



AnyLogic 4.1

Overview



© 1993-2002 XJ Technologies
www.xjtek.com

Copyright © 1993-2002 XJ Technologies. All rights reserved.

XJ Technologies Company Ltd

AnyLogic@xjtek.com

<http://www.xjtek.com/products/anylogic/40/>

What is AnyLogic?

AnyLogic is a general-purpose modeling and simulation tool for discrete, continuous and hybrid systems. Its application area includes

- Control Systems
- Manufacturing
- Telecom
- Mechanics
- Military
- Traffic
- Supply Chain
- Networks
- Chemical
- Education
- System Dynamics
- Logistics
- Computer Systems
- Water Treatment

AnyLogic modeling language offers maximum power and flexibility and allows multiple modeling approaches:

- UML-based OO modeling
- Block-based flowchart modeling
- Statecharts (state machines) – regular and hybrid
- Differential and algebraic equations
- Explicit modeling in Java

AnyLogic is the most innovative simulation software based on the latest advances in the complex system design methodology. It is the first tool that brings the power of Unified Modeling Language to wider and more diverse simulation modeling domain.

AnyLogic is the only commercial tool supporting hybrid state machines – powerful and elegant construct to combine discrete and continuous behaviors.

AnyLogic gives you the unique ability to run simulation models on any Java-enabled platform, or even as applets in Web browsers.

AnyLogic is the only visual tool that provides for creation of truly dynamic models – the ones with dynamically evolving structure and component interconnection.

The Modeling Language of AnyLogic

The modeling language of AnyLogic is an extension of UML-RT – a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The main building block of AnyLogic model is active object.

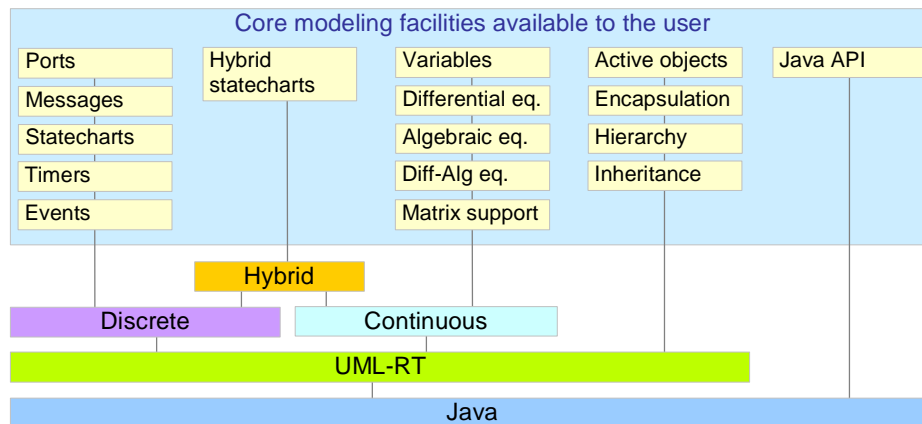


Figure 1 AnyLogic modeling framework

Active objects have internal structure and behavior. They may encapsulate other objects to any desired depth. When you develop AnyLogic model, you actually develop classes of active objects and define their relationships. At runtime the model can be viewed as a hierarchy of active object instances.

