VANDERBILT UNIVERSITY

INSTITUTE FOR SOFTWARE
INTEGRATED SYSTEMS

Institute for Software-Integrated Systems
Vanderbilt University
Nashville, Tennessee 37203

NEXT-GENERATION COMMAND AND CONTROL WIND TUNNEL FOR COURSES OF ACTION SIMULATION

Himanshu Neema, Gabor Karsai, Alexander Levis

TECHNICAL REPORT

ISIS-15-119

May 21, 2015

TABLE OF CONTENTS

Section **Page**

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

The increasing complexity and heterogeneity of operations necessitate the use of sophisticated planning tools and processes that allow the rapid evaluation of alternative Courses of Action (COA) to give feedback to planners. As the domain of Air Force operations extends across air, space, and cyberspace, the performance of existing processes and their supporting computational tools is not meeting the operators' needs. COA evaluation should be rapid, allowing fast planning and change cycles. While raw computational power is becoming readily available and connectivity is improving, software tools that allow the effective harnessing of that power are lagging behind.

Simulation-based evaluation of COAs is complex, as it involves multiple, heterogeneous domains, each with its own tools and simulations. The configuration and integration of these simulations into a coherent framework, namely, a federation of simulations, is a very difficult, time-consuming, labor-intensive, and error-prone task. Consequently, COA evaluation cannot be done rapidly and in a timely manner to provide answers to the planners. Because a COA has to be tested against a number of scenarios and situations, at most one COA with some minor variants is analyzed. The problem becomes more acute as the focus is shifting to integrated C2 where several COAs developed by different Command Centers (e.g., the different command centers for Air, Space, and Cyber Operations) have to be integrated and the result evaluated. Designing and efficiently deploying such a computational resource on a high-performance computing platform is a major challenge.

In the course of the work described in this report, our team has devised, designed, constructed, and demonstrated a novel software framework and tool environment to support the evaluation of operational sequences that are prepared by and derived from COA by human experts. The framework provides (a) a model-based development environment for rapidly composing and integrating heterogeneous simulation models and constructing models for operational sequences; and (b) a run-time execution environment for executing all the simulation models in a coordinated manner while constantly interacting with the operational sequence models. This toolsuite provides a novel environment for integrating pre-existing models of (dynamic) systems, and performing experiments under the control of various operational sequences.
The tools have been evaluated using scenarios of events unfolding in the world of Pacifica: an island with three nations. A Colored Petri Net models have been created that simulated organizational and decision making process followed by the nations, and OMNeT++ models have been created for the interlinked computer networks of the simulated nations. The two models interacted with each other: human processes relied on networks for communication. Operational sequence models represented different cyber-operations that interfered with the (simulated) networks, thus impacting the (simulated) human processes leading to different outcomes.

2.0 INTRODUCTION

The quantitative analysis and evaluation of COAs is a daunting problem. Given the complexity of the operational environment in which the Air Force conducts operations, commanders need the capability to consider multiple alternative courses of actions and assess their expected effects in light of the resources needed for their execution. An additional complication is the challenge of integrated C2 in which the selected COA must work synergistically with the COAs produced by other component commanders. Cyber security considerations will also affect the choice of COA.

The results of the work summarized in this report address the challenge of developing an integrated, easily composable modeling and simulation environment to create realistic cyber warfare operational scenarios for course of action (COA) exploration. The approach is based on a new generation C2 Wind Tunnel by enhancing the already existing C2 Wind Tunnel (developed under AFOSR funding) through the addition of several software tools and a modeling language for operational sequences.

The project entailed the following activities:

1. Analysis of the problem domain, i.e., the simulation based evaluation of COAs, covering information operations in general and cyber operations specifically.
2. Design and implementation of a novel domain-specific modeling to allow the rapid integration and configuration of CASIM-based COA evaluations. The evaluation architecture is captured in the form of graphical models that can be rapidly composed from pre-fabricated building blocks. The goal of the language is (1) to simplify the integration activities, and (2) to support the modeling of operational sequences derived from COA specifications, by experts.
3. Development of software generators for the configuration and deployment of the simulations on a (potentially distributed) platform: the C2 Wind Tunnel software infrastructure.
4. Identification and integration of a specific set of simulation and analysis tools for CASIM that are focused on the cyber mission.
5. Development of tools for experiment control. The goal of this task was to develop the infrastructure to deploy and run CASIM experiments in a highly controlled and repeatable manner.

The report below summarizes the project activities: describes the methods and procedures followed, as well as the assumptions made.

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

Our objective with this project was to develop methods and software tools for the evaluation of Courses of Actions (COAs). Our project was conducted in conjunction with a parallel effort by BAE Systems, with responsibilities split up between the two organizations as follows.

1. BAE Systems is responsible for (1) the preparatory activities of COA evaluation, and (2) the post-processing of the raw computational results into a meaningful Measures of Effectiveness (MoEs).
2. Vanderbilt/ISIS is responsible for the modeling and design tools and the run-time computational infrastructure for performing the evaluation.

In this report we focus on the Vanderbilt/ISIS activities of the project.

Based on our studies of the relevant, open literature and the communication with experts, we have concluded that the problem of evaluating Courses of Action necessitates a human involvement in the process, and it cannot be completely automated. The reason for this is the very high-level nature of COAs: they do not have the necessary operational details for computational evaluation. In fact, as far as we understand, COAs today are mostly evaluated by human teams, relying on human expertise and experience.

While we had to recognize this admittedly negative observation early in the course of the project, we have designed a method for COA evaluation that is not completely automated. The method can be summarized as follows:

Human experts develop operational sequence models (OSM) based the (high-level) COA-s and evaluate these using a set of federated of simulation models.

OSMs represent specific operational *actions* to be executed and expected *outcomes* to be observed.

3.1 Assumptions

There are several assumptions in this method:

1. High-quality simulation models *are* available for the relevant aspects of the systems involved. For instance, if the objective of a COA evaluation is to gain insight into what will happen with the Command and Control (C2) system under different operational scenarios then simulation models are needed to represent (a) the human organizational and decision making processes involved, (b) the computer network that supports those decision making processes, (c) the cyber effects that the network can be subjected to, and (d) the integration of the above.
2. The simulation models *are* capable of reacting to operational actions and generating outcomes as described in the OSM. In other words, the simulation models have the correct input and output interfaces through which they accept control commands (to change behavior) and provide status reports (to report specific events).
3. Operational actions can be mapped into control commands (to be sent to the simulation models), and expected outcomes can be detected by receiving status reports (from the running simulation model).

4. The simulation models are dynamic models, i.e. they simulate the behavior of some system over time. Each simulation model must be able to progress forward in time, i.e. given the current state of the model and the current input at time t, the simulation engine computes the state and the output of the model at some later time t'.

3.2 Conceptual Architecture

Our conceptual architecture for evaluating COA-s through the use of simulation models and operational sequence models is as follows.

- Each model simulates the dynamic behavior of some subsystem in the larger scenario. For instance: (1) a Colored Petri Net (CPN) simulation model can simulate how human decision making processes take place, how they unfold in time, and how they use a computer network; while (2) a OMNeT++ (network simulator) model simulates how a computer network operates, including what information flows are blocked, delayed, or modified under cyber effects.
- The simulation models interact with each other, as the simulated scenario unfolds. For example, the CPN model sends and receives simulated messages that are processed (possibly blocked, delayed, and modified) by the OMNeT++ model, as the simulated time progresses.
- The operational sequence model (OSM) represents an operational flow that includes *actions* (that, for instance, instruct the OMNeT++ model to start blocking a simulated network flow) and *responses* (that, for instance, are generated by the CPN model to indicate that certain decisions have been made).
- The interacting simulation models capture a representative scenario in the lifetime of a 'system' that is the target of the COA, simulated by the execution of the OSM.
- The OSM does not require a separate simulation engine, as it represents COA-specific interactions with the simulation models.
- The same simulation models (and thus scenario) can be used with different OSM-s (potentially derived from different COA-s).

The figure below illustrates the conceptual architecture.

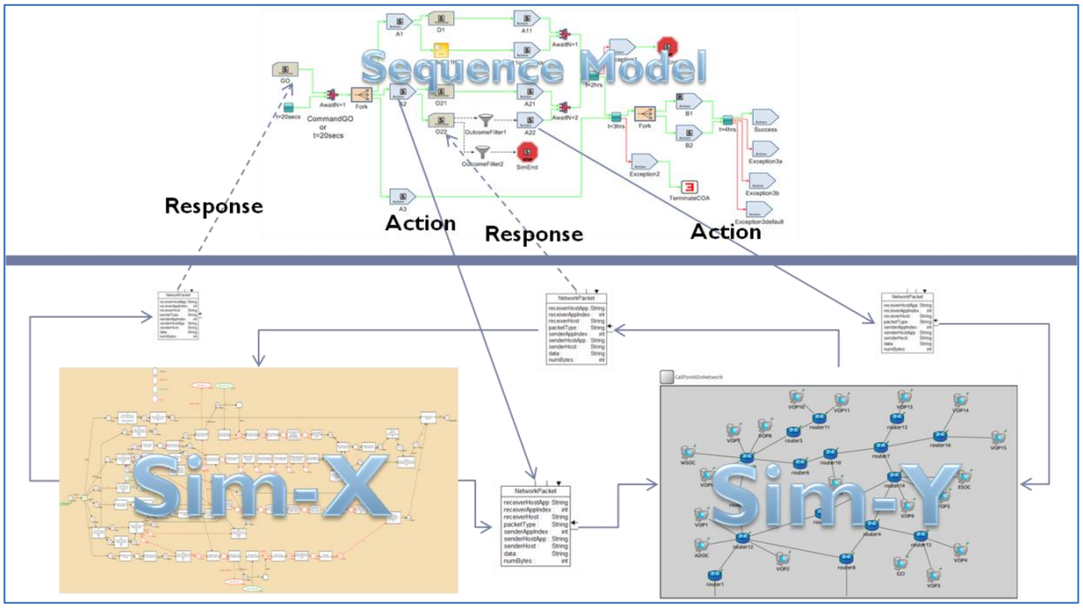


Figure 1: Conceptual Architecture

3.3 Procedures

The procedure to perform a COA evaluation is as follows:

1. Domain-specific simulation models are prepared that model the various subsystems.
2. The interactions of these models are designed in the context of a scenario.
3. Operational Sequence Models are created for the COA(s) that are to be evaluated. This is performed by an expert.
4. The simulation models are integrated and executed in an integrated simulation framework. During execution, the OSMs initiate specific actions and anticipate responses from the simulation models.
5. Data is collected and analyzed to determine the effectiveness of the OSM that was derived from the COA. Note: the analysis work was performed by BAE Systems and will not be further discussed here.

Our effort focused on designing and developing the software infrastructure to support the above conceptual architecture. For this, we have relied on results from a previous project on the Command and Control Wind Tunnel (C2WT)[1]. We have extended these results and added new capabilities specifically for the purposes of CASIM. These results are described in the next section.

3.4 Objectives

To summarize, the objectives of the project were as follows:

1. Develop technology for the rapid integration of heterogeneous domain-specific simulation models.

[1] AFOSR/PRET: Human Centric Design Environments for Command and Control Systems: The C2 Wind Tunnel, FA9550-06-1-0267

2. Develop technology for the rapid modeling of operational sequences that capture salient aspects of a COA.
3. Develop technology for the execution of integrated simulations under the control of and in interaction with the operational sequences.
4. Develop technology for the collecting data from the simulation runs and make it available for further analysis.

The next section summarizes our results.

4.0 RESULTS AND DISCUSSION

The C2WT has been developed in an AFOSR/PRET project of Vanderbilt, UC Berkeley and GMU for the rapid evaluation and assessment of C2 concepts and system designs in a human-centered environment. The key insight of the project has been that integration of heterogeneous, multi-model simulations can be decomposed into a simulation integration and model integration problem. While simulation integration is a well understood problem and DoD’s High Level Architecture (HLA) provides a sound framework for composing simulations based on discrete event semantics, model integration has not been sufficiently addressed. The primary outcome of the C2WT project has been a meta-modeling based model–integration framework. The framework includes a model integration language for capturing the interaction among component models and provides tools for generating “glue code” from the integration model to couple the individual simulation tools to the HLA runtime infrastructure. While the C2WT framework described above is functional and has been demonstrated in a range of applications for some time, several additional features were required to enable its use in large-scale COA testing and evaluation. These features were based on the objectives described above. The extended C2WT framework is called the ‘Next Generation C2WT’ (i.e. the title of the project).

4.1 Method for integration

One objective of the project has been to enable the rapid integration of simulation models. To support this, we developed and tested a method for integration that allows a high degree of reusability and allows quick integration. The method works as follows.

- Each simulation model has a dedicated simulation engine (that performs the calculations) and the model data itself (that determines what those calculations are).
- Each simulation engine is integrated strictly once, and this integration involves developing the adaptor code modules (i.e. the ‘wrappers’) that connect the engine to the HLA framework. This integration means that messages (‘interactions’ in the HLA) can be sent to and received from the simulation engine as it is executing. These messages are produced and consumed by other simulators in the HLA federation.
- The model data may change between simulation models (even if the same engine is used), so that is integrated on a case-by-case basis. Integration of the specific model involves specifying how to translate messages between simulators. This is done with the help of a ‘pseudo-simulator’: an HLA federate that acts as phone exchange between the simulation engines that can translate the messages.

As an example, let’s consider two simulators X and Y that need to talk to each other, and each has a corresponding model data set: A and B, respectively. Both can produce and consume certain *type* of messages and the HLA wrapper consumes and produces these messages. Now if X wants to exchange messages with Y and these messages may be dependent on the model data (i.e. A and B), there has to be a ‘mapper’ that translates the messages in both directions. Note that this translator may have to be configured according to the A and B models. However, configuring the mapping is much simpler than writing (or modifying) an HLA wrapper. The figure below illustrates the example instantiation of the architecture with the two simulators and the mapper.

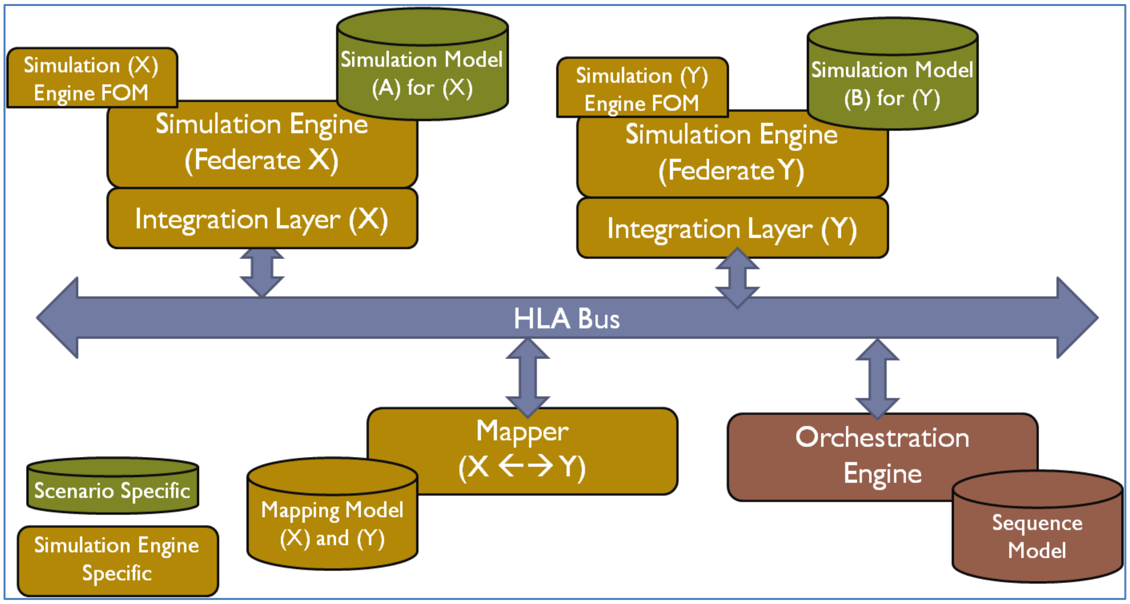

Simulation (X) Engine FOM
Simulation Model (A) for (X)
Simulation Engine (Federate X)
Integration Layer (X)
Simulation (Y) Engine FOM
Simulation Model (B) for (Y)
Simulation Engine (Federate Y)
Integration Layer (Y)
HLA Bus
Mapper (X ←→ Y)
Orchestration Engine
Scenario Specific
Simulation Engine Specific
Mapping Model (X) and (Y)
Sequence Model

Figure 2: Next Generation C2WT Architecture - Example

Note that for large configurations (more than 2 simulators) there could be multiple mappers. Additionally, the simulators and the mapper can run on different nodes of a distributed network, providing higher performance.

We claim that the above architecture solves the rapid integration problem, because (1) simulation engines are integrated once (and for all), and (2) if there are any variations in the model data, then those can be handled in the mapper component that handles the message translations.

4.2 Sequence models and the Orchestration Engine

Note also the component on the right lower side: the Orchestration Engine (OE). This is not a separate federate, rather part of the main HLA federate: the 'Federation Manager' that controls the overall execution of the federation of simulations. The task of the Orchestration Engine is to execute Operational Sequence Models (or sequence models, for short).

Sequence models are created using the GME modeling tool, using an extended form of the C2WT modeling language. In the course of the predecessor AFOSR project we have defined a domain-specific modeling language for modeling the C2WT configuration. For CASIM, we have extended that modeling language with new features for (1) modeling sequences, and (2) modeling multiple experiments (to be discussed later). The figure below shows an example sequence model. The language for sequence models is described in detail in the Appendix.

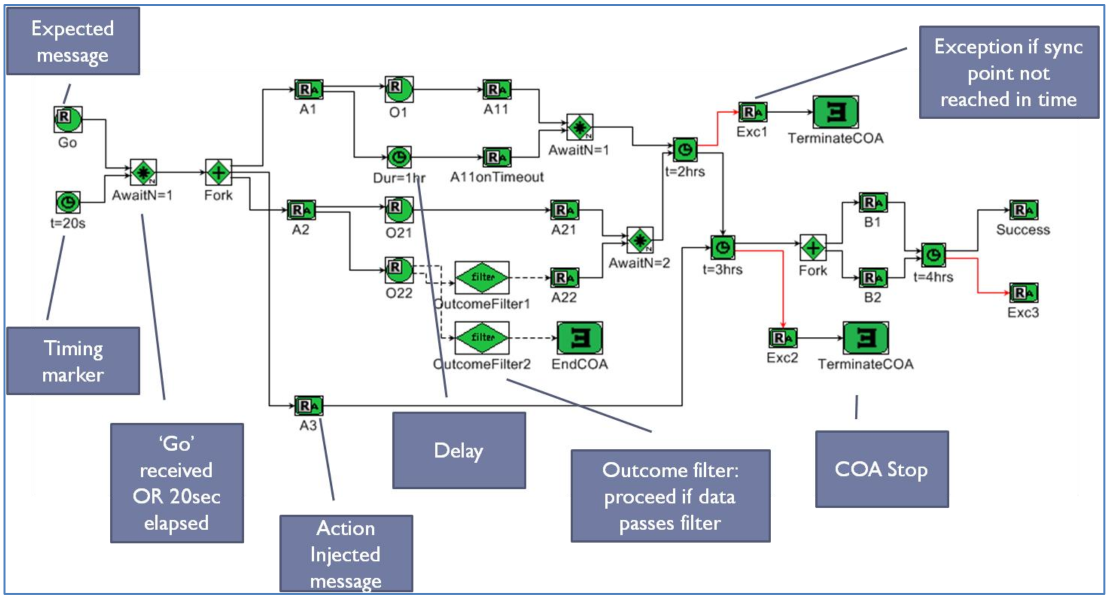
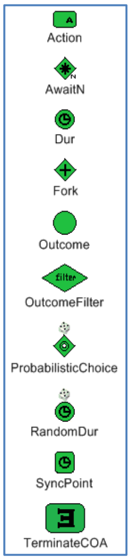


Figure 3: Example sequence model

The sequence model is similar to a workflow model; in fact, the icons are the same as that of the Business Process Modeling Notation – BPMN[2]. The figure on the side shows the icon palette. The black arrows represent the normal flow of events, while red arrows indicate exceptional cases. Note the temporal operators: (1) Dur(ation) and (2) Sync(hronization)Point – these can be used to represent the facts that (1) some time must elapse before proceeding with the workflow, and (2) the execution must reach a certain stage by a certain time (from the start) otherwise an exception is generated.



The sequence model has a precisely defined executable semantics – and it is the same as for BPMN, with two exceptions: (1) 'Action' means: 'Send a message of a certain type with a specific content to federation of the simulation models'. (2) 'Outcome' means: 'Wait here for the arrival of a message of a certain type from the simulation models'. Hence the sequence model can be enacted by an execution engine: the Orchestration Engine (OE) above that interacts with the federation of simulation models while executing the workflow; the engine sends and receives messages as the simulated scenario unfolds. Note that the HLA federation has a global simulation clock which is used by the OE to keep track of time, wait for events until a timeout, and control, in general, the temporal behavior.

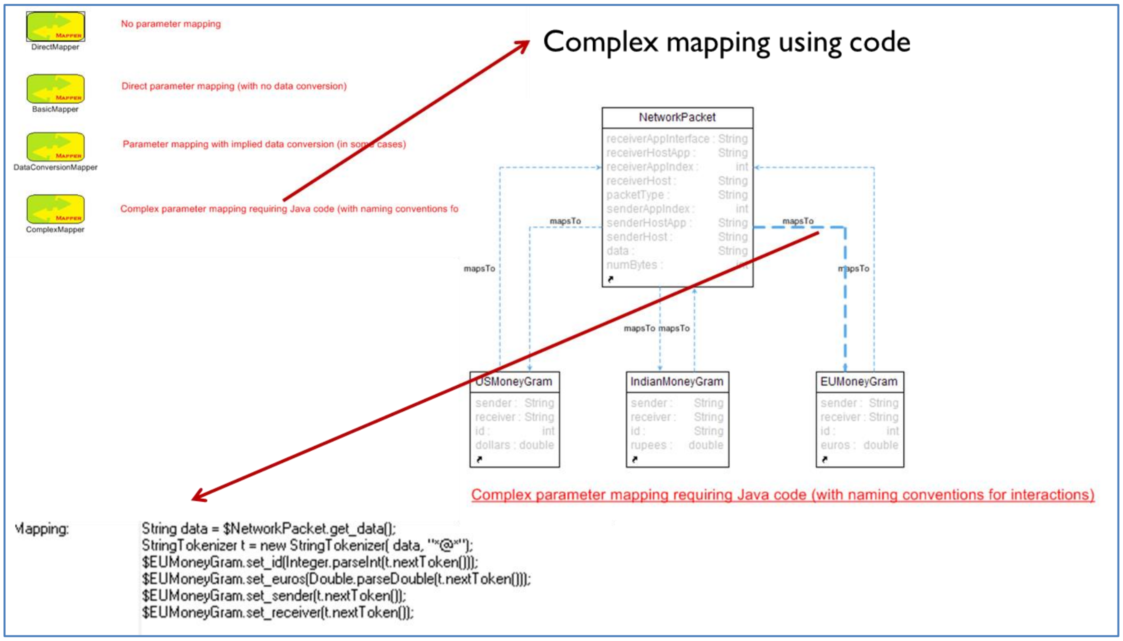
4.3 The Mapper

While the sequence models and the OE are essential for performing the evaluation task, the Mapper component is also a critical component: it acts as the universal translator between the different federates containing the different simulation models. In the new, 'Next Generation C2WT' a simulation engine is integrated into the framework. This means that the adaptor code that connects a simulation engine to the HLA bus is developed once and reused across multiple

[2] http://www.bpmn.org/

models, without change. This is a necessity, as adaptor development is a non-trivial task, and requires effort. Each adaptor henceforth defines a set of message (interaction) types the simulation engine understands and produces – this set is independent of the specific model used in the engine, and is fixed by the adaptor developer. As a result, each simulation federate has a well-defined interface, in the form of these messages that they are able to send and receive. However, in a particular scenario involving multiple federates, the messages should undergo a translation process for the simulation models to communicate. This translation process is performed by the Mapper.

