New Features Guide

New Diagrams
Diagram Framework Enhancements
New Formal Documents
and
additional refinements
Contents
Overview of GENESYS 2.0 .......................................................... 5
New Diagrams .............................................................................. 6
   N2 Diagram ........................................................................... 6
   Interface and Physical N2 Diagrams .......................................... 6
   Packages .............................................................................. 7
   Spider Diagrams ..................................................................... 8
   Use Case Diagram .................................................................. 9
Diagram Framework Enhancements .............................................. 10
   New Entities from the Toolbox .................................................. 10
   Images as Nodes .................................................................... 10
   Node Image Path .................................................................... 11
   Database Classes Image Assignment ........................................ 12
   Element Image Association ...................................................... 13
New Formal Documents and Enhancements ................................. 14
   System Segment Specification (SSS) ......................................... 14
   System Description Document (SDD) ........................................ 14
Other Enhancements ..................................................................... 14
   PUID Generation Utility ........................................................... 14
   Streamlined Activation Experience .......................................... 14
   Support for A4 Size Paper ........................................................ 14
   Improved Performance in Simulator and Report Generator .......... 14
   Schema Migration Wizard ....................................................... 15
   Expanding the model .............................................................. 15
Comprehensive list of GENESYS 2.0 Schema Changes .................. 15
   General Changes Included in All Schemas (Including Legacy Schemas) .......................................................... 15
   Systems Engineering Schema Changes (Also Reflected in the DoDAF 2.0 Schema) ............................................ 16
   DoDAF 2.0 Schema Changes .................................................... 18
THIS PAGE INTENTIONALLY LEFT BLANK
Overview of GENESYS 2.0

Building upon the successful launch of GENESYS in 2011, GENESYS 2.0 delivers enhancements and richness, resulting in greater out-of-the-box value. New diagrams and robust representations give you more control and faster response, while still utilizing the proven STRATA™ approach.

What's New?

- **New Diagrams**: We’ve brought you the N2, Interface N2, Physical N2, Spider, Package, and Use Case diagrams.
- **Diagram Framework Enhancements**: Enhancements include improved drag-drop support through new constructs in the Toolbox. We’ve also included the ability to show images in place of nodes on diagrams.
- **New Formal Documents**: Using the drag-drop report designer, we’ve created the System Segment Specification and System Description Document. In creating these we added support for a table of contents in your document and use of A4 size paper.
- **Other Enhancements**: You’ll find a new wizard that facilitates the maintenance of the Project Unique Identifier PUID attribute values for various entities across the project, a new schema migration tool to help you upgrade your project to the latest schema, and improvements to performance in general use, data import, and running reports. Also included is an enhancement to the Activation Wizard that allows a user to switch between local and network licenses.
New Diagrams

We’ve rounded out the set of diagrams in GENESYS to give you more ways to create and examine your model.

N2 Diagram

The N2 (pronounced "N-squared") diagram represents the logical data flow for a system or system segment. Part of the behavioral (logical architecture) representation set, the N2 diagram displays the data dimension of the behavior model and helps focus attention on this subset of the model. In particular, this is helpful in partitioning and allocating the system behavior to manage internal and external interfaces.

The N2 diagram is available for elements in the Function class (as well as any other subclasses of ProcessingUnit). On a functional N2 diagram, the subfunctions are shown on the main diagonal forming an N x N matrix of cells. Items that are output from a function are shown in the function’s row. Items that are input to or trigger a function are shown in the function’s column. If multiple items are output from and input to/trigger the same pair of functions, multiple items will be shown in the same item cell. If no items are exchanged between a pair of functions, the item cell will be empty.

The N2 diagram has been extended to display external inputs and outputs which represent external interfaces for this function. Items appearing in the top row are inputs / triggers for the function that either (i) are output by a function not displayed on this diagram or (ii) are not output by any function at all. Similarly, items in the right-hand column are outputs that are either (i) input to/trigger a function not displayed on this diagram or (ii) are not input to / trigger any function in the system model.

Interface and Physical N2 Diagrams

These diagrams complement the existing block diagrams, presenting the information in a more structured manner suitable for interfacing, clustering, and packaging analysis.