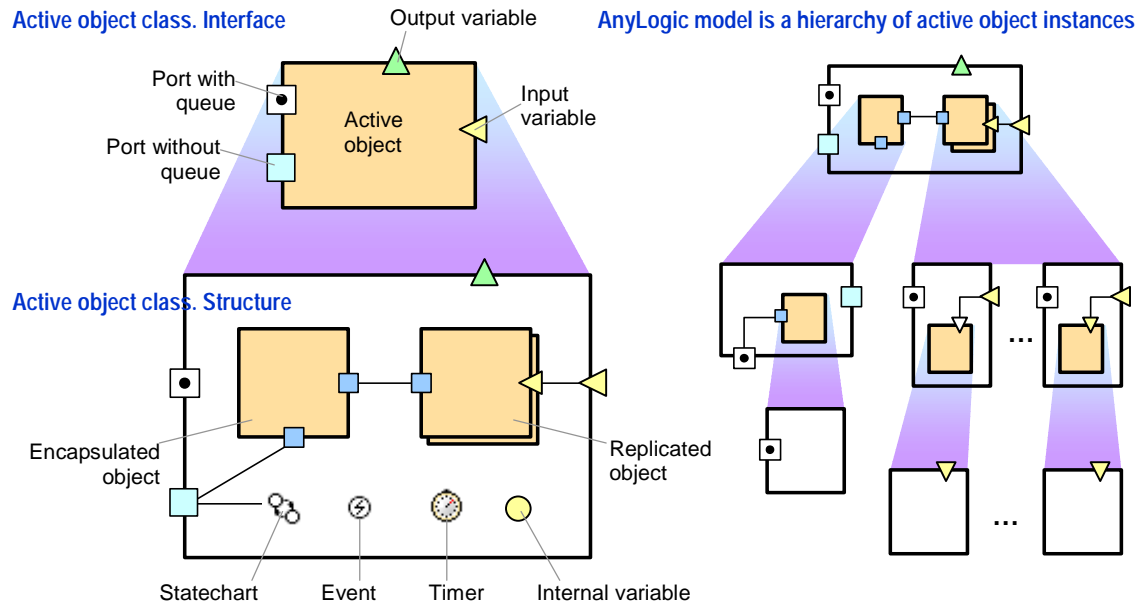


Figure 2 Active objects – building blocks of AnyLogic model

Active objects interact with their surroundings solely through boundary objects: ports (for discrete communication) or variables (for continuous communication). This de-coupling of the internal object implementation from any direct knowledge about the environment makes active objects highly reusable.

Discrete Modeling

The discrete modeling framework of AnyLogic includes message passing mechanism at inter-object communication level, and statecharts and various primitives (timers and events) at the level of object internal behavior.

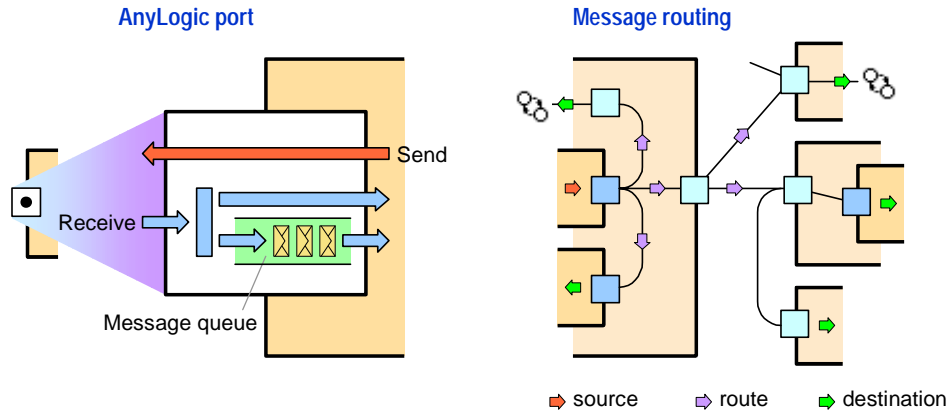


Figure 3 Ports and message passing in AnyLogic

Messages are sent and received through ports. Ports are bi-directional and may have queue for incoming messages. When a message is sent out it is broadcasted along all the external connections of the port. The incoming message may be stored in the queue, and/or forwarded along the internal connections. The default behavior of ports may be changed in any desired way.

The activities within the object can be defined using timers in very simple cases or using statecharts (extended state machines) in cases when event and time ordering becomes more complicated.

