AFRL-RI-RS-TR-2013-029

# GRAPH AND NETWORK FOR MODEL ELICITATION (GNOME PHASE II)

CUBRC

*FEBRUARY 2013*

FINAL TECHNICAL REPORT

**AIR FORCE RESEARCH LABORATORY**
**INFORMATION DIRECTORATE**

**■ AIR FORCE MATERIEL COMMAND ■ UNITED STATES AIR FORCE ■ ROME, NY 13441**

# NOTICE AND SIGNATURE PAGE

| REPORT DOCUMENTATION PAGE | *Form Approved* **OMB No. 0704-0188** |
|---|---|

| | | |
|---|---|---|
| **1. REPORT DATE** *(DD-MM-YYYY)* FEBRUARY 2013 | **2. REPORT TYPE** FINAL TECHNICAL REPORT | **3. DATES COVERED** *(From - To)* JAN 2011 – OCT 2012 |

| | |
|---|---|
| **4. TITLE AND SUBTITLE** GRAPH AND NETWORK OBJECTS FOR MODEL ELICITATION (GNOME PHASE II) | **5a. CONTRACT NUMBER** FA8750-11-C-0021 |
| | **5b. GRANT NUMBER** N/A |
| | **5c. PROGRAM ELEMENT NUMBER** 63788F |
| **6. AUTHOR(S)** Kevin Costantini and Jeff Spaulding | **5d. PROJECT NUMBER** E3MK |
| | **5e. TASK NUMBER** 00 |
| | **5f. WORK UNIT NUMBER** 01 |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** CUBRC 4455 Genesee St. Buffalo, NY 14225 | **8. PERFORMING ORGANIZATION REPORT NUMBER** N/A |
| **9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)** Air Force Research Laboratory/RIED 525 Brooks Road Rome NY 13441-4505 | **10. SPONSOR/MONITOR'S ACRONYM(S)** N/A |
| | **11. SPONSORING/MONITORING AGENCY REPORT NUMBER** AFRL-RI-RS-TR-2013-029 |

**12. DISTRIBUTION AVAILABILITY STATEMENT**
Approved for Public Release; Distribution Unlimited. PA# 88ABW-2013-0593
Date Cleared: 11 February 2013

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**
GNOME II continues the model elicitation work of GNOME and implements the established methods and exploration techniques into the current visualization environment for NOEM. From a modeling standpoint, we extract knowledge from the relationships and structure of the Nation Building simulation model to answer the questions: what does an input parameter affect?, how much of an effect exists?, and how can I achieve a certain state? From a software standpoint, we provide additional interface and visualization components to aid the user in navigating a nation-building model, understanding the key relationships, and retrieve suggested inputs and outputs of interest based on either what they can modify or what they are trying to affect within the model.

**15. SUBJECT TERMS**
Nation-building, Model Elicitation, Graph Theory

| **16. SECURITY CLASSIFICATION OF:** | | | **17. LIMITATION OF ABSTRACT** | **18. NUMBER OF PAGES** | **19a. NAME OF RESPONSIBLE PERSON** **PATRICK K. McCABE** |
|---|---|---|---|---|---|
| **a. REPORT** U | **b. ABSTRACT** U | **c. THIS PAGE** U | SAR | 44 | **19b. TELEPONE NUMBER (*Include area code)*** **(315) 330-3197** |

## Table of Contents

## List of Figures

## List of Tables

## 1 Summary

This report encompasses the analysis concepts and software implementations through GNOME (Graph and Network Objects for Model Elicitation) phase II. GNOME was developed to aid Nation Building decision makers in answering three very important questions: What affects what, how much of an effect exists, and how can a certain state be achieved. Over the course of this phase of development, GNOME has been integrated within the NOEM (National Operational Environment Model) and acts as a middleware between the interactions of the simulation model and the decision maker.

A key component of GNOME is the Model Exploration capability. GNOME extracts the structure of the underlying model into a neutral graphical form. The decision maker is then able to interact directly with this graphical representation of the model through a user interface component called the Model Explorer. The Model Explorer provides a hierarchical view of the model where a user may identify high-level relationships between modules of the model, or delve into further detail and look at how sub-systems and specific inputs and outputs are related.

Another capability involves the analytical tools to compare inputs and outputs within the models. An experimental sampling procedure is implemented to identify main effects between inputs and outputs in the model. A comparative weighting algorithm is also implemented and aids the user in comparing inputs of interest for an output. Likewise, a similar algorithm compares outputs of interest for an input. Visualization tools are provided to identify, compare, and select these inputs and outputs of interest for use in experimentation within NOEM.

Through the achievements of GNOME and the integration work in phase II, decision makers are provided with the tools to address the three important questions addressed earlier. They can identify what affects what at varying levels of detail within the model. They can utilize experimentation and comparative weighting to identify how much of an effect exists for particular inputs or particular outputs. Lastly, they can combine the main effects of the sampling results (in a linear combination) to identify how a particular state can be achieved for the outputs of interest.

## 2 Introduction

Under the Full Spectrum Targeting Program, the Air Force Research Laboratory at Rome, NY is leading a project on a holistic modeling and simulation environment for Nation Building of failed states or for Stabilization and Reconstruction operations. Nation Building is a comprehensive effort aimed at engineering major social, political and economic reconstruction in a failed nation, and assisting the new government to establish all civilian functions to normalcy. Nation Building decisions have moved from a subjective to a more quantitatively driven process. Modeling and large-scale simulation are the prevalent approaches in practice today. The initial GNOME (Graph and Network Objects for Model Elicitation) was developed to overcome the complexity of these simulation models by using a set of graph and network-based methods. GNOME II continues this work and implements the established methods and exploration techniques into the current visualization environment for NOEM. Specifically, we extract knowledge from the relationships and structure of the Nation Building simulation model to answer the questions below.

- What affects what?
  - What does an input parameter affect?
  - What inputs affect an output of interest?
- How much of an effect exists?
  - Of the determined effects, which is the strongest?
  - How do a set of inputs compare in their effect on something?
- How can I achieve a certain state?
  - Is the state achievable?
  - How long will it take to achieve?
  - What is the optimal way (when given constraints)?

This report discusses the software-related achievements and modeling implementations made under GNOME II.

# 3 Methods, Assumptions, and Procedures

The intent of the GNOME data structure is to store the NOEM modules in a clearly defined, neutral graphical form that can be used separate from the simulation. This structure allows for custom analysis tools and also provides useful metrics to other applications where analysis can be performed. This section of the document identifies the architecture and format of the GNOME data structure along with the approach taken to transform and visualize NOEM models in this graphical structure. The general approach of the GNOME framework is illustrated in *Figure 1*, where GNOME acts as a middleware between the interactions of the simulation model and the applications that a decision maker would interact with.

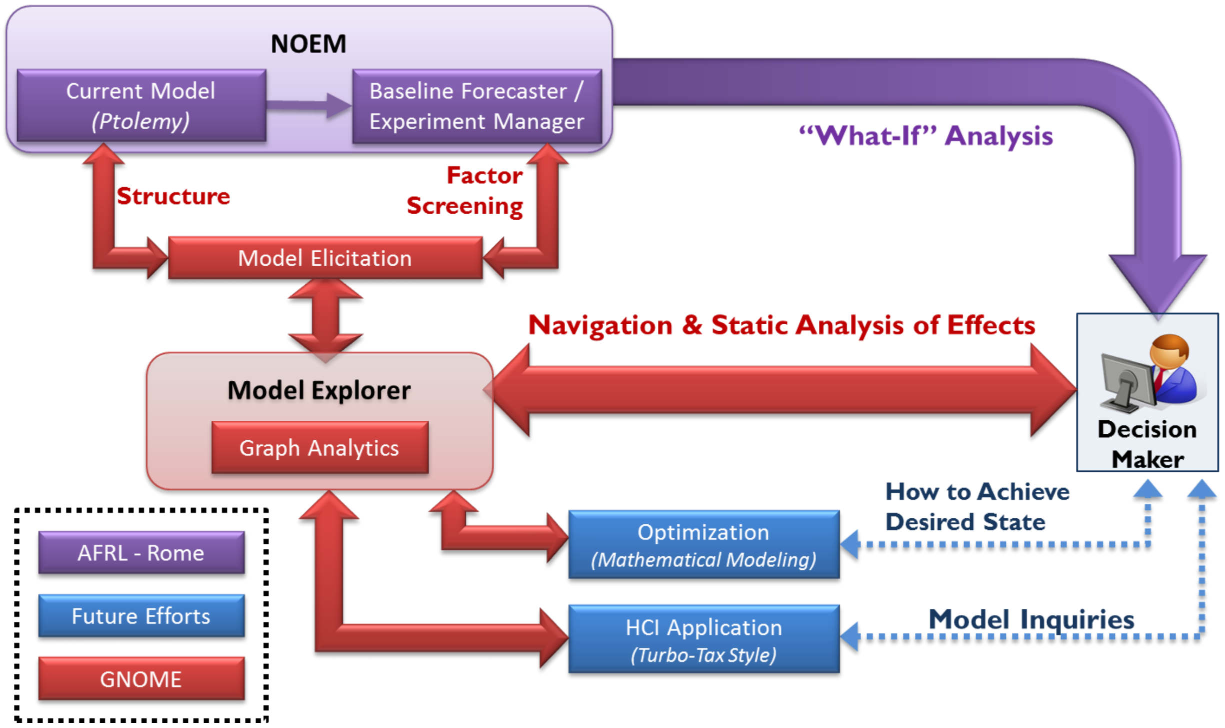


**Figure 1: GNOME Overview**

## 3.1 Framework

The models and model data for NOEM is stored using Ptolemy, which offers a strong actor-oriented simulation environment for running the model. Each actor in the model has some form of data flowing through it, and the actor performs an action on the data. Overall, the structure of actors within a particular Ptolemy model allows the model to be treated as both a graph and a simulation. The data structure developed through GNOME represents the graphical aspects of the Ptolemy model in a more interoperable manner with additional analysis capabilities. The model structure used for GNOME II borrows most of the previous functionality for representing and running algorithms on the underlying JUNG graph. Some modifications have been made to support the new server-side approach.

### 3.1.1 JUNG

JUNG is a "Java Universal Network and Graph" package that includes class libraries that assist in representing, storing, and visualizing graphs in the Java environment. This package allows for GNOME to create a custom-defined graph type (with custom vertex, edge, and attribute types) as a simplification of the Ptolemy models. Furthermore, JUNG provides fairly generic implementations of numerous graph-theory based algorithms.

### 3.1.2 Basic conversion process

A Ptolemy model is initialized and a JUNG-based graph representation is created based on its network properties. The JUNG graph is stored to a database in two forms (one for algorithmic use and one for visualization), and loaded by services needing to interact with such models. This intermediate step removes the need to pre-load and initialize Ptolemy in order to interact with the model (typically the most time-consuming step). This allows for the graph structure to be loaded completely independent of Ptolemy. *Figure 2* displays the interactions of this approach, with GNOME's contributions in red.

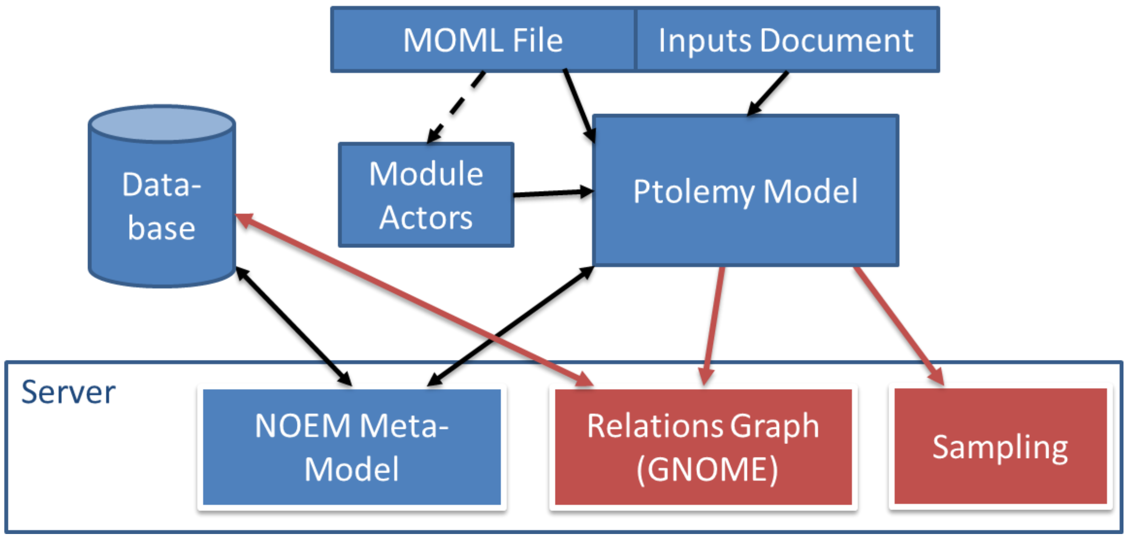


**Figure 2: Model Loading Process**

In relation to the newer, more java-based models being developed and used for NOEM, we have adjusted the graph and analysis components to interact properly with the approach. An inputs document is combined with the MOML file, and all input adjustments are done through this document.

In terms of model storage, the server-side graph for GNOME includes session management, where certain user interactions and settings are tied to a specific session and graph data is unloaded as needed when sessions are no longer active. Additionally, routines are available to identify input parameters held internally by custom NOEM modules and incorporate those into the GNOME graph and hierarchy. This corresponds with the newer, more java-based models being developed.

### 3.1.3 Graphical analysis

The key purpose of the elicitation process is giving a user the ability to determine meaningful relationships and conclusions based on the structure of the model without the requirement of simulating through all possible configurations and being subject to randomness. These relationships are currently provided by the analysis capabilities of GNOME; which includes model exploration, input & output connectivity, and weighting & ranking of linked components.

#### *3.1.3.1 Model Exploration*

Model exploration refers to the process of identifying interesting relationships between components of the graph at both high levels and low levels. The analytical components for model exploration include the expand and collapse features of a nested graph, the hierarchical and categorical organization of the model, the reachability algorithms, and the abstraction of components of interest to a higher level. Section 2.3.2 further discusses these capabilities as well as the GWT visualization components corresponding to GNOME's Model Explorer.

#### *3.1.3.2 Input-Output Connectivity*

One of the drawbacks of using model exploration techniques with the whole elicited model is the amount of clutter that quickly accumulates. More often than not, users will care only to focus on model inputs and output. Identifying the connectivity of model inputs and outputs involves narrowing the scope of model exploration capabilities to focus simply on relating input parameters and published outputs in the elicited model. This approach primarily uses the hierarchical and categorical organization of the model in combination with reachability algorithms to establish bipartite graphs indicating how a set of inputs relates to a set of outputs (and vice versa).

Several interface components rely on this filtered approach to connectivity, including the query and associate views, discussed in sections 3.3.3 and 3.3.4, respectively.

#### *3.1.3.3 Weighting / Ranking*

Weighting goes a step beyond connectivity to give a normalized weight to each input-output or output-input relationship. Weights can be provided either through an algorithm performed with the GNOME graph or through main effects calculated through experimental sampling with the model (discussed in section 3.2).

A percent impact algorithm was first implemented (and tested with the NOEM thick-client) to indicate the 'level-of-effect' of inputs on distant outputs. The approach scales together the arc weights pertaining to the input-output effect ratio for individual actors along all paths from the input to the output. When a cycle is involved, the process works iteratively until all paths have converged to a value based on a desired precision. *Figure 3* displays the simple visualization used for this approach using a small sample model, where a 100% increase in the selected input value will likely cause the indicated % increase or decrease in value for the displayed outputs.

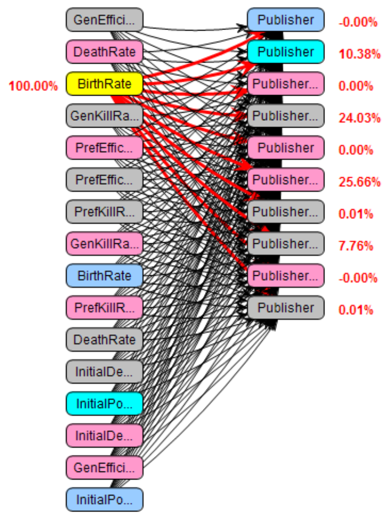


**Figure 3: Percent Impact Algorithm**

We determined that representing the likely impact as an individual value may not be as useful as indicating a range in the which the level of impact will likely fall within. To handle this, we used the experimental sampling data to calculate and include a confidence interval for each effect ratio arc weight. This is discussed further in section 3.4

The algorithm has been tested with nation-building models to identify strengths and weaknesses of the approach. The algorithm is significantly faster than the weighted random walk approach, taking roughly two seconds for a typical comparison that would take 30 seconds using weighted random walk; however, outputs in different modules and regions tend to receive very little impact even when modifying an input by 100%. This result is partially intrinsic to the large size of these models and wide array of inputs and variables used, but there are particular instances where more of an effect should be exhibited. Clearly, the assumption of linearity plays a role in this, and using piecewise or higher-order weighting would improve the strength/accuracy of the weights used.

