1998 International Computer Symposium
Workshop on Computer Graphics and Virtual Reality
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

Modelling of a Virtual Environment with a Reusable Method

Jiung-Yao Huang+, Wei-Chuan Lin*, Jui-Fa Chen+, Chi-Ming Chung+, Wen-Hsin Chiu+, Gao-Chereng Sheu+, John C.C. Tien**

+Department of Information Engineering, TamKang University
+E-mail:jhuang@mail.tku.edu.tw
*Department of International Trade, Tak-Ming Junior College of Commerce,
*E-mail: wayne@mailsv.tmjcc.edu.tw
**Director, System Analysis Division, Bureau of Planning, Navy H.Q. R. O. C.

Abstract

One of the important issues of the virtual reality researches is the design and management of the scenario database. Traditionally, the historical data of a virtual world are imported manually and laboriously into the scenario database. Based on the standard proposed by American DOD(Department Of Defense), the virtual world can be expressed by the Federation Object Models(FOM) and Simulation Object Models(SOM) from the High Level Architecture(HLA). For the future HLA-compliant simulation, the scenario database format should base on the standard of Object Model Template(OMT). This paper proposed an easy-use, semi-automatic, and reusable framework for managing the historical scenario data. The major components of the framework are a ***Scenario Designer*** *and a* ***Data Importer****. In this way, the data in the scenario database can be easily reused to restore an original virtual world or generate a new virtual world. An example is also presented to verify this framework.*

Keywords: HLA, SOM, FOM, Scenario Designer

1. Introduction

VRML(Virtual Reality Modeling Language) is an ASCII language that is used to describe the three dimensional virtual environment. The major purpose of using an ASCII file to specify a three dimensional virtual world is to assure the efficiency of transmission and interaction in a low bandwidth network environment across different platforms. As the VRML is a standardized scene description language, it is used to describe the component objects of a virtual world. In additions, the VRML also take the event triggering and time sequencing into consideration to provide the man-machine interaction feature. In this way, the VRML can describe most of the virtual world scenario and is used in many applications. However, it still has limitations in certain special applications. The major limitation of the VRML is that it just discusses the shape of objects in a scenario and the behavior of objects depend on other third party mechanisms such as JAVA from SUN or ActiveX from Microsoft[7].

On the other hand, the *HLA(High Level Architecture)* which is proposed by American DoD is defined by three components such as the HLA rules, Interface specification, and the *OMT(Object Model Template)*[4,5,6]. The *OMT* discussed in *HLA(High Level Architecture)* is used to describe the specific behavior of objects in a virtual scene. The characteristics of OMT is to provide the recognition, realization, repeatability and reusability of virtual scene[3,15,16]. There are two components in OMT which are **FOM**(Federation Object Models), SOM(Simulation Object Models). The SOM contains the information of potential capabilities that a federate can provide to a federation. The FOM records the capabilities of all the participants in a specific federation execution. These two components are expressed in terms of tables that define shapes, interactions, attributes, parameters of objects.

The OMT also provides the guideline to reuse objects of a virtual world, however, OMT does not provide the visual view to achieve this task. To overcome this problem, this paper proposes a reuse framework that is based on OMT to describe the reuse method of the historical data. The data formats of the reusable framework are compliant with the OMT and stored in a scenario database for future reuse. The major components of the framework are ***a Scenario Designer and a Data Importer*** which use ***UML(Unified Model Language)*** as the basis to reuse objects inside the scenario database and to convert the historical data into the scenario database. The other component such as the ***Data Maintainer*** is provided the interface of OMT tables for the user to add the new data into the scenario database step by step. It also can modify or delete the historical data of the scenario database.

In the rest of this paper, section 2 surveys the existing approaches for solving the data management of a virtual environment. Section 3 elaborates the architecture of the proposed framework and the result of implementation. Section 4 gives an example to verify the proposed framework. Finally, section 5 is the conclusion and the future work.

