SAND REPORT

SAND 2012-1117
Unlimited Release
Printed February 2012

Complex Adaptive Systems of Systems Engineering Environment Version 1.0

John M. Linebarger, Richard J. Detry, Robert J. Glass, Walter E. Beyeler, Arlo L. Ames, Patrick D. Finley, S. Louise Maffitt

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND 2011-1117

Unlimited Release
Printed February 2011

COMPLEX ADAPTIVE SYSTEMS OF SYSTEMS (CASOS) ENGINEERING ENVIRONMENT

John M. Linebarger, Systems Research, Analysis, & Applications

Richard J. Detry, Operations Research & Knowledge Systems

Robert J. Glass, Systems Research, Analysis, & Applications

Walter E. Beyeler, Policy and Decision Analytics

Arlo L. Ames, Infrastructure Modeling and Analysis

Patrick D. Finley, Operations Research and Knowledge Systems

S. Louise Maffitt, Systems Research, Analysis, & Applications

Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0826

Abstract

Complex Adaptive Systems of Systems, or *CASoS*, are vastly complex *physical-socio-technical systems* which we must understand to design a secure future for the nation. The Phoenix initiative implements CASoS Engineering principles combining the bottom up Complex Systems and Complex Adaptive Systems view with the top down Systems Engineering and System-of-Systems view. CASoS Engineering theory and practice must be conducted together to develop a discipline that is grounded in reality, extends our understanding of how CASoS behave and allows us to better control the outcomes. The pull of applications (real world problems) is critical to this effort, as is the articulation of a CASoS Engineering Framework that grounds an engineering approach in the theory of complex adaptive systems of systems. Successful application of the CASoS Engineering Framework requires modeling, simulation and analysis (MS&A) capabilities and the cultivation of a CASoS Engineering Community of Practice through knowledge sharing and facilitation. The CASoS Engineering Environment, itself a complex adaptive system of systems, constitutes the two platforms that provide these capabilities.

PREFACE

This living manuscript is one of four documenting the evolving capability of CASoS Engineering from its beginnings within multiple programs at Sandia National Laboratories. It summarizes the Engineering Environment in which development (modeling and simulation) capabilities and knowledge facilitation take place; the others provide greater detail on the history and guiding principles of Phoenix, CASoS Engineering Applications, and the theoretical Framework driving our efforts to engineer change within complex adaptive systems of systems. Additional living manuscripts will be added to this list in coming years.

At the time of this printing, people who have contributed to this effort are listed as co-authors; they have taken leadership roles and are encouraging others to join the growing Phoenix effort. People who join the project contribute directly to this and other manuscripts: thus, as with all living documents, some sections are slightly out of sync with others, there is some redundancy, and content is presented in a variety of styles. We accept all of these idiosyncrasies for the benefit of increased ownership.

Periodically, more concise documentation of Phoenix and its projects will be distilled, such as *Complex Adaptive Systems of Systems (CASoS) Engineering: Mapping Aspirations to Problem Solutions*, written for the New England Complex Systems Institute's 8th International Conference on Complex Systems, and also presented as the keynote at the 6th IEEE International Conference on Systems of Systems Engineering, both in June 2011.

ACKNOWLEDGEMENTS

Following the favorable reception of the 2008 *Sandia National Laboratories A Roadmap for the Complex Adaptive Systems of Systems (CASoS) Engineering Initiative*, initial funding was provided in November 2008 through Sandia's Laboratory Directed Research and Development (LDRD) from the Energy Resources and Nonproliferation (ERN, reconfigured to be ECIS or Environment, Climate and Infrastructure Security in 2010) Strategic Management Unit (SMU) to develop a pilot for the initiative, Phoenix, in context of analysis for the Global Energy System. Reported on in *A General Engineering Framework for the Definition, Design, Testing and Actualization of Solutions within Complex Adaptive Systems of Systems (CASoS) with Application to the Global Energy System (GES)*, this initial development has continued to evolve with additional contributions from Sandia LDRD within both ERN-ECIS and Homeland Security and Defense (HSD, reconfigured to be IHNS or International, Homeland and Nuclear Security in 2010) and from projects funded by a wide range of institutions:

- National Infrastructure Simulation and Analysis Center (NISAC), Department of Homeland Security **(**DHS**)**
- Science and Technology Division (S&T), DHS
- Public Health & Environmental Hazards (OPHEH), Veterans Health **Administration** (VHA), Department of Veterans Affairs (DVA)
- Center for Tobacco Products (CTP), U.S. Food and Drug Administration (FDA) Department of Health and Human Services (HHS)
- Department of Defense (DOD)

- Air Force Office of Scientific Research (AFOSR), DOD
- Office of the Secretary of Defense (OSD), Human Social Culture Behavior Modeling (HSCB) Program, DOD
- Center for International Security and Cooperation (CISAC), Stanford University
- New Mexico Small Business Administration (NMSBA), New Mexico Livestock Board (NMLB)

This work benefited greatly from the support of the following individuals within Sandia National Laboratories administration: Les Shephard (retired, previously 6000), Steve Roehrig (retired, previously 6300), Margie Tatro (6100), Rush Robinett (6110), Richard Griffith (6130), Pablo Garcia (6920), and Steve Kleban (6132). While members of Phoenix have come and gone, all played critical roles in the Initiative's development.

The development of the CASoS Engineering Framework required the efforts of the many people and significant stretching from traditional work processes, as well as significant time and resource commitments in order to frame and begin to test the ideas represented herein.

TABLE OF CONTENTS

FIGURES

1. INTRODUCTION TO THE COMPLEX ADAPTIVE SYSTEMS OF SYSTEMS ENGINEERING INITIATIVE

The concept of CASoS Engineering is described in the CASoS Engineering Initiative Roadmap[1] which was developed in 2007-2008 at Sandia National Laboratories to find ways to understand and solve the world's greatest problems. The Roadmap defines CASoS, CASoS Engineering, and the process for building the discipline of CASoS Engineering. The classical linear compartmentalized progression from Research (R) to Development (D) to Application (A) is replaced with an intrinsically integrated R&D&A process that develops CASoS Engineering theory and principles in context of solving high-impact (national and global) problems. The theoretical approach for CASoS Engineering outlined in the Roadmap emphasizes the importance of treating the Initiative itself as a CASoS and an object of CASoS Engineering. This means that, as we proceed, we must apply the developing CASoS Engineering principles reflexively to the Initiative's organization, development and growth.

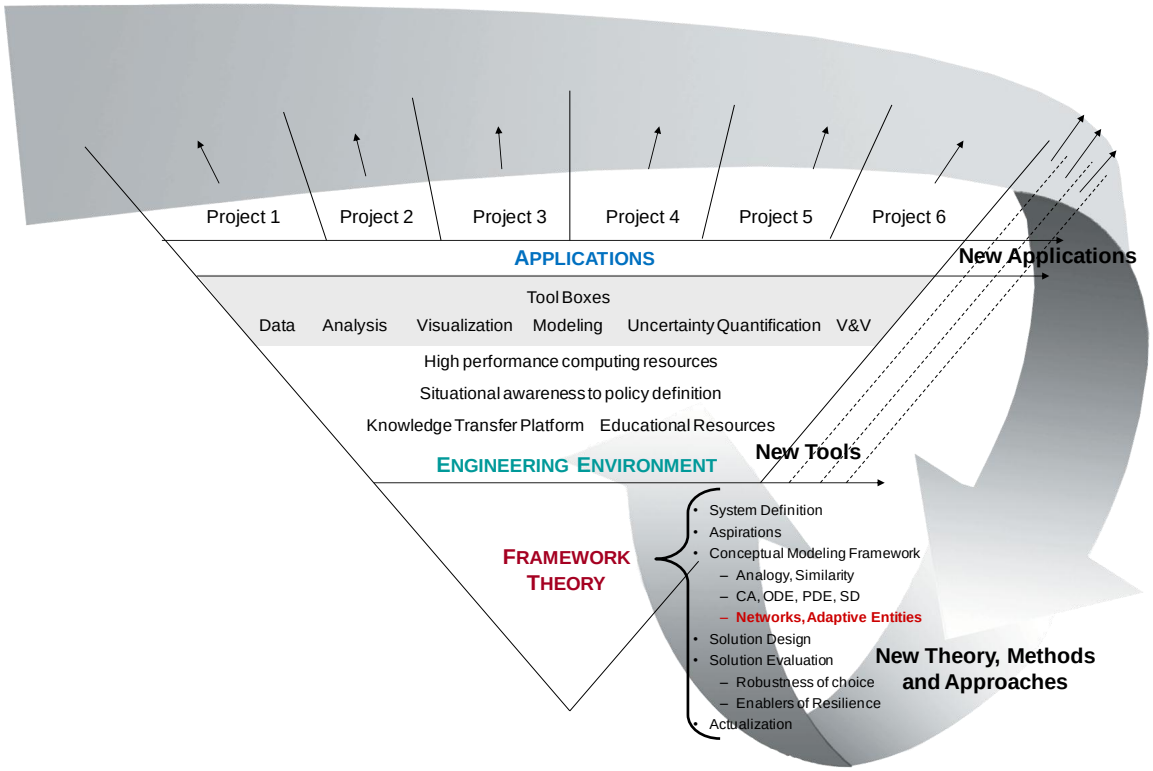


Figure 1, Integrated Research, Development, and Applications Structure

As illustrated in **Figure 1**, the three areas of study are fundamentally integrated:

[1] Glass RJ, Ames AL, Stubblefield WA, Conrad SH, Maffitt SL, Malczynski LA, Wilson DG, Backus GA, Ehlen MA, Engi D, Engineering Solutions in an Interdependent World: The CASoS Roadmap, Sandia National Laboratories, February 2008. http://www.sandia.gov/CasosEngineering/docs/CASoSEngineeringRoadmap_09.22.08.pdf

- The **CASoS Engineering Framework**[2] represents the Research that evolves the science of CASoS engineering. The Framework systematizes the theory & practice of CASoS Engineering across wide ranging disciplines and with wide ranging aspirations for affecting CASoS behavior.
- The **CASoS Engineering Environment** is the Development that supports the CASoS Engineering Framework by providing:
  - A modeling, simulation and analysis platform in which modular computational tools can be assembled in many ways and for many purposes
  - A knowledge facilitation platform for the capture, integration and evolution of the theory and practice of CASoS engineering, providing for the education and training of newly emerging CASoS Engineers
- **CASoS Engineering Applications**[3] are the set of high-impact Applications chosen to drive research and development. Application choices are based on the specific problem, system orientation and collaborations. The choice, sequence, and integration of applications are critical to the success of the CASoS Engineering Initiative. They must be adequately funded and foster the logical progression of research and development.

The CASoS Engineering Environment is emerging as new Applications are funded and existing Applications are expanded and evolve, which can trigger new developments in both the Framework and the Environment to address Application needs. This is a living document which will be updated as needed to address new capabilities in the Environment and changes in the Engineering Framework or Applications that influence the theory and practice for the Environment.

[2] Ames AL, Glass RJ, Brown TJ, Linebarger JM, Beyeler WE, Finley PD, Moore TW, Complex Adaptive System of Systems (CASoS) Engineering Framework Version 1.0, Sandia National Laboratories (SAND 2011-8793), November 2011.

[3] Brown TJ, Glass RJ, Beyeler WE, Ames AL, Linebarger JM, Maffitt SL, Complex Adaptive System of Systems(CASoS) Engineering Applications: Version 1.0, Sandia National Laboratories Report, SAND 2011-8032, October 2011.

2. CONTEXT

The CASoS Engineering Environment supports all aspects of the CASoS Engineering effort by providing integrated platforms for modeling, simulation, analysis, education, training, and collaboration. Computational and human resources are brought together through Framework principles and techniques, enabled by the Environment, to design, test, and implement solutions to CASoS problems.

The CASoS Engineering Environment (diagrammed in **Figure 2)** illustrates the overlapping platforms for Conceptualization, Analysis, and Design, Modeling and Simulation, and Knowledge Storage and Transfer.