Both diagrams use the traditional N2 representation of primary blocks on the diagonal and related objects on the off-diagonal. In these representations, the child components are shown on the diagonal. The off-diagonal represents the interfaces (interface N2) or links (physical N2) that connect the components.

In this format, the nature of the connections becomes clear. You can quickly assess whether the model has a single highly connected component (which could represent a technical or interface risk), multiple clusters of interconnected components (valuable insight for packaging), etc. While the same information can be gleaned from a block diagram, this insight screams out when presented in an N2 format.

There are several notable items with respect to physical representations of N2 diagrams:

- Whereas there is clear directionality in a behavioral N2 (an item is output from one function and input to or triggers another function), the physical N2s speak more to connection than directionality. For that reason, no arrows are shown on the physical N2s. Instead, the diagram simply represents who is connected to whom.
- The lack of directionality means that half of the off-diagonal locations are redundant. If A is connected to B, we know that B is connected to A. Rather than showing this information twice, only the upper half of the diagram is used. The lower off-diagonal cells will be empty by definition. In addition, the external input and external output locations become redundant. For this reason, we have hidden the external input cells at the top of the diagram.
The physical N2 diagrams are generally higher level diagrams. For this reason, they are level 0 diagrams which do not use the connected thru and joins thru relationships. Instead, they rely solely on the connections defined for the specified components.

External connections on a physical N2 represent one of two conditions. First, it may be a connection with an outside component or part of the environment. Second, it may be a connection with a child of a component not shown on this diagram. In that case, the connection should be with the component itself, not the child. This connection can then be further elaborated at a lower level of the physical architecture.

There is one notable command that has been included with the N2 diagram to support clustering analysis for both physical and behavioral N2 diagrams. The Diagram>>Change Node Position allows you to change the position of the selected component / function on the diagonal. The diagram then redraws the matrix in this configuration. This capability dramatically increases the value of the N2 in visualizing (behavioral) allocation and (physical) packaging.

**Note:** By default, the physical N2 diagrams compute their order from the sorted order of their children. The behavioral N2 diagram computes its order by traversing the decomposition structure (visible in an activity diagram or an Enhanced Functional Flow Block Diagram EFFBD). Once you manually change the node position on an N2 diagram, all future additions will be appended to the end of the diagonal. Resetting the diagram layout via the Layout Diagram command will restore the computed order, and GENESYS will once again maintain the order automatically.

**Packages**

To date, navigation within GENESYS has been database-centric. To access the element of interest, you started by drilling into the specific class and then down into folders until you found what you were looking for. For those with a database-centric frame of reference and command of the model-based systems engineering (MBSE) language, wonderful! Those without these pre-requisites could easily struggle.

With GENESYS 2.0, we raised the question of alternate navigation approaches, not to replace the database-centric paradigm but to complement it. We hypothesized role-based structures, level-based structures, and view-based structures (to name just a few). To enable all of these approaches, we have leveraged the SysML approach of packages.

In SysML, packages are fundamentally tools for model management. They are a type of container that can hold diagrams, elements, and other packages (creating a nested structure). GENESYS now includes a Package class that enables this behavior. As packages are created, they are shown in the project list pane, enabling you to implement whatever packaging and navigation approaches are desired.

Of particular note:

Elements can be included in more than one package. This allows you to establish multiple package-based navigation schemes – by Integrated Project Team IPT, by level, …

As with element folders, packages can be set to show subpackage contents. This allows you to look all the way down the package hierarchy in a single, integrated list. Elements included in subpackages are shown in italics.

Package contents can span multiple classes. This enables you to apply operations across elements from multiple classes (such as setting access control, purging, baselining, etc.).
Spider Diagrams

To date, GENESYS has focused on structured engineering representations with defined formats – EFFBDs, hierarchies, and sequence diagrams. These formats are critical to the engineering analysis but sometimes limit greater insight into and communication of the essence of the system. The spider diagram helps restore and highlight that context.