The Mapper federate is specific to an ensemble of simulation models, and possibly to a scenario. In order to ease the development of the Mapper, we created a modeling tool and software generator to model the message translation operations and to generate the complete executable Mapper federate from the models. The models simply capture the mapping between message types: they are essentially short specifications for transcribing messages from one form to another. The models of the mapping range from simple mappings (where messages are isomorphic and thus can be directly mapped) to more complex cases (where custom Java code is needed to transcribe data). The Mapper is described in detail in the Appendix. Here (on the figure below) we just give an example for a model of the most complex case where custom code is needed.



```
Mapping:    String data = $NetworkPacket.get_data();
            StringTokenizer t = new StringTokenizer( data, "*@*");
            $EUMoneyGram.set_id(Integer.parseInt(t.nextToken()));
            $EUMoneyGram.set_euros(Double.parseDouble(t.nextToken()));
            $EUMoneyGram.set_sender(t.nextToken());
            $EUMoneyGram.set_receiver(t.nextToken());
```

Figure 4: Example for a complex mapping

The Mapper models support one-to-one, one-to-many, many-to-one, and many-to-many translation relationships between messages.

4.4 Specific simulation engines integrated

In the course of the project, we have integrated the following simulation engines.

1. Colored Petri Net Tools (CPNTools). We have developed a generic adaptor for CPN Tools and a model-driven configuration mechanism to integrate specific CPN models into the new C2WT platform. The generic adaptor contains all the code needed to execute CPN models. The model-driven configuration mechanism provides support as follows: a CPN model can be imported into the C2WT modeling tool, and if the CPN model followed certain conventions, model specific message types and model elements are automatically generated. These are then used to configure the integrated simulation scenario (including the Mapper and other simulation engines).
2. OMNeT++, a network simulator. We have developed a generic adaptor and corresponding Simulation Object Model (SOM) for the network simulator. The SOM supports basic message exchange (sending and receiving network datagrams through the simulated network), as well as turning on and off various cyber effects. We have developed simulation models for these cyber effects in OMNeT++: disable network, disable a specific node on the network, replay: capture packets and replay them, scramble the order of packets flowing through a node, sniffer: capture packets in flight, modify route, delay node: slow down specific host, network filter: block specific packets going through a node. These cyber effects are available as a simulation module that can be activated by an appropriate message sent to the simulator.

4.5 Mode of operation

Once the C2WT models for the integrated federate of the simulations are created (including the models for the Mapper), the collection of Sequence Models have to be prepared using the modeling approach briefly described above. As the Sequence Models can represent the COAs of multiple participants (typically: ‘blue’ and ‘red’), and there could be many variants of them, we have built support for modeling such variability. Using the same simulation models (e.g. CPN models and OMNeT++ models) one can create several variants of Sequence Models representing the operations of various parties (e.g. ‘red’ and ‘blue’) and group them together in experiment models. Note that COA evaluation may require the execution of various operational sequences, of different participants, and all the possible combinations of those. A graphical modeling language is available to model such combinations in a compact form – these models are then translated into scripts that perform the complete sequence of experiments, in an automated mode. Having prepared the models for the C2WT, software generators produce all the artifacts (executable code, configuration files, scripts, etc.) needed to run the experiments. Experiments, i.e. simulation runs under the control of the sequence models, can be executed in interactive mode or in batch mode. In interactive mode the evaluator starts and controls a single execution of the models, potentially interacting with the simulation models as they run. In batch mode, everything is automatic, and no interaction with the evaluator is needed (or possible). In either case, the results are collected in a database, in the form of logs of messages produced during execution. For this the, the C2WT logging facilities are used. The figures below illustrate an interactive run of an experiment.

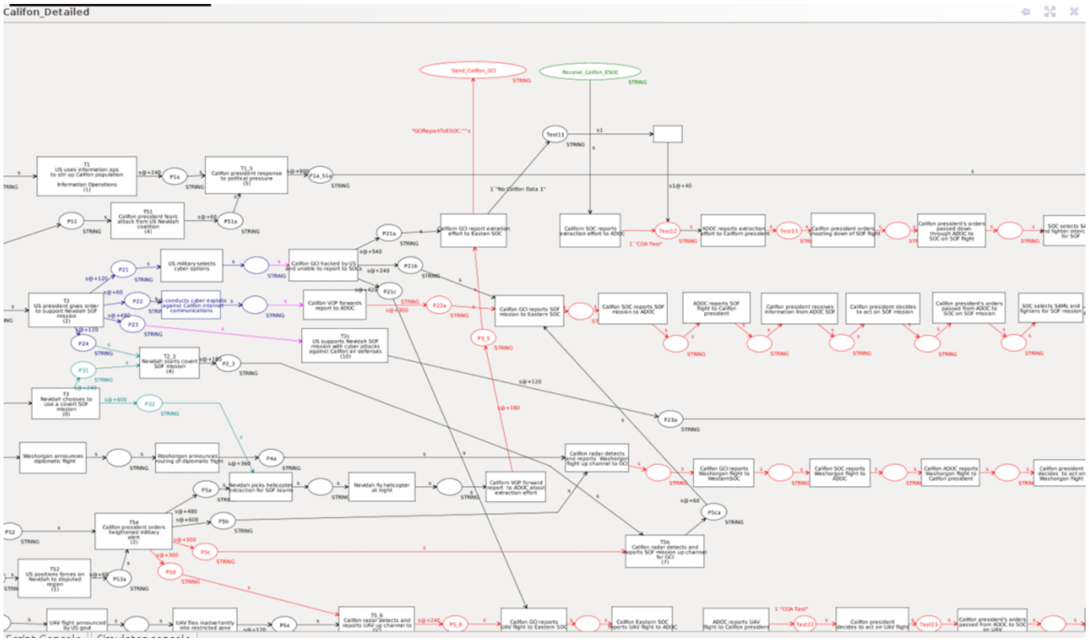
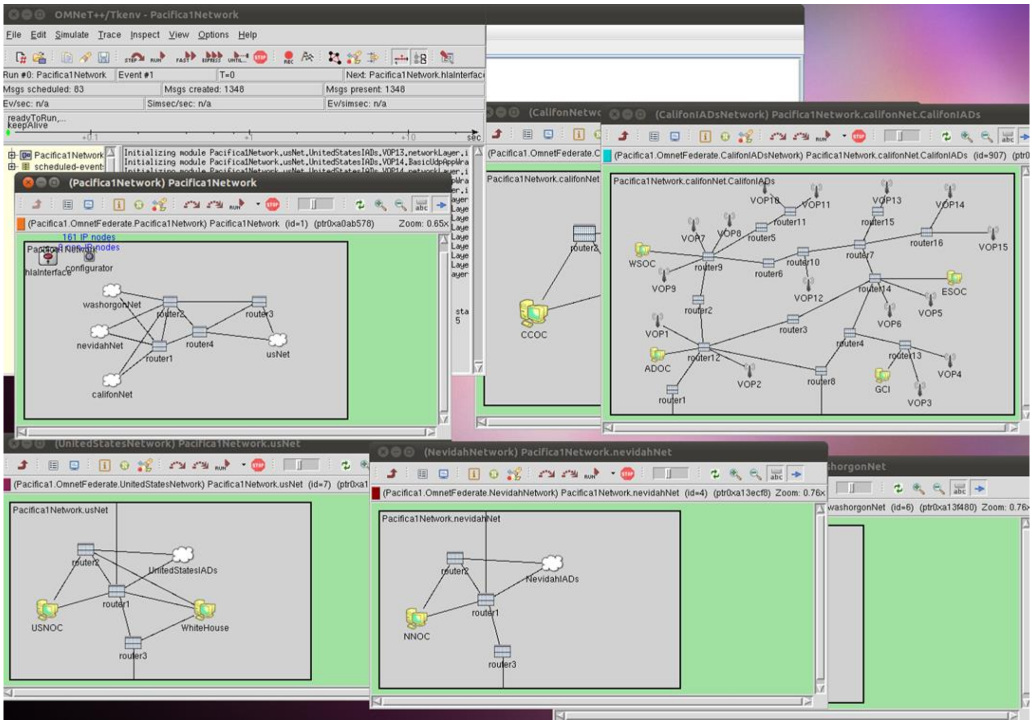

Califon_Detailed
Send_Califon_GCI
Receive_Califon_ESOC
US uses information ops to stir up Califon population Information Operations
Califon president response to political pressure
Califon president fears attack from US Nevidah coalition
US military selects cyber options
US president gives order to support Nevidah SOF mission
Nevidah starts covert SOF mission
Nevidah chooses to use a covert SOF mission
Washorgon announces diplomatic flight
Califon GCI report extraction effort to Eastern SOC
Califon SOC reports extraction effort to ADOC
ADOC reports extraction effort to Califon president
Califon president orders shooting down of SOF flight
SOC selects SAMs and fighter interceptors for SOF
Califon VOP forwards report to ADOC
US supports Nevidah SOF mission with cyber attacks against Califon air defenses
Califon GCI reports SOF mission to Eastern SOC
Califon SOC reports SOF mission to ADOC
ADOC reports SOF flight to Califon president
Califon president receives information from ADOC SOF
Califon president decides to act on SOF mission
Califon president's orders passed from ADOC to SOC on SOF mission
SOC selects SAMs and fighters for SOF mission
Califon radar detects and reports Washorgon flight
Califon GCI reports Washorgon flight to Western SOC
Califon SOC reports Washorgon flight to ADOC
Califon ADOC reports Washorgon flight to Califon president
Califon president decides to act on Washorgon flight
Nevidah picks helicopter extraction for SOF team
Nevidah fly helicopter at night
Califon VOP forward report to ADOC about extraction effort
Califon president orders heightened military alert
US positions forces on Nevidah to disputed region
Califon radar detects and reports SOF mission up channel for GCI
UAV flight announced by US gov
UAV flies inadvertently into restricted zone
Califon radar detects and reports UAV up channel to
Califon GCI reports UAV flight to Eastern SOC
Califon Eastern SOC reports UAV flight to ADOC
ADOC reports UAV flight to Califon president
Califon president decides to act on UAV flight
Califon president's orders passed from ADOC to SOC on UAV
STRING

Figure 5: Example CPN Model

The CPN model above is was developed for one of the scenarios described in the next section.



Run #0: Pacifica1Network | Event #1 | T=0 | Next: Pacifica1Network.hlaInterfac

Msgs scheduled: 83 | Msgs created: 1348 | Msgs present: 1348

Ev/sec: n/a | Simsec/sec: n/a | Ev/simsec: n/a

readyToRun,...
keepAlive

```
Initializing module Pacifica1Network.usNet.UnitedStatesIADs.VOP13.networkLayer.i
Initializing module Pacifica1Network.usNet.UnitedStatesIADs.VOP14.BasicUdpAppWra
```

(Pacifica1.OmnetFederate.Pacifica1Network) Pacifica1Network (id=1) (ptr0xa0ab578) Zoom: 0.65x

(Pacifica1.OmnetFederate.CalifonIADsNetwork) Pacifica1Network.califonNet.CalifonIADs (id=907)

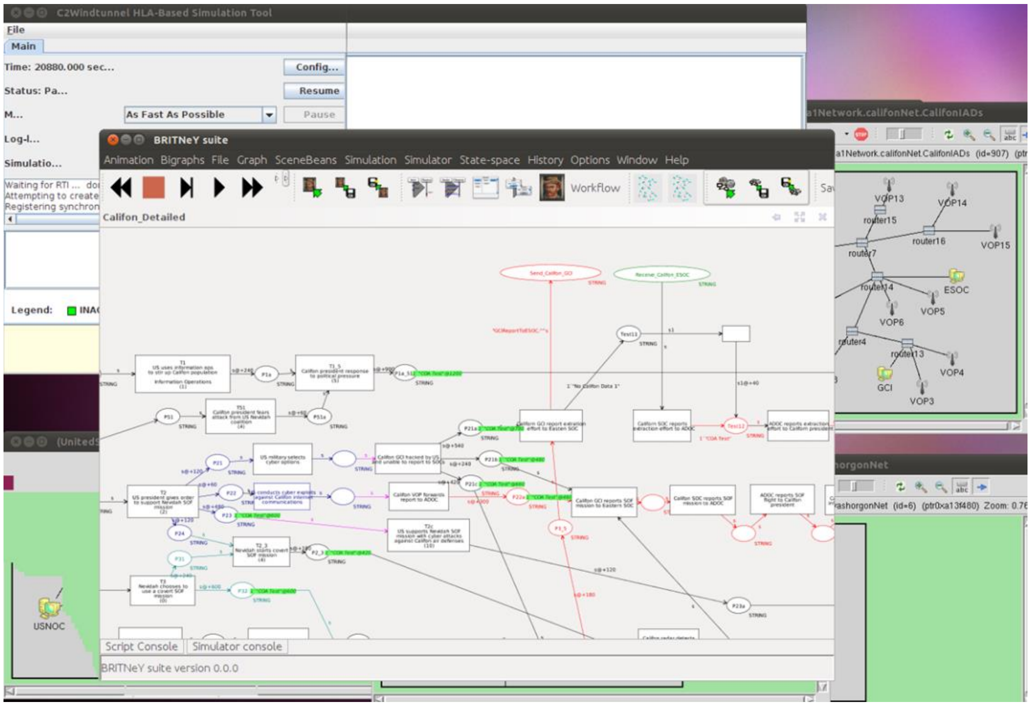
(Pacifica1.OmnetFederate.UnitedStatesNetwork) Pacifica1Network.usNet (id=7)

(Pacifica1.OmnetFederate.NevidahNetwork) Pacifica1Network.nevidahNet (id=4) (ptr0xa13ecf8) Zoom: 0.76x

washorgonNet (id=6) (ptr0xa13f480) Zoom: 0.76x

Figure 6: Example OMNeT++ model

The above model shows the networks used in the scenario.



Time: 20880.000 sec...
Status: Pa...
M... As Fast As Possible
Log-l...
Simulatio...
Waiting for RTI ... do
Attempting to create
Registering synchron
a1Network.califonNet.CalifonIADs (id=907)
ashorgonNet (id=6) (ptr0xa13f480) Zoom: 0.76
BRITNeY suite version 0.0.0

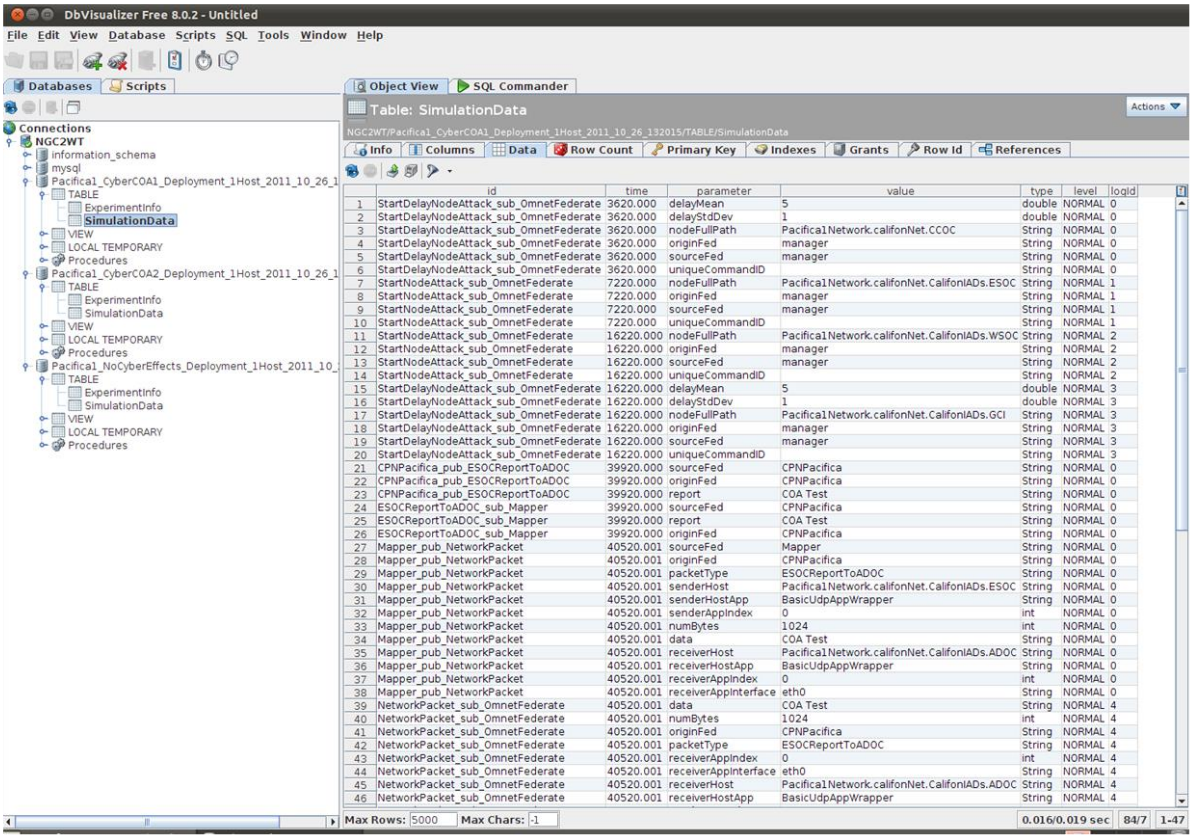
Figure 7: CPN and OMNeT++ models executing

The above figure shows the two models executing concurrently.

4.6 Experiment execution and analysis

As described above, the software infrastructure supports the modeling of multiple operational sequences and the execution of the combinations of these sequences. To assist with this activity, an experiment controller tool has been developed. The tool is used after the models were processed by the software generators, and the tool shows, on a GUI, all the possible experiments that were generated from a model, and allows the interactive selection and batch-mode execution of multiple experiments.

The results of the experiment are captured in an SQL database. The results are message logs: selected interactions are logged (with content and time-stamp) in specific tables. Any standard SQL tool can be used for post-processing and analysis. Our project did not develop analytical techniques, but provided that facilities for raw data collection and archival. The figure below illustrates the content of database after an experimental run.



	id	time	parameter	value	type	level	logId
1	StartDelayNodeAttack_sub_OmnetFederate	3620.000	delayMean	5	double	NORMAL	0
2	StartDelayNodeAttack_sub_OmnetFederate	3620.000	delayStdDev	1	double	NORMAL	0
3	StartDelayNodeAttack_sub_OmnetFederate	3620.000	nodeFullPath	Pacifical Network.califonNet.CCOC	String	NORMAL	0
4	StartDelayNodeAttack_sub_OmnetFederate	3620.000	originFed	manager	String	NORMAL	0
5	StartDelayNodeAttack_sub_OmnetFederate	3620.000	sourceFed	manager	String	NORMAL	0
6	StartDelayNodeAttack_sub_OmnetFederate	3620.000	uniqueCommandID		String	NORMAL	0
7	StartNodeAttack_sub_OmnetFederate	7220.000	nodeFullPath	Pacifical Network.califonNet.CalifonIADs.ESOC	String	NORMAL	1
8	StartNodeAttack_sub_OmnetFederate	7220.000	originFed	manager	String	NORMAL	1
9	StartNodeAttack_sub_OmnetFederate	7220.000	sourceFed	manager	String	NORMAL	1
10	StartNodeAttack_sub_OmnetFederate	7220.000	uniqueCommandID		String	NORMAL	1
11	StartNodeAttack_sub_OmnetFederate	16220.000	nodeFullPath	Pacifical Network.califonNet.CalifonIADs.WSOC	String	NORMAL	2
12	StartNodeAttack_sub_OmnetFederate	16220.000	originFed	manager	String	NORMAL	2
13	StartNodeAttack_sub_OmnetFederate	16220.000	sourceFed	manager	String	NORMAL	2
14	StartNodeAttack_sub_OmnetFederate	16220.000	uniqueCommandID		String	NORMAL	2
15	StartDelayNodeAttack_sub_OmnetFederate	16220.000	delayMean	5	double	NORMAL	3
16	StartDelayNodeAttack_sub_OmnetFederate	16220.000	delayStdDev	1	double	NORMAL	3
17	StartDelayNodeAttack_sub_OmnetFederate	16220.000	nodeFullPath	Pacifical Network.califonNet.CalifonIADs.GCI	String	NORMAL	3
18	StartDelayNodeAttack_sub_OmnetFederate	16220.000	originFed	manager	String	NORMAL	3
19	StartDelayNodeAttack_sub_OmnetFederate	16220.000	sourceFed	manager	String	NORMAL	3
20	StartDelayNodeAttack_sub_OmnetFederate	16220.000	uniqueCommandID		String	NORMAL	3
21	CPNPacifica_pub_ESOCReportToADOC	39920.000	sourceFed	CPNPacifica	String	NORMAL	0
22	CPNPacifica_pub_ESOCReportToADOC	39920.000	originFed	CPNPacifica	String	NORMAL	0
23	CPNPacifica_pub_ESOCReportToADOC	39920.000	report	COA Test	String	NORMAL	0
24	ESOCReportToADOC_sub_Mapper	39920.000	sourceFed	CPNPacifica	String	NORMAL	0
25	ESOCReportToADOC_sub_Mapper	39920.000	report	COA Test	String	NORMAL	0
26	ESOCReportToADOC_sub_Mapper	39920.000	originFed	CPNPacifica	String	NORMAL	0
27	Mapper_pub_NetworkPacket	40520.001	sourceFed	Mapper	String	NORMAL	0
28	Mapper_pub_NetworkPacket	40520.001	originFed	CPNPacifica	String	NORMAL	0
29	Mapper_pub_NetworkPacket	40520.001	packetType	ESOCReportToADOC	String	NORMAL	0
30	Mapper_pub_NetworkPacket	40520.001	senderHost	Pacifical Network.califonNet.CalifonIADs.ESOC	String	NORMAL	0
31	Mapper_pub_NetworkPacket	40520.001	senderHostApp	BasicUdpAppWrapper	String	NORMAL	0
32	Mapper_pub_NetworkPacket	40520.001	senderAppIndex	0	int	NORMAL	0
33	Mapper_pub_NetworkPacket	40520.001	numBytes	1024	int	NORMAL	0
34	Mapper_pub_NetworkPacket	40520.001	data	COA Test	String	NORMAL	0
35	Mapper_pub_NetworkPacket	40520.001	receiverHost	Pacifical Network.califonNet.CalifonIADs.ADOC	String	NORMAL	0
36	Mapper_pub_NetworkPacket	40520.001	receiverHostApp	BasicUdpAppWrapper	String	NORMAL	0
37	Mapper_pub_NetworkPacket	40520.001	receiverAppIndex	0	int	NORMAL	0
38	Mapper_pub_NetworkPacket	40520.001	receiverAppInterface	eth0	String	NORMAL	0
39	NetworkPacket_sub_OmnetFederate	40520.001	data	COA Test	String	NORMAL	4
40	NetworkPacket_sub_OmnetFederate	40520.001	numBytes	1024	int	NORMAL	4
41	NetworkPacket_sub_OmnetFederate	40520.001	originFed	CPNPacifica	String	NORMAL	4
42	NetworkPacket_sub_OmnetFederate	40520.001	packetType	ESOCReportToADOC	String	NORMAL	4
43	NetworkPacket_sub_OmnetFederate	40520.001	receiverAppIndex	0	int	NORMAL	4
44	NetworkPacket_sub_OmnetFederate	40520.001	receiverAppInterface	eth0	String	NORMAL	4
45	NetworkPacket_sub_OmnetFederate	40520.001	receiverHost	Pacifical Network.califonNet.CalifonIADs.ADOC	String	NORMAL	4
46	NetworkPacket_sub_OmnetFederate	40520.001	receiverHostApp	BasicUdpAppWrapper	String	NORMAL	4

Figure 8: Example SQL database showing experiment results

5.0 THE CASIM PACIFICA SCENARIO

In order to test and demonstrate the capabilities of the Next Generation C2 Wind Tunnel, a scenario was developed based on the unclassified Pacifica scenario [1]. Two specific scenarios were developed; Scenario 1 was focused on the Deterrence phase (Phase 1) and Scenario 2 on Seizing the Initiative (Phase 2).