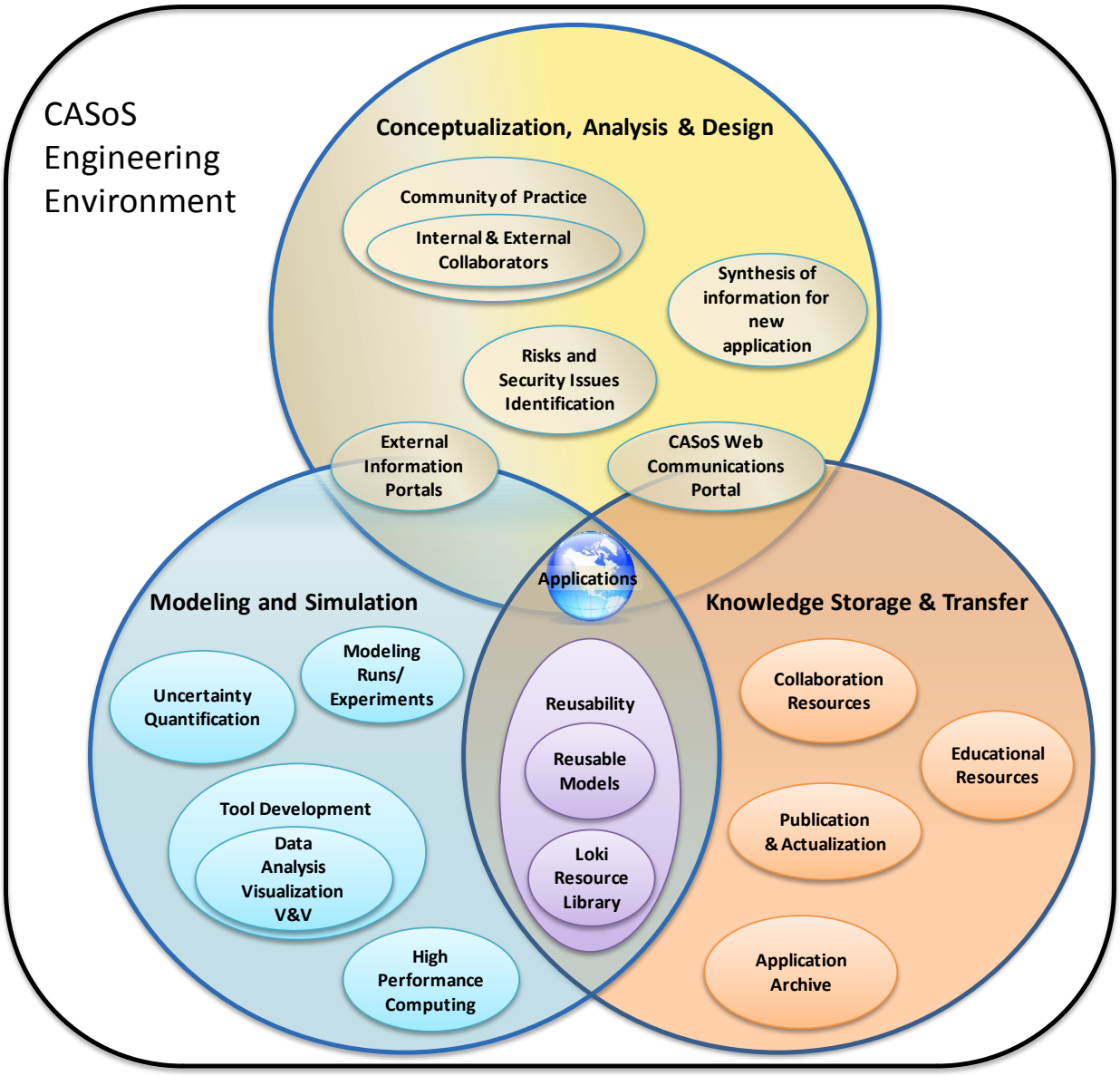


Figure 2. CASoS Engineering Environment Integrated Platforms

Conceptualization, Analysis, and Design (upper circle) focuses on CASoS Engineers and analytical processes. A community of practice is developed in which CASoS can be defined, researched, and analyzed. The process of CASoS Engineering, as described by the Framework, is implemented here. The Modeling and Simulation Environment (lower left circle) forms the cornerstone of computational capability required to implement the analysis process. Knowledge Storage and Transfer (lower right) organizes the information gained from analyses, captures the wisdom gained from Applications, and formulates it in ways that can be relevant to other problems. All elements of the Environment are connected through Applications which provide the opportunity to implement solutions and drive capability development.

This CASoS Engineering Environment supports the community of practice, communications, modeling resources, uncertainty quantification, and data analysis and visualization tools. Other modeling, simulation, visualization, and analysis capabilities, both internal and external to Sandia, continue to be explored for potential integration. The CASoS Engineering external, open website currently provides access to the growing body of theory and application from across Phoenix projects and the community. It is continuously evolving as an outreach mechanism, enabling us to more broadly engage with others working in complex adaptive systems-of-systems and related fields. Other components in active development include support for literature search, archive, retrieval, and exchange, both internally and with external partners, and the creation of a technology-enhanced collaboration space that facilitates group brainstorming and design activities and seamlessly captures the results.

The development of the CASoS Engineering Environment tracks the development of the CASoS Engineering Framework, and enables the development of applications using the Framework. The Environment instantiates core principles and processes defined by the Framework, supports the creation of applications that are guided by the Framework, and captures the knowledge and experiences gained through the application of the Framework for potential reuse.

The overall goals of the CASoS Engineering Environment are several-fold:

- To support the CASoS Engineering Framework with a computational environment that embodies its principles and techniques
- To facilitate computationally a multiple-model approach to modeling CASoS problems, either as independent perspectives of the same problem or as integrated (“docked”) models at different scales
- To support the creation of conceptual and computational models of CASoS problems
- To facilitate the reuse of knowledge, models, and code from previous CASoS efforts
- To create and support a robust software engineering process for the development and reuse of CASoS computational models
- To support a growing set of computational model categories (*e.g.,* Contagion, Exchange) with core modeling and implementation components.
- To foster the development of a CASoS Engineering community of practice, and provide computational support for all of its critical activities, whether directly related to model development or not
- To support the education and training of new CASoS engineers, both inside and outside of Sandia

- To enhance the development of the discipline of CASoS Engineering
- Recognizing that the CASoS Engineering Environment is itself a complex adaptive system, to strike the right balance between expansion (exploration) and convergence (realization) in the evolution of the capabilities of the Environment.

The current MS&A Platform is primarily composed of the Loki Network Modeling Resource Library, and is specialized for the analysis and visualization of agent-based network simulations. Other modeling, simulation, visualization, and analysis capabilities, both internal and external to Sandia, will continue to be explored for potential integration into the Environment.

Key goals of the MS&A Platform include:

- To create an Implementation Environment that facilitates the development, testing, simulation, and maintenance of CASoS models; at a minimum such an environment should consist of a set of code libraries, a version control system, a bug tracking system, and a continuous build and testing system
- To support conceptual model formulation and model realization within an application context
- To provide constructs that allow a CASoS to be modeled in several ways, using multiple paradigms and representations, and at several scales and levels of abstraction
- To capture potentially-reusable computational artifacts and model templates into a well-documented and easily accessible software library
- To support and simplify the analysis and visualization of simulation results
- To enable model assumptions to be tested quickly and easily
- To support sensitivity analysis to understand the key elements and relationships
- To characterize and assess the importance of uncertainties
- To enable an analyst to use all of the above to design mitigations that are robust to uncertainty, practical to implement, and cost-effective.

The Knowledge Platform currently consists of a Wiki site and a document repository site; the Wiki site is used to capture technical information that supports model creation and the development of the MS&A Platform, while the document repository site is focused more on the definition and documentation of models and on training materials for the Phoenix effort itself. These two sites may be merged into a single team collaboration site in the near future. Additionally, an external, open website is being developed to more broadly engage with others working in CASoS Engineering and related fields. Other Knowledge Platform components in active development include support for literature search, archive, retrieval, and exchange, both internally and with external partners; and the creation of a technology-enhanced collaboration space that facilitates group brainstorming and design activities and seamlessly captures the results.

Key goals of the Knowledge Platform include:

- To capture, integrate, and grow the theory and practice of CASoS Engineering in general, and the use of the CASoS Engineering Framework in particular, by fostering the creation of a CASoS Engineering community of practice

- To support the geographically distributed CASoS Engineering community in their literature search, archive, and retrieval activities, as well as in their collaborative brainstorming and design activities
- To archive the models, applications, and results of previous CASoS activities, to support both the reproducibility of results and the reuse of models at multiple levels, from concept to code
- To design and deliver education and training to develop CASoS engineers and grow the CASoS Engineering effort at Sandia
- To create an education program for CASoS Engineering that would allow such expertise to be established outside of Sandia.

Making the CASoS Engineering Environment function requires the dedicated interaction of a number of people with distinct skills. The Environment is an organic, growing and evolving collection of capabilities that integrates projects and people through the CASoS Engineering Framework. CASoS Engineering Environment specialists are critical participants within each project. Their work is coordinated across the various projects and drives the development of methodologies and structures. In this way, the CASoS Engineering Environment evolves with the applications and with each new project.

This report describes the emerging CASoS Engineering Environment in terms of:

The Environment as and within a CASoS: how the CASoS Engineering Environment is a CASoS itself, how it connects to the CASoS Engineering Framework, and how it relates to the Phoenix initiative;

Modeling, Simulation, and Analysis Platform: the tools, processes, and methods currently provided by the environment and those that have been identified as necessary to effectively model, simulate, and analyze CASoS problems;

Education and Training Platform: the tools, processes, and methods that provide open and collaborative access to the captured experience of practitioners working with emergent CASoS modeling techniques and capabilities (both successes and failures);

Development Environment: the software development infrastructure, such as a version control system, an issue tracking system, and software development guidelines, that has been created in support of the Phoenix initiative; and

Future work: describes key strategic goals for the CASoS Engineering Environment.

2.1. The Environment as and within a CASoS

We are creating the CASoS Engineering Environment recognizing that it is and will always be a CASoS itself. The CASoS Engineering Environment is being created by implementing the CASoS Engineering Framework, which provides the steps to define the Environment as a CASoS; to declare its aspirations; and to design, test, and actualize solutions.

In addition to being a CASoS itself, the Engineering Environment is a major component of Phoenix, which is also a CASoS. illustrates the functional structure of Phoenix and the importance of the Environment within it. The Environment concretizes the applicable CASoS Engineering theories in

order to construct solutions to a problem within a CASoS application. The Environment itself is improved in a spiral manner by advances in CASoS engineering theory and by application development. As theory advances, the Environment strives to enable the use of the advancements within applications. As applications are developed, the capabilities and knowledge generated during the application development are driven back into the Environment. Thus, the Environment grows and matures in step with its own use to solve CASoS problems.

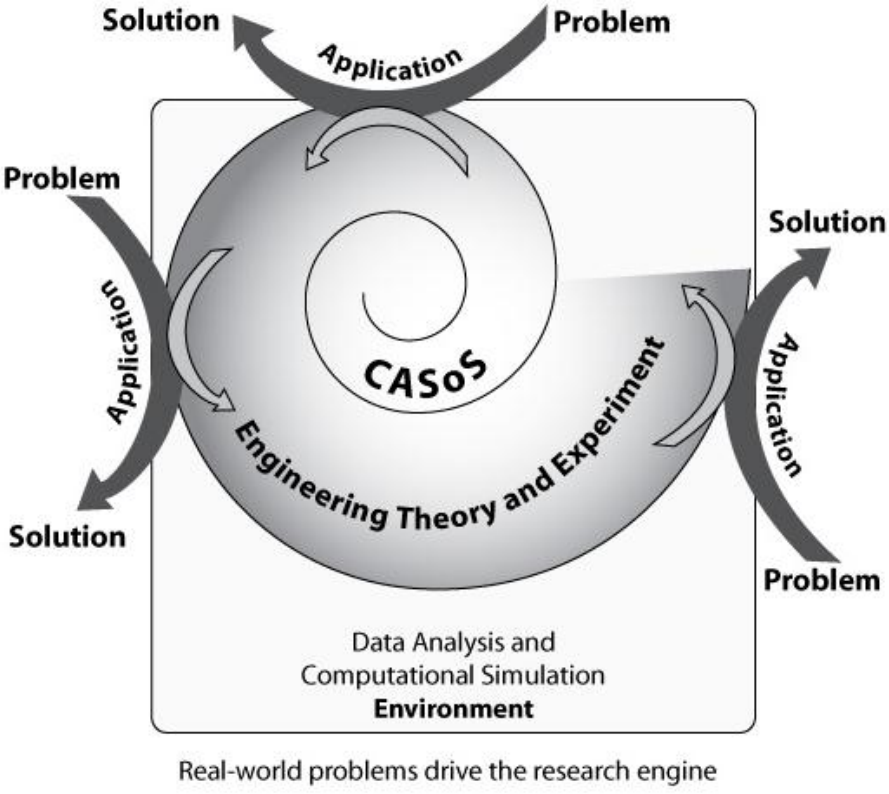


Figure 3. CASoS Engineering Initiative Development Structure

A major step defined within the CASoS Engineering Framework for moving from theory into practice is through the development of a conceptual model. The purpose of conceptual modeling is to develop a formal description of the system that allows us to derive the effects of actions on aspirations. The conceptual model is a description of the system, which can be used to infer the consequences of the interventions being considered. The knowledge facilitation platform within the CASoS Engineering Environment supports conceptual modeling by defining consistent content and formats for conceptual model descriptions and by providing a repository for the conceptual models and for capturing experiences gained during the development of the model. The modeling, simulation, and analysis platform enables the concretization of the conceptual model and the execution of the model to study the system and the effects of the proposed actions.

