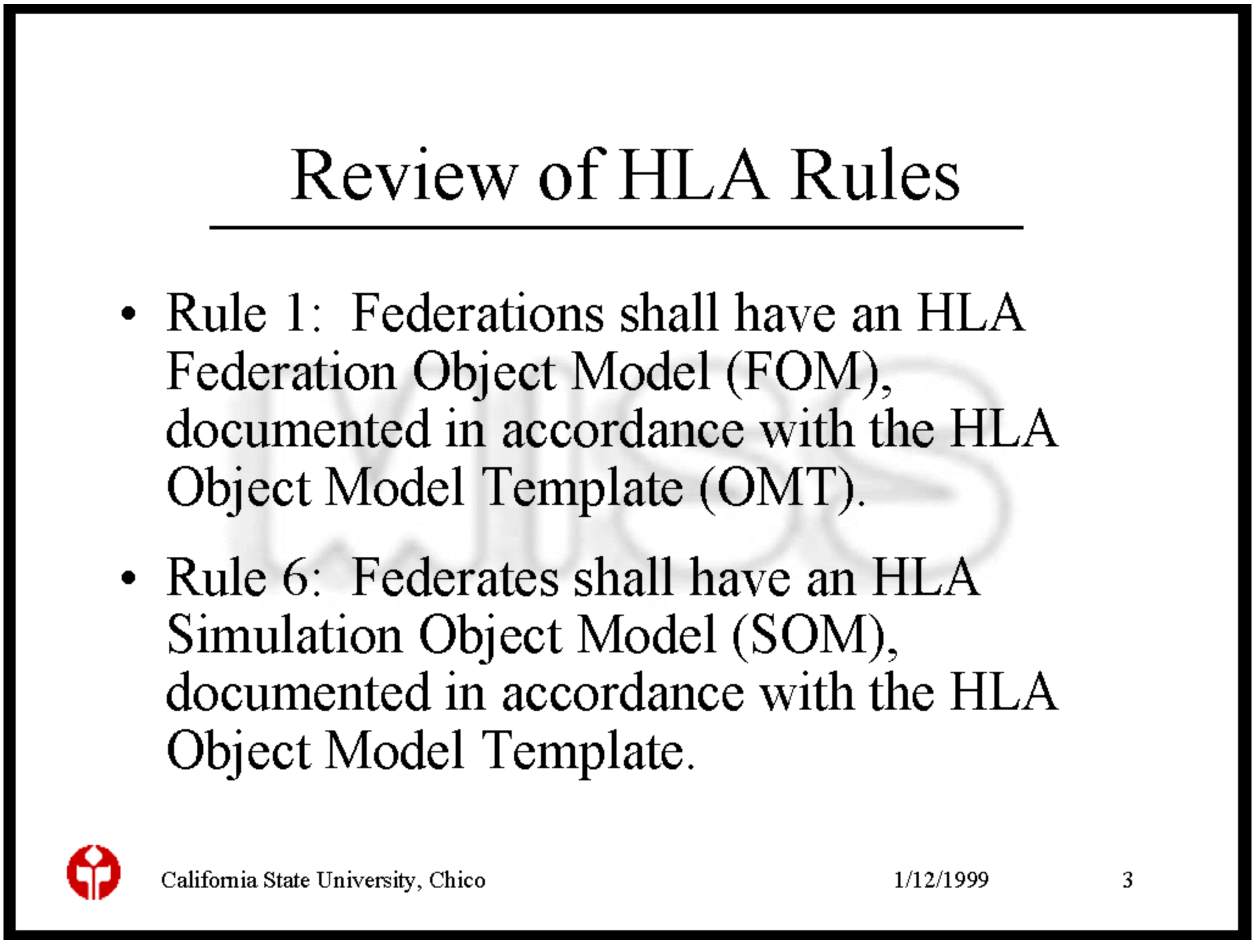
HLA MODULE 1

Part 3 - The Object Model Template

3.0 Introduction

Recall HLA Rules 1 and 6, which are repeated in Figure 3.1.



Review of HLA Rules

- Rule 1: Federations shall have an HLA Federation Object Model (FOM), documented in accordance with the HLA Object Model Template (OMT).
- Rule 6: Federates shall have an HLA Simulation Object Model (SOM), documented in accordance with the HLA Object Model Template.

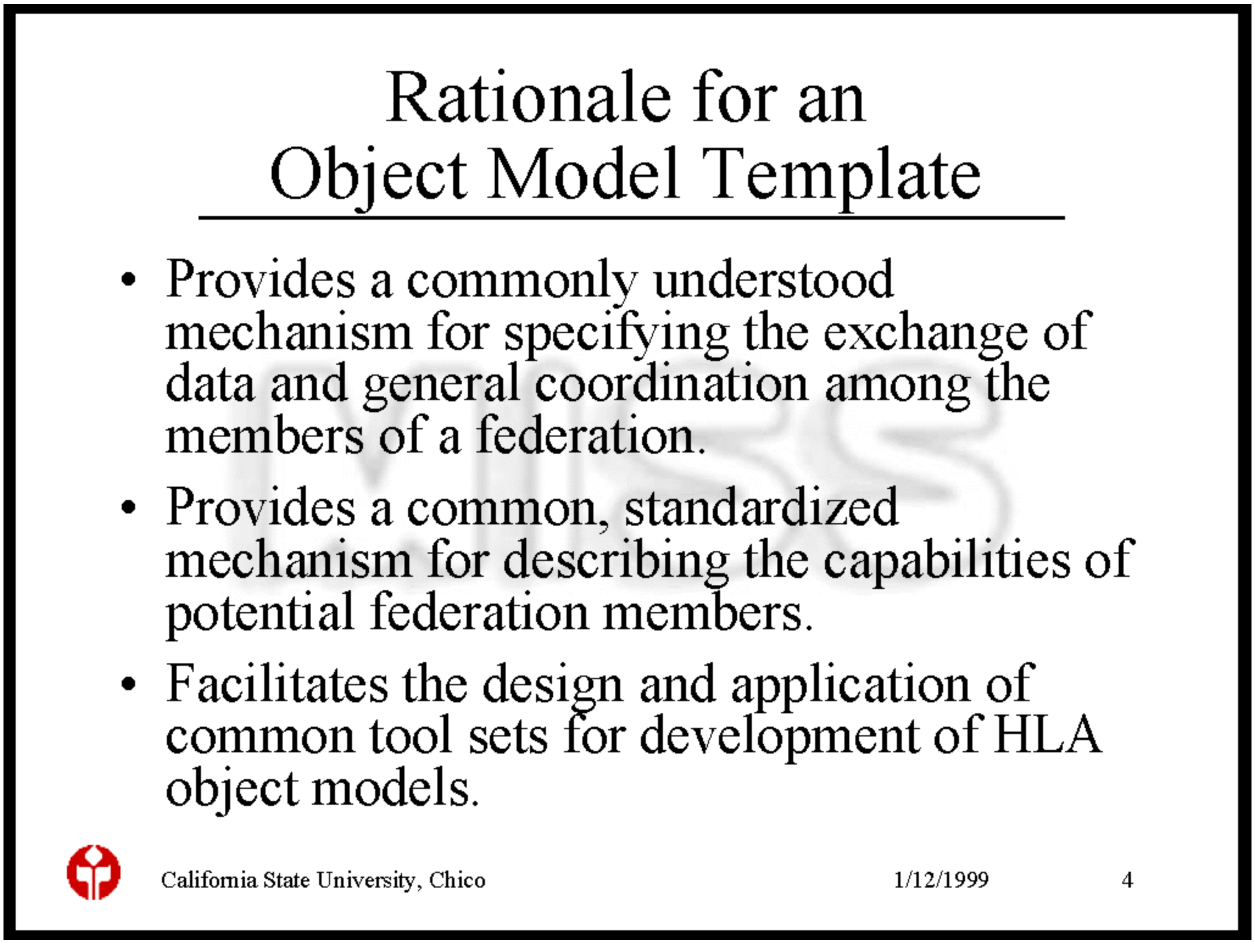
California State University, Chico 1/12/1999 3