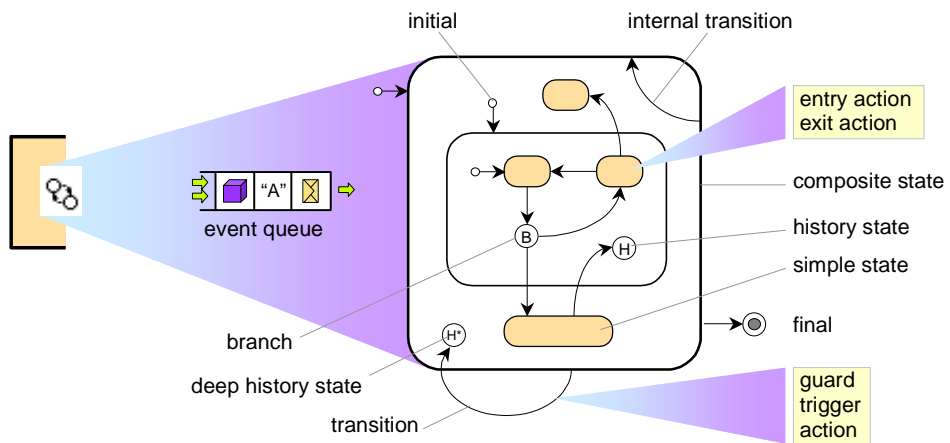


Figure 4 AnyLogic statecharts

AnyLogic supports UML statecharts including composite states, branches, history states, etc. Statechart transitions may be triggered by messages, various kinds of events and conditions, and timeouts.

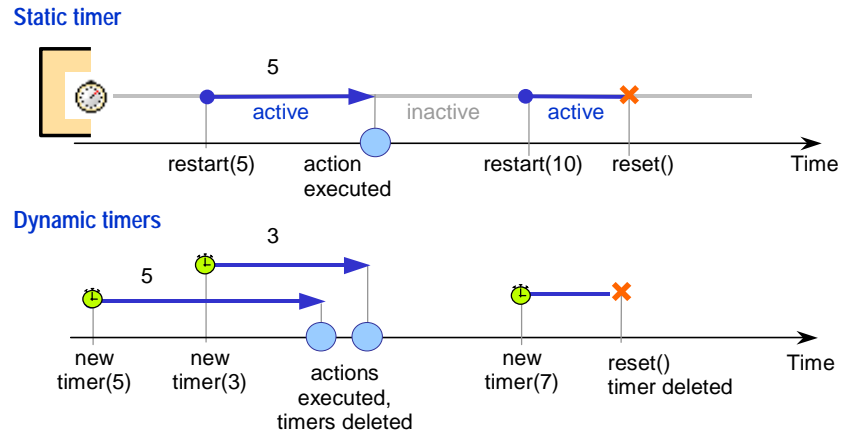


Figure 5 Static and dynamic timers in AnyLogic

There are static and dynamic timers in AnyLogic. The latter are used to schedule multiple events, which may be associated with multiple objects or messages.

AnyLogic simulation engine carefully handles the execution of discrete events, preserving the ordering and atomicity of actions declared in the language semantics. Simultaneous events are executed in random order

Continuous Modeling

Whereas the discrete logic in AnyLogic is specified using statecharts, events, timers and messages, the continuous processes are described with algebraic-differential equations over continuously changing variables. Those variables may be exposed at the active object interface and connected to other objects.

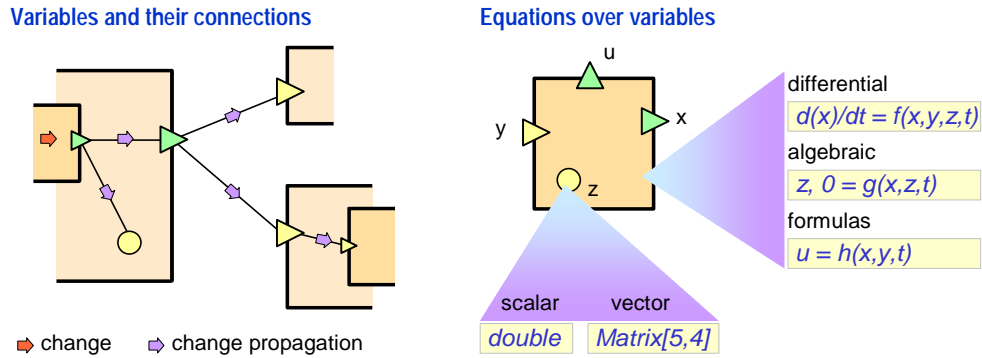


Figure 6 Active objects – building blocks of AnyLogic model

AnyLogic supports ordinary differential equations, algebraic equations and their combinations. The variables in the equations may be of scalar or vector types. The set of numerical methods included with the simulation engine can handle both stiff and non-stiff systems. External numerical libraries may also be used.