3. MODELING, SIMULATION, AND ANALYSIS PLATFORM

A wide variety of modeling, simulation, and analysis tools, processes, and methods are required to support the broad CASoS application space. We have identified an initial set of capabilities necessary to create solutions for the current and near-term CASoS applications and are designing and implementing a platform upon which CASoS applications can be constructed. The capabilities have been initially categorized as either modeling capabilities or simulation and analysis capabilities to reflect the different needs of model development and model execution and analysis. Modeling capabilities are software components that are used to construct the computational model from the conceptual model, while simulation and analysis capabilities are software components, tools, and other capabilities that simplify the execution of the computational model and the analysis of the results.

3.1. The Loki Network Modeling Resource Library

The current modeling, simulation, and analysis platform is based heavily on the Loki network modeling resource library. The Advanced Modeling Techniques Investigation (AMTI) project (funded by the U.S. Department of Homeland Security [DHS] through the National Infrastructure Simulation and Analysis Center [NISAC]) developed the Loki Network Modeling Resource Library to model and study CASoS relevant to critical infrastructure interdependencies. The Loki Resource Library embodies a generalized network and agent-based approach to modeling complex adaptive systems. It contains a set of components that can be selected, specialized, and combined to create models of diverse systems including power systems, pipelines, social networks, and financial systems, as well as interactions across these different networks and systems. Loki has been applied to generic congestive cascade, power grids, payment systems, social simulation, and infectious diseases as well as petrochemical and natural gas systems.

Basic resources such as database interaction, network construction, market exchanges, and definition of technologies as abstract chemical systems are available in the Loki library.

The library is composed of a set of Java classes for representing the structure and behavior of complex adaptive systems as networks and for exploring their response to stress, disruption, and/or novel conditions.

3.2. Modeling Structure and Dynamics

In building a conceptual model of CASoS we do not need a perfect or complete language suitable for all systems. Rather, we need a set of terms that are clearly defined, mutually consistent, and applicable to a meaningfully large set of cases. Towards this end, we have selected an entity/relationship approach to describe the structure of the system and we have defined several options for describing state dynamics, as illustrated in Figure 4. Entity/relationship formalisms have proven their usefulness over several decades of application in both database and software design. When a computational model is developed from the conceptual model, expressing the conceptual model in entity/relationship terms not only dovetails well with the definition of a complex system—a system whose behavior emerges from a network of (potentially simple) interactions—but it also eases translation into object-oriented languages.

Conceptual models describe the objects that compose the system (structure) and how they change over time (dynamics). The structure consists of entities and relationships among entities; in object-oriented terms, structure constitutes the object model or object map of the system. State variables are the pieces of information needed to define the condition of each entity and relationship in the model, and equations change the contents of the state variables. Correctly representing the system of state variables and equations is the essence of what is required to capture the behavior of a dynamic system. Dynamics describe changes to other elements of the conceptual model. Structural dynamics describe creation and destruction of entities and relationships; internal dynamics describe changes to state variables within an entity or relationship; and interaction dynamics describe changes to state variables based on state variables in other entities or relationships. Note that in practice, a static model is often created first, and then dynamics are added iteratively.

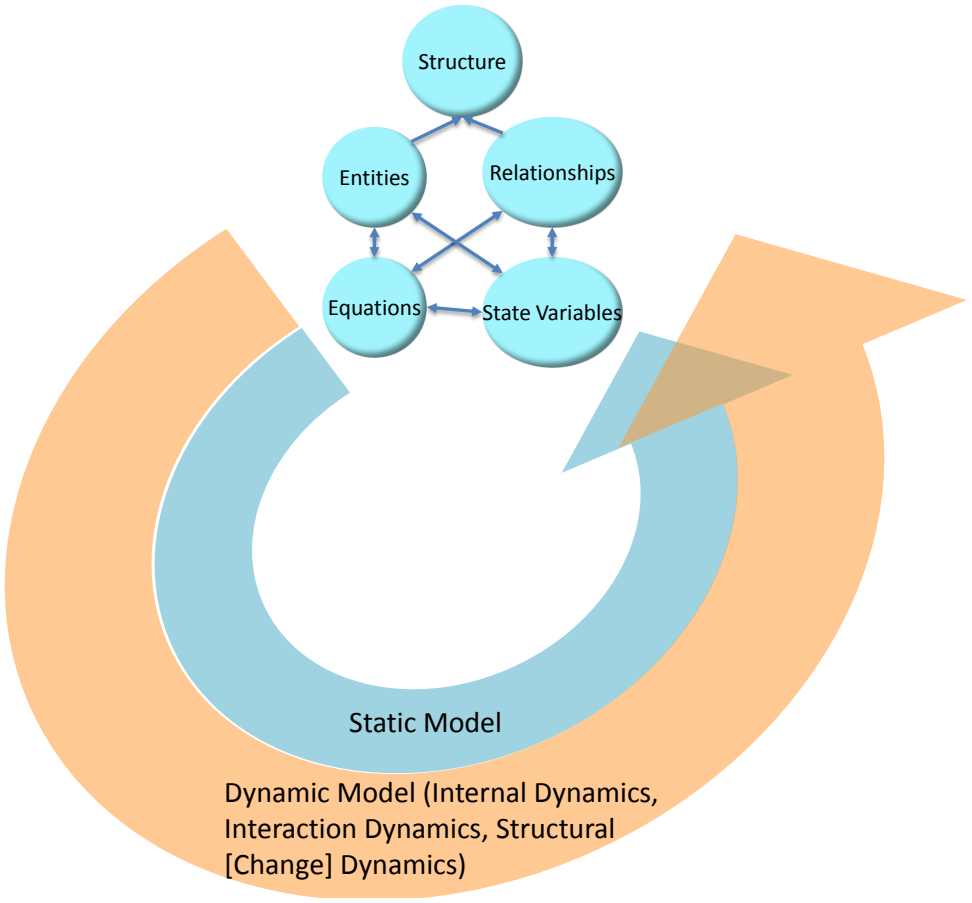


Figure 4 Conceptual Model Components and Development Flow

The structure and dynamics components have been refined into an initial set of entity/relationship capabilities needed to construct models for current and near-term CASoS applications, as shown in Figure 5. It is important to note that this is not an exhaustive set. Similar to the CASoS problems that they are being used to solve, the necessary capabilities will emerge and adapt over time as new types of applications arise. The capabilities shown in Figure 4 are briefly discussed below.

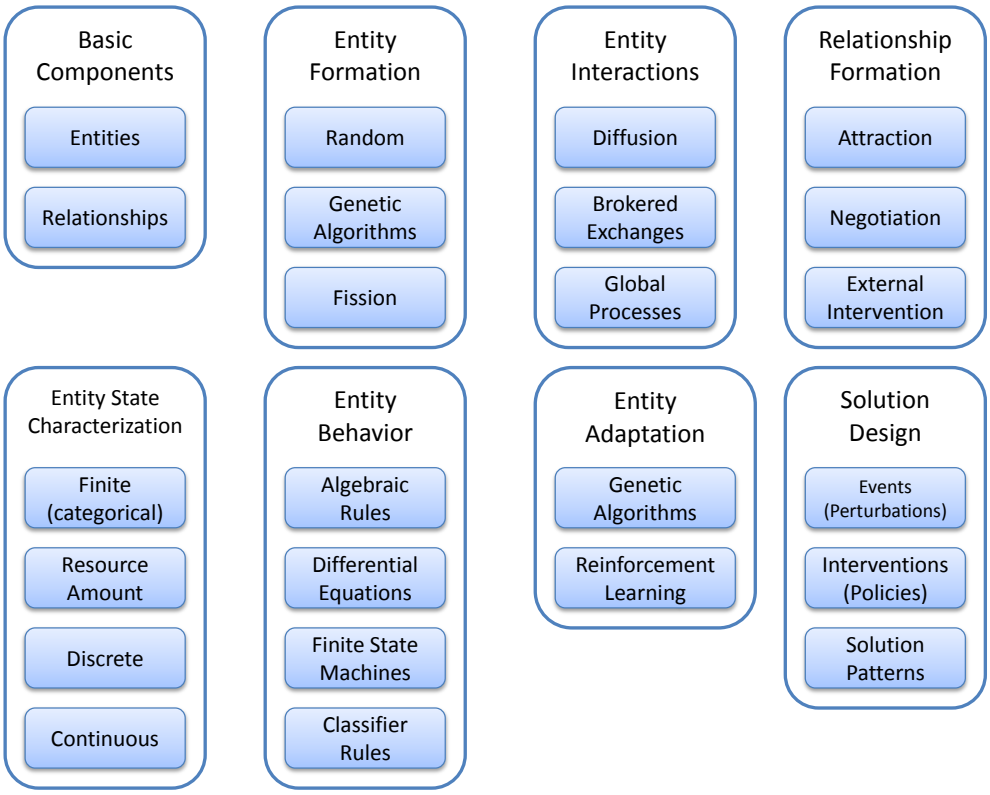


Figure 5. Modeling and Solution Design Capabilities

As mentioned above, the basic modeling components are entities and relationships. Entities are the objects in the system that get created, act and adapt, and form relationships with other entities. Entities can be individuals, facilities, institutions, or generally any kind of object that must be represented by the system. Entities adapt, behave, have a dynamic, descriptive state, interact, and form relationships with other entities for the purposes of exchanging resources. Entities and relationships can be created and destroyed at any point during a simulation.

The following sections discuss the modeling capabilities in more detail and provide examples taken from actual CASoS applications.

3.2.1. Entities and Entity State Characterization

Entities can initially be created and populated directly from system data, from statistical representations of the system or system data, or from other representations of the system. Entities have a state description, which is information that characterizes the state of an entity. Continuous and discrete states are very generic, while concepts such as resource amounts and categorical state variables are broadly applicable specializations.

For example, in the Global Financial System (GFS) model, entities include households, firms producing diverse goods and services, financial institutions, governments, and central banks. These entities use resource amounts, such as the amounts of diverse traded goods and services, tradable contracts, and utilities as part of the entity state description. Entities in the Veterans Affairs (VA) Threat Modeling project model include hospitals, diseases, treatments, supplies, patients, and staff.

Entities in the Critical Infrastructure Recovery model include substations, cell towers, wire centers, hospitals, and communities. These entities use some finite state descriptions, such as the functional conditions of the entities, along with resource amounts, such as inventories of spare parts, back up fuel, and critical materials, to describe entity state.

3.2.2. Entity Formation

Throughout a simulation, entities can be destroyed or formed. Some techniques for entity formation include random creation or replacement, the use of genetic algorithms to create new entities based on scored qualities of other entities, and entity fission or fusion.

3.2.3. Entity Behavior

Entity behavior is represented by the entity's state and by how the entity's state changes over time based on internal dynamics or on the dynamics of interactions with other entities via relationships. Algebraic rules, differential equations, finite state machines and classifier rules are a few examples of the techniques used to define entity behavior. In some cases, entities use just a single technique, while in other cases a mixture of techniques is used.

An example of an entity behavior expression in the Global Financial System model is the use of ordinary differential equations to formulate production processes as chemical reactions. The Critical Infrastructure Recovery model uses finite state machines to capture the operational status of various infrastructure components and backup systems, and finite state machines are also used in the Infectious Disease model to specify values for variables such as disease progression, response to immunization, and response to public information.

3.2.4. Entity Interactions

Interactions between entities are one form of exchange dynamics that define how entity or relationship state variable values are influenced by other entities in the model, generally by adjacent entities in a relationship network. Some of interaction patterns include diffusion, brokered exchanges, and global processes. With diffusion (also called contagion), some material or property spreads from entity to entity along network links. Two different kinds of diffusion or contagion dynamics are possible—infection dynamics or opinion dynamics. In a brokered exchange some material or property is routed between entities by a third entity (broker or router) according to a rule. A global process concurrently determines the condition of all entities based on their current properties. Non-brokered exchanges are also possible; the Exchange2 model uses a set of ordinary differential equations to govern production, consumption, and the exchange between firms in order to maintain health in a homeostatic fashion. Note that the exchange model as exemplified by Exchange2 conserves mass, while contagion models (whether for infection disease or for opinion) do not.

The Global Financial System model uses both brokered exchanges and diffusion to model entity interaction dynamics. Brokered exchanges are used to model markets for diverse goods and services, including final and intermediate goods and financial instruments. Diffusion is used to model the spread of risk tolerance among consumers and the regulatory constraints imposed by owning or controlling entities. Other models use diffusion to handle the spread of information and

perceptions among households, the spread of perceptions influencing mental state, and the spread of information used to make resource allocation decisions.

3.2.5. Entity Adaptation

Given that adaptation is a core CASoS concept, some capabilities to model entity adaptation are critical. Modeling adaptation amounts to using information about an entity's performance to feed back and change the structure or parameters of its behavioral model. Two techniques currently used are genetic algorithms and reinforcement learning. With genetic algorithms, parameters are determined through periodic recombination or mutation of scored entities, whereas reinforcement learning adjusts an entity's parameters over time, driven by episodic feedback from the environment.

An example of adaptation is the pricing market used within the Global Energy System model. In the pricing market, entities place bids to buy and sell commodities. The entities adapt by learning over time to adjust to market conditions in order to maximize profits. Another example of adaptation can be found within the Infectious Disease model. In this model, entities have a set of activities they perform every day. The entities adapt by learning to stay home, for example, when they sense that they can potentially get sick.