5.1 Pacifica Overview

The island of Pacifica contains three sovereign countries: (a) The Confederation of Washorgon States, (b) The Republic of Nevidah, and (c) The Peoples Republic of Califon that has unitary system with no defined state administrative boundaries. This is shown on Figure 9 below. The topography, trafficability, climatology, and surface lines of communication of the countries of Califon, Nevidah, and Washorgon match those of real-world geographic entities. Thus the island reflects diverse terrain, soil, and weather characteristics as well as abundant natural resources. In the north and northwest, lush mountains and moderate climates are the norm. Further south, the mountain chain and arid high desert prevail. In the southeast, the mountains are less extensive, but extremely dry conditions limit most forms of agricultural and industrial development. The island's population consists of immigrants from North America (35% of the population), South America (25%), Asia (20%), and Europe (15%). Five percent are native Pacific Islanders. Washorgon and Nevidah are organized as federal systems with states that have defined regional powers and responsibilities**.**

Califon is a Regional Hegemon in long-standing conflict with Nevidah over minerals and other economic issues. Nevidah is in mutual defense arrangement with Pacific nations including the USA. Washorgon traditionally maintains neutrality with Califon and Nevidah due to trade relationships and access to port facilities in Califon and Nevidah. The year is 2022; the Pacifica mineral fields are proving to be a great natural resource of rare minerals (the other major source is China). Califon has been conducting a campaign against Nevidah to obtain exclusive control of the mineral fields. The Hegemonic regional competitor, Califon, seeks to limit US influence on Nevidah and the ability of US to provide assurance to Nevidah by exploiting the dependence of US forces on spectrum & cyber space.

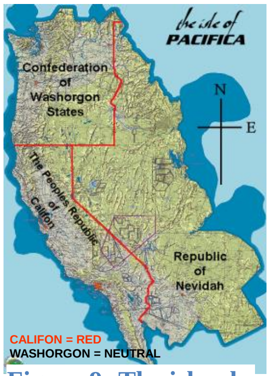


Figure 9: The island of Pacifica

Califon, with technical support from China, has established a Cyber Ops Center on the mountainous northwestern part of the country. The Califon Cyber Ops Center (CCOC) is a secure underground facility in an isolated area that is not easily accessible or targetable from the air. The Cyber Ops Center has been conducting operations that are disrupting the economic life of Nevidah: disrupting the financial sector, disrupting the SCADA systems, conducting extensive identity theft. Up to this point, Califon has not attacked any of the military systems, or, to be more precise, no abnormal behaviors have been identified by Nevidah's Network Operations Center (NNOC). The Califon Cyber warfare against Nevidah has been accelerating; Nevidah in consultation with the US is prepared to take action to persuade the Califon president to cease the cyber attacks and start serious negotiations to resolve the dispute about mineral rights. The United Nations has imposed sanctions on Califon in response to Nevidah's protests.

5.2 Phase 1: Deter Califon aggression via cyber operations

The Commander's mission can be described as follows: US Joint Force Commander will develop full spectrum courses of action (COAs) in support of Nevidah to cause Califon to cease its cyber campaign and deter it from further aggressive actions. Should deterrence fail and Califon invades Nevidah to occupy the Mineral Fields region (Phase 2), the Joint Force Commander will be: (a) prepared to defend US and Nevidah forces and interests in the Pacifica theater, (b) defeat Califon offensive operations, and (c) ensure effective Command, Control, Communications, Computers and Intelligence (C4I) throughout the theater. On order, US and Nevidah forces will repel Califon forces from Nevidah and restore Nevidah's sovereignty. The applicable constraints and restraints are that operations should minimize risk to non-combatants and must minimize risk to US forces, all actions must comply with international law, and no actions can infringe upon Washorgon's neutral status.

The Phase 1 scenario starts with Califon contesting Nevidah mineral rights with cyber warfare. As a result, Nevidah requests US assistance and the US deploys Air Force aircraft to Nevidah airfields and a carrier strike force in order to conduct a combined exercise "Rising Storm."
The Commander's intent is expressed in terms of the purpose and the end state.

<u>Purpose:</u> This operation is to <u>deter</u> Califon from continuing its cyber exploits against of Nevidah as well as to <u>assure</u> Nevidah of US support. Should deterrence fail, US forces will be prepared to rapidly seize the initiative against Califon.
<u>End State:</u> Nevidah's lines of communication are open and operating free from the threat of cyber exploits. Califon ceases to conduct cyber exploits. The Califon president is deterred from escalation and seeks a diplomatic solution to the mineral fields dispute.

The situation between Califon and Nevidah is upsetting the world markets and affecting electronics manufacturing because of the disruptions in production at the contested Mineral Fields. Many countries are anxious to defuse the situation but do not wish to see the US take unilateral action. The UN has already established sanctions on Califon and there is talk of establishing tougher sanction.

The Califon president is demanding that the UN sanctions, put in place at the request of the US, be removed. The Califon president is admired domestically for challenging the US and these factors are influencing the Califon president.

Washorgon is suffering economically from the Califon-Nevidah dispute. The government of Washorgon, under pressure from its population, is considering sending a diplomatic mission to Califon to talk to the president.

The US considers the use of Information Operations (strategic communications) to influence world opinion and also the Califon population. The US and Nevidah will conduct a combined military exercise called "Rising Storm." This involved bringing a US Navy carrier strike group in the territorial waters of Nevidah and also the stationing of US Air Force units on Nevidah military airfields, especially in the one near the Mineral Field. In addition to manned aircraft, the US brings several UAVs, some for ISR but some are weaponized.

Nevidah considers a Special Forces mission to attack and disable the Califon Cyber Operations Center and asks the US to provide cyber support to the operation – first by providing intelligence and then helping to extract the Special Forces from the target area. The US may use UAVs to conduct surveillance but also to support the Nevidah operation. The US is able to apply cyber exploits against the military assets of Califon, particularly the sensor assets of Califon and the communication links that enable Command and Control.

Califon has an Integrated Air Defense System (IADS) that is divided into two sectors: the Eastern Sector with the Eastern Sector Operations Center (SOC) and the Western Sector with Western SOC. There is one Air Defense Operations Center (ADOC); there is one GCI (Ground Control Intercept) Station; and there are Visual Observation Points (VOPs). The Califon Cyber Ops Center is the source of the cyber attacks.

Washorgon is neutral and is concerned that hostilities between its neighbors are disrupting trade. Washorgon decides independently to send a diplomatic flight to Califon in an effort to diffuse the crisis. Nevidah and the US are aware of the Washorgon plan.

Nevidah and the US decide to execute a SOF Mission in which Nevidah will conduct a SOF raid to Califon's Cyber Ops Center. The objective of the SOF mission is to penetrate and corrupt the Califon Cyber Ops Center. There are two aspects to the operation. First, it is to insert in a covert manner the Special Forces team near the Cyber Ops Center and have the team incapacitate the Center by destroying its power sources and its communications lines (land lines and satellite links); then to extract the SOF team once the mission is accomplished. Nevidah asks the US for support in conducting the SOF raid by conducting cyber exploits against the Califon COC such as attacking the internal network of the Cyber Ops Center; execute sniffer attacks on nodes of the system; modify route attacks (misdirect exploits by Center); and transmit out of order packets to corrupt/disable exploits.

In addition, the US plans to conduct Information Operations using social media to stir up Califon population against Califon president's actions. The specific cyber actions by the US are: establish specific Califon cyber information requirements to defend US military networks and attack Califon IADS; deploy ELINT collection to enable EW and NW operations against Califon IADS; and special technical operations cyber attacks (e.g., disable routers, switches, and other key systems.)

5.3 Phase 2: Seize the Initiative

Phase 2 operations begin when Califon invades Nevidah to occupy the Mineral Fields region; the Joint Force Commander's objectives are to defend US and Nevidah forces and interests in the Pacifica theater; deny objectives of Califon offensive operations, compel Califon to resume negotiations with Nevidah, and ensure effective Command, Control, Communications, Computers and Intelligence (C4I) throughout the theater.

The current situation is that Califon has occupied the disputed mineral fields and accelerated its cyber exploits against Nevidah and US forces.

The Commander's intent for Phase 2 is as follows:
Purpose: This objective is to seize the initiative from Califon so it ceases cyber exploits against Nevidah and agrees to enter negotiations regarding use of the disputed mineral fields. Should this fail, US forces will be prepared to conduct major combat operations in support of Nevidah to forcibly remove Califon forces.
End State: Nevidah's lines of communication are open and operating free from the threat of cyber exploits; Califon ceases to conduct cyber exploits; Califon president seeks a diplomatic solution to the mineral fields dispute.

In Phase 1 the US employed limited cyber exploits against the military assets of Califon, primarily against the sensor assets of Califon and the communication links that enable Califon's Command and Control. Califon has now moved forces into the disputed areas and has directed all Nevidah-based mining concerns to depart the area or "face consequences." Califon has begun an aggressive cyber campaign to deny US capability to deploy additional forces or respond with existing capabilities in theater to include: cyber attacks on communications, GPS spoofing, and injection of false data into command centers and tactical platform computers.

In response to Nevidah's successful SOF attack, Califon use of its primary cyber networks is reduced dramatically, but there has been minimal detectable impact on its C2 capabilities suggesting that it has shifted to a war reserve capability and/or implemented a variety of mission assurance protocols which reduce the demand for information exchange and associated bandwidth.

The US directs implementation of its plan to deploy additional fighters and ISR RPVs to Nevidah, another carrier strike group to the region, as well as bombers, ISR and C2/BM aircraft, and aerial tankers to Gamu but finds that Califon is disrupting unclassified deployment systems and impairing even classified networks through attacks on commercial communication routers and switches.

The US postures forces and declares its intent to implement a full air and naval blockade of Califon until it agrees to negotiations. Then it develops defensive options for Counter-Counter-C2 (CC-C2) operations to protect routers and switches and Counter-Counter-Cyber (CC-Cyber) operations to detect and prevent data injection attacks and mitigate Denial-of-Service (DOS) attacks against servers and clients. The US prepares NW and EW attacks to disable Califon IADS and EW Radars and Counter C2 and Counter Cyber attacks against Califon forces. Furthermore, the US develops options to employ Cyber-IO capabilities in the disputed region to undermine effectiveness of the Califon occupation forces and assure Nevidah-based entities that help is on the way. To that effect, the US develops an IO campaign against Califon to counter Califon propaganda among its people and international supporters and induce mistrust in the Califon government; although traditional delivery methods are also employed, the primary tool is the use of Cyber-IO exploits using social media and precision targeting of e-mail, blog sites, and propaganda websites. The US implements high mission assurance TTPs (tactics, techniques, and procedures) that require very little bandwidth to be effective or employs point-point laser communications (very difficult to disrupt). The US employs military deception (Mil-Dec) in its primary C2 systems to disrupt Califon intelligence and planning.

5.4 The Pacifica Scenario Models

On the basis of the scenario described, a Timed Influence Net (TIN) model [6] was generated using the software application Pythia [7]. One constructs a Pythia model by defining on the right the desired effects and then unfolding backwards towards a set of actionable events that influence the desired effects. The Pythia model for the Phase 1 scenario is shown in Figure 10.

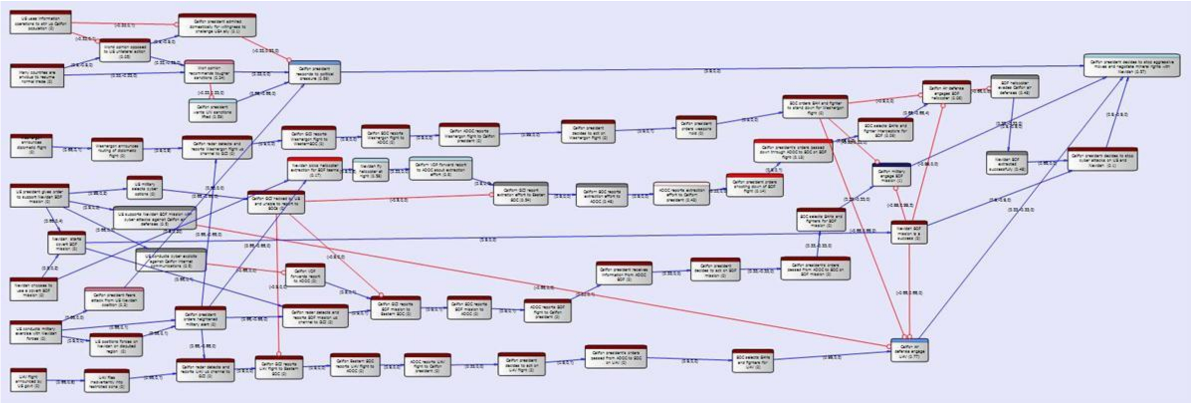


Figure 10: The Timed Influence Net model for the Phase 1 scenario

The desired effects are depicted in the rightmost three nodes in Figure 10. They are: Califon president decides to stop aggressive actions against Nevidah and to negotiate mineral rights; Califon president decides to stop cyber attacks against the US and Nevidah; and Nevidah SOF team is extracted after conducting a successful mission. The complete set of nodes is shown below; they give a sense of how the model has represented the scenario.

- Washorgon announces diplomatic flight
- Califon president admired domestically for willingness to challenge USA ally
- UAV flies inadvertently into restricted zone
- Califon president wants UN sanctions lifted
- Califon president responds to political pressure
- US conducts cyber exploits against Califon internet communications
- Califon radar detects and reports SOF mission up channel to GCI
- Califon radar detects and reports Washorgon flight up channel to GCI
- Califon radar detects and reports UAV up channel to GCI
- Califon VOP forwards report to ADOC
- Califon GCI reports SOF mission to Eastern SOC
- Califon GCI reports Washorgon flight to Western SOC
- Califon GCI reports UAV flight to Eastern SOC
- UAV flight announced by US government

- Nevidah chooses to use a covert SOF mission
- Califon president fears attack from US Nevidah coalition
- US positions forces on Nevidah on disputed region
- Califon GCI hacked by US and unable to report to SOCs
- US conducts military exercise with Nevidah forces
- Califon president orders heightened military alert
- Califon SOC reports SOF mission to ADOC
- Califon SOC reports Washorgon flight to ADOC
- Califon Eastern SOC reports UAV flight to ADOC
- Nevidah starts covert SOF mission
- Nevidah flies helicopter at night
- Nevidah picks helicopter extraction for SOF team
- US military selects cyber options
- Califon president decides to act on Washorgon flight
- Califon president decides to act on UAV flight
- ADOC reports SOF flight to Califon president
- Califon ADOC reports Washorgon flight to Califon president
- ADOC reports UAV flight to Califon president
- SOC orders SAM and fighter to stand down for Washorgon flight
- Califon president orders weapons hold
- Califon president orders shooting down of SOF flight
- World opinion opposed to US unilateral action
- Califon president decides to stop cyber attacks on US and Nevidah
- Nevidah SOF mission is a success
- Califon president's orders passed down through ADOC to SOC on SOF flight
- Washorgon announces diplomatic flight
- Califon president admired domestically for willingness to challenge USA ally
- UAV flies inadvertently into restricted zone
- Califon president wants UN sanctions lifted
- Califon president responds to political pressure
- US conducts cyber exploits against Califon internet communications
- Califon radar detects and reports SOF mission up channel to GCI
- Califon radar detects and reports Washorgon flight up channel to GCI
- Califon radar detects and reports UAV up channel to GCI

- Califon VOP forwards report to ADOC
- Califon GCI reports SOF mission to Eastern SOC
- Califon GCI reports Washorgon flight to Western SOC
- Califon GCI reports UAV flight to Eastern SOC
- UAV flight announced by US government
- Nevidah chooses to use a covert SOF mission
- Califon president fears attack from US Nevidah coalition
- US positions forces on Nevidah on disputed region
- Califon GCI hacked by US and unable to report to SOCs
- US conducts military exercise with Nevidah forces

The TIN model was used to develop and analyze courses of action that were to be used in the computational experiments. Once it was determined that the model did express the scenario correctly and had the desired properties, it was converted into an executable discrete event model expressed as a Colored Petri Net [8] using the application CPN Tools [5]. This is shown in Figure 11. Note the similarity in structure between the TIN and CPN models.

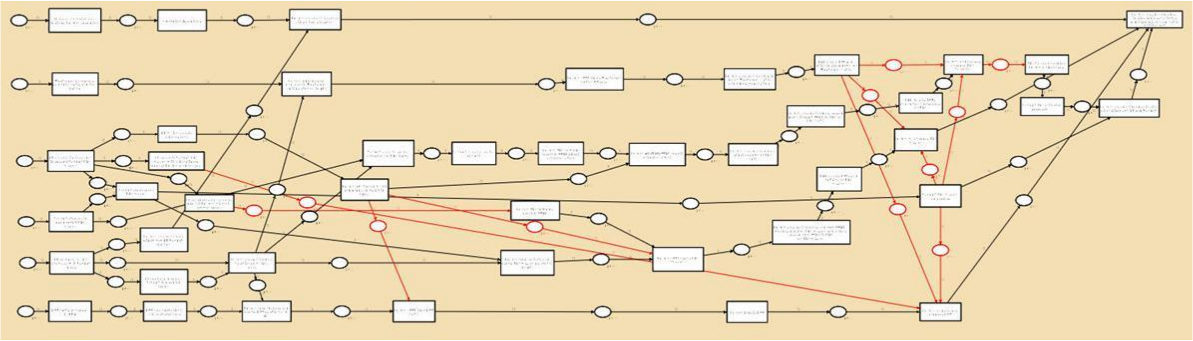


Figure 11: The Colored Petri Net model of the Phase 1 scenario

The red arcs indicate cyber exploits by Nevidah and the US. The CPN model interacts with the Pacifica and US networks modeled in OMNeT++ where the specific exploits (network attacks) are implemented.

In implementing the exploits of Phase 2, a more detailed version of the Colored Petri Net model of Phase 1 was developed. A segment of the more detailed model is shown in Figure 12. Again, red arcs indicate locations where cyber exploits can be applied. In addition, when red and green nodes are shown side by side, they signify that the signal goes to the corresponding OMNeT++ network model for transmission and then comes back to the CPN model.

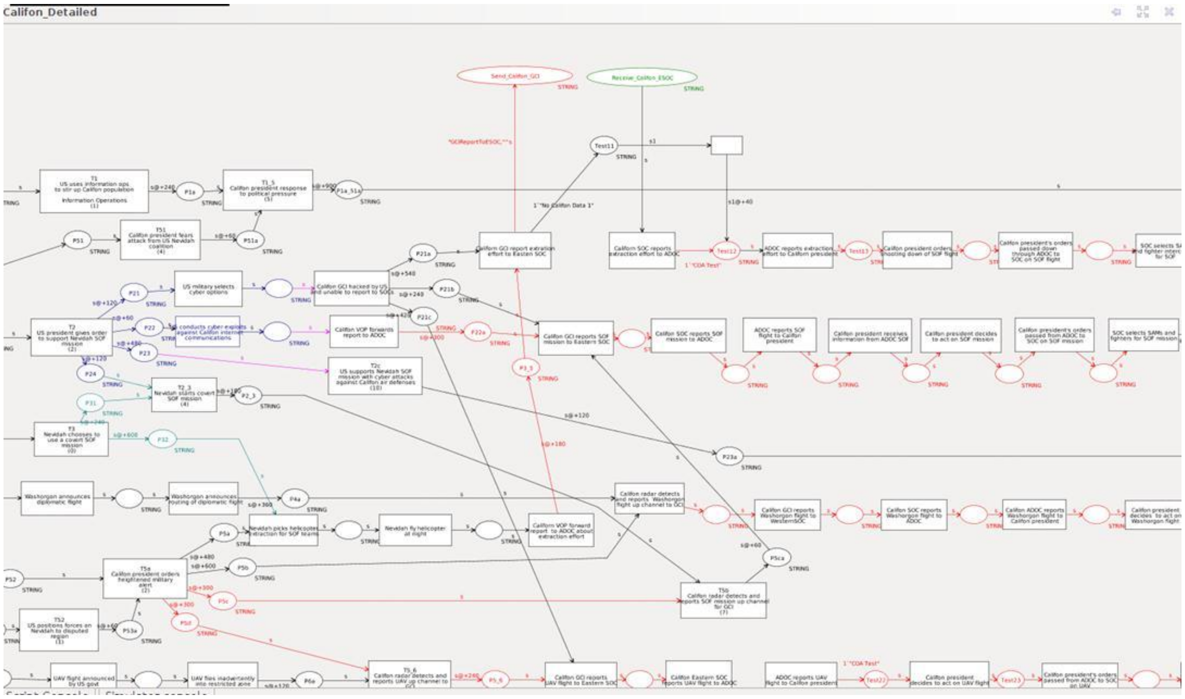


Figure 12: Detailed CPN model

6.0 CONCLUSIONS

The Next Generation Command and Control (C2) Wind-Tunnel project has developed an approach for the simulation-based evaluation of Courses of Actions, extended the C2WT toolsuite previously developed with new capabilities to model operational sequences and message mappings, integrated two simulation engines and developed cyber effects models for a network simulator, and demonstrated the use of the tools suite on specific examples. The toolsuite is functional, documented (see Appendices), and is available as Linux Virtual Machine. The framework is predicated on the assumptions that (1) simulation models for relevant operational scenarios are available, and (2) experts are able to derive operational sequence models from COA-s that in turn can be executed in the context of the simulation models. The key observation here is that the sequence models interact with and perturb the simulation models (as they are executing) resulting in different and detectable outcomes. The need for realistic models is a very strong requirement – but it is natural: without simulation models that reflect our understanding and assumptions of the problem and which are predictive in nature one cannot do any sort of computational evaluation. The second issue stems from the very high-level nature of the COA-s: while the simulation models are very concrete, the COAs cannot be directly interpreted in their context – they have to be operationalized by an expert. This requires human skills and knowledge but is a feasible task.

The project's results clearly show the feasibility and the limitation of the approach. Together with the other project results (from BAE Systems) the toolsuite can serve as the foundation for an 'industry-grade' tool for the analyst and planner.

7.0 REFERENCES

[1] Pacifica Crisis Scenario January 2002, http://www.globalsecurity.org/military/library/report/1998/ex2/index.html

[2] A. H. Levis, "Time Sensitive Course of Action Development and Evaluation," Proc. NATO HFM-202 Symposium on Human Modeling for Military Applications, Amsterdam, The Netherlands, 18-20 October 2010.

[3] Peter Pachowicz, Lee W. Wagenhals, John Pham, and A. H. Levis, "Building and Analyzing Timed Influence Net Models with Internet-enabled Pythia," Proc. of SPIE, 2007 Defense and Security Symposium, Orlando, FL, April 2007.