2. The Problem-Solving Approaches

There are two approaches for developing the OMT/HLA compliant systems. The first is using the CASE tool to provide a visual view to develop the OMT for a virtual scenario[16]. This tool is developed by the Naval Undersea Warfare Center, Division Newport for the Submarine Multi-Mission Team Trainer(SMMTT) program. The tool called the Run-Time Infrastructure Modeling Extensions(RTIME) provides the development of Entity Relationship Diagrams(ERDs), state modeling, and Program Design Language(PDL) pseudocode. The RTIME Developing Approach is shown as Fig. 1 in which the ERDs of RTIME is drawn by the user manually through the Visio™ (graphical editor) on a Windows 95 platform. The SOM, that is produced by the RTIME tool, is generated by MS Excel™. The RTIME tool set supports the HLA federation development by automatically producing SOM spreadsheets from the RTIME OO models.

The advantage for using the CASE Tool to develop HLA compliant system is easy and understandable. However, it also shows that this CASE tool is unable to completely express the data of HLA OMT. That is, the Object Class Structure Table of OMT is fully support, and the other tables of OMT such as the Attribute/Parameter Table and Object Interaction Table are partially support in this CASE tool.

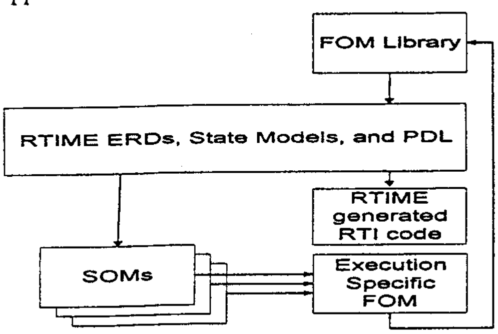

Fig. 1 RTIME tool Developing Approach

The second approach is using the UML(Unified Modeling Language) compliant methodoloy to propose a reusable method[13]. The proposed UML compliant method is called ICON Computing Catalysis and is used to define the objects in an acoustics simulation. The Catalysis process is compliant with both Object Modeling Technique (OMT) and UML and is emphasized on the reusable frameworks or design patterns. Catalysis offers "design by contract", multiple inheritance, reusable frameworks of interacting objects, building on previous Object Modeling Technique (OMT) experience and is implicitly supported by UML tools developed by Rational Corp.

Although the Method of Catalysis can express the detailed behaviors and interactions of objects, it does not generate tables that is fully compliant with the OMT tables of HLA. Besides, it cannot produce the object attributes and Interaction parameters for the OMT tables. Therefore, It cannot generate the fed file which is used by a simulation program using the RTI service of HLA. Another problem is that the Catalysis is an UML compliant method, it has made the extensions such as multiple inheritance, ..etc. Therefore, it is not easy for users to use the Catalysis method. To overcome the above problems, this paper proposes a framework to keep the essence of the CASE tool of the first approach and use the UML diagrams to semi-automatically produce the OMT tables of HLA. If the data of UML diagram is not enough to generate OMT tables, the framwork triggered a dialog box to ask the user to input the required data for OMT tables. The UML diagrams is used to describe the objects and interactions of a system significantly. These diagrams are followed the format of OMT tables.

3. The Architecture of the Framework and the Result of Implementation

3.1 The Architecture of the Framework

This paper proposes a reusable framework to extend the OMT architecture with the help of tools to develop the OMT objects. Through these tools[11], objects can be created and reused. The architecture of this framework is divided into three phases such as off line, initial time, and run time workspaces as depicted in Fig. 2. In the off line workspace, it provides mechanisms to construct SOMs from ferderates into a scenario database. In the initial time workspace, it can retrieve data of the federate from the scenario database to organize a new FOM for a new federation. That is, the FED and DIF files for the simulation program are produced in the initial time workspace. In the run time workspace, it utilizes the API from RTI of HLA to organize the FOM and SOM into a execution status of a simulation program.