3.2.6. Entity Relationship Formation

Entities form relationships with other entities in the model in order to interact. New relationships can be formed in many ways, including attraction, negotiation, or external intervention. These are just representative relationship formation patterns; many other techniques for forming relationships are also possible.

In the Global Financial System model, relationships include formal and habitual business relationships, political and regulatory control, and ownership, while relationships in the Critical Infrastructure Recovery model include physical network elements and business connections. In the VA Threat Modeling project model, relationships include employment relations of medical and administrative staff; habitual use of clinics/hospitals by patients; and patient/provider relations determined by proximity.

3.2.7. Solution Design

The capabilities listed above apply primarily to the modeling process. The capabilities in the Solution Design category pertain to the identification and development of engineering solutions to the complex adaptive systems of systems problems being modeled. Events perturb the behavior of the system model, which is often in steady-state. Interventions (often in the form of policies) are special kinds of events that are introduced into the system in order to achieve engineering goals, such as enabling the system to be robust to a wide variety of perturbations. Solution patterns are collections of interventions that achieve a particular type of engineering goal, and as such serve as engineering idioms that can be applied to new problems of the same type.

3.2.8. Status of the Modeling Capabilities

Loki provides many of the capabilities discussed above, while others remain model-specific. The typical path for providing a capability is to first understand the detailed requirements of the model or models that need the capability, then attempt to find an existing solution within Loki, within another CASoS model, or from an external source. If an existing solution cannot be found, the capability is developed in order to meet the requirements of the model while also striving for the resulting capability, or at least the development and application patterns of the capability, to be useable by other models as well. New capabilities generally start life at the application level, then migrate to the toolkit when they prove to be more broadly applicable.

3.3. Simulation, Analysis, and Development Support Capabilities

The capabilities described in the above section are used to construct a computational model of a complex adaptive system from a conceptual model of the system. Once an initial computational model has been developed, it is used to run a number of simulations in order to study the system it represents. Results of the simulation are then used to refine the model, and the sequence is repeated. Performing all of the steps to define, configure, and run simulations and collect, analyze, and synthesize the simulation output can be time-consuming. A goal of our modeling, simulation and analysis platform is to reduce the complexity of these steps in order to allow the CASoS practitioner to focus more time and energy on developing and using a model rather than on the mechanics of simulation and analysis. Towards that goal, we have identified core simulation and analysis capabilities (see Figure 6) that are required to enable and simplify the CASoS modeling process. Just as with the modeling capabilities discussed in the above section, these simulation and analysis capabilities are not static, but are emergent and adaptable in order to meet the dynamic needs of the CASoS community.

The simulation and analysis capabilities have been grouped into three categories: 1) input and output, 2) execution and analysis, and 3) visualization and user interfaces. Input and output capabilities are focused on simplifying problem setup and capturing model output. Execution and analysis capabilities are focused on simplifying the process of running simulations on a desktop computer or on a high-performance computer and on performing such tasks as parameter sweeping, sensitivity analysis, and uncertainty quantification. The visualization and user interface capabilities focus on simplifying the often-difficult task of visualizing model output in order to analyze and understand and communicate the model results.

Appendix A contains an evolving list of recommended or exploratory solutions to common needs in the CASoS modeling, simulation, and analysis lifecycle.

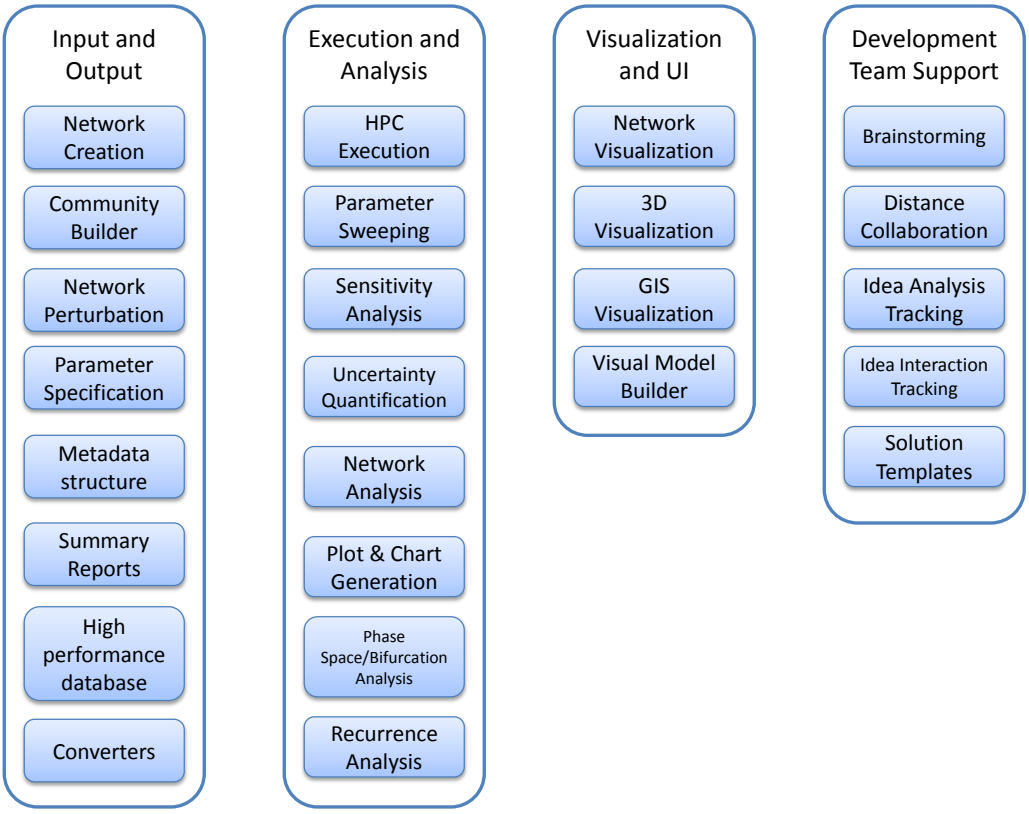


Figure 6. Simulation, Analysis, and Development Team Support Capabilities

3.3.1. Input and Output

The necessary capabilities related to input and output that have been identified thus far are focused on simplifying problem setup and capturing model output for visualization and analysis.

Since the use of networks is a core capability in CASoS engineering, Loki provides the ability to readily create various types of networks. Supported network types include small world, random, lattice, ring, star, wheel, scale-free, and bipartite. Additionally, networks can be created from spatial data, which is often required to model critical infrastructures. Networks can also be created to represent a community somewhere within the United States. These stylized communities are based on US census data and are used to construct various networks of people for the purpose of modeling social interaction.

Loki also provides the capability to perturb a network, which is a capability that is often required when modeling physical infrastructures to determine the effects of a large natural disaster such as a hurricane.

While the approach used to define input parameters to a CASoS model can be specific to the model or model type, we are seeking to identify patterns that can then be turned into a common capability. Not only will this capability simplify model development, but it will also enable the use of a common approach towards core analysis activities such as running parameter sweeps, uncertainty quantification, and sensitivity analysis. Since uncertainty quantification is critical to policy discrimination in a policy-oriented CASoS application, and represents a differentiating capability

for Sandia's CASoS Engineering effort, developing a common Application Programming Interface (API) and job submission system for uncertainty quantification is vital, and one of our top priorities. (See the Future Directions section of this document for more details.)

We have taken initial steps towards the definition of a common metadata structure for model output. While the output of each model will certainly differ, the goal of this task is to capture common metadata that would then facilitate the use of a core set of visualization and analysis tools and the automatic generation of summary reports.

Writing model output to a database facilitates visualization and analysis, but can often result in a performance choke point. The CASoS Engineering Environment provides an Oracle database and host machine that is dedicated to capturing the output of CASoS models. The database is configured and tuned specifically to support rapid data insertion. While this is a good first step, it is an area that requires further effort and we are monitoring and supporting the development of a corporate high-performance database capability. Efforts in this area have also led to the identification of initial patterns for model configuration such that the volume of output data can be changed depending on the goal of a specific model execution. For example, during initial model development and tuning, large amounts of output data will likely be required, but a subset of the output data may be sufficient when the model is mature and the modeler is performing a sensitivity analysis, for example.

In order to support the use of other common network analysis and visualization tools, Loki provides a number of converters that output Loki networks in other formats, such as JUNG (Java Universal Network/Graph Framework) and GraphML.

3.3.2. Execution and Analysis

The execution and analysis capabilities being developed as part of the simulation and analysis platform are currently focused on simplifying the process of running simulations on a desktop computer or on a high-performance computer, and on performing such tasks as parameter sweeping, sensitivity analysis, and uncertainty quantification. Work in this area is just beginning, as these activities have thus far been generally model and/or modeler specific. Some capabilities have been developed to simplify the use of corporate clusters to perform simulations, but additional efforts are needed in this area. Some potential tools—such as DAKOTA[4]—to perform parameter sweeps, uncertainty quantification, and sensitivity analysis have been identified and tested with a small number of models. Additional efforts, with support from experts in this area, will occur as needed and as resources allow.

Loki provides a network analysis capability that can be used to generate common measures such as degree, betweenness, and centrality, and to apply network analysis algorithms such as page rank, voltage cluster, Markov steady state, shortest path, and component clustering. Phase space analysis, bifurcation analysis, and recurrence analysis are additional analytic approaches that can be applied to simulation results.

We have implemented initial capabilities to automate, or at least simplify, the generation of common plots and charts. Time series plots, phase portrait plots, and common statistical plots are

[4] http://dakota.sandia.gov/index.html

all frequently generated and used for model analysis. Tools such as MiniTab and JMP are also used regularly. These tools can readily be connected to the database tables that contain model output, and the use of common metadata for model output additionally simplifies their use.

3.3.3. Visualization and User Interfaces

The visualization and user interface capabilities being provided and developed as part of the simulation and analysis platform are currently focused on simplifying the often-difficult task of visualizing model output in order to analyze, understand and communicate the model results. To a large degree, existing visualization capabilities are most frequently used, as implementing new visualization capabilities is resource-intensive. Additionally, CASoS practitioners often have personal preferences or model-specific requirements that drive the visualization choice.

Packages that have been successfully utilized to visualize networks include JUNG, Pajek, Prefuse, Piccolo, and Network Workbench. An initial 3D network visualization tool has been developed and is being prototyped within a financial system model.

For applications that are modeling underlying physical networks, such as the Critical Infrastructure Recovery model and the Natural Gas Network model, Google Earth has successfully been used for geospatial visualization. Our simulation and analysis platform includes an API that generates a network representation in Google Earth's native format, KML (Keyhole Markup Format). The CASoS Engineering Environment also includes a rudimentary capability to generate a network representation in a native format used by ArcMap, a commercial tool from ESRI that is often used for geospatial visualization and analysis.

A desired future capability is a visual model builder that allows a CASoS practitioner to construct an initial model by connecting core modeling components via a graphical user interface. Such an interface would enable model development without requiring software development.

3.3.4. Development Team Support

The activities of the model development team (both conceptual and computational) can benefit from computational support in the CASoS Engineering environment, not just the individual modeler. At this point, such support is still in the planning stages, but the following capabilities are considered to be useful to a model development team:

- Brainstorming support, perhaps using a mind-mapping tool such as Compendium[5] or FreeMind;[6]
- Distance collaboration support for distributed design and development;[7]

[5] http://compendium.open.ac.uk/

[6] http://freemind.sourceforge.net/wiki/index.php/Main_Page

[7] Jabber/XMPP is being investigated as a protocol to support synchronous multimedia collaboration.

- Idea tracking support, both in tracking analysis activities that actualize an idea and in tracking the interactions between ideas; and
- Solution templates, in the form of either application-level or capability-level code shells, which facilitate the implementation of a new solution of a particular type. This promotes reusability at a lower level than conceptual or design reuse.

4. KNOWLEDGE FACILITATION PLATFORM

Knowledge facilitation in support of education and training is a primary area of focus for the CASoS Engineering initiative and is being supported by the CASoS Engineering Environment through a platform comprised of a Wiki site, SharePoint sites, an external web site, and the identification of common processes and methods for effective education and training. Since the goal of the Knowledge Facilitation Platform is to facilitate the formation of a CASoS Engineering Community of Practice, several other desired capabilities fall into this area as well, such as document exchange, search, and retrieval with external partners;[8] and a so-called "Educational Sandbox.[9]"

4.1. Wiki

A Wiki is being used to support the development of the modeling, simulation, and analysis platform and the use of the platform to create CASoS applications. The Wiki makes it possible and easy for all members of the CASoS community to contribute knowledge and information and organize it as appropriate. A large, up-front design is unnecessary, as the content and structure emerge, grow, and adapt as contributions are made from the community. The health and utility of the Wiki reflects and depends directly upon the vibrancy of the CASoS community. The value of the Wiki dwindles in the absence of regular contributions, but thrives with frequent updates from multiple community members.