[4] Kurt Jensen and Lars Kristensen, *Coloured Petri Nets*, Springer, Heidelberg, 2009

[5] CPN tools: http://cpntools.org

[6] Wagenhals, L. W. and Levis, A. H. 2002. Modeling Support of Effect-Based Operations in War Games, *In Proceedings of the Command and Control Research and Technology Symposium.*

[7] L. W. Wagenhals, and Levis, A. H., Course of Action Analysis in a Cultural Landscape Using Influence Nets, *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2007).*

[8] K. Jensen: Coloured Petri Nets and the Invariant Method. *Theoretical Computer Science 14 (1981), North-Holland, 317-336.*

APPENDIX A – INSTALLATION AND CONFIGURATION USING A CASIM VIRTUAL MACHINE

This Appendix describes how to setup the Virtual Machine (VM) and how to run Pacifica2 experiments. It also describes how to view MySQL database and collect and view experiment analysis results.

A.1 SETTING UP THE VIRTUAL MACHINE (VM)

For the VM, we prefer to use the player Oracle VM VirtualBox. Download the latest VirtualBox and install it on your machine (Windows, Linux, or Mac).

In VirtualBox, create a new VM by pressing the New button. It will start a “Create New Virtual Machine” wizard.

Click “Next” on the wizard. On the next screen type a suitable name of the VM, say “CASIMAllInUbuntu32Bit”. Next on the same screen make sure that “Linux” is selected as the “Operating System” and “Ubuntu” as the “Version” and click Next.

One the next screen assign 2048 MB (2GB) RAM to the VM. Although 1GB should be enough, you get better performance with 2GB RAM assigned. Click Next.

One next screen, “Virtual Hard Disk”, make sure that “Start-up Disk” checkbox is checked and the radio button “Use existing hard disk” is selected. Next click on the folder icon next to the radio button “Use existing hard disk” to browse to the VM file (with .VDI extension) to select it. Make sure to select the CASIMAllInUbuntu32Bit.vdi file. Click Finish to complete the VM creation wizard. Now, you should see a “CASIMAllInUbuntu32Bit” appear in the Oracle VM VirtualBox. Before we can start the VM, we must adjust some key VM settings as described in the next section.

A.2 UPDATING VIRTUAL MACHINE SETTINGS BEFORE RUNNING FOR THE FIRST TIME

Most of the settings should already be rightly set when you create the VM. However, there are certain settings that must be updated, before we can start the VM for the first time. Listed below are only the settings that must be updated:

System | Motherboard tab:

In the Extended Features section, make sure that “Enable IO APIC”, “Hardware clock in UTC time”, and “Enable absolute pointing device” is checked and “Enable EFI (special OSes only) is NOT checked.

System | Processor tab:

It is better to assign 4 processors to the VM. Also, make sure that in Extended Features "Enable PAE/NX" is NOT checked.

System | Acceleration tab:

Make sure that both the checkboxes "Enable VT-x/AMD-V" and "Enable Nested Paging" are checked.

Display | Video tab:

Make sure that you assign 64MB of RAM for the VM video memory. Make sure that the checkboxes "Enable 3D Acceleration" and "Enable 2D Acceleration" are NOT checked.

Network | Adapter 1 tab:

Check the checkbox "Enable Network Adapter". Next in the "Attached to" ComboBox, select "Host-only Adapter". The default settings that come up with Host-only Adapter are fine.

Network | Adapter 2 tab:

Check the checkbox "Enable Network Adapter". Next in the "Attached to" ComboBox, select "NAT". The default settings that come up with NAT are fine as well. Also, we only need two Network adapter, so we do not need to worry about Network Adapters 3 and 4.

USB | USB tab:

Make sure that the checkbox "Enable USB Controller" is checked.

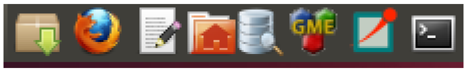
This should complete the settings of the Virtual Machine. Now the VM can be started by selecting it in the Oracle VM VirtualBox and pressing the "Start" button.

IMPORTANT: Once the VM is started make sure that the VirtualBox Host-only adapter has statically assigned IP address 192.168.56.101 as that is used in demos packaged in the Virtual Machine.

A.3 VIRTUAL MACHINE OVERVIEW

When Virtual Machine is started, it waits for "c2wt" user to enter a password. The password for "c2wt" user is "C2WTunnel" and is case-sensitive. Before inputting the password though, make sure by pressing the "Cog Wheel" above the password field that "GNOME" desktop layout is selected. Enter the password and press Enter to login as user "c2wt".

There are short-cuts for the key applications used in the C2WT on the desktop. Once booted, you should see an icon bar next to Applications and Places like:



The first icon is for “Synaptic Package Manager” and is used to install, remove, or update Ubuntu application or library software. This is a graphical and user-friendly application. If you do not plan to install new software or libraries in the Ubuntu VM, you should NOT need to use it.

The next icon is the Mozilla Firefox Internet Browser. Since we added a NAT Adapter in the VMs Network settings, if the host machine (on which the Oracle VM VirtualBox is installed) has an internet connection, we should have an internet connection in the VM and Browser should work just fine.

The next icon is for the Ubuntu Text file editor (“gedit” application). It can be used to view and edit textual files.

The next icon is the File Browser. This is like “Windows Explorer” and can be used to navigate inside the folder hierarchy and create, rename, and delete files and folders.

The next icon is for the “DBVisualizer”. This can be used to view the MySQL databases created by C2WT as a result of running an experiment. The database names include the name of the demo (federation name) at the beginning, followed by the date and time at which it was created, followed by the details of COAs used in that experiment. The name in MySQL is limited to 64 characters, so if the COA names used are longer, they will be truncated when the 64 characters limit is reached.

For the first time, you might have to connect the DBVisualizer with the MySQL. To do this, right click on C2WTDBConnection, and select “Reconnect”. It should already be set with the connection parameters (for example, the username is “root” and password is “c2wt” and is case-sensitive).

DBVisualizer has a database tree browser on the left panel where all the databases are listed. You can open the database nodes and then the Tables node to see the Tables created in that particular database. Each database should have three Tables in it. The first Table is called “EventLog”, which records the publish of an interaction by a federate and the time at which it was published. EventLog table is very good for debugging purposes and was recently added in C2WT. The second table is called “ExperimentInfo”, which records the Publish and Subscribe map in the experiment. This describes which federate publishes and subscribes which interactions. The third table is called “SimulationData” is the biggest and records the actual simulation data. Based on what was configured in the GME mode to be logged, SimulationData can record every single interaction that was exchanged among federates or any object update that was made by any federate.

The next icon is for the GME application. GME is a Windows application, but we run GME inside the Ubuntu VM using WINE (WINdows Emulator). As such we no longer require users to have a Windows machine to be able to run C2WT experiments.

The next icon is for the OMNeT++ IDE (Integrated Development Environment). This is used to edit C2WT source code and to run and debug C2WT internal applications (if needed). The GME supplied with the VM already is setup to run the two key interpreters that we use, viz. C2WMainInterpreter (to generate source code and configuration files), and C2WDeploymentInterpreter (to generate COA scripts and deployment scripts to run the experiments). As such, the C2WMainInterpreter is needed to be run on the GME model only if user has added new (or updated existing) interactions, objects, or federates, or if the user has changed any publish and subscribe relationships. If all user does, is to define new COAs, updates existing COAs, change deployment settings or define new experiment models, ONLY C2WDeploymentInterpreter needs to be run on the updated GME model. This is important because if the C2WMainInterpreter is run, the new source code will be generated that might need to be recompiled to execute the experiment. So, unless, user changes any of the FOMs or federates, usually only C2WDeploymentInterpreter will need to be run.

The next icon is to start a Terminal Command Window. This is used to start/stop the experiments. The command details are listed in the section below.

A.4 MANUALLY RUNNING EXPERIMENTS USING A BASH SHELL

These commands are run from the Terminal Command Window. IMPORTANT: ALL commands are run from the ngc2wt project root directory. For that change to project root directory by running:

```
cd /home/c2wt/Projects/ngc2wt/
```

To run Pacifica2 NoCyber experiment, run:

```
bash -x generated/Pacifica2/scripts/NoCyberDep1h/Main/start.sh
```

If you do not want see all the command-line output, you can run the above command without the "-x" option. The bash script is needed because the symlink "/bin/sh" points to "dash" shell. Apart from NoCyberDep1h deployment, there are other deployment called NoCyberDep that can be used if you are using two Virtual Machines to increase the simulation performance.

If you want to run the above experiment without any GUIs, please use the command:

```
bash -x generated/Pacifica2/scripts/NoCyberDep1h/Main/start.sh --batch
```

Pacifica2 experiment comes with a CyberGaming experiment that involves running 5 BLUE COAs against 6 RED COAs – a total of 30 experiments. C2WT now has the capability to run all of these 30 experiments in batch mode. These batch mode 30 experiments can be run by running the following command:

```
bash -x generated/Pacifica2/scripts/CyberGamingDep1h/Main/runAllExperimentsInBatchMode.sh
```

Above command will run all the 30 experiments in batch mode, one-by-one. If you want to run a particular experiment out of these 30 experiments, you can do so. For example:

```
bash -x generated/Pacifica2/scripts/CyberGamingDep1h/Main/start_RC16_BLC15.sh
```

When batch mode of experiments are run, the scripts create a file called "Pacifica2_CyberGamingDep1h_database_names.txt" in directory "/home/c2wt/Projects/ngc2wt/generated/Pacifica2/scripts/CyberGamingDep1h/Main" directory. This file contains the names of the databases that were generated for each of these 30 experiments. This file is used by the data-collection script to read the MySQL databases and generate an Excel spreadsheet of the results.

To run the data-collection script, run the following command:

```
bash -x src/Pacifica2/scripts/CollectStats.sh
```

The above command should produce a file "Pacifica2_Stats_<DATE>_<TIME>.xls" file corresponding to the experiment results. If you want the NoCyber experiment results to be also part of the above Excel spreadsheet, simply edit the database names file "Pacifica2_CyberGamingDep1h_database_names.txt" in directory "/home/c2wt/Projects/ngc2wt/generated/Pacifica2/scripts/CyberGamingDep1h/Main" directory by adding a new line of text corresponding to the name of the database generated for the NoCyber experiment. For example, you may want to add a line with text "Pacifica2_NoCyberDep1h_2012_04_26_110422" at the end of the file. Now, if you run the above command, it will also include NoCyber experiment in the data-collection.

A.5 RUNNING EXPERIMENTS USING EXPERIMENT CONTROLLER

The CASIM Virtual Machine comes with a tool called Experiment Controller. This tool automatically searches for all available scenarios, experiments, and runs within those experiments (depending on all COA combinations possible in them), and makes them available in a tree view in a Graphical User Interface (GUI). User can use this tool to easily select and run experiments in Interactive or Batch mode. When run in Interactive mode all GUIs of the simulators used are shown including CASIM's Federation Manager and the COA Simulation Display (if the experiment uses any COAs). When run in Batch mode, no GUIs are shown, not even any error or information dialogs. During Batch mode of execution, multiple experiment runs can be selected to be executed including those from different scenarios (demos). During execution, the experiment controller shows status of experiments being executed and log messages captured on console for execution information and errors, if any. For more about experiment controller, user is referred to Appendix I dediicated to the full description of the experiment controller.

A.6 SOURCES

All sources for the Pacifica2 are located in /home/c2wt/Projects/ngc2wt/src/Pacifica2 directory. All generated files for the Pacifica2 scenario are located in /home/c2wt/Projects/ngc2wt/generated/Pacifica2 directory.

APPENDIX B – INSTALLATION AND CONFIGURATION FROM SOURCES

B.1 PRELIMINARIES

This section lists what is needed before the C2 Wind Tunnel and related software is installed. Some steps described in this Appendix require the user performing them has root/administrator privileges.

B.1.1 Computer Environments

To run the C2 Wind Tunnel, two computer environments are needed: Linux and Windows. A computer environment allows programs for a particular operating system to be executed.

Obviously, the Windows operating system is also a Windows environment, as it allows Windows programs to be executed. However, WINE, which is run in Linux, is also a Windows environment for the same reason.

IMPORTANT: In CASIM, it is recommended that users use only the Ubuntu Linux computer environments. The main reason for this is that OMNeT++ modules developed in CASIM are supported only under Ubuntu Linux. Moreover, the third-party library used for execution of CPN models (viz. BRITNeY Java library for CPN Tools), works correctly only in Ubuntu Linux environment.

For the C2 Wind Tunnel, the Linux environment is in fact a Linux operating system, specifically Ubuntu 10.10 (32 bit) or later. The Windows environment, however, can either be a Windows operation system (Window XP or Windows 7), or WINE run on Ubuntu. To reiterate, the following computer environments are needed before installing the C2 Wind Tunnel:

- Linux environment: Ubuntu Linux 10.10 (32-bit) or later
- Windows environment. This can be either:
    - A Windows operating system (Windows XP or Windows 7)
    - WINE running in the Linux environment

IMPORTANT: In Ubuntu VMs, make sure the username you use is “c2wt”. Also, on main development VM, get all sources in $HOME/Projects directory such that the directory /home/c2wt/Projects/ngc2wt contains folders ‘src’, ‘3rdParty’, etc.

Incidentally, to use WINE in the Ubuntu Linux environment, it must be installed. To do this, enter the following command in a terminal (see section B.1.2.2 for an explanation of the 'sudo' command):

```
sudo apt-get install wine
```

B.1.2 Executing commands as ‘root’

In the Ubuntu Linux environment, some commands must be executed as 'root', i.e. under the super-user account. There are two main ways to do this, described below.

B.1.2.1 The ‘su’ command. The first way for a user to execute commands as 'root' is to use the 'su' command. That is, in a terminal they enter (words in italics are printed by the computer – don't type them in):

```
su -c “<command>”
Password:
```

This command executes <command> as 'root'. However, before it does so it prompts for the password of the 'root' account, and will not execute the command if this password is not entered. So, if a user is to be able to execute commands as root using the 'su' command, they must know the root password. This is, of course, very insecure, as this user will be able to execute any command as 'root' by using the 'su' command.

B.1.2.2 The ‘sudo’ command. A better means of allowing a user to execute commands as 'root' is by allowing them to use the 'sudo' command:

```
sudo <command>
Password:
```

This will also execute <command> as root, but the password to be entered is the password of the user executing the 'sudo' command, i.e. not 'root'. Consequently, they don't need the root password, and the root password can be kept secret. As well, the sudo command can be configured to allow a given user only to execute certain commands as root. If “sudo” is not currently installed in your Linux environment, in a terminal execute the following command as 'root':

```
apt-get install sudo
```

As described above, if you are currently using an account other than 'root', you can enter:

```
su -c “apt-get install sudo”
```

Once “sudo” is installed can configured, it can be used to execute commands as 'root'.

B.1.3 Setting Environment Variables

A very important part of configuring the C2 Wind Tunnel is the setting of *environment variables*. The C2 Wind Tunnel software uses the values of these environment variables to find needed data and generate code in the appropriate directories.

As a matter of convenience, it is a good idea to configure the computer environments so that the environment variables have their correct values every time the environments are used, rather than having to reset the environment variables every time the environments are used. This section covers how to do this.

B.1.3.1 Setting persistent environment variables in the Linux computer environment. To set (or *assign*) an environment variable in the Linux computer environment, one only need bring up a terminal and execute a command like the following in a terminal:

```
export <VARIABLE_NAME>=”<VARIABLE_VALUE>”
```

Here VARIABLE_NAME is the name of the environment variable, and VARIABLE_VALUE is the value that will be assigned to the variable. Note two important things:

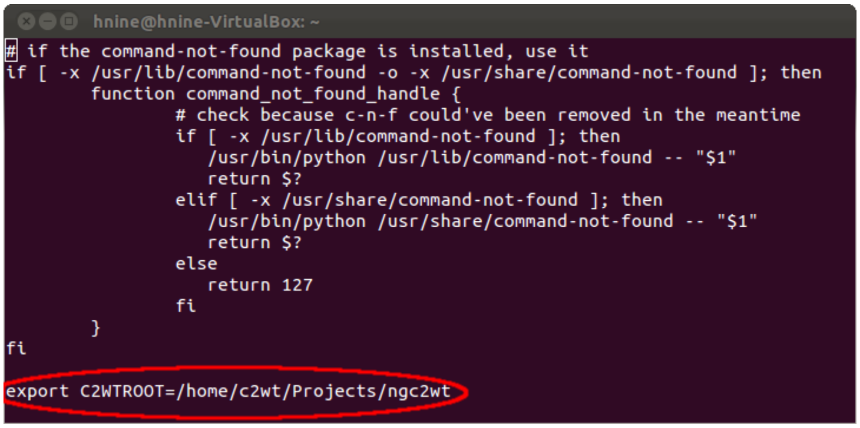
- **“export”**: if this isn't used, the variable will be ineffective for the C2 Wind Tunnel.
- **There are no spaces to either side of the equals-sign '='**: Use of spaces on either side of the equals-sign will result in an error.

In the following command, the C2WTROOT environment variable is assigned the value “/home/c2wt/Projects/ngc2wt”:

```
export C2WTROOT=”/home/c2wt/Projects/ngc2wt”
```

Having set this environment variable, *it will retain this value for the life of the terminal*. That is, once the terminal is closed, the environment variable will no longer exist.

To make an environment variable *persistent*, i.e. so that it will have a given value in every terminal (and every other application), the command to set it must be placed in either the <HOME>/.bashrc file (where <HOME> is your home directory) or the /etc/bash.bashrc file (the latter can only be modified by the *root* user). That is, at the end of one of these files, place the command that sets the environment variable to the desired value, as shown below (everything above the command was already in the file):



```
# if the command-not-found package is installed, use it
if [ -x /usr/lib/command-not-found -o -x /usr/share/command-not-found ]; then
        function command_not_found_handle {
                # check because c-n-f could've been removed in the meantime
                if [ -x /usr/lib/command-not-found ]; then
                   /usr/bin/python /usr/lib/command-not-found -- "$1"
                   return $?
                elif [ -x /usr/share/command-not-found ]; then
                   /usr/bin/python /usr/share/command-not-found -- "$1"
                   return $?
                else
                   return 127
                fi
        }
fi

export C2WTROOT=/home/c2wt/Projects/ngc2wt
```

Figure B - 1: Exporting C2WTROOT environment variable

Once this is done, exit all terminals. Then execute another terminal and enter the following command for any of the variables you set in the <HOME>/.bashrd or /etc/bash.bashrc files:

```
echo $<VARIABLE_NAME>
```

The output of this command should be the value of the <VARIABLE_NAME> environment variable. For example, for the C2WTROOT environment variable, the following should be displayed on your terminal:

```
echo $C2WTROOT
/home/c2wt/Projects/ngc2wt
```

B.1.3.2 Setting persistent environment variables in the Windows computer environment.

Setting an environment variable in the Windows computer environment, in particular so that it is persistent, depends on whether the environment is a Windows operating system or WINE.

B.1.3.2.1 Setting a persistent environment variable in a Windows operating system.

The following instructions pertain to the Windows XP operating system. The instructions for Windows 7 are similar.

Click the “Start” button, right-click on “My Computer” in the resulting pop-up window, and select “Properties” in resulting menu:

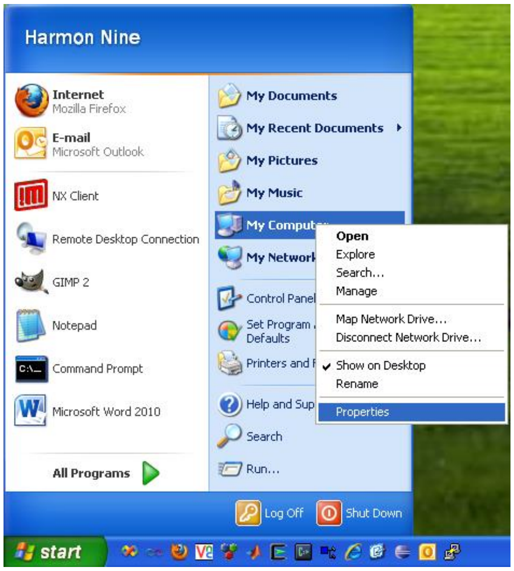
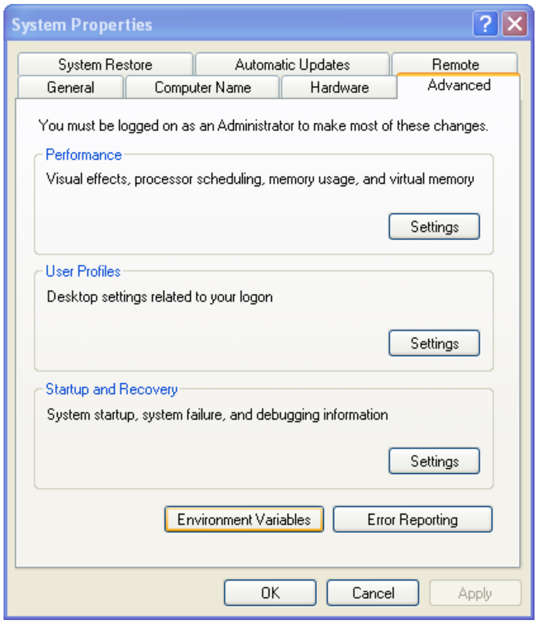


Figure B - 2: Loading System Properties

In the resulting “System Properties” window, click on the “Advanced” tab, then click the “Environment Variables” button:



You must be logged on as an Administrator to make most of these changes.

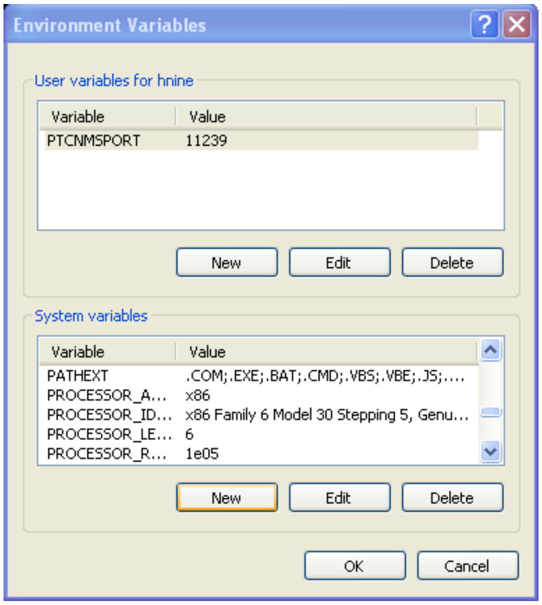
Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your logon

System startup, system failure, and debugging information

Figure B - 3: Loading Environment Variables

The resulting "Environment Variables" window, shown below, allows you to create new or modify existing environment variables. **It is best to do this for System Variables, as it will make the environment variables you create/modify visible for all users.** In other words, perform the following actions in the "System variables" section of the window. To create a new environment variable, click the "New" button. To modify an existing variable, select it and click the "Edit" button.

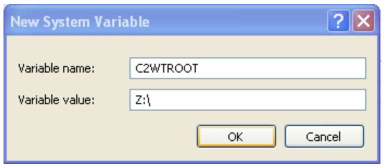


Variable	Value
PTCNMSPORT	11239

Variable	Value
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....
PROCESSOR_A...	x86
PROCESSOR_ID...	x86 Family 6 Model 30 Stepping 5, Genu...
PROCESSOR_LE...	6
PROCESSOR_R...	1e05

Figure B - 4: Editing Environment Variables

Enter the variable name and value in the resulting window. Then click OK. Once you are done creating/modifying environment variables, click OK in the “Environment Variables” and “System Properties” windows.