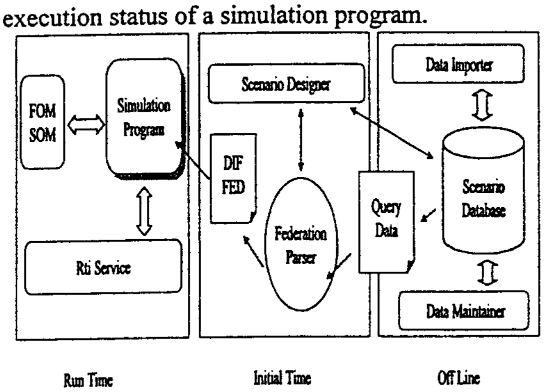

Fig. 2 The architecture of the framework

Three phases of this framework are elaborated as followed:

1.Off Line workspace:

There are three components in this workspace such as the Data importer, the Scenario database, and the Data maintainer. The Data importer is a CASE tool which provides a graphical user interface which is based on the UML diagram for user. This tool is to help the user that are not familiar with the OMT to convert the data of UML diagrams into the tables of OMT. Through the diagrams provided by UML, the user can easily model a SOM of OMT. Although the user can use the UML diagrams to describe the relations and interactions of object classes, there are still lack of information to combine the data of UML diagrams into the tables of OMT. To solve this problem, the Data importer also provide a dialog box for user to describe the missing data that are required by the tables of OMT. In this way, it can semi-automatically convert the data in UML into the tables of OMT. Fig. 3 shows the internal flow and module relations of the Data importer. A user can describe a virtual world by the visual diagrams provided by the Data importer. As shown in Fig.3, the kernel of the Data importer is divided into the UML Standard and UML Extension modules. The UML Standard module is the basic diagrams of the UML such as the use case, class, sequence diagrams, ...etc. The UML Extension module is the module to handle the missing data of the UML diagrams for tables of OMT. After these steps, the Meta Data are produced by combining the data receiving from the UML standard and UML extension modules. Finally, the OMT Generator Module use the Meta Data and the Scenario database to produce the **OMT** Instance Information(**OII**). The **OII** can be used to construct the federate object into the Scenario databases.

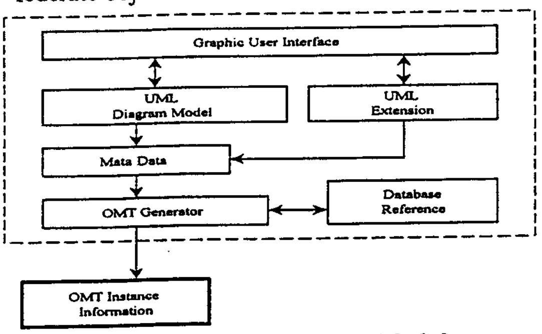

Fig. 3 Data Importer Module

The scenario database[9,10] is the database to store the SOM data and some other extension for system module. As a part of SOM data container, the scenario database should follow the specification of OMT. Owing to the object-oriented concepts of OMT, the data of OMT should be stored in an object-oriented database. This paper suggests that the scenario database can be a relational database which uses the OO-API and rules to describe the Object Model Identification Table, Object class structure table, Interaction class structure table, attribute table, parameter table, Attribute table/parameter table sub-components, Routing Space table. For future reuse, the data in the scenario database can be reorganized into the tables of OMT by the predefined SQL language. The extension of the scenario database is to provide the required information of a federation such as the rules of a federation, the emergency handling of a federation, the analysis of a federation ... etc. Through these data, the modules of a federation can be plug-and-play dynamically. Fig. 4 shows part of the ER Model of the scenario database[8]. In Fig. 4, most of the table and attribute design follows the data format of OMT and some of them are designed as needed in this framework.

