1. Modeling of a real-world system is the first step in simulation. Simulation models may be based on physical, mathematical, or logical representations; expert rules; empirical data, etc. in order to describe the behavior of the system being modeled. Simulation is the process of generating the behavior of the model using computing systems (computer, algorithm or human mind). In addition to its use as a tool to better understand and optimize performance and/or reliability of systems, simulation is also extensively used to verify the correctness of designs. Since the data available in the real world is abundant, we define a specification of the conditions called experimental framework, under which the system is observed. System, experimental framework, models and simulators are basic entities of modeling and simulation.

2. During the design and implementation of a simulator, various techniques and strategies may be adopted to model the behavior of a given system. Simulators are designed using either *continuous* or *discrete-event* [1] techniques to simulate a given system. Continuous simulators are characterized by the extensive use of mathematical formulae, which describe how a simulated component responds when subjected to various conditions. The discrete event modeling provides a general framework for time-oriented simulations of systems. Within the context of discrete-event simulation, an event is defined as an incident, which causes the system to change its state in some way. What separates discrete-event simulation from continuous simulation is the fact that the events in a discrete-event simulator can occur only during a distinct unit of time during the simulation - events are not permitted to occur in between time units. Discrete Event System Specification (DEVS) is a formalism used to create simulation model for discrete event systems. The recent development of the High Level Architecture for Modeling and Simulation (HLA) has
stimulated interest in the use of distributed interoperable simulation models.

3. The recent advent of the High Level Architecture for Modeling and Simulation (HLA) has greatly increased interest in the use of distributed, interoperable simulation model components. To date, most models using HLA have been developed in conventional highlevel languages (primarily C++). This paper discusses concepts by which HLA can be used to interconnect models developed using commercially available, off-theshelf simulation software. General approaches for adapting such software for use with HLA are presented. Two generalized solutions for the simulators SLX and SIMPLEX 3 are discussed, as well as problems with developing solutions for other tools like Pro Model and Automod.

4. The High Level Architecture is defined by:

a. rules which govern the behavior of the overall distributed simulation (*Federation*) and its members (*Federates*).

b. an interface specification, which defines the interface between each federate and the Runtime Infrastructure (*RTI*), a supporting software component that is responsible for providing communication and coordination services to the federates.

c. an *Object Model Template (OMT)* which defines how federations and federates have to be documented. Although HLA seems to have some similarities with CORBA, HLA offers more than CORBA can do for simulations tools. HLA has integrated mechanisms for the synchronization of simulation tools regarding time and data exchange as well as intelligent data distribution mechanisms.

5. HLA INTERFACE OF SIMULATION TOOLS

HLA defines a two-part interface which federates are required to use for communicating with the Runtime Infrastructure (RTI) (DMSO 1998). This interface is based on the ambassador paradigm. A federate communicates with the RTI using its RTI ambassador .Conversely, the RTI communicates with a federate via the federate's ambassador. From the federate programmer's point of view these ambassadors are objects and the communication between the participants is performed by calling methods of these objects.

6. HLA Integration Concepts

In order to enable simulation tools to access the HLA Application Programming Interface (API) it is inevitable to perform low level programming in a typical programming language like C++, Java, or ADA. It is highly desirable for simulation developers to only have to perform this task once for a variety of models. Therefore simulation model independent solutions are needed. This kind of approach can be classified as a "tool enhancement approach".