Presented in a free-form layout, the spider diagram presents a complete contextual view of a set of entities and their interrelationships. The diagram blends concepts present in an Entity Relationship ER diagram (displaying relationships relative to an entity of interest) with concepts of a hierarchy diagram (showing multiple levels of relationships). However, unlike a hierarchy diagram, each entity is represented once and only once. In addition, the free-form presentation does not artificially imply a hierarchical relationship that may not exist. The result is an extremely powerful representation – neither traditional nor SysML – to further enrich our toolkit for analysis and communication.

Available on all elements regardless of class, the spider diagram is opened on a specific element and hierarchy definition. The element represents the center of reference for this diagram. The hierarchy definition specifies the relationships and target classes of interest.

From a command and preferences perspective, the spider diagram has a great deal in common with the hierarchy diagram:

The diagram is opened to a default number of levels (specified in the preferences). You can change this number in the diagram options or expand / collapse specific nodes of interest. Expanding and collapsing is handled by control-double-clicking on the nodes of interest.

The diagram is defined by the combination of a central entity and a hierarchy definition. You specify which hierarchy definition is desired when opening the diagram (the default associated with the class is used on the project explorer tab). A command to switch to a different hierarchy definition is available from the toolbar. Like the hierarchy diagram, because each entity can have a number of different spider diagrams associated with it, GENESYS automatically tracks and reuses the settings associated with each specific combination of element and hierarchy definition.

Adding elements is best handled by dragging elements on to nodes from the Toolbox to establish the desired relationships.

While there is a great deal of commonality (and one could argue repetition) between the hierarchy and spider diagram, there are a number of notable differences as well.

The spider diagram is a free-form layout. Nodes can be moved anywhere on the diagram, and labels can be moved as well. This presents great flexibility in presentation, but also transfers additional maintenance to the user (with great power comes greater responsibility). In this aspect, the spider diagram has more in common with the block diagrams than anything else.

Whereas the same element can be repeated on a hierarchy diagram (with only the first occurrence being expandable), an entity will only appear once on a spider diagram. This emphasizes the interconnected nature of the model.
Note that this representation introduces some key interaction differences. If you select a node on a spider diagram and ask to remove it, you will remove all relationships which connect it to elements on the diagram. If you wish to remove just a single relationship, select the associated label and use the Remove Target command.

**Use Case Diagram**

SysML use cases are effectively the names of MBSE threads, capturing concepts that should be translated into threads and the integrated system logic. The conceptual modeling flow (in top-down design) begins with the requirements. From the requirements, you define the system boundary – physical and functional – and identify the context. From this context and the operational concept that drives it, use cases are identified. Use cases are then elaborated by functional threads which provide the insight needed to develop the integrated logic.

The subject (the large box in the middle of the diagram labeled Surveillance System) identifies the system that this use case describes. Note that this subject is inherited and does not have to be linked to every level of the use case model. GENESYS starts with the element Use Cases and looks up the ancestor hierarchy until it finds a use case that describes a component. In this traversal, GENESYS considers the included in, extends, and generalization of relationships.

Two levels of use cases are represented within the system frame:

The direct children of Use Cases as identified by the *includes* relationship. In this example, Monitor Environment and Setup Track are *included in* Use Cases.

The use cases linked to these children by the *includes, extends*, and generalization of relationships. In this example, Initialize System and Shutdown System are *included in* Monitor Environment. Handle Camera Fault *extends* the Monitor Environment in the event of a fault. Manually Monitor Environment is a *specialization of* the Monitor Environment use case.

The actors represented around the boundary of the diagram are the components involved in (i) the use case on which the diagram was opened plus (ii) any use case shown on the diagram.
Actors are shown in two different formats. By default, components of type Human are drawn as stick figures with the name displayed below. All other component types are drawn as rectangles with a corresponding icon template defining the content. In the example above, no non-human actors are displayed.

Classification (the generalization of / kind of relationship) is shown for both use cases and actors if both elements in the relationship appear on the use case diagram.

Extension points represent the places at which a use case can be extended. On each extends relationship, there is an Extension Point attribute. This field is displayed in the label of the extended use case as well as on the line connecting the extending use case. In the example, Handle Camera Fault uses the extension point fault for Monitor Environment.

