

Comparison of Foresight and Mil3 OPNET

Following is a detailed feature-by-feature comparison of Foresight compared to MIL3 OPNET. At a high level, Foresight is a general purpose system design and simulation environment suitable for all types of system design, and OPNET is a specialized design tool suitable for network modeling.

Foresight is a comprehensive system modeling environment for large scale system design. The general purpose modeling language makes Foresight suitable for modeling all types of systems. The Foresight modeling language is based on the Hatley/Pirbhai¹ real-time structured analysis methodology. Foresight adds executability to the system model through discreet event simulation allowing for functional, architectural and information modeling. Foresight is frequently used to model the behavior of complex systems containing many CPUs and other resources. Such models provide the ability to accurately predict the functional and performance behaviors of multiple functional threads contending for common resources.

OPNET is designed for modeling architectures for networking devices and protocols, using modeling constructs such as network models, node models and process models. OPNET Network Models define positions of communications nodes within a network. Nodes are described by a block structured data flow diagram. OPNET Node Models depict the interrelation of processes, protocols, and subsystems. A Process Model defines each programmable block. State transition diagrams define OPNET process models, with actions within the states being described by C/C++ language. OPNET includes a library of functions designed for protocol programming. OPNET has a library of standard communications network models.

Foresight's modeling capability extends beyond modeling just processors and busses. Using the resource constructs in Foresight, any process that is constrained by the availability of an object can be modeled. The process resource constructs allow for the modeling of any shared resource that does work. As an example, in an embedded system design the constraints are system memory size and CPU work rate, but in a factory the system constraints may be machine tool availability and human labor. Both can be modeled using Foresight.

A Foresight system can be decomposed into subsystems, allowing for concurrent development of the system model. Subsystems are composed of data flow diagrams, which can be hierarchical (i.e. data flow diagrams contain instances of other data flow diagrams to an unlimited depth.) At the lowest level, hierarchical state transition diagrams and language-based process models, called Mini Specs, describe processes. Data passes between processes over Data Flows. A variety of different data flow types, each with a different semantic behavior, are supported. Supported data flow types include signal (event), discrete, continuous, and queued. In addition, the data type of the information transmitted by a flow is defined in a subsystem-wide Data Dictionary. Supported data types range from scalar types to arbitrarily complex structures, arrays, and sets.

Foresight provides library elements for data manipulation, database access, user input and output, math, logic, signal processing, timing and validation. Foresight also allows users to define their own library elements for reuse in other systems models.

The language used to describe behavior in Foresight state transition diagrams and mini specs uses VHDL syntax and semantics. There are additionally over 120 library functions within the language to support the development of complex system functionality. External C/C++ functions and procedures may be called to allow the inclusion of extended functionality within the performance model.

Foresight also provides an interface to requirement traceability tools such as QSS/DOORS. This allows models defined in Foresight to be linked to requirements in the requirements database. This linkage allows users to determine which specification items in the Foresight system model result from a given requirement in the database, and vice-versa. In addition, simulation-time parameters can be extracted from the requirements database and passed in to the simulation. The Foresight model can then take the form of an executable specification.

Foresight has successfully been used to model the performance of large system architectures that contain multiple processors connected by busses and network connections. Foresight is also being used to provide system-level test benches for the implementation of embedded system designs, with links to HW/SW co-verification tools such as Mentor Graphics' Seamless and ModelSim products.

¹ D. Hatley and I. Pirbhai, *Strategies for Real-Time System Specification*, Dorset House, New York, 1987.

| Mil3 OPNET | Foresight |
|---|--|
| Hierarchical network models. | Unlimited Hierarchy of models, models not limited to Network Systems |
| Object-Oriented modeling | Object-Oriented Modeling based on Hatley/Pirbhai methodology |
| Finite state machine modeling using C/C++ language | Hierarchical State Machines using VHDL language Minispec procedure using VHDL language |
| Library of functions to support creation of network models. | 90 Library Elements and calls to support functional and architectural modeling. User may create additional library elements by defining Re-Usables. Procedural modeling language capability can be extended through creation of Callable MiniSpecs and linking in external 'C' and 'C++' functions. |
| | Detailed software implementation and algorithms can be included in the system model via External Call Interface. |
| Wireless, point-to-point and multipoint links | Wireless, point-to-point and multipoint links via the Foresight Teledoor library element. |
| Geographical and dynamic mobility modeling | |
| | Resource and Process Modeling Constructs allow modeling of any system where a shared resources are constraints. |
| API's to HLA and OMI | FS/Bridgeway allows interfacing to any external simulation environment. Specialized links to Mentor Graphics Seamless products to support HW/SW Codesign |
| Integrated analysis tools | Simulation-time and post-processing analysis tools included with the product. Export to spreadsheet formats via MiniSpec functions also supported. |
| Animated Simulations | Animated Simulations |
| Integrated debugger | Integrated debugger |
| Import data from other environments | File input functions allows importation of data from any ASCII source. |
| Integrated simulation engine | Highly efficient simulation engine within the Foresight modeling environment. FS/CoderCPP allows generation of standalone executable of the simulation model providing 50 x improvements in simulation speed. |
| Runtime environment (Modeler XE) | Runtime environment included with product. |
| Solaris, Windows NT, and HP-UX | Solaris, Windows NT, and Linux |