Variable name: C2WTROOT

Variable value: Z:\

Figure B - 5: Entering value of the Environment Variable

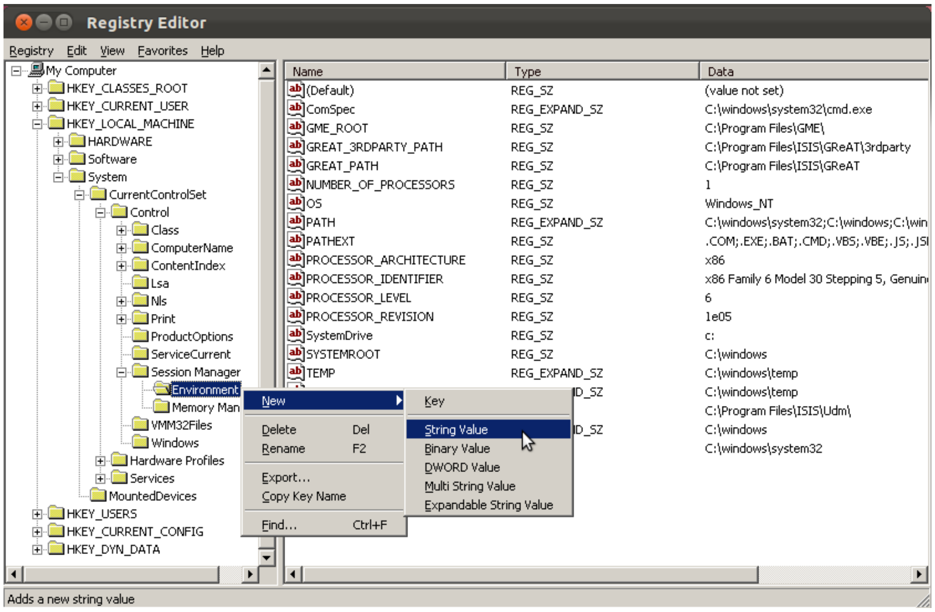
B.1.3.2.2 Setting a persistent environment variable in WINE. The only way to set a persistent environment variable in the WINE Windows environment to by using WINE's registry editor (this same method can used to modify the System variables in a Windows operating system). To execute the WINE's registry editor, enter the following command in a terminal the single-quotes are very important):

```
wine 'C:\windows\regedit.exe'
```

In the resulting window, in the left-hand sub-window, navigate to:

“HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Environment”

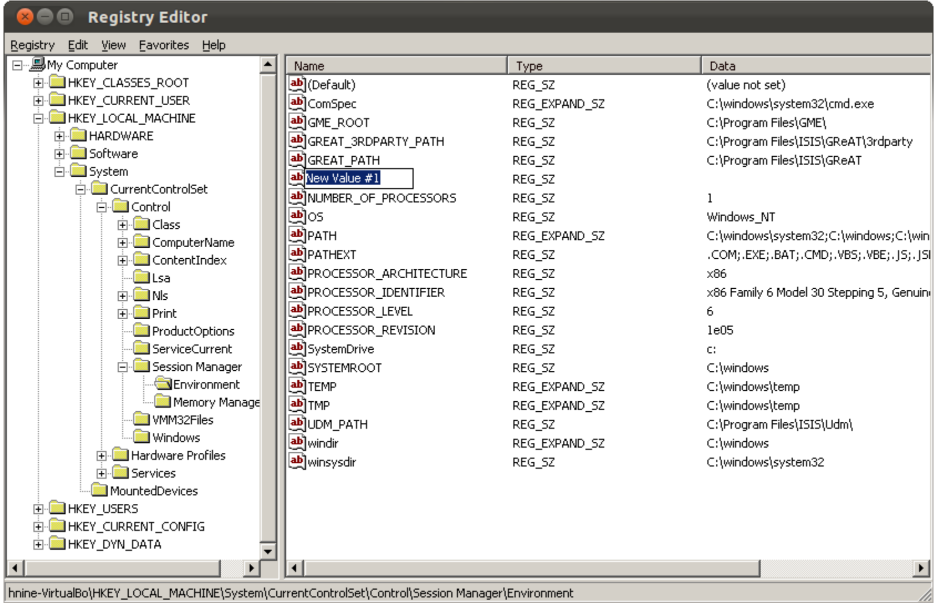
In the right-hand sub-window will appear a list of the names, types, and values of all of WINE's current System environment variables. To create a new variable right-click on the “Environment” folder (highlighted in blue below) and select “New->String Value”:



Name	Type	Data
(Default)	REG_SZ	(value not set)
ComSpec	REG_EXPAND_SZ	C:\windows\system32\cmd.exe
GME_ROOT	REG_SZ	C:\Program Files\GME\
GREAT_3RDPARTY_PATH	REG_SZ	C:\Program Files\ISIS\GReAT\3rdparty
GREAT_PATH	REG_SZ	C:\Program Files\ISIS\GReAT
NUMBER_OF_PROCESSORS	REG_SZ	1
OS	REG_SZ	Windows_NT
PATH	REG_EXPAND_SZ	C:\windows\system32;C:\windows;C:\win
PATHEXT	REG_SZ	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSI
PROCESSOR_ARCHITECTURE	REG_SZ	x86
PROCESSOR_IDENTIFIER	REG_SZ	x86 Family 6 Model 30 Stepping 5, Genuin
PROCESSOR_LEVEL	REG_SZ	6
PROCESSOR_REVISION	REG_SZ	1e05
SystemDrive	REG_SZ	c:
SYSTEMROOT	REG_SZ	C:\windows
TEMP	REG_EXPAND_SZ	C:\windows\temp
	D_SZ	C:\windows\temp
		C:\Program Files\ISIS\Udm\
	D_SZ	C:\windows
		C:\windows\system32

Figure B - 6: Registry Editor

A new variable will appear with a generic name, e.g. "New Value #1". The new variable should be in a state such that its name is highlighted in blue, and is in an editable text field. Its name can be changed by typing its new name in this text field and pressing enter. If it is not in this state, you can right-click on its name and select "rename" from the resulting menu. Once you have entered the name, the variable may change its position in the list. This is because the variables are kept in alphabetical order by name.

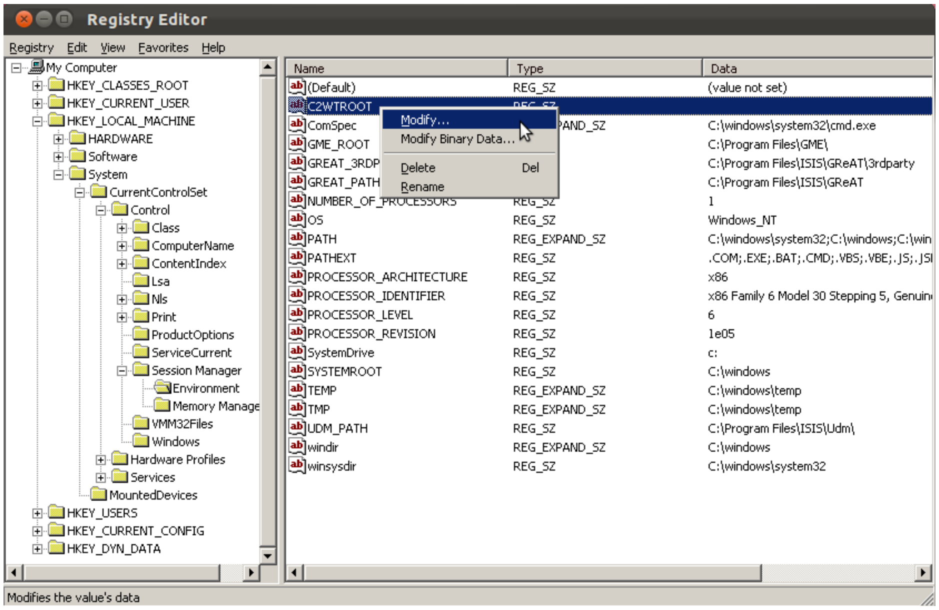


Name	Type	Data
(Default)	REG_SZ	(value not set)
ComSpec	REG_EXPAND_SZ	C:\windows\system32\cmd.exe
GME_ROOT	REG_SZ	C:\Program Files\GME\
GREAT_3RDPARTY_PATH	REG_SZ	C:\Program Files\ISIS\GReAT\3rdparty
GREAT_PATH	REG_SZ	C:\Program Files\ISIS\GReAT
New Value #1	REG_SZ	
NUMBER_OF_PROCESSORS	REG_SZ	1
OS	REG_SZ	Windows_NT
PATH	REG_EXPAND_SZ	C:\windows\system32;C:\windows;C:\win
PATHEXT	REG_SZ	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSI
PROCESSOR_ARCHITECTURE	REG_SZ	x86
PROCESSOR_IDENTIFIER	REG_SZ	x86 Family 6 Model 30 Stepping 5, Genuin
PROCESSOR_LEVEL	REG_SZ	6
PROCESSOR_REVISION	REG_SZ	1e05
SystemDrive	REG_SZ	c:
SYSTEMROOT	REG_SZ	C:\windows
TEMP	REG_EXPAND_SZ	C:\windows\temp
TMP	REG_EXPAND_SZ	C:\windows\temp
UDM_PATH	REG_SZ	C:\Program Files\ISIS\Udm\
windir	REG_EXPAND_SZ	C:\windows
winsysdir	REG_SZ	C:\windows\system32

hnine-VirtualBo\HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Environment

Figure B - 7: Creating a new String variable in Registry Editor

To give the variable a value, right-click on it and select "modify" from the resulting menu.

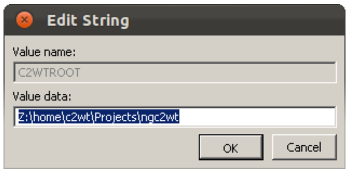


Name	Type	Data
(Default)	REG_SZ	(value not set)
C2WTROOT	REG_SZ	
ComSpec	PAND_SZ	C:\windows\system32\cmd.exe
GME_ROOT		C:\Program Files\GME\
GREAT_3RDP		C:\Program Files\ISIS\GReAT\3rdparty
GREAT_PATH		C:\Program Files\ISIS\GReAT
NUMBER_OF_PROCESSORS	REG_SZ	1
OS	REG_SZ	Windows_NT
PATH	REG_EXPAND_SZ	C:\windows\system32;C:\windows;C:\win
PATHEXT	REG_SZ	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSI
PROCESSOR_ARCHITECTURE	REG_SZ	x86
PROCESSOR_IDENTIFIER	REG_SZ	x86 Family 6 Model 30 Stepping 5, Genuin
PROCESSOR_LEVEL	REG_SZ	6
PROCESSOR_REVISION	REG_SZ	1e05
SystemDrive	REG_SZ	c:
SYSTEMROOT	REG_SZ	C:\windows
TEMP	REG_EXPAND_SZ	C:\windows\temp
TMP	REG_EXPAND_SZ	C:\windows\temp
UDM_PATH	REG_SZ	C:\Program Files\ISIS\Udm\
windir	REG_EXPAND_SZ	C:\windows
winsysdir	REG_SZ	C:\windows\system32

Modifies the value's data

Figure B - 8: Editing variable value in Registry Editor

A new window will appear in which the value for the variable can be entered. Press the "OK" button once this is done.

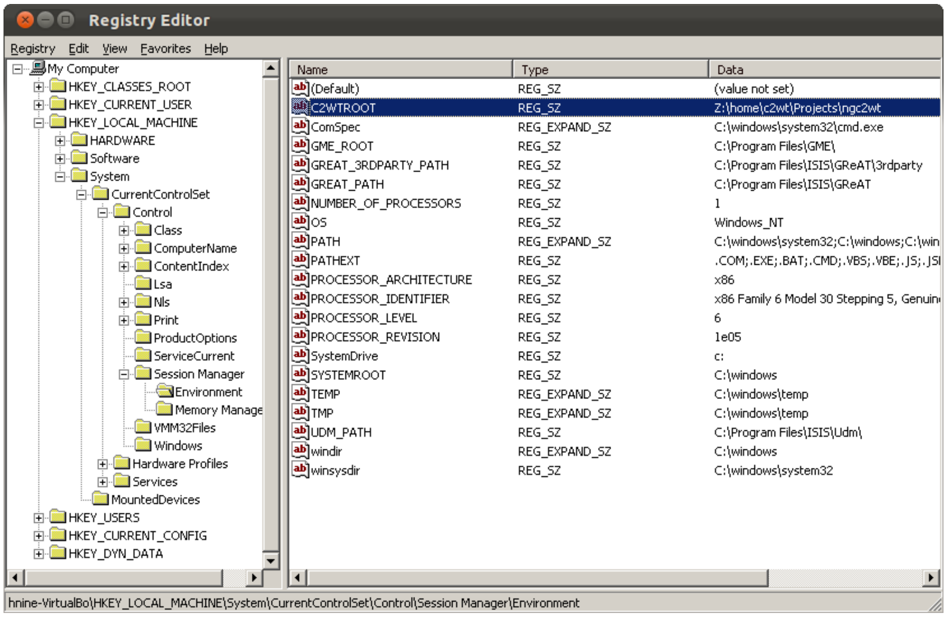


Value name: C2WTROOT

Value data: Z:\home\c2wt\Projects\ngc2wt

Figure B - 9: Entering new value

A persistent environment variable has been created in the WINE Windows environment.



Name	Type	Data
(Default)	REG_SZ	(value not set)
C2WTROOT	REG_SZ	Z:\home\c2wt\Projects\ngc2wt
ComSpec	REG_EXPAND_SZ	C:\windows\system32\cmd.exe
GME_ROOT	REG_SZ	C:\Program Files\GME\
GREAT_3RDPARTY_PATH	REG_SZ	C:\Program Files\ISIS\GReAT\3rdparty
GREAT_PATH	REG_SZ	C:\Program Files\ISIS\GReAT
NUMBER_OF_PROCESSORS	REG_SZ	1
OS	REG_SZ	Windows_NT
PATH	REG_EXPAND_SZ	C:\windows\system32;C:\windows;C:\win
PATHEXT	REG_SZ	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSI
PROCESSOR_ARCHITECTURE	REG_SZ	x86
PROCESSOR_IDENTIFIER	REG_SZ	x86 Family 6 Model 30 Stepping 5, Genuin
PROCESSOR_LEVEL	REG_SZ	6
PROCESSOR_REVISION	REG_SZ	1e05
SystemDrive	REG_SZ	c:
SYSTEMROOT	REG_SZ	C:\windows
TEMP	REG_EXPAND_SZ	C:\windows\temp
TMP	REG_EXPAND_SZ	C:\windows\temp
UDM_PATH	REG_SZ	C:\Program Files\ISIS\Udm\
windir	REG_EXPAND_SZ	C:\windows
winsysdir	REG_SZ	C:\windows\system32

hnine-VirtualBo\HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Environment

Figure B - 10: New registry variable created

B.2 SHARING C2WTROOT IN THE UBUNTU LINUX ENVIRONMENT WITH THE WINDOWS ENVIRONMENT

Whereas the C2 Wind Tunnel uses the Ubuntu Linux environment to develop and compile code (Java and C++), it uses GME for modeling and code-generation, and GME only runs in a Window environment. Because of this, Linux needs to be setup up to share a directory (and all file/directories under it) with the Windows environment.

The Linux directory to share should be called "ngc2wt", e.g. "<HOME>/Projects/ngc2wt", where <HOME> is your home directory (e.g. /home/c2wt). If this directory does not exist, please create it now. One means of doing this is to enter the following command in a terminal:

```
mkdir -p <directory>
```

From this point onward, we will refer to this <directory> as $C2WTROOT. In fact, please create a persistent environment variable C2WTROOT in the Linux environment (see section B.1.3.1) that has this directory as its value, i.e.:

```
export C2WTROOT=<directory>
```

The means of sharing $C2WTROOT with the Windows environment depends on which environment you are using, a Windows OS or WINE.

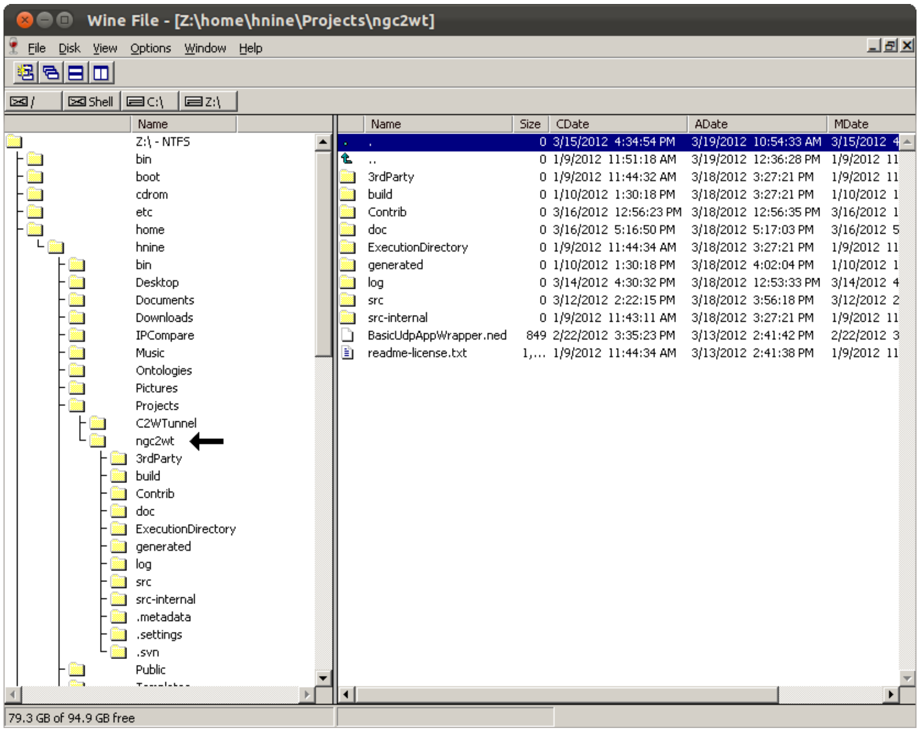
B.2.1 Sharing C2WTROOT with WINE

If you are using WINE as your Windows environment, please define a C2WTROOT environment variable as described as the end of this section.

1. Nothing special need be done to access the $C2WTROOT directory from the WINE windows environment. To demonstrate this, open a terminal and execute the “winefile” command:

   ```
   winefile
   ```

   A window like the one below should appear. The root directory of the Ubuntu environment should be mapped to drive “Z:” in the WINE windows environment. So, for example, if $C2WTROOT in Ubuntu is “/home/c2wt/Projects/ngc2wt”, in WINE it is “Z:\home\c2wt\Projects\ngc2wt”. This directory is indicated by an arrow the below window. Any file in Ubuntu can be accessed this way from WINE.



Name	Size	CDate	ADate	MDate
.	0	3/15/2012 4:34:54 PM	3/19/2012 10:54:33 AM	3/15/2012 4
..	0	1/9/2012 11:51:18 AM	3/19/2012 12:36:28 PM	1/9/2012 11
3rdParty	0	1/9/2012 11:44:32 AM	3/18/2012 3:27:21 PM	1/9/2012 11
build	0	1/10/2012 1:30:18 PM	3/18/2012 3:27:21 PM	1/10/2012 1
Contrib	0	3/16/2012 12:56:23 PM	3/18/2012 12:56:35 PM	3/16/2012 1
doc	0	3/16/2012 5:16:50 PM	3/18/2012 5:17:03 PM	3/16/2012 5
ExecutionDirectory	0	1/9/2012 11:44:34 AM	3/18/2012 3:27:21 PM	1/9/2012 11
generated	0	1/10/2012 1:30:18 PM	3/18/2012 4:02:04 PM	1/10/2012 1
log	0	3/14/2012 4:30:32 PM	3/18/2012 12:53:33 PM	3/14/2012 4
src	0	3/12/2012 2:22:15 PM	3/18/2012 3:56:18 PM	3/12/2012 2
src-internal	0	1/9/2012 11:43:11 AM	3/18/2012 3:27:21 PM	1/9/2012 11
BasicUdpAppWrapper.ned	849	2/22/2012 3:35:23 PM	3/13/2012 2:41:42 PM	2/22/2012 3
readme-license.txt	1,...	1/9/2012 11:44:34 AM	3/13/2012 2:41:38 PM	1/9/2012 11

79.3 GB of 94.9 GB free

Figure B - 11: Windows Explorer like directory browser using WINE

2. Define an environment variable in WINE (see section B.1.3.2.2) called C2WTROOT with the value of the $C2WTROOT directory in Linux, but in the WINE format as described in 1) above.

B.2.2 Sharing C2WTROOT with a Windows OS

If you are using Window operating system (either Windows XP or Windows 7) as your Windows environment, please define a C2WTROOT environment variable as described as the end of this section.

1) First, a share must be set up in the Ubuntu environment:

   A) Make sure the following packages are installed:
      1. samba
      2. nautilus

      It can be installed using the following command in a terminal:

```
sudo apt-get install samba nautilus
```

   B) Once these packages are installed, execute nautilus (a graphical file browser). One way of doing this is to open a terminal and execute the "nautilus" command:

```
nautilus
```

      You should see a window like the one below:

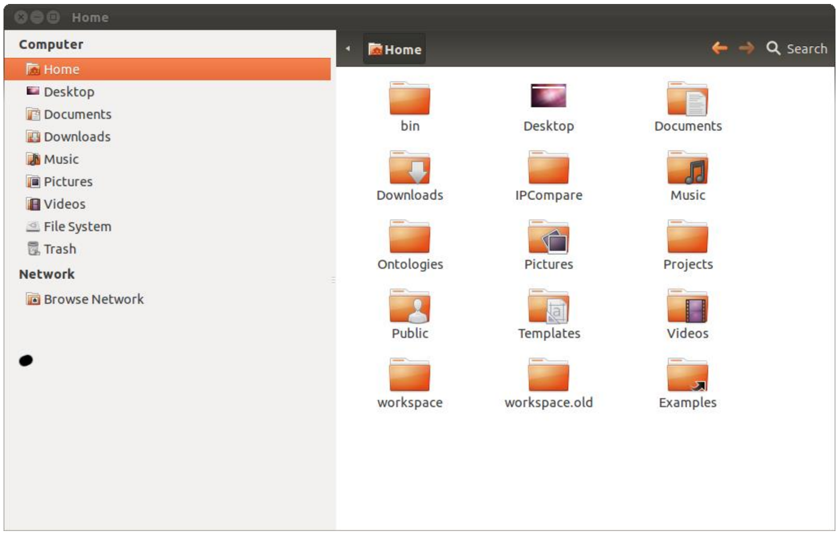


Figure B - 12: Ubuntu File Browser - Nautilus

   C) Navigate the $C2WTROOT directory, right-click on this directory, and select "Sharing options" from the drop-down menu:

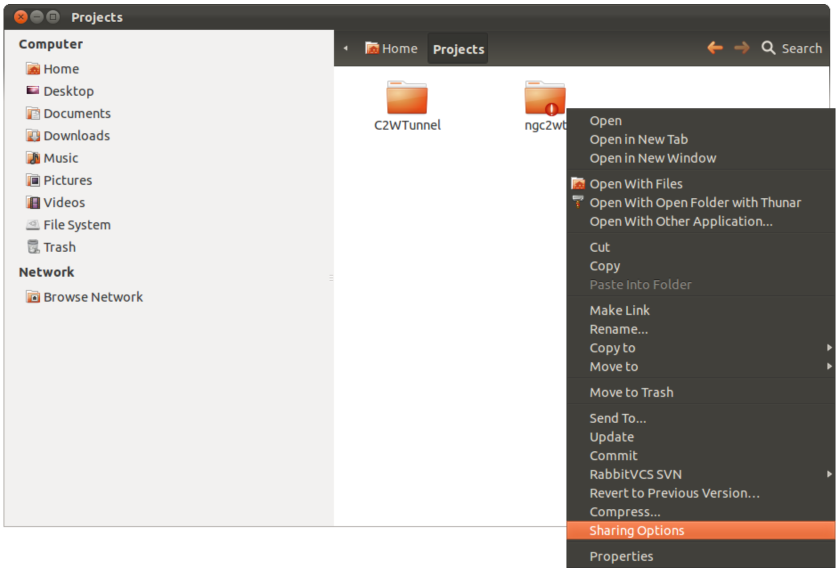
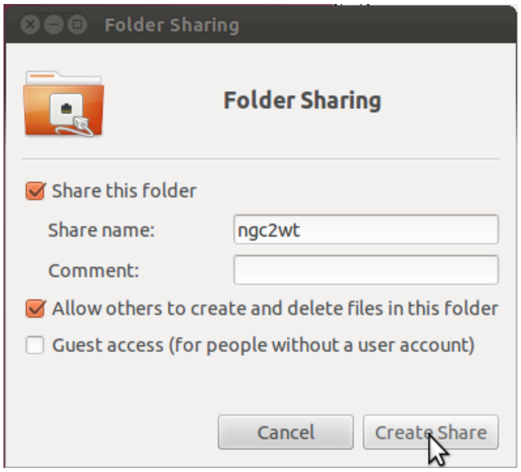


Figure B - 13: Sharing options on a directory

D) A window like the one below should appear. Be sure to check the "Share this folder" checkbox, fill in the share name (e.g. "ngc2wt"), and check the "Allow others to create and delete files in this folder" checkbox. The, click the "Create Share" button. If nautilus asks you to grant more permissions to $C2WTROOT, please grant these permissions.