Diagram Framework Enhancements

New Entities from the Toolbox

When using the new entity options on the Insert tab, you can create new GENESYS elements simply by dragging these constructs onto the diagram. In addition to dragging existing nodes onto the diagram, you can drag a New Node which will auto-create a new function and insert it accordingly. This can accelerate model development and enable you to move at think-speed, much like standing at a white board and sketching your system while having GENESYS build the model behind the scenes.

Images as Nodes

GENESYS 2.0 introduces the ability to represent GENESYS elements as traditional geometric frames or as graphical images. For an individual node or the entire diagram, you can select the representation that best suits your needs. In this case, a picture is truly worth a thousand words.
Behind this feature are a host of changes that provide a rich and flexible framework. Following is a step-by-step walk-through of the various aspects from the project-level on through the individual diagram node.

**Node Image Path**

As with the existing external references in GENESYS, user-provided graphical images used on diagrams are maintained and managed externally to GENESYS. This provides a highly flexible, high performance system allowing you to reference the graphic of interest. To simplify management of these external graphic files, each project has a setting for its node image path.
Analogous to other GENESYS external references, individual images can be located within a base graphics path or outside of it. Images within this base path directory structure will automatically be referenced relatively, easing the management from machine to machine. Images outside of this directory structure will be referenced absolutely, meaning that multiple users must be able to reach the same location using the same network path.

Database Classes Image Assignment
GENESYS 2.0 includes the ability to associate images with database classes. Not only does this enable the display of graphical images on the schema-side diagrams, it provides a default image for all elements of that class. As with the color model in GENESYS, when elements are queried for their graphical images, they first look to see if an image has been specified directly on the element. If not, they then look to their class and return the image there, if any.

All GENESYS schemas have been extended with an option to define images for each class. This provides a strong default representation, should you choose to use class-based images as shown below.
Element Image Association

While images can (and should) be used to visually represent database classes, the primary use will be to associate images with individual elements. All GENESYS elements include a new node image attribute for all classes.

By default, the node image attribute is shown on the Properties tab of the property sheet, immediately below the node border color attribute. The property sheet displays the current image identified for the element, and the edit button in the right of the pane allows you to identify the appropriate external image.

In addition, a command to show the image for the selected nodes has been added to diagrams on the Properties tab of the Toolbox. It is context sensitive and only appears when an entity is selected. Note that in addition to setting the node image attribute for the selected elements, this command toggles the node to use the image display rather than the geometric frame.

**Tip #1:** Don’t overdo a good thing. Emphasize graphics for level 0 / level 1 and critical stakeholder communications.

**Tip #2:** Use moderation in your color palette. GENESYS supports up to 16M colors, but 256 color bitmaps produce almost the same level of results.

**Tip #3:** Use moderation in image size. A 128x128 image gives richness without increasing the diagram (pixel) size.

**Tip #4:** When toggling between geometric frames and graphical images, the auto-size command is your friend. You can quickly resize a specific icon or – if nothing is selected – resize all nodes on the diagram.
New Formal Documents and Enhancements

Leveraging the drag-drop report designer, our team has built for you the System Segment Specification and System Description Document. We also added support for a table of contents and the use of A4 size paper.

System Segment Specification (SSS)
The SSS specifies the requirements for a system or subsystem and the methods to be used to ensure that each requirement has been met. This report, commonly required by customers, complies with DID DI-IPSC-81431A and can be easily generated using the GENESYS reporting framework.

System Description Document (SDD)
The SDD describes system components, originating requirements, design constraints, performance requirements, issues and decisions, risks, use cases, functional behavior models, items, resources, and interfaces. It also includes a Requirements Traceability Matrix, list of Acronyms, and a Glossary.

Other Enhancements

PUID Generation Utility
Available from the Project ribbon tab, the PUID Generation Utility will allow you to generate PUIDs for classes you select. A class must have its abbreviation property set and contain a PUID attribute to be eligible to use this utility. The assignment of a PUID to each element in such classes supports testing and traceability for those projects conforming to Department of Defense Data Item Descriptions (DIDs.)