Figure 3.1: HLA Rules that Refer to an Object Model Template

These rules require that there shall be an Object Model Template (OMT) that describes all objects and interactions managed by a federate and visible outside a federate. The specification of a standard format for the Object Model Template is part of the HLA standard. The HLA separates data and structure. It prescribes that OMT objects and interactions, defined according to the OMT, can be constructed and exchanged with no adjustments to HLA-derived software.

The OMT is defined by the High-Level Architecture Object Model Template

Specification. This document provides the rationale in Figure 3.2 for using a standardized structural framework, or template, for specifying HLA object models:



Rationale for an Object Model Template

- Provides a commonly understood mechanism for specifying the exchange of data and general coordination among the members of a federation.
- Provides a common, standardized mechanism for describing the capabilities of potential federation members.
- Facilitates the design and application of common tool sets for development of HLA object models.

California State University, Chico 1/12/1999 4

Figure 3.2 Rationale for an Object Model Template

HLA object models may be used to describe an individual federation member (federate), creating an HLA simulation object model (SOM); or to describe a named set of multiple interacting federates (federation), creating a federation object model (FOM).

The primary function of the FOM is to specify, in a common standardized format, the nature of the data exchange among federates. This data includes an enumeration of all object and interaction classes pertinent to the federation, along with a specification of the attributes or parameters characterizing these classes.

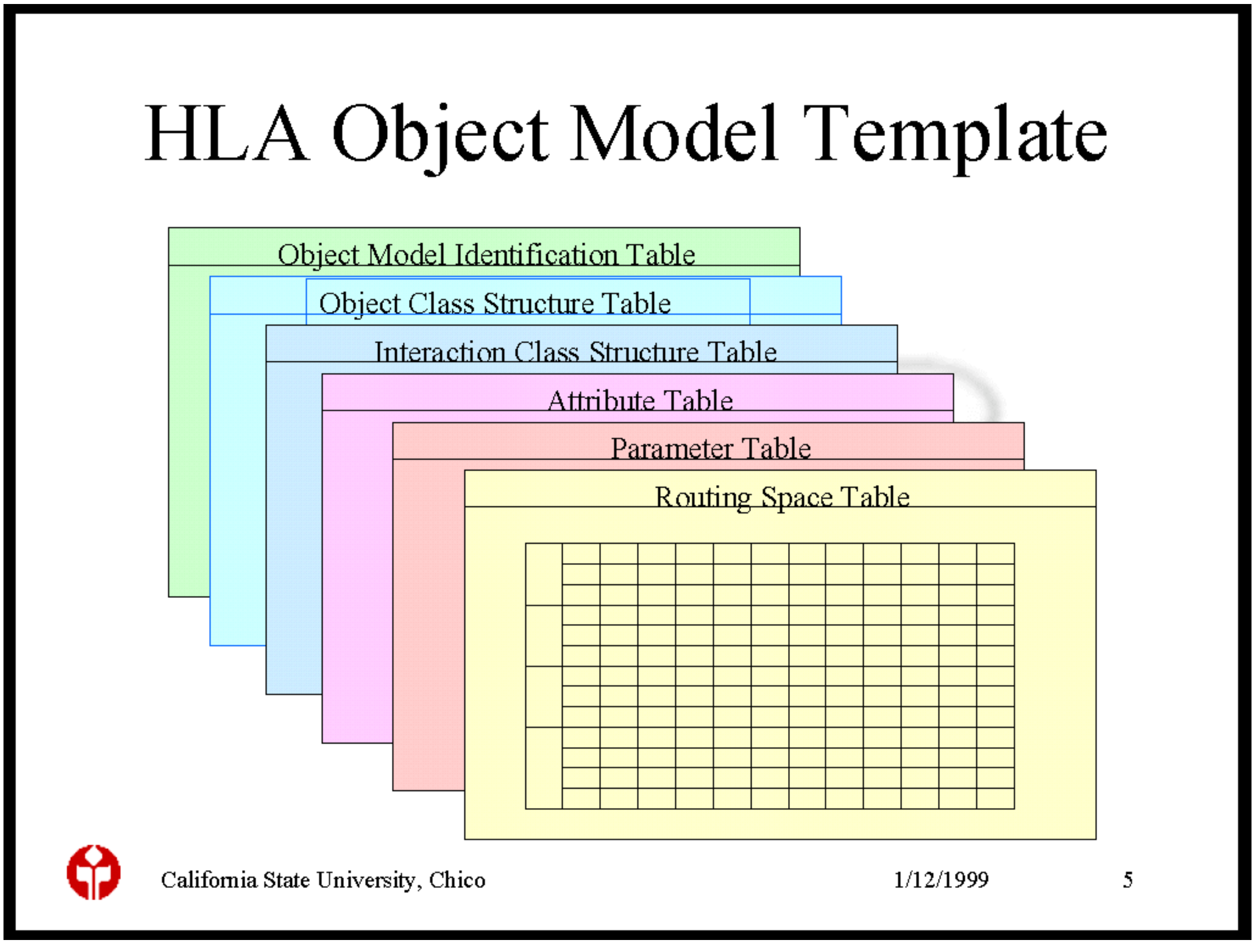
An HLA SOM is a specification of the intrinsic capabilities that an individual simulation could provide to HLA federations. The SOM provides an indication of the suitability of a simulation system for participation in a particular federation.