☑ Share this folder

Share name: ngc2wt

Comment:

☑ Allow others to create and delete files in this folder

☐ Guest access (for people without a user account)

Figure B - 14: Creating a folder share

E) In the original nautilus window that contains $C2WTROOT, the folder icon for $C2WTROOT should have an overlay indicating that it is shared, as highlighted in the below window:

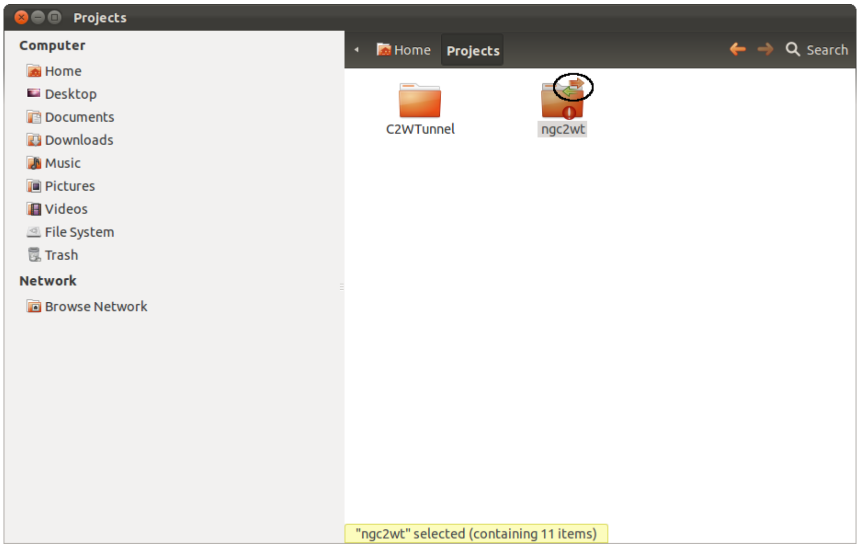


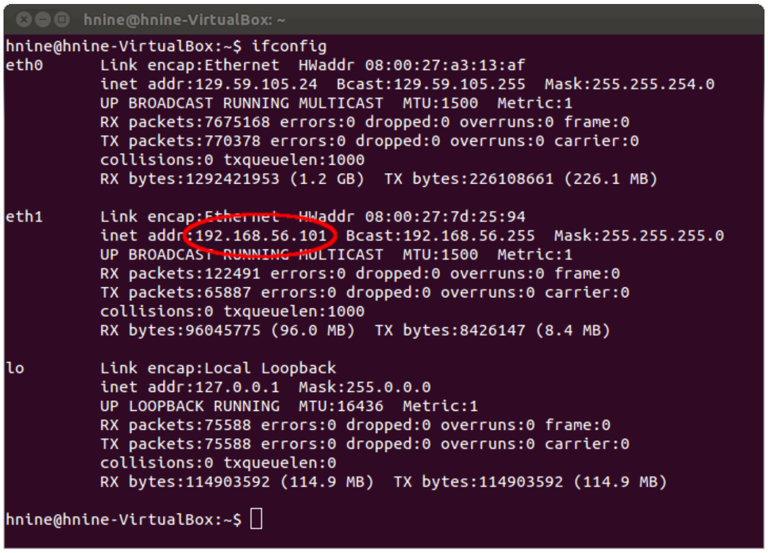
Figure B - 15: Shared folder

2) Now that $C2WTROOT has been made a share in the Ubuntu environment, to access it from the Windows OS, one only need to map it as a network drive:

   A) Acquire the IP address of the Ubuntu environment. One way to do this is, in a terminal, execute the "ifconfig" command:

      ifconfig

      In the resulting output from the command, next to "inet addr:", are the IP addresses of the Ubuntu environment.



```
hnine@hnine-VirtualBox: ~
hnine@hnine-VirtualBox:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:a3:13:af
          inet addr:129.59.105.24  Bcast:129.59.105.255  Mask:255.255.254.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:7675168 errors:0 dropped:0 overruns:0 frame:0
          TX packets:770378 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1292421953 (1.2 GB)  TX bytes:226108661 (226.1 MB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:7d:25:94
          inet addr:192.168.56.101  Bcast:192.168.56.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:122491 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65887 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:96045775 (96.0 MB)  TX bytes:8426147 (8.4 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:75588 errors:0 dropped:0 overruns:0 frame:0
          TX packets:75588 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:114903592 (114.9 MB)  TX bytes:114903592 (114.9 MB)

hnine@hnine-VirtualBox:~$ 
```

Figure B - 16: Checking IP-Address

The IP address you use depends on the network to which the Windows environment is connected. In this example, it is the one circled in red.

B) On the Windows OS (specifically Windows XP, although Windows 7 is similar), open a “My Computer” window in the explorer. This can be done by clicking on the “Start” button and selecting “My Computer”.

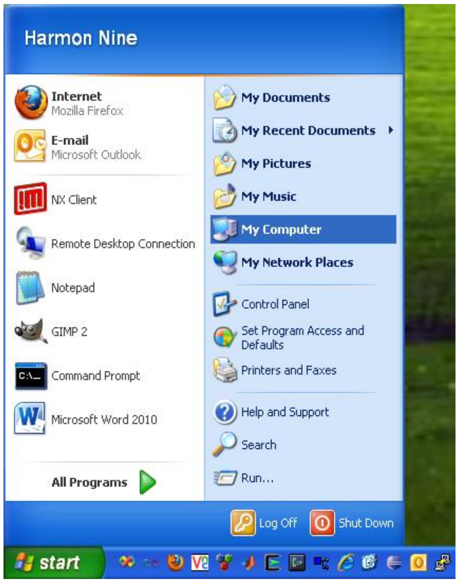


Figure B - 17: Opening ‘My Computer’

C) In the window, select “Tools → Map Network Drive ...”

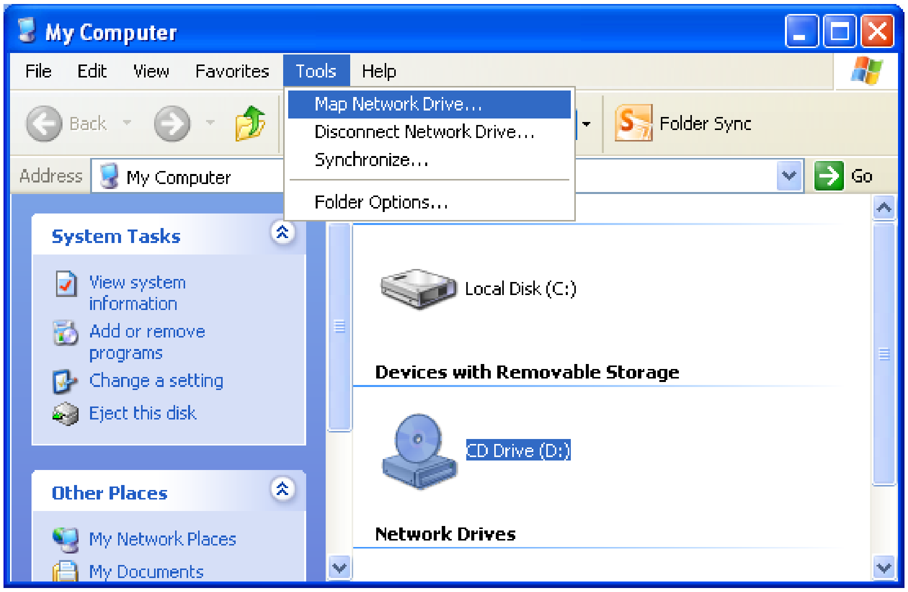
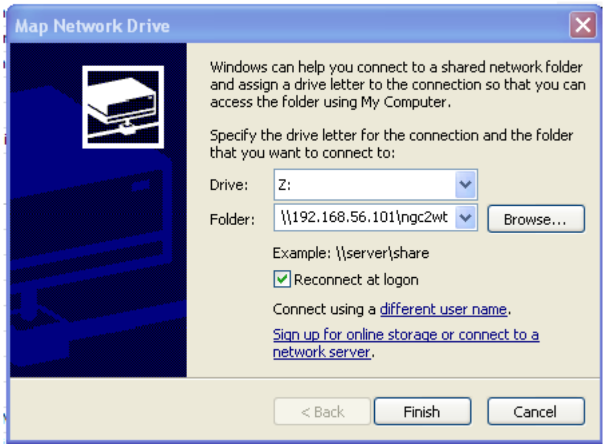


Figure B - 18: Mapping Network Drive

D) In the resulting window, choose the "Drive:" to which to map the Ubuntu share (e.g. "Z:"), and enter the share location using the IP address you gleaned from step A), and the name of the share you specified in step 1.D) above. It should be in the form "\\<ip-address>\<share-name>".
For our example, this would be "\\192.168.56.101\ngc2wt". Also, it is a good idea to check the "Reconnected at logon" checkbox. Then click "Finish".



Map Network Drive

Windows can help you connect to a shared network folder and assign a drive letter to the connection so that you can access the folder using My Computer.

Specify the drive letter for the connection and the folder that you want to connect to:

Drive: Z:

Folder: \\192.168.56.101\ngc2wt

Example: \\server\share

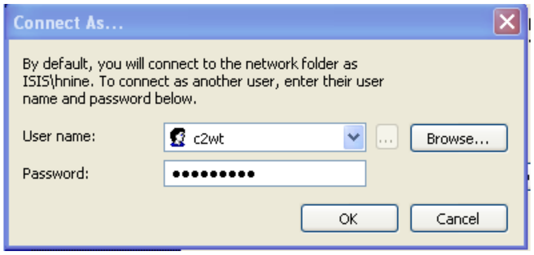
☑ Reconnect at logon

Connect using a different user name.

Sign up for online storage or connect to a network server.

Figure B - 19: Providing network path for mapped drive

E) A window may pop up that asks for a username and password to access the Ubuntu share. Use the username and password of the account under which the share was created.



Connect As...

By default, you will connect to the network folder as ISIS\hnine. To connect as another user, enter their user name and password below.

User name: c2wt

Password: ●●●●●●●●●

Figure B - 20: Entering username and password for network share

3) Define an environment variable in the Windows operating system (see section B.1.3.2.1) called C2WTROOT with the value of the root directory of the drive specified in stop 2E) above. For example, if you chose drive Z:, set C2WTROOT to "Z:\" (without the quotes).

B.3 DOWNLOAD THE C2 WIND TUNNEL SOFTWARE REPOSITORY

The C2 Wind Tunnel software is currently maintained in a subversion (svn) software repository. So, to download the repository into your Linux environment, a subversion client must be installed. The subversion client we will use in this Appendix included in the “subversion” package that is provided for Ubuntu. To install it, in a terminal execute the following command:

```
sudo apt-get install subversion
```

Enter your password if prompted, and answer “yes” to any questions posed.

Recall that, in section B.2, we created a $C2WTROOT directory, and set the C2WTROOT environment variable to this directory. To make sure this environment variable has the correct value, in a terminal, enter the following command:

```
echo “$C2WTROOT”
```

This command should output the directory you want as C2WTROOT. If it doesn't, please go back and create this variable as described at the start of section B.2 and exit all terminals. Then, in a (new) terminal type the above command again. If it still doesn't output the directory, ask for help. Now, in a terminal, change to the $C2WTROOT directory:

```
cd “$C2WTROOT”
```

Next, execute the following command **(the space and dot ' .' at the end of the command is very important)**:

```
svn checkout https://svn.isis.vanderbilt.edu/ngc2wt/trunk .
```

You may be prompted for the username and password for your account on this repository. If you don't currently have an account, please contact ISIS to get one.

At this point, if you list the contents of your $C2WTROOT directory (using the 'ls' command), you should see something like the following:

```
> ls

3rdParty  Contrib  ExecutionDirectory  log              src
build     doc      generated           readme-license.txt
```

If all you see is:

```
> ls

trunk
```

you probably forgot the space and dot at the end of the 'svn' command above. Please execute the following command:

```
rm -rf trunk
```

And, re-execute the svn command above with the space and dot.

B.4 C2 WIND TUNNEL SOFTWARE INSTALLATION

Please download and install the software listed on the following pages and make sure it is working properly. This software can easily be obtained from the web and does not require special configuration.

B.4.1 Java

Java is used extensively in the C2 Wind Tunnel, both in the Ubuntu Linux and Windows environments.

B.4.1.1 Installing Java into the Linux environment. The C2 Wind Tunnel has not been tested with any Java distribution other than that provided by Oracle, so it is this distribution that should be installed. To do this, enter the following command:

```
bash $C2WTROOT/3rdParty/java/install/oab-java.sh
```

This will build ".deb" Ubuntu package files for Oracle's Java, and make them available for installation. Then, enter the following command on one line. Accept all license agreements, and answer "yes" to any questions posed:

```
sudo apt-get install sun-java6-bin sun-java6-fonts sun-java6-jdk sun-java6-jre sun-java6-plugin
```

This will install all necessary java packages into Ubuntu. Now, importantly enter the following command:

```
sudo update-java-alternatives --set java-6-sun
```

This will select Oracle's java as the primary java installation to use in your Ubuntu Linux environment (there are other java installations available, like openjdk, but these have not been tested with the C2 Wind Tunnel).

Now, define the following persistent environment variable (see section B.1.3.1) with the value as shown:

```
Variable Name:   JAVA_HOME
Variable Value:  /usr/lib/jvm/java-6-sun
```

B.4.1.2 Installing Java into the Windows environment. Java is installed into your Windows in essentially the same way whether you are using WINE or a Windows operating system.

IMPORTANT: Please make sure that no JDK/JRE is already installed on your system (i.e. no directory like 'C:\Program Files\Java\" should be present). If there are, please remove them using the Add/Remove Programs utility in Windows.

1) Go to JDK download page. If you are using WINE, use a browser in your Linux environment. If you are using a Windows OS, use a browser in the Window OS.

   A) Using your browser, go to http://www.oracle.com/technetwork/java/javase/downloads/index.html

   B) Click the "Download" button under “JDK” in the area titled "Java SE 6 Update VN”, where VN is the latest version of Java 6 (31 at time this Appendix was being written).

2) After getting to the required download page:

   A) Click on the “Accept License Agreement” radio button.

   B) Click on “jdk-6uVN-windows-i586.exe” on the line that starts with “Windows x86 (32-bit)”.

   C) Click “Save File” in the resulting pop-up window.

3) Once "jdk-6uVN-windows-i586.exe" is downloaded, execute it (AS ADMINISTRATOR IN WINDOWS 7) to install Java Development Kit 6.0 Update VN:

   - For WINE, execute the following in a terminal:

     ```
     wine <PATH>
     ```

     where <PATH> is the path of the jdk-6uVN-windows-i586.exe, e.g. /home/c2wt/Downloads/jdk-6uVN-windows-i586.exe

   - In a Windows OS, execute the jdk-6uVN-windows-i586.exe Java installer using a command-prompt or explorer window.

4) Define the following persistent environment variables with the values as shown below. See section B.1.3.2.1 if you are using a Windows OS, or section B.1.3.2.2 if you are using WINE:

Table B - 1: Environment Variables need to be set

Variable Name	Variable Value
JAVA_HOME	C:\Program Files\Java\jdk1.6.0_VN
JDK_PATH	%JAVA_HOME%
JDK_HOME	%JAVA_HOME%

B.4.2 GME

GME is used for CASIM model creation and code-generation, and so is of paramount importance for the C2 Wind Tunnel. Currently, GME only runs in a Windows environment. We recommended you install the latest version of GME, particularly if you are using WINE (at least version 12.2.7, as it is known to work with WINE).

B.4.2.1 Installation of GME. Below are the instructions for installing GME:

1) Create a “C:\Temp” directory in your Windows environment:

   - For WINE, this can be done using a terminal in the Ubuntu Linux environment and executing the command:

     ```
     mkdir <HOME>/.wine/drive_c/Temp
     ```

     Where <HOME> is your home directory. Incidentally, <HOME>/.wine/drive_c is the root of the directory structure that WINE uses for drive “C:”.

   - In a Windows OS, use the explorer or a command-prompt to create the “C:\Temp” directory.

2) Using your browser, go to http://escher.isis.vanderbilt.edu/downloads/. If you are using WINE, use a browser in your Linux environment. If you are using a Windows OS, use a browser in the Window OS.

3) Click on "GME" (which is to the left of the words “Generic Modeling Environment (GME)”)

4) Enter your company name, click on “I accept the agreement”, and then click on the “Submit” button.

5) Click on "GME-VN.msi", where VN is the most recent version of GME.

6) Save GME-VN.msi to your disk by clicking "Save" in the resulting pop-up window.

7) Once "GME-VN.msi" is downloaded, execute it (AS ADMINISTRATOR IN WINDOWS 7) to install GME:

   - For WINE, execute the following command in a terminal:

     ```
     winefile
     ```

     and navigate to <PATH>, where <PATH> is the path of GME-VN.msi (e.g. /home/c2wt/Downloads/GME-VN.msi). Double-click on GME-VN.msi in the winefile window and the GME installation process should commence.

- In a Windows OS, execute the GME-VN.msi using a command-prompt or explorer window.

B.4.2.2 GME Configuration. There are several steps involved in configuring GME for the C2 Wind Tunnel: registering the CASIM paradigm, registering components for this paradigm, and registering interpreters for this paradigm.

B.4.2.2.1 Registering the CASIM paradigm. Here are the steps to register CASIM paradigm:

1) Execute GME:

   - For WINE, execute the following command in a terminal (the single quotes are very important):

     ```
     wine 'C:\Program Files\GME\Bin\GME.exe'
     ```

   - For a Windows OS, execute the “C:\Program Files\GME\Bin\GME.exe” using a command-prompt or explorer window.

2) Import the CASIM paradigm.

   A. From the "File" menu, select "Import XML"

   B. Select the following file (*don't actually type $C2WTROOT – manually substitute it with its value in your Windows environment whether you are using WINE or a Windows OS*).

      ```
      $C2WTROOT\src\c2w\models\gme\hla\CASIM.xme
      ```

      Recall that $C2WTROOT is going to be on the Z: drive for WINE, and whatever drive you mapped the Ubuntu share to in a Windows OS.

   C. Save the paradigm as “CASIM.mga”

3) Register the CASIM paradigm

   A. Click on the icon that is shaped like a gear (or "cog") that is located on the tool-bar near the top of the GME window. It is on the right-hand side of this tool-bar. (The tooltip for this icon, which you can see by centering the mouse pointer over the icon, is "MetaGME interpreter").

   B. Save the interpreted CASIM paradigm as "CASIM.xmp".

   C. Press "Ok" in the succeeding windows.

D. When the window appears that has the text "Successfully generated $C2WTROOT\src\c2w\models\gme\hla\CASIM.xmp. Would you like to register your new paradigm?", Click "Yes".

E. Accept defaults on any other dialogs.

F. *DO NOT EXIT GME YET – Proceed directly to the next section to register the HLADecorator component*.

B.4.2.2.2 Registering the GME Components for CASIM. The GME Components for CASIM paradigm make the development and examination of CASIM models easier. There are two components: HLADecorator and HLAHelper

B.4.2.2.2.1 Registering the HLADecorator component for CASIM. When we are working with CASIM models, it is convenient to have the "HLADecorator" plugin installed as it allows CASIM Interaction-parameters and Object-attributes to be displayed as part of an interaction or object respectively. Depending on whether you have Microsoft Visual Studio 2008+ or not, use the following methods to access the HLADecorator.dll file:

1) If you do not have Microsoft Visual Studio 2008+ (i.e. 2008 or later) installed, proceed to the next step. Otherwise, open the $C2WTROOT\src\c2w\cpp\HLADecorator\HLADecorator.sln solution file in Visual Studio and build the “Release” version.

2) If you exited GME in the previous section, open CASIM.mga project in GME before proceeding:

   A. Execute GME as in step 1) of section B.4.2.2.1.

   B. The CASIM.mga file (along with its full path) should appear near the bottom of the “File” menu. Select it to open the CASIM.mga project.

3) To register the HLADecorator:

   A. Go to “Tools → Register Components ...” in GME.

   B. In the resulting window, select “Systemwide” radio button in “Register:” choices and click on “Install New...”

   C. In the resulting window, navigate to and select the HLADecorator.dll file:

      - If you built the HLADecorator using Visual Studio in step 1) above, select $C2WTROOT\src\c2w\cpp\HLADecorator\Release\HLADecorator.dll

    - If you don't have Visual Studio, select $C2WTROOT\src\c2w\models\gme\hla\HLADecorator.dll, which is a pre-compiled dll that comes with the C2 Wind Tunnel.

D. Double-click on the HLADecorator.dll file, or click on the “Open” button. This will install the HLADecorator and close the window.

E. In the original window, click the “Close” button.

F. *DO NOT EXIT GME YET – Proceed directly to the next section to register the HLADecorator component.*

B.4.2.2.2.2 Registering the HLAHelper component for CASIM. When we are working with CASIM models, it is convenient to have the "HLADecorator" plugin installed as it allows HLA Interaction-parameters and Object-attributes to be displayed as part of an interaction or object respectively. Depending on whether you have Microsoft Visual Studio 2008+ or not, use the following methods to access the HLADecorator.dll file.

Interactions and Objects in the CASIM paradigm have an inheritance hierarchy. As inheritance dictates, the parameters and attributes of a base Interaction or Object class are available in a derived class. The “HLAHelper” add-on enhances the GME modeling environment for CASIM to visually display inherited parameters in a derived class. Do the following to install the HLAHelper component for the CASIM paradigm:

1) If you do not have Microsoft Visual Studio 2008+ (i.e. 2008 or later) installed, proceed to the next step. Otherwise, open the $C2WTROOT\src\c2w\cpp\HLAHelper\HLAHelperComponent.sln solution file in Visual Studio and build the “Release” version.

2) If you exited GME in the previous section, open CASIM.mga project in GME as in step 2) of section B.4.2.2.2.1.