Streamlined Activation Experience
The Activation Wizard now prompts users if they wish to use a local or network license. Without this prompt, a user who already had one type of license available would not be able to switch to another type. This enhancement is also included in the License Utility, easily launched from the Start menu. With this a user can switch from a local to a network license, and vice-versa. In teams where more than one license type is employed, this new feature will enable you to utilize the flexibility offered by floating licenses to their fullest extent.

Support for A4 Size Paper
Widely used around the world, A4 size paper is now supported in the GENESYS Report Designer. A4 size paper is selected during installation and all reports are adjusted when they run to use that size.

Improved Performance in Simulator and Report Generator
GENESYS 2.0 includes significant performance and scalability improvements to report generation and general improvements to overall application performance.
Schema Migration Wizard

With each release, Vitech continues to enhance the underlying schemas to reflect on lessons learned and best practices in the application of MBSE. In order to simplify the process of migrating your model to the new schema, we have created a Schema Migration Wizard. This wizard will make a copy of your project and perform the migration in the copy, preserving your original project.

Expanding the model

Leveraging lessons learned and best practices, the v20 versions of our base and DoDAF schemas include extensions and revisions in the area of:

- New classes and relations to support States and Modes.
- Edits to the Risk class to support risk management for program managers.
- Added Classification class, classifies relationship, and related attributes to DoDAF schema to better support DoD classification.

A complete list of all schema changes is included at the end of these release notes.

Note: You do not have to migrate your schemas in order to use GENESYS 2.0. You should assess your project needs, your project lifecycle, and the changes present in the v20 schemas to determine if, and when, to migrate your schemas. Projects nearing a major milestone or approaching conclusion should strongly consider remaining with their current schema. We recommend that others consider moving to the v20 schemas to take advantage of the latest improvements in the MBSE language.

Comprehensive list of GENESYS 2.0 Schema Changes

General Changes Included in All Schemas (Including Legacy Schemas)

- New classes
  - Event
  - Transition
  - Rationale for additions: Adding these classes allows GENESYS to render the State Transition Diagram from the database.

- New relation pairs
  - contained by / contains
  - elicited by / elicits
  - encompassed by / encompasses
  - entered by / enters
  - exited by / exits
  - incorporated by / incorporates
  - reflected in / reflects

- Class changes
  - ChangeRequestPackage
    - Added attribute action
GENESYS 2.0 New Features Guide

- Added target class Document to generated by relationship
- Added target class Section to impacts relationship
  - **Document**
    - Added attribute documentOverview
    - Added attribute identification
    - Added attribute revisionNumber
    - Added attribute scope
  - **Function**
    - Added incorporated by relationship with target class State
    - Added reflected in relationship with target class Function
    - Added reflects relationship with target class Function
    - Add service relationship with target class Transition
  - **Requirement**
    - Added elicited by relationship with target class UseCase
    - Rationale for addition: use cases are used in the elicitation process to create requirements. This relation will allow traceability of requirements back to the use case.
  - **State/Mode**
    - Added puid attribute
    - Added entered by relationship with target class Transition
    - Added exhibited by relationship with ImplementationUnit
    - Added exited by relationship with target class Transition
    - Added incorporates relationship with target class Function
  - **UseCase**
    - Added puid attribute
    - Added elicits relationship with target class Requirement
  - **VerificationRequirement**
    - Added puid attribute

**Systems Engineering Schema Changes (Also Reflected in the DoDAF 2.0 Schema)**

- **New class**
  - **Mode**
- **Relationship changes**
  - **built from / built in** relationship attribute wholeMultiplicity
    - Changed type to enumeration of integer
    - Rationale for change: Either there is whole multiplicity or there is not. Changing to a 0 or 1 allows GENESYS to diagrammatically show the whole multiplicity value in the diagrams.
- **Class changes**
  - **Component**
    - Move joined to relationship from parent to Component
    - Rationale for move: Not all ImplementationUnit subclasses should have a relationship with Interfaces.
  - **ConnectingUnit**
    - Added includes relationship
  - **Function**
    - Removed defines relationship
    - Rationale for removal: Changed the relation meaning between Function and State. Added the capability to show behaviors of the State.
  - **ImplementationUnit**
    - Removed joined to relationship (to a subclass)
- Added `contains` relationship with target class **Mode**
- Added `exhibits` relationship with target class **State** (moved from a subclass)

  **Interface**
  - Added `includes` relationship with target class **Interface**
  - Added `included in` relationship with target class **Interface**
  - Set Parent-Child relationship to `includes`
  - Rationale for changes: Decomposition of **Interfaces** was added to support parallel decomposition within the model. Until now relationships were broken at the higher level in order to connect them at the lower level. By decomposing Interfaces throughout the model the model stays consistent at each level.