## 3.2 Experimental Sampling

Experimental sampling involves comparing high and low settings of an input vs. the main effect on observed outputs. We worked with two different approaches to the process in order to devise a weighting system for inputs and outputs.

### 3.2.1 Initial Approach

The starting approach was through factor screening. In general, the problem of factor screening in simulation is the search through all of a model's potentially important input factors, k, for the most important subset of those factors, g, where |g| << |k| in most applications (Montgomery, 1979)(B. Bettonvil, 1996). Typically, factor screening in a simulation such as NOEM is carried out through the use of designed statistical experiments. A good overview of many of these techniques is given by Kleijnen (1975).

Through the initial phase of GNOME, we developed a technique that separated out the actors (or sub-components) of Ptolemy models and performed sampling experiments on each actor. This involved running low and high values for each actor input and observing the response deviation as the main effect (or weight) for each output. This was further modified to calculate and store weights as a ratio of input-output effect. For example, if an X percent increase of an input creates a Y percent increase in an output, the weight is stored as the ratio X/Y. The direction of effect is also indicated (positive vs. negative). Further development in regards to this approach involves a 'level-of-effect' algorithm that scales the arc weights together to determine the effect of an input on distant outputs.

With GNOME II, the analysis for certain actors was simplified due to either a large set of input/output ports or more complex functionality than simple data-flow. Experimentation was modified for particular actors, such as the Vector Disassembler and Sample Delay, in order to account for needed information stored within the actors or ports involved. Basic ANNOVA (analysis of variance) calculations were utilized during the experimental sampling phase. These methods efficiently calculate and store the main effects and error associated with the weighting process. The confidence interval calculations were also updated by using the $MS_E$ (mean-square error) available through this implementation. Through our analysis, we noticed that no error is present for most actors without interaction effects. Typically, such actors are exact calculations and the output variation can be perfectly predicted per input variation.

An $n^k$ experiment configuration was added that allows users to define how many levels (n) to test for each factor. This builds upon the typical low-high approach of the $2^k$ configuration and splits the difference appropriately to form the remaining levels. The results of experimentation can also be saved in an XML file to aid in further analysis of data. This structure stores an output data point along with each level the available factors were set at. An example of the data structure for experimentation with an Expression actor is shown in *Figure 4*; where X0 and X1 are factors and Y is the output.

```
X0=". Baghdad.CrimeModule.Population.ageGroup_25_49.longTermIncarceration.SampleDelay.output"
X1=". Baghdad.CrimeModule.Population.ageGroup_25_49.longTermIncarceration.meanIncarcerationTime"
output=". Baghdad.CrimeModule.Population.ageGroup_25_49.longTermIncarceration.Expression.output">
    <Data X0="4350.0" X1="945.0" Y="4.6031746031746"/>
    <Data X0="5075.0" X1="945.0" Y="5.3703703703704"/>
    <Data X0="5800.0" X1="945.0" Y="6.1375661375661"/>
    <Data X0="6525.0" X1="945.0" Y="6.9047619047619"/>
    <Data X0="7250.0" X1="945.0" Y="7.6719576719577"/>
    <Data X0="4350.0" X1="1102.5" Y="3.9455782312925"/>
    <Data X0="5075.0" X1="1102.5" Y="4.6031746031746"/>
    <Data X0="5800.0" X1="1102.5" Y="5.2607709750567"/>
    <Data X0="6525.0" X1="1102.5" Y="5.9183673469388"/>
    <Data X0="7250.0" X1="1102.5" Y="6.5759637188209"/>
    <Data X0="4350.0" X1="1260.0" Y="3.452380952381"/>
    <Data X0="5075.0" X1="1260.0" Y="4.0277777777778"/>
    <Data X0="5800.0" X1="1260.0" Y="4.6031746031746"/>
    <Data X0="6525.0" X1="1260.0" Y="5.1785714285714"/>
    <Data X0="7250.0" X1="1260.0" Y="5.7539682539683"/>
    <Data X0="4350.0" X1="1417.5" Y="3.0687830687831"/>
    <Data X0="5075.0" X1="1417.5" Y="3.5802469135802"/>
    <Data X0="5800.0" X1="1417.5" Y="4.0917107583774"/>
    <Data X0="6525.0" X1="1417.5" Y="4.6031746031746"/>
    <Data X0="7250.0" X1="1417.5" Y="5.1146384479718"/>
    <Data X0="4350.0" X1="1575.0" Y="2.7619047619048"/>
    <Data X0="5075.0" X1="1575.0" Y="3.2222222222222"/>
    <Data X0="5800.0" X1="1575.0" Y="3.6825396825397"/>
    <Data X0="6525.0" X1="1575.0" Y="4.1428571428571"/>
    <Data X0="7250.0" X1="1575.0" Y="4.6031746031746"/>
```

**Figure 4: Experimental Sampling Output**

An optional piecewise-linear weighting algorithm was implemented within the experimental sampling process to more adequately represent more complex simulation actors. This algorithm is based on the findings from Goldsmith (2010). Two parameters were added to the sampling approach:

- Max number of segments
- Minimum R-squared value.

These control how many pieces each function can be separated into and what the stopping condition is for finding a good fit. The resulting weight includes four bits of data for each segment to represent the input-output relationship:

- Slope (the main-effect calculated by regression analysis)
- Intercept (the base value independent of the input)
- Starting input-value for segment
- Ending input-value for segment

While these experimentation approaches work well for most of the model components, there are still several complex, custom actors that cannot be properly modeled based on direct inputs alone. In many cases, these interact with other custom actors in a manner that varies depending on the source module, and separating the actors for low-level experimentation has not been possible. For these reasons a newer simplified experimentation approach was adopted for the latter portion of GNOME II.

### 3.2.2 New Approach

The experimentation approach was adjusted to move much of the experimenting, weighting and analysis to server-side computations that are done upon importing and initializing a model (prior to loading the model in the web-client). The server-side service can run and generate data asynchronously, allowing a cluster of servers to run the sampling. Also, a background process runs upon detecting a new model or changes to the model.

The experimentation has been streamlined to focus on providing data for the main effects of each input in the model (for each affected output). Additionally, the sampling output was adjusted to utilize area calculations rather than the final output points. The area between the policy response and the baseline response is recorded for each output. This involves Riemann sums to estimate the area held under 10-iteration intervals in the response. Options have been added to use a subset of the available inputs if desired, based on the "category" tags in the NOEM models (Policy, Environment, etc.). For typical nation-building models, the simulations require roughly 1 minute per input to test three different levels, and this scales linearly with the number of inputs in the model. This new Experimental Sampling service is used by many components to run specific input configurations and observe the output values. *Figure 5* how the Experimental Sampling (in blue) and Weight Normalization (in red) are used to provide weights for the models.

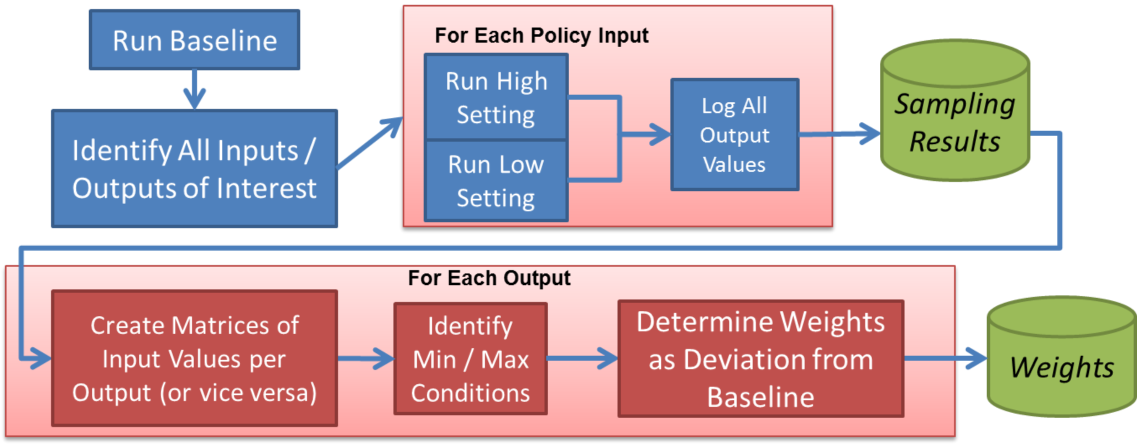

Run Baseline
Identify All Inputs / Outputs of Interest
For Each Policy Input
Run High Setting
Run Low Setting
Log All Output Values
Sampling Results
For Each Output
Create Matrices of Input Values per Output (or vice versa)
Identify Min / Max Conditions
Determine Weights as Deviation from Baseline
Weights

**Figure 5: Experimental Sampling and Weighting**

### 3.2.3 Weighting Inputs and Outputs

The weighting routine integrates with the service to pull in the generated data and push weights/scores to a database for the other analysis components to access through NOEM. This is illustrated in the bottom portion of *Figure 5*. For each output, two separable linear programs are solved to give the MAX weights ($W^{MAX}$) and the MIN weights ($W^{MIN}$) for each input to be ranked. The normalization occurs for both input-output weighting (used in the "Associate" tab of the Output Set dialog) and output-input weighting, although there are slight differences in how the weights are normalized. So far, we have generated sampling data and normalized weights for comparing all model outputs with "Policy" inputs for the DR Congo and Venezuela models.

## 3.3 GNOME UI Components for NOEM Web Client

For GNOME II, HCI components utilize the newly developed NOEM web client and GWT software capabilities. As GWT is still java-based, the model interaction and graph-theory algorithms utilized in GNOME were compatible. The majority of work in integrating the GNOME software involves the Model Explorer visualization components. The new HCI components were to be loosely based on the GNOME RCP plug-in implementation, with several components redesigned to take into consideration the suggestions and feedback from other NOEM developers.

### 3.3.1 Overview

The GNOME UI project includes a Model Explorer editor tab, a Query editor tab, hierarchy display components, and services that retrieve graph components and provide algorithm results for the web-client. Additionally, some modifications were made to NOEM's UI components to use GNOME in providing input and output suggestions in the Input Set Dialog and Output Set Dialog. A separate sampling service is also linked with GNOME to allow the user to setup experimental sampling to run server-side. Some modifications were made to JUNG's structure to support the use of the GWT-compliant Raphael visualization library.

### 3.3.2 Model Explorer

Since the previous version of GNOME was developed as an Eclipse RCP plug-in, it allowed CUBRC to develop the Model Explorer separately without modifying the base framework of NOEM. When NOEM transitioned to a thin-client architecture, there no longer was support for plug-ins. As a result, the requirements for developing the new thin-client Model Explorer included software that would be compatible with GWT (Google Web Toolkit) and support for standard web browsers (e.g. Internet Explorer and Mozilla Firefox).

#### 3.3.2.1 Web Visualization Software

In order to implement the graph visualization component of the Model Explorer to run on a web browser, we looked at several software packages:

- *mxGraph*
  - A fully-featured JavaScript library that supports diagrams and interactive drawings.
  - Proprietary software; requires a commercial license.
- *Protovis*
  - Free and open-source, provided by the Stanford Visualization Group.
  - Uses JavaScript and SVG (Scalable Vector Graphics) for web-native visualizations.
  - No longer under active development and Internet Explorer 8 does not support SVG.
- *JavaScript InfoVis Toolkit*
  - Free and open-source under the BSD License.
  - Features advanced graph layouts, but no support for Internet Explorer 8 and below since it uses HTML5.
- *Flare*
  - Free and open-source under the BSD License.
  - Uses Adobe's ActionScript library for creating visualizations that only runs in the Adobe Flash Player.

As we explored the available software packages that specialize in graph visualizations for web browsers, we quickly learned that most of them use SVG (Scalable Vector Graphics) or HTML5—both of which are not supported by Internet Explorer 8 and below. Since Internet Explorer is the most widely-used web browser, we continued to find a solution that would satisfy this requirement.

Another option we considered was using *Java Applets*, which works with most standard web browsers and offered performance similar to that of native-installed software. However, a disadvantage of using *Java Applets* is that they require the web browser to have the Java plug-in and a specific Java Runtime Environment (JRE). Potential users might have to wait for a large JRE download if the Applet is incompatible--which could cause problems if users do not have administrative permissions on their machines to install the JRE.

The Adobe web products such as *Adobe AIR* (Adobe Integrated Runtime) and *Adobe Flash* were another option we considered since they offered cross-platform support and the ability to display graphics. Similar to using *Java Applets*, a potential concern is that users must be able to download the *Adobe Flash Plug-in* for their browser if they have an incompatible version.

The final software package we found was *Raphaël,* an open-source JavaScript library that uses vector graphics such as SVG (Scalable Vector Graphics) and VML (Vector Markup Language). Since *Raphaël* uses both SVG and VML, it supports all popular web browsers such as Mozilla Firefox (version 3.0 and above) and Microsoft Internet Explorer (version 6.0 and above). Since the *Raphaël* library only requires

JavaScript, potential users would not have to download plug-ins for their web browsers or be dependent on 3rd-party software to use the GNOME thin-client.

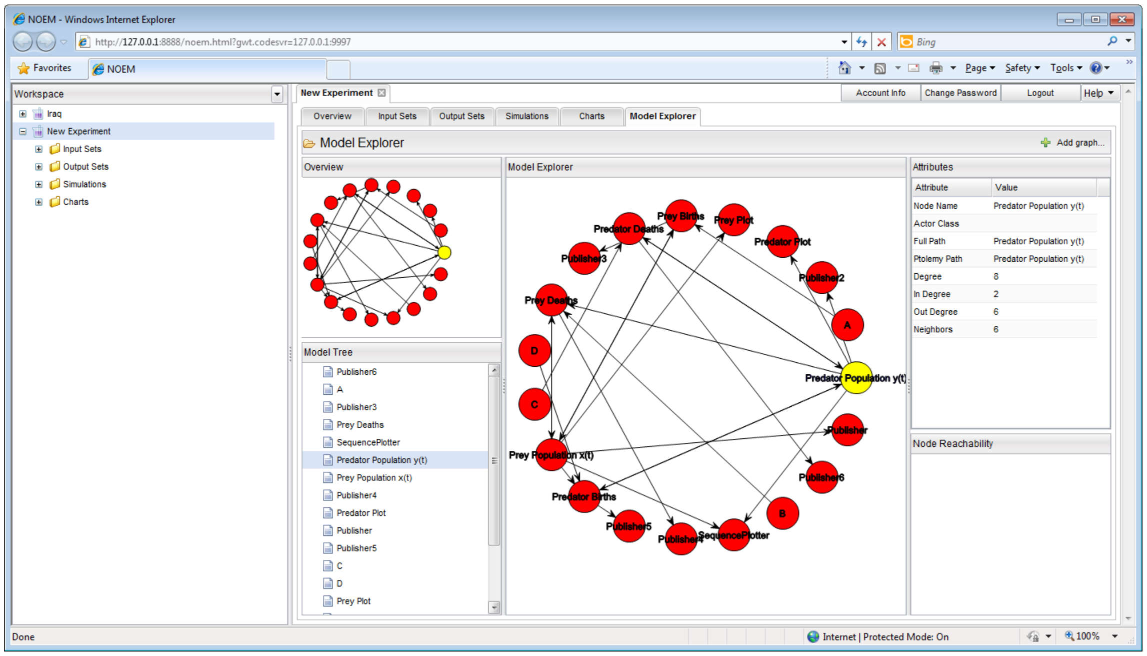
### *3.3.2.2 Implementing JUNG for the web*

The JUNG software package was the primary choice of the thick-client version of GNOME since it included the ability to represent, store and visualize graphs in a Java environment. With built-in layout and analysis algorithms such as graph clustering and metrics for node centrality, we sought to utilize this existing framework to display graphs in a web browser environment.

Much to our advantage, the original authors of JUNG decoupled the visualization component so it could be easily modified without affecting the base components—such as graph data structures and layout algorithms. In fact, the previous thick-client version of GNOME utilized this design feature and employed the use of the SWT (Standard Widget Toolkit) version of JUNG since the original JUNG visualization component used the AWT (Abstract Widget Toolkit) to actually draw the graph visualization to the screen.

