewSXml()

5.5.11 function para object

1. **FuncPara =_NewFunctionPara()**

   java: StarFunctionParaClass _NewFunctionPara()

5.5.12 CommInterface object

1. **CommInterface =_NewCommInterface()**

   java: StarCommInterfaceClass _NewCommInterface()

5.5.13 execute script segments or files

1. **bool = _RunScript(ScriptInterface,ScriptBuf,Name);**

   java: boolean _RunScript(String ScriptInterface,String ScriptBuf,String Name)

Name is segment name.

Static script language does not support this function, include java/c#.

2. **bool = _RunScriptEx(ScriptInterface,BinBuf,Name);**

   java: boolean _RunScriptEx(String ScriptInterface,StarBinBufClass BinBuf,String Name)

BinBuf is created by _NewBinBuf

Name is segment name.

Static script language does not support this function, include java/c#.

3. **bool = _DoFile(ScriptInterface,FileName);**

   java: boolean _DoFile(String ScriptInterface,String FileName)

If ScriptInterface is ““, default is lua. It can be take value from lua,python,java,c#, or others registered script language.

5.5.14 miscellaneous function

How to process such functions depends on environment. The corresponding command is send to host program by CLE.

1. **_IsDefaultServer ();**

   java: boolean _IsDefaultServer()

   Whether the server is default server, returns bool.

2. **_IsWindowVisible ();**

   java: boolean _IsWindowVisible()

   Whether manager window is visible, returns bool.

3. **_HideWindow ();**

   java: void _HideWindow()

   Hide manager window

4. **_ShowWindow ();**

   java: void _ShowWindow()

   Show manager window.

5. **_SetCaption(Caption);**

   java: void _SetCaption(String Caption)

   Set manager window caption, which is a string.

   When create and load a service, the caption of manager window is set automaticly. So the function should be called after service is created or loaded.

6. **_ExitVSSystem (ErrorInfo);**

   java: void _ExitVSSystem(String ErrorInfo)

   exit program.

7. **_StartVSService (ServiceID);The function will be presented in the future.**

   java: void _StartVSService(String ServiceID)

   Start a service

   ServiceID is the service ID to be started.

8. **_IsAppActive();**

   java: boolean _IsAppActive()

   Whether application is active, returns bool

9. **_SetIdleActive(True/False);**

java: void _SetIdleActive(boolean Args)

10. **_GetVersion ();**

java: Object[] _GetVersion()

Get cle version, return version table.

11. **_GetVersionInfo ();**

java: String _GetVersionInfo()

Get cle version information,return string.

12. **_SetColor(Text,Explane,ObjName,AttrType,Number,Error);**

java: void _SetColor(int Text,int Explane,int ObjName,int AttrType,int NumberColor,int ErrorColor)

set text color

13. **_SetBkColor(BkColor);**

java: void _SetBkColor(int BkColor)

set text background color.

14. **_SetClientBkColor (BkColor);**

java: void _SetClientBkColor(int BkColor)

set client window background color

15. **_ShowStatusMenu (showmenuflag,showstatusflag);**

java: void _ShowStatusMenu(boolean showmenuflag,boolean showstatusflag)

Whether to show menu or status bar, input is bool.

16. **_ClearClientWnd ();**

java: void _ClearClientWnd()

clear client window

17. **_HideClientWnd ();**

java: void _HideClientWnd()

Hide client window.

18. **_ShowClientWnd ();**

java: void _ShowClientWnd()

Show client window.

19. **_SetClientSize (Width,Height);**

void _SetClientSize(int Width,int Height)

Set client window width and height.

20. **Width,Height =_GetClientSize ();**

java: Object[] _GetClientSize()

Get client window width and height.

21. **KernelAllocSize , DataAllocSize , AppAllocSize , ScriptMemoryUsed =_MemorySize ();**

java: Object[] _MemorySize()

KernelAllocSize: memory of core.

DataAllocSize: memory of static data

AppAllocSize: memory of application.

ScriptMemoryUsed: memory of lua.

22. **_SetWindowStyle (bool :SystemMenuFlag, MinimizeFlag, MaximizeFlag, ShowBorderFlag, SizeableFlag);**

java: void _SetWindowStyle(boolean SystemMenuFlag,boolean MinimizeFlag,boolean MaximizeFlag,boolean ShowBorderFlag,boolean SizeableFlag)

Set window parameter.

23. _MoveWindow (X, Y, nWidth, nHeight, bool RepaintFlag);

java: void _MoveWindow(int X,int Y,int nWidth,int nHeight,boolean RepaintFlag)

Move window, If nWidth,nHeight equal to 0, then current window size remains unchanged.

24. X, Y, nWidth, nHeight = _GetWindowPos ();

java: Object[] _GetWindowPos()

get client position.

25. _SetWindowStatus (int Status);

java: void _SetWindowStatus(int Status)

Status //--0 normal 1 minimize other maximize

5.5.15 RawSocket function

If RawSocket fails to setup connection, it will not retry; Application should retry to setup the connection.

1. int =_SetupSocketServer (Interface,PortNumber,ServiceGroup_AcceptFunc);

java: int _SetupSocketServer(String LInterface,int PortNumber,String AcceptFunc)

Set up a SOCKET server. If returns 0, then fails, or else is the ID of the connection.

Interface is link-layer interface, may be ““

```
def ServiceGroup_AcceptFunc( self,ConnectionID, IPAddr, IPPort, MachineID ) :

    return ClientFunc    #-- should return the function which processes the connection.
```

```
function ServiceGroup:AcceptFunc( ConnectionID, IPAddr, IPPort, MachineID )

    return ClientFunc    -- should return the function to processes the connection.
end
```

```
PHP:
function ServiceGroup_AcceptFunc( self,ConnectionID, IPAddr, IPPort, MachineID )
{
    return “ClientFunc”;
}
```

```
java/c#:
public String AcceptFunc(int ConnectionID, String IPAddr, int IPPort, int MachineID)
{
}
```

MachineID is the state machine ID allocated for the client.

2. int =_SetupSocketClient (ServerInterface ,ServerName, PortNumber,

ServiceGroup_ClientFunc);

java: int _SetupSocketClient(String ServerInterface,String ServerName,int PortNumber,String ClientFunc)

Setup a SOCKET client, which return the request ID which can be used to close the connection. The ID will be invalid after the callback function is called.

Interface is link-layer interface, may be ““

```
def ServiceGroup_ClientFunc( self,uMes,MachineID, LinkInterfaceStatus,Para1, Para2 ) :
    uMes == 1   :   VS_SOCKET_ONCONNECT,   Para1 is ipaddress, Para2 is port
    uMes == 2   :   connection is closed
    uMes == 3   :   fail to setup connection
    uMes == 4   :   receive one package ,Para1 is ParaPkg.
```

```
function ServiceGroup:ClientFunc( uMes,MachineID, LinkInterfaceStatus,Para1, Para2 )
    uMes == 1   :   VS_SOCKET_ONCONNECT,   Para1 is ipaddress, Para2 is port
    uMes == 2   :   connection is closed
    uMes == 3   :   ail to setup connection
    uMes == 4   :   receive one package ,Para1 is ParaPkg
    return
end
```

```
java/c#:
public void ClientFunc( int uMes,int MachineID, int LinkInterfaceStatus,Object Para1, Object Para2 ){
}
```

LinkInterfaceStatus =0 normal 1 download 2 error

3. **_CloseSocketConnect (ConnectionID/MachineID);**

   java: void _CloseSocketConnect(int ConnectionID_MachineID)

   Close connection. Input may be connection ID, or request ID, or client MachineID.

4. **Bool =_SocketSend (MachineID,ParaPkg,bool assure);**

   java: boolean _SocketSend(int MachineID,StarParaPkgClass ParaPkg,boolean Assure)

   send one package.

5.5.16 Timer function

1. **_SetTimer (Ticket, ServiceGroup**_Func**, int Arg1, int Arg2)**

   java: int _SetTimer(int Ticket,String TimerFunc,int Arg1,int Arg2)

   Function is function which should be defined in service group.

   Ticket is the interval which uint is 10ms;

   call format:a = **ServiceGroup**._SetTimer(100,Func, 0,0)

   Parameter should be number, and the return value is TimerID

   def **ServiceGroup_**Func(self,TimerID,Arg1,Arg2) :

```
self is service group object
```

```
call format:a = ServiceGroup:_SetTimer(100,Func, 0 )
Parameter should be number, and the return value is TimerID
function ServiceGroup:Func(TimerID,Arg1,Arg2)


end
```

```
java: public void Func(int TimerID,int Arg1,int Arg2)
      {
         return;
      }
```

2. _KillTimer (TimerID)

```
java: void _KillTimer(int TimerID)
```

TimerID is the ID of the timer.

5.5.17 Restart function

1. _ProgramRestart();

```
java: void _ProgramRestart()
```

The function is valid only when cle is started by manager program.

5.5.18 Http up/download function

1. _HttpDownLoad (ServerUrl,ClientPath,FileName);

```
java: void _HttpDownLoad(String ServerUrl,String ClientPath,String FileName)
```

FileName is the file name. The download process is asynchornous and the file downloaded will be saved at local.

SrvGroup._HttpDownLoad("http://www.srplab.com/files","e:","srirrlicht_index.htm")

2. _HttpDownLoadAbort ();

```
java: void _HttpDownLoadAbort()
```

Cancel all http/ftp download.

should be registered only one.

3. _RegWebDownFunction (ServiceGroup_Func);

```
java: void _RegWebDownFunction(String DownFunc)
```

The prototype is :

```
def ServiceGroup_Func (self,uMes,FileName,MaxSize,CurSize)
```

```
function ServiceGroup:Func (uMes,FileName,MaxSize,CurSize)


end
```

```
java:
        public void DownFunc(int uMes,String FileName,int MaxSize,int CurSize)
        {
            return;
        }
```

uMes:

0 //---Start download

1 //---download process

2 //---finish

3 //---error

5.5.19 set current Url and service parameter

1. **_SetEnvCurrentUrl (Url)**

   java: void _SetEnvCurrentUrl(String Args)

Set current service Url

2. **_SetEnvPara (ParaPkg)**

   java: void _SetEnvPara(StarParaPkgClass ParaPkg)

set current service parameter

5.5.20 script edit[Windows]

1. **Bool=_OpenLuaEdit (Module,Config,bool CloseEnable)**

   java: boolean _OpenLuaEdit(String Module,int Config,boolean CloseEnable)

Open editor which is compiled to single sharelib SRPLuaEdit.DLL

Config is the combination of the following value:

#define SRPLUAEDITMODULECONFIG_SCRIPTCONSOLE 0x00000001

#define SRPLUAEDITMODULECONFIG_PROJECT 0x00000002

#define SRPLUAEDITMODULECONFIG_SRPDOC 0x00000004

Module is reserved.

2. **_LuaEditDisp (Info)**

   java: void _LuaEditDisp(String Info)

Display information in editor.

3. **_LuaEditHelp (Type,HelpFile)**

   java: void _LuaEditHelp(int Type,String HelpFile)

Type=0 help info

Display help in editor. Input is help file name which is located on local disk.

=1 Examples project, at this time, the input is help string.

4. **_CloseLuaEdit ()**

   java: void _CloseLuaEdit()

Close editor.

5.5.21 service path

1. **_InsertSearchPath (String Path);**

   java: void _InsertSearchPath(String Path)

Set service seraching path.

2. **_ClearSearchPath ();**

   java: void _ClearSearchPath()

Clear service searching path.

3. **Path=_FirstSearchPath (QueryRecord);**

   java: String _FirstSearchPath(StarQueryRecordClass QueryRecord)

get first searching path.

4. **Path=_NextSearchPath (QueryRecord);**

   java: String _NextSearchPath(StarQueryRecordClass QueryRecord)

get next searching path.

5.5.22 get doc object registered

1. **Object,DocName= _FirstDoc(QueryRecord)**

   java: Object[] _FirstDoc(StarQueryRecordClass QueryRecord)

Get first doc object registered.

2. **Object,DocName = _NextDoc(QueryRecord)**

   java: Object[] _NextDoc(StarQueryRecordClass QueryRecord)

Get next doc object registered.

3. **_RegisterDoc(Object,DocName)**

   java: void _RegisterDoc(Object ObjectArg,String DocName)

Register Doc object.

4. **_UnRegisterDoc(Object)**

   java: void _UnRegisterDoc(Object ObjectArg)

Unregister Doc object.

5.5.23 get static data version

1. **Version=_GetStaticVersion (BinBuf/FileName)**

   String _GetStaticVersion(Object Arg)

Compute static data version, input is file name or binary buffer.

5.5.24 ClipBoard[Windows]

1. **_ToClipBoard(String)**

   java: void _ToClipBoard(String Arg)

Copy string to clipboard.

2. **String=_FromClipBoard()**

   java: String _FromClipBoard()

Get string from clipboard.

5.5.25 Load service from Url

1. int Result=_RunFromUrl(Url,Type,bool WaitFlag)

java: int _RunFromUrl(String Url,int Type,boolean WaitFlag)

Type:

VS_RUMFROMURL_NORESTART 0
VS_RUMFROMURL_RESTART 1
VS_RUMFROMURL_WAITRESTART 2

Result:

SRPLOADPROCESS_OK 0
SRPLOADPROCESS_BUSY -1
SRPLOADPROCESS_DISABLE -2
SRPLOADPROCESS_FAIL -3

After Url may be attach parameter such as"?parameter";
hostip=XXX: Redirect host IP, which is valid for files on http or ftp server. In this case, the url uses this ip address other than returned by DNS.
depend=#depend service 1,# depend service 2, depend service 3; '#' is in front of depend service name, which indicates to download from starcore website, otherwise download from parent url ["depend=" without spaces]
script=lua/python/..; ["script=" without spaces]
The last string is command argument string, service can read it from EnvInputPara
for examples:

http://XXX/XX?depend=AAA,#bbb;para1=111
http://XXX/XX?depend=AAA,#bbb;script=python;para1=111
http://XXX/XX?para1=111

depend,script,and command string are seperated by ";".

If use FTP, the user name and password is input as follow:
ftp://XXX?USER=XXX;PASS=XXX/XXX/XXX
ftp://XXX:21?USER=XXX;PASS=XXX/XXX/XXX

If hostip is set, then when download, url will be replaces by hostip. For example:
ftp://XXX?USER=XXX;PASS=XXX/XXX/XXX?hostip=127.0.0.1,
ftp://XXX:21?USER=XXX;PASS=XXX/XXX/XXX?hostip=127.0.0.1,
Download will be started from ftp://127.0.0.1 和 ftp://127.0.0.1:21

If Url starts with char "@", then the service file locates at local disk.

5.5.26 Set debug and client port

Valid for service group 0

1. **Bool=_SetClientPort (“Interface”,Portnumber)**
   java: boolean _SetClientPort(String LInterface,int Portnumber)
   Interface may be ““
2. **Bool =_SetDebugPort (“Interface”,Portnumber)**
   java: boolean _SetDebugPort(String LInterface,int Portnumber)
   Interface may be ““

Interface is string, which format is :
“Host=interface address;if=share lib file name(include extension);site=share lib download address, ftp/http address ;para=parameter string”
for example:" host=192.168.0.1;if=SRPTcpLinkInterface1.dll;site=ftp://127.0.0.1"

5.5.27 set Telnet,Web,and output port

Valid for service group 0

1. **Bool= _SetTelnetPort (Portnumber)**
   java: boolean _SetTelnetPort(int Portnumber)
2. **Bool= _SetOutputPort (Host,Portnumber)**
   java: boolean _SetOutputPort(String Host,int Portnumber)

Using syslog to recrive infromation
Output is coded as UTF-8 format.