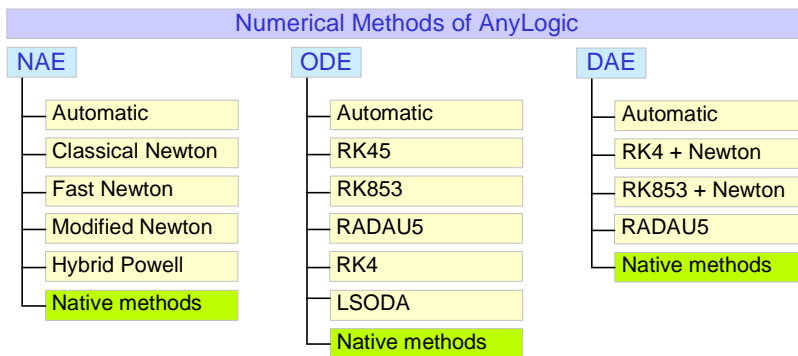


Figure 7 Numerical methods of AnyLogic

The simulation engine automatically checks the correctness of the (dynamically changing) equation set on the fly, tunes numerical methods, detects and breaks algebraic loops.

Hybrid Modeling

The world around us is in fact hybrid: continuous time processes are mixed with discrete events. In many real world systems these two types of behavior are interdependent, and this needs a special approach in simulation modeling. Traditional tools usually support either discrete-only or continuous-only modeling, or some awkward combination.

AnyLogic is the one and only commercial simulation tool originally developed for hybrid modeling. It is equally powerful in discrete and continuous domains and especially in cases when those two behavior types are highly interrelated.

Different equation sets are active in different states

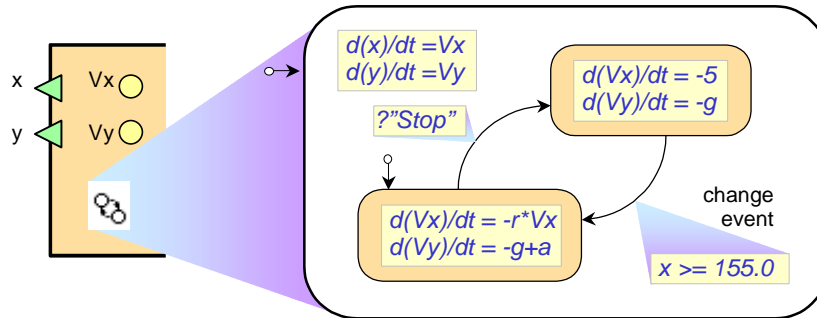


Figure 8 Hybrid statecharts

The most notable feature of AnyLogic with respect to hybrid modeling is hybrid statecharts. In hybrid statecharts you can associate a set of equations with a statechart state. Then state transitions will alter the continuous behavior. And also you can specify a condition over continuously changing variables as a trigger of a transition. Then continuous process will drive the discrete logic.

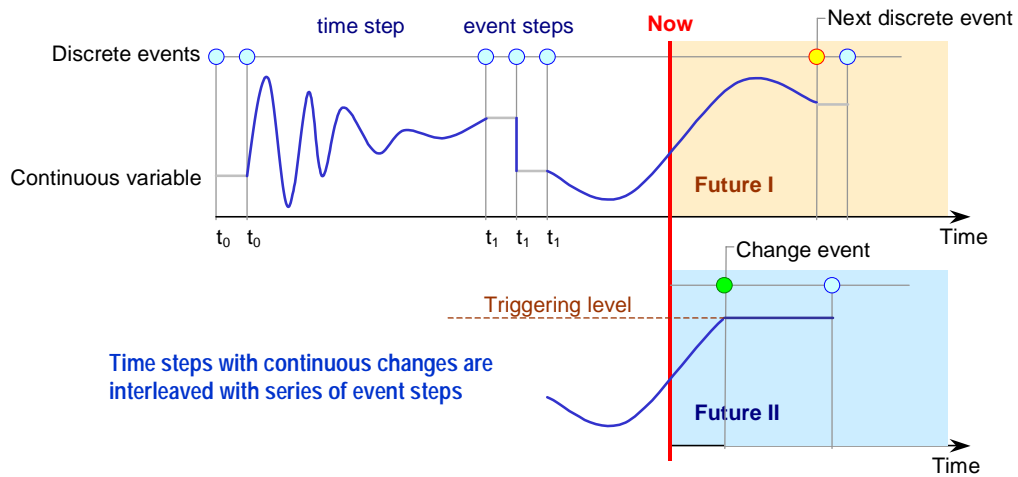


Figure 9 Hybrid system trajectories

This natural integration of two behavior types in this simple extension of UML provides for very compact, concise and efficient models of hybrid systems.