The use of the Google Web Toolkit (GWT) by the thin-client version of NOEM allowed us to maintain much of the existing Java code from JUNG. One of the primary features of GWT is the ability for developers to write code in Java and have it automatically translated to JavaScript—which most web browsers can understand and process.

We started with the base components of JUNG (e.g. Graph data structures, Layout Algorithms) and slightly modified the Java source code to be compatible with GWT. Using these newly formed components, we can now represent the basic components of a graph (i.e. nodes and edges) in JavaScript. However, modifying the JUNG visualization component for GWT posed more of a challenge since it relied heavily on Java libraries (e.g. 2D Geometry) that the *GWT Java-to-JavaScript Compiler* did not support. In order to implement the low-level instructions of drawing a graph to the screen, there needs to be source code that describes basic 2D geometry so the program could understand what a circle or line is, for example. Using the OpenJDK (Open Java Development Kit) source code and other available libraries, we were able to implement the required Java classes that would be GWT-compatible and move forward. The final link in the new JUNG visualization component required integrating the *Raphaël* JavaScript library to do the low-level graphics for web browsers. *Figure 6* shows an early build of the thin-client Model Explorer within NOEM that is being rendered by Internet Explorer 8:



Attributes

| Attribute | Value |
|---|---|
| Node Name | Predator Population y(t) |
| Actor Class | |
| Full Path | Predator Population y(t) |
| Ptolemy Path | Predator Population y(t) |
| Degree | 8 |
| In Degree | 2 |
| Out Degree | 6 |
| Neighbors | 6 |

**Figure 6: Early build of the thin-client Model Explorer under Internet Explorer 8**

### *3.3.2.3 Representing Hierarchical Graphs*

While JUNG includes useful features like data structures to represent graphs and sophisticated layout algorithms, one drawback is the ability to represent hierarchical graphs with composite nodes—which are nodes that contain other nodes. The current functionality in JUNG to expand composite nodes is to remove the composite node from the view and replace it with its child nodes. *Figure 7* demonstrates how JUNG expands composite nodes, where the yellow node labeled "Baghdad" on the left was selected for expansion. The graph view on the right shows the result of expanding the "Baghdad" node into its several child nodes that are colored green. If a user were to continue expanding composite nodes, the visualization can become quickly cluttered if you are working with large models whose graph representation can contain hundreds or thousands of nodes and edges.

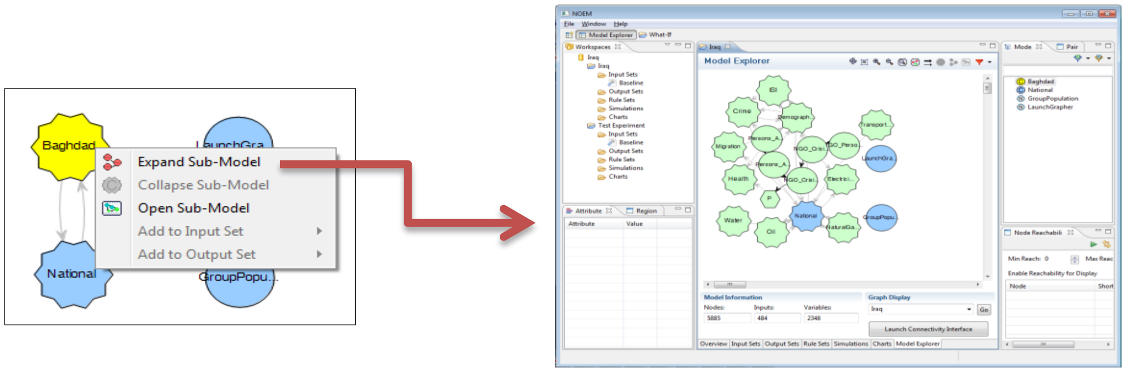
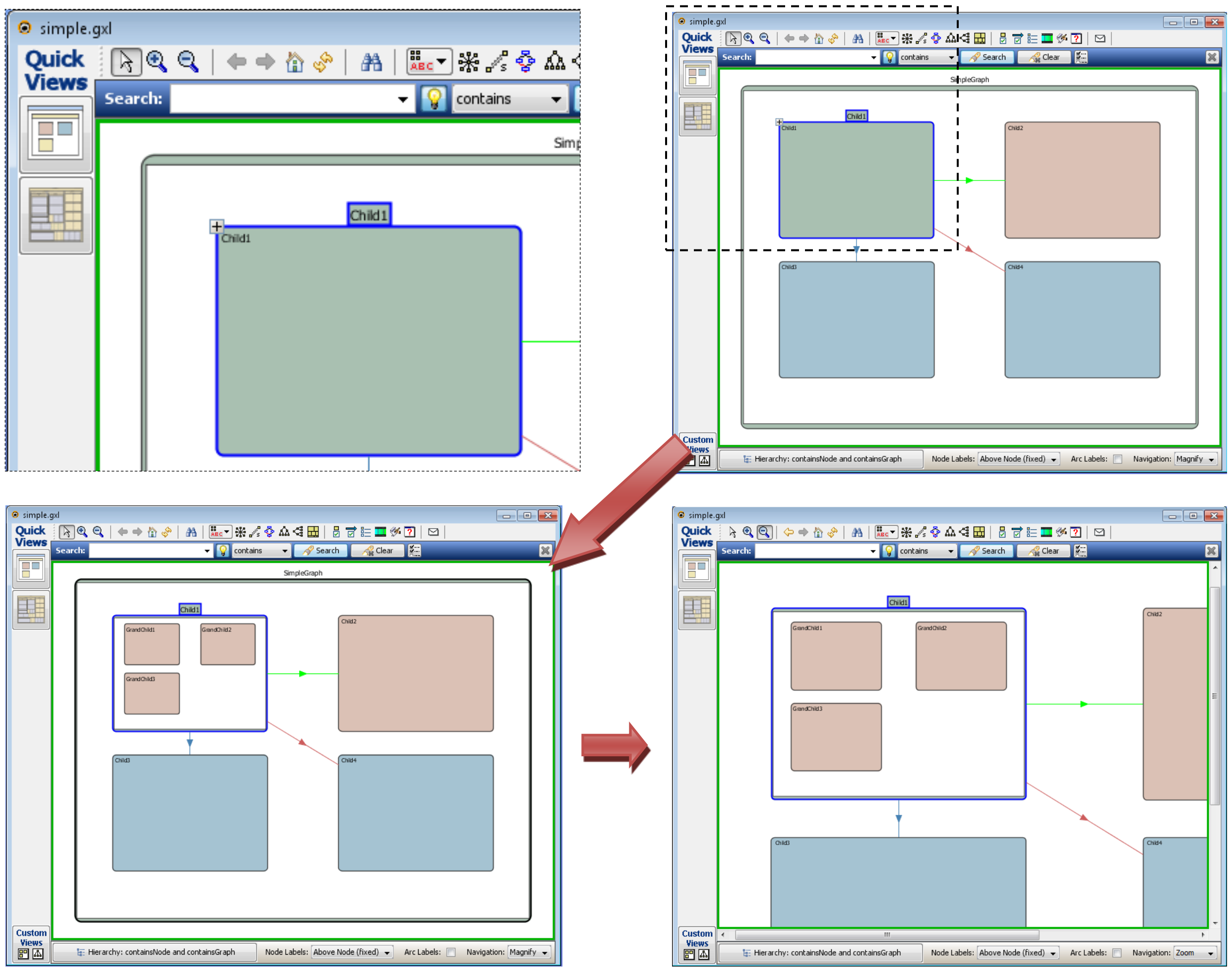


**Figure 7: Expanding Composite Nodes in JUNG**

Realizing that it would become difficult for users to fully explore large graphs of models in NOEM, we looked towards other visualization software for additional ideas. One particular software package called *SHriMP* (Simple Hierarchical Multi-Perspective) developed by the University of Victoria in Canada offered the ability to display interactive nested graphs with a "zoom-able" interface. *Figure 8* demonstrates how a user in *SHriMP* can explore a nested graph containing composite nodes. The green node in the upper- left labeled "Child1" is a composite node in a "closed" state (as indicate by a plus '+' icon). When you double-click on "Child1", it transitions to an "open" state to show its child nodes within its boundaries—as shown in the lower-left. You then can "zoom" into the nested graph to get a closer look as indicated by the image in the lower-right. The *SHriMP* project's novel approach for displaying nested graphs provided an opportunity for GNOME users to easily explore large graph models within NOEM and addressed the shortcomings of JUNG's ability to handle large nested graphs.



Hierarchy: containsNode and containsGraph   Node Labels: Above Node (fixed)   Arc Labels:   Navigation: Magnify

Hierarchy: containsNode and containsGraph   Node Labels: Above Node (fixed)   Arc Labels:   Navigation: Magnify

Hierarchy: containsNode and containsGraph   Node Labels: Above Node (fixed)   Arc Labels:   Navigation: Zoom

**Figure 8: Exploring Nested Graphs in *SHriMP***

### 3.3.2.4 Integration with NOEM

Initially, we looked at integrating the nested graph functionality of *SHriMP* with JUNG, but it quickly became apparent that the visualization component of JUNG would require a complete re-write to satisfy the requirements of *SHriMP*. Using the knowledge gained from converting the JUNG visualization component to be GWT-compatible (and thus be web-capable), we started to convert the *SHriMP* project to work with GWT.

One of the standard graph data formats that the *SHriMP* project primarily works with is GXL (Graph eXchange Language), since it supports a nested graph structure. In order to display the graphs representing the Ptolemy Models from NOEM, we had to augment the data conversion pipeline we previously implemented for GNOME to convert a Ptolemy model into a JUNG graph. We began with our JUNG graph data format and developed a converter to translate it to the GXL graph format, thus enabling our *SHriMP* component to process the data and display its graph representation.

### 3.3.2.5 Graph Display

*Figure 9* displays the most recent version of the Model Explorer tab for the thin-client version of NOEM running on Microsoft Internet Explorer 8.0.

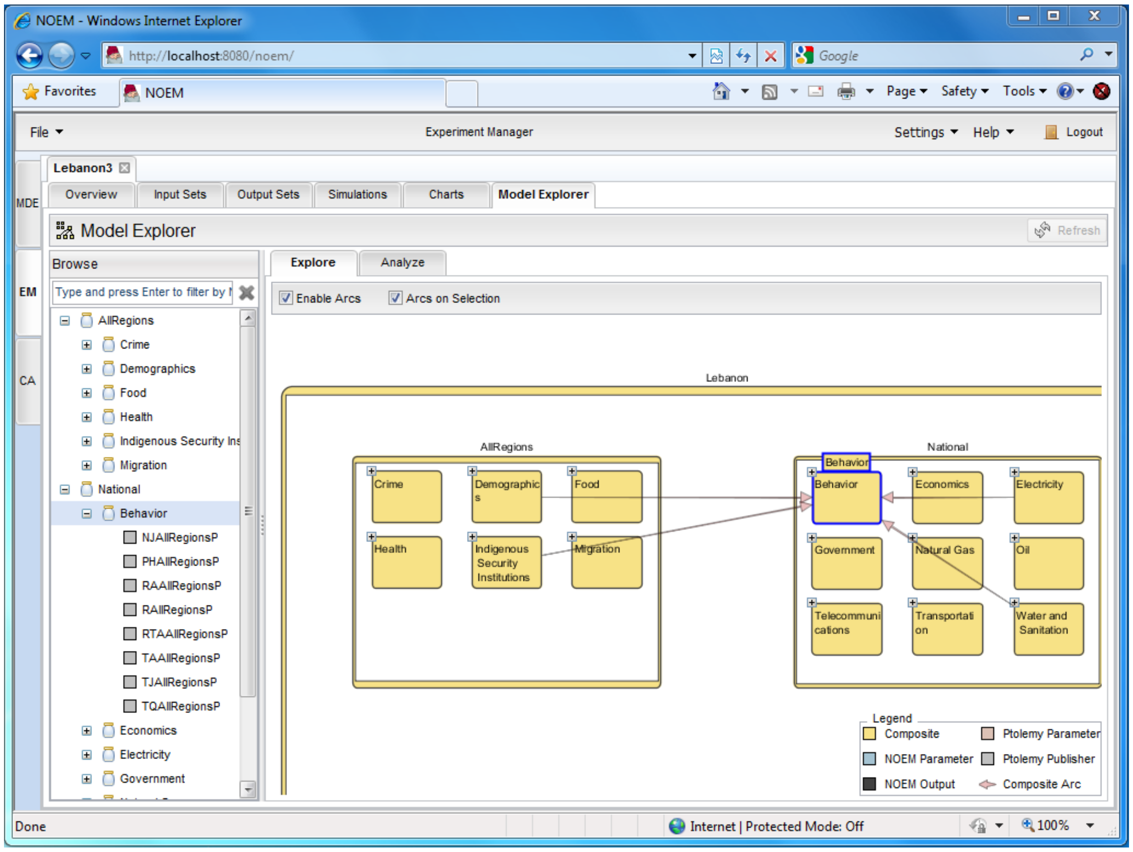


**Figure 9: Latest version of the thin-client Model Explorer under Internet Explorer 8**

The main viewport displays the graph visualization powered by a web-capable version of the *SHriMP* project. Users can interact with the graph view by using the mouse to perform various functions, such as click & drag to pan or the mouse wheel to zoom in/out. Double-clicking on composite nodes (indicated by the plus '+' icon) changes the node's state to "open" to allow its child nodes to become visible. All layout algorithms are performed on the server machine to alleviate processing time on the client machine, resulting in increased overall performance.

A useful, but not well-documented feature of the *SHriMP* visualization is composite arcs—which are similar to composite nodes in that they contain child arcs. Composite arcs help display relationships between nodes at a high-level since nodes may be hidden due to the parent-child relationships of nested graphs. *Figure 10* demonstrates how a composite arc (colored red) shows the relationships between the children nodes of two composite nodes. When both composite nodes are expanded into the "open" state to display their child nodes, the composite arc also expands to show the relationships among the child nodes. Much work was performed to ensure composite arcs were automatically created for the given NOEM model and to dynamically update them as the user explored the graph.

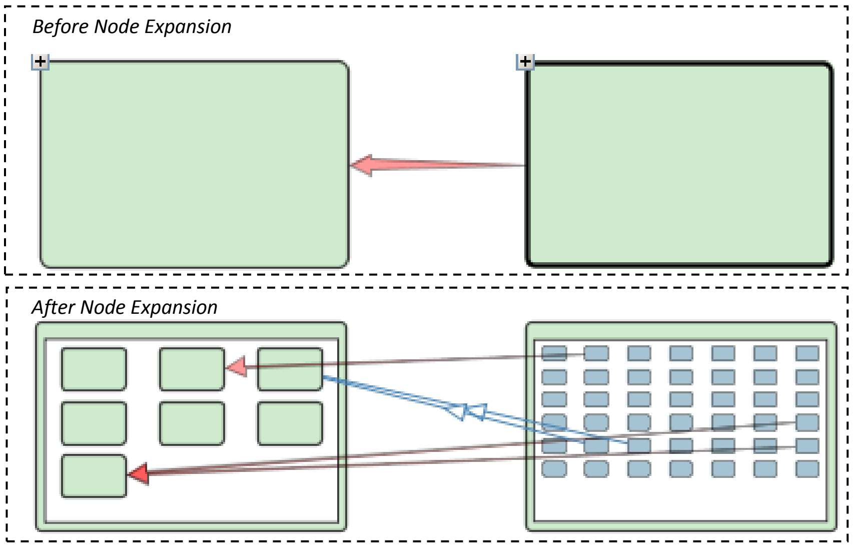


**Figure 10: Expanding composite arcs within the Model Explorer**

A custom "Browse" tree as shown in the left-side of *Figure 9* was developed to display the full hierarchy of the underlying graph. Icons are displayed next to the name of the tree node to indicate what type of object is represented, which are in turn represented in the "Legend" panel on the bottom-right of *Figure 9*. The text-box immediately at the top of the "Browse" tree allows the user to search for a particular node by name, filtering out all nodes that do match the given criteria. Double-clicking on a node in the "Browse" tree will cause the main graph view to zoom directly towards the selected node, which allows greater flexibility in exploring large and complex graphs.

Since the graph representation of NOEM models can become very complex, the "Analysis" pane was developed so users can isolate certain nodes and explore the relationships between them. As shown in *Figure 11*, the user can drag & drop nodes from the "Browse" tree into the "Analysis" pane to populate the graph. The "Clear" button in the top toolbar removes all nodes from the Analysis view.