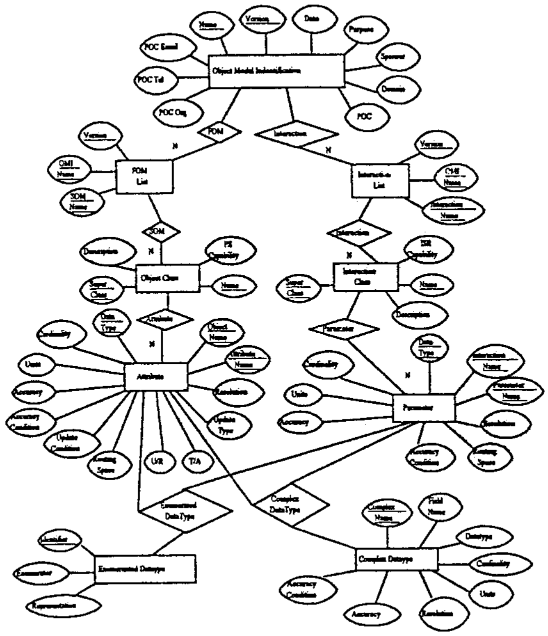

Fig. 4 The ER Model of the Scenario Database

The Data maintainer[2] is a CASE tool for the user that are familiar with the OMT of HLA. This tool provides a dialog box for users to maintain the scenario database. A user who are not familiar with the OMT can use the data maintainer as a guide to construct a SOM step by step. The other user that are master in the OMT can modify or delete an existed SOM directly. The data maintainer also provides some utilities for user to get more related information from the scenario database. Fig. 5 shows the internal flow and module relations of the Data Maintainer. In Fig. 5, a user can use the data maintainer to describe a virtual world follows the concepts of OMT. Through the data maintainer, the OMT Sheet Model is produced. In the meantime, the OMT Generator Module uses the information of the OMT Sheet Model and the Scenario Database to produce the same as **OII** in Data Importer Module. The **OII** can be used to construct the federate object into the Scenario database.

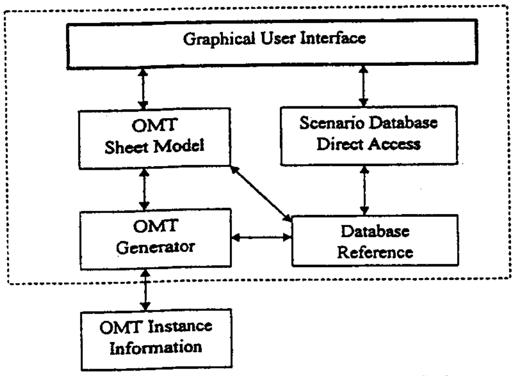

Fig. 5 Data Maintainer Module

2.Initial Time workspace

There are two components in this workspace which are the Scenario Designer, and the Federation Parser. The Data importer is a CASE tool which provides the graphical user interface that are used to model a FOM of OMT. This tool uses the UML diagrams to describe the FOM of OMT. Fig. 6 shows the internal flow and module relations of the Scenario Designer. In Fig. 6, when a user use the UML diagram to describe a federation, the scenario designer will reuse the data of the scenario database. Since the detail information of a federate already exists in the scenario database, the user does not need to specify the detail specification of a federate, in stead, the user just input the object class and corresponding interactions. The scenario designer collects all the user inputs to the Scenario Generator Module. The Scenario Generator Module will according to the user requirement and the data in the scenario database generate the scenario information for the Federation(Fed) Parser.

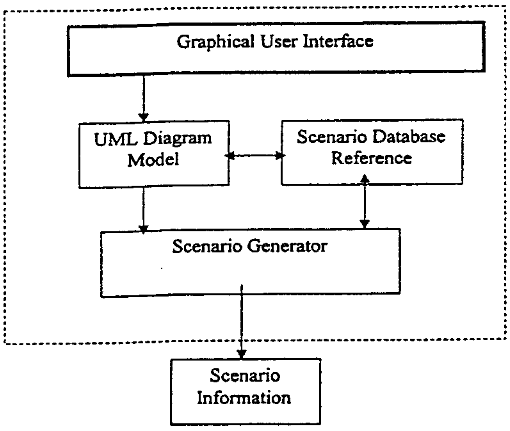

Fig. 6 Scenario Designer Module

Based on received scenario information, the Fed Parser produces the SOM and interactions needed by a federation. These SOMs and interactions are generated in FED and DIF file format for simulation program. Fig. 7 shows the internal flow and module relations of the Federation Parser. The Federation Generator Module will reference the scenario information and the data in the scenario database to produce the FED or DIF file[1].

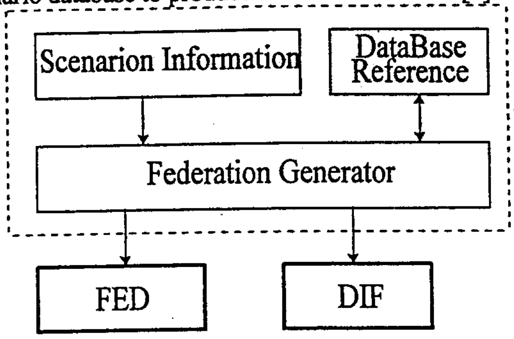

Fig. 7 Federation Parser Module

3.Run Time workspace

There are two components in this workspace, they are the Simulation program and the Rti service that are all proposed by the DoD. The Simulation program follows the spec of HLA by using the c++ version of API to interact with the Rti Service and builds a federation for simulation[15].