The Wiki is currently being used to capture information about the overall CASoS effort as well as detailed information about CASoS applications. The high-level CASoS information includes details about the CASoS process and the application of the CASoS Engineering Framework, patterns, announcements, monthly plans, and contact information for people involved in the CASoS effort. The application-specific Wiki pages reflect the activities of the application team and can vary considerably in content, but there is some common structure to help practitioners find similar information from within different application pages.

The software that is being used to implement the Wiki is TeamForge, a commercial product marketed by CollabNet.[10] In addition to Wiki support, TeamForge provides an issue tracking system, a release management system, a rudimentary project management system, and Subversion source code control, which is an Open Source version management system. These combined features make TeamForge an excellent solution for the CASoS effort as it supports both a knowledge management environment and a software development environment, which is discussed in the next section. An annual fee is required to maintain a TeamForge account.

---

[8] The use of Zotero (http://www.zotero.org/) is being explored to provide such capability.

[9] See the Future Directions section of this document for a brief discussion of the Educational Sandbox.

[10] http://www.collab.net/products/ctf/

Access to the Wiki is access controlled via Phoenix project membership. The Wiki is accessible via the reverse proxy,[11] which facilitates collaboration with offsite partners. However, such partners must have a Sandia account, a CryptoCard, an active TeamForge account, and be a member of the Phoenix project on TeamForge.

4.2. SharePoint Sites

Our education and training platform provides SharePoint sites to facilitate document sharing and archiving. While the Wiki works well for generating and managing dynamic content, SharePoint does a better job of managing documents and providing an environment within which we can collaboratively write and edit.

In addition to a SharePoint site for the overall CASoS effort,[12] we also create and use application-specific SharePoint sites. The main site has tabbed links to each of the application sites, each of which contains a common core structure and basic content. The core structure and content attempts to map directly to the processes outlined within the CASoS Engineering Framework, with some modifications and additions based on experience with applying the Framework. This common structure facilitates the sharing of knowledge and experience among CASoS practitioners and the archival of important documents and related information.

The SharePoint sites are access controlled and specific permissions can be granted to groups or individuals. The sites are accessible remotely and require authentication via a CryptoCard and appropriate permissions.

4.3. Public Web Site

For Sandia to become a recognized leader in the CASoS domain, the knowledge and experience gained internally must be used to educate and train a broad and unrestricted audience. To help achieve this vision, we have constructed an external Web site[13] that is hosted on the Sandia open network and contains the corpus of CASoS artifacts that have been approved for public release. This web site enables the sharing of knowledge and straightforward collaboration with external CASoS researchers and practitioners.

4.4. Processes and Methods

The processes and methods of CASoS implementers, practitioners, and supporters form a critical component of the education and training platform. Without core processes and methods, the CASoS tools and capabilities will be of limited value. The CASoS Engineering Framework describes a potential flow that can be used as a guide for CASoS practitioners to identify aspirations, design and test solutions, and the actualization of the solutions, while the core CASoS

---

[11] https://rproxy.sandia.gov/teamforge

[12] https://sharepoint.sandia.gov/sites/Phoenix/default.aspx

[13] http://www.sandia.gov/CasosEngineering

developers and initial practitioners are working to identify effective processes and methods that catalyze the building of a community of practice while actively solving CASoS problems. The CASoS Engineering Environment and the education and training platform in particular, are being designed and developed to track these emergent processes and methods.

Initial processes and methods have focused on the application of the CASoS Engineering Framework. In particular, efforts have been directed at the identification and articulation of aspirations, as this is a critical step towards addressing a CASoS problem. Efforts have also been directed towards identifying the essential components of a conceptual model and towards the construction of a template to guide conceptual modeling. A group of CASoS practitioners have met regularly to gain experience with CASoS conceptual modeling by applying the template to real applications and subsequently implementing the model. A spiral approach was used, and the template, the process, and the implementation improved with each spiral.

The CASoS Engineering Environment is supporting and enhancing the processes and methods of the CASoS community via the use of the Wiki, SharePoint sites, and an external web site.

5. DEVELOPMENT ENVIRONMENT

The CASoS development environment has been constructed to facilitate the use of software to create and execute CASoS models. The foundation of the development environment is the Loki toolkit. Agile development methodologies are being used, and TeamForge-based issue tracking and version control systems support development efforts. The development environment has been implemented to facilitate collaboration. Documentation and testing guidelines have been developed, in accordance with the CASoS philosophy, to promote the generation of consistent, readable, usable, testable, and maintainable code. The following paragraphs document the existing development environment; see Appendix B for a categorized discussion of external sources that could enrich the current development environment.

5.1. Version Control System

The source code version control system provided by TeamForge is Subversion (SVN). A migration of code from other version control systems to a Subversion repository was undertaken earlier this year, during which time the code was also cleansed and trimmed. Access to the repository requires a TeamForge account and membership in the Phoenix project. Subversion is accessible remotely via VPN with a CryptoCard to facilitate external collaboration, but this has yet to be thoroughly tested with the CASoS repository specifically; external collaborators need an SRN account and a CryptoCard in order to access the TeamForge SVN repository.

Core CASoS capability modules and other reusable libraries each have their own top-level directory within the repository that contains a *trunk* sub-directory and optionally *tags* and *branches* sub-directories. Models that are built using these modules and libraries are stored in a sub-directory of the *models* top-level directory. The model's code is stored directly in this directory without the *trunk/tags/branches* structure. This repository structure will jumpstart new model development and will simplify application sharing.

Note that two other source code repositories exist, each of which are based on CVS (Concurrent Versions System) instead of SVN. One is on the Sandia Open Network (SON) and consists of legacy CASoS applications as well as applications developed with external collaborators. The other is on the SRN and primarily consists of CASoS applications being developed for the NISAC project. However, the canonical version of the Loki toolkit is maintained in the SVN repository.

5.2. Issue Tracking System

An issue tracking system that is a component of the TeamForge Phoenix project can be used to track defects, enhancements and milestones. Since the TeamForge project is also used to manage the overall CASoS project, software development milestones and issues can be readily tied to high-level deliverables. Issues can be created, tracked, and addressed in accordance with agile development methodologies.

5.3. CASoS Server

The CASoS server is an integral part of the computational capabilities of the CASoS Engineering Environment. It currently consists of a Dell PowerEdge R815 Rack Server, with 48 cores, 256GM of memory, and over a terabyte of storage, running the Red Hat Linux operating system. The primary uses of the CASoS Server are for initial characterization of Uncertainty Quantification for a CASoS model, and for processing restricted-use data such as Add Health Restricted Data.

5.4. Database Back-end

The CASoS Engineering Environment provides an Oracle database and host machine that is dedicated to capturing the output of CASoS models. The database is configured and tuned specifically to support rapid data insertion from simulations running on massively parallel supercomputers such as Red Sky. Currently, the machine is a dedicated 64-bit Windows Server 2003 machine running an Oracle 11g R2 database engine. The Oracle database has been optimized for a large number (greater than 500) of simultaneous connections. Persistent storage is split into an Operation System drive of 100GB and an Oracle data drive of 1,900 GB.

5.5. Guidelines

In addition to some basic principles for code development, the development environment also includes guidelines for documentation and testing to help ensure that developed code is tested, understandable, and useful.

5.5.1. Documentation

There are a several classes of documentation relevant to software development including, but not limited to, API documentation, design and architecture documentation, process documentation, requirements documentation, and configuration "cookbook" documentation (see Appendix C for the current cookbook). Each of these has a unique purpose and the processes and organization around each have been designed to make them as effective as possible.

Documentation of APIs is done alongside the code using whatever format and tool is appropriate for the implementation language (*e.g.*, Javadoc comments for Java code). Missing documentation for a public API should generally be considered a defect to be fixed.

Design and architecture documentation informs both the usage of a module and its future development. This type of documentation should be kept in the source repository with the code it describes. Keeping the documentation close to the code establishes a context for understanding. This type of documentation should elucidate the concepts embodied by the code and how they fit together through examples and prose. Future development of the module should be aligned with the existing architecture. Design and architecture documentation may alternatively be stored in the Wiki, especially when its scope extends beyond the scope of a single module.

Process documentation describes how non-creative (*i.e.*, routine) software development activities are performed. These documents should be specific enough that two different people following the

process will produce the same results. These documents should be kept in the Wiki. Ideally documents of this sort will be few since processes are automated whenever possible.

Code requirements are entered into the issue tracking system as artifacts. Subtasks for implementation of a particular requirement will also be entered as artifacts and linked from the main artifact description.

5.5.2. Testing

As the number of models using the CASoS development environment grows it becomes increasingly important that APIs are stable and a module's behavior does not inadvertently change over time. Automated tests (primarily JUnit tests implemented in the Eclipse IDE) are used to ensure that the code's behavior functions as specified. Testing is not limited to unit tests, but includes tests of system behavior as well. When a defect is discovered, a test for it should be written. This will ensure that bugs are not accidentally introduced or reintroduced. Ideally, the CASoS application development culture should adopt a philosophy of Test-Driven Development (TDD), in which software tests for a new feature are written *first*, before the code for the actual feature. Should more extensive or rigorous testing be required, project build tools such as Maven[14] or build automation tools such as Hudson[15] should be investigated.

[14] http://maven.apache.org/

[15] http://hudson-ci.org/

6. FUTURE DIRECTIONS

Future directions are discussed in terms of key strategic goals for the CASoS Engineering Environment. Two high-level strategic goals have been articulated, corresponding to the two high-level platforms of the CASoS Engineering Environment: To support the development of CASoS Engineering applications (via the Modeling, Simulation, and Analysis Platform), and to cultivate a CASoS Engineering Community of Practice (via the Knowledge Platform). The categorized subgoals in the following section are considered the most strategic at the present time.

6.1. Support the Development of CASoS Engineering Applications

6.1.1. Kokopelli Toolkit

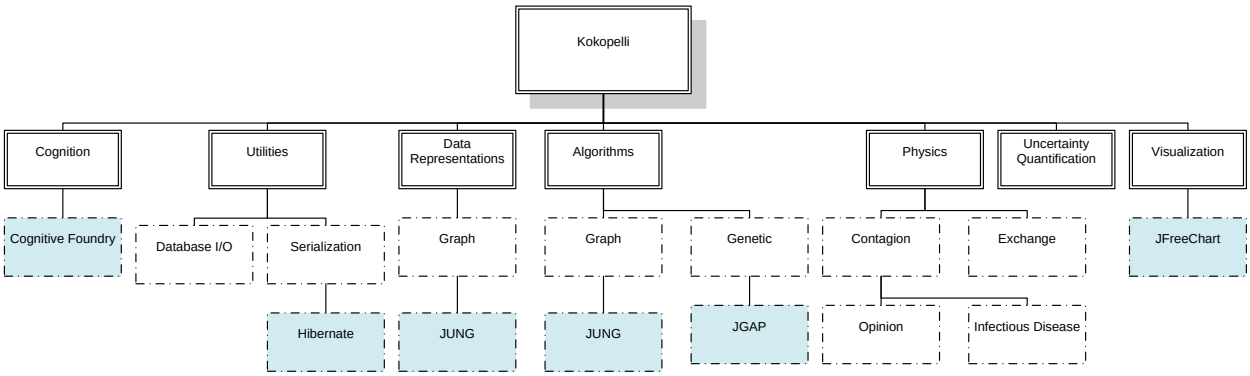


Figure 7. Notional Architecture for the Kokopelli Toolkit

Refactor and extend the Loki software framework such that it is understandable, usable, actually used in practice, and represents a differentiating capability relative to other frameworks in the CASoS space (such as Repast Simphony, MAVEN, and NetLogo). Towards that end, the following actions are recommended:

- Rebrand Loki as Kokopelli (a shape-shifter like Loki, but from the Pueblo Indian tradition)
- Reduce the cognitive load of understanding and using the framework by rearchitecting it to consist of a thin value-added interface layer over third-party software packages and Sandia-developed differentiating capabilites (such as Uncertainty Quantification and physics modules for common CASoS solution structures). The diagram below depicts an incomplete notional software architecture for the purposes of illustration.
- Relative to other CASoS toolkits (such as Repast Simphony, NetLogo, and AnyLogic), a differentiating capability of Kokopelli is the categorization and implementation of common CASoS problem dynamics (a.k.a. “physics”) in the toolkit itself. Since such dynamics currently only exist at the application level, they will need to be modularized and ported to the toolkit library.
- Initiate the Copyright Assertion Process to authorize the distribution of Kokopelli as an open-source software toolkit.
- Create and maintain the infrastructure to support Kokopelli as an open-source toolkit. Such resources include an external Web server, a SourceForge-like repository that accepts

changes from external users, a bug tracking and feature request system, a comprehensive series of tests in the source code (both unit-level and system-level), documentation at several levels (Javadoc, Tutorial, User's Guide, Reference Manual), example programs, and perhaps even a bulletin board or phone line for support requests. The resources required for such support should not be underestimated.

6.1.2. Uncertainty Quantification

Develop a standard methodology and I/O interface for performing Uncertainty Quantification (UQ) on CASoS simulations, and make the deployment on either the internal cluster or the HPC machines as seamless and transparent as possible. Since this represents a differentiating capability relative to other CASoS Engineering efforts, implementing this API through the Kokopelli toolkit is highly desirable.