Note that there are differences between the way object models are defined in the HLA and in the techniques of Object-Oriented Analysis and Design (OOAD). In OOAD an object model provides a much wider description of an object including complete details of the internal relationships characterizing the object. HLA object models, on the other hand, are much narrower, focusing on federate / federation information interchange.

The following information about the composition of the OMT applies equally to FOMs and SOMs unless specified otherwise.

3.1 Components of the OMT

The OMT specifies that HLA object models must be documented in the form of a number of tables that specify information about classes of objects, their attributes and their interactions. The following seven components are required by the OMT Specification:



HLA Object Model Template

Object Model Identification Table

Object Class Structure Table

Interaction Class Structure Table

Attribute Table

Parameter Table

Routing Space Table

California State University, Chico 1/12/1999 5

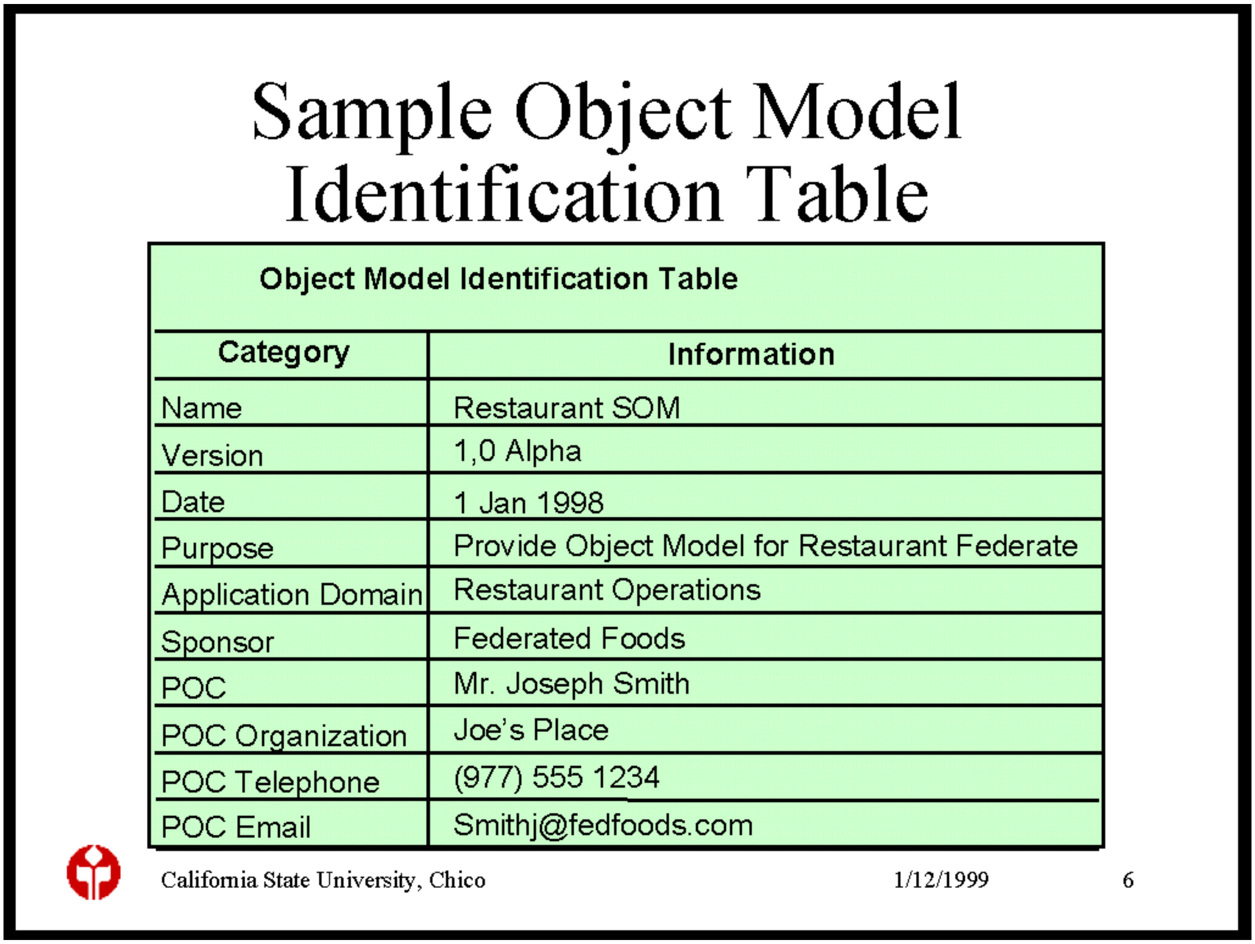
Figure 3.3 Components of the OMT

The OMT Specification requires that both federations and individual simulations (federates) shall use all seven of the OMT components although some tables may be empty in some cases. All HLA object models should contain at least one object class or interaction class.

The seven components of the OMT are now described briefly. More complete descriptions can be found in the HLA OMT Specification.

3.1.1 Object Model Identification Table

The purpose of the object model identification table is to provide key identifying information about the federation or federate. Figure 3.4 illustrates the structure of the table and provides an example of its contents, which are largely self-explanatory.