Advanced Modeling Techniques

The visual representation of AnyLogic model is mapped to Java, and the model is naturally open for adding Java code at any level. This can be done directly in the model editor. Using Java you can easily customize the functionality of objects, ports, messages, timers, etc. Combined with the OO modeling paradigm, this gives you the unique power and flexibility in dealing with sophisticated or exotic modeling tasks.

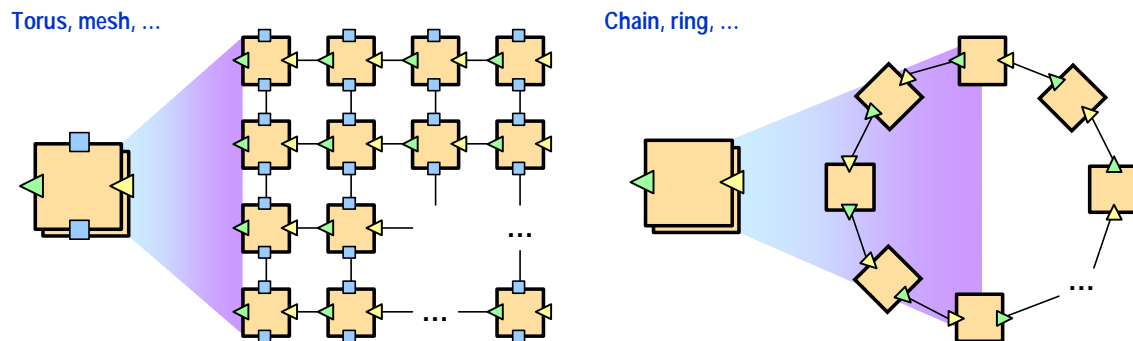


Figure 10 Systems with regular topologic can be easily modeled in AnyLogic

Regularly structured systems

Suppose you need to model a regularly structured system, e.g. mesh, torus, hypercube, chain, ring, etc. of variable size. In AnyLogic you simply declare a replicated object, specify the number of instances as a parameter, and write a couple of Java lines to connect them in the desired way.

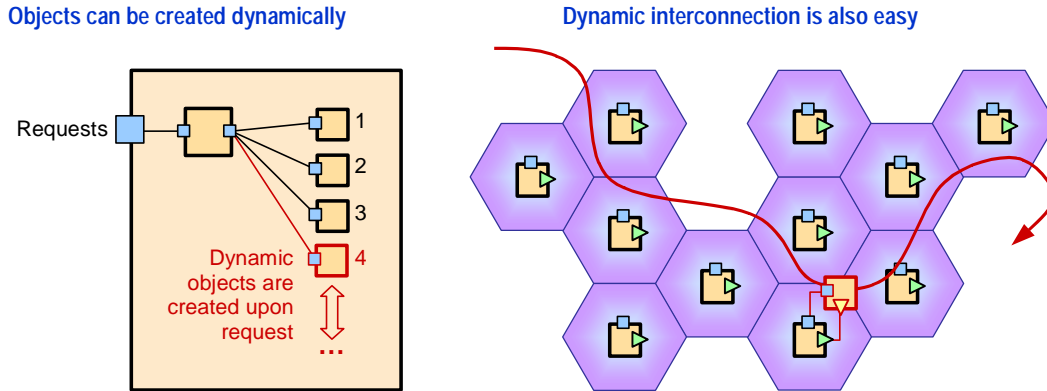


Figure 11 Objects can be created, destroyed and interconnected dynamically

Dynamically changing structures

If objects and/or object connections in the system you are modeling have limited lifetime (are created and destroyed as the system evolves), this can be easily handled by calling the constructing / disposing or connecting/disconnecting methods at the right moments of time.