3.2 The Implementation

The proposed system is implemented in Windows platform. The purpose of the data importer is to provide a CASE tool for user to enter the SOM data by the diagrams of UML. Through the conversion of the Data Importer, the SOM data is produced and is stored in the scenario database for future reuse. Fig. 8 shows the Use Case diagrams of the data importer. The Use Case diagram is one of the diagrams for the user. The other diagrams such as class diagram, sequence diagrams ... etc. are also used by the user to describe the scenario data. To compensate the missing data of SOM, the data importer provides the dialog windows to guide the user to input the required data. As shown in Fig. 9, the extended data of the UML diagram for SOM are input through the dialog windows. The data importer will shows the required information in the dialog box to guide the user to input the missing data.

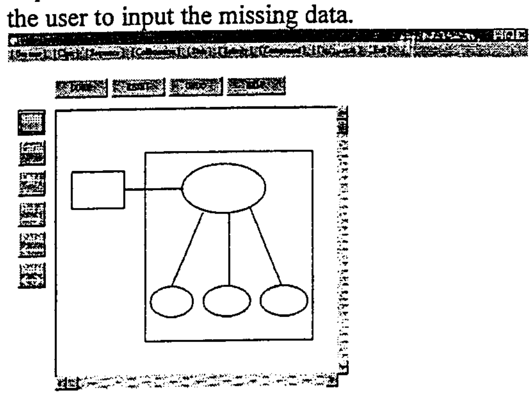

Fig. 8 The Use Case diagram of the data importer

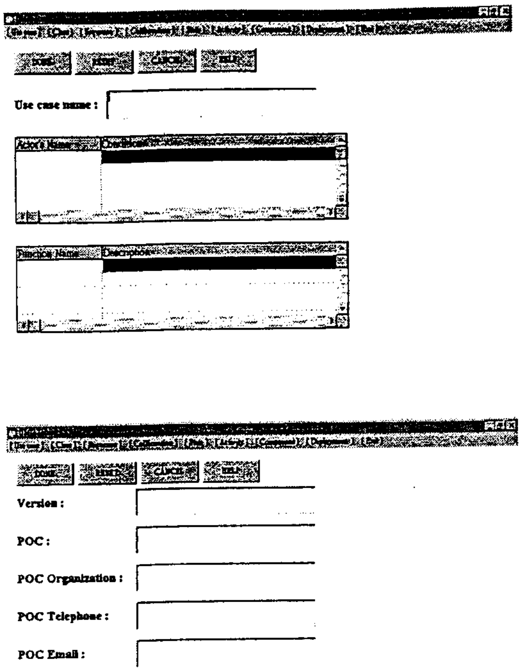

Fig. 9 The dialog boxes for the data importer

4. An Example to Verify the Proposed Framework

In this section, an example is shown to explain how to use UML diagram to map the data format of HLA OMT. There are eight diagrams in UML to describe a system. These diagrams are Use Case diagram, Class diagram, Sequence diagram, Collaboration diagram, State diagram, Activity diagram, Component diagram, Deployment diagram. These diagrams are obtained from the data importer except the Component diagram and Deployment diagram which are input from the dialog windows. This example is taking from the federation of Navy. From this example, the Use Case for the Use Case diagram are Platform and Munition. The interactions are also happened between the two objects and the Use Case diagram is shown as Fig. 10.