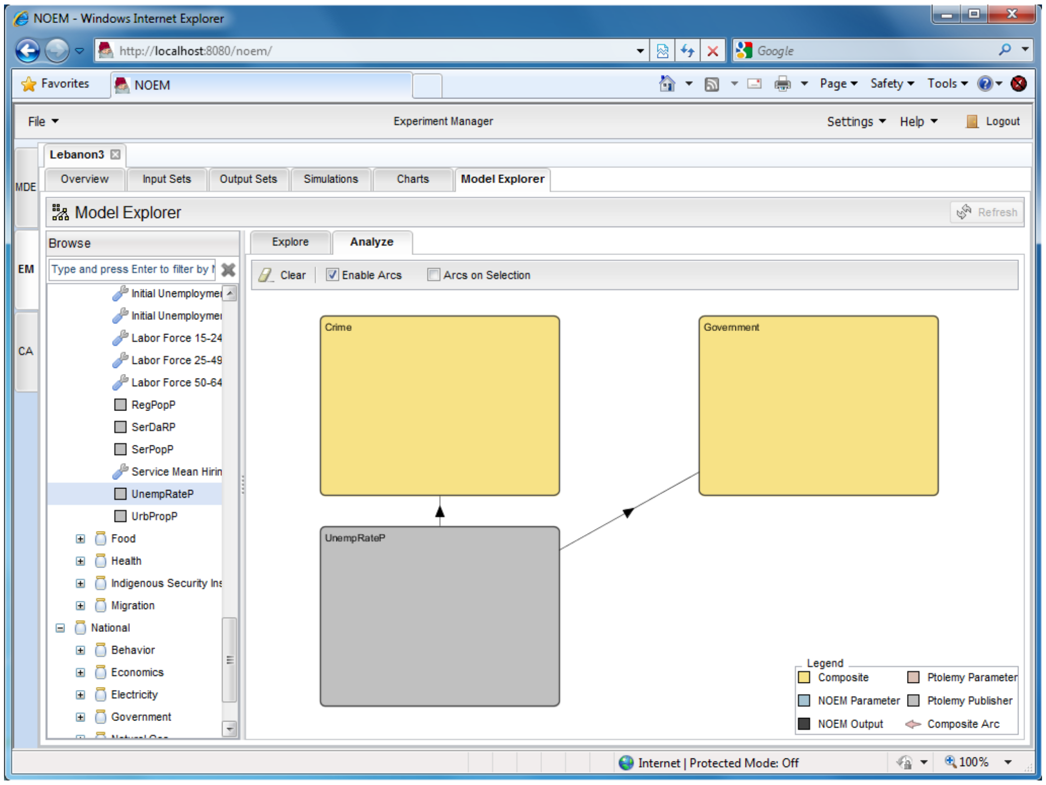
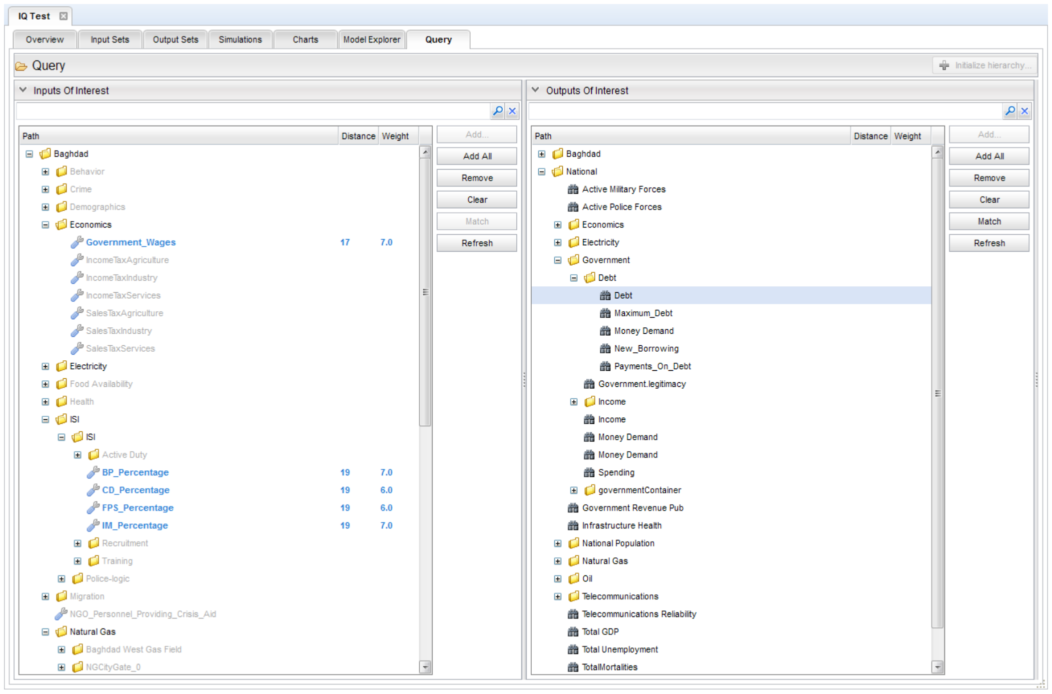


**Figure 11: The Analysis Pane within the Model Explorer**

### 3.3.3 Input-Output Query

A model query UI has been implemented in the thin-client (currently as a separate tab). This allows the user to display lists/trees of all inputs and outputs in the model, or just the ones that are of interest to them. Upon selecting an input or output, the user has the option to "Match" any linked outputs/inputs. Initially, this action displayed the shortest path distance through the model as well as random walk weights, but this was later updated to pull in normalized weights when available from the sampling phase. This also works with multi-selection to show the intersection (transitive closure) of linked results.

Initial weights were based on a random walk algorithm, and a scale from 0 to 100 was used to indicate the estimated likelihood that a significant effect existed. Additionally, the feasible inputs/outputs can be filtered down the top 'X' results (where X is specified by the user) based on the weighted values. When the user selects "Match" a prompt will allow them to indicate the ranking cutoff and a weighting algorithm will be run for each match, building a result set of size ‘X’. *Figure 12* and *Figure 13* Illustrate the original query UI for the output selection and input selection, respectively.



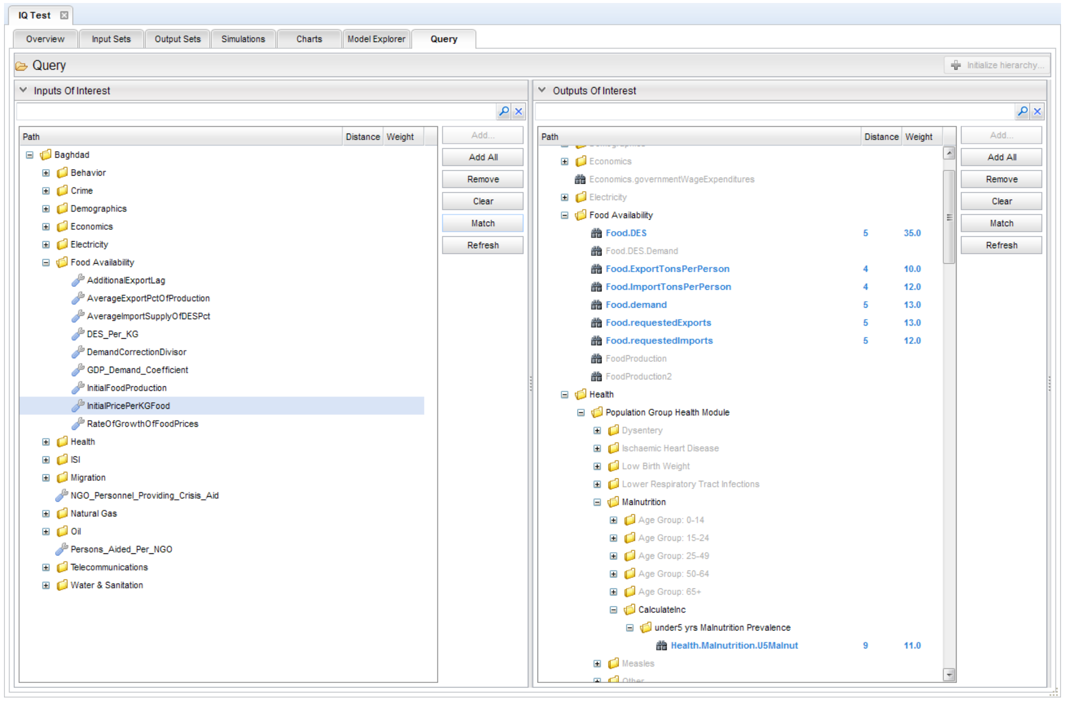
Inputs Of Interest

```
Path                                          Distance  Weight
Baghdad
  Behavior
  Crime
  Demographics
  Economics
    Government_Wages                          17        7.0
    IncomeTaxAgriculture
    IncomeTaxIndustry
    IncomeTaxServices
    SalesTaxAgriculture
    SalesTaxIndustry
    SalesTaxServices
  Electricity
  Food Availability
  Health
  ISI
    ISI
      Active Duty
      BP_Percentage                           19        7.0
      CD_Percentage                           19        6.0
      FPS_Percentage                          19        6.0
      IM_Percentage                           19        7.0
      Recruitment
      Training
    Police-logic
  Migration
  NGO_Personnel_Providing_Crisis_Aid
  Natural Gas
    Baghdad West Gas Field
    NGCityGate_0
```

Outputs Of Interest

```
Path                                          Distance  Weight
Baghdad
National
  Active Military Forces
  Active Police Forces
  Economics
  Electricity
  Government
    Debt
      Debt
      Maximum_Debt
      Money Demand
      New_Borrowing
      Payments_On_Debt
    Government.legitimacy
    Income
    Income
    Money Demand
    Money Demand
    Spending
    governmentContainer
  Government Revenue Pub
  Infrastructure Health
  National Population
  Natural Gas
  Oil
  Telecommunications
  Telecommunications Reliability
  Total GDP
  Total Unemployment
  TotalMortalities
```

**Figure 12: *Feasible Inputs for Selected Output***



Inputs Of Interest

```
Path                                          Distance  Weight
Baghdad
  Behavior
  Crime
  Demographics
  Economics
  Electricity
  Food Availability
    AdditionalExportLag
    AverageExportPctOfProduction
    AverageImportSupplyOfDESPct
    DES_Per_KG
    DemandCorrectionDivisor
    GDP_Demand_Coefficient
    InitialFoodProduction
    InitialPricePerKGFood
    RateOfGrowthOfFoodPrices
  Health
  ISI
  Migration
  NGO_Personnel_Providing_Crisis_Aid
  Natural Gas
  Oil
  Persons_Aided_Per_NGO
  Telecommunications
  Water & Sanitation
```

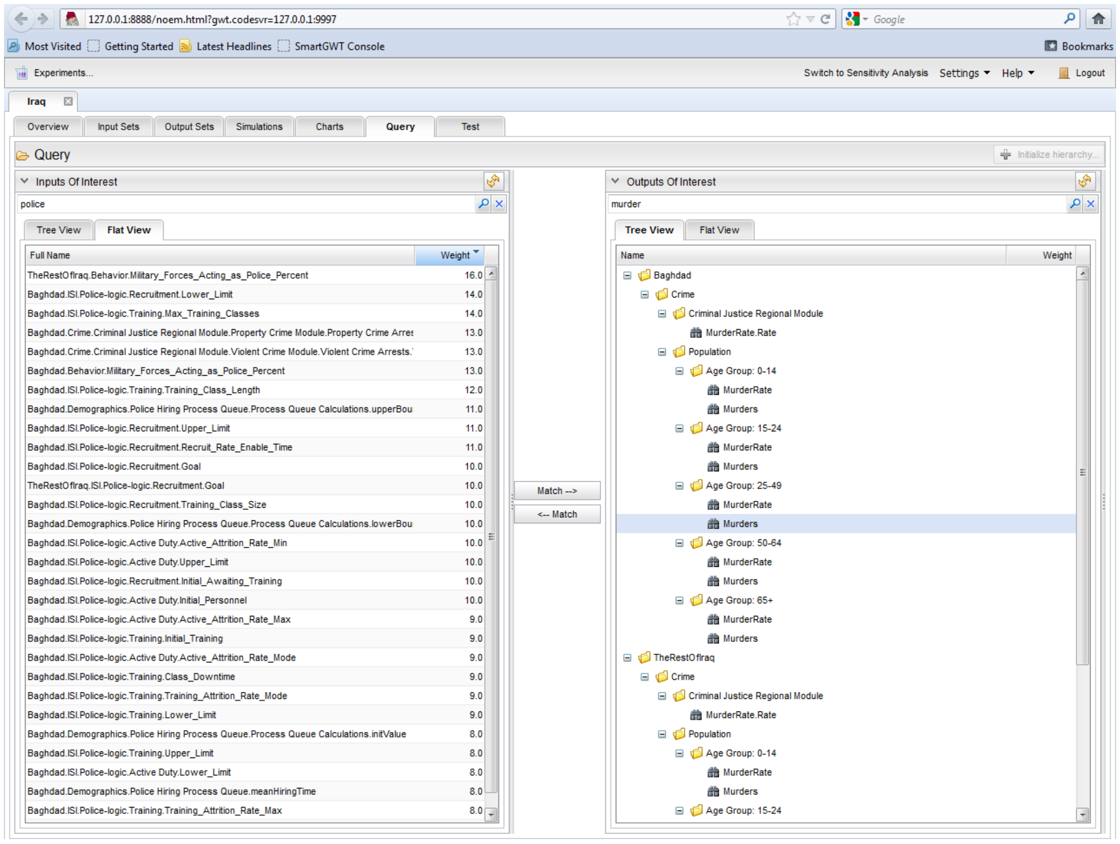
Outputs Of Interest

```
Path                                          Distance  Weight
  Economics
  Economics.governmentWageExpenditures
  Electricity
  Food Availability
    Food.DES                                  5         35.0
    Food.DES.Demand
    Food.ExportTonsPerPerson                  4         10.0
    Food.ImportTonsPerPerson                  4         12.0
    Food.demand                               5         13.0
    Food.requestedExports                     5         13.0
    Food.requestedImports                     5         12.0
    FoodProduction
    FoodProduction2
  Health
    Population Group Health Module
      Dysentery
      Ischaemic Heart Disease
      Low Birth Weight
      Lower Respiratory Tract Infections
      Malnutrition
        Age Group: 0-14
        Age Group: 15-24
        Age Group: 25-49
        Age Group: 50-64
        Age Group: 65+
        CalculateInc
          under5 yrs Malnutrition Prevalence
            Health.Malnutrition.U5Malnut      9         11.0
      Measles
      Other
```

**Figure 13*: Reachable Outputs for Selected Input***

The model query service structure uses SmartGWT for handling data. Data Source components are utilized for inputs and outputs, improving the client-side response including searching and filtering available results. This includes expression handling for searching/filtering as well.

Further UI development included a reorganized button setup and an additional list grid or "flat view" of the search and match results for inputs and outputs (see *Figure 14*). This flat view can be sorted easily using column headings and is useful for ranking the weighted values. Also, the button panels have been simplified with a single central panel to match inputs or outputs.