Sample Object Model Identification Table

Object Model Identification Table

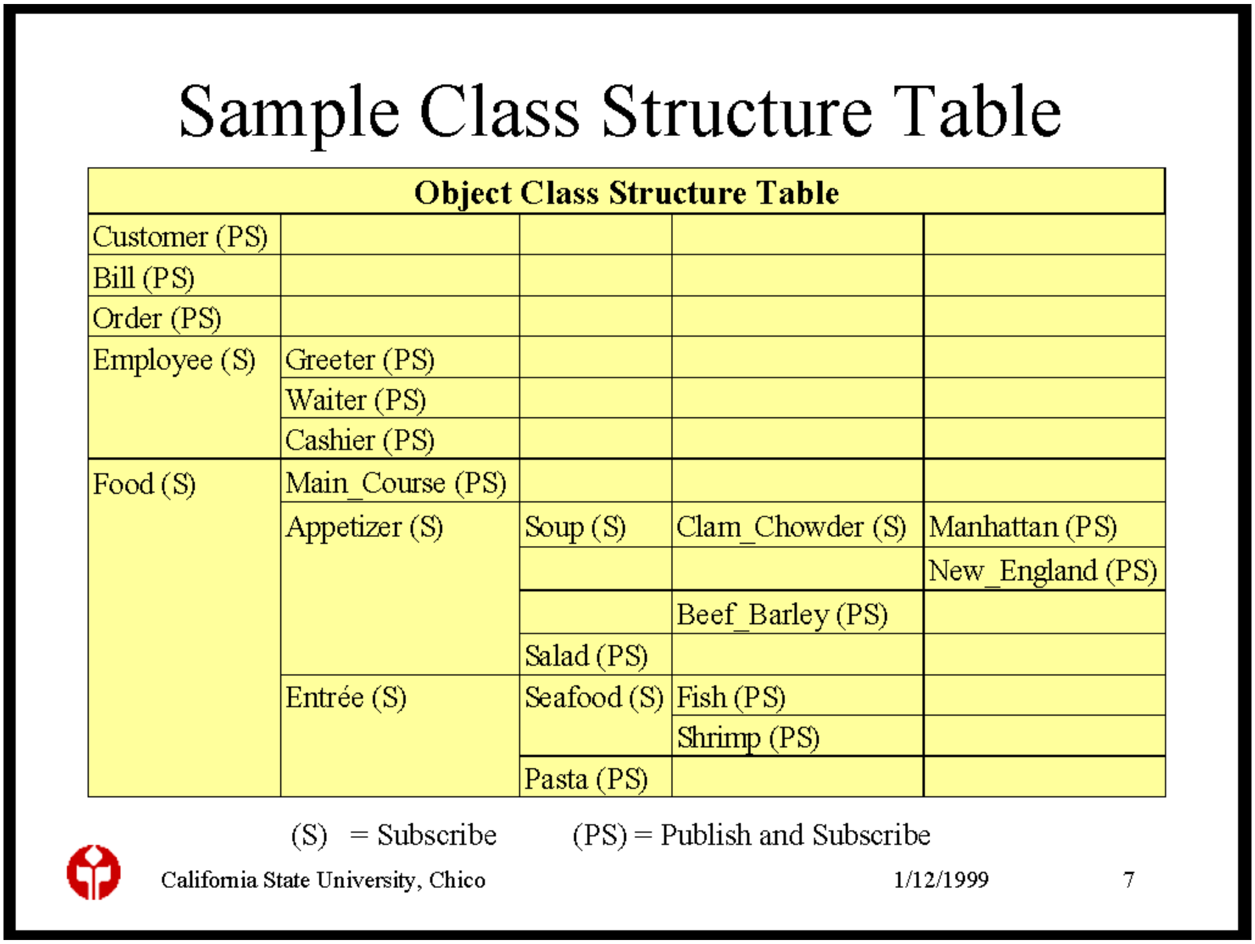
Category	Information
Name	Restaurant SOM
Version	1,0 Alpha
Date	1 Jan 1998
Purpose	Provide Object Model for Restaurant Federate
Application Domain	Restaurant Operations
Sponsor	Federated Foods
POC	Mr. Joseph Smith
POC Organization	Joe’s Place
POC Telephone	(977) 555 1234
POC Email	Smithj@fedfoods.com

California State University, Chico 1/12/1999 6

Figure 3.4 Object Model Identification Table

3.1.2 Object Class Structure Table

The Object Class Structure Table contains information about the class-subclass hierarchy of the object classes. Figure 3.5 shows a partially completed example of the contents of the Table.



Sample Class Structure Table

Object Class Structure Table				
Customer (PS)				
Bill (PS)				
Order (PS)				
Employee (S)	Greeter (PS)			
	Waiter (PS)			
	Cashier (PS)			
Food (S)	Main_Course (PS)			
	Appetizer (S)	Soup (S)	Clam_Chowder (S)	Manhattan (PS)
				New_England (PS)
			Beef_Barley (PS)	
		Salad (PS)		
	Entrée (S)	Seafood (S)	Fish (PS)	
			Shrimp (PS)	
		Pasta (PS)		