3. **Bool= _SetWebServerPort (Host,Portnumber, ConnectionNumber, PostSize)**
   java: boolean _SetWebServerPort(String Host,int Portnumber,int ConnectionNumber,int PostSize)

Host is Url,may be set to None
ConnectionNumber: the maximum connection number
PostSize : permit upload size, unit is Kbytes.

5.5.28 service register and alloc Cooperator

1. **Bool= _RegisterServer (ServiceName)**
   java: boolean _RegisterServer(String ServiceName)
2. **_AllocCooperator (ServiceName)**
   java: void _AllocCooperator(String ServiceName)
3. **_FreeCooperator(ServiceName)**
   java: void _FreeCooperator(String ServiceName)
4. **Interface,Host,Port= _GetServerUrlInfo ()**
   java: Object[] _GetServerUrlInfo()

Local function to get parameters for client to connect

5.5.29 WebService object refresh

1. **_WebServiceRefresh()**
   void _WebServiceRefresh()

5.5.30 get WSDL

1. **bool =_GetWSDL(int WSDLVersion, string Host, BinBuf)**
   java: boolean _GetWSDL(int WSDLVersion,String Host,StarBinBufClass BinBuf)
   WSDLVersion: reserved, current is version 1.1.
   Host: If equals to nil, then uses default host.
   Mapping type:
   If object sets WebServiceFlag, then it acts as a PortType mapped to WSDL.
   The functions defined in object act as Operation.

5.5.31 string format convert

1. **Utf8= _StringToUtf8(String)**
2. **String= _Utf8ToString(Utf8)**

java/c# does not support these functions.

5.5.32 get platform information

1. **EnvTag= _GetConfigEnvTag()**
   java: String _GetConfigEnvTag()
   Get environment Tag which is set through VS_STARCONFIGEX at cle initial process.
   Three types are predefined:
"", "noloop", "activex"
2. **bool debugserver port result,client port config result,telnet port config result,WebServer port config result = _GetConfigResult()**
   java: Object[] _GetConfigResult()
3. **_GetConfig (SXml)**
   Get config parameter.
   java: void _GetConfig(StarSXmlClass SXml)
4. **Host=_GetConfigHost()**
   java: String _GetConfigHost()
   Get config Host and Web port number.

5.5.33 client download callback (used at server service group)

1. **RefValue=_RegFileReqCallBack(SrvGroup.Function);**
   java: int _RegFileReqCallBack(String Function)
2. **_UnRegFileReqCallBack(RefValue);**
   java: void _UnRegFileReqCallBack(int RefValue)

Function prototype:
```
def ServiceGroup_CallBackProc( self,ClientID, ClientPrivateTag, uMsg, DataFile, DataSize, FileName Or ObjectID, UniqueDataUnitID,Version )
```

```
java:
	public boolean Function(int ClientID,int ClientPrivateTag,int uMsg,boolean DataFile,int
DataSize,String FileName_ObjectID,int UniqueDataUnitID,String Version)
	{
		return true/false;
	}
```

If DataFile = true, then is FileName
If DataFile = false, then is ObjectID.
UniqueDataUnitID represents APPID os static data, and Version represents the version of static data.
uMsg takes value:
0 : start to download
4 : start to upload
if returns true, data can be downloaded or uploaded, or else can not be downloaded or uploaded.

5.5.34 Dispatch callback

1. **bool _RegDispatchCallBack(SrvGroup_Function)**
   java: void _RegDispatchCallBack(String Function)

If input is None, then previous callback will be canceled.
def SrvGroup_Function(self)

```
java:
	public void Function()
	{
		return;
	}
```

Should only register on service group 0, and only once.

5.5.35 lock lua table

1. **bool _LockLuaTable()**
   java: boolean _LockLuaTable()
2. **bool _UnLockLuaTable()**
   java: boolean _UnLockLuaTable()

After being locked, the lua table is readonly, and is not writeable. Tables include global table and objects.
The function is only valid for lua or embedded lua script.

5.6 Service object

5.6.1 attribute

1. **Service._Path**
   service path.
2. **Service._Name**
   service name.
3. **Service._FrameTimerInterval**
   Get interval between frames. The return is number which unit is 10ms.
4. **Service._ID**
   ID of the service
5. **Service._FrameTicket**
   Frame ticket of the service.
6. **Service._ServiceGroup**
   Service group which service belongs to.

5.6.2 basic function

1. **_DeactiveAll ()**
   java: void _DeactiveAll()
   deactive all objects.
2. **_GetObject (ObjectName)**
   java: StarObjectClass _GetObject(String ObjectName)
   get object, input is object name
3. **_GetObjectEx (ObjectID);**
   java: StarObjectClass _GetObjectEx(String ObjectID)
   ObjectID is the ID of the object.
   calling format : a = Service._GetObjectEx ("66efc37a-a16c-442b-bc2a-3df5b03b3294")
4. **_GetObjectEx2 (ServiceName,ObjectName);**
   java: StarObjectClass _GetObjectEx2(String ServiceName,String ObjectName)
   Get object of special service.
5. **_GetSysRootItem(SysRootItemName)**
   java: StarServiceItemClass _GetSysRootItem(String SysRootItemName)
   Get service item.If service item does not active, the returns None.
6. **_Exit ()**
   java: void _Exit()
   exit the service.
7. **_Save (Path=None)**
   java: void _Save(String Path)
   Save service
8. **_IsChange()**
   java: boolean _IsChange()
   whether the service is changed

9. **_IsActive ()**

java: boolean _IsActive()

whether the service isactive.

10. **_QueryFirstFromSDT()**

java: StarObjectClass _QueryFirstFromSDT()

query first object from service table

11. **_QueryNextFromSDT()**

java: StarObjectClass _QueryNextFromSDT()

query next object from service table

12. **_PrintInfo ()**

java: void _PrintInfo()

print service information.

13. **Bool =_CreateSysRootItem(SysRootItemName, DependSysRootItem, SysRootItemID=None,SysRootItemIDEx=None)/SysRootItem=_CreateSysRootItemEx(SysRootItemName, DependSysRootItem,SysRootItemID=None, SysRootItemIDEx=None)**

java: boolean _CreateSysRootItem(String SysRootItemName,String DependSysRootItem,String SysRootItemID,String SysRootItemIDEx)

java: StarServiceItemClass _CreateSysRootItemEx(String SysRootItemName,String DependSysRootItem,String SysRootItemID,String SysRootItemIDEx)

create service item,SysRootItemID is UUID string, SysRootItemIDEx is UUID string which may be ““,DependSysRootItem is the depended service item which may be ““ for not existing.

14. **_ActiveSysRootItem(SysRootItemName)**

java: void _ActiveSysRootItem(String SysRootItemName)

Activate service item, no return value.

15. **_ActiveAllSysRootItem()**

java: void _ActiveAllSysRootItem()

Activate all service items, no return value, valid at server service group.

16. **_DeactiveSysRootItem (SysRootItemName)**

java: void _DeactiveSysRootItem(String SysRootItemName)

deactivate service item, no return value.

17. **_ActiveCSysRootItem(ClientID,SysRootItemName)**

java: void _ActiveCSysRootItem(int ClientID,String SysRootItemName)

activate service item at client service group, no return value, should be called at server service group.

18. **_DeactiveCSysRootItem (ClientID,SysRootItemName)**

java: void _DeactiveCSysRootItem(int ClientID,String SysRootItemName)

deactivate service item at client side, no return value, valid at server service group.

19. **Name = _QueryFirstSysRootItem ()**

java: String _QueryFirstSysRootItem()

query first service item name

20. **Name = _QueryNextSysRootItem ()**

java: String _QueryNextSysRootItem()

query next service item name

21. **Name,ID,Continue = _QueryFirstDepend (QueryRecord)**

java: Object[] _QueryFirstDepend()

query first service depended.

22. **Name,ID,Continue =_QueryNextDepend (QueryRecord)**

java: Object[] _QueryNextDepend()

query next service depended.

23. **bool =_IsOsSupport (ProgramRunType,OsType)**

java: boolean _IsOsSupport(int ProgramRunType,int OsType)

Returns true if the corresponding OSType sharelib of modules of the service and depended service exists.

24. **bool =_SetClientObject (ClientID,Object)**

java: boolean _SetClientObject(int ClientID,StarObjectClass Obj)

Called at server, each client should only exist one. ClientID can not be set to 0.

25. **Object =_GetClientObject()**

java: StarObjectClass _GetClientObject()

Gets client object, called at client side.

26. **_SetPrivateTag (ClientPrivateTag)**

java: void _SetPrivateTag(int ClientPrivateTag)

Valid at client side. The tag is used for server to check the validity of remotecall, global object created, changed or deleted operations.

27. **value=Service._GetOPPermission()**

java: int _GetOPPermission()

Get operation permission. valid at client side.

5.6.3 client login function

1. **_RegMachineFunction(Service_ClientRegisterProc);**

java: void _RegMachineFunction(String FunctionName)

Valid at server service group. The callback function is registered to process client login, which format is:

```
def Service_ClientRegisterProc( self,int uMes,int ClientID,char SrcServiceID,
SrcServiceAdd,int SrcServicePort,ParaPkg,UserName,UserPassword) :
```

```
function Service:ClientRegisterProc( int uMes,int ClientID,char SrcServiceID,
SrcServiceAdd,int SrcServicePort,ParaPkg,UserName,UserPassword):
```

java/c#:

```
public void ClientRegisterProc(int uMes,int ClientID,String SrcServiceID, String
 SrcServiceAdd,int SrcServicePort,StarParaPkgClass ParaPkg, String UserName,
 String UserPassword)
```

No result needed return from the callback. If client is permit to login, then the callback should call _AcceptClient(ClientID,True,True). If not, should call _AcceptClient(ClientID,False,False). In addition, _AcceptClient may be called sometime after the callback function.

For Login, uMes = 1
SrcServiceAdd and SrcServicePort are source service address and port which are invalid other than in service redirect process.ParaPkg is login parameter.

For Logout, uMes = 3,
SrcServiceID,SrcServiceAdd ,SrcServicePort,ParaPkg is set to None

For LoadServiceOk, uMes = 2,
SrcServiceID,SrcServiceAdd ,SrcServicePort,ParaPkg is set to None

2. **Bool =_AcceptClient (ClientID,True/False,bool ReSyncFlag,char *TermOldScript, char *InitNewScript, ClientPrivateTag ,ClientOPPermission, ClientUploadMaxSize);**

   java: boolean _AcceptClient(int ClientID,boolean Result,boolean ReSyncFlag,String TermOldScript,String InitNewScript,int ClientPrivateTag,int ClientOPPermission,int ClientUploadMaxSize)

Valid at server which is used to accept or reject client login, and is called in ClientRegisterProc function or sometime later. Login procedure is finished after the function is called.If ReSyncFlag == True, then client will be forced to resychronize for the same service, otherwise not.

TermOldScript: is the lua script called before service is load, should use carefully. If no active service exists at client side, then it is ignored. Before running the script, CLE creates a lua global variable "_gService" which represents current service.

InitNewScript: is the lua script called before service is load Before running the script, CLE creates a lua global variable "_gService" represents current service.

TermOldScript + InitNewScript: total length should < 32Kbytes.

If is redirected from the same service, then only InitNewScript is valid.

When InitNewScript is executed, the service is not synchronized, do not operate objects in the service.

ClientOPPermission takes value from list:

#define VSCLIENTOP_CREATE ((VS_ULONG)0x00000001)
#define VSCLIENTOP_DELETE ((VS_ULONG)0x00000002)
#define VSCLIENTOP_CHANGE ((VS_ULONG)0x00000004)

ClientUploadMaxSize: maximum size permitted for upload at client side, which unit is bytes. Client side uses SetStaticData to upload data.

3. **IPStr,OsType,RunType =_GetClientInfo (ClientID);**

   java: Object[] _GetClientInfo(int ClientID)

Valid at server, returns client informaton

4. **_GetClientNumber ();**

   java: int _GetClientNumber()

returns client number.

5. **_PrintClientInfo ();**

   java: void _PrintClientInfo()

prints all clients information.

6. _DelClient (ClientID);

java: void _DelClient(int ClientID)

Valid at client and server. The connection will be closed after the function call.

5.6.4 object operation callback—vaid at server [object change,create and delete]

1. _RegClientOpFunction(Service_Func);

java: void _RegClientOpFunction(String FunctionName)

The function prototype is:

```
def Service_Func (self,Op,ClientID,ClientPrivateTag,Object,AttrIndex,ClassID) :

    return True
```

```
function Service:Func (Op,ClientID,ClientPrivateTag,Object, AttrIndex,ClassID)

    return true
end
```

java/c#:

```
public boolean OpFunction(int Op,int ClientID,int ClientPrivateTag,StarObjectClass Obj,int AttrIndex,String ClassID)
```

ClientPrivateTag: is used to judge whether the client is legal.

If the function returns False, then the operation is denied, otherwise is permitted

//---for delete Op = 0; Object is valid

//---for change Op = 1; Object is valid

//---for create Op = 2; Object is the parent object,ClassID is id of the class.

5.6.5 service redirect

1. _Redirect (ClientID, DesServerInterface,DesServerName, DesServerPortNumber,ParaPackage, Service_ClientRedirectProc=None);

java: void _Redirect(int ClientID,String DesServerInterface,String DesServerName,int DesServerPortNumber,StarParaPkgClass ParaPackage,String RedirectProcName)

DesServerInterface may be ““

Valid at server, direct client to connect to other server, no return value.

```
def Service_ClientRedirectProc( self,uMsg, ClientID, DesServerName,DesServerPortNumber) :
```

```
function Service:ClientRedirectProc( uMsg, ClientID, DesServerName,DesServerPortNumber)
```

```
java/c#:
public void ClientRedirectProc( int uMsg, int ClientID, String DesServerName,int
DesServerPortNumber)
```

uMsg = 0 : redirect succeed. Server will close the connection after the callback by cle.
uMsg = 1 : redirect failed.
call examples:
_Redirect(ClientID,DesServerName,DesServerPortNumber)
_Redirect(ClientID,DesServerName,DesServerPortNumber, ClientRedirectProc)
_Redirect(ClientID,DesServerName,DesServerPortNumber,ParaPackage, ClientRedirectProc)

5.6.6 file upload and download function

1. **_DownLoad (ServerPath,ClientPath,FileName); [reserved]**
   java: void _DownLoad(String ServerPath,String ClientPath,String FileName)

Download file from server, valid at client or debug.

2. **_HttpDownLoad (ServerUrl, ClientPath,FileName);**
   java: void _HttpDownLoad(String ServerUrl,String ClientPath,String FileName)

Service._HttpDownLoad("http://www.srplab.com/files","e:","srirrlicht_index.htm")

3. **_HttpDownLoadAbort ();**
   java: void _HttpDownLoadAbort()

SrvGroup: _HttpDownLoadAbort ()
Cancel all http/ftp download process.

4. **_UpLoad (ServerPath,ClientPath,FileName); [reserved]**
   java: void _UpLoad(String ServerPath,String ClientPath,String FileName)

Upload file to server, valid at client or debug.