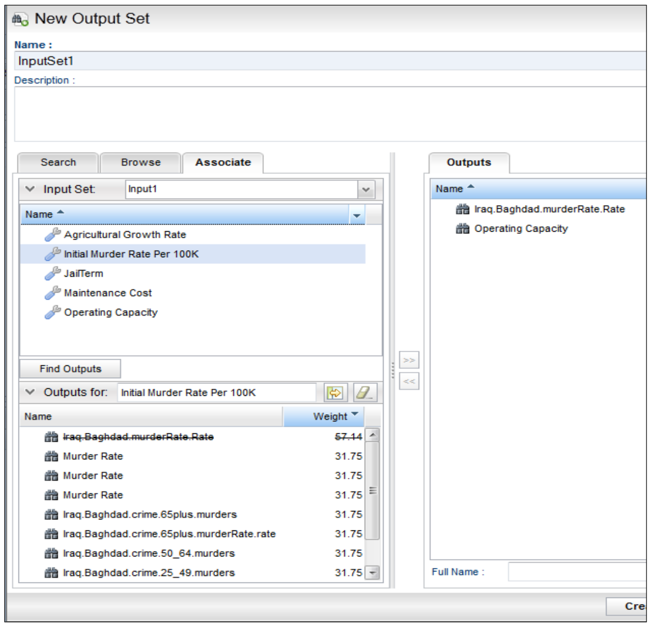
| Full Name | Weight |
|---|---|
| TheRestOfIraq.Behavior.Military_Forces_Acting_as_Police_Percent | 16.0 |
| Baghdad.ISI.Police-logic.Recruitment.Lower_Limit | 14.0 |
| Baghdad.ISI.Police-logic.Training.Max_Training_Classes | 14.0 |
| Baghdad.Crime.Criminal Justice Regional Module.Property Crime Module.Property Crime Arres | 13.0 |
| Baghdad.Crime.Criminal Justice Regional Module.Violent Crime Module.Violent Crime Arrests.' | 13.0 |
| Baghdad.Behavior.Military_Forces_Acting_as_Police_Percent | 13.0 |
| Baghdad.ISI.Police-logic.Training.Training_Class_Length | 12.0 |
| Baghdad.Demographics.Police Hiring Process Queue.Process Queue Calculations.upperBou | 11.0 |
| Baghdad.ISI.Police-logic.Recruitment.Upper_Limit | 11.0 |
| Baghdad.ISI.Police-logic.Recruitment.Recruit_Rate_Enable_Time | 11.0 |
| Baghdad.ISI.Police-logic.Recruitment.Goal | 10.0 |
| TheRestOfIraq.ISI.Police-logic.Recruitment.Goal | 10.0 |
| Baghdad.ISI.Police-logic.Recruitment.Training_Class_Size | 10.0 |
| Baghdad.Demographics.Police Hiring Process Queue.Process Queue Calculations.lowerBou | 10.0 |
| Baghdad.ISI.Police-logic.Active Duty.Active_Attrition_Rate_Min | 10.0 |
| Baghdad.ISI.Police-logic.Active Duty.Upper_Limit | 10.0 |
| Baghdad.ISI.Police-logic.Recruitment.Initial_Awaiting_Training | 10.0 |
| Baghdad.ISI.Police-logic.Active Duty.Initial_Personnel | 10.0 |
| Baghdad.ISI.Police-logic.Active Duty.Active_Attrition_Rate_Max | 9.0 |
| Baghdad.ISI.Police-logic.Training.Initial_Training | 9.0 |
| Baghdad.ISI.Police-logic.Active Duty.Active_Attrition_Rate_Mode | 9.0 |
| Baghdad.ISI.Police-logic.Training.Class_Downtime | 9.0 |
| Baghdad.ISI.Police-logic.Training.Training_Attrition_Rate_Mode | 9.0 |
| Baghdad.ISI.Police-logic.Training.Lower_Limit | 9.0 |
| Baghdad.Demographics.Police Hiring Process Queue.Process Queue Calculations.initValue | 8.0 |
| Baghdad.ISI.Police-logic.Training.Upper_Limit | 8.0 |
| Baghdad.ISI.Police-logic.Active Duty.Lower_Limit | 8.0 |
| Baghdad.Demographics.Police Hiring Process Queue.meanHiringTime | 8.0 |
| Baghdad.ISI.Police-logic.Training.Training_Attrition_Rate_Max | 8.0 |

| Name | Weight |
|---|---|
| Baghdad | |
| Crime | |
| Criminal Justice Regional Module | |
| MurderRate.Rate | |
| Population | |
| Age Group: 0-14 | |
| MurderRate | |
| Murders | |
| Age Group: 15-24 | |
| MurderRate | |
| Murders | |
| Age Group: 25-49 | |
| MurderRate | |
| Murders | |
| Age Group: 50-64 | |
| MurderRate | |
| Murders | |
| Age Group: 65+ | |
| MurderRate | |
| Murders | |
| TheRestOfIraq | |
| Crime | |
| Criminal Justice Regional Module | |
| MurderRate.Rate | |
| Population | |
| Age Group: 0-14 | |
| MurderRate | |
| Murders | |
| Age Group: 15-24 | |

**Figure 14: Input/Output Connectivity Interface**

### 3.3.4 Associate Feature

The UI for the model query was also implemented (in a simplified form) within the output set dialog to aid in selecting outputs that are properly linked to the current input set's inputs. *Figure 15* illustrates this new query UI as the "Associate" tab within the dialog. The inputs available are populated from a selected input set, and the "Find Outputs" option performs a service call to utilize GNOME's matching and weighting algorithms.


New Output Set

Name : InputSet1

Description :

Search | Browse | Associate

Input Set: Input1

| Name |
|---|
| Agricultural Growth Rate |
| Initial Murder Rate Per 100K |
| JailTerm |
| Maintenance Cost |
| Operating Capacity |

Find Outputs

Outputs for: Initial Murder Rate Per 100K

| Name | Weight |
|---|---|
| ~~Iraq.Baghdad.murderRate.Rate~~ | ~~57.14~~ |
| Murder Rate | 31.75 |
| Murder Rate | 31.75 |
| Murder Rate | 31.75 |
| Iraq.Baghdad.crime.65plus.murders | 31.75 |
| Iraq.Baghdad.crime.65plus.murderRate.rate | 31.75 |
| Iraq.Baghdad.crime.50_64.murders | 31.75 |
| Iraq.Baghdad.crime.25_49.murders | 31.75 |

Outputs

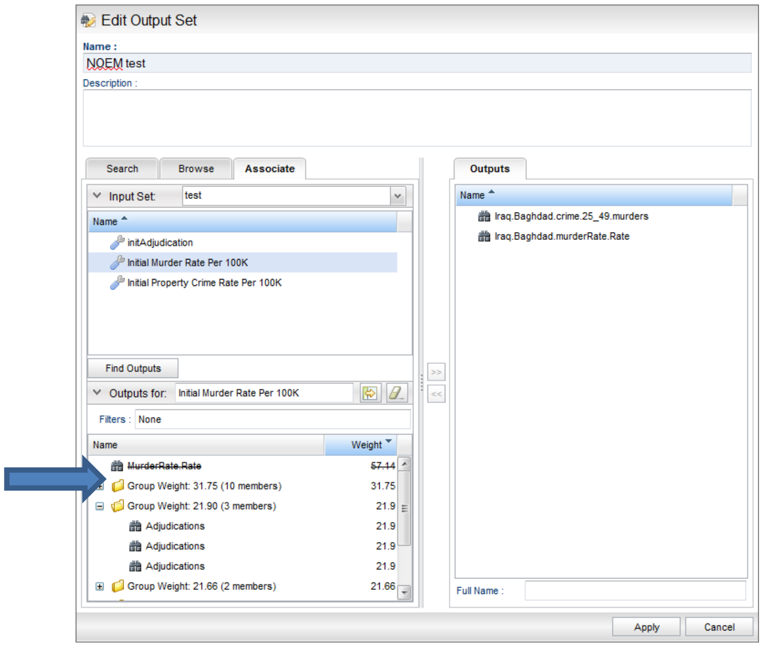
| Name |
|---|
| Iraq.Baghdad.murderRate.Rate |
| Operating Capacity |

Full Name :

Cre

**Figure 15: Associate Panel within Output Set Dialog**

Further development of the associate tab provided filters to match or ignore a specified expression. Additionally, similar outputs are grouped into folders within the feasible output listing, as is shown in *Figure 16*. Both features further aid in finding outputs of interest. A similar "associate" interface was also established within the Input Set Dialog to find feasible inputs for selected outputs.


Edit Output Set

Name : NOEM test

Description :

Search | Browse | Associate

Input Set: test

| Name |
|---|
| initAdjudication |
| Initial Murder Rate Per 100K |
| Initial Property Crime Rate Per 100K |

Find Outputs

Outputs for: Initial Murder Rate Per 100K

Filters : None

| Name | Weight |
|---|---|
| ~~MurderRate.Rate~~ | ~~57.14~~ |
| Group Weight: 31.75 (10 members) | 31.75 |
| Group Weight: 21.90 (3 members) | 21.9 |
| Adjudications | 21.9 |
| Adjudications | 21.9 |
| Adjudications | 21.9 |
| Group Weight: 21.66 (2 members) | 21.66 |

Outputs

| Name |
|---|
| Iraq.Baghdad.crime.25_49.murders |
| Iraq.Baghdad.murderRate.Rate |

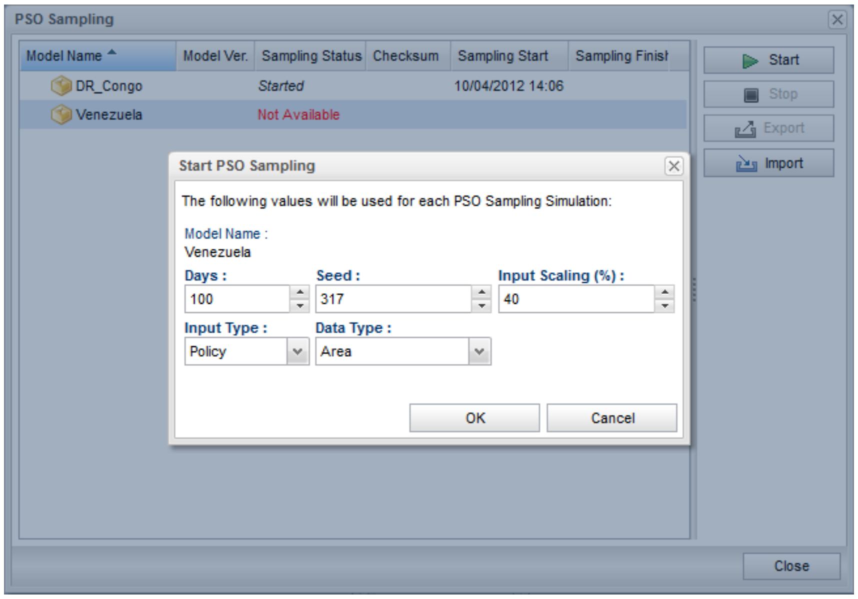
Full Name :

Apply | Cancel

**Figure 16: Output Set Dialog with Weight Grouping**

### 3.3.5 Sampling UI

A sampling dialog has been added to the "Administration" menu in NOEM, and allows sampling experiments to be performed with the loaded models. The UI includes a list of the available models and details, and the start dialog provides some simulation and input configuration settings (see *Figure 17*). The input scaling indicates how much spread to use in the difference between low and high sampling values for each input. The input type utilizes categorical information for the parameters to limit the sampling to a more specific subset if desired. Also, the data type allows the sample data to be represented as either final values or total area of the output's response function.



PSO Sampling

| Model Name | Model Ver. | Sampling Status | Checksum | Sampling Start | Sampling Finish |
|---|---|---|---|---|---|
| DR_Congo | | Started | | 10/04/2012 14:06 | |
| Venezuela | | Not Available | | | |

Start PSO Sampling

The following values will be used for each PSO Sampling Simulation:

Model Name : Venezuela

Days : 100

Seed : 317

Input Scaling (%) : 40

Input Type : Policy

Data Type : Area

**Figure 17: Sampling in Web-client**

## 3.4 Goal-based Parameter Value Estimation

The purpose of this research work is to enhance the ability to describe the relationships between input and output values under uncertainty in a graph-based simulation model. In particular, we propose a method for estimating the range of input parameter values that are expected to result in a range of user-specified output values (or goals) for an individual output performance measure. The following sections show the development of the methodology first looking at the procedures for estimating output performance measures across a path given input parameter values and then the methods used to reverse that process for estimating the input values needed to achieve an output performance measure within a specified range.

### 3.4.1 Confidence Intervals

Error exists when data comes from a sample of a population instead of the entire population. When this data is used in experiments or calculations of statistical estimates, uncertainty exists in those measures. Confidence intervals are used as a way to measure the uncertainty.

The confidence interval of any statistic is calculated based on three things, the confidence level (α), the amount of error, and the value of the statistic. The confidence level represents the percentage of samples of a certain size expected to include the true mean. It is typically set at 95%, which means that 95% of the confidence intervals calculated from all samples collected are likely to contain the true mean.

One of the most useful and simple statistics is the mean. In calculating the confidence interval for the mean, the standard deviation (σ) is used for the measurement error and the mean (μ) is used for the value of the statistic. The equation for the two-sided confidence interval on the mean is:

$$\mu \pm z_{1-\alpha/2} * \sigma \tag{1}$$

where z is the critical value for the Normal distribution. Since these values are not usually known, the sample standard deviation (s) is used to approximate the population standard deviation, and the sample mean ($\overline{x}$) is used to approximate the population mean. The Student-t distribution with n-1 degrees of freedom is used to approximate the Normal distribution. The confidence interval is:

$$\overline{x} \pm t_{n-1,1-\alpha/2} \frac{s}{\sqrt{n}} \tag{2}$$

In GNOME, edge weights can be created to represent the effect of one actor on another. A confidence interval for each edge weight can be made based on repeated replications of the model. Using the above equation, the weight's confidence interval is calculated using the number of replications for the sample size, and a sample mean and standard deviation computed from the weights resulting from each of the runs. The edge weights and their intervals are used in both the sequential sampling and the path weighing procedures.

### 3.4.2 Sequential Sampling

Sequential sampling is a sampling method that aims to reach some target size for a confidence interval while minimizing the number of samples taken to calculate it (see *Figure 18)*. The method involves taking a few initial samples/replications to get a rough idea of the mean and standard deviation, then using that information to calculate the number of samples needed in order to reach the target size. Then additional samples are taken, and a new confidence interval is calculated. If the size of the new confidence interval is small enough the process stops, otherwise more samples are taken and the confidence interval is recalculated until the wanted precision is reached.

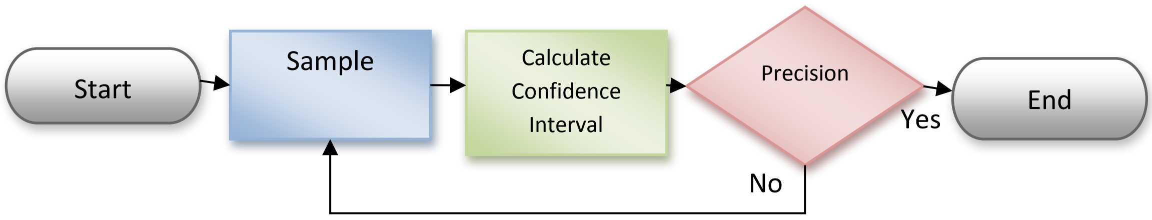


**Figure 18: Sampling for Confidence Intervals**

Typical implementations of this method involve taking one sample at a time in order, hence the name sequential sampling. This is often used for lot acceptance quality testing in manufacturing industries where it is ideal to minimize the number of samples taken due to high testing costs. In a simulation model, the main cost associated with sampling is time. The cost per sample is relatively low, so it is faster to batch several samples at once before recalculating the confidence interval. The same amount of initial samples is taken for every actor, and then additional samples are taken based on how many are needed to reduce the size of its confidence interval.

The half-width for the confidence interval is the part after the ±, or:

$$h = t_{n-1,1-\alpha/2} \frac{s}{\sqrt{n}} \tag{3}$$

This equation shows that the half-width depends on the t-value (which itself depends on sample size and the confidence level), standard deviation, and sample size. The easiest of these to control is the sample size. Solving for n:

$$n = t_{n-1,1-\alpha/2}^2 \frac{s^2}{h^2} \tag{4}$$

Since the t-value depends on the sample size as well, it can be approximated with a z-value from the normal distribution to get rid of sample size on the right side of the equation. As n approaches infinity, the Student-t distribution becomes more and more like the normal distribution, so this approximation is more accurate for larger values of n. The equation becomes:

$$n \cong z_{1-\alpha/2}^2 \frac{s^2}{h^2} \tag{5}$$

The calculation comes up with the necessary number of samples for the confidence interval to be the wanted size, for the given confidence level, assuming both the mean and variance remain relatively constant for all additional samples. This assumption becomes more and more valid the more samples are taken. However sometimes the initial sample, or even subsequent samples, does not accurately represent the actor's true behavior. A way to avoid the problems caused by a bad initial sample is to recheck the confidence interval, the mean, and the standard deviation after sampling only a percent of the calculated sample size instead of after sampling the entire amount. More information is available each time, so incremental inspection balances sampling and calculation processing times while maintaining a high level of accuracy.

Sample and iteration limits are set as safeguards to prevent an actor from potentially taking excessively long amounts of time to process.  The sample limit is the maximum number of samples to take before moving on. This prevents a very bad initial sample or a highly variable actor from stalling the weighting process.  The iteration limit sets the maximum number of sampling iterations to go through for an actor before stopping. This avoids the situation that arises sometimes where a small number of samples needed, when the current confidence interval is already very close to the desired precision, are calculated each time, but the desired precision is never fully reached.