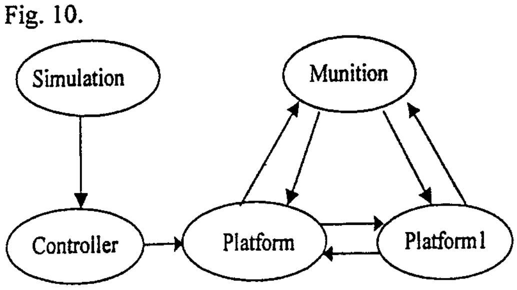

Fig. 10 UML Use Case Diagram

The corresponding Use Case description is listed as followed:

Use case name: Naval Simulation
Actor's name:
Condition: The platform of ships are simulation in a virtual world, The ship can use the weapon to attack the enemy and can detect whether it was hit by some objects.
Function: Weapon_Fire, Collision, Detonation
Use case description: Weapon_Fire – Platform fires the weapon to the target and destroys it.
Use case description: Collision – Platform moves and detects whether a collision occurred.
Use case description: Detonation -- The weapon destroys the target and detonates.

The Class diagrams of the example are divided into two parts. One is to describe the static structure of objects(federates) as shown in Fig. 11. The other is to describe the internal dependency of federates in the entire federation as shown in Fig. 12. From these two Figures the inheritance relationship of HLA OMT Object Class are obtained.

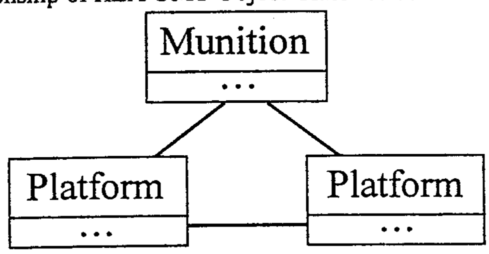

Fig. 11 Class Diagram(1)

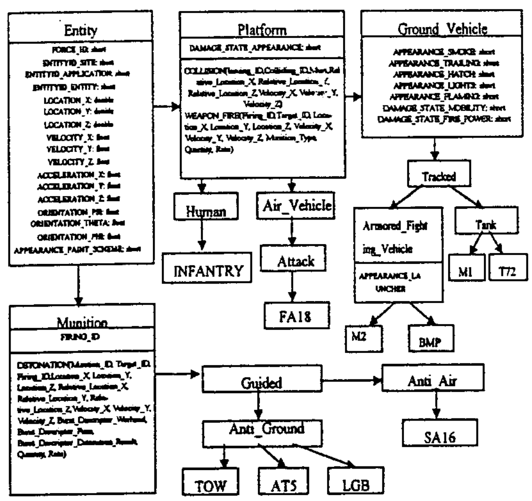

Fig. 12 Class Diagram(2)

The Sequence diagram is to describe the event handling according to the time. As shown in Fig. 13, the events are platform move, platform fire weapon and weapon detonate.

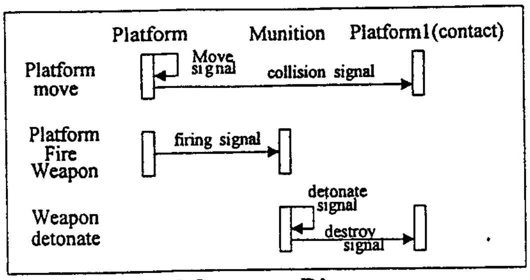

Fig. 13 Sequence Diagram

The Collaboration diagram is showing the object relations and the message exchanging status as shown in Fig. 14.

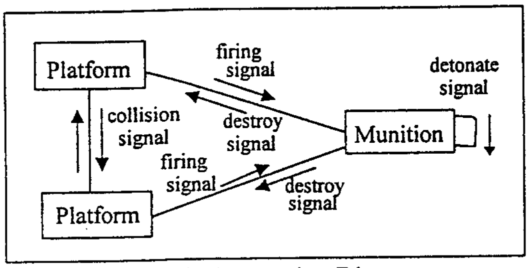

Fig. 14 Collaboration Diagram

The State diagram shows the internal state transition by the events. This is the micro-view of a system to describe the status of internal changes. The internal state change of the Munition object is shown in Fig. 15.

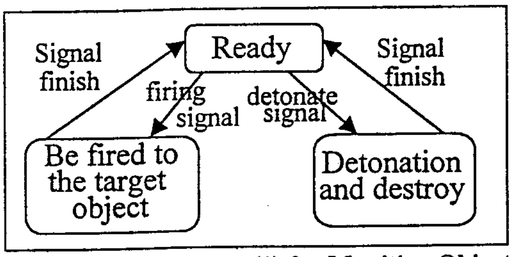

Fig. 15 State Diagram(1) for Munition Object

The Activity diagram shows the detailed event handling procedures as shown in Fig. 16.