6.1.3. High-Volume Visualization Capability

Identify and adopt, or specify and create, a standard high-volume visualization capability for the output of CASoS simulations

6.1.4. Facilitate the Transition between Conceptual and Computational Model

Currently, the process for creating conceptual models is almost completely manual, consisting largely of brainstorming sessions and high-level "block and stick" diagrams. So is the process for transforming a conceptual model into a computational model. Computational support for creating conceptual models (perhaps by using visual mapping software such as Compendium[16]), capturing assumptions and considerations related to the development of those conceptual models, noting test cases that the computational model should support, and for transitioning those conceptual models to computational models, would greatly facilitate the CASoS modeling process.

6.2. Cultivate a CASoS Engineering Community of Practice

6.2.1. Source Code Collaboration Architecture

Create and get the appropriate approvals for a three-tier CASoS source code collaboration architecture. The flow of code between each tier needs to be carefully identified, approved, and facilitated by software tools. The three categories of collaboration are:

- Internal collaboration (includes Sandia-proprietary artifacts)
- Trusted external collaborators (probably involving account approvals and cryptocards; includes customer-proprietary artifacts)

[16] http://compendium.open.ac.uk/institute/

- Arbitrary external collaboration (anywhere on the open Internet, even internationally; requires an open source version of the CASoS software framework)

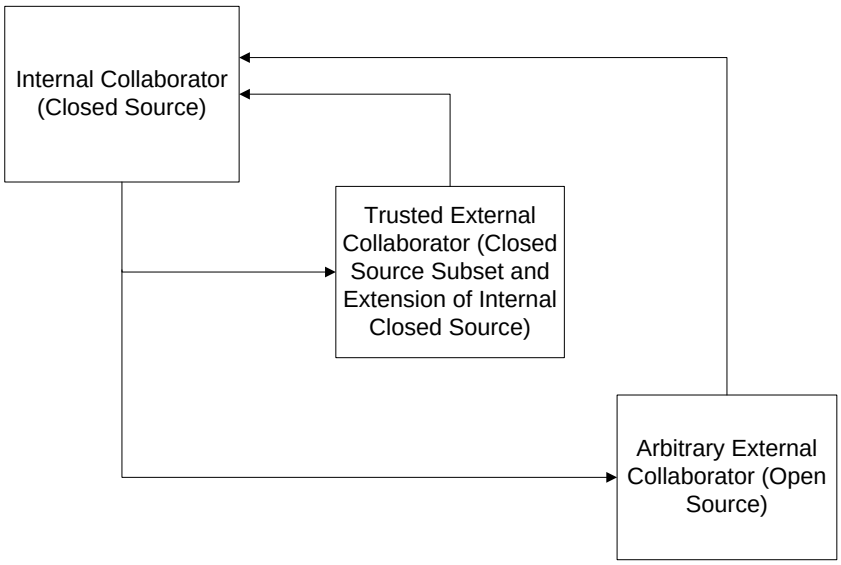


Figure 8. Three-Tier Architecture for External Collaboration

6.2.2. Educational Sandbox

Assemble and make freely available a "sandbox" of runnable CASoS simulations in a comprehensive set of problem categories for paedogogical purposes. The categories should map to the evolving lexicon of CASoS problems being collected and organized by the Framework effort.

7. APPENDIX A: RECOMMENDED SOLUTIONS LIST

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
2D Charting and Graphing	JFreeChart		Pat Finley John M. Line-barger (for the Developer Guide and demo code)		JFreeChart is free but the Developer Guide and demo code (which are highly recommended) are not
Agent-Based Modeling Toolkits	Loki	Almost all Phoenix projects use Loki to some extent (the Recovery model is an exception)	Walt Beyeler Daniel Cannon	Repast (which was the precursor to Loki); Swarm (which now has Java bindings, but which seems to no longer be maintained); the MASON multiagent simulation library; and NetLogo. Repast Simphony is the latest version of Repast.	Only open-source Java-based ABM toolkits are considered here Loki is currently being refactored for greater usability and utility, and may be rebranded as Kokopelli.

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Cognition and Behavior			Justin Basilico	Cognitive Foundry is a Sandia-developed, Java-based potential solution that should be explored first.	Cognitive Foundry has both open source and internal versions.
Dependency Mgt (external JAR file dependency resolution)	Ivy The IvyDE Eclipse plugin	The loki-common, VAMOOSE and hospitals projects	John M. Line-barger Chris Harmon Geoffrey Reedy	Maven (a more heavyweight solution than Ivy) JAR files in the lib directory of each project (but this has been a mess, and does not exploit common project JAR needs)	Most of the Phoenix projects have now been "Ivyized"
Encryption and Decryption (program-matic)			Tom Moore	JCE (Java Crypto-graphy Extension), part of the JDK, but the Jurisdiction Policy Files must be downloaded separately	

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Genetic Algorithms	Loki contains some GA functionality. So does Cognitive Foundry.		Justin Basilico John Eddy	JGAP JGAL Genetic Algorithms Toolkit (seems old) JAGA Jenetics (nice API)	JGAP has been around for awhile and seems fairly mature
Graphical User Interface Widgets	InfoNode Docking Windows	Infect3	Daniel Cannon		A five-developer license for InfoNode Docking Windows has been purchased in order to remove the GPL license constraint from the free version. This allows us to distribute binaries that use the product freely and externally.

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Hybrid SD-ABM Simulation Tool			John Siirola Pat Finley Guylaine Pollock	EMPaSE (formerly known as Omega-AB) AnyLogic (expensive commercial product, but widely used in the public health community) Simile appears to be another example.	A drawback of EmPaSE is that it is written in C++, unlike AnyLogic, which is written in Java. Pat Finley is exploring the potential utility of AnyLogic on the VA Threat Modeling project, using the free 30-day license Exchange2 is an example of such a hybrid model
Integrated Develop-ment Environ-ment (IDE)	Eclipse	SocNet VAMOOSE		NetBeans	Virtually all CASoS Engineering projects use Eclipse as the IDE.

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Layout Manager	The default (Swing) built into the Java JDK	Almost all Phoenix GUI applications use the default Java layout manager	Tom Moore	MiG Layout, which works with both Swing and SWT SWT (Standard Widget Toolkit) from the Eclipse project is another option	The VAMOOSE and FDA CTP projects are using MiG Layout as the GUI layout manager because it is so easy to configure

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Literature organization and exchange			John M. Line-barger Tom Moore	For UUR documents and collaboration with external partners, the Zotero service could prove useful For OUO and internal documents, a shared network drive on the SRN that is indexed by Google Desktop or the Google Search Appliance (GSA) may be a promising approach	The FDA CTP project is exploring the use of Zotero to organize documents and exchange them with external partners (such as FDA CTP and CDC). The stumbling block is not technical but legal – posting subscription-controlled content such that non-subscribers have access to it requires proper licenses and individual agreements with publishers.
Machine Learning			Brian Jones	Weka 3 is the standard Java-based machine learning algorithms package	

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Network Visualization and Analysis	JUNG for low-to-mid volume data For mid-to-high volume data, export the graph in a standard format (like GraphML) and visualize in another tool	VAMOOSE	Tom Moore	For mid-to-high volume data, try the Network Workbench, Pajek (in conjunction with UCINET for graph analysis), Gephi (but beware of instability), or the Titan Informatics Toolkit (via a Python script). For prototyping, try the Microsoft Excel plugin NodeXL.	The hospitals paedogogical application uses Piccolo2D for network visualization; Piccolo2D provides a zoomable user interface. Prefuse is another option for Java-based network visualization. However, it no longer appears to be maintained.
Neural Networks			Jacob Hobbs	Encog	Encog is Java-based but also has a .NET implementation

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Object Persistence (which supports checkpoint and restart)			Tu-Thach Quach Brian Jones	Hibernate (may be better if you are starting with an object model) iBATIS (may be better if you are starting with a schema)	The Hospitals paedogogical application uses Hibernate/Spring for persistence
ODE Solver	Apache Commons Math library	Exchange2 application	Walt Beyeler Jacob Hobbs		
Program-ming Language	Java	Loki toolkit VAMOOSE model for the VA Infect3 model	The entire Phoenix team	Python (which was the initial solution chosen for the Exchange2 model) Scala (which runs on the Java JVM and can be considered a dialect of Java)	Geoff Reedy was involved in the exploratory efforts using Python and Scala

Task	Recommended Solution	Example	Contact	Other Potential Solutions to Explore	Notes
Statistics	The Colt Project	The IBM in the FDA CTP project	Steve Verzi		Colt also contains other mathematical capabilities, such as linear algebra, histograms, and random numbers.
Text Analysis				Citrus is being developed by Travis Bauer (05635) as part of an LDRD	Citrus is fast, Java-based, and can be used either at an application level through its GUI or at the JAR level through its API
XML I/O	JAXB (built in to the JDK distribution since 6.x) XJC Eclipse Plugin	The VAMOOSE application (for the VA Threats project)	John M. Line-barger Brian Jones	Simple XML Serialization XStream	JAXB is a good choice if you are starting with an XSD Schema. Simple XML is easier if you are starting with a Java class. XStream is used by the Cognitive Foundry.

8. APPENDIX B: EXTERNAL SOURCES TO CONSIDER

a. Simulation Tools

i. John Siirola (1433) has created a multi-paradigm simulation engine called EMPaSE. The interface is written in C# using .NET and allows drag-and-drop creation of simulation models. The simulation engine itself is written in C++, and can run on HPC machines. A plugin architecture is used to achieve a high degree of modularity. The internal URL is https://giskard.sandia.gov/trac/empase.

ii. AnyLogic is a commercial Java-based multiparadigm simulation tool (primarily a hybrid of agent-based and systems dynamics modeling). It was created in Russia, and a license costs on the order of $6K. The URL is http://www.xjtek.com/.

iii. Steve Goldsmith and Shannon Spires (both 06351) have created a CLOS (Common Lisp Object System)-based framework for agent modeling. Their tool has been for internal use up to this point, but an API could be developed that would allow their tools to be used by others.

iv. Seldon is a Java-based toolkit for performing large-scale social simulation. It was developed at Sandia California, and the current PI is Karim Mahrous (8116). A version of Seldon which supports cognitive agents—called Cognitive Seldon—uses the Cognitive Foundry to implement cognition-related capabilities (see below).

v. Umbra and SoSAT (Systems of Systems Analysis Tool) are more specialized simulation tools, created by 6344 and 6342, respectively. Neither Umbra nor SoSAT can currently run on HPC machines, but they have been exercised on large clusters by creating multiple simulation processes and farming them out to cluster nodes. A narrative should be carefully crafted that embraces these two tools within the CASoSE methodology and framework.

- Umbra is a time-stepped simulation engine with integrated real-time visualization capability that originated as a robotics simulator. It is best suited for embodied agents in a 3D environment. Dante is an application of Umbra that simulates force-on-force engagements. Umbra is written in C++ with either Tcl or C# wrappers, uses Open Scene Graph (OSG) as the visualization engine, and supports HLA for distributed simulation capability. The URL is http://umbra.sandia.gov/index.html.
- SoSAT is a discrete event simulation tool that models logistics and sustainment of military systems-of-systems over the duration of a mission. It is written in Visual Basic.

vi. Adevs (http://www.ornl.gov/~1qn/adevs/) is the C++ simulation library used by Carl Diegert, Jean-Paul Watson, and Danny Rintoul (all of 1412) in their recent obesity modeling short-duration LDRD. Adevs was chosen because its ability to simulate both discrete and continuous variables was a good fit for the domain. However, Carl Diegert has stated that though Adevs was a good choice for the rapid prototype delivered for the LDRD, he would not recommend its continued use in any follow-on work.

b. *Open Source Java-based Agent-based Modeling Frameworks*

i. Swarm (http://www.swarm.org/index.php/Main_Page) Originally an Obective-C library inspired by the work of the Santa Fe Institute, Java Swarm allows Swarm's Objective-C libraries to be used by Java code. Note that a SwarmFest is held every year. Cygwin is required for Windows installations.

ii. Repast (http://repast.sourceforge.net/). PolyNet (the precursor to Loki) and the Loki Toolkit had their origins in Repast.

iii. NetLogo (http://ccl.northwestern.edu/netlogo/). NetLogo was inspired by MIT's StarLogo, which is intended to be an educational programming environment.

iv. MASON (http://cs.gmu.edu/~eclab/projects/mason/) is a discrete event multiagent simulation library from George Mason University.

c. *Open Source Java-based Visualization Tools*

i. JUNG (http://jung.sourceforge.net/), the Java Universal Network/Graph framework.

ii. Piccolo (http://www.cs.umd.edu/hcil/jazz/index.shtml) is a 2D graphics framework

iii. Prefuse (http://prefuse.org/) is an interactive visualization framework that may no longer be under active development, at least in its Java form.

iv. Network Workbench. An Eclipse-based network construction, analysis, and visualization tool created by Albert Barabási and his students at Indiana University. Perhaps not technically open-source, but very extensible; algorithms and datasets can be plugged in to the existing framework. The URL is http://nwb.slis.indiana.edu/.