  **Link**
  - Added `includes` relationship with target class **Link**
  - Added `included in` relationship with target class **Link**
  - Set Parent-Child relationship to `includes`
  - Rationale for changes: Decomposition of **Links** was added to support parallel decomposition within the model. Until now relationships were broken at the higher level in order to connect them at the lower level. By decomposing **Links** throughout the model the model stays consistent at each level.

  **State/Mode**
  - Renamed to **State**
  - Changed abbreviation to ST
  - Removed type attribute
  - Removed `defined by` relationship
  - Remove `includes` relationship
  - Removed `included in` relationship
  - Added `decomposed` relationship with target class **State**
  - Added `decomposed by` relationship with target class **State**
  - Added `encompassed by` relationship with target class **Mode**

  **Risk**
  - Renamed consequences attribute to significance
  - Changed impact attribute
    - Renamed to consequence
    - Change possible value “Low” to “1 – Minimal”
    - Change possible value “Medium” to “3 – Moderate”
    - Change possible value “High” to “4 – Severe”
    - Added possible value “2 – Minor”
    - Added possible value “4 – Significant”
  - Changed likelihood attribute possible values
    - Changed “Low” to “1 – Not Likely”
    - Changed “Medium” to “3 – Likely”
    - Changed “High” to “5 – Near Certainty”
    - Added “2 – Low Likelihood”
    - Added “4 – High Likelihood”
  - Added mitigationCost attribute
  - Changed riskFactor attribute
    - Renamed to riskRating
    - Changed type to an enumeration
  - Added possible values to type attribute
    - Logistics
    - Requirements
    - Test and Evaluation

- Facility changes
GENESYS 2.0 New Features Guide

- Document Management
  - Added Event
  - Added Mode
  - Added Transition
- Essentials
  - Added State
- Systems Engineering
  - Added Event
  - Added Mode
  - Added Transition
- Verification Facility
  - Added Event
  - Added Mode
  - Added Transition

DoDAF 2.0 Schema Changes

- New Classes
  - Classification
- New Relation pairs
  - classified by / classifies
- Class changes
  - Capability
    - Added puid attribute
  - ConnectingUnit
    - Added accessControl attribute
    - Added availability attribute
    - Added confidentiality attribute
    - Added integrity attribute
    - Added nonrepudiationConsumer attribute
    - Added nonrepudiationProducer attribute
    - Added includes relationship
    - Set Parent-Child relationship to includes
    - Rationale for additions: The Architecture schema is used primarily for DoD projects which make use of some standard policies which need to be captured for document generation.
  - Event
    - Added target class OperationalInformation to responsible for relationship
  - Needline
    - Added includes relationship with target class Needline
    - Added included in relationship with target class Needline
    - Set parent-child relationship to includes
    - Rationale for changes: Decomposition of Needlines was added to support parallel decomposition within the model. Until now relations were broken at the higher level in order to connect them at the lower level. By decomposing Needlines throughout the model the model stays consistent at each level.
  - OperationalActivity
    - Removed defines relationship with target class State
    - Added incorporated by relationship with target class State
    - Added target class Transition to the services relationship
  - OperationalInformation
    - Added criticality attribute
• Added target class **Event** to the *assigned to* relationship
  o **Requirement**
    • Added objective attribute
    • Added performanceParamterType attribute
  o **State**
    • Added target class **OperationalActivity** to the *incorporates* relationship
  o **Transition**
    • Added target class **OperationalActivity** to the *serviced by* relationship
  • Facility Changes
    o **Architecture Facility**
      • Added **Classification**
      • Added **Event**
      • Added **Mode**
      • Added **Transition**