3) To register the HLAHelper:

    A. Go to “Tools → Register Components ...” in GME.

    B. In the resulting window, select “Systemwide” radio button in “Register:” choices and click on “Install New...”

    C. In the resulting window, navigate to and select the HLAHelperComponent.dll file:

        - If you built the HLAHelperComponent using Visual Studio in step 1) above, select $C2WTROOT\src\c2w\cpp\HLAHelper\Release\HLAHelperComponent.dll

- If you don't have Visual Studio, select $C2WTROOT\src\c2w\models\gme\hla\HLAHelperComponent.dll, which is a pre-compiled dll that comes with the C2 Wind Tunnel.

D. Double-click on the HLAHelperComponent.dll file, or click on the “Open” button. This will install the HLAHelper and close the window.

E. In the original window, click the “Close” button.

F. *OK, now you can exit GME.*

B.4.2.1.3 Registering Java-based GME interpreters for CASIM. The interpreters for CASIM generate Java, C++, and other files based on the structure of a given CASIM model. In addition to registering these interpreters, GME must be configured to look in the necessary directories for needed Java-class files so that these interpreters can execute.

B.4.2.1.3.1 Modifying the GME classpath. Here are the steps to modify the GME classpath:

1) Execute the registry editor for your Windows environment:

    - For WINE, enter the following command at a terminal (the single-quotes are very important):

      ```
      wine 'C:\windows\regedit.exe'
      ```
    - For a Windows OS, in particular Windows XP, where the instructions for Windows 7 are similar:
        A) On the Windows desktop, click "Start" -> "Run..."
        B) In the resulting pop-up window, enter "regedt32" in the text field.
        C) Click "OK" to execute "regedt32", the registry editor.
2) In the registry editor, in the left window, navigate to select the "HKEY_LOCAL_MACHINE\SOFTWARE\GME" folder.
3) In the right window, RIGHT-click on "JavaClassPath" and select "Modify".
4) In the resulting pop-up window, enter data below in "Value data:" text field.

    PLEASE DO THIS:
    - *Without the newlines*
    - *Replace $JAVA_HOME, $GME_ROOT, and $C2WTROOT with their values. Their values can be seen by navigating to “HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\ Session Manager\Environment” in the registry editor*
    - Here's the data:

      ```
      $JAVA_HOME\jre\lib\rt.jar;
      $JAVA_HOME\lib\tools.jar;
      $GME_ROOT\SDK\Java\gme.jar;
      $C2WTROOT\3rdParty\java\britney\CPNToolsSimulator.jar;
      $C2WTROOT\3rdParty\java\britney\Headless.jar;
      $C2WTROOT\3rdParty\java\britney\NetLoader.jar;
      $C2WTROOT\3rdParty\java\britney\NetModel.jar;
      ```

$C2WTROOT\3rdParty\java\britney\Simulation.jar;
$C2WTROOT\3rdParty\java\britney\Simulator.jar;
$C2WTROOT\3rdParty\java\stringtemplate\antlr-3.1.3.jar;
$C2WTROOT\3rdParty\java\stringtemplate\stringtemplate-3.2.jar

5) Click “OK”
6) Exit the registry editor

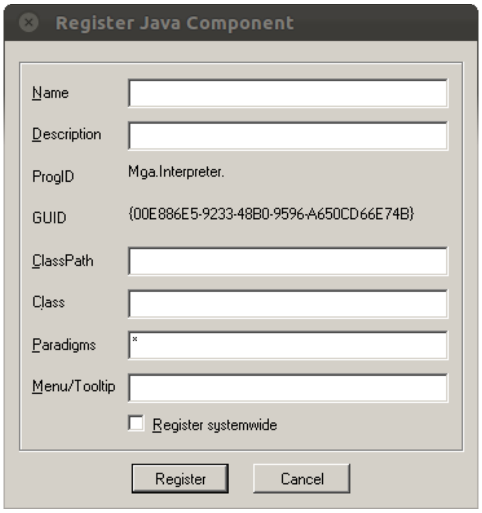
B.4.2.1.3.2 Registering the interpreters. To register the interpreters, you must execute the JavaCompRegister.exe in your Windows environment.

- For WINE, in your Ubuntu Linux environment, enter the following command in a terminal (the single quotes are very important):

```
wine 'C:\Program Files\GME\SDK\Java\JavaCompRegister.exe'
```

- In a Windows OS, execute "C:\Program Files\GME\SDK\Java\JavaCompRegister.exe" in a command window or by using the file explorer.

When JavaCompRegister.exe is executed, the following window will appear:



Register Java Component

Name	
Description	
ProgID	Mga.Interpreter.
GUID	{00E886E5-9233-48B0-9596-A650CD66E74B}
ClassPath	
Class	
Paradigms	*
Menu/Tooltip	

☐ Register systemwide

Figure B - 21: Registering GME components

JavaCompRegister.exe must be run 4 separated times, once for each table below.

- The tables show the data that should be entered into the JavaCompRegister.exe window.
- After all data has been entered for a particular run, click “Register” button.
- *$C2WTROOT should be substituted with its actual value, as it was in step 2 of section B.4.2.1.3.1.*

Table B - 2: Values for registering C2W Main Interpreter

Field/Checkbox Name	Data/Action
Name	C2WInterpreter
Description	C2WInterpreter
ClassPath	$C2WTROOT\build\c2w\java
Class	c2w.hla.gmeinterpreter.Interpreter
Paradigm	CASIM
Menu/Tooltip	C2W Main Interpreter
Register systemwide	(check this checkbox)

Table B - 3: Values for registering C2W Deployment Interpreter

Field/Checkbox Name	Data/Action
Name	C2WDeploymentInterpreter
Description	C2WDeploymentInterpreter
ClassPath	$C2WTROOT\build\c2w\java
Class	c2w.hla.gmeinterpreter.DeploymentInterpreter
Paradigm	CASIM
Menu/Tooltip	C2WDeploymentInterpreter
Register systemwide	(check this checkbox)

Table B - 4: Values for registering CPN Import Interpreter

Field/Checkbox Name	Data/Action
Name	CPNInterpreter
Description	CPNInterpreter
ClassPath	$C2WTROOT\build\c2w\java
Class	c2w.hla.gmeinterpreter.CPNImportInterpreter
Paradigm	CASIM
Menu/Tooltip	CPN Import Interpreter
Register systemwide	(check this checkbox)

Table B - 5: Values for registering Fedfile Import Interpreter

Field/Checkbox Name	Data/Action
Name	C2WFedFileInterpreter
Description	C2WFedFileInterpreter
ClassPath	$C2WTROOT\build\c2w\java
Class	c2w.hla.gmeinterpreter.fedfile.FEDImportInterpreter
Paradigm	CASIM
Menu/Tooltip	Fedfile Import Interpreter
Register systemwide	(check this checkbox)

B.4.3 BOOST

Boost is needed in the Ubuntu Linux environment for the C++ code-generation component of C2 Wind Tunnel. To install, execute the following command in a terminal (see section B.1.2.2 for an explanation of the "sudo" command):

```
sudo apt-get install libboost-all-dev
```

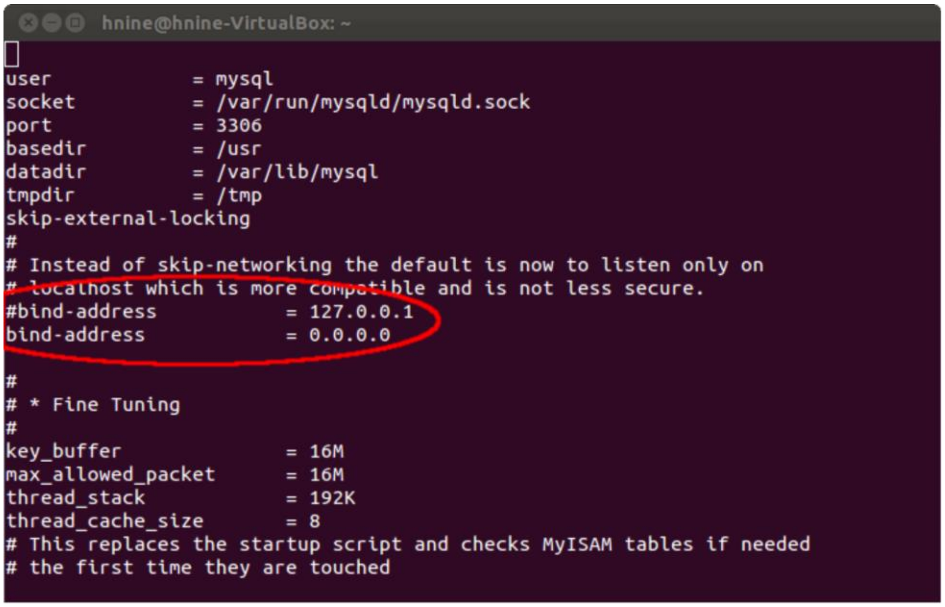
Enter your password if needed, and answer "yes" to any questions posed.

B.4.4 MYSQL SERVER

A MySQL database is used to store the event history of a given execution of the C2 Wind Tunnel.

1) To install MySQL server in your Ubuntu Linux Environment (see section B.1.2.2 for an explanation of the "sudo" command):
   A) Open a terminal and execute:
      ```
      sudo apt-get install mysql-server.
      ```
      Answer "y" to any yes/no question posed.
   B) The installation process might ask you to specify a password for the root user of the MySQL server. Enter "**c2wt**" (without quotes, all lowercase).
   C) If the installation process does not ask you to specify a password for the root user, execute this command in a terminal:
      ```
      sudo dpkg-reconfigure mysql-server
      ```

2) To configure MySQL server, two things must be done:
   A) First, MySQL must be configured to accept connections from any machine. To do this:
      1. Edit the /etc/mysql/my.cnf file using your favorite editor (e.g. the "vi" editor). Keep in mind that you must execute the editor as "root" (you can use "sudo" to do this):
         ```
         sudo vi /etc/mysql/my.cnf
         ```
         The line containing "bind-address = xxx.xxx.xxx.xxx" must be changed to:
         ```
         bind-address = 0.0.0.0
         ```
         This is shown in the figure below. Once this is done, save the file and exit the editor.



```
user                = mysql
socket              = /var/run/mysqld/mysqld.sock
port                = 3306
basedir             = /usr
datadir             = /var/lib/mysql
tmpdir              = /tmp
skip-external-locking
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address               = 127.0.0.1
bind-address                = 0.0.0.0

#
# * Fine Tuning
#
key_buffer                  = 16M
max_allowed_packet          = 16M
thread_stack                = 192K
thread_cache_size           = 8
# This replaces the startup script and checks MyISAM tables if needed
# the first time they are touched
```

Figure B - 22: Configuring MySQL Server

2. MySQL must now be restarted. To do this, in a terminal enter the following command:
   sudo restart mysql

B) Now, MySQL must be configured to allow connections from other hosts to have permission to create and manipulate databases. To do this, first execute the "mysql" command in a terminal as follows:
   mysql --user=root --password=c2wt

   At the resulting "mysql>" prompt, enter the following command:
   GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'c2wt' WITH GRANT OPTION;

   This should all be typed on one line, and all punctuation is very important. MySQL should respond with:
   Query OK, 0 rows affected (0.00 sec)

   At this point, configuration of MySQL is complete, so simply quit by entering at the "mysql>" prompt:
   quit

B.4.5 MySQL C++ Connector

The MySQL C++ Connector is needed in the Ubuntu Linux environment for the C++ code-generation component of C2 Wind Tunnel. To install, execute the following command in a terminal (see section B.1.2.2 for an explanation of the "sudo" command):

```
sudo apt-get install libmysqlcppconn-dev
```

Answer “y” to any questions posed.

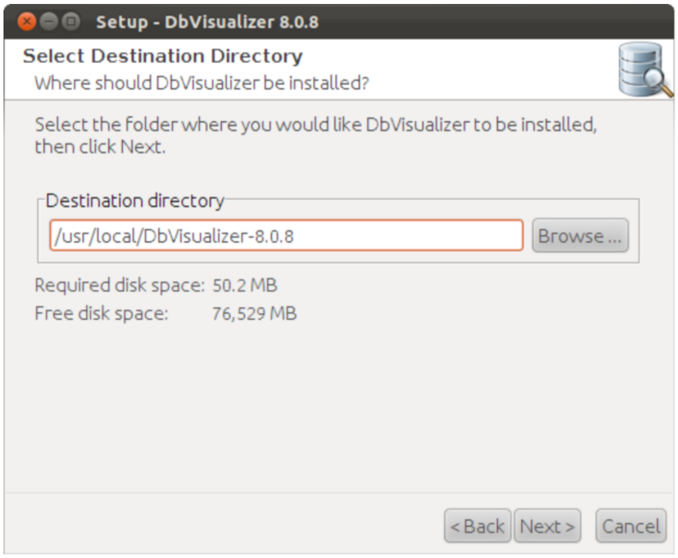
B.4.6 DbVisualizer

DbVisualizer provides a means for visually examining the MySQL database, in particular the events that are recorded there during a C2 Wind Tunnel simulation.

1) To install DbVisualizer, do the following:

    A) Download the DbVisualizer setup installer package:
    1. Using your browser in your Ubuntu Linux environment, go to the following URL: http://www.dbvis.com/products/dbvis/download/
    2. On the resulting page, click on “Download” next to “Linux x86 (setup installer) in the “Without Java VM” column.
    3. Save the file dbvis_linux_VN.sh using the resulting pop-up window, where VN is the version of DbVisualizer you are downloading.

    B) Now we must run dbvis_linux_VN.sh:
    1. In a terminal, type the following command:
       ```
       sudo bash <PATH>
       ```
       where <PATH> is the path to the dbvis_linux_VN.sh file.
    2. In the resulting window, click “Next”.
       In the next window, accept the license agreement and click “Next”.
    3. We'll install DbVisualizer in the “/usr/local/DbVisualizer-VN” directory, so in the next window, enter “/usr/local” in the “Destination directory” textbox. Then click “Next”.



Select Destination Directory

Where should DbVisualizer be installed?

Select the folder where you would like DbVisualizer to be installed, then click Next.

Destination directory: /usr/local/DbVisualizer-8.0.8

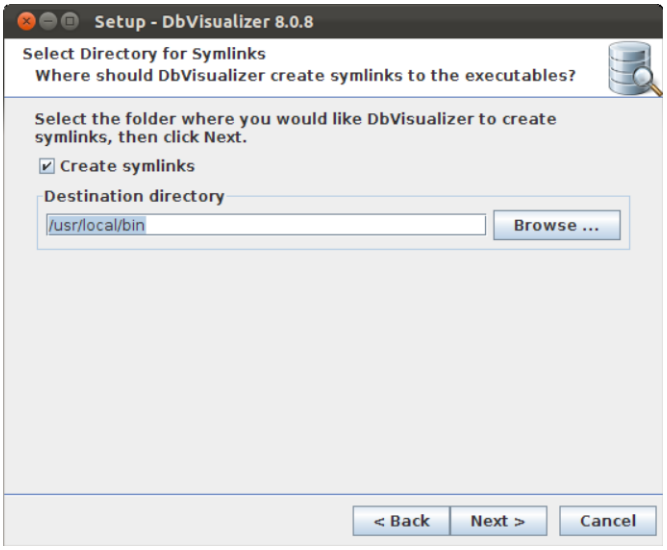
Required disk space: 50.2 MB

Free disk space: 76,529 MB

Figure B - 23: Setting DbVisualizer installation directory

4. In the next window, “DbVisualizer VN” and “Database Drivers (JDBC)” should both be selected for installation. Then click “Next”.

5. In the next window, make sure the “Create symlinks” checkbox is checked, and “/usr/local/bin” is entered in the “Destination directory” textbox. Click “Next”.



Setup - DbVisualizer 8.0.8

Select Directory for Symlinks

Where should DbVisualizer create symlinks to the executables?

Select the folder where you would like DbVisualizer to create symlinks, then click Next.

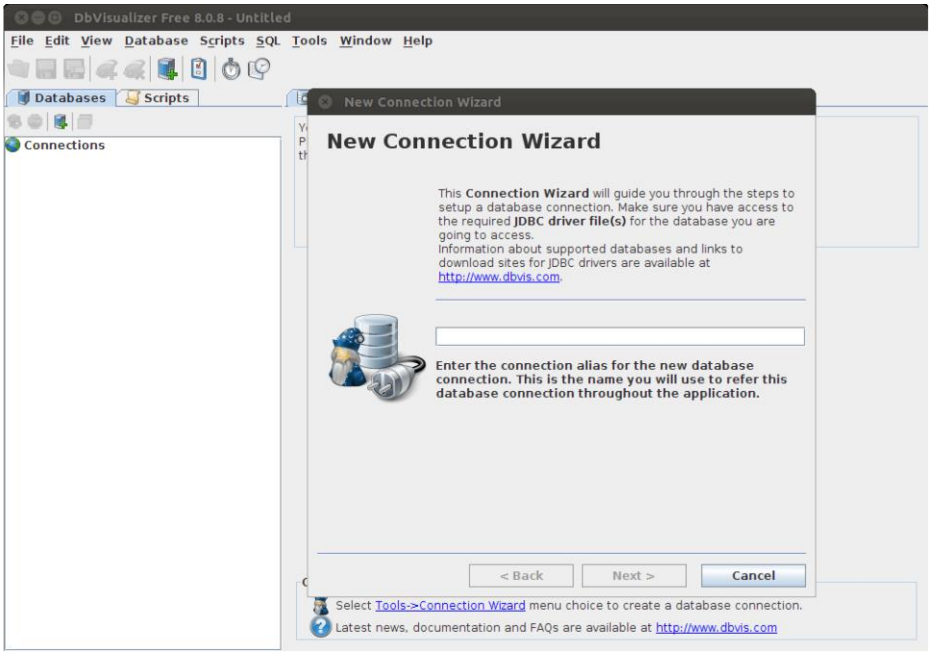
☑ Create symlinks

Destination directory: /usr/local/bin

Figure B - 24: Setting executable symlinks location for DbVisualizer

6. In the next window, there is a checkbox that appears to have no effect whether it is checked or not. Click “Next”.
7. The next window will show the progress of installing DbVisualizer.
8. In the final window, make sure the “Run DbVisualizer” checkbox is checked, and click “Finish”.

2) To configure DbVisualizer for the C2 Wind Tunnel:
   A) Run DbVisualizer either via step 1.B.9 above, or just type the following into a terminal:
   /usr/local/bin/dbvis
   B) When initially run with no database connections, DbVisualizer should bring up the “New Connection Wizard”. If it doesn't, then select
   “Database → Create Database Connection”, and click “Use Wizard” in the resulting pop-up window:



DbVisualizer Free 8.0.8 - Untitled
File Edit View Database Scripts SQL Tools Window Help
Databases
Scripts
Connections
New Connection Wizard
< Back
Next >
Cancel

New Connection Wizard

This **Connection Wizard** will guide you through the steps to setup a database connection. Make sure you have access to the required **JDBC driver file(s)** for the database you are going to access.
Information about supported databases and links to download sites for JDBC drivers are available at http://www.dbvis.com.

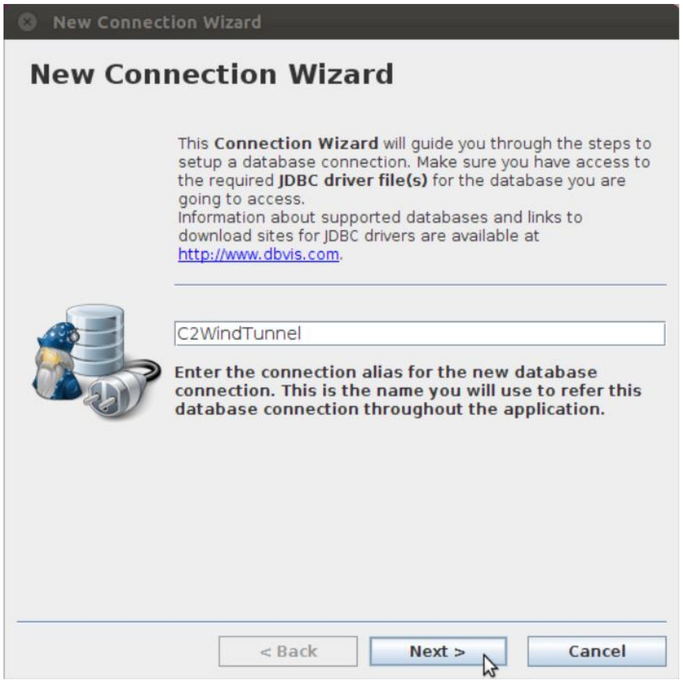
Enter the connection alias for the new database connection. This is the name you will use to refer this database connection throughout the application.

Select Tools->Connection Wizard menu choice to create a database connection.

Latest news, documentation and FAQs are available at http://www.dbvis.com

Figure B - 25: Setting up new database connection in DbVisualizer

C) In the initial “New Connection Wizard”, enter “C2WindTunnel” into the textbox and click “Next”.



New Connection Wizard

This **Connection Wizard** will guide you through the steps to setup a database connection. Make sure you have access to the required **JDBC driver file(s)** for the database you are going to access.
Information about supported databases and links to download sites for JDBC drivers are available at http://www.dbvis.com.

C2WindTunnel

Enter the connection alias for the new database connection. This is the name you will use to refer this database connection throughout the application.

Figure B - 26: Entering database name for the new database connection in DbVisualizer

D) In the resulting “Select Database Driver” window, select “MySQL” from the drop-down menu. Then click “Next”.

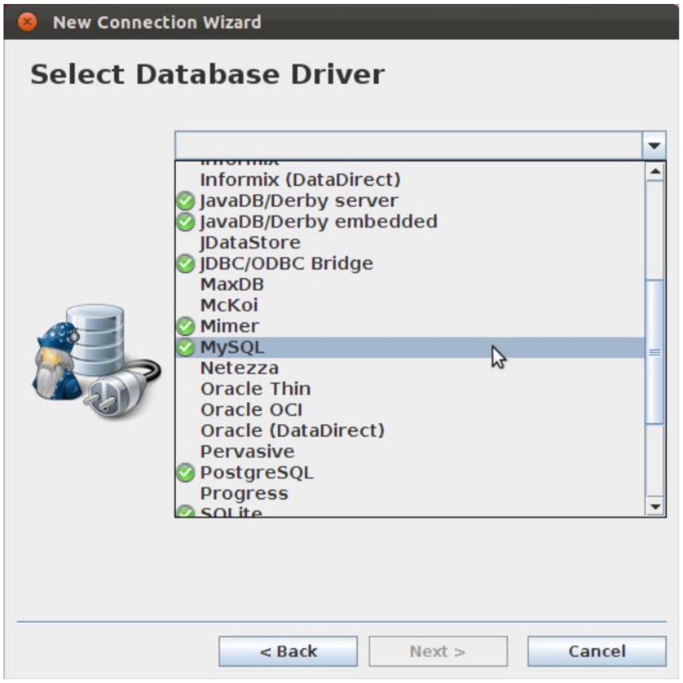
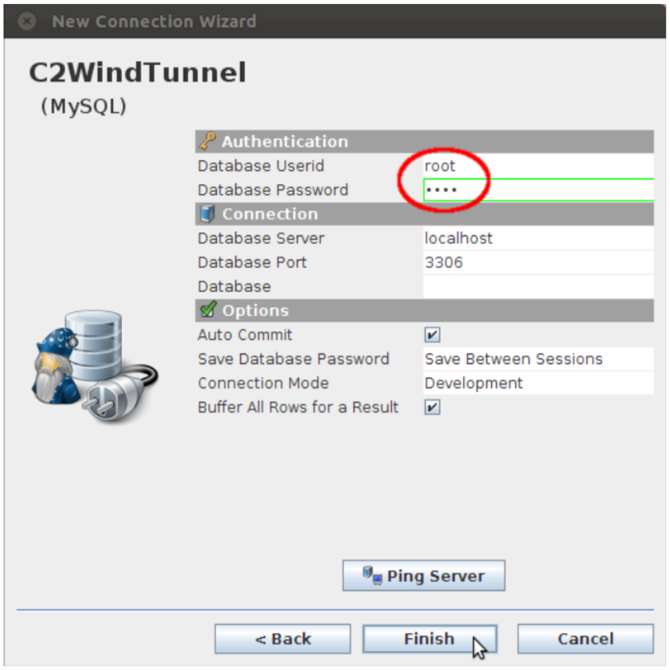


Figure B - 27: Selecting database driver for the new database connection in DbVisualizer

E) In the final window, enter “root” for the Database Userid and “c2wt” as the “Database Password” (this will appear as dots when entered). Leave everything else as is. Click “Finish”. Your new “C2WindTunnel” database connection should appear under the “Connections” heading in the left-had sub-window of the DbVisualizer main window.