v. NodeXL is a Microsoft Excel plugin that displays and analyzes network graphs. Free, but not technically open-source.

d. *Uncertainty Quantification (UQ) Tools*

i. DAKOTA is a design analysis and optimization tool created at Sandia. It includes UQ capabilities. The URL is http://www.cs.sandia.gov/dakota/index.html.

ii. The DART (Design through Realization Analysis Team) project (8962) has simplified the setup of DAKOTA UQ simulations. The DART/UQ environment consists of an Eclipse plugin that allows parameterization and submission to an HPC machine with the click of a button. Robert Clay and Ernest Friedman-Hill are the appropriate DART contacts. The URL is https://dta.ran.sandia.gov/dart/.

iii. iSIGHT is a Java-based commercial design integration and optimization tool. It is especially suited to optimize simulations—even coupled simulations—over a parameter space. iSIGHT is now marketed by Simulia (http://www.simulia.com/).

iv. Phoenix Integration's ModelCenter is a commercial tool similar to iSIGHT. It also supports optimization and uncertainty quantification. Its strength is in integration of coupled simulation tools, while iSIGHT's strength is in optimization of simulation tools. The URL is http://www.phoenix-int.com/software/phx_modelcenter.php.

e. *Non-Java-Based Visualization Tools*

i. Visualization Toolkit. The Visualization Toolkit (VTK) is a mature open-source cross-platform framework for scientific visualization. It is written in C++ with bindings to several scripting languages, such as Tcl, Python, and Java. The URL is http://www.vtk.org/ .

ii. Titan Informatics Toolkit. Titan is an informatics toolkit developed primarily at Sandia as part of the ASC program, then integrated with VTK and open-sourced. It uses the same modular pipelined architecture as VTK but with informatics-related entities, such as tables, graphs, and databases. Brian Wylie (1424) is the PI for Titan. The URL is http://titan.sandia.gov/. Note that considerable C++ programming expertise is needed to create a VTK/Titan application, and knowledge of the open source cross-platform Qt windowing toolkit is also required.

iii. ParaView is an open source IDE (Integrated Development Environment) for scientific and information visualization. It essentially is an IDE for creating VTK/Titan pipelines and visualizing them. Like VTK/Titan, it is based on C++, but supports a Python-based scripting interface as well. The URL is http://www.paraview.org/.

iv. EnSight is a commercial tool for scientific visualization. Sandia has a site license. The internal URL is http://info.sandia.gov/ensight/.

f. Libraries

i. Jon Berry (1416) has created the Multi-Threaded Graph Library (MTGL), which is optimized to create and analyze large graphs on the massively multithreaded Cray XMT machine. MTGL can be used on other platforms as well. It is written in C++. MTGL is especially suited for graph analysis, such as community finding in social networks. However, to date 1400 has not performed social network *simulation*, only social network *analysis*. The internal URL is https://software.sandia.gov/trac/mtgl.

ii. The Boost Graph Library (BGL) is now an integral part of the Boost distribution, as is its instantiation for distributed-memory parallelism, Parallel BGL. A wealth of graph theoretic algorithms is supported by BGL, such as betweenness and centrality. Boost is written in heavily templated C++ and is freely available in open source form at http://www.boost.org/.

iii. The Cognitive Foundry is a Java-based library of cognition and machine learning routines. The PI is Justin Basilico (1432). The internal URL is http://cognition.sandia.gov/Projects/CognitiveFoundry/.

iv. JFreeChart is a popular and free Java chart library. The URL is http://www.jfree.org/jfreechart/.

9. APPENDIX C: CASOS CONFIGURATION COOKBOOK

a. Apply for TeamForge using WebCARS. Go to the Techweb homepage and select WebCARS from the list on the green bar on the left side of the page. In the "Search for New Accounts" box, enter "TeamForge" and press the Search button. Add "CEE SRN A La Carte TeamForge Access" to your cart, and when you check out, enter a project and task for the account. Checking out submits the request to your manager for approval. It can take up to a week for the account to be approved by your manager and created in the TeamForge system. When it is created in TeamForge, make sure that someone (generally the person in charge of the CASoS Engineering Environment) adds you to the phoenix project on TeamForge.

b. Install the latest Java JDK for your operating system (32- or 64-bit) from http://www.oracle.com/technetwork/java/javase/downloads/index.html, and put the bin directory in your system PATH

c. Install the latest version of Eclipse (currently Helios; the "Classic" package is best) from http://www.eclipse.org. Make sure the architecture matches your JDK (32- or 64-bit).

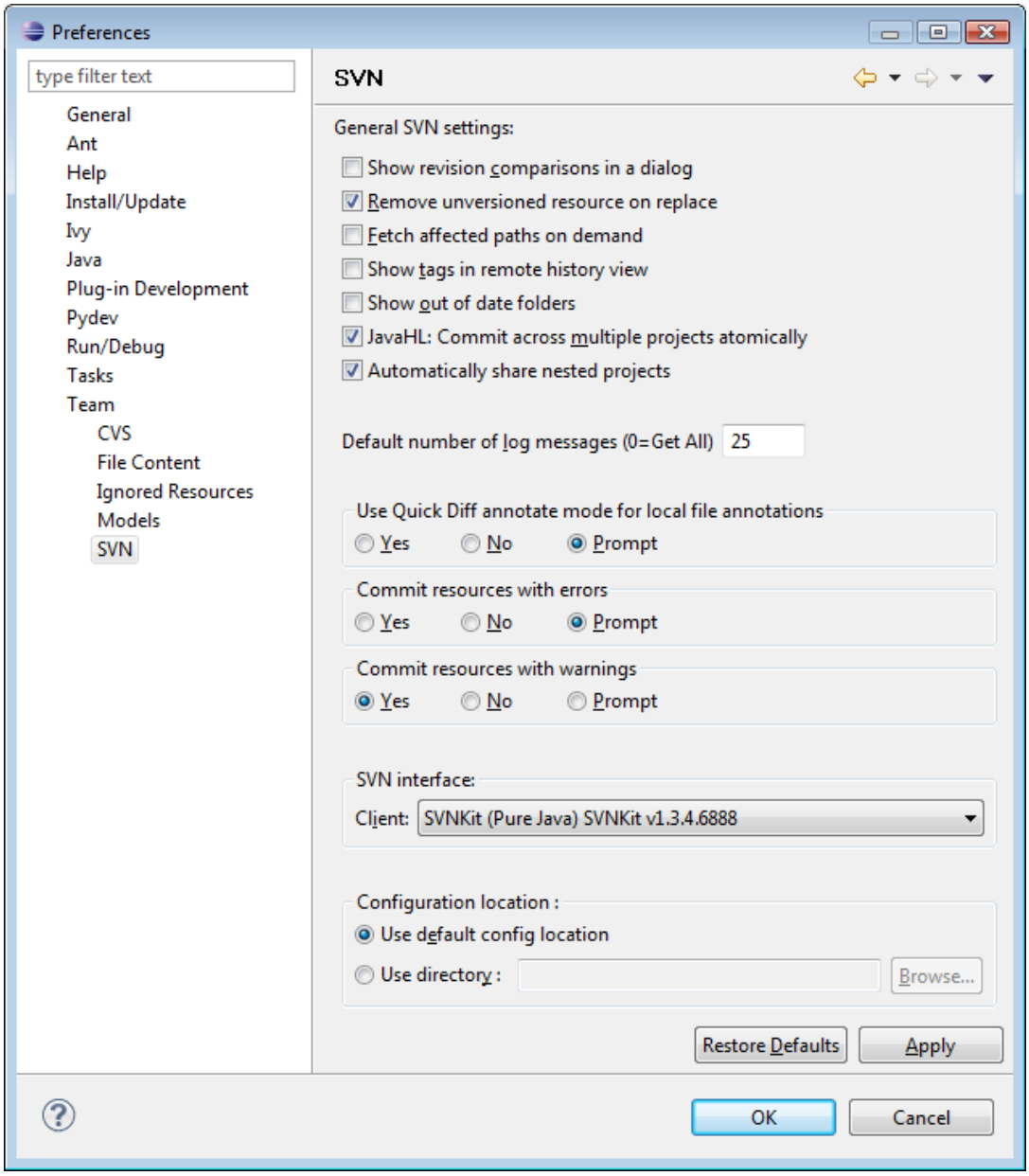
d. Make sure that your Eclipse workspace is stored in a directory that does not have spaces embedded in its system path (*e.g.*, anything off of "Documents and Settings" or "My Documents"). This is especially relevant to Windows XP and below.

e. Invoke Eclipse, and set the network proxies to allow external Internet access from Sandia's SRN. Click on Window → Preferences → General → Network Connections. Set Active Provider to Manual, then edit both the HTTP and the HTTPS entries, setting the Host to wwwproxy.sandia.gov and the Port to 80. Don't specify any user authentication information.

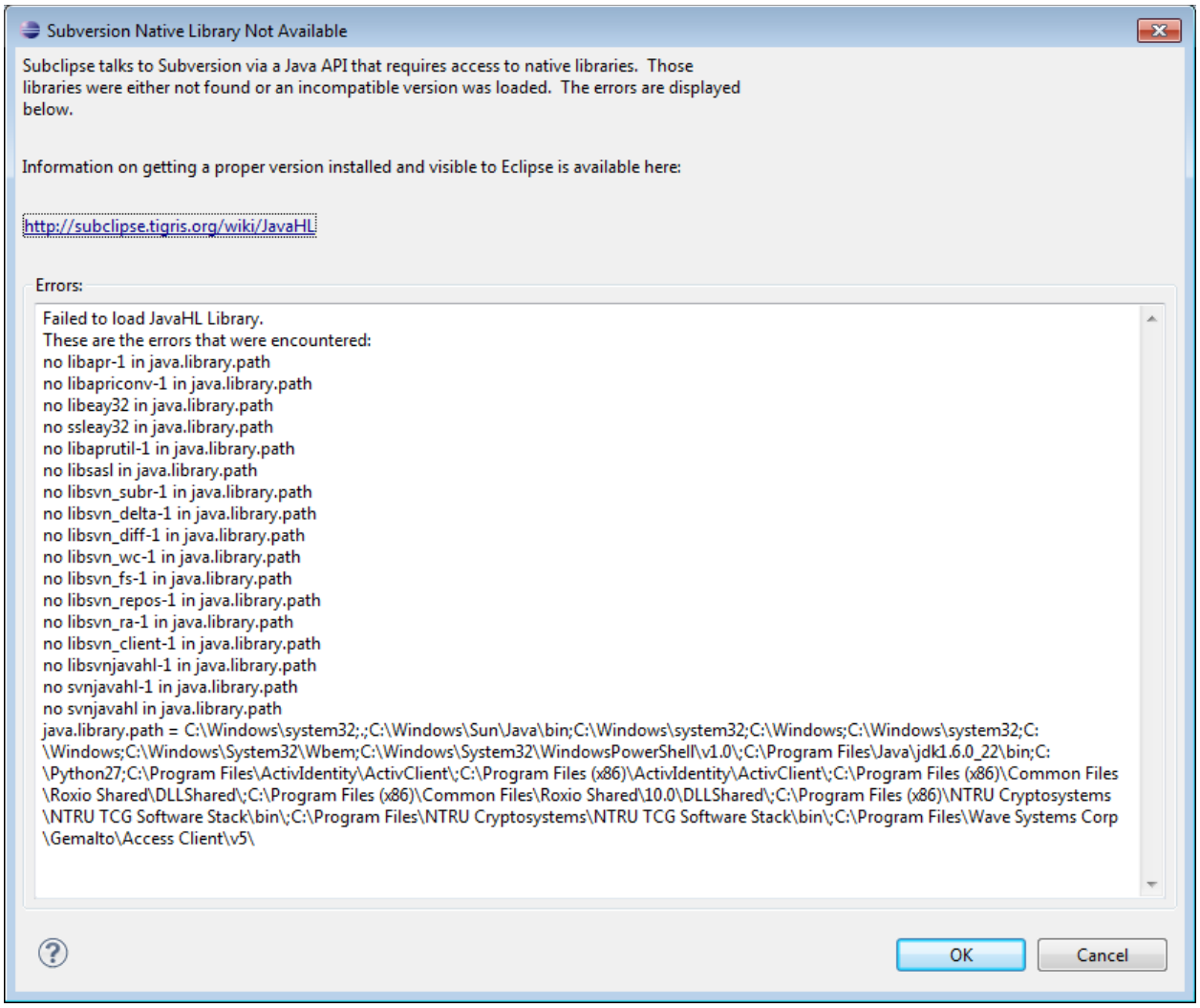
f. While still in Eclipse, install the latest versions of the IvyDE plugin and Ivy itself from the Ivy update site (http://www.apache.org/dist/ant/ivyde/updatesite). Click on Help → Install New Software… Press the Add button, name the update site properly, select both Ivy and IvyDE when they show up in the list box, and install them. Restart Eclipse when the installation is done.

g. Install the latest version of the Subclipse plugin from the Sublipse update site (currently http://subclipse.tigris.org/update_1.6.x). Also install the CollabNet Merge Client, the Subversion Revision Graph (which requires the EMF library), the SVNKit Library, the JNA Library, and the SVNKit Client Adapter. (Even better, install **everything** listed on this update site.) Restart Eclipse when the installation is complete.