5. **int RefValue=_RegFileCallBack(Service_Function);**
   java: int _RegFileCallBack(String FunctionName)

valid at client or debug. should only register one.

6. **_UnRegFileCallBack(RefValue);**
   java: void _UnRegFileCallBack(int RefValue)

callback function prototype

```
def Service_DownUpProc( self, uMsg, DataFile,ReceiveOrSendOffset,DataSize,
FileName Or Object, UniqueDataUnitID,Version )
```

```
function Service: DownUpProc( uMsg, DataFile,ReceiveOrSendOffset,DataSize,
FileName Or Object, UniqueDataUnitID,Version )
```

```
java/c#:
public void DownUpProc( int uMsg, boolean DataFile,int ReceiveOrSendOffset,int
DataSize, Object FileName_Object, int UniqueDataUnitID,String Version )
```

If DataFile = true, then variable FileName_Object is FileName
If DataFile = false, then variable FileName_Object is ID, UniqueDataUnitID represents APPID of static data, and Version represents version of static data.
uMsg takes value:
0 : start download, 1:download progress,2:finish,3:error
4 : start upload, 5:upload progress,6:finish,7:error

5.6.7 create object (base class not exist)

1. **_New/_NewEx (ObjectID,QueueAttrName,ParentObject)**
   java: StarObjectClass _New(Object...Args)
   java: StarObjectClass _NewEx(Object...Args)
   refer to Object._New
2. **_NewGlobal/_NewGlobalEx(ObjectID,ClientID, QueueAttrName,ParentObject)**
   java: StarObjectClass _NewGlobal(Object...Args)
   java: StarObjectClass _NewGlobalEx(Object...Args)
   refer to Object._NewGlobal
3. **_NewClient/_NewClientEx(ObjectID,ClientID, QueueAttrName,ParentObject)**
   java: StarObjectClass _NewClient(Object...Args)
   java: StarObjectClass _NewClientEx(Object...Args)
   refer to Object._NewClient

5.6.8 service macro definition

1. **_PrintMacro (ServiceName.MacroName);**
   java: void _PrintMacro(String ServiceName_MacroName)
   print macro defininition. If MacroName does not exist,then print macro list of the service.

5.6.9 user management—valid at server,service object should get by root user

1. **Bool = _CreateUser (UserName,UserPass, ReadWriteOrExecute);**
   java: boolean _CreateUser(String UserName,String UserPass,int ReadWriteOrExecute)

If user does not exist, then create new one, or else change its information.
ReadWriteOrExecute may be combination of the following values:

READ	0x01
WRITE	0x02
EXECUTE	0x04
EXPORTXML	0x08

2. **_DeleteUser(UserName);**
   java: void _DeleteUser(String UserName)
3. **UserName, ReadWriteOrExecute= _FirstUser(QueryRecord);**
   java: Object[] _FirstUser(StarQueryRecordClass QueryRecord)

QueryRecord is allocated using function ServiceGroup._NewQueryRecord

4. **UserName, ReadWriteOrExecute= _NextUser(QueryRecord);**

   java: Object[] _NextUser(StarQueryRecordClass QueryRecord)

   If user does not exist, the function returns None.

5.6.10 execute script

1. **True/false, ErrorInfo =_RunScript(ScriptInterface,ScriptBuf, Name, WorkDirectory);**

   java: Object[] _RunScript(String ScriptInterface,String ScriptBuf,String Name,String WorkDirectory)

Executes script,Name is segment name,WorkDirectory is work directory, may be ““.

Static language does not support the function, such as java/c#.

2. **True/false, ErrorInfo =_RunScriptEx(ScriptInterface,BinBuf, Name, WorkDirectory);**

   java: Object[] _RunScriptEx(String ScriptInterface,StarBinBufClass BinBuf,String Name,String WorkDirectory)

Executes script,Name is segment name,WorkDirectory is work directory, may be ““.

BinBuf is allocated by _NewBinBuf.

Static language does not support the function, such as java/c#.

3. **True/false, ErrorInfo = _DoFile(ScriptInterface,FileName, WorkDirectory);**

   java: Object[] _DoFile(String ScriptInterface,String FileName,String WorkDirectory)

If ScriptInterface is ““, then default is lua. ScriptInterface may be lua,python,java,c# or other script language registered.

This function can load sharelibrary, in this case, ScriptInterface should be set to ““.

5.6.11 get peer ip address

1. **string IP =_GetPeerIP (ClientID)**

   java: String _GetPeerIP(int ClientID)

Called at client side, the input ClientID is ignored.

Called at server side, ClientID indicates which client.

5.6.12 get server ID at client

1. **ServerID =_GetServerID()**

   java: int _GetServerID()

Valid at client side, which is used to get server ID for the communication with server.

5.6.13 force to save service static data (valid at server side)

1. **_ForceToSaveStatic ()**

   java: void _ForceToSaveStatic()

When service exits, then service static data is forced to save.

5.6.14 remove expired data

1. **_ClearStatic (Days)**

java: void _ClearStatic(int Days)

5.6.15 static data http download callback—valid at server side

1. **_RegServerWebDownFunction (Service_Func);**

   java: void _RegServerWebDownFunction(String FuncName)

function prototype is:

```
def Service_Func (self,uMes,FileName,MaxSize,CurSize)
```

```
function Service:Func (uMes,FileName,MaxSize,CurSize)

end
```

```
java/c#:
public void Func (int uMes,String FileName,int MaxSize,int CurSize)
{}
```

uMes:

```
#define VSFILE_ONDOWNSTART          0       //---start download
#define VSFILE_ONDOWNPROGRESS       1       //---download progress
#define VSFILE_ONDOWNFINISH         2       //---finish
#define VSFILE_ONDOWNERROR          3       //---error
```

5.6.16 get object string from Lua

1. **_GetObjectFromLua(Str)**

   java: StarObjectClass _GetObjectFromLua(String Str)

The format of String is as "Service.DriveClass".

5.6.17 Pack static data

1. **_PackStaticData()**

   java: void _PackStaticData()

Removes redundant space of static data file.

5.6.18 Xml file

1. **bool = _XmlToSysRootItem (SXml,DataPath, SegmentName,PrintFunc=None)**

   java: boolean _XmlToSysRootItem(StarSXmlClass SXml,String DataPath,String SegmentName,String PrintFuncName)

```
def PrintFunc(Info ) :
    Print( Info)
```

2. **bool = _XmlToObject (SXml,Object/SystemRootItem,QueueName,DataPath,**

SegmentName,PrintFunc=None)

java: boolean _XmlToObject(StarSXmlClass SXml,Object Obj,String QueueName,String DataPath,String SegmentName,String PrintFuncName)

3. **bool = _ ServiceToXml (SXml,PassWord,DataPath,bool CFunctionFlag, bool OutputObjectID ,PrintFunc=None)**

   java: boolean _ServiceToXml(StarSXmlClass SXml,String PassWord,String DataPath,boolean CFunctionFlag,boolean OutputObjectID,String PrintFuncName)

4. **bool = _SysRootItemToXml (SXml, SysRootItemName,DataPath, bool CFunctionFlag, bool OutputObjectID ,PrintFunc=None)**

   java: boolean _SysRootItemToXml(StarSXmlClass SXml,String SysRootItemName,String DataPath,boolean CFunctionFlag,boolean OutputObjectID,String PrintFuncName)

5. **bool = _ ObjectToXml (SXml,Object,DataPath, bool CfunctionFlag, bool OutputObjectID ,PrintFunc=None)**

   java: boolean _ObjectToXml(StarSXmlClass SXml,Object Obj,String DataPath,boolean CfunctionFlag,boolean OutputObjectID,String PrintFuncName)

```
function Service:PrintFunc(Info)
end
```

```
java/c#:
public void PrintFunc(String Info)
{
}
```

5.6.19 atomic function

1. **AtomicDepend= _CreateAtomicDepend (DependServiceName)**

   java: int _CreateAtomicDepend(String DependServiceName)

2. **AtomicMacro= _CreateAtomicMacro (MacroName,MacroType)**

   java: int _CreateAtomicMacro(String MacroName,int MacroType)

   MacroType = 0Integer MacroType = 1 float MacroType = 2 string

3. **AtomicMacroItem= _CreateAtomicMacroItem (MacroObject, MacroItemName,string MacroItemValue)**

   java: int _CreateAtomicMacroItem(int MacroObject,String MacroItemName,String MacroItemValue)

4. **AtomicModule= _CreateAtomicModule (ModuleName,ModuleType,ObjectID)**

   java: int _CreateAtomicModule(String ModuleName,int ModuleType,String ObjectID)

5. **AtomicModule=_CreateAtomicEditModule (ModuleName,ObjectID)**

   java: int _CreateAtomicEditModule(String ModuleName,String ObjectID)

6. **AtomicStruct=_CreateAtomicStruct (StructName,StructCaption,ObjectID)**

   java: int _CreateAtomicStruct(String StructName,String StructCaption,String ObjectID)

7. **AtomicSysRootItem= _CreateAtomicSysRootItem (SysRootItemName,DependSysRootItem)**

   java: int _CreateAtomicSysRootItem(String SysRootItemName,String DependSysRootItem)

8. **AtomicObject=_CreateAtomicObject (ParentAtomicObject, ParentQueueName, AtomicClassObject, ObjectName,ObjectID)**

   java: int _CreateAtomicObject(int ParentAtomicObject,String ParentQueueName,int AtomicClassObject,String ObjectName,String ObjectID)

   If ParentObject is service item, then ParentQueueName may be nil.

9. **AtomicAttribute= _CreateAtomicAttachAttribute (**AtomicObject, AttributeName, Caption, int Type, int StaticID, int SyncFlag, int CreateFlag, int NotifyFlag, int EditType, int EditControl, int EditReadOnly, string Default,string Desc)

   java: int _CreateAtomicAttachAttribute(int AtomicObject,String AttributeName,String Caption,int Type,int StaticID,int SyncFlag,int CreateFlag,int NotifyFlag,int EditType,int EditControl,int EditReadOnly,String Default,String Desc)

   Attribute type supported is:

   TYPE_BOOL :
   TYPE_INT8 :
   TYPE_UINT8 :
   TYPE_INT16 :
   TYPE_UINT16 :
   TYPE_INT32 :
   TYPE_UINT32 :
   TYPE_FLOAT :
   TYPE_LONG :
   TYPE_ULONG :
   TYPE_CHAR :
   TYPE_COLOR :
   TYPE_RECT :
   TYPE_FONT :
   TYPE_TIME :
   TYPE_STRUCT :

   a. “SyncFlag”,TYPE_UINT8,default is 0 note: 0 global 1 local
   b. “CreateFlag”,TYPE_UINT8,default is 0 note: 0 do not need 1 need
   c. “NotifyFlag”,TYPE_UINT8,default is 0 note:1 notify before change 2 notify after change, or both.
   d. “StaticID”,TYPE_ULONG,default is 0
   e. “EditType”, TYPE_UINT8,default is 0 note:0 normal edit 1 combobox 3 button 4 check box 5 can not edit 6 mask edit
   f. “EditControl”, TYPE_UINT8,default is 0 note:0 none 1 display callback
   g. “EditReadOnly”, TYPE_UINT8,default is 0 note:0can edit 1 readonly
   h. “ComboBox”,string,default is ““, or indicates the macro name.

10. **AtomicAttribute= _CreateAtomicAttribute (**AtomicObject, AttributeName, Caption, Type, StaticID, SyncFlag, CreateFlag, NotifyFlag, EditType, EditControl, EditReadOnly, Default,Desc)

    java: int _CreateAtomicAttribute(int AtomicObject,String AttributeName,String Caption,int Type,int StaticID,int SyncFlag,int CreateFlag,int NotifyFlag,int EditType,int EditControl,int EditReadOnly,String Default,String Desc)

Attribute type supported is:

TYPE_BOOL :
TYPE_INT8 :
TYPE_UINT8 :
TYPE_INT16 :
TYPE_UINT16 :
TYPE_INT32 :
TYPE_UINT32 :
TYPE_FLOAT :
TYPE_LONG :
TYPE_ULONG :
TYPE_LONGHEX :
TYPE_ULONGHEX :
TYPE_VSTRING :
TYPE_PTR :
TYPE_STRUCT :
TYPE_CHAR :
TYPE_COLOR :
TYPE_RECT :
TYPE_FONT :
TYPE_TIME :
TYPE_UUID :
TYPE_STATICID :

11. AtomicAttribute= _CreateAtomicFuncRetAttribute (AtomicObject, int Type,Desc)

java: int _CreateAtomicFuncRetAttribute(int AtomicObject,int Type,String Desc)

Return value type supported is:

TYPE_BOOL :
TYPE_INT8 :
TYPE_UINT8 :
TYPE_INT16 :
TYPE_UINT16 :
TYPE_INT32 :
TYPE_UINT32 :
TYPE_FLOAT :
TYPE_LONG :
TYPE_ULONG :

TYPE_CHARPTR :
TYPE_PARAPKGPTR :
TYPE_PTR :
TYPE_VOID :

TYPE_INT8PTR :
TYPE_UINT8PTR :
TYPE_INT16PTR :
TYPE_UINT16PTR :
TYPE_INT32PTR :
TYPE_UINT32PTR :
TYPE_FLOATPTR :
TYPE_LONGPTR :
TYPE_ULONGPTR :
TYPE_STRUCTPTR :
TYPE_COLORPTR :
TYPE_RECTPTR :
TYPE_FONTPTR :
TYPE_TIMEPTR :
TYPE_UUIDPTR :
TYPE_OBJPTR :

12. AtomicAttribute= _CreateAtomicFuncParaAttribute (AtomicObject, AttributeName, AttributeCaption ,Type,Desc)

java: int _CreateAtomicFuncParaAttribute(int AtomicObject,String AttributeName,String AttributeCaption,int Type,String Desc)

Parameter type supported is:

TYPE_BOOL :
TYPE_INT8 :
TYPE_UINT8 :
TYPE_INT16 :
TYPE_UINT16 :
TYPE_INT32 :
TYPE_UINT32 :
TYPE_FLOAT :
TYPE_LONG :
TYPE_ULONG :
TYPE_COLOR :
TYPE_RECT :
TYPE_FONT :
TYPE_TIME :
TYPE_UUID :
TYPE_PARAPKGPTR :

TYPE_PTR :
TYPE_VOID :
TYPE_CHARPTR :

TYPE_INT8PTR :

TYPE_UINT8PTR :
TYPE_INT16PTR :
TYPE_UINT16PTR :
TYPE_INT32PTR :
TYPE_UINT32PTR :
TYPE_FLOATPTR :
TYPE_LONGPTR :
TYPE_ULONGPTR :
TYPE_STRUCTPTR :
TYPE_COLORPTR :
TYPE_RECTPTR :
TYPE_FONTPTR :
TYPE_TIMEPTR :
TYPE_UUIDPTR :
TYPE_OBJPTR :

13. **AtomicAttribute= _CreateAtomicStructAttribute (**AtomicObject, AttributeName, Caption, Type, Desc**)**

java: int _CreateAtomicStructAttribute(int AtomicObject,String AttributeName,String Caption,int Type,String Desc)

Attribute type of struct supported is:

TYPE_BOOL :
TYPE_INT8 :
TYPE_UINT8 :
TYPE_INT16 :
TYPE_UINT16 :
TYPE_INT32 :
TYPE_UINT32 :
TYPE_FLOAT :
TYPE_LONG :
TYPE_ULONG :
TYPE_CHAR :
TYPE_COLOR :
TYPE_RECT :
TYPE_FONT :
TYPE_TIME :
TYPE_UUID :
TYPE_MEMORY(Reserved) :

14. **Bool= _SetAtomicAttributeLength (AtomicObject,Length)**

java: boolean _SetAtomicAttributeLength(int AtomicObject,int Length)

used for char array.