In order to take the different magnitudes of weights into account, the desired size of the confidence interval varies per actor. The sample mean is used not only as the center of the confidence interval, but also the basis of the range.

$$h = \bar{x} * (halfwidth\ precision) \tag{6}$$

The half-width precision is the percent of the mean that is added or subtracted from the mean itself to determine the confidence interval’s desired upper and lower bounds, respectively. This allows the size goal to scale to the likely values output by the actor, as opposed to having a fixed size for all actors, which would be a poor fit for most. An actor with expected values between zero and one and another with expected values in the millions should have different half-widths in order to be reasonably useful.

Processing times for the sequential sampling method are greater than the fixed replication method. Going from sampling the entire model at once to sampling each actor individually is the primary reason for the time difference. Establishing the sample size is an iterative process that requires more calculations than just using a random value, which increases also processing time. However, it efficiently determines the necessary sample size and only takes as many as needed. The fixed replication method has no way to control the size of the confidence interval, and the number of replications is arbitrarily determined and constant across all actors in the model.

### 3.4.3 Path Weighting

A path is the set of vertices and edges that connect any two vertices in the model. Having information on the path between nodes can help in analyzing the relationship between the nodes themselves. Using the confidence intervals of the members in the path, it is possible to gauge the variability in the relationship of the nodes, determine a “best/worst case scenario” for what would happen if all the outputs along the path came out at the highest or lowest level that they would probably be, and develop a confidence interval for the path describing the upper and lower bound of these extreme cases. The path weighting process has two parts: finding the path, and then altering the confidence intervals.

The only required inputs to determine any path in the model are two connected nodes: the start node and the end node. The two are usually not interchangeable, as the path from any node A to any other node B is probably not the same as the path from node B to node A, if it exists at all. The method to determine the path uses the Reachability feature in GNOME, which detects all the nodes that can be reached from the selected node, and the shortest distance between the two.  It can also be toggled to find Reachability in the reverse direction. Any member belonging to the shortest path can be found by comparing its distance from the start node using the normal direction and the end node with Reachability in reverse. This is illustrated in *Figure 19*.  Only members of the path between the start and end nodes would be in both the forward and backward runs, at distances corresponding to the shortest path's length. This gets not only what nodes and edges comprise the path, but their sequence as well.

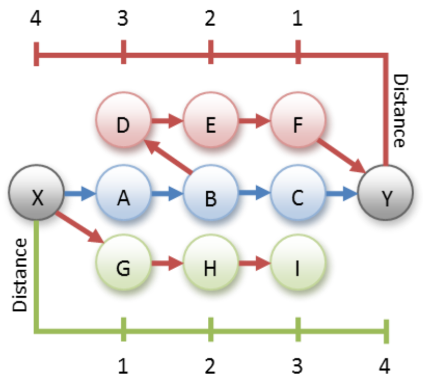


**Figure 19: Path Determination**

Node X is the start of the path and node Y is the end. The path XABCY is the shortest path from node X to node Y. Path XGHI is eliminated because it is not reachable from node Y, and likewise DEFY is eliminated because it is not reachable from node X.  Path XABDEFY is a complete path from node X to node Y, but it isn't a shortest path. Using the path XABDEFY, the distance between X and Y would be six, but the shortest distance from X to Y using path XABCY is only four.

A conflict with this method is when multiple paths of the same length exist between the start and end nodes. One way to deal with this is to manually set the entire path. Since path determination is based on shortest distances, manual setup is also recommended for alternative and complex paths involving things like non-shortest distance paths, loops, or multiple points of convergence and divergence. A unique path can still be found using another forward/backward check, from a node from each branch to the start and end nodes (illustrated in *Figure 20*). These branch nodes that are selected are the first nodes after a path splits into several branches.

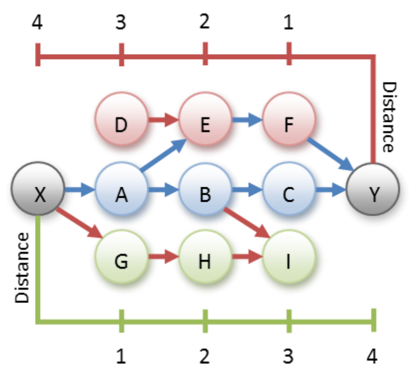


**Figure 20: Multi-path Determination**

The path XABCY and XAEFY are both valid, shortest paths. Nodes B and E are the branch nodes used to identify each unique path.

The second part of the path weighting process is the actual weighting of the path. Once the path has been set and the model's edge weights have been loaded, the path weighting process moves node by node, modifying each node's input based on the previous node. Both the overall mean and variance of the path are updated in this piecewise manner. Means are combined with the following equation:

$$\bar{x} * \bar{y} \tag{7}$$

and variances are combined with this equation:

$$\left(\bar{x}^2 * \sigma_y^2\right) + \left(\bar{y}^2 * \sigma_x^2\right) + \left(\sigma_x^2 * \sigma_y^2\right) \tag{8}$$

Nodes with no mean or variance simply pass the information through without modification.

### 3.4.4 Estimating Ranges for Input Values

In addition to using confidence intervals to predict outputs, they can also be used to predict inputs for a certain goal. Once a confidence interval on the outputs has been determined, it can be used to calculate a range of inputs that should result in a given range of outputs. A base input and base output are needed to determine the range of inputs. The input range is calculated from the following equations, which were obtained from the confidence interval equations, with the output interval bound known and solving for the input:

$$\text{Input LB: } \left(\frac{\left((Output_{LB} - Output_{Base}) * Input_{Base}\right)}{\left(Output_{Base} * (Path\,Weight + Path\,Halfwidth)\right)}\right) + Input_{Base} \tag{9}$$

$$\text{Input UB: } \left(\frac{\left((Output_{UB} - Output_{Base}) * Input_{Base}\right)}{\left(Output_{Base} * (Path\,Weight - Path\,Halfwidth)\right)}\right) + Input_{Base}$$

These equations try to answer the question, “Given these bounds for an output, what input would result in an output within these bounds 1-α% of the time.” The biggest shortcoming of this method is when the desired output range is small and the variance of the path is large.

#### 3.4.4.1 Measures of Significance for Estimates

Once a range of inputs for a given set of outputs has been, it is possible to determine the “minimum level of significance” of that input range. That is to say, what confidence can be achieved that any input in the calculated input range will result in an output within the goal range. Several points between the input bounds are selected, and an associated alpha value is determined using the t-distribution.

$$|output\ bound \pm \overline{x_i}| = t_{\alpha/2,n-1} * \frac{s}{\sqrt{n}} \rightarrow \mathrm{t}_{\alpha/2,n-1} = |output\ bound \pm \overline{x_i}| \frac{\sqrt{n}}{s} \qquad (10)$$

The highest alpha value returned is the “minimum level of significance.” For that set of input and output intervals, there is at least a certain level of confidence that any input in the input interval will result in an output in the output interval.

# 4 Results and Discussion

This section illustrates some of the results and findings from the methods and implementations covered in Section 3. The results of GNOME II are divided into two categories: the sampling and weighting (a quantitative approach), and model exploration (a qualitative approach). Examples are provided to aid in the discussion.

## 4.1 Sampling and Weighting

Part of the focus of GNOME II is to provide quantitative metrics by which to compare relationships within the models under study. Such metrics are provided through the experimental sampling and weighting capabilities of the software. The sampling allows individual input-to-output relationships to be identified and quantified. For example, *Table 1* depicts the effects that 8 inputs have on 16 outputs in the DR Congo model.

Table 1: Example Main-effects for DR Congo Inputs and Outputs

| | Field | Input Value | Region 5, 25-49 Unemployment Rate | Region 5, GINI value | Region 3, 25-49 Murder Rate | Region 4, Food Granted Exports | National Murder Rate | Region 5, 50-64 Unemployment Rate | Region 1, 25-49 Unemployed | Region 5, 25-49 Murder Rate | Government Income | National GDP | Region 2, Food Production per Person | Region 4, 50-64 Unemployment Rate | Region 2, 15-24 Murder Rate | Region 2, 15-24 Adjudication Rate | Region 3, Granted Food Exports | Region 5, 0-14 Murders |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Baseline* | - | N/A | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2215.00892 | 15.213238 | 35139505.26 | 9E-06 | 10891230460 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | - | | | | | | | | | | | | | | | | | |
| *DRCongo.Region3.EconomicsParameters.Agriculture_Wages* | **H** | 676 | 13.19673659 | 51.30083866 | 8.65969E-06 | 2.4E+08 | 1956.74700 | 15.213238 | 35139505.26 | 9E-06 | 10974790624 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | -7.0409E-06 | -164.25 | -258.26192 | 0 | 0 | 0 | 83560163.82 | -4818.1 | 0 | 0 | 0 | 0 | -46560 | 0 |
| | +/- % | | 0% | 0% | -45% | 0% | -12% | 0% | 0% | 0% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | **L** | 364 | 13.19673659 | 51.30083866 | 4.16321E-05 | 2.4E+08 | 3166.49147 | 15.213238 | 35139505.26 | 9E-06 | 10807603651 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 2.59315E-05 | 458.272 | 951.48255 | 0 | 0 | 0 | -83626808.67 | -192467 | 0 | 0 | 0 | 0 | -2E+05 | 0 |
| | +/- % | | 0% | 0% | 165% | 0% | 43% | 0% | 0% | 0% | -1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| *DRCongo.Region5.CrimeModule.Criminal Justice Regional Module.initialMurderRatePer100k* | **H** | 32.63 | 13.19650103 | 51.29849942 | 1.57006E-05 | 2.4E+08 | 2266.98197 | 15.194457 | 35139505.26 | 1E-05 | 10891217845 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 30.640715 |
| | +/- | | -0.00023556 | -0.00233923 | 0 | 158.161 | 51.97305 | -0.018781 | 0 | 1E-06 | -12614.52744 | -15187 | 0 | 0 | 0 | 0 | 308.26 | 4.323836 |
| | +/- % | | 0% | 0% | 0% | 0% | 2% | -0.1% | 0% | 16% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 16% |
| | **L** | 17.57 | 13.18846951 | 51.29937906 | 1.57006E-05 | 2.4E+08 | 2155.83800 | 15.219071 | 35139505.26 | 7E-06 | 10891234873 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 21.394327 |
| | +/- | | -0.00826708 | -0.00145959 | 0 | -31.282 | -59.17092 | 0.005833 | 0 | -2E-06 | 4413.33523 | -5697 | 0 | 0 | 0 | 0 | -61.23 | -4.922552 |
| | +/- % | | 0% | 0% | 0% | 0% | -3% | 0.0% | 0% | -19% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | -19% |
| *DRCongo.Region5.DemographicsModule.initIndustryShareOfEmp* | **H** | 0.143 | 13.06472885 | 50.88075812 | 1.57006E-05 | 2.4E+08 | 2211.68399 | 15.064882 | 35139505.26 | 9E-06 | 10909066388 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.041118 |
| | +/- | | -0.13200773 | -0.42008054 | 0 | -708.43 | -3.32494 | -0.148356 | 0 | -9E-08 | 17835927.85 | 201637 | 0 | 0 | 0 | 0 | -1380 | -0.275761 |
| | +/- % | | -1% | -1% | 0% | 0% | 0% | -1.0% | 0% | -1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | -1% |
| | **L** | 0.077 | 13.32195631 | 51.67062294 | 1.57006E-05 | 2.4E+08 | 2217.93209 | 15.353671 | 35139505.26 | 9E-06 | 10873361678 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.559287 |
| | +/- | | 0.12521972 | 0.369784284 | 0 | 1330.57 | 2.92317 | 0.140433 | 0 | 8E-08 | -17868781.74 | -307735 | 0 | 0 | 0 | 0 | 2592.2 | 0.242408 |
| | +/- % | | 1% | 1% | 0% | 0% | 0% | 0.9% | 0% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 1% |
| *DRCongo.Region5.LandArea* | **H** | 431867 | 13.19515944 | 51.3011578 | 1.57006E-05 | 2.4E+08 | 2214.56192 | 15.213175 | 35139505.26 | 9E-06 | 10891227648 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.279763 |
| | +/- | | -0.00157715 | 0.000319142 | 0 | 138.342 | -0.44701 | -0.000063 | 0 | -1E-08 | -2811.923925 | -22050 | 0 | 0 | 0 | 0 | 269.31 | -0.037116 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | 0.0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | **L** | 232544 | 13.19587271 | 51.3003642 | 1.57006E-05 | 2.4E+08 | 2215.85998 | 15.194918 | 35139505.26 | 9E-06 | 10891226879 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.387525 |
| | +/- | | -0.00086388 | -0.00047446 | 0 | 202.984 | 0.85105 | -0.018320 | 0 | 2E-08 | -3581.002277 | -10141 | 0 | 0 | 0 | 0 | 395.59 | 0.070646 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | -0.1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| *DRCongo.Region.EconomicsParameters.IncomeTaxServices* | **H** | 0.6461 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2215.0089 | 15.213238 | 35139505.26 | 9E-06 | 11012899693 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 121669233.6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | 0.0% | 0% | 0% | 1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | **L** | 0.3479 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2215.0089 | 15.213238 | 35139505.26 | 9E-06 | 10769561226 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -121669233.6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | 0.0% | 0% | 0% | -1% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| *DRCongo.Region5.FoodAvailability.RateOfGrowthOfFoodPrices* | **H** | 1.3 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2215.0089 | 15.213238 | 35139505.26 | 9E-06 | 1.53E+10 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4387667598 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 40% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | **L** | 0.7 | 15.41976733 | 53.17770727 | 1.58471E-05 | 2.1E+08 | 2258.3845 | 17.641914 | 40964263.16 | 9E-06 | 9777632427 | 1.2E+10 | 123639 | 17.95006 | 2.1869E-05 | 0.003382 | 4E+08 | 26.773024 |
| | +/- | | 2.22303074 | 1.876868612 | 1.46519E-07 | -2E+07 | 43.3755 | 2.428676 | 5824757.895 | 2E-07 | -1113598033 | -1E+08 | -1839 | 2.65316 | 3.5615E-07 | 0.000619 | -4E+07 | 0.456145 |
| | +/- % | | 17% | 4% | 1% | -10% | 2% | 16.0% | 17% | 2% | -10% | -1% | -1% | 17% | 2% | 22% | -8% | 2% |
| *DRCongo.Region2.CrimeModule.Criminal Justice Regional Module.Violent Crime Module.InitialMR* | **H** | 32.63 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2267.8067 | 15.213238 | 35139505.26 | 9E-06 | 10891230748 | 1.2E+10 | 125480 | 15.29690 | 2.4643E-05 | 0.003856 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | -6.3196 | 52.7978 | 0 | 0 | 0 | 288.2584629 | 15493.4 | 1.44569 | 0 | 3.1305E-06 | 0.001093 | -11.78 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | 2% | 0.0% | 0% | 0% | 0% | 0% | 0% | 0% | 15% | 40% | 0% | 0% |
| | **L** | 17.57 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2153.6029 | 15.213238 | 35139505.26 | 9E-06 | 10891244521 | 1.2E+10 | 125480 | 15.29690 | 1.7872E-05 | 0.002767 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | -336.27 | -61.4060 | 0 | 0 | 0 | 14061.35396 | 46482.4 | 2.13499 | 0 | -3.6408E-06 | 0.000004 | -655.5 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | -3% | 0.0% | 0% | 0% | 0% | 0% | 0% | 0% | -17% | 0% | 0% | 0% |
| *DRCongo.Region.FoodAvailability.DES_Per_KG* | **H** | 4.2817 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2215.0089 | 15.213238 | 35139505.26 | 9E-06 | 10459673962 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -431556497.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | 0.0% | 0% | 0% | -4% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | **L** | 2.3055 | 13.19673659 | 51.30083866 | 1.57006E-05 | 2.4E+08 | 2215.0089 | 15.213238 | 35139505.26 | 9E-06 | 11692692527 | 1.2E+10 | 125478 | 15.29690 | 2.1513E-05 | 0.002763 | 5E+08 | 26.316879 |
| | +/- | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 801462066.9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | +/- % | | 0% | 0% | 0% | 0% | 0% | 0.0% | 0% | 0% | 7% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |

A baseline run provides default results for all of the outputs. From there, each input is varied to a higher or lower value (with the value labeled as H or L) and a new simulation is run with just that input modified. Each new sampling run is compared with the baseline, and the deviation of each output is indicated and labeled as +/-. The deviation is also indicated as a percentage of the baseline and labeled as +/- %. These terms represent the main effect, and the effect can be linear or piecewise linear (different between high and low settings). Positive main effects are highlighted green, while negative main effects are highlighted red. Stronger effects are more opaque (this is relative to the other outputs for a specific input). One of the more interesting findings is the connection between the Region 3 Agricultural wages and the murder rate. A 30% decrease in agricultural wages (from the 520 to 364) results in a substantial increase in the murder rate, about 43% nationally. A 30% increase in the agricultural wages, though, will lower the national murder rate by 12%. This indicates that wages alone can control the nation's murder rate, but creating an effective policy to control agricultural wages may still be a great challenge.