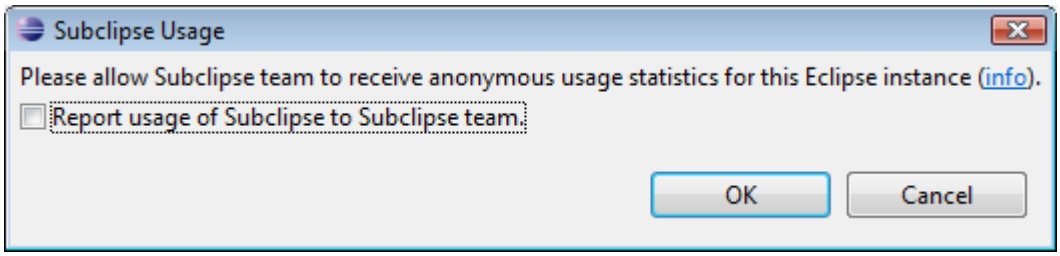
7. Now pick SVNKit as the SVN interface client to use. Click on Window → Preferences → Team → SVN, and make sure that the Client setting of the SVN interface box looks like the screenshot below.



Note that if you are running 64-bit Windows, you may get an error message box like the one below when you bring up the panel above; this occurs because you don't have a JavaHL-compatible SVN client installed (like SlikSVN). Just press the Cancel button, and select SVNKit from the Client dropdown in the SVN interface combo box.

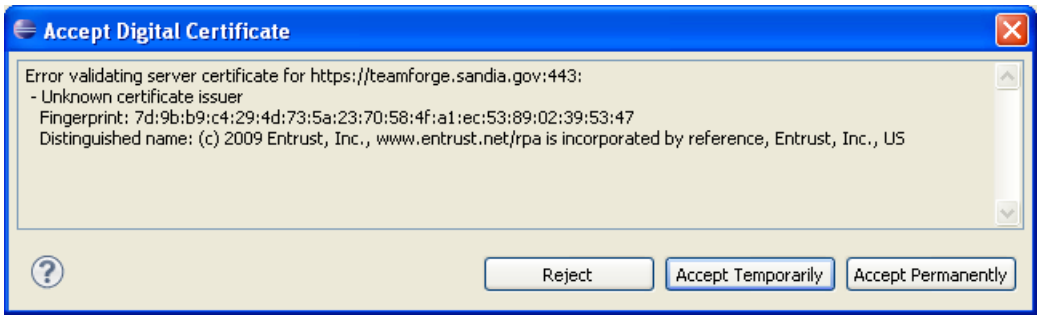


h. When you restart Eclipse, you will get a message like this from the Subclipse plugin. Make sure the box is unchecked so that information is not sent to the Subclipse mother ship.

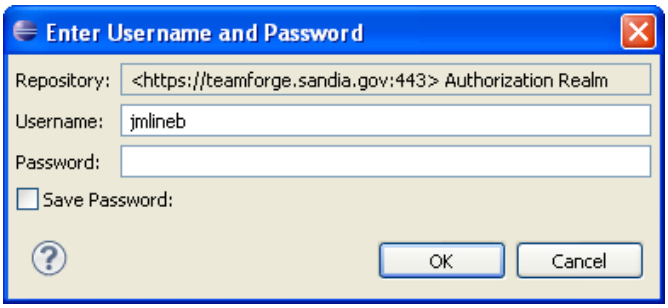


i. Finally, you can now check out CASoS projects in Eclipse from the SVN repository on TeamForge. Simply Import them from https://teamforge.sandia.gov/svn/repos/phoenix. Please note two wrinkles that may be encountered in the process. First, when you initially

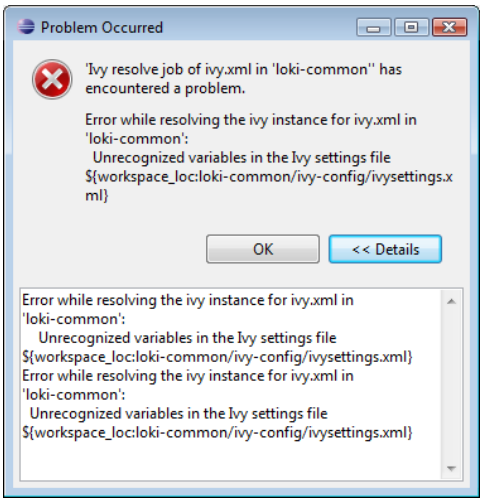
connect to the TeamForge repository, you may get the Digital Certificate error displayed below. Simply press the "Accept Permanently" button to resolve this error.



Second, you will get a password prompt when you import (or update) a TeamForge repository; the prompt looks like the screenshot below. You should only have to enter your password once for each import or update. However, it is important from a security standpoint that you do **not** save your Kerberos password to your local machine.



j. Note that the type of error displayed in the screenshot below ("Unrecognized variables in the Ivy settings file") can sometimes be encountered on the initial Import. This is evidently a timing error, due to the fact that the ivysettings.xml file for each project is referenced using an svn:externals directive, and thus requires a second invocation of SVN in order to download it to your machine. The Ivy resolution often begins before that second download is complete, which triggers the error. To resolve this error, simply wait a bit to make sure that all SVN access has successfully completed, then close the project and reopen it; the Ivy resolution and build steps should now complete successfully.


Problem Occurred
'Ivy resolve job of ivy.xml in 'loki-common'' has encountered a problem.
Error while resolving the ivy instance for ivy.xml in 'loki-common':
Unrecognized variables in the Ivy settings file ${workspace_loc:loki-common/ivy-config/ivysettings.xml}
OK
<< Details
Error while resolving the ivy instance for ivy.xml in 'loki-common':
Unrecognized variables in the Ivy settings file ${workspace_loc:loki-common/ivy-config/ivysettings.xml}
Error while resolving the ivy instance for ivy.xml in 'loki-common':
Unrecognized variables in the Ivy settings file ${workspace_loc:loki-common/ivy-config/ivysettings.xml}

Distribution

1	MS1209	Larson, J.A.	90
1	MS0131	Drayer, D.D.	163
1	MS0166	Brown, L.E.	165
1	MS0145	Wall, F.D.	210
1	MS0431	Vinson, J.B.	230
1	MS0421	Skocypec, R.D.	240
1	MS0116	Hingorani, S.L.	250
1	MS0421	Cox, A.B.	280
1	MS0127	Woodall, T.	430
1	MS0351	Rottler, J.S.	1000
1	MS0123	Chavez, D. LDRD Office	1011
1	MS1427	Barbour, C	1100
1	MS1421	Tsao, J.Y.	1120
1	MS1315	Shinn, N.D.	1130
1	MS1315	Nelson, J.S.	1131
1	MS0321	Leland, R.W.	1400
1	MS1322	Dosahjh, S.S.	1420
1	MS1319	Ang, J.A.	1422
1	MS1319	Brightwell, R.B.	1423
1	MS1322	Aidun, J.B.	1425
1	MS1318	Hoekstra. R.J.	1426
1	MS1318	Hendrickson, B.A.	1440
1	MS1318	Stewart, J.R.	1441
1	MS1320	Collis, S.S.	1442
1	MS0378	Strack, O.E.	1443
1	MS1321	Summers, R.M.	1444
1	MS0316	Hutchinson, S.A.	1445
1	MS0321	Mitchiner, J.L.	1460
1	MS1323	Rogers, D.H.	1461
1	MS1188	Glickman, M.R.	1462
1	MS1188	Lakkaraju, K.	1462
1	MS1327	Wagner, J.S.	1462
1	MS1327	Bennett, P.C.	1463
1	MS1327	Raybourn, E.M.	1463
1	MS1327	Speed, A.E.	1463
1	MS1327	Hart, W.E.	1464
1	MS1326	Backus, G.A.	1465
1	MS1326	Boslough, M.B.	1465
1	MS1316	Rintoul, M.D.	1465

1	MS0110	Johannes, J.E.	1520
1	MS1078	Hermina, W.L.	1710
1	MS1411	Sun, A.C.	1814
1	MS0351	Croessmann, C.D.	1910
1	MS0359	Westrich, H.R.	1911
1	MS1137	Jones, K.A.	1932
1	MS0457	Sleefe, G.E.	2200
1	MS1454	Pfeifle, T.W.	2556
1	MS0968	Linebarger, J.M.	2661
1	MS0640	Carson, T.L.	2994
1	MS0725	Gutierrez, S.M.	4100
1	MS1231	Thornton, A.L.	5220
1	MS1221	Langheim, G.R.	5230
1	MS1164	Keese, D.L.	5400
1	MS1157	Yourick, P.D.	5430
1	MS1221	Peery, J. S.	5600
1	MS1328	Cook, B.K.	5624
10	MS1027	Ames, A.L.	5635
1	MS1027	Colbaugh, R.	5635
1	MS1027	Johnson, C.M.	5635
1	MS0620	Horton, R.D.	5640
1	MS0936	Scholand, A.J.	5741
1	MS0724	Hruby, J.M.	6000
1	MS0721	Tatro, M.L.	6100
1	MS1104	Zayas, J.R.	6102
1	MS1104	Robinett, R.D.	6110
1	MS1108	Carlson, J.J.	6111
1	MS1108	Finley, R. E.	6111
1	MS1033	Hanley, C.J.	6112
1	MS1140	Guttromson, R.	6113
1	MS1440	Schoenwald, D.A.	6113
1	MS1188	Nanco, A.S.	6114
1	MS1104	Torres, J.J.	6120
1	MS1124	Minster, D.G.	6121
1	MS1124	Laird, D.L.	6122
1	MS1124	Wilson, D.G.	6122
1	MS0734	Martino, A.	6124
1	MS1188	Griffith, R.O.	6130
1	MS1137	Detry, R.	6131

1	MS1138	Finley, P.D.	6131
1	MS1138	Jones, B.S.	6131
1	MS1138	Jones, D.A.	6131
20	MS1138	Glass, R.J.	6132
1	MS1138	Kleban, S.D.	6132
1	MS1137	Mitchell, M.	6132
1	MS1138	Mitchell, R.	6132
1	MS1138	Moore, T.W.	6132
1	MS1137	Starks, S	6132
1	MS1138	Verzi, S.J.	6132
1	MS1188	Thompson, B.M.	6133
1	MS1004	Skroch, M.J.	6134
1	MS0771	Orrell, S.A.	6200
1	MS1002	Corbett, D.W.	6500
1	MS0781	Nelson, J.E.	6520
1	MS0781	Richardson, C.B.	6525
1	MS0769	Moya, R.W.	6600
1	MS0757	Jordan, S.E.	6612
1	MS0734	Kelley, J.B.	6632
1	MS1375	Wilson, R.K.	6800
1	MS1377	Webb, E.K.	6814
1	MS0701	Walck, M.C.	6900
1	MS0735	Merson, J.A.	6910
1	MS0706	Borns, D.J.	6912
1	MS0751	McKenna, S.A.	6912
1	MS0750	Halloran, A.R.	6913
1	MS0734	Ivey, M.D.	6913
1	MS0754	Rigali, M.J.	6915
1	MS1033	Blankenship, D.A.	6916
1	MS1138	Ammerlahn, H.R.	6920
1	MS1138	Garcia, P.	6920
1	MS1138	Brodsky, N.S.	6921

1	MS1138	McCornack, M.T.	6923
1	MS1138	Beyeler, W.E.	6924
1	MS1138	Brown, T.J.	6924
1	MS1138	Conrad, S.H.	6924
1	MS1138	Corbet, T.F.	6924
1	MS1137	Ehlen, M.A.	6924
1	MS1138	Kaplan, P.G.	6924
1	MS1137	Kelic, A.	6924
1	MS1138	Parrott, L.K.	6924
1	MS1138	Pless, D.J.	6924
1	MS1138	Horschel, D.S.	6925
1	MS1138	Kuzio, S.P.	6926
1	MS1137	Malczynski, L.A.	6926
1	MS1137	Passell, H.D.	6926
1	MS1137	Tidwell, V.C.	6926
1	MS9001	Stulen, R.H.	8000
1	MS0124	Hwang, R.Q.	8004
1	MS9004	Davies, P.B.	8100
1	MS9003	Wu, J.	8112
1	MS9054	Carling, R.W.	8300
1	MS0734	Downes, P.S.	8360
1	MS9054	Pontau, A.E.	8360
1	MS9151	Napolitano, L.M.	8900
1	MS9159	Mayo, J.	8953
1	MS9159	Ray, J.	8954
1	MA9151	Hutchinson, R.L.	8960
1	MS9158	Armstrong, R.C.	8961
1	MS9158	Vanderveen, K.B.	8961
1	MS0805	Cook, W.R.	9530
1	MS0899	RIM-Reports Management	9532 (electronic copy)


Sandia National Laboratories