15. **Bool= _SetAtomicAttributeStruct (AtomicObject**,**AtomicStruct)**

java: boolean _SetAtomicAttributeStruct(int AtomicObject,int AtomicStruct)

Set the attribute corresponding struct or class.

16. **Bool= _SetAtomicAttributeCombobox (AtomicObject,MacroName)**

java: boolean _SetAtomicAttributeCombobox(int AtomicObject,String MacroName)

Set attribute macro.

17. **Bool= _SetAtomicAttributeSyncFlag (AtomicObject,int SyncFlag)**

java: boolean _SetAtomicAttributeSyncFlag(int AtomicObject,int SyncFlag)

Set attribute sync : type ==0 sync attribute, ==1 local attribute.

18. **Bool= _SetAtomicObjectSyncGroup (AtomicObject,int SyncGroup)**

java: boolean _SetAtomicObjectSyncGroup(int AtomicObject,int SyncGroup)

set object syncgroup.

19. **Bool= _SetAtomicObjectAttribute (AtomicObject,SysEvent(bool), int SpecialEvent,int ActiveCmd, int SaveFlag)**

java: boolean _SetAtomicObjectAttribute(int AtomicObject,boolean SysEvent,int SpecialEvent,int ActiveCmd, int SaveFlag)

set object attribute

i. "SysEvent", TYPE_BOOL,default is false

j. "SpecialEvent", TYPE_UINT8,default is 0

#define VSSYSEVENT_PROCESS_TICKET 0x0001 //---10msTicket event

#define VSSYSEVENT_PROCESS_FRAMETICKET 0x0002 //---service frame pulse event.

#define VSSYSEVENT_PROCESS_IDLE 0x0004 //---program Idle event

#define VSSYSEVENT_PROCESS_APPACTIVE 0x0008

#define VSSYSEVENT_PROCESS_APPDEACTIVE 0x0010

#define VSSYSEVENT_PROCESS_SERVICEACTIVE 0x0020

#define VSSYSEVENT_PROCESS_SERVICEDEACTIVE 0x0040

k. "ActiveCmd", TYPE_UINT8,default is 0

#define VSACTIVE_ALONE 0 //---Activated or deactivated through command.

#define VSACTIVE_FOLLOW 1 //---Activvated or deactivvated with parent object. If parent object is service item, then the object will be activvated automaticlly

//--The following two commands is dynamic, can not be saved.

l. "SaveFlag", TYPE_UINT8, default is list below.

#define VSSTATIC_SAVE 0 //---save global and static object.

#define VSSTATIC_CLIENTSAVE 1 //---Saved at client, for client dynamic object and global dynamic object

#define VSSTATIC_NONE 2 //---do not save, local object

20. **AtomicScript= _CreateAtomicScript (AtomicObject, ScriptName,ObjectID, Desc, ScriptBuf)**

java: int _CreateAtomicScript(int AtomicObject,String ScriptName,String ObjectID,String Desc,String ScriptBuf)

creates object script.

21. **AtomicFunction= _CreateAtomicFunction (AtomicObject, FunctionName,ObjectID, Desc, CantOvl(bool), CallBack(bool),StdCallFlag(bool), GlobalFunctionFlag(bool))**

AtomicFunction,ErrorInfo= _CreateAtomicFunctionEx (AtomicObject, FunctionName,ObjectID, Desc, CantOvl(bool), CallBack(bool),Type(int) ,StdCallFlag(bool), GlobalFunctionFlag(bool))

java: int _CreateAtomicFunction(int AtomicObject,String FunctionName,String ObjectID,String Desc,boolean CantOvl,boolean CallBack,boolean StdCallFlag,boolean GlobalFunctionFlag)

java: Object[] _CreateAtomicFunctionEx(int AtomicObject,String FunctionName,String ObjectID,String Desc,boolean CantOvl,boolean CallBack,String Type,boolean StdCallFlag,boolean GlobalFunctionFlag)

creates object funciton

Type is functipn prototype. For example: "VS_CHAR *GetBackImg(VS_INT32 IndexX,VS_INT32 IndexY)"

22. **AtomicFunction= _CreateAtomicLuaFunction(AtomicObject, LuaFunctionName,ObjectID, Desc)**

java: int _CreateAtomicLuaFunction(int AtomicObject,String LuaFunctionName,String ObjectID,String Desc)

creates lua function

23. **AtomicFunction= _CreateAtomicOvlFunction (AtomicObject, FunctionName, OriginFunctionName,ObjectID, Desc, CantOvl(bool))**

java: int _CreateAtomicOvlFunction(int AtomicObject,String FunctionName,String OriginFunctionName,String ObjectID,String Desc,boolean CantOvl)

creates object overloading function,OriginFunctionName is the function defined in parent class.

24. **AtomicInEvent= _CreateAtomicInEvent (AtomicObject,** InEventName**,ObjectID** , OutEventName**)**

java: int _CreateAtomicInEvent(int AtomicObject,String InEventName,String ObjectID,String OutEventName)

creates object event,OutEventName is the output event defined in parent class

25. **AtomicOutEvent= _CreateAtomicOutEvent (AtomicObject, OutEventName,ObjectID, Desc, DynamicFlag(bool))**

java: int _CreateAtomicOutEvent(int AtomicObject,String OutEventName,String ObjectID,String Desc,boolean DynamicFlag)

creates object output event

26. **AtomicSysRootItem= _GetAtomicSysRootItem(SysRootItemName)**

java: int _GetAtomicSysRootItem(String SysRootItemName)

gets service item atomic object.

27. **AtomicObject= _GetAtomicObjectEx (ParentAtomicObject, ObjectName)**

java: int _GetAtomicObjectEx(int ParentAtomicObject,String ObjectName)

gest atomic object

28. **ID= _GetAtomicID (AtomicObject)**

java: String _GetAtomicID(int AtomicObject)

gets ID of atomic object

29. **AtomicObject= _ObjectToAtomic (Object)**

java: int _ObjectToAtomic(StarObjectClass Obj)

Converts normal object to atomic object

30. Object= _AtomicToObject(AtomicObject)

java: StarObjectClass _AtomicToObject(int AtomicObject)

Converts atomic object to normal object

31. Create simple object, struct, object attribute and function

AtomicObject,ErrorInfo = _CreateAtomicObjectSimple(SysRootItemName,ObjectName,Attribute**,ObjectID**)

java: Object[] _CreateAtomicObjectSimple(String SysRootItemName,String ObjectName,String Attribute,String ObjectID)

If SysRootItemName does not exist, then create new one.

Attribute is string of the attribute, such as "VS_CHAR aaa;VS_INT32 bbb;" it may be set to nil; Local attribute should add prefix "local", such as "local VS_CHAR aaa;VS_INT32 bbb;"

AtomicStruct,ErrorInfo = _CreateAtomicStructSimple(StructName,Attribute**,ObjectID**)

java: Object[] _CreateAtomicStructSimple(String StructName,String Attribute,String ObjectID)

Attribute is string of the attribute, such as "VS_CHAR aaa;VS_INT32 bbb;"

AtomicObject,ErrorInfo = _CreateAtomicObjectAttributeSimple(AtomicObject,Attribut)

java: Object[] _CreateAtomicObjectAttributeSimple(int AtomicObject,String Attribut)

Attribute is string of the attribute, such as "VS_CHAR aaa;VS_INT32 bbb;" Local attribute should add prefix "local", such as "local VS_CHAR aaa;VS_INT32 bbb;"

AtomicFunction,ErrorInfo = _CreateAtomicFunctionSimple(AtomicObject,FunctionName,Attribute**,ObjectID,StdCallFlag(bool),GlobalFunctionFlag(bool)**)

java: Object[] _CreateAtomicFunctionSimple(int AtomicObject,String FunctionName,String Attribute,String ObjectID,boolean StdCallFlag,boolean GlobalFunctionFlag)

Attribute is the function prototype. For example: "VS_CHAR *GetBackImg(VS_INT32 IndexX,VS_INT32 IndexY)"

32. attach sharelib

bool = _AtomicAttach(int atomicobject, ShareLibName)

java: boolean _AtomicAttach(int AtomicObject,String ShareLibName)

The function should be called after all function definition has been created.

_AtomicAttach(atomicobject,"user32.dll");

5.6.20 output service header and skeleton file

1. bool,errorInfo= _ExportModule (XmlCfgFile)

java: Object[] _ExportModule(String XmlCfgFile)

The format of XmlCfgFile is as follow:

<?xml version="1.0" standalone="no" encoding="utf-8" ?>

```
<ExportModuleInfo    ExportModuleDir=D:\Work\VS_NEW\Examples>
   <SRPFSEngineBasicBCEditModule>
        <DriveClass/>
   </SRPFSEngineBasicBCEditModule>
   <SRPFSEngineBCModule>
        <FileToolClass/>
   </SRPFSEngineBCModule>
   <SRPFSEngineBasicModule>
        <DriveClass/>
        <DirectoryClass/>
        <FileClass/>
   </SRPFSEngineBasicModule>
</ExportModuleInfo>
```

5.6.21 object's EditLog/CheckPoint[Reserved]

With the checkpoint mechanism to provide high reliability.

1. **_SetLog (Object/SysRootItem,true/false)**

   java: void _SetLog(Object Arg,boolean BeginOrEnd)

Set the object or service item to start to log, then any changes of object attribute, child object created or deleted, child object attribute change will be recorded in log file, which can be recover by ApplyLog.

2. **_SetLogFile(FileName)**

   java: void _SetLogFile(String FileName)

Set Log file name which default path is service directory.

3. **FileName =_GetLogFile()**

   java: String _GetLogFile()

Get log file name.

4. **_ClearLog()**

   java: void _ClearLog()

Clear Log content.

5. **bool =_ApplyLog()**

   java: boolean _ApplyLog()

Recover.

5.6.22 Authorize

1. **bool= _IsServiceRegistered ()**

   java: boolean _IsServiceRegistered()

   whether service is registered. Do not call the function frequently.

2. **void= _CheckPassword(true/false)**

   java: boolean _CheckPassword(boolean Flag)

If check password is set to false, then when _GetService of ServiceGroupObject is called, the cle will not check user password.

5.7 Service item object

5.7.1 Attribute

1. **SrvItem._Name**
   service item name
2. **SrvItem._Service**
   Corresponding service of the service item

5.7.2 Functon

1. **BOOL=_IsSync () [valid at client]**
   java: boolean _IsSync()
   Whether service item is sync.
2. **BOOL=_WaitSync () [valid at client]**
   java: boolean _WaitSync()
   Wait service item becomes sync. Return False indicates failure, and True indicates success.
3. **int _GetGroupSyncStatus(GroupIndex)**
   java: int _GetGroupSyncStatus(int GroupIndex)
   Get syncgroup status.
4. Active set table=**_GetActiveSet ()**
   java: Object[] _GetActiveSet()
   Get active set of servie item. Return None or active set.
5. **_SetActiveSet (**active set**)**
   java: void _SetActiveSet(Object...Arg)
   Set service item active set.
6. **_SetClientActiveSet (ClientID,** active set**)**
   java: void _SetClientActiveSet(int ClientID,Object...Arg)
   Set active set of service item for client side. If the service item is not active at client, then it is activated automaticly.
   ClientID can be obtained from Object._ClientID.
7. **_QueryFirstGroupObject(GroupIndex)**
   java: StarObjectClass _QueryFirstGroupObject(int GroupIndex)
   Get first object of the service item
   GroupIndex should not be set to 0;
8. **_QueryNextGroupObject()**
   java: StarObjectClass _QueryNextGroupObject()
   Get next object of the service item

5.7.3 Callback funcion

1. **_OnClientToSync(ClientID,SyncGroupIndex)**

When client changes to sync, the callback function is called at server side.

```
def SrvItem_OnClientToSync( self,ClientID,SyncGroupIndex ) :

SrvItem._OnClientToSync = SrvItem_OnClientToSync
```

```
function SrvItem:_OnClientToSync( ClientID,SyncGroupIndex )

end
```

```
java/c#:
public void _OnClientToSync(int ClientID,int SyncGroupIndex )
{
}
```

5.8 Object's predefined attribute and function

1. **Object._Service**
   The corresponding service of the object
2. **Object._Parent**
   Get parent object
   Parent object may be service item object, which can be distinguished by function SrvGroup._IsObject.If the function returns True, then it is normal object, or else is service item object
3. **Object._Index**
   Get queue name of parent object.
4. **Object._Order**
   Get order of the object in queue of parent object. Returns 0 for no order. Order starts from 1.
5. **Object._Class**
   Get class of the object
6. **Object._ThisService**
   Is the object of this service**[==True this service,False other service]**
7. **Object._ActiveService**
   Is the object of active service**[==True active service,False, not active service]**
8. **Object._ClassID**
   Get class ID of the object, is string.
9. **Object._ID**
   Get object ID,is string.
10. **Object._Next**
    Get next object in the same queue of the parent
11. **Object._Prev**
    Get previous object in the same queue of the parent
12. **Object._NextEx**

Get next object with same name

13. **Object._PrevEx**

Get previous object with same name.

14. **Object._Name**

object name[read and write]

15. **Object._ClientID**

Object's ClientID（the value represents client machine ID at server side）

16. **Object._SysRootItem**

The service item of the object

17. **Object._SyncStatus**

Object sync status

18. **Object._SyncGroup**

Object syncgroup, can be read or write

For write:

For global dynamic object, static object, or client dynamic object, the function is only valid at server. Object's syncgroup starts from 1.

Object must belong to one service item, otherwise the function takes no effect.

If parent object syncgroup has been set, then child object syncgroup can not be set , which is to avoid sync problem if they are different.

19. **Bool = Object._IsLocalControl**

Return True if object is under local control, otherwise is under server control.

For static object, global dynamic object or client object, the function returns True at server side.

20. **Bool = Object._IsRemoteCreate**

If returns true, then the object is created by local side, or else is created by remote side.

21. **Bool = Object. _IDInParent[read or write]**

Object tag in parent object, 0 is reserved.

22. **Object._Layer**

Object layer of class, start from 1. If the layer is 1, then the object has no parent class.

23. **Object._RemoteID**

Client ID of remotecall, valid at server. the value is set when client start remotecall. The remotecall function can use this value to determine which client initiates the remotecall.

24. **Object._RemoteSourceTag**

Source tag, is used to determine the source type.

25. **Object._RemotePrivateTag**

Client private tag, which is used to determine the legality of the client.

26. **Object._RemoteCallID**

Remotecall ID.

27. **Bool = Object._IsRemoteCall**

Whether object is in remotecall process.

28. **Bool = Object. _RemoteCallName**

Remotecall function name, used in response.

29. **Object._AllocType**

Object alloc type

30. **Object._SaveFlag**

can be read or write. Takes value from 0,1,2, corresponding to

SAVE_NONE

SAVE_LOCAL

SAVE_GLOBAL

31. **Object._EditMode**

Can be read or write, the value is True or False, the value should be set when the object is edited.

32. **Object._IsValid**

Returns nil indicates the object is invalid.

33. **Object._WebServiceFlag**

The object is published for webservice. Returns true indicates public [read or write]

Function

1. **_V/_F()**

print object attributes or lua functions

For Java/c#, should use _V/_F(null).