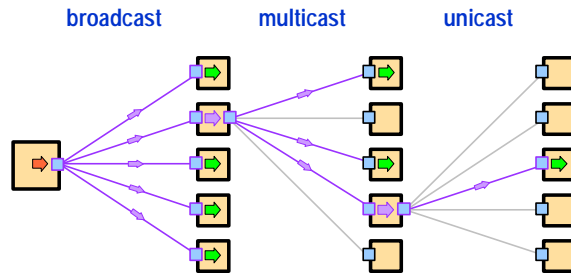


Figure 12 Message passing in AnyLogic can be customized in any way you wish

Customized message passing

You can make AnyLogic message more intelligent by defining its methods, you can pack one message into the other, or you can change the semantics of message passing by overriding the default behavior of ports. For example, you can model message passing with addresses on top of broadcast – all you need is just to define a destination field in the message class and write simple filtering code at ports.

Adding Java modules

In case you feel more comfortable with writing algorithms in Java than drawing statecharts, you can easily do that. The handwritten Java methods will be handled as “threads” running concurrently with all other activities in the model.

In general, you can add arbitrary Java classes, empowering your model with wide variety of existing Java controls and libraries. You can program network communication, interact with physical devices, and using JNI you can access code written in other languages.

Animation

AnyLogic offers original technology that enables the user to rapidly create interactive 2D animations. Animations are developed within the model editor, but they are logically separated from the model.

You associate individual drawings with active object classes, and you can place the drawings of the encapsulated objects on the drawing of the container object. The global picture is assembled from those drawings when the model is created. This way the animations become highly reusable and scalable. For example, in case of a replicated object there will appear as many copies of its images as there exist its instances.

Animations may include elementary shapes as well as various types of indicators and graphs. Interactive elements such as buttons, sliders and edit fields may be added, so the user is able to impact the model at runtime. There is also rich API that allows you to develop very sophisticated animations.

AnyLogic animation is also 100% Java, just like the model itself, and it is displayed in the browser window in case you generate an applet from the model.

Cross-Platform and Web simulation

As long as AnyLogic models (including the generated part, the animation and the engine) are 100% Java, they can be run on any Java enabled platform. If needed, the Windows-based viewer can connect to the model remotely via TCP.

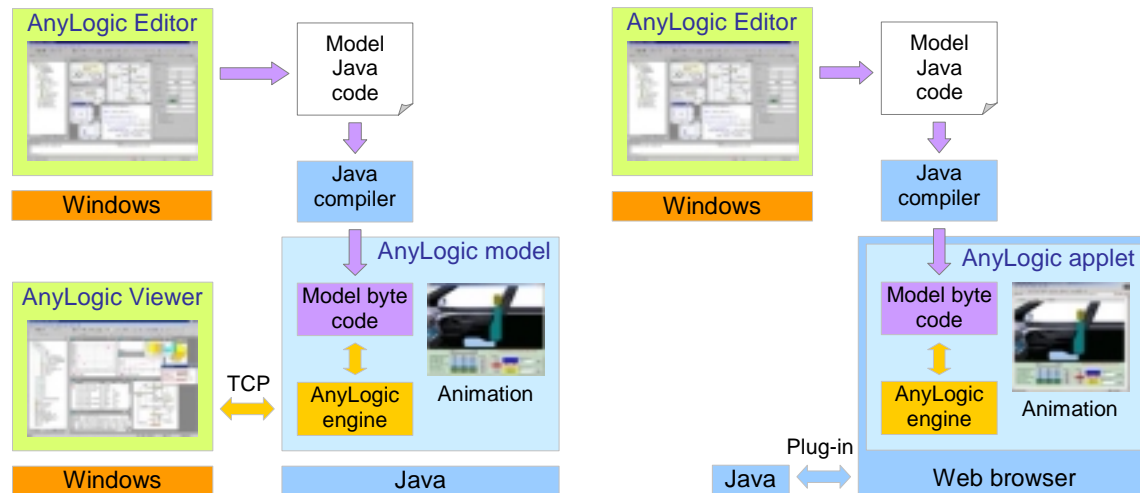


Figure 13 How AnyLogic models run

Moreover, you can pack the model into an applet. The applet displays the interactive animation and includes a simple control panel with which you can run, stop and restart the model. The simulation engine used by the applet has size of less than 300K. It is loaded to the client machine once and then shared between the models.