The analysis of main effects can go a step further and determine likely input values to achieve a specific output value. Using the same DR Congo example with Agricultural Wages as the input and Murder Rate as the output, we can calculate a few weights for different ranges of the input and output space:

- Low/High : $W_{LH} = (I_H / I_L) / (O_H / O_L)$
- Low/Baseline: $W_{LB} = (I_B / I_L) / (O_B / O_L)$
- Baseline/High: $W_{BH} = (I_H / I_B) / (O_H / O_B)$
- Average Weight (of above options)

These weights can now be used in a linear combination with one of the known input/output pairings to predict input values for a desired output. For example, we can start from the low input value (364) and the corresponding output response (3166.49) and use the above weights to determine a likely input setting to achieve a different response, such as 2595.6, with a linear function:

$$I_x = I_L + (I_L * W_k * O_\Delta / O_L) \qquad (11)$$

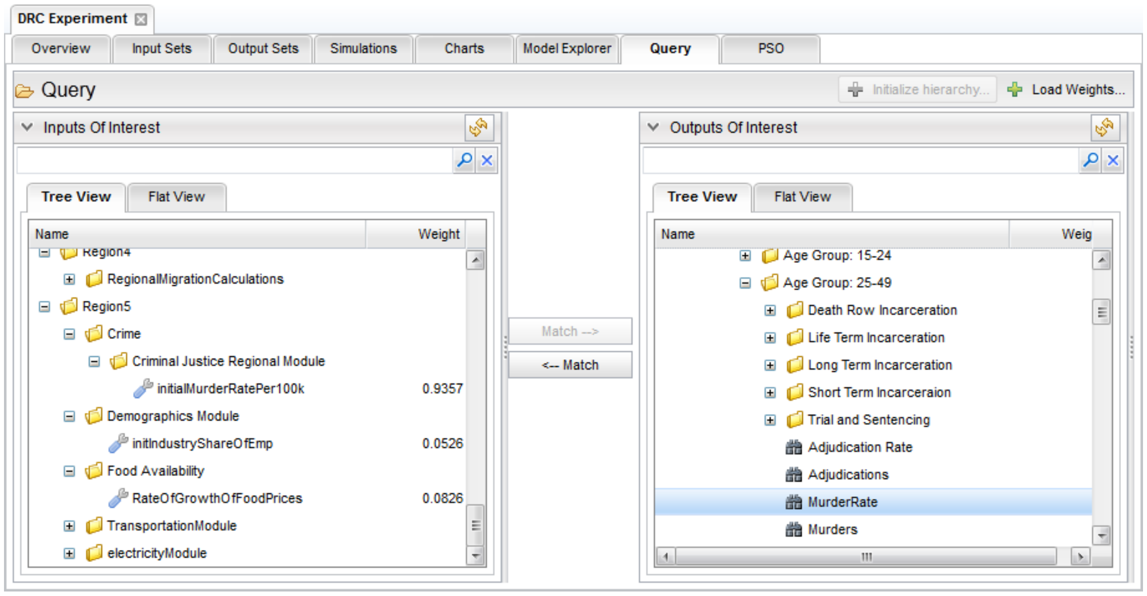
A set of output predictions featuring a combination of all weights is provided for the three different starting points in *Table 2*. This first column identifies some midpoints between known response values to try and predict input settings for. The second column is a basic linear interpolation of the input. The three remaining sections indicate predictions with the four different weights starting with the low point, baseline, and high point, respectively. The final prediction indicates which values are closest to a linear response by comparing all calculations to the second column.

Table 2: Weighted Predictions for Agricultural Wages

| Output | Input | LH L | BL L | HB L | AVG L | LH B | BL B | HB B | AVG B | LH H | BL H | HB H | AVG H | **Prediction** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3166.49 | 364 | 364 | 364 | 364 | 364 | -151.3 | 63.8 | 191.3 | 34.6 | -580.0 | -177.5 | 61.0 | -232.2 | 364 |
| 2976.19 | *395.2* | 429.7 | 408.7 | 396.2 | 411.5 | -17.0 | 155.1 | 257.0 | 131.7 | -382.5 | -43.3 | 157.7 | -89.3 | 396.2 |
| 2785.90 | *426.4* | 495.5 | 453.3 | 428.4 | 459.1 | 117.2 | 246.3 | 322.8 | 228.8 | -184.9 | 91.0 | 254.5 | 53.5 | 428.4 |
| 2595.60 | *457.6* | 561.2 | 498.0 | 460.6 | 506.6 | 251.5 | 337.5 | 388.5 | 325.8 | 12.7 | 225.3 | 351.2 | 196.4 | 460.6 |
| 2405.31 | *488.8* | 627.0 | 542.7 | 492.8 | 554.1 | 385.7 | 428.8 | 454.3 | 422.9 | 210.3 | 359.5 | 448.0 | 339.2 | 492.8 |
| 2215.01 | 520 | 692.7 | 587.4 | 525.0 | 601.7 | 520 | 520 | 520 | 520 | 407.8 | 493.8 | 544.7 | 482.1 | 520 |
| 2163.36 | *551.2* | 710.6 | 599.5 | 533.7 | 614.6 | 556.4 | 544.8 | 537.8 | 546.4 | 461.5 | 530.2 | 571.0 | 520.9 | 546.4 |
| 2111.70 | *582.4* | 728.4 | 611.6 | 542.4 | 627.5 | 592.9 | 569.5 | 555.7 | 572.7 | 515.1 | 566.7 | 597.2 | 559.7 | 572.7 |
| 2060.05 | *613.6* | 746.2 | 623.8 | 551.2 | 640.4 | 629.3 | 594.3 | 573.5 | 599.1 | 568.7 | 603.1 | 623.5 | 598.4 | 623.5 |
| 2008.39 | *644.8* | 764.1 | 635.9 | 559.9 | 653.3 | 665.8 | 619.1 | 591.4 | 625.4 | 622.4 | 639.6 | 649.7 | 637.2 | 649.7 |
| 1956.74 | 676 | 781.9 | 648.0 | 568.6 | 666.2 | 702.2 | 643.8 | 609.2 | 651.8 | 676 | 676 | 676 | 676 | 676 |

Alternatively, the prediction could be based on maintaining the output below the specified value. In this case, we would want to know the worst case scenario where an input setting would still result in that output. Since the input and output are inversely correlated, the highest input setting calculated would be used in this case.

The sampling results are further used by GNOME to compare inputs (and outputs) against each other. For a particular output, all of the minimum and maximum deviations (+/- %) are calculated, per input, and used to create the weights displayed within the query and associate UI tools. *Figure 21* illustrates how effects of inputs can be compared in the Query tool by selecting an output in the right-hand display (Region 3, 25-49 Murder Rate in this case).



Inputs Of Interest

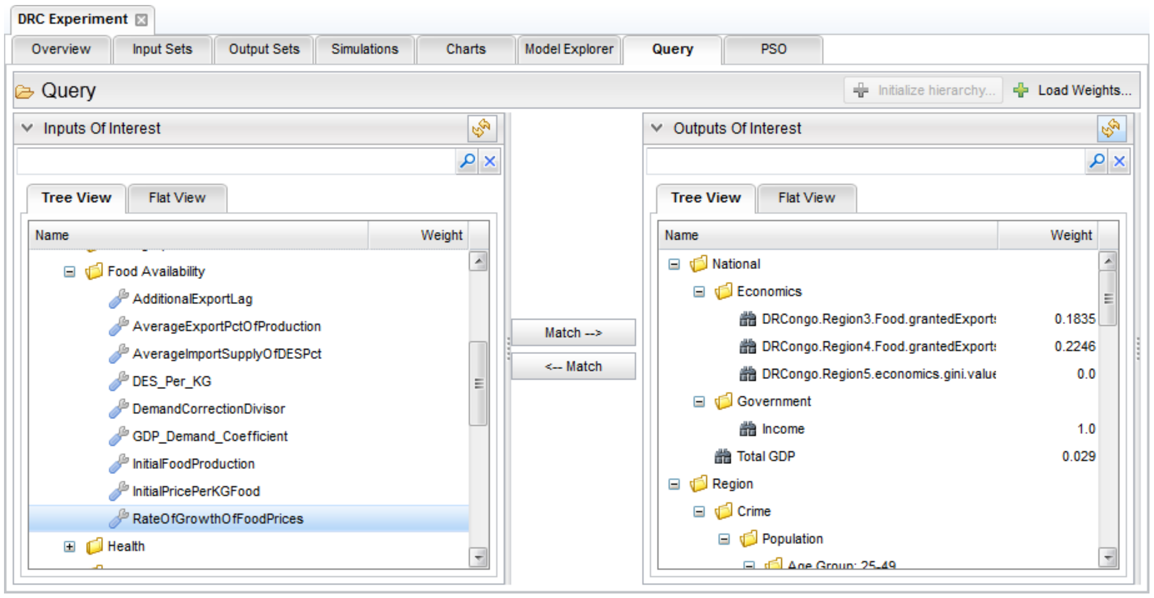
| Name | Weight |
|---|---|
| Region4 | |
| RegionalMigrationCalculations | |
| Region5 | |
| Crime | |
| Criminal Justice Regional Module | |
| initialMurderRatePer100k | 0.9357 |
| Demographics Module | |
| initIndustryShareOfEmp | 0.0526 |
| Food Availability | |
| RateOfGrowthOfFoodPrices | 0.0826 |
| TransportationModule | |
| electricityModule | |

Outputs Of Interest

| Name | Weig |
|---|---|
| Age Group: 15-24 | |
| Age Group: 25-49 | |
| Death Row Incarceration | |
| Life Term Incarceration | |
| Long Term Incarceration | |
| Short Term Incarceraion | |
| Trial and Sentencing | |
| Adjudication Rate | |
| Adjudications | |
| MurderRate | |
| Murders | |

**Figure 21: Comparing DR Congo Inputs for Murder Rate**

Likewise, the minimum and maximum output deviations caused by a particular input are calculated and compared to identify which outputs an input can have a stronger effect on. *Figure 22* illustrates how effects on outputs can be compared in the Query tool by selecting an input in the left-hand display (Rate of growth of food prices).

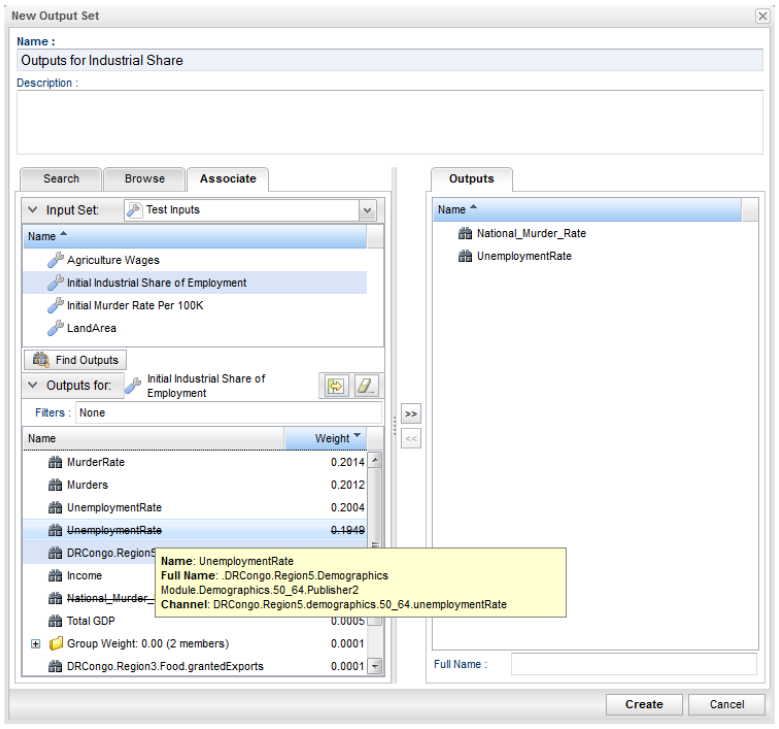


| Name | Weight |
|---|---|
| Food Availability | |
| AdditionalExportLag | |
| AverageExportPctOfProduction | |
| AverageImportSupplyOfDESPct | |
| DES_Per_KG | |
| DemandCorrectionDivisor | |
| GDP_Demand_Coefficient | |
| InitialFoodProduction | |
| InitialPricePerKGFood | |
| RateOfGrowthOfFoodPrices | |
| Health | |

| Name | Weight |
|---|---|
| National | |
| Economics | |
| DRCongo.Region3.Food.grantedExport | 0.1835 |
| DRCongo.Region4.Food.grantedExport | 0.2246 |
| DRCongo.Region5.economics.gini.value | 0.0 |
| Government | |
| Income | 1.0 |
| Total GDP | 0.029 |
| Region | |
| Crime | |
| Population | |
| Age Group: 25-49 | |

**Figure 22: Comparing DR Congo Outputs for Growth of Food Prices**

The "Associate tool" within the Input Set and Output Set editors also utilizes the calculated weights to compare and suggest inputs and outputs to use, respectively. *Figure 23* displays outputs and weights being compared for the input *Initial Industrial Share of Employment*. Here the murder rate, unemployment rate, and income are shown to be significantly affected.



Name : Outputs for Industrial Share

Description :

Input Set: Test Inputs

| Name |
|---|
| Agriculture Wages |
| Initial Industrial Share of Employment |
| Initial Murder Rate Per 100K |
| LandArea |

Outputs for: Initial Industrial Share of Employment

Filters : None

| Name | Weight |
|---|---|
| MurderRate | 0.2014 |
| Murders | 0.2012 |
| UnemploymentRate | 0.2004 |
| ~~UnemploymentRate~~ | ~~0.1949~~ |
| DRCongo.Region5 | |
| Income | |
| ~~National_Murder_~~ | |
| Total GDP | 0.0005 |
| Group Weight: 0.00 (2 members) | 0.0001 |
| DRCongo.Region3.Food.grantedExports | 0.0001 |

**Name**: UnemploymentRate
**Full Name**: .DRCongo.Region5.Demographics Module.Demographics.50_64.Publisher2
**Channel**: DRCongo.Region5.demographics.50_64.unemploymentRate

| Name |
|---|
| National_Murder_Rate |
| UnemploymentRate |

Full Name :

**Figure 23: Using Associate Tool to Compare Outputs**

## 4.2 Model Exploration

The following figures depict the typical flow of viewing a NOEM model at a high-level where we wish to understand the relationships between the various modules. For this example, a NOEM model depicting the country of Iceland was generated using NOEM's *MDE* (Model Definition Environment) tool with all regions of Iceland being represented as one large region. *Figure 24* illustrates the initial view of the Iceland model loaded in the Model Explorer.