2. **_V/_F (FuncName)**

java: void _V(String Name)

java: void _F(String Name)

print description information of attribute or lua function

3. **_E ()**

java: void _E()

print output event name and attribute.

4. **_S()/_S(ScriptName)**

java: void _S(String Name)

print script list or script contents

5. **_NV()**

java: void _NV()

print object's name value paires.

6. **_GetChild (ObjectName)**

java: StarObjectClass _GetChild(String Name)

Get child object

7. **_GetChildByID (AttributeName,int IDInParent)**

java: StarObjectClass _GetChildByID(String AttributeName,int IDInParent)

Get child object based on IDInParent from special queue.

8. **_FirstInst (QueryRecord)**

java: StarObjectClass _FirstInst(StarQueryRecordClass QueryRecord)

get first instance.

9. **_NextInst (QueryRecord)**

java: StarObjectClass _NextInst(StarQueryRecordClass QueryRecord)

get next instance

10. **_QueryClose (QueryRecord)**

java: void _QueryClose(StarQueryRecordClass QueryRecord)

close query.

11. **Child,int Context =_FirstActiveChild ()**

java: Object[] _FirstActiveChild()

get first active child.

12. **Child, int Context =_NextActiveChild (int Context)**

java: Object[] _NextActiveChild(int Context)

get next active child.

13. **bool = _IsInActiveSet ()**

java: boolean _IsInActiveSet()

Whether object is in active set of its service item.

14. **bool=_IsInst (Object)**

java: boolean _IsInst(StarObjectClass Obj)

Input object is instance of the object or not.

15. **bool=_IsDirectInst (Object)**

java: boolean _IsDirectInst(StarObjectClass Obj)

Input object is direct instance of the object or not.

16. **bool=_IsChild (Object)**

java: boolean _IsChild(StarObjectClass Obj)

Input object is child of the object.

17. **bool=_IsThisClient ()**

java: boolean _IsThisClient()

Object belongs to this client or not, valid at client side.

18. **_SetPrivateValue (int ClassLayer,int Index,Value)**

java: void _SetPrivateValue(int ClassLayer,int Index,Object Value)

Set object private value

19. **Value =_GetPrivateValue (ClassLayer,Index)**

java: Object _GetPrivateValue(int ClassLayer,int Index)

Get object private value

20. **_InsertToSDT ( )**

java: void _InsertToSDT( )

Insert object to service table

21. **_DelFromSDT( )**

java: void _DelFromSDT( )

Delete object from service table.

22. **Object = _QueryFirstInstFromSDT (QueryRecord)**

java: StarObjectClass _QueryFirstInstFromSDT(**QueryRecord**)

Query first instance of object from service table.

23. **Object = _QueryNextInstFromSDT (QueryRecord)**

java: StarObjectClass _QueryNextInstFromSDT(**QueryRecord**)

Query next instance of object from service table.

24. **_ChangeParent (ParentObject,QueueAttrName)**

java: void _ChangeParent(StarObjectClass ParentObject,String QueueAttrName)

Change parent object. **ParentObject may be object or service item. If it is service item, then QueueAttrName may be omitted.**

Calling format:

Object:_ChangeParent (ParentObject)

Object:_ChangeParent (ParentObject, "QueueAttrName");

For global static,dynamic, or client object, the function is invalid at debug.

When change parent object, the object status will be reset. If object is active then the following event will be generated:

ONDEACTIVATING,ONDEACTIVATE,ONDESTORY,ONCREATE,ONACTIVATING,ONACTIVATE.

25. **bool=_ActiveCmd (int Cmd)**

java: boolean _ActiveCmd(int Cmd)

Called at server, return value is bool type.

26. **Cmd =_GetActiveCmd ()**

java: int _GetActiveCmd()

Called at server, return object's active command

27. **bool=_ActiveClient(int ClientID)**

java: boolean _ActiveClient(int ClientID)

Called at server, active object at client

If ClientID equals to 0, then object is activated on all client.

28. **_DeactiveClient (int ClientID)**

java: void _DeactiveClient(int ClientID)

Called at server, deactive object at client,no return value.

If ClientID equals to 0, then the object is deactivated on all client.

29. **_Active ( )**

java: boolean _Active( )

activate object,no return value.

30. **_Deactive ( )**

java: void _Deactive( )

deactivate object,no return value.

31. **Bool =_IsActive ()**

java: boolean _IsActive()

whether object is active.

32. **Object=_QueryFirstActiveInst (QueryRecord)**

java: StarObjectClass _QueryFirstActiveInst()

get first active instance of object from current active service.

33. **Object=_QueryNextActiveInst (QueryRecord)**

java: StarObjectClass _QueryNextActiveInst()

get next active instance of object from current active service.

34. **int RefValue = _RegEventFunction(InObject,EventName, Object_Function)**

java: int _RegEventFunction(StarObjectClass InObject,String EventName,String Object_Function)

Register object event process function, Function Object_Function processes InObject.EventName event.

```
def Object_Function ( self,Event,Args…) :
        Print(self, Event )
        return True,… (…values)
```

```
function Object:Function ( Event,Args…)
        print(self, Event )
        return true,… (…values)
end
```

```
java/c#:
      public Object[] _OnObjectEvent(StarObjectClass self, Hashtable Event, Args)
      {
       return null; or Object[]{true,…};
      }
```

Parameter event contains four values :_SrcObject,_DesObject,_EventID, _ServiceGroupID.
Args depends on event definition of object.

35. **_UnRegEventFunction(InObject,EventName, int RefValue)**

java: void _UnRegEventFunction(StarObjectClass InObject,String EventName, int RefValue)

Unregister object event process function.

36. **ReturnValue = _ProcessEvent(EventName,…)**

java: Object[] _ProcessEvent(String EventName,Object...Args)

Create object event,…is args.

Types of argments and return value may be string, integer,bool,or float.

37. **_PostProcessEvent(EventName,…)**

java: void _PostProcessEvent(String EventName,Object...Args)

Create object event,…is args, no return value.

Types of argments may be string, integer,bool,or float.

38. **string EventID =_EventID(EventName)**

java: String _EventID(String EventName)

Get ID of the event.

39. **TimerID=_SetTimer (Ticket, Object. Func, Arg1, Arg2)**

java: int _SetTimer(int Ticket,String Object_Func,int Arg1,int Arg2)

Function should be defined within the object

Ticket is interval which unit is 10ms;

Calling format : a = Object.SetTimer(100,Func, 0,0)

Args should be number, returns TimerID

```
def Object_Func(self,TimerID,Arg1,Arg2) :
```

```
function Object:Func(TimerID,Arg1,Arg2)

end
```

```
java/c#:
    public void TimerFunc(StarObjectClass self,int TimerID,int Arg1,int Arg2)
```

```
{
}
```

40. _KillTimer (TimerID)

```
java: void _KillTimer(int TimerID)
```

TimerID is the ID of the timer.

41. _New(QueueAttrName,ParentObject, ObjectName,InitScript)/_NewEx (ObjectID,QueueAttrName,ParentObject, ObjectName,InitScript)

```
java: StarObjectClass _New(Object...Args)
java: StarObjectClass _NewEx(Object...Args)
```

ParentObject is parent object,QueueAttrName is queue name of the parent object. ParentObject may be object or servie item. If it is service item, then QueueAttrName may be ““.

Calling format:

Object._New(“QueueAttrName”,ParentObject)

Object._New(ParentObject)

Object._New() : no parent object

If there is no parent object, then the object is created under context of the active service. If the active service is changed, then the object will be freed by CLE.

InitSctipt is lua script used to assign attribute or perform object’function. Each sentence is seperated by ’;’.

For example :Object._New(““,”attr1=value; attr2=value2; func();”)

’;’ should not be omitted.

syntax format:

sentences are seperated by ’;’

for each sentence,

If the first char is’$’, then the following ’$O’ will be replaced by Object before executed.

For example:”$$O:_Active();” is translated to Object:_Active() .

If not, then the following rule takes effect.

If contains ’=’, then translate to Object. XXXX

otherwise translate to Object: XXXX

$O is not case sensitive.

42. _NewGlobal(ClientID, QueueAttrName,ParentObject, ObjectName,InitScript)/_NewGlobalEx(ObjectID,ClientID, QueueAttrName,ParentObject, ObjectName,InitScript)

```
java: StarObjectClass _NewGlobal(Object...Args)
java: StarObjectClass _NewGlobalEx(Object...Args)
```

ParentObject is parent object, QueueAttrName is queue name of parent object. ClientID is ID of the client. ClientID should be set to 0 at client side and set to ID of client at server side. If the object is not created for special client, Client ID should also be set to 0. ParentObject may be object or servie item. If it is service item, then QueueAttrName may be ““.

ClientID may be omitted.

Calling format:

Object._NewGlobal(“QueueAttrName”,ParentObject)

Object._NewGlobal(ParentObject)

Object._NewGlobal(ClientID,"QueueAttrName",ParentObject)
Object._NewGlobal(ClientID,ParentObject)
The function is invalid at debug.

InitSctipt is lua script used to assign attribute or perform object'function. Each sentence is seperated by ';'.
For example :Object._New("","attr1=value; attr2=value2; func();")
';' should not be omitted.
syntax format:
sentences are seperated by ';'
for each sentence,
If the first char is'$', then the following '$O' will be replaced by Object before executed.
For example:"$$O:_Active();" is translated to Object:_Active() .
If not, then the following rule takes effect.
If contains '=', then translate to Object. XXXX
otherwise translate to Object: XXXX
$O is not case sensitive.

43. _NewClient(ClientID, QueueAttrName,ParentObject, ObjectName,InitScript))/_NewClientEx(ObjectID,ClientID, QueueAttrName,ParentObject, ObjectName,InitScript)

java: StarObjectClass _NewClient(Object...Args)
java: StarObjectClass _NewClientEx(Object...Args)

ParentObject is parent object, QueueAttrName is queue name of parent object. ClientID is the ID of the client. ClientID should be set to 0 at client side and set to ID of client at server side. ParentObject may be object or service item. If it is service item, then QueueAttrName may be "".
ClientID may be omitted.
Calling format:
Object._NewClient("QueueAttrName",ParentObject)
Object._NewClient (ParentObject)
Object._NewClient (ClientID,"QueueAttrName",ParentObject)
Object._NewClient (ClientID,ParentObject)
The function is invalid at debug.

InitSctipt is lua script used to assign attribute or perform object'function. Each sentence is seperated by ';'.
For example :Object._New("","attr1=value; attr2=value2; func();")
';' should not be omitted.
syntax format:
sentences are seperated by ';'
for each sentence,
If the first char is'$', then the following '$O' will be replaced by Object before executed.
For example:"$$O:_Active();" is translated to Object:_Active() .
If not, then the following rule takes effect.

If contains '=', then translate to Object. XXXX

otherwise translate to Object: XXXX

$O is not case sensitive.

44. _Change("AttrName",…)

java: void _Change(String AttrName,Object Arg)

Chane object attribute at local

45. _MarkChange("AttrName")

java: void _MarkChange(String AttrName)

For attribute change, it will sync to client at next service frame. [for global object, static object, and client object],valid at server.

46. bool = _WaitMalloc()

java: boolean _WaitMalloc()

Called at client side to wait server to confirm the allocation of global or client object. If the return value is false, then the object is not permitted by server and has been freed or the connection to server is closed.

47. _Copy(SrcObject)

java: void _Copy(StarObjectClass SrcObject)

Does not copy pointer and local attribute.

The two objects should belong to the same class.

48. _Free()

java: void _Free()

Free object. For global, static or client object, the function is invalid at debug.

At client side, the object is not freed intermidately because the request is send to server.

49. _DeferFree()

java: void _DeferFree()

Free object. For global, static or client object, the function is invalid at debug.

The object is freed at next Ticket. At client side, the process is the same as FreeObject.

50. Bool=_IsInFree()

java: boolean _IsInFree()

Returns True if object is being freed.

51. int RefValue =_RegFileCallBack(Object. _RegFileCallBack)

java: int _RegFileCallBack(String Object_RegFileCallBack)

The callback function should be defined within the object. The prototype is as follows:

```
def function (self, uMsg, DataFile,ReceiveOrSendOffset,
DataSize, FileName Or Object, UniqueDataUnitID,Version ) :

Object:_RegFileCallBack( function (self, uMsg, DataFile,ReceiveOrSendOffset,
DataSize, FileName Or Object, UniqueDataUnitID,Version )  … end )

java/c#:
public void Obj_RegFileCallBack(StarObjectClass self,int uMsg, boolean
DataFile,int ReceiveOrSendOffset,int DataSize, Object FileName_Object, int
UniqueDataUnitID,String Version )
```

```
{
}
```

52. **_UnRegFileCallBack(RefValue)**

java: void _UnRegFileCallBack(int RefValue)

53. **RetValue = _Call(FuncName,…)**

java: Object _Call(String FuncName,Object...Args)

Call local C/C++ function, if the body does not exist, then _OnCall event is trigered.

54. **void = _RemoteCall(ClientID, FuncName,…) / _RemoteCallEx(ExcludeClientID, FuncName,…)**

java: void _RemoteCall(int ClientID,String FuncName,Object...Args)

java: void _RemoteCallEx(int ExcludeClientID,String FuncName,Object...Args)

ClientID is ignored at debug and client side; At server, if ClientID equals 0, then the call will be proceed by all the client, or else the special client.

FuncName is the function name. … is input argments, which may be number,bool,parapkg,binbuf,or string.

Calling format:

Object._RemoteCall("FuncName",Arg1,Arg2..)

Object._RemoteCall(ClientID,"FuncName",Arg1,Arg2,…)

Object._RemoteCallEx(ExcludeClientID,FuncName,…)

_RemoteCallEx indicates all client except ExcludeClientID, and only valid at server.

Object._RemoteCallEx(ExcludeClientID,"FuncName",Arg1,Arg2,…)

55. **RetCode, RetVal1,RetValue,… =_SRemoteCall(WaitTime(ms),ClientID,FuncName,…) valid at client**

java: Object[] _SRemoteCall(int WaitTime,int ClientID,String FuncName,Object...Args)

FuncName is the function name. … is input argments, which may be number,bool,parapkg,binbuf,or string.

RetCode = 0 for succeed

Calling format:

Object._SRemoteCall(0,0,"FuncName",Arg1,Arg2..)

WaitTime: unit is ms, if equals to 0, then will wait for event.

56. **_ARemoteCall(WaitTime(ms),ClientID,CallBackFunc, FuncName, (int)Para,…)**

java: void _ARemoteCall(int WaitTime,int ClientID,String CallBackFunc,String FuncName,int Para,Object...Args)

FuncName is the function name. … is input argments, which may be number,bool,parapkg,binbuf,or string.

Calling format:

Object._ARemoteCall(0,0,CallBackFunc,"FuncName",Para,Arg1,Arg2..)

```
def CallBackFunc( self, RetCode,ServiceGroupID,Para, RetValue) :
```

```
function Object:CallBackFunc( RetCode,ServiceGroupID, Para, RetVal1)

end
```

java/c#:

```
public void ARemoteCallBack(StarObjectClass self,int RetCode,int ServiceGroupID,int Para,Object RetValue)
{
}
```

WaitTime: unit is ms, if equals to 0, then will wait forever.