(S) = Subscribe (PS) = Publish and Subscribe

California State University, Chico 1/12/1999 7

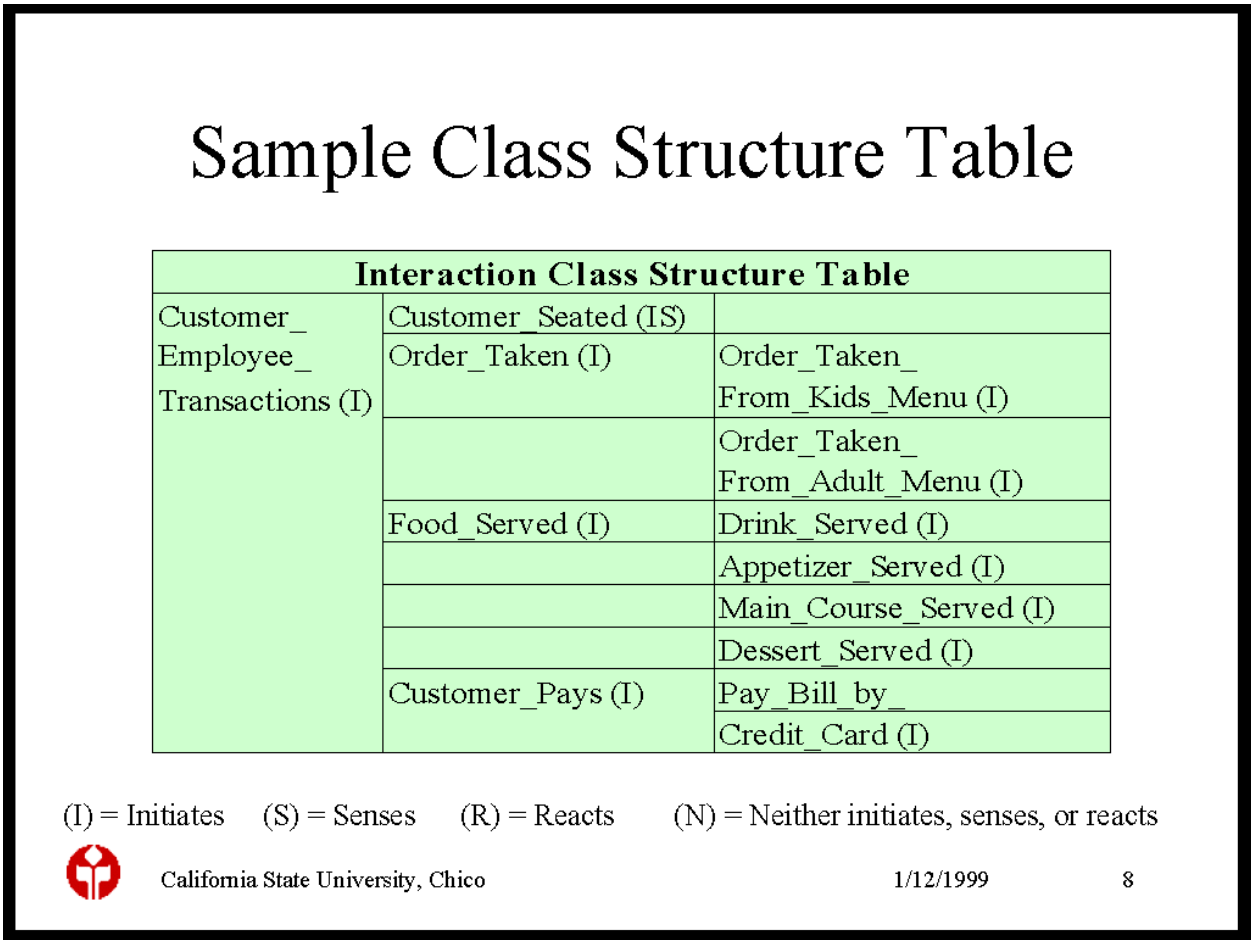
Figure 3.5 Object Class Structure Table

Each object class is classified according to its publication and subscription capabilities. The letters P,S and N indicate *publishable, subscribable* and *neither publishable nor subscribable* respectively. A class can be classified as both publishable and subscribable using the letters PS.

Publishable means that a federate using the Publish Object Class service can publish the specified object class. Subscribable means that a federate is capable of utilizing and reacting to information on objects in the given class.

3.1.3 Interaction Class Structure Table

An interaction is an action taken by an object (or set of objects) in one federate that may have an effect on objects in a different federate. These interactions are defined in the Interaction Class Structure Table. Figure 3.6 shows a partially completed example of the table.



Sample Class Structure Table

Interaction Class Structure Table		
Customer_Employee_Transactions (I)	Customer_Seated (IS)	
	Order_Taken (I)	Order_Taken_From_Kids_Menu (I)
		Order_Taken_From_Adult_Menu (I)
	Food_Served (I)	Drink_Served (I)
		Appetizer_Served (I)
		Main_Course_Served (I)
		Dessert_Served (I)
	Customer_Pays (I)	Pay_Bill_by_
		Credit_Card (I)

(I) = Initiates (S) = Senses (R) = Reacts (N) = Neither initiates, senses, or reacts

California State University, Chico 1/12/1999 8

Figure 3.6 Object Class Structure Table

As with the object class structure table, the interaction class structure table identifies federate / federation capabilities with respect to its interaction classes. The four basic categories are designated by the letters I, S, R and N which indicate that a federate is capable of:

- initiating and sending the given interactions (I)
- sensing the interaction by subscribing to it and utilizing its information without necessarily being able to effect the appropriate changes to affected objects (S)
- reacting to it by subscribing to it and properly reacting to it by effecting the

appropriate changes to any owned attributes of affected objects (R)

- neither initiating, sensing or reacting to this interaction class (N).