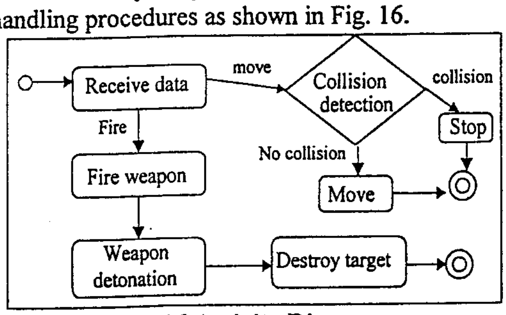

Fig. 16 Activity Diagram

The Component Diagram and Deployment Diagram are to describe the physical component and the configuration of running processing elements. As these two diagrams are focus on the system physical software components and systems architecture, these two diagrams are not necessary for the data importer.

For the conversion from the UML diagrams to the tables of OMT, the needed data that are collected most from the diagrams. The other data that are collect from the user by using some dialog windows as shown in Fig. 9 are necessary. These data such as the fields of the Attribute table and Parameter table. In the Class diagram of UML, the attribute and parameter just focus on the data-type. However, there are still some fields that are needed in the Attribute table and Parameter table of HLA OMT as shown followed:

- Cardinality
- Units
- Resolution
- Accuracy
- Accuracy condition
- Update type (Parameter Table doesn't have this field)
- Update rate/Condition (Parameter Table doesn't have this field)
- Transferable/Acceptable (Parameter Table doesn't have this field)
- Updateable/Reflectable (Parameter Table doesn't have this field)
- Routing Space

The above data will be obtained from the user through the dialog windows. The conversion result is the tables as shown from Table 1 to 4.

Object Class Structure Table					
E(ps)	P(ps)	GV(ps)	Tk(ps)	T(ps)	M1(ps)
					T72(ps)
				AFV(ps)	M2(ps)
					BMP(ps)
		AV(ps)	A(ps)		
		H(ps)	I(ps)		
	M(ps)	G(ps)	AG(ps)	TOW(ps)	
				AT5(ps)	
				LGB(ps)	
			AA(ps)	SA16(ps)	

Table 1 Object Class Structure Table

Interaction Class Structure Table	
COLLISION (IR)	
WEAPON_FIRE (IR)	
DETONATION (IR)	

Table 2 Interaction Class Structure Table

Attribute Table					
Object	**Attribute**	**Datatype**	**Cardinality**	**Units**	**...**
E	F_ID	short	1	E	...
	...	...	...	...	...
	V_X	double	1	m/s	...
	A_X	double	1	m/s2	...
P	D_S_A	short	1	E	...
M	FIRE_ID	long	1	N/A	...
GV	A_S	short	1	E	...
	...	...	...	...	...
	D_S_F_P	short	1	E	...
AFV	A_L	short	1	E	...

Table 3 Attribute Table

Parameter Table					
Interaction	**Parameter**	**Datatype**	**Cardi..**	**Units**	**...**
COLLISIO N	Coll_ID	long	1	N/A	...
	...	...	...	...	...
	R_L_X	double	1	M	...
WEAPON_ FIRE	Firing_ID	long	1	N/A	...
	...	...	...	...	
	M_Type	string	1	N/A	...
DETONATI ON	M_ID	long	1	N/A	...
	...	...	...	...	...
	B_D_W	short	1	E	...

Table 4 Parameter Table

The other table such as the Object Model Identification Table is to record the information of the Object Model. The data of this table can be obtained from the user as shown in table 5.