57. Value=_GetRemoteAttach(ParaName)

java: Object _GetRemoteAttach(String ParaName)

Get attach parameters of the request, which is used for VSRCALLSRC_WEBSERVICE, and defines as follow:

```
struct StructOfVSRemoteCallRequestAttach_WebService{
    struct StructOfSRPComm_HttpOnRequest *HttpRequest;
    class ClassOfSRPSXMLInterface *SoapInfo;
    VS_CHAR *OperationName;
    struct{
        VS_ULONG MimeDataSize;
        VS_INT8    *MimeDataBuf;
    }MimeData;   ->    map to BinBuf
    };
```

58. _SetDeferRspFlag ()

java: void _SetDeferRspFlag()

Set the response will be delayed, the function should be executed at the function being called.

59. _SetRetCode (int RetCode)

java: void _SetRetCode(int RetCode)

Set return code for response. the function should be executed at the function being called.

60. _SetRemoteRspAttach (RemoteAttach) [Reserved]

java: void _SetRemoteRspAttach(Object...RemoteAttach)

The function is valid for VSRCALLSRC_WEBSERVICE:

```
struct StructOfVSRemoteCallResponseAttach_WebService{
    class ClassOfSRPSXMLInterface *SoapInfo;
    struct{
        VS_ULONG MimeDataSize;
        VS_INT8    *MimeDataBuf;
    }MimeData;
    VS_CHAR    *MimeContentType;
};
```

SoapInfo: is used to define Envelop,Head,and Body,Operation, if the element exists, it should be integrated.

for example:

Object:_SetRemoteRspAttach(SoapInf_XMLInterface,MiniData_BinBuf, MimeContentType)

java/c# should use Hashtable.

61. _RemoteCallRsp (int ClientID, int RemoteCallID, String RemoteCallName,int

RemoteSourceTag,int RetCode, Object[] RemoteAttach,int RetType, Object RetValue)

java: void _RemoteCallRsp(int ClientID,int RemoteCallID,String RemoteCallName,int RemoteSourceTag,int RetCode,Object[] RemoteAttach,int RetType,Object RetValue)

If the remotecall response is set to deferred which may be used to send response to client.

ClientID: get from **Object._RemoteID** in the function being called.

RemoteCallID: get from **Object._RemoteCallID** in the function being called.

RemoteSourceTag: get from **Object._RemoteSourceTag** in the function being called.

RetValue: supported types include number,bool,parapkg,binbuf,string and object.

RemoteAttach: if not exist,should set to None

If returns Table,then RetType should be set to 255.

string ,Service.TYPE_CHARPTR

parapkg,Service.TYPE_PARAPKG

object,Service.TYPE_OBJPTR

62. Bool = _FillSoapRspHeader (SXml)

java: boolean _FillSoapRspHeader(StarSXmlClass SXml)

fill SOAP response header, the contents is:

```
<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ns1="urn:starcore-XXXXX">
   </SOAP-ENV:Envelope>
```

63. Bool =_CreateFunc(FuncName,"FuncScript")

java: boolean _CreateFunc(String FuncName,String FuncScript)

Valid at client and server, which is used to create lua script function.

FunctionScript format:

script starts by prefix '@', follows by script interface name, and a space. then the script body.

if '@' does not exist, default is lua script.

script interface length should be less than 15 bytes.

note: only support lua in current version.

64. Bool =_CreateFuncEx(FuncName,"ScriptFileName")

java: boolean _CreateFuncEx(String FuncName,String ScriptFileName)

Valid at client and server, which is used to create lua script function through file.

FunctionScript format:

script starts by prefix '@', follows by script interface name, and a space. then the script body.

if '@' does not exist, default is lua script.

script interface length should be less than 15 bytes.

note: only support lua in current version.

65. _DelFunc(FuncName)

java: void _DelFunc(String FuncName)

Valid at client and server, which is used to delete script.

66. **Bool =_SaveToFile(FileName,char *PassWord,int SaveFlag, Boolean SaveNameValue)**

java: boolean _SaveToFile(String FileName,String PassWord,int SaveFlag,boolean SaveNameValue)

save object to file.

The function can not save static object and client object.

67. **bool =_LoadFromFile(FileName, char *Password, bool LoadAsLocal ,bool LoadNameValue,bool UpdateFlag, bool StaticDataUseFile)**

bool =_LoadFromBuf(BinBuf, char *Password, bool LoadAsLocal ,bool LoadNameValue,bool UpdateFlag, bool StaticDataUseFile)

java: boolean _LoadFromFile(String FileName, String Password, boolean LoadAsLocal,boolean LoadNameValue,boolean UpdateFlag, boolean StaticDataUseFile)

java: boolean _LoadFromBuf(StarBinBufClass BinBuf,String Password,boolean LoadAsLocal,boolean LoadNameValue,boolean UpdateFlag,boolean StaticDataUseFile)

Restore object from file or binbuf. If UpdateFlag is true, ID of object is updated.

68. **_DeferLoadFromFile(FileName, char *Password, bool LoadAsLocal ,bool LoadNameValue,bool UpdateFlag, bool StaticDataUseFile)**

java: void _DeferLoadFromFile(String FileName,String Password,boolean LoadAsLocal,boolean LoadNameValue,boolean UpdateFlag,boolean StaticDataUseFile)

Restore object from file or binbuf. If UpdateFlag is true, ID of object is updated.

for example:

VDisk._LoadFromFile("d:\\Disk.Obj", "123",True, True,True)

Restore object will lead to object being deactivated,and then reactivated. If is restored in object's active event, DeferLoadFromFile function should be used, which will restore object at next Ticket(10ms).

when handles the active event, application should check the event paramater LParam. I it equals to 1, then should not continue to call this function.

69. **_ResetLoad()**

java: void _ResetLoad()

clear all child objects being loaded

//Name Value functions should be called at server and client.If they are called at server, the changes will be sync to client automaticly.If they are called at client, the changes is at local.

70. **Bool =_SetNameInt("Name",int,bool LocalChange)**

java: boolean _SetNameInt(String Name,int Value,boolean LocalChange)

Set integer name value.

71. **Int =_GetNameInt("Name",DefaultInt)**

java: int _GetNameInt(String Name,int DefaultInt)

Get integer name value.

72. **Bool =_SetNameFloat("Name",float,bool LocalChange)**

java: boolean _SetNameFloat(String Name,double Value,boolean LocalChange)

Set float name value.

73. **float =_GetNameFloat(“Name”,DefaultFloat)**

java: double _GetNameFloat(String Name,double DefaultFloat)

Get float name value.

74. **Bool =_SetNameStr(“Name”,Str,bool LocalChange)**

java: boolean _SetNameStr(String Name,String Str,boolean LocalChange)

Set string name value.

75. **Str =_GetNameStr(“Name” ,DefaultStr)**

java: String _GetNameStr(String Name,String DefaultStr)

Get string name value.

76. **Bool =_SetNameTime(“Name”,Time,bool LocalChange)**

java: boolean _SetNameTime(String Name,StarTimeClass Tm,boolean LocalChange)

Set datetime name value.

77. **Time =_GetNameTime(“Name”,DefaultTime)**

java: StarTimeClass _GetNameTime(String Name,StarTimeClass DefaultTm)

Get datetime name value.

78. **_FreeNameValue(“Name”)**

java: void _FreeNameValue(String Name)

free name value, **Object._FreeNameValue() means free all name value.**

79. **Type =_GetNameValueType(“Name”)**

java: int _GetNameValueType(String Name)

Get type of name value.

80. **name value change call back**

```
def Object_OnNameValueChange(self, Name,NameHashValue) :

Object._OnNameValueChange = Object_OnNameValueChange
```

```
function Object:_OnNameValueChange(Name,NameHashValue)

end
```

```
function _OnNameValueChange(self,Name,NameHashValue)

end
```

java/c#:

```
public void _OnNameValueChange(StarObjectClass self,String Name,int NameHashValue)
{
}
```

81. **simple format to set name value(name is started by ’___’,three ‘_’), LocalChange is False as default.**

```
Object.___Name = Value
Value = Object. ___Name
```

82. attribute change callback

```
def Object_OnChange(self,AttributeName) :

Object_OnChange = Object_OnChange
```

```
function Object:_OnChange(AttributeName)

end
function _OnChange (self, AttributeName)

end
```

java/c#:

```
public void _OnChange(StarObjectClass self,String Name)
{
}
```

object static data management.

83. bool =_CanSetStaticData(Size)

java: boolean _CanSetStaticData(int Size)

whether permits to set object static data, valid at client.

84. String DataVersion =_SetStaticData(StaticAttributeName,BinBuf)

java: String _SetStaticData(String StaticAttributeName,StarBinBufClass BinBuf)

Update object static data. If succeed, the function returns static data version. Which should be set to corresponding attribute of the object. If fails, the function returns None.

StaticAttributeName: is attribute name of static data of object

BinBuf: binary buffer.

85. DataVersion =_SetStaticDataEx(StaticAttributeName, DataSize,Offset, FileName)[Reserved]

java: String _SetStaticDataEx(String StaticAttributeName,int DataSize,int Offset,String FileName)

StaticAttributeName is attribute name of static data of object

86. DataVersion =_GetStaticData(StaticAttributeName,BinBuf,DataVersion, (BOOL) AutoDownLoad)

java: String _GetStaticData(String StaticAttributeName,StarBinBufClass BinBuf,String DataVersion,boolean AutoDownLoad)

StaticAttributeName is attribute name of static data of object

BinBuf: binary buffer.

DataVersion may be ““, which means not care version.

If fails, the function returns None.

87. bool =_WaitGetStaticData (StaticAttributeName,Object_CallBack, bool WaitFlag)

java: boolean _WaitGetStaticData(String StaticAttributeName,String Object_CallBack,boolean WaitFlag)

If returns False, then the static data is not downloaded or error occurred in download process.

WaitFlag=False, query whether to wait upload or download.

88. bool =_WaitSetStaticData (StaticAttributeName, Object_CallBack, bool WaitFlag)

java: boolean _WaitSetStaticData(String StaticAttributeName,String Object_CallBack,boolean WaitFlag)

If returns False, then the static data is not downloaded or error occurred in download process.

WaitFlag=False, query whether to wait upload or download.

```
def Object_CallBack(self,uMes,CurSize,MaxSize)

    return 0,1
```

```
function Object:CallBack(uMes,CurSize,MaxSize)

    return 0,1
end
```

```
java/c#:
public int CallBack(StarObjectClass self,int uMes, int CurSize, int MaxSize)
{
    return 0,1;
}
```

If returns 0, then process is continue. Otherwise the process is canceled.the return value takes effect in VSFILE_ONDOWNPROGRESS/ VSFILE_ONUPPROGRESS process.

89. bool=_SaveToLuaFunc (FileName,Funcname)

java: boolean _SaveToLuaFunc(String FileName,String Funcname)

The function can be used to create the lua script of object attribute value.

90. _Init ("attr1=value; attr2=value2; func();")

java: void _Init(String Arg)

Each sentence should be followed by';'.

syntax format:

sentences are seperated by ';'

for each sentence,

If the first char is'$', then the following '$O' will be replaced by Object before executed.

For example:"$$O:_Active();" is translated to Object:_Active() .

If not, then the following rule takes effect.

If contains '=', then translates to Object. XXXX

otherwise translate to Object: XXXX

for example:

"attr1=value; attr2=value2; func();"

$O is not case sensitive.

91. bool=_RemoteSend(ClientID, ParaPkg)

java: boolean _RemoteSend(int ClientID,StarParaPkgClass ParaPkg)

At server side, if ClientID equals to 0, then the message is sent to all client, and the object must be global object; If ClientID does not equal to 0, then the message is sent to the corresponding client;

At client side, the function means sending message to server, and ClientID is ignored.

At receive side, application may be define event handler Object_OnRemoteSend(self,Event) to process the message.

92. int RefValue= _WaitEvent(InObject,EventName, Object.Function,bool AutoDelete)
Object:_UnWaitEvent(InObject,EventName, RefValue)
Para1,…= _GetEventPara(int Event)

Only valid for lua.

Wait object event, when InObject.EventName event is triggered, the callback function is called.

```
function Object:Function (SrcObject,int Event)
        print(self, Event )
        return
end
```

5.9 object defined attributes and functions

5.9.1 dynamic script language

1. set or get object attribute

For global pointer attribute, should only get, Which will return the first object in the queue. The change will take effect at local. If it is needed to sync to client, should use function Object._Change("AttrName",value)

struct attribute should be changed with tuple or array.For example:

attribute P1 of Object is struct, which include two variables X and Y.

print(Object.P1.X,Object.P1.Y); lua/python

echo Object->P1->X,Object->P1->Y; php

Object.P1={2,3} lua

Object.P1=(2,3) python

Object->P1=array(2,3) php

2. call object function

Call lua functions of object directly using function name.

calling format:

Object.FuncName(….).

FuncName is function name, may be define in object' class. The function is lua function, which prototype is int FunctionName(void *LuaState).

5.9.2 static script language

1. set or get object attribute

Using _Get or _Set method, for example:
Value = Object._Get(attribute name)
Object._Set(attribute name,Value);

2. call object function

Object._Call(function name, args, …)

5.10 ParaPkg object

Support bool, integer, float, string , time, and binbuf.

5.10.1 attribute

1. ParaPkg._Number
The number of values in the package.

5.10.2 function

1. int _T(Index)
java: int _T(int Index)
Get type of value at Index, which defines as follow:

```
SRPPARATYPE_INVALID     0       //--invalid
SRPPARATYPE_INT         1       //--integer
SRPPARATYPE_FLOAT       2       //--float
SRPPARATYPE_BIN         3
SRPPARATYPE_CHARPTR     4       //--string
SRPPARATYPE_TIME    5       //--time
SRPPARATYPE_BOOL        6       //--bool
```

For C#, please use _TT(Index)

2. _Clear()
java: void _Clear()
Clear all values.

3. bool = _Exchange(DesIndex,SrcIndex)
java: boolean _Exchange(int DesIndex,int SrcIndex)
Exchange position of two values.

4. void _Del(Index)
java: void _Del(int Index)
Delete a value.

For lua, app can use Para[Index] to get or set values.

5. **Bool =_AppendFrom(SrcParaPkg)**

   java: boolean _AppendFrom(StarParaPkgClass SrcParaPkg)

   Duplicate values from SrcParapkg to ParaPkg.

6. **_GetUUID(Index)**

   java: String _GetUUID(int Index)

   Get UUID value of binbuf, which returns a string.

7. **int = _GetHash(Index)**

   java: int _GetHash(int Index)

   Get hash code of the value.

8. **BOOL =_SaveToFile(Index,FileName)**

   java: boolean _SaveToFile(int Index,String FileName)

   Save binbuf to disk file. The value at Index must be binbuf

9. **BOOL =_LoadFromFile(Index,FileName)**

   java: boolean _LoadFromFile(int Index,String FileName)

   Load binbuf from disk file.

10. **BOOL =_CopyBin(Index, SrcParaPkg,SrcIndex)**

    java: boolean _CopyBin(int Index,StarParaPkgClass SrcParaPkg,int SrcIndex)

    Duplicate binbuf from parapkg. The value at SrcIndex must be binbuf.

11. **Value =_Get (Index)**

    java: Object _Get(int Index)

    Get value from parapkg.For lua, app can use ParaPkg[Index].

12. **bool _Set(Index,Value)**

    java: boolean _Set(int Index,Object Value)

    Set value to parapkg, For lua, app can use ParaPkg[Index].