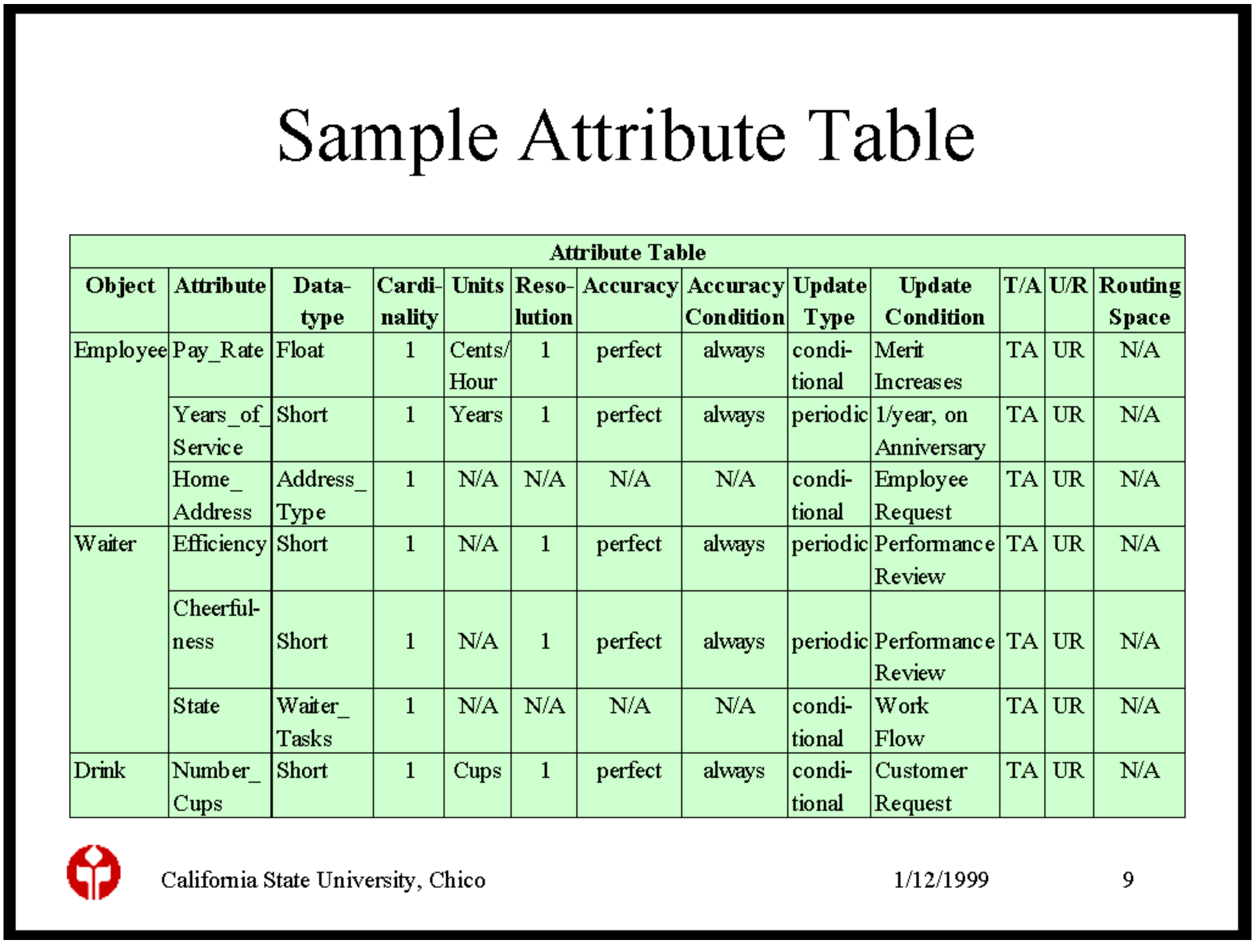
3.1.4 Attribute Table

Each class of simulation domain objects is characterized by a fixed set of attribute types. These attributes are named portions of the state of their object whose values can change over time (such as the velocity of a vehicle). Updates to the values of HLA object class attributes are published through the RTI and provided to other federates in a federation. An HLA object model documents these attributes in the attribute table.

For each attribute the table records the following characteristics (see the HLA OMT Specification pp. 16-19 for further details):

- Object class
- Attribute name
- Datatype
- Cardinality
- Units
- Resolution
- Accuracy
- Accuracy condition
- Update type
- Update rate / condition
- Transferable / acceptable
- Updateable / reflectable
- Routing Space

Figure 3.7 shows a partially completed example of the table.



Sample Attribute Table

Attribute Table												
Object	**Attribute**	**Datatype**	**Cardinality**	**Units**	**Resolution**	**Accuracy**	**Accuracy Condition**	**Update Type**	**Update Condition**	**T/A**	**U/R**	**Routing Space**
Employee	Pay_Rate	Float	1	Cents/Hour	1	perfect	always	conditional	Merit Increases	TA	UR	N/A
	Years_of_Service	Short	1	Years	1	perfect	always	periodic	1/year, on Anniversary	TA	UR	N/A
	Home_Address	Address_Type	1	N/A	N/A	N/A	N/A	conditional	Employee Request	TA	UR	N/A
Waiter	Efficiency	Short	1	N/A	1	perfect	always	periodic	Performance Review	TA	UR	N/A
	Cheerfulness	Short	1	N/A	1	perfect	always	periodic	Performance Review	TA	UR	N/A
	State	Waiter_Tasks	1	N/A	N/A	N/A	N/A	conditional	Work Flow	TA	UR	N/A
Drink	Number_Cups	Short	1	Cups	1	perfect	always	conditional	Customer Request	TA	UR	N/A

California State University, Chico 1/12/1999 9

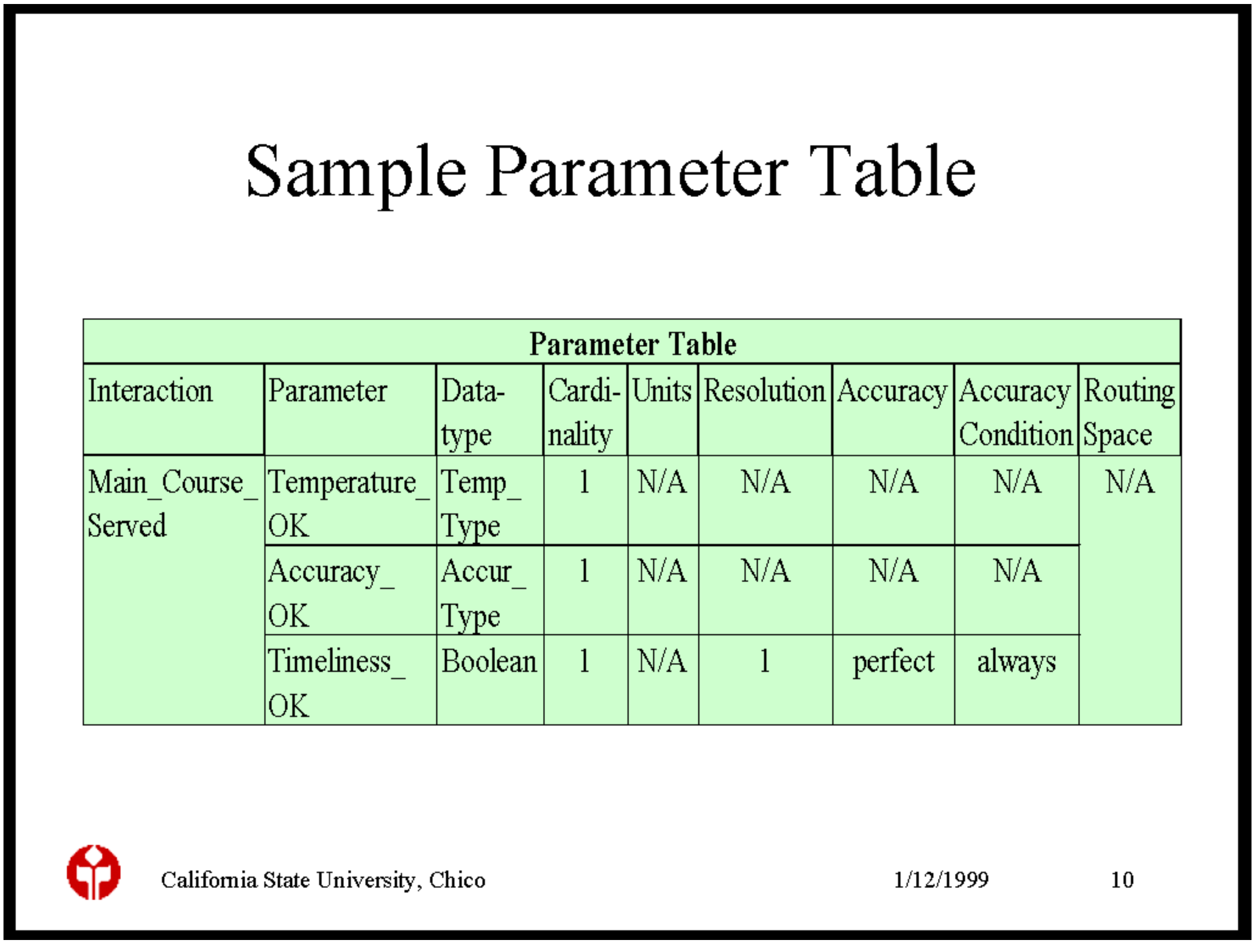
Figure 3.7 Attribute Table

Additional table formats are available to support enumerated and complex datatypes (see HLA OMT Specification pp. 24-28).