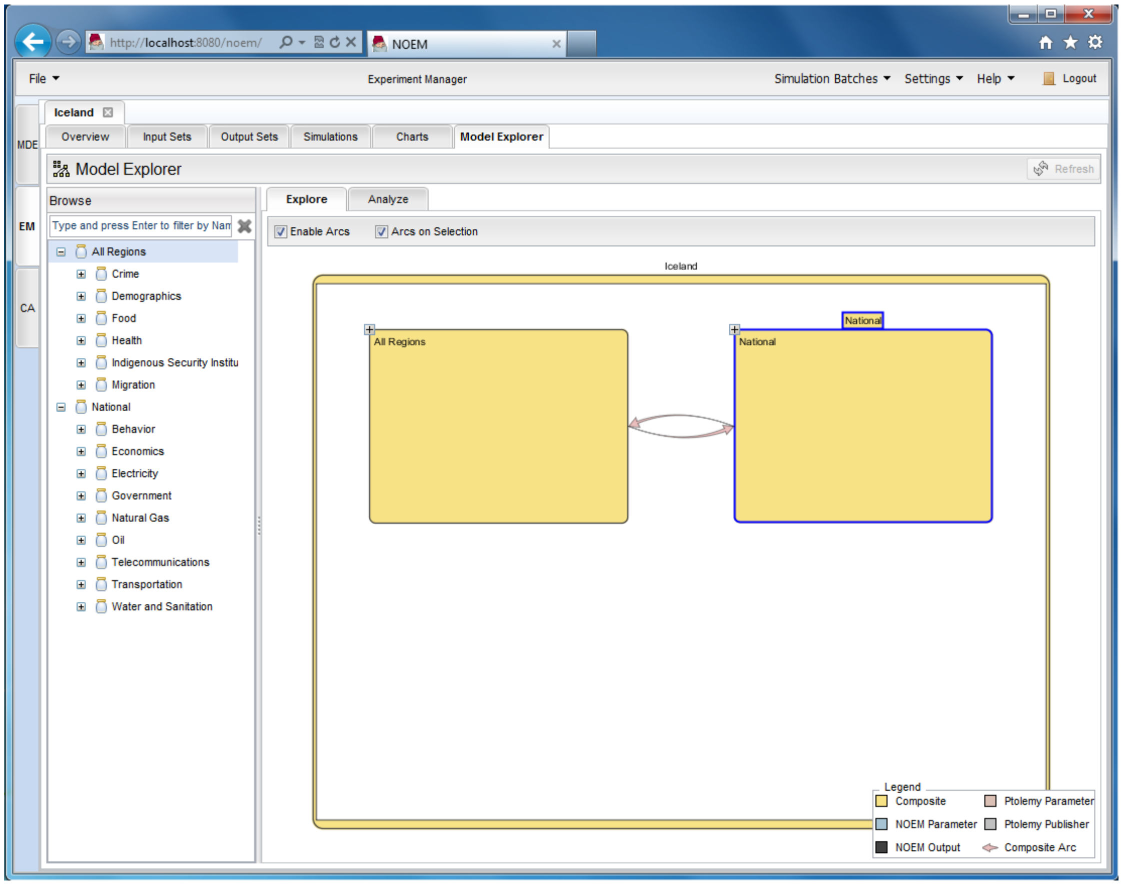


**Figure 24: Initial View of Iceland in the Model Explorer**

The above figure shows the *National* and *All Regions* nodes with two red composite arcs between them—representing a high-level relationship. The *Browse* tree in the left panel shows the hierarchy of nodes contained in the Iceland model. Double-clicking on the *Government* node under the *National* node in the *Browse* tree will cause the display to expand the *National* node and "zoom" into the *Government* node, as depicted in *Figure 25*.

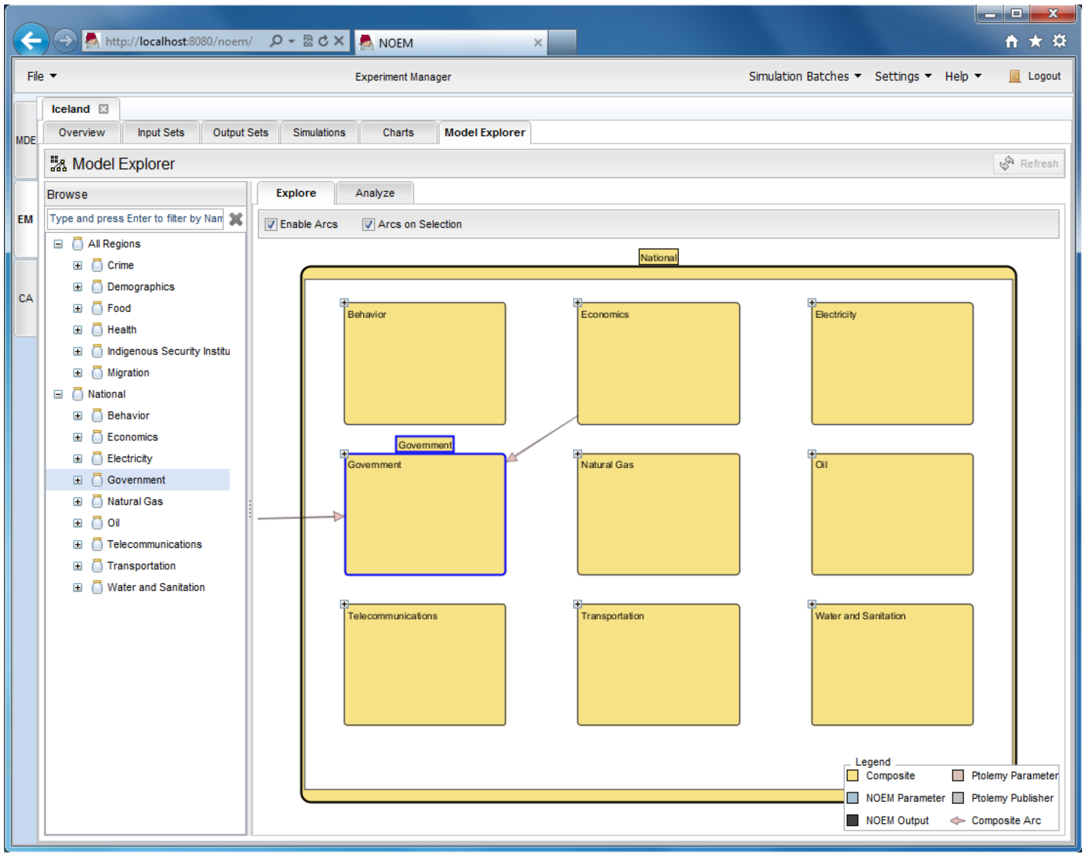


**Figure 25: Navigating to the Government Node from the Browse Tree**

Upon further inspection of the above image we see that there is a composite arc connecting the *Economics* node to the *Government* node. As shown in *Figure 26,* double-clicking on the *Economics* node reveals that there is a direct relationship between a node labeled "TotalGDPP" and the *Government* module. In this particular case, "TotalGDPP" represents a Ptolemy Publisher (shown by the gray-colored icon in the legend) which is a Ptolemy actor inside the NOEM Iceland model that publishes "Total GDP" data to the *Government* module.

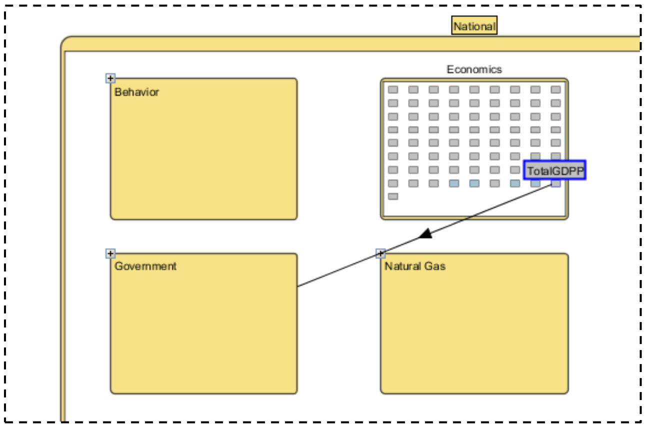


**Figure 26: Expanding the Economics Module**

Scrolling the mouse-wheel down allows us to "zoom out" the graph visualization to see the other composite arc connected to the *Government* module. *Figure 27* illustrates the change in visualization when you zoom out from the *Government* module and double-click on the *All Regions* node to expand it.

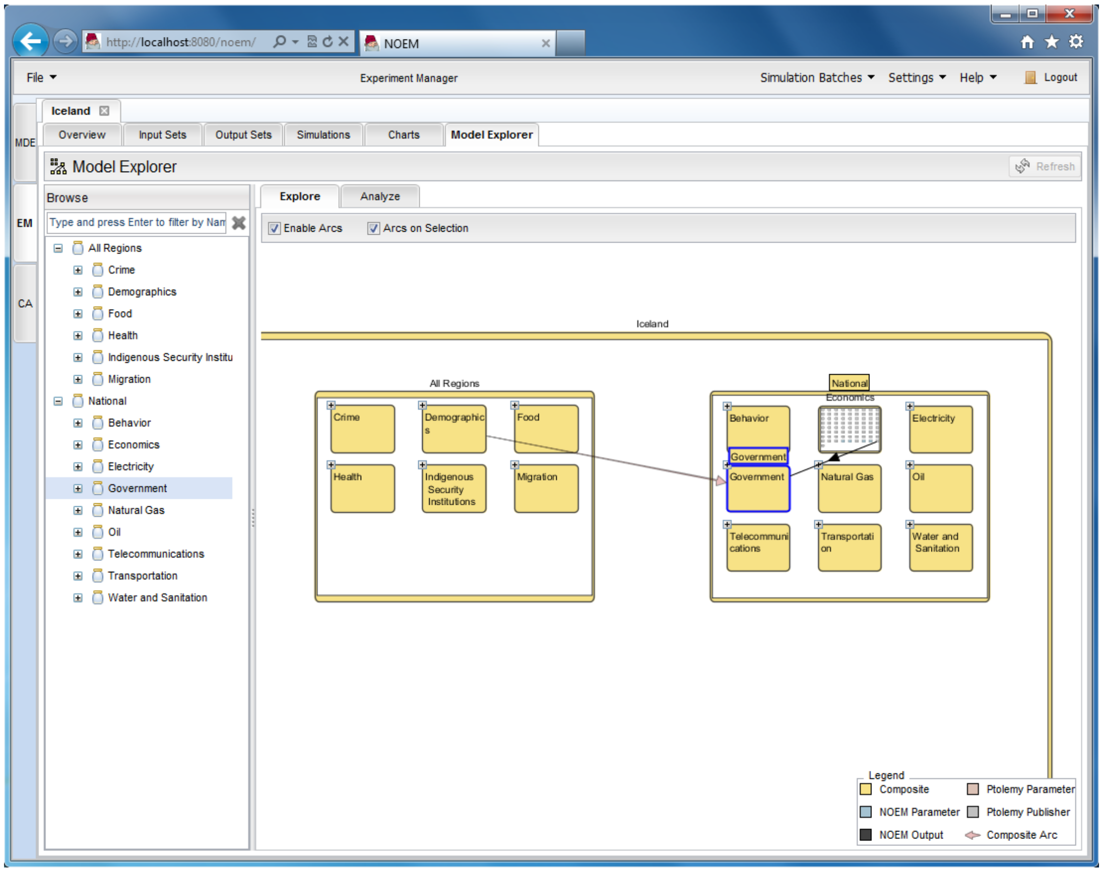


**Figure 27: Scrolling the Mouse-Wheel to Zoom Out**

*Figure 28* illustrates the effect of double-clicking on the *Demographics* module--which reveals a direct connection between a Ptolemy Publisher node called "UnempRateP" (which typically signifies Unemployment Rates) and the *Government* module.

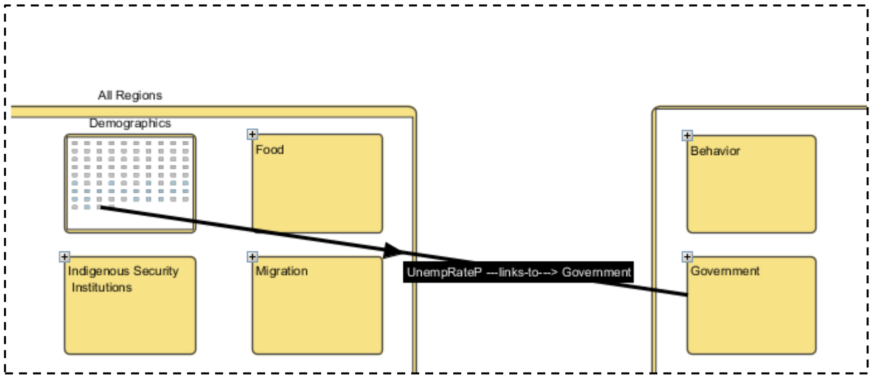

All Regions
Demographics
Food
Indigenous Security Institutions
Migration
UnempRateP ---links-to---> Government
Behavior
Government

**Figure 28: Expanding the Demographics Module**

## 4.3 Software and Source Code

The Software applications, components, and routines developed under GNOME II include the following:

- **GNOME UI:** A web-client user interface package used within NOEM. This contains all the visualization components and services created for use in NOEM, which includes the model explorer, query/associate capabilities, and hierarchy structures.
- **GNOME Model:** A java project containing all of the graph and model-based components for GNOME. Includes routines to:
    - Convert a Ptolemy MOML file into a strict graph-based representation,
    - Support the structural views of the model explorer, and
    - Perform reach and weighting algorithms with the graph structure.
- **Sampling:** A project package used to run the Experimental Sampling and Weight Normalization. Currently utilized by GNOME, but also suitable for PSO and TRACER. The package is kept within the PSO project due to the similarity in references/dependencies.

# 5 Conclusions

The results of GNOME provide analysts and decision makers with an awareness of the key impacts and consequences of pursuing a policy. The components developed through GNOME have provided a great deal of insight into nation building, and they have aided in the use and analysis of current Nation Building models.

Since the models and model data for NOEM is stored using the Ptolemy simulation environment, the original intent of GNOME was to provide a data structure to store the NOEM modules in a clearly defined, neutral graphical form that can be used separate from Ptolemy. While Ptolemy offers a strong actor-oriented simulation environment with a robust synchronous data flow (SDF) domain, it lacked a HCI component to visualize the relationships between the actors--a crucial component of understanding how each NOEM module interacts with one another in the overall model.

To address the issue of visualizing model relationships, the Model Explorer was first introduced as an Eclipse RCP plug-in to render the graphical form of the data that GNOME elicited from the NOEM model. The latest iteration of the Model Explorer continued to satisfy the original goal to visualize model relationships by using standard web graphics for the thin-client version of NOEM. As mentioned in this report, several software packages were considered in order to implement the graph visualization component to run on a standard web browser. Ultimately, an open-source JavaScript library that used vector graphics was chosen due to its wide browser support and its independence from 3rd party software. Additional improvements to the Model Explorer included integrating SHriMP (Simple Hierarchical Multi-Perspective) to visualize nested/hierarchical graphs and the "Analysis" pane to help users isolate certain graph nodes to better understand their relationships in large models.

Understanding and comparing relationships between more specific sets of model inputs and outputs was further explored by establishing a methodology and a set of associated interface components to weight these relationships. Experimental sampling provided a base set of main effects between individual model inputs and the outputs of interest. Comparing the main effects allowed for a series of inputs to be weighted or ranked against each other terms of their effect on a particular output. Similarly, comparing the percentage-deviation of main effects allowed for a series of outputs to be weighted and ranked against each other for a particular input. Although this analysis is limited by the assumption of linearity in the model, it has provided a means to identify key inputs and outputs of interest without the need to perform a full-factorial simulation analysis.

## 6 References

C. Brooks, E. L. (2008). *Heterogeneous Concurrent Modeling and Design in Java.* Electrical Engineering and Computer Sciences, University of California Berkeley.

Dobbins, J. (2005). Nation-Building: UN Surpasses U.S. on Learning Curve. *RAND Review , 29* (1), pp. 24-29.

Goldsmith, J.  (2010).  *Analysis and Optimization Methods of Graph Based Meta-Models for Data Flow Simulation.*  Rochester: Rochester Institute of Technology.

Kleijnen, J. (1975). Screening Designs for Poly-Factor Experimentation. *Technometrics , 17* (4), 487-493.

Lee, E. A. (2003). *Overview of the Ptolemy Project.* Berkely: University of California Berkely.

Montgomery, D. (1979). *Methods for Factor Screening in Computer Simulation Experiments.* Atlanta: Georgia Institute of Technology.

Richardson, D. (2004). *Modeling and Analysis of Post-Conflict Reconstruction.* M.S. Thesis, United States Air Force Institute of Technology.

Robbins, M. J. (2005). *Investigating the Complexities of Nation Building: A Sub-National Regional.* M.S. Thesis, United States Air Force Institute of Technology.

Tauer, G. (2009). *A Graph-Based Factor Screening Method for Synchronous Data Flow Simulation Models.* Rochester: Rochester Institute of Technology.

Wu, J. & Storey, M. D. (2000). *A Multi-Perspective Software Visualization Environment*. University of Victoria, Victoria BC, Canada.

# 7 List of Acronyms

| | |
|---|---|
| GNOME | Graph and Network Objects for Model Elicitation |
| HCI | Human Computer Interface |
| JUNG | Java Universal Network and Graph package |
| LP | Linear Program |
| MDE | Model Definition Environment |
| NOEM | National Operational Environment Model |
| PSO | Policy Set Optimization |
| RCP | Rich Client Platform |
| RPC | Remote Procedure Call |
| SDF | Synchronous Data Flow |
| SHriMP | Simple Hierarchical Multi-Perspective |