13. **Value =_GetTime (Index)**

    java: StarTimeClass _GetTime(int Index)

    Get time value from parapkg, For lua, app can use ParaPkg[Index].

14. **bool = _SetTime (Index,Value)**

    java: boolean _SetTime(int Index,StarTimeClass Value)

    Set time value to parapkg, For lua, app can use ParaPkg[Index].

15. **_SetChangeFlag (Index)**

    java: void _SetChangeFlag(int Index)

    Set field change flag.

16. **_SetChangeFlagEx ()**

    java: void _SetChangeFlagEx()

    Set all fields change flag.

17. **_ClearChangeFlag (Index)**

    java: void _ClearChangeFlag(int Index)

    Clear field change flag.

18. **_ClearChangeFlagEx ()**

    java: void _ClearChangeFlagEx()

    Clear all fields change flag.

19. **Bool =_IsChangeFlag (Index)**

java: boolean _IsChangeFlag(int Index)

whether the field is changed

20. **Bool =_IsChangeFlagEx()**

java: boolean _IsChangeFlagEx()

whether there is any field changed

21. **Bool =_SaveChangeToBuf(BinBuf)**

java: boolean _SaveChangeToBuf(StarBinBufClass BinBuf)

Pack changed fields to binbuf.

22. **Bool =_SaveChangeToBufEx(BinBuf)**

java: boolean _SaveChangeToBufEx(StarBinBufClass BinBuf)

Pack all fields to binbuf.

23. **Bool = _LoadChangeFromBuf (BinBuf)**

java: boolean _LoadChangeFromBuf(StarBinBufClass BinBuf)

Restore fields from binbuf.

24. **Bool = _FromDict (table/dict/tuple/array)**

java: boolean _FromDict(Hashtable dict)

Key and Value are saved at two adjacent fields

The type of Key may be integer or string, and type of Value may be bool,integer, float,or string.

25. **dict = _ToDict ()**

java: Hashtable _ToDict()

Restore table,dictionary,or array. If error occurs, the function returns nil/NULL/None.

26. **Bool = _FromTuple(tuple)**

java: boolean _FromTuple(Object[] tuple)

Only save Value.

27. **tuple = _ToTuple ()**

java: Object[] _ToTuple()

Restore table, tuple,or array. If error occurs, the function returns nil/NULL/None.

5.11 Binbuf object

Types supported:

```
#define SRPBINTYPE_BOOL     ((VS_UINT8)'z')
#define SRPBINTYPE_BYTE     ((VS_UINT8)'b')       //map to int
#define SRPBINTYPE_UBYTE    ((VS_UINT8)'B')      // map to int
#define SRPBINTYPE_CHAR     ((VS_UINT8)'c')       // map to int
#define SRPBINTYPE_SHORT    ((VS_UINT8)'h')       // map to int
#define SRPBINTYPE_USHORT ((VS_UINT8)'H')       // map to int
#define SRPBINTYPE_INT       ((VS_UINT8)'i')          // map to int
#define SRPBINTYPE_UINT     ((VS_UINT8)'I')         // map to int
#define SRPBINTYPE_LONG     ((VS_UINT8)'l')       // map to int
#define SRPBINTYPE_ULONG    ((VS_UINT8)'L')      // map to int
#define SRPBINTYPE_FLOAT    ((VS_UINT8)'d)                //double
```

```
#define SRPBINTYPE_STRING          ((VS_UINT8)'s')          //with the end of 0
#define SRPBINTYPE_STRINGEX        ((VS_UINT8)'S')          //without 0 at end
#define SRPBINTYPE_UNICODE         ((VS_UINT8)'u')           // with the end of 0
#define SRPBINTYPE_UNICODEEX       ((VS_UINT8)'U')          //without 0 at end
#define SRPBINTYPE_UTF8            ((VS_UINT8)'t')          // with the end of 0
#define SRPBINTYPE_UTF8EX          ((VS_UINT8)'T')          // without 0 at end
#define SRPBINTYPE_BIN         ((VS_UINT8)'r')          //binary
```

Input uses string with single char, for example :"o", "b",etc.

5.11.1 attribute

1. **BinBuf._Size**
   size of the memory allocated
2. **BinBuf._Offset**
   size of data
3. **BinBuf._Buf**
   buffer address

5.11.2 function

1. **_Init (BufSize)**
   java: void _Init(int BufSize)
   alloc memory with BufSize, the function clear existed data.
2. **_Clear ()**
   java: void _Clear()
   Clear existed data
3. **_ClearEx (Offset,Length)**
   java: void _ClearEx(int Offset,int Length)
   Clear data from Offset and Length size.
4. **Length=_Set (Offset,Length,Type,Object Value)**
   java: int _Set(int Offset,int Length,String Type,Object Value)
   Returns Length; if the function returns 0, then there is error occures. Length is reserved.
5. **object Value = _Get (Offset,Length,Type)**
   java: Object _Get(int Offset,int Length,String Type)
   If fails, returns None. Length is reserved.
6. **BOOL =_SaveToFile(FileName,TxtFileFlag = true/false)**
   java: boolean _SaveToFile(String FileName,boolean TxtFileFlag)
   Save binay data to diak file. If **TxtFileFlag=true**, then save as text file.
7. **BOOL =_LoadFromFile(FileName,TxtFileFlag = true/false)**
   java: boolean _LoadFromFile(String FileName,boolean TxtFileFlag)
   Load binary data from disk file. if **TxtFileFlag=true**, then load as text file.
8. **BOOL = _SetOffset (Offset)**
   java: boolean _SetOffset(int Offset)
   Set size of data.

9. **BOOL = _Expand (NewBufSize)**

java: boolean _Expand(int NewBufSize)

expand buffer size.

10. **BOOL = _Fill (Offset,Length,Char)**

java: boolean _Fill(int Offset,int Length,String Val)

fill with char

11. **BOOL = _PackObject(Object)/_UnPackObject(Object)**

java: boolean _PackObject(StarObjectClass Obj)

java: boolean _UnPackObject(StarObjectClass Obj)

pack object into binbuf

12. **BOOL = _ToUTF8()**

java: boolean _ToUTF8()

Change data coding to UTF8. The buffer should be a string. If the string is ended with 0, then the converted string is ended with 0. Otherwise, without 0.

13. **BOOL = _ToAnsi ()**

java: boolean _ToAnsi()

Change data coding to Ansi, The buffer should be a string. If the string is ended with 0, then the converted string is ended with 0. Otherwise, without 0.

14. **_InsertStr(Offset,String)**

java: void _InsertStr(int Offset,String Val)

15. int =**_FindStr(Offset,String)**

java: int _FindStr(int Offset,String Val)

If not find, returns a value less than 0

16. int = **_FindStri(Offset,String)**

java: int _FindStri(int Offset,String Val)

If not find, returns a value less than 0

17. **_Print (..)**

java: void _Print(String Arg)

print string to the tail of the buf.

18. **int Handle =_OpenFile(FileName,String Mode)**

java: int _OpenFile(String FileName,String Mode)

19. **Size =_GetFileSize(int Handle)**

java: int _GetFileSize(int Handle)

20. **Length =_ReadFile(int Handle,Offset,Length)**

java: int _ReadFile(int Handle,int Offset,int Length)

21. **Length =_WriteFile(int Handle,Offset,Length)**

java: int _WriteFile(int Handle,int Offset,int Length)

22. **_CloseFile(int Handle)**

java: void _CloseFile(int Handle)

23. **bool=_IsLightBuf()**

java: boolean _IsLightBuf()

24. **BOOL =_AnsiToUnicode (code,BytesPerChar)**

java: boolean _AnsiToUnicode(String code,int BytesPerChar)

Change data coding to unicode. The buffer should be a string. If the string is ended with 0,

then the converted string is ended with 0. Otherwise, without 0. Code will be ignored on windows,which may be take values fro :UCS2 UCS4 UTF-16 UTF-32 UTF-16BE UTF-16LE UTF-32BE UTF-32LE,BytesPerChar should be set to 2 on windows.

25. **BOOL =_UnicodeToAnsi (code,BytesPerChar)**

java: boolean _UnicodeToAnsi(String code,int BytesPerChar)

Change data coding to ansi, The buffer should be a string. If the string is ended with 0, then the converted string is ended with 0. Otherwise, without 0.Code is ignored on windows, which may be take values from: UCS2 UCS4 UTF-16 UTF-32 UTF-16BE UTF-16LE UTF-32BE UTF-32LE, BytesPerChar should be set to 2 on windows.

26. **lstring=_Read(Offset,Length)/java/c#:int Length=_Read(byte[] buf,int Offset,int Length)**

java: int _Read(byte[] Buf,int Offset,int Length)

27. **Length =_Write(Offset, lstring)/java/c#:int Length=_Write(int Offset,byte[] buf,int Length)**

java: int _Write(int Offset,byte[] Buf,int Length)

28. **Length = _WriteFromMemoryFile (Service,Offset, FileName)**

java: int _WriteFromMemoryFile(StarServiceClass Service,int Offset,String FileName)

Load from Memory Files of the service

29. **String= _GetMD5 ()**

java: String _GetMD5()

Get MD5

30. **Int = _GetHash()**

java: int _GetHash()

Get Hash Value.

5.12 SXML object

For Element,Text,Attribute,Comment, function returns integer, if does not extst, function returns 0 or NULL.

5.12.1 load and save

1. **Bool,ErrorInfo=_LoadFromFile(FileName)**

java: Object[] _LoadFromFile(String FileName)

2. **Bool,ErrorInfo=_LoadFromBuf(BinBuf)**

java: Object[] _LoadFromBuf(StarBinBufClass BinBuf)

3. **bool,ErrorInfo=_LoadFromBufEx(String) String UTF-8 format**

java: Object[] _LoadFromBufEx(String UtfArg)

4. **bool=_SaveToFile(FileName)**

java: boolean _SaveToFile(String FileName)

5. **bool=_SaveToBuf(BinBuf)**

java: boolean _SaveToBuf(StarBinBufClass BinBuf)

5.12.2 read

1. **string=_GetStandalone()**
   java: String _GetStandalone()
2. **string=_GetVersion()**
   java: String _GetVersion()
3. **string=_GetEncoding()**
   java: String _GetEncoding()
4. **int Element=_FindElement(string Value)**
   java: int _FindElement(String Value)
   suppport format such as “a.b.c.d”
5. **int Element=_FindElementEx(int ParentElement, string Value)**
   java: int _FindElementEx(int ParentElement,String Value)
6. **int Element=_FirstElement(int ParentElement)**
   java: int _FirstElement(int ParentElement)
   ParentElement may be 0
7. **Element=_NextElement(int Element)**
   java: int _NextElement(int Element)
8. **Element=_ParentElement(Element)**
   java: int _ParentElement(int Element)
9. **string Value=_GetElement(Element)**
   java: String _GetElement(int Element)
10. **string Value=_GetElementEx(Element) a.b.c.d format**
    java: String _GetElementEx(int Element)
11. **Bool,nsName,nsValue= _GetNs(Element)**
    java: Object[] _GetNs(int Element)
12. **sting nsValue=_GetNsValue(Element,nsName)**
    java: String _GetNsValue(int Element,String nsName)
13. **int Attribute=_FindAttribute(Element,Name)**
    java: int _FindAttribute(int Element,String Name)
14. **int Attribute=_FirstAttribute(Element)**
    java: int _FirstAttribute(int Element)
15. **int Attribute=_NextAttribute(Attribute)**
    java: int _NextAttribute(int Attribute)
16. **string Name=_GetAttributeName(Attribute)**
    java: String _GetAttributeName(int Attribute)
17. **string Value=_GetAttributeValue(Attribute)**
    java: String _GetAttributeValue(int Attribute)
18. **string=_GetSingleText(Element)**
    java: String _GetSingleText(int Element)
19. **int Text=_FirstText(Element)**
    java: int _FirstText(int Element)
20. **int Text=_NextText(Text)**
    java: int _NextText(int Text)

21. **string Value=_GetText(Text)**
    java: String _GetText(int Text)

5.12.3 change

1. **_SetDeclaration(Version,Encoding,Standalone)**
   java: void _SetDeclaration(String Version,String Encoding,String Standalone)
2. **_RemoveDeclaration()**
   java: void _RemoveDeclaration()
3. **Element=_InsertElementBefore(ParentElement,Element,Value)**
   java: int _InsertElementBefore(int ParentElement,int Element,String Value)
   ParentElement/Element may be 0
4. **Element=_InsertElementAfter(ParentElement,Element,Value)**
   java: int _InsertElementAfter(int ParentElement,int Element,String Value)
   ParentElement/Element may be 0
5. **_RemoveElement(Element)**
   java: void _RemoveElement(int Element)
6. **_SetElement(Element,Value)**
   java: void _SetElement(int Element,String Value)
7. **_SetNs (Element,nsName,nsValue)**
   java: void _SetNs(int Element,String nsName,String nsValue)
8. **int Text=_InsertTextBefore(ParentElement,int Text,Value,bool CDataFlag)**
   java: int _InsertTextBefore(int ParentElement,int Text,String Value,boolean CDataFlag)
   ParentElement/Text may be 0
9. **int Text=_InsertTextAfter(ParentElement, int Text,Value,bool CDataFlag)**
   java: int _InsertTextAfter(int ParentElement,int Text,String Value,boolean CDataFlag)
   ParentElement/Text may be 0
10. **_RemoveText(int Text)**
    java: void _RemoveText(int Text)
11. **_SetText(Text,Value)**
    java: void _SetText(int Text,String Value,boolean CDataFlag)
12. **int Comment=_InsertCommentBefore(ParentElement, int Comment,String Value)**
    java: int _InsertCommentBefore(int ParentElement,int Comment,String Value)
    ParentElement/Comment may be 0
13. **int Comment=_InsertCommentAfter(ParentElement, int Comment,Value)**
    java: int _InsertCommentAfter(int ParentElement,int Comment,String Value)
    ParentElement/Comment may be 0
14. **_RemoveComment(Comment)**
    java: void _RemoveComment(int Comment)
15. **_SetComment(Comment,Value)**
    java: void _SetComment(int Comment,String Value)
16. **_SetAttribute(Element,Name,Value)**
    java: void _SetAttribute(int Element,String Name,String Value)
17. **_RemoveAttribute(Element,Name)**

java: void _RemoveAttribute(int Element,String Name)

5.12.4 duplicate

1. **Element=_CopyElementBefore(ParentElement,Element,SrcElement)**
   java: int _CopyElementBefore(int ParentElement,int Element,int SrcElement)
   ParentElement/Element may be set to 0,SrcElement may belongs to another SXml object.
2. **Element=_CopyElementAfter(ParentElement,Element, SrcElement)**
   java: int _CopyElementAfter(int ParentElement,int Element,int SrcElement)
   ParentElement/Element may be set to 0,SrcElement may belongs to another SXml object.
3. **bool=_CopyChild(DesElement, SrcElement)**
   java: boolean _CopyChild(int DesElement, int SrcElement)
   SrcElement may belongs to another SXml object.
4. **bool=_Dup(SrcSXML)**
   java: boolean _Dup(StarSXmlClass SrcSXML)

5.13 FunctionPara object

5.13.1 function

1. **Number=_GetNumber()**
   java: int _GetNumber()
   get number of parameter
2. **Value= _GetValue (Index)**
   java: Object _GetValue(int Index)
   get value of parameter
3. **_Clear ()**
   java: void _Clear()
   clear all parameters
4. **bool= _SetValue (Index,Value)**
   java: boolean _SetValue(int Index,Object Value)
   set parameter value
5. **Result= _Call (Object,FunctionName)**
   java: Object _Call(StarObjectClass Obj ,String FunctionName)

5.14 CommInterface object

Running in the thread context of cle.