Framework for Experiments

The framework includes stochastic input to the model, collection, analysis and presentation of statistical output, parameter variation mechanism, interfaces to databases, spreadsheets, and other data storages, optimization toolbox, and also rich engine API.

Stochastic modeling

AnyLogic models can be stochastic as well as deterministic. By default, AnyLogic uses standard Java random number generator, but you can plug in your own. There are over 35 probability distribution functions in AnyLogic package.

Datasets

AnyLogic dataset objects help you to collect, display, and analyze data during the model execution. Datasets can contain scalar or vector samples, possibly with timestamps. Each dataset has a block of statistical data including mean, variance, minimum, maximum, confidence intervals, etc. AnyLogic model viewer supports various dataset presentation types: plots, histograms and Gantt charts.

Parameters

Active objects may have parameters, which can be linked to the parameters of the encapsulated objects. Parameters can be changed at runtime, and those changes propagate down the model hierarchy. You can define handler methods invoked on parameter changes.

Interfacing external data storages

AnyLogic model can work with databases, spreadsheets and other files. For example, you can extract the model parameters from a database and output the statistics into a spreadsheet.

Optimization

AnyLogic optimization toolbox includes simple minimum search, Newton-type and random search methods. Interface to external optimization software is provided. Using AnyLogic engine API you can also write your own control of simulation replications, organize parameter variation, or implement a custom optimization algorithm.

Other Features

Libraries are collections of active object classes, animations, message classes, and Java modules developed for some particular application area or modeling task. Libraries provide for better reuse of objects across multiple models. Several libraries come with the tool, and you can easily create your own ones. By developing the right library you can convert AnyLogic into a high-level modeling tool with point-and-click interface for a specific domain.

Model Editor

AnyLogic model editor is built up to the state-of-the-art Windows UI level. It features customizable windows, toolbars, colors, images, drag and drop editing, diagram zooming, syntax highlighting, etc. The workspace tree provides easy navigation throughout the project. The editor performs on-the-fly checking of types, parameters, and diagram syntax. Errors are highlighted. The editor generates comprehensive cross-referenced HTML or XML report including diagrams, element properties, and code.

Model Viewer / Debugger

The visual model viewer/debugger is a part of AnyLogic. It features:

- Local or remote TCP connection to model
- Step / Play / Run modes with a variety of options
- Easy navigation with model explorer: any object is accessible
- On-the-fly animation of structure and statecharts
- Graphical breakpoints, log and inspect windows
- Statistics visualization (Histogram, Gantt, plots)
- Event viewer
- Variable and parameter changing at runtime

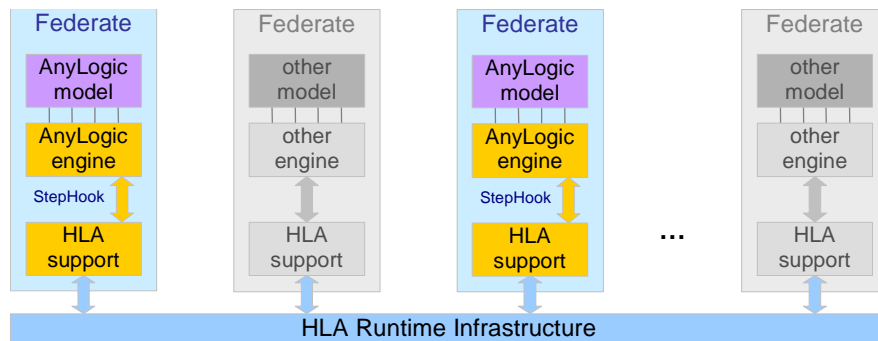


Figure 14 Distributed simulation with HLA support

Distributed simulation with HLA support

AnyLogic supports HLA – a standard framework for simulation distribution and interoperability. AnyLogic Java HLA module communicates with the simulation engine via special interface and provides for coherent work with (possibly distributed) models of arbitrary origin as well as other AnyLogic simulations.