Mission Impact Situation Awareness Modeling and Visualization

Dagger is a modeling and visualization tool suite that shows how system failures impact mission status. Updated with manual or real-time status, Dagger is used for mission/system planning, situational awareness during mission execution, and course-of-action analysis.

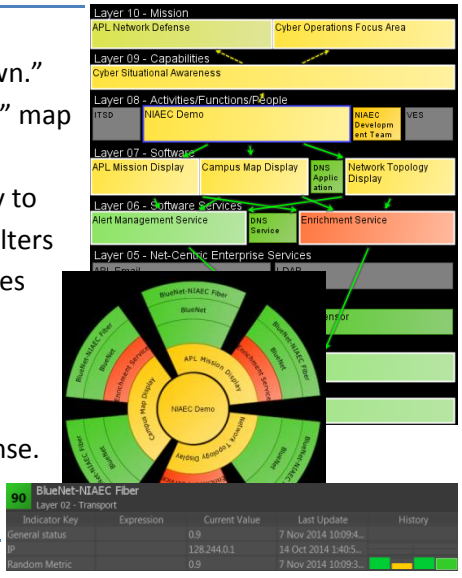
What problem does it solve?

In environments with *large volumes* of mission or system status information, Dagger's dependency model can put that information in context, addressing questions such as:

- Given the current state of the system, can the mission succeed?
- Is there any part of the system that will significantly impact mission success?
- What is the impact of loss or degradation of component X on the mission?
- Is there system information missing that is needed to provide the answers to these questions?

In *dynamic* environments, Dagger's model editing capabilities make it easy to build, maintain, and visualize components and dependencies in that environment and evaluate the impact of changes.

Standard Visualizations



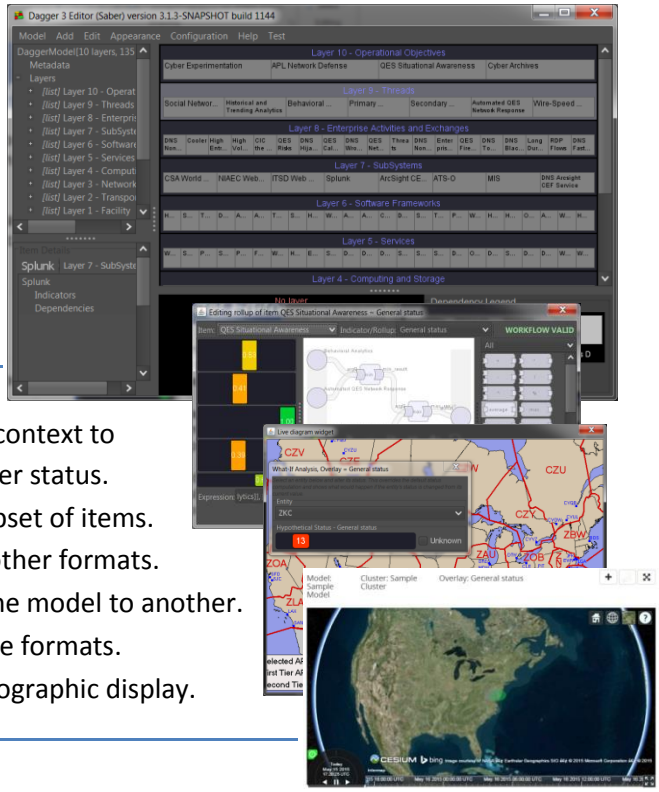
- *Color* represents real-time or historical status. Gray indicates "unknown." Different notions of status such as "availability" and "security posture" map to different color *overlays*.
- *Layer visualization* shows the big picture at a glance and makes it easy to quickly find items of interest. Dependencies are shown on demand. Filters can be applied to large models to focus the view on what the user cares about most.
- *Hierarchy visualization* shows critical dependency paths from a given item, e.g. to find the cause of mission/system degradation.
- *Item details view* shows more information to support action or response. Including historic observational data.

Analysis

- Interactively create and visualize *hypothetical scenarios* such as "What happens if this system, server, or facility goes down?" with one or more items.
- *Sensitivity analysis* computation shows the most important items supporting a target mission, helping to identify critical infrastructure and key terrain.

User Interface

- Web client provides customizable dashboard for viewing models, and can run inside OWF/OWP.
- Java client provides full-featured client and editor.
- The *model editor* provides a drag-and-drop interface for creating and customizing dependency models.
- Built-in templates and example models provided for specific domains such as information systems.



Widgets and Plugins

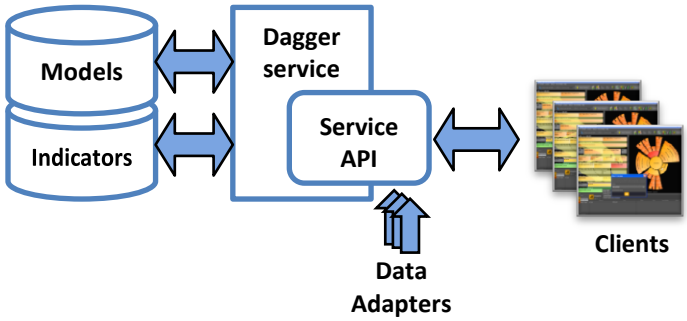
- *Live diagram widgets* adds geographic or domain-specific context to model items, e.g. to draw items on a map colored by Dagger status.
- *Live table widgets* displays live observational data for a subset of items.
- *Model importers* create models from CSV, GraphML, and other formats.
- *Publisher* plugin publishes computed status values from one model to another.
- *Reporting* plugins export status or observations in alternate formats.
- *Geospatial* widget (web client only) visualizes status on geographic display.

Service & Storage

- Data stores for observational data (indicators) and Dagger models.
- RESTful API for adding and querying observational data, publishing and retrieving Dagger models, and getting computed model status.
- Supports linking multiple clients to share real-time status or hypothetical/historical scenarios across multiple views.

Deployment

A typical Dagger deployment requires the following steps:



1. Install the Dagger client and service.
2. Identify the mission to model. Build the model with supporting components and dependencies.
3. Identify potential data sources. Either configure existing data adapters or develop new data adapters to map the data to observations in the model. Deploy the adapters.
4. Iterate to refine the model and data feeds.

Contact

Technical Lead Jackie Soenneker
LAVA Technical Business Lead Keith Wichmann
dagger@jhuapl.edu