5.14.1 attribute

1. **CommInterface. _MsgProc**

message callback:

```
function CommInterface_MsgProc(self,uMes,Msg)

    return;
end
```

```
java/c#:
public void _MsgProc(int uMes,Object[] Msg)
{
}
```

uMes's value and Msg definition
CommInteface.TCP_ONCLOSE,Msg:{ConnectionID}
CommInteface.TCP_ONCONNECT,Msg:{ ServerConnectionID,ConnectionID, BinBuf_LocalSockAddr, BinBuf_PeerSockAddr, Result}
CommInteface.TCP_ONREAD,Msg:{ConnectionID}
CommInteface.TCP_ONWRITE,Msg:{ConnectionID}
CommInteface.UDP_ONREAD,Msg:{ConnectionID}
CommInteface.UDP_ONWRITE,Msg:{ConnectionID}
CommInteface.HTTP_ONSTART,Msg:{ConnectionID, FileSize, (char*)ResponseHeader}
CommInteface.HTTP_ONREAD,Msg:{ConnectionID}
CommInteface.HTTP_ONWRITE,Msg:{ConnectionID}
CommInteface.HTTP_ONFINISH,Msg:{ConnectionID,Bin_ResidualData}
CommInteface.HTTP_ONERROR,Msg:{ConnectionID, (char *)ErrorInfo}
CommInteface.HTTP_ONREQUEST,Msg:{ConnectionID, BinBuf_PeerSockAddr, RequestType, BoundaryNumber, ServiceGroupIndexOrName ,FileSize, (char*)FileName, (char*)ContentType, (char*)Cookie, BinBuf_BoundaryInfo, (char*)RequestHeader, BinBuf_RequestBody}
CommInteface.TIMER,Msg:{TimerID}
BinBuf_RequestBody: is a light-weight buffer.

2. CommInterface. _WebServerProc

```
WebServer callback:
function CommInterface_WebServerProc(self,uMes,Msg)

    return true/false(Result), true/false(continue)
end
```

```
java/c#:
public Object[] _WebServerProc(int uMes,Object[] Msg)
{
}
```

uMes's value and Msg definition
CommInteface.HTTP_ONREQUEST,Msg:{ConnectionID, PeerSockAddr, RequestType,

BoundaryNumber, ServiceGroupIndexOrName ,FileSize, FileName, ContentType, Cookie, BoundaryInfo, RequestHeader, RequestBody}
RequestType : CommInteface.HTTPREQUEST_GET / CommInteface.HTTPREQUEST_POST
CommInteface.HTTP_ONWRITE,Msg:{ConnectionID}
CommInteface.HTTP_ONFINISH,Msg:{ConnectionID}
BinBuf_RequestBody: is a light-weight buffer.

5.14.2 function

1. **ConnectionID=_TCPSetupServer(BufferPkgNum, LocalServerName, PortNumber)**
   java: int _TCPSetupServer(int BufferPkgNum,String LocalServerName,int PortNumber)

If fails,it returns 0.

2. **ConnectionID=_TCPSetupClient(BufferPkgNum, ServerName, PortNumber)**
   java: int _TCPSetupClient(int BufferPkgNum,String ServerName,int PortNumber)

If fails,it returns 0.

3. **Size=_TCPSend (ConnectionID,BinBuf,Offset,bool MoreData)**
   java: int _TCPSend(int ConnectionID,StarBinBufClass BinBuf,int Offset,boolean MoreData)

It returns the size of data has been sent. If the data is not sent completely, then application should wait message to continue send.

4. **Size=_TCPRecv (ConnectionID,BinBuf,Offset)**
   java: int _TCPRecv(int ConnectionID,StarBinBufClass BinBuf,int Offset)

return the size of data received

5. **Size=_TCPRecvLine (ConnectionID,BinBuf)**
   java: int _TCPRecvLine(int ConnectionID,StarBinBufClass BinBuf)

return the size of data received

6. **_TCPRelease (ConnectionID)**
   java: void _TCPRelease(int ConnectionID)
7. **ConnectionID=_UDPSetupServer(BufferPkgNum, LocalServerName, PortNumber)**
   java: int _UDPSetupServer(int BufferPkgNum,String LocalServerName,int PortNumber)

If fails,it returns 0.

8. **ConnectionID=_UDPSetupClient(BufferPkgNum)**
   java: int _UDPSetupClient(int BufferPkgNum)

If fails,it returns 0.

9. **Size=_UDPSend (ConnectionID,BinBuf,BinBuf_IP)**
   java: int _UDPSend(int ConnectionID,StarBinBufClass BinBuf,StarBinBufClass BinBuf_IP)

It returns the size of data has been sent. If the data is not sent completely, then application should wait message to continue send. BinBuf_IP is filled with destination address, using function _UDPSetSockAddr.

10. **Size=_UDPRecv (ConnectionID,BinBuf,BinBuf_IP)**
    java: int _UDPRecv(int ConnectionID,StarBinBufClass BinBuf,StarBinBufClass BinBuf_IP)

return the size of data received

11. **_UDPRelease (ConnectionID)**
    java: void _UDPRelease(int ConnectionID)
12. **Bool=_UDPSetSockAddr (Name,Port,BinBuf_IP)**

java: boolean _UDPSetSockAddr(String Name,int Port,StarBinBufClass BinBuf_IP)

13. **IpString=_GetIP (BinBuf_IP)**

java: String _GetIP(StarBinBufClass BinBuf_IP)

14. **IpPort=_GetPort (BinBuf_IP)**

java: int _GetPort(StarBinBufClass BinBuf_IP)

15. **ConnectionID=_HttpDownLoad (Url,FileName)**

java: int _HttpDownLoad(String Url,String FileName)

If fails,it returns 0.

16. **ConnectionID=_HttpUpLoad (Url,FileName, FileSize,string ContentType,** Boolean **MultiPartFlag,** string **SaveFileName)**

java: int _HttpUpLoad(String Url,String FileName,int FileSize,String ContentType,boolean MultiPartFlag,String SaveFileName)

If fails,it returns 0.

17. **ConnectionID=_HttpDownLoadEx (Url,FileName,** string **RequestHeader)**

java: int _HttpDownLoadEx(String Url,String FileName,String RequestHeader)

If fails,it returns 0.

18. **ConnectionID=_HttpUpLoadEx (Url,FileName, FileSize,** string **RequestHeader)**

java: int _HttpUpLoadEx(String Url,String FileName,int FileSize,String RequestHeader)

If fails,it returns 0.

19. **Size= _HttpSend (ConnectionID,BinBuf,Offset,bool MoreData)**

java: int _HttpSend(int ConnectionID,StarBinBufClass BinBuf,int Offset,boolean MoreData)

It returns the size of data has been sent. If the data is not sent completely, then application should wait message to continue send.

20. **Size=_HttpRecv (ConnectionID,BinBuf,Offset)**

java: int _HttpRecv(int ConnectionID,StarBinBufClass BinBuf,int Offset)

return the size of data received

21. **_HttpRelease (ConnectionID)**

java: void _HttpRelease(int ConnectionID)

22. **ConnectionID= _HttpServer (LocalServerName, PortNumber,MaxPostSize)**

java: int _HttpServer(String LocalServerName,int PortNumber,int MaxPostSize)

If fails,it returns 0.

23. **RspString= _FormatRspHeader (*RspInfo,*ServerInfo,*Connection,*ContentType,** int **ContentLength)**

java: String _FormatRspHeader(String RspInfo,String ServerInfo,String LConnection,String ContentType,int ContentLength)

/*FormatRspHeader("200 OK","Microsoft-IIS/5.1","Close","text/html",268);*/

24. **string ParaValue=_ParsePara(Info,ParaName);**

java: String _ParsePara(String Info,String ParaName)

//----Para=XXX&Para2=XXX

25. **RspCode,RspInfo=_GetResponseCode (BinBuf);**

java: Object[] _GetResponseCode(StarBinBufClass BinBuf)

26. **Str=_GetResponseStr (BinBuf,string Title);**

java: String _GetResponseStr(StarBinBufClass BinBuf,String Title)

/*-- Title
Date:
Connection:
Content-Type:
Content-Length:
and so on

27. **Lenght=_GetResponseLength (BinBuf);**

java: int _GetResponseLength(StarBinBufClass BinBuf)

```
Content-Length:
```

28. **bool=_GetResponseBody(BinBuf,BodyBinBuf);**

java: boolean _GetResponseBody(StarBinBufClass BinBuf,StarBinBufClass BodyBinBuf)

29. **ConnectionID=_HttpLocalRequest (int RequestType, FileName,** string **ContentType,** string **Cookie, BinBuf);**

java: int _HttpLocalRequest(int RequestType,String FileName,String ContentType,String Cookie,StarBinBufClass BinBuf)

//---BinBuf:Request Body, which may be nil.

30. **ConnectionID=_HttpLocalRequestEx (HtmlPlainText);**

java: int _HttpLocalRequestEx(String HtmlPlainText)

//---Input is complete Http request. In callback _MsgProc, using HTTP_READ message to get the response. The response is complete http response.

31. **TimerID= _SetupTimer(Interval,NumberOfValid)**

java: int _SetupTimer(int Interval,int NumberOfValid)

If fails,it returns 0. Interval unit is 10ms.

32. **_KillTimer (Timer)**

java: void _KillTimer(int Timer)

33. **_WebServerRelease (ConnectionID)**

java: void _WebServerRelease(int ConnectionID)

Used by HttpServer to close connection.

34. **RspString=_FormatRspHeaderEx (*RspInfo,*ServerInfo,*Connection,*ContentType, ContentLength,ExtendInfo)**

java: String _FormatRspHeaderEx(String RspInfo,String ServerInfo,String LConnection,String ContentType,int ContentLength,String ExtendInfo)

/*FormatRspHeaderEx("200 OK","Microsoft-IIS/5.1","Close","text/html",268, "Cookie: aa=aa");*/

35. **RspString= _HttpGetHeaderItem (string_Header, ItemIndex, ItemName)**

java: String _HttpGetHeaderItem(String string_Header,int ItemIndex,String ItemName)

ItemName may be ““,ItemIndex starts from 0

36. **RspString= _HttpGetHeaderSubItem (string_Item, SubItemIndex, SubItemName)**

java: String _HttpGetHeaderSubItem(String string_Item,int SubItemIndex,String SubItemName)

SubItemName may be ““,SubItemIndex starts from 0

SubItem is seperated by “;”. Each SubItem has the format of name=value.

37. **RspString=_HttpGetNVValue (Buf, Name)**

java: String _HttpGetNVValue(String Buf,String Name)

SubItem is seperated by ";". Each SubItem has the format of name=value.

38. **RspString=_TimeToHttpTime (local time)**

java: String _TimeToHttpTime(StarTimeClass Tm)

39. **local time=_HttpTimeToTime (String)**

java: StarTimeClass _HttpTimeToTime(String Stm)

40. **_HttpSetCookie(string Domain,Path,Cookie,bool_secure)**

java: void _HttpSetCookie(String Domain,String Path,String Cookie,boolean bool_secure)

41. **_HttpClearCookie(Domain,Path,Cookie)**

java: void _HttpClearCookie(String Domain,String Path,String Cookie)

42. **String= _HttpGetMediaType(FileName)**

java: String _HttpGetMediaType(String FileName)

43. **_HttpSetMaxPostSize(ConnectionID,Size)**

java: void _HttpSetMaxPostSize(int ConnectionID,int Size)

The unit of Size is KBytes

44. **int PartLength,int PartOffset,string PartHeader = _HttpGetMultiPart (binbuf_RequestBody, Index,int BoundaryNumber, binbuf_Boundary)**

java: Object[] _HttpGetMultiPart(StarBinBufClass RequestBody,int Index,int BoundaryNumber,StarBinBufClass Boundary)

Parse parameters in the request.

45. **bool= _IsTCPConnect(ConnectionID)**

java: boolean _IsTCPConnect(int ConnectionID)

If 0 is returned for send or receive function, the function may be called to determine the connection is active or not. If the connection has been closed, it returns 0.

46. **bool= _IsHttpConnect(ConnectionID)**

java: boolean _IsHttpConnect(int ConnectionID)

If 0 is returned for send or receive function, the function may be called to determine the connection is active or not. If the connection has been closed, it returns 0.

47. **bool= _FileDownLoad (Url,LocalFileName,bool WaitFlag, Comm. Callback)**

java: boolean _FileDownLoad(String Url,String LocalFileName,boolean WaitFlag,String Comm_Callback)

```
java: public void Comm_Callback(int uMes,String FileName,int MaxSize,int CurSize)
      {
      }
```

http/ftp download, for example:

comm:_FileDownLoad(http://127.0.0.1/index.html,"d:/index.html",true,nil)

comm:_FileDownLoad(http://127.0.0.1/index.html,"d:/index.html",true,comm.callback)

function comm:CallBack(uMes,FileName,MaxSize,CurSize)

end

uMes takes value from:

0 //---star download or upload

1 //---download process

2 //---finish
3 //---error
5 //---upload process

48. bool= _BufDownLoad(Url,BinBuf,WaitFlag,Comm.Callback)

java: boolean _BufDownLoad(String Url,StarBinBufClass BinBuf,boolean WaitFlag,String Comm_Callback)

http/ftp download, download data into buf

49. bool=_FileUpLoad (Url,LocalFileName, RemoteFileName,RetBinBuf, MultiPartFlag,ContentType,WaitFlag, Comm. Callback)

java: boolean _FileUpLoad(String Url,String LocalFileName,String RemoteFileName,StarBinBufClass RetBinBuf,boolean MultiPartFlag,String ContentType,boolean WaitFlag,String Comm_Callback)

http/ftp upload,RetBinBuf may be set to nil,for example:
comm:_FileUpLoad(http://127.0.0.1/index.html,"d:/index.html", "index.html" ,None,false,""",true,nil)
comm:_FileUpLoad(http://127.0.0.1/index.html,"d:/index.html", "index.html" ,,None,false,""",true,comm.callback)
default Content-Type: application/octet-stream

50. bool=_BufUpLoad (Url,BinBuf, RemoteFileName,RetBinBuf, MultiPartFlag,ContentType,WaitFlag, Comm. Callback)

java: boolean _BufUpLoad(String Url,StarBinBufClass BinBuf,String RemoteFileName,StarBinBufClass RetBinBuf,boolean MultiPartFlag,String ContentType,boolean WaitFlag,String Comm_Callback)

http/ftp upload,RetBinBuf may be set to nil.

5.15 Object garbage collect

Service group and service object will not being garbage collected.

If the object is create by script language, then the object is freed automaticlly when the corresponding script memory is garbage collected. If the object is created by cle, then the object is not freed when the corresponding script memory is garbage collected.

1. Object. _LockGC()/_UnLockGC()

java: void _LockGC()
java: void _UnLockGC()

After lock, the object is not freed when the corresponding script memory is garbage collected.

2. Object. _SLockGC()

java: void _SLockGC()

Strong lock, after the function is called, the corresponding script memory will not be garbage collected.

Defining object's event handler, setting attributes corresponding to the script language, will prevent garbage collection of objects. CLE will call _SLockGC() automatically.