Object Model Identification Table	
Category	**Information**
Name	CCTT SAF
Version	Build 7
Date	08/01/1997
Purpose	Support team training
Application Domain	Training and Analysis
Sponsor	STRICOM / DMSO
POC	Ms.Christina Bouwens
POC Organization	SAIC
POC Telephone	Christina Bouwens
POC Email	Christina.Bouwens@cpmx.saic.com

Table 5 Object Model Identification Table

5. Conclusion and Future Work

This paper proposes a reusable framework architecture to handle the scenario data of a federation. The framework is divided into three workspaces to process the scenario data. The off line workspace is to collect the historical data and add the new data in the scenario database for future reuse. The initial time workspace is to describe the needed reusable scenario for a new federation. The outputs of this workspace are the files needed by a federation. The run time workspace is followed the HLA RTI to simulate a real world environment. Through this framework, the historical scenario data can be reuse easily. The advantages of this framework are the developing process from design to the result is visualized and reusable, the work for add in or reuse the scenario data is easy and clarified, and the modulized design of each mechanisms in this architecture can be plug in and play easily.

For the future work, there are three parts of works can be investigated. The first is that the basis methodology of kernel – UML can be replaced by other methodology that is more feasible to convert into the HLA OMT. The second is that the data in the scenario database can be transformed by the VRML parser into the VRML file for simulating in the internet browser. The third is that the HLA RTI service can be rewritten as a plug in and play or replace module for the flexibility of the system. In this way, it can promote the performance and flexibility of the simulation.

Acknowledge

The authors want to thank for Doctor K. H. Chung who are the chairman of the national defense management college for his direct of this paper.

References

[1] "High Level Architecture Object Model Template Data Interchange Format(DIF)-DRAFT, version 1.0-8", 30 Jan., 1997

[2]Christina Bouwens ,Raymond L. Miller, Roy Scrudder, Robert Lutz, "Object Model Development: Tools and Techniques", 1998 Spring Simulation Interoperability Workshop, 9-13 March, 1998 URL http://siso.sc.ist.ucf.edu/siw/98spring/docs/papers/98s-siw-142.pdf

[3] Defense Modeling and Simulation Office, "HLA Federation Development and Execution Process(FEDEP) Model, version 1.1(draft)", 22 November 1997

[4] Defense Modeling and Simulation Office, "HLA Object Model Template, version 1.3", 5 Feb., 1998

[5] Defense Modeling and Simulation Office, "HLA Interface Specification, version 1.3", 5 Feb., 1998

[6] Defense Modeling and Simulation Office, "HLA Rules, version 1.3", 5 Feb., 1998

[7] Don Brutzman, "The Virtual Reality Modeling Language and Java", to appear 1998, CACM, http://www.stl.nps.navy.mil/~brutzman/vrml/vrmljava.ps

[8] Elmasri, Navathe, "Fundamentals of Database System", 2nd edition, Addison-Wesley, 1994

[9] Jack H. Sheehan, "HLA Data and OMT Library Plan", Briefing to 16th HLA Architecture Management Group Meeting, 17 Dec., 1996

[10] James Hammond, "Populating the HLA Object Model Data Dictionary -- Bottom-Up Approach" URL http://siso.sc.ist.ucf.edu/siw/98spring/docs/papers/98S-SIW-075.pdf

[11] Ken Hunt, "Planning For the Evolution of Automated Tools in HLA", Simulation Interoperability Workshop, 3-7 March, 1997. URL http://ftp.sc.ist.ucf.edu/SISO/SIW/97spring/papers/067.pdf

[12] Object Model Library URL http://www.omlibrary.epgc4i.com/

[13] Patricia Devine, Glenn Gross, "How to Develop HLA FOMs/SOMs using a Repeatable, Formal OOA Methodology and Process", 1997 Fall Simulation Interoperability Workshop, 8-12 Sep., 1997 URL http://www.sisostds.org/SIW/97fall/DOCS/PAPERS/97F-SIW-167.pdf

[14] Scrudder, R., "High Level Architecture (HLA) Object Model Tool Development", Simulation Interoperability Workshop, 8-12 Sep., 1997 URL http://ftp.sc.ist.ucf.edu/SISO/SIW/97spring/papers/180.pdf

[15] Scrudder, R., "HLA FOM/SOM Content Standards", Simulation Interoperability Workshop, 3-7 March, 1997 URL http://www.sisostds.org/SIW/97fall/DOCS/PAPERS/97F-SIW-164.pdf

[16] Wayne Belanger, Joshua Trainer, John Ackerman, "A Case Tool for Development of HLA Systems"