3.1.5 Parameter Table

For every interaction class identified in the interaction class structure table, the full set of parameters associated with that interaction class is specified in the parameter table. The table entries consist of the relevant interaction classes, the parameter names associated with the interaction classes and information specifying their datatype, cardinality, units, resolution, accuracy, accuracy condition and routing space.

Figure 3.8 shows a partially completed example of the table.



Sample Parameter Table

Parameter Table								
Interaction	Parameter	Data-type	Cardi-nality	Units	Resolution	Accuracy	Accuracy Condition	Routing Space
Main_Course_ Served	Temperature_ OK	Temp_ Type	1	N/A	N/A	N/A	N/A	N/A
	Accuracy_ OK	Accur_ Type	1	N/A	N/A	N/A	N/A	
	Timeliness_ OK	Boolean	1	N/A	1	perfect	always	

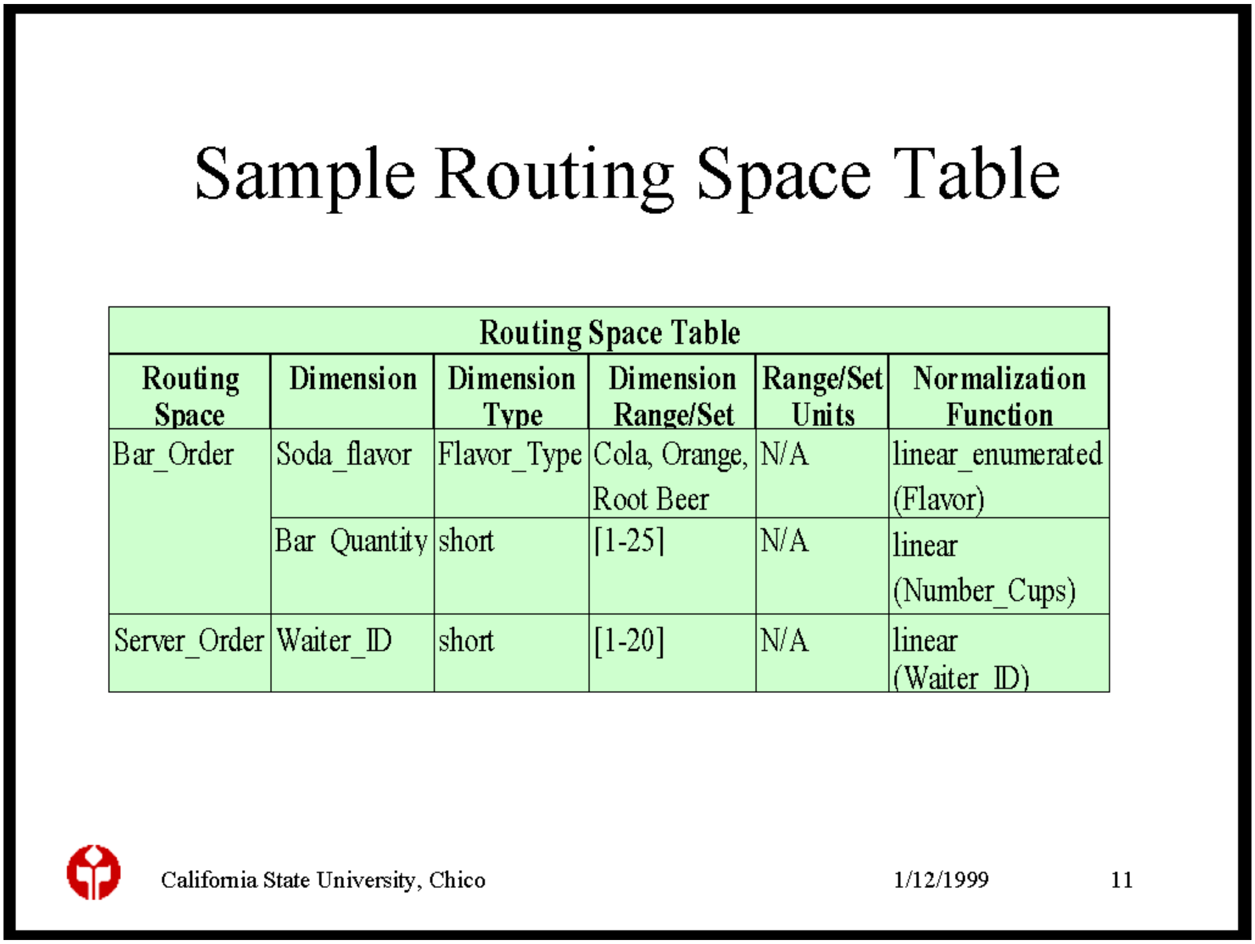
California State University, Chico 1/12/1999 10

Figure 3.8 Parameter Table

3.1.6 Routing Space Table

Federations, in order to enable, but also to limit, the flow of object attributes and interaction data between the federates, use declaration management (DM) services. Sometimes data distribution management (DDM) services are necessary to reduce further the volume of data delivered to federates. In these cases, a common framework is required for specifying the data distribution model for the federation. Routing spaces are the most fundamental way in which DDM exercises the control of data distribution. Routing spaces are specified in the Routing Space Table.

Figure 3.9 shows a partially completed example of the contents of the table.



Sample Routing Space Table

Routing Space Table					
Routing Space	Dimension	Dimension Type	Dimension Range/Set	Range/Set Units	Normalization Function
Bar_Order	Soda_flavor	Flavor_Type	Cola, Orange, Root Beer	N/A	linear_enumerated (Flavor)
	Bar Quantity	short	[1-25]	N/A	linear (Number_Cups)
Server_Order	Waiter_ID	short	[1-20]	N/A	linear (Waiter ID)

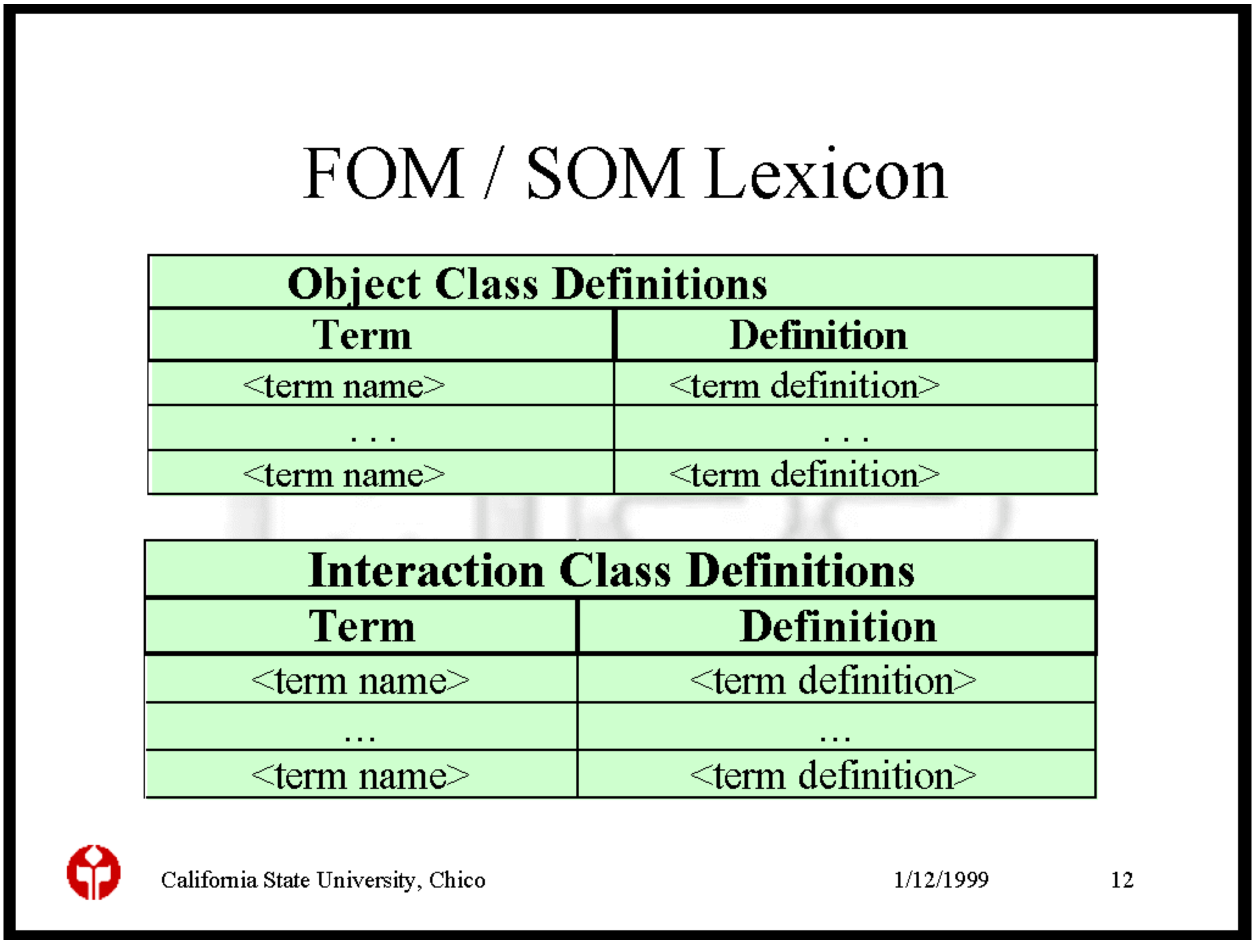
California State University, Chico 1/12/1999 11

Figure 3.9 Routing Space Table

The example in figure 3.9 includes a routing space *Bar_Order* that is used to filter drink orders. This routing space has an enumerated dimension (*Soda_Flavor*) and an integer dimension *(Bar_Quantity)*. The second routing space in the example filters on the IDs of the waiters that are in the range 1-20. No units are specified for dimension *Soda_Flavor* because it is an enumerated set. Nor are units necessary for *Bar_Quantity* or*Waiter_ID.*

3.1.7 FOM/SOM Lexicon

The FOM / SOM Lexicon is included in the OMT to provide semantic information necessary to an understanding of the information contained in the OMT tables. The lexicon consists of tables of definitions of object classes, interaction classes, attributes and parameters. Figures 3.10 show the simple templates used to describe this information.



FOM / SOM Lexicon

Object Class Definitions	
Term	**Definition**
<term name>	<term definition>
. . .	. . .
<term name>	<term definition>

Interaction Class Definitions	
Term	**Definition**
<term name>	<term definition>
…	…
<term name>	<term definition>

California State University, Chico 1/12/1999 12

Figure 3.10 FOM/SOM Lexicon

3.2 OMT Data Interchange Format (DIF) and Federation Execution Data (FED) Files

The HLA OMT data interchange format (DIF) is a standard file exchange format used to store and transfer HLA FOMs and SOMs between FOM / SOM builders via Federation Execution Data (FED) Files. DIF is built upon a common meta-model that represents the information needed to represent and manage object models created using the HLA OMT standard. A formal definition of the DIF in Backus Naur Form (BNF) and an example of a DIF file can be found in the HLA OMT Specification, (Annex E, pp. 39-57). An example of a FED file appears in the M&S HLA-Federate Interface Specification p. 235.

The question of how to document a simulation model in a formal way that expresses all of the details needed to judge its capabilities has long been a major concern of simulation developers and users. For HLA simulations, at least, the use of DIF solves this problem since it tells a potential user of a federate or federation, or a developer of a new HLA component, exactly what to expect in terms of the external characteristics of a federate, or the nature of the data exchange within a federation. Furthermore, the DIF provides the link between documentation and execution through the FED file, which contains the data necessary for execution of the simulation.

3.3 Object Model Development Tool (OMDT)

At first sight, the development of the OMT and the FED file looks like a daunting task, but tools are available that automate and simplify this process. The key tool provided for this purpose is the Object Model Development Tool (OMDT).

The OMDT, which is available free from the DMSO web-site, automates the process of constructing SOMs and FOMs. Once an object model has been constructed, the OMDT can generate the FED file required by the RTI to execute the federation. Full details of the use of the OMDT can be found in the HLA OMDT User's Guide available from DMSO.

Assignment

Use the *HLA Course Lab Notes* to help you make a FOM and a SOM for the initial version of *HelloWorld* used in Part 1 of this course. Generate a new FED file using the OMDT and compare against the FED file you have been using for *HelloWorld* in Parts 1 and 2.

Suggested Readings

Peruse *IEEE P1516.2, Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – HLA Object Model Template (OMT).*

Peruse *High Level Architecture (HLA) Object Model Development Tool (OMDT) User's Guide*