C2WindTunnel
(MySQL)

Authentication	
Database Userid	root
Database Password	••••
Connection	
Database Server	localhost
Database Port	3306
Database	
Options	
Auto Commit	☑
Save Database Password	Save Between Sessions
Connection Mode	Development
Buffer All Rows for a Result	☑

Figure B - 28: Entering username and password for the new database connection in DbVisualizer

B.4.7 Java3D

To install Java3D in your Ubuntu Linux environment, type the following command at a terminal:

```
sudo apt-get install libjava3d-java
```

The dependent packages “libjava3d-jni” and “libvecmath-java” should be installed automatically when this command executes.

B.4.8 Java ProcessId JNI Module

The java ProcessId JNI module allows a java process to be managed by the C2 Wind Tunnel, it particular so it may be terminated automatically. To build this module, execute the following command in a terminal.

```
bash $C2WTROOT/src/c2w/java/c2w/util/process/buildProcessIdJNI.sh
```

B.4.9 PORTICO

Portico provides the RTI functionality of the C2 Wind Tunnel, and so a key component of the C2 Wind Tunnel. It is installed in the Ubuntu Linux environment.

VERY IMPORTANT

Portico uses IP Multicast for network communication. In the versions of Ubuntu we have used for the C2 Wind Tunnel, IP Multicast will not work if IP version 6 (IPv6) is enabled. Therefore IPv6 must be disabled.

To disable IPv6 in Ubuntu, edit the file "/etc/sysctl.conf" AS ROOT and insert the following 3 lines at the end of the file:

```
net.ipv6.conf.all.disable_ipv6=1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

This is shown below, using the "vi" editor in a terminal window, i.e. the command to edit the /etc/sysctl.conf file is:

```
sudo vi /etc/sysctl.conf
```

ONCE YOU HAVE MADE THIS EDIT TO /etc/sysctl.conf, REBOOT YOUR SYSTEM.

The most recent version of portico known to work with the C2 Wind Tunnel is version 1.0.2. From this point onward, "VN", in reference to Portico will mean "1.0.2". To install portico, so the following:

1) Using your browser, go to http://sourceforge.net/projects/portico/files/Portico/
2) Click on "portico-VN"
3) On the resulting page, scroll down and click on "portico-VN-linux.tar.gz".
4) In the resulting window select to save it, and press the "OK" or "Save" button.
5) Extract the contents of portico-VN-linux.tar.gz at an install location (e.g., /opt/portico), and give read and execute permissions to everyone. This can be done by entering the following commands:
   ```
   cd /opt
   sudo mkdir portico
   cd /opt/portico
   sudo tar xf <PATH>
   sudo chmod -R ugo+rx .
   ```
   where <PATH> is the path of the portico-VN-linux.tar.gz file, i.e. where it was saved by your browser. THE SPACE AND PERIOD ' .' AT THE END OF THE LAST COMMAND ARE VERY IMPORTANT.

   These commands will result in the creating of the directory "/opt/portico/portico-VN", which should contain files LICENSE.portico and RTI.rid.
6) Create a persistent environment variable RTI_HOME and give it the value "/opt/portico/portico-VN" (again VN is the current version of portico, e.g. 1.0.2). See section B.1.3.1 for how to create a persistent environment variable in the Ubuntu Linux environment, i.e. put this command at the end of the appropriate file:
   ```
   export RTI_HOME="/opt/portico/portico-VN"
   ```

B.4.10 CPNTools

CPNTools is a Colored Petri Net (CPN) development and simulation tool that is used in the C2 Wind Tunnel to model decision-making processes. CPNTools is installed into the Ubuntu Linux environment. The version that works with the C2 Wind Tunnel is 2.2.0. From this point onward "VN" will be used to refer to this version.

1) Using your browser, go to http://cpntools.org/download. You might need to apply for a license and you may choose to be notified of future release announcements.
2) On the resulting page, in the "Linux/Mac OS X" section, click on "Download Linux Version VN".
3) Save "cpntools_VN.tar.gz" to your disk by clicking "Save" in the resulting pop-up window.
4) Once "cpntools_VN.tar.gz" is downloaded, extract it to a location on the hard drive (e.g. /opt/CPNTools). From this point forward, this directory will be referred to as $CPNTOOLS_DIR.
5) Set the persistent environment variable "PATH" to the following value, i.e. add the following line to the BOTTOM of the appropriate file (see section B.1.3.1 for how to set persistent environment variables in the Ubuntu Linux environment):
   ```
   export PATH="$PATH:$CPNTOOLS_DIR"
   ```

   Substitute $CPNTOOLS_DIR with its actual value from step 4) above. Note that is command will append ":$CPNTOOLS_DIR" to the end of value of the PATH environment variable.
6) To use CPNTools for CPN development, make sure that 3D acceleration is enabled for your Ubuntu Linux environment. If your Ubuntu Linux environment is inside of a Virtual Machine, then use its Display settings to enable 3D acceleration. DO THIS ONLY IF YOU PLAN ON DEVELOPING CPN'S USING CPNTools. We've had display problems on Ubuntu VirtualBox VM's that have 3D acceleration enabled.
7) Check if CPNTools is correctly installed, execute the following commands in a terminal (substitute $CPNTOOLS_DIR with its actual value from step 4) above):
   ```
   cd $CPNTOOLS_DIR
   ./cpntools
   ```

B.4.11 OMNeT++

OMNeT++ is a used for network simulation in the C2 Wind Tunnel. The latest version that works with C2WT is 4.2.1. From this point, "VN" will refer to the 4.2.1 version of OMNeT++. Also, OMNeT++ modules in CASIM work only in Ubuntu Linux, so we install it in Ubuntu.

B.4.11.1 OMNeT++ Installation. To install OMNeT++ version VN:

1) Using your browser, go to http://www.omnetpp.org/
2) Click on the big green "Download" button on the left-hand-side of the resulting page.
3) On the resulting page, click on "OMNeT++ VN (source + IDE, tgz)". If you don't see version VN you want to download:
   A) Click on "Older versions" at the top of the screen

B) On the resulting page, click on “OMNeT++ VN (source + IDE, tgz)”, where VN is the version you wish to download (e.g. 4.2.1).

4) On the resulting page, click on “OMNeT++ VN (source + IDE, tgz)” to download OMNeT++ version VN.

5) In the resulting pop-up window, select “OK” or “Save” to save OMNeT++ installation package, i.e. omnetpp-VN-src.tgz, in your Ubuntu Linux environment.

6) Extract the contents of omnetpp-VN-src.tgz at an install location (e.g., /opt/omnetpp), and give read, write, and execute permissions to everyone. This can be done by entering the following commands:

```
cd /opt
sudo mkdir omnetpp
cd /opt/omnetpp
sudo tar xf <PATH>
sudo chmod -R 777 .
```

where <PATH> is the path of the omnetpp-VN-src.tgz file, i.e. where it was saved by your browser. THE SPACE AND PERIOD ' .' AT THE END OF THE LAST COMMAND ARE VERY IMPORTANT. These commands will result in the creation of the directory:
“/opt/omnetpp/omnetpp-VN”.
From this point forward, we will refer to this directory as <OMNETDIR>.

7) Refer to the installation guide “InstallGuide.pdf” in <OMNETDIR>/doc directory for installation on the Ubuntu machine. In particular, follow the instructions in chapter 5 first, then chapter 4. Basically it involves the following steps:

A) Run the following commands in a terminal:

```
sudo apt-get update
sudo apt-get install build-essential gcc g++ bison flex perl tcl-dev tk-dev blt libxml2-dev zlib1g-dev
doxygen graphviz openmpi-bin libopenmpi-dev
libpcap-dev
```

B) Define the following persistent environment variables to have the values shown (see section B.1.3.1 on how to do this in the Ubuntu Linux environment):

Table B - 6: Environment variables needed for OMNeT++

VARIABLE NAME	VARIABLE VALUE
OMNETPP_ROOT	“<OMNETDIR>”
TCL_LIBRARY	“/usr/share/tcltk/tcl8.5”
PATH	“$PATH:$OMNETPP_ROOT/bin”

Recall that <OMNETDIR> was defined in step 6) of this section. Also, note that the assignment to the PATH variable appends “:$OMNETPP_ROOT/bin”
to its current value.

C) Run the following commands in a terminal:

```
cd <OMNETDIR>
```

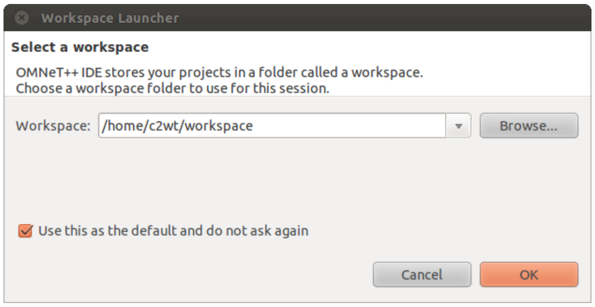
```
./configure
make
```

This last command may take some time to complete.

B.4.11.2 OMNeT++ configuration. Configuration of OMNeT++ consists of installing the Java Developer Tools, and setting some internal variables for OMNeT++.

B.4.11.2.1 Installing Java Developer Tools. OMNeT++ is built on top of the Eclipse IDE. Further, as it is implemented in C++, it uses the Eclipse CDT (C++ Development Tools). However, whereas the C2 Wind Tunnel uses C++, it also uses Java. Because of this, OMNeT++ needs to have the Eclipse JDT (Java Development Tools) installed as well. OMNeT++ configuration thus consists of installing the Eclipse JDT. It also involves defining some variables internal to OMNeT++ for the C2 Wind Tunnel.

1) First, we must execute the OMNeT++ IDE. To do this type the following in a terminal:
```
<OMNETDIR>/ide/omnetpp
```
2) In the resulting window, enter the location (i.e. directory) for OMNeT++ to use as its workspace (the default location for the workspace, i.e. the "workspace" directory in your home directory, should be fine). Also, if you don't want to see this window every time you execute OMNeT++, check the "Use this as the default and do not ask again" checkbox. Click on the "OK" button.



OMNeT++ IDE stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: /home/c2wt/workspace

☑ Use this as the default and do not ask again

Figure B - 29: Entering a folder for workspace in OMNeT++

3) When OMNeT++ is started for the first time, it will display the following window. Click on the "workbench" icon on the right side of the screen.

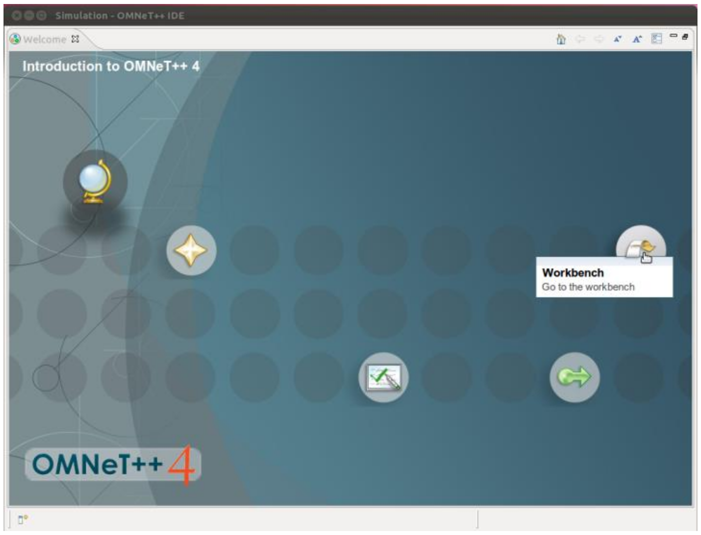


Figure B - 30: OMNeT++ start-up screen

4) The main OMNeT++ window should now be displayed (shown below).

To install the JDT, click on “Help” and select “Install new software ...”

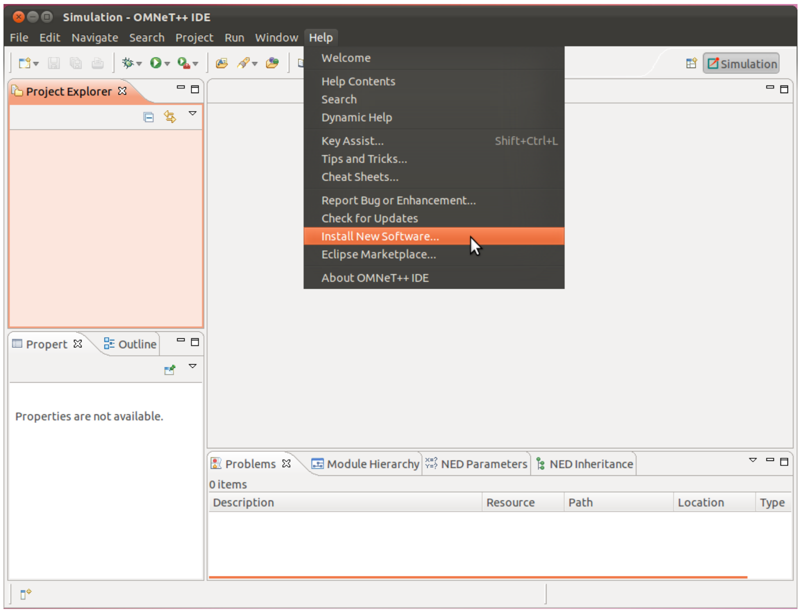
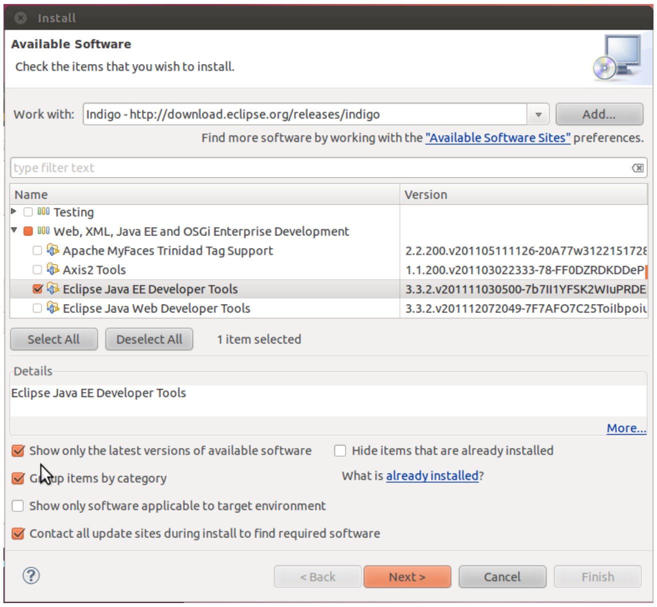


Figure B - 31: Installing new software in OMNeT++

5) In the resulting window, in text field labeled “Work with:”, enter the following text: http://download.eclipse.org/releases/indigo

In the main subwindow will appear a list of categories of add-ons that can be installed into OMNeT++:



Available Software

Check the items that you wish to install.

Work with: Indigo - http://download.eclipse.org/releases/indigo

Find more software by working with the "Available Software Sites" preferences.

Name	Version
▶ □ Testing	
▼ ■ Web, XML, Java EE and OSGi Enterprise Development	
□ Apache MyFaces Trinidad Tag Support	2.2.200.v201105111126-20A77w312215172ε
□ Axis2 Tools	1.1.200.v201103022333-78-FF0DZRDKDDeP
☑ Eclipse Java EE Developer Tools	3.3.2.v201111030500-7b7II1YFSK2WIuPRDE
□ Eclipse Java Web Developer Tools	3.3.2.v201112072049-7F7AFO7C25Toilbpoiu

1 item selected

Details

Eclipse Java EE Developer Tools

☑ Show only the latest versions of available software

□ Hide items that are already installed

☑ Group items by category

What is already installed?

□ Show only software applicable to target environment

☑ Contact all update sites during install to find required software

Figure B - 32: JDT installation settings in OMNeT++

   A) Scroll down and click on the little triangle next to "Web, XML, Java EE and OSGi Enterprise Development". A sub-menu should appear under this
   B) Scroll down and click on the checkbox next to "Eclipse Java EE Developer Tools".
   C) Click "Next".

6) In the resulting window, click "Next".
7) In the resulting window, select the "I accept the terms of the license agreement". Then click "Finish". The Java EE Developer toolkit, and all packages on which it depends, should be installed. This could take a while.
8) A window should appear indicating that OMNeT++ needs to restart for the Java EE Developer Tools installation to complete, click on "Restart Now".

B.4.11.2.2 Setting internal variables for OMNeT++. To set the internal variables necessary for the C2 Wind Tunnel in OMNeT++, do the following:

1) Execute OMNeT++ as described in steps 1) – 3) in the previous section.
2) Select Window->Preferences

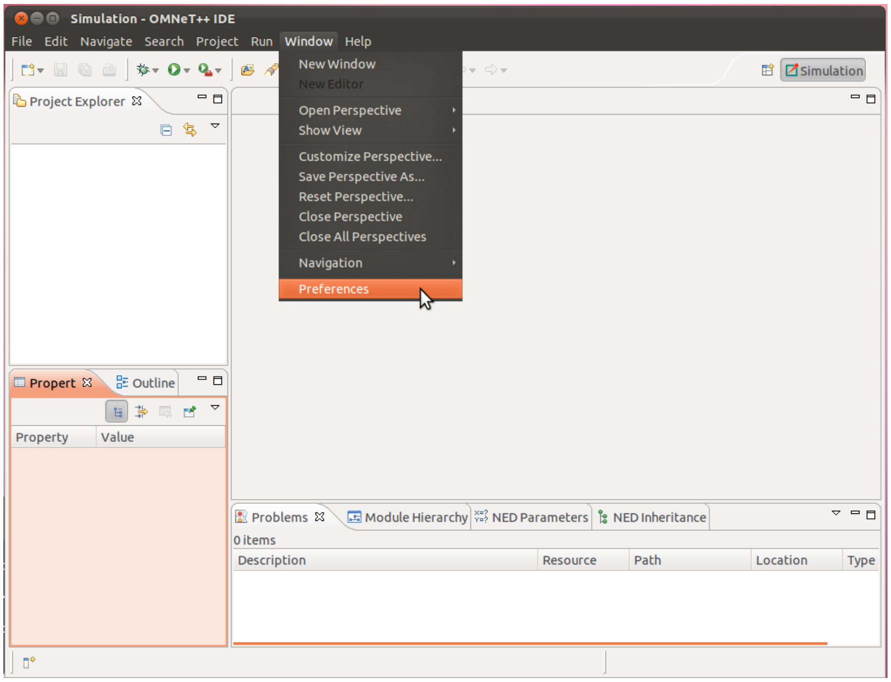
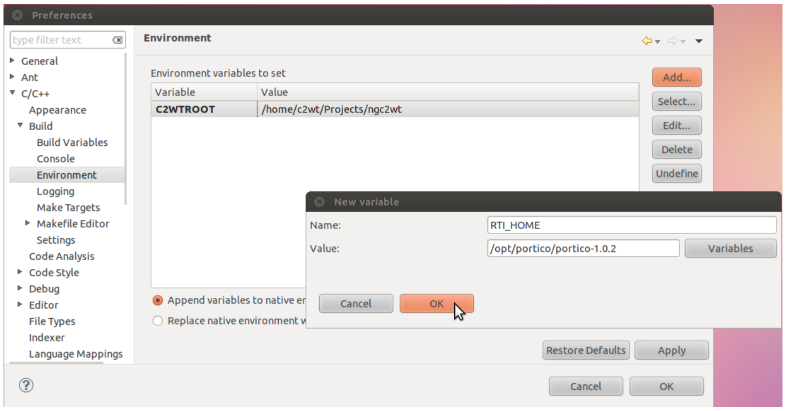


Figure B - 33: Opening preferences in OMNeT++

3) In the resulting window:
   A) Click on the small triangle next to "C++". A sub-menu will be diplayed under "C++".
   B) Click on the small triangle next to "Build", and another sub-menu will be displayed under "Build".
   C) Click on "Environment".
   D) Click on the "Add ..." button for each variable to be input. When the variable's information has been entered, click on the "OK" button.



Environment variables to set

Variable	Value
C2WTROOT	/home/c2wt/Projects/ngc2wt

Name: RTI_HOME

Value: /opt/portico/portico-1.0.2

● Append variables to native er

○ Replace native environment w

Figure B - 34: Adding C2WTROOT environment variable in OMNeT++

E) Enter the variables shown in the table below. Substitute $C2WTROOT and $RTI_HOME with their actual values as in section B.2 and section B.4.4 step 6), respectively.

Table B - 7: Environment variable values for OMNeT++ configuration

Variable Name	Variable Value
C2WTROOT	$C2WTROOT
RTI_HOME	$RTI_HOME

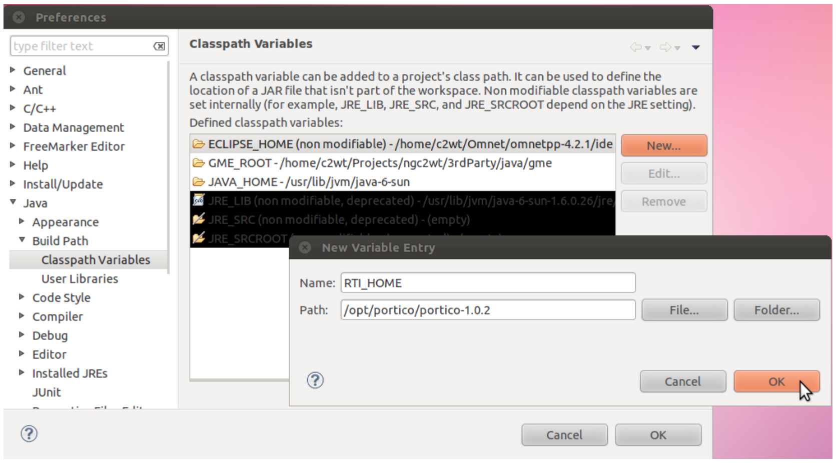
4) In the same window:

A) Click on the small triangle next to "Java". A sub-menu will be diplayed under "Java".

B) Click on the small triangle next to "Build Path", and another sub-menu will be displayed under "Build Path".

C) Click on "Classpath Variables".

D) Click on the "New ..." button for each variable to be input. When the variable's information has been entered, click on the "OK" button.



A classpath variable can be added to a project's class path. It can be used to define the location of a JAR file that isn't part of the workspace. Non modifiable classpath variables are set internally (for example, JRE_LIB, JRE_SRC, and JRE_SRCROOT depend on the JRE setting).

Defined classpath variables:

```
ECLIPSE_HOME (non modifiable) - /home/c2wt/Omnet/omnetpp-4.2.1/ide
GME_ROOT - /home/c2wt/Projects/ngc2wt/3rdParty/java/gme
JAVA_HOME - /usr/lib/jvm/java-6-sun
JRE_LIB (non modifiable, deprecated) - /usr/lib/jvm/java-6-sun-1.6.0.26/jre/
JRE_SRC (non modifiable, deprecated) - (empty)
JRE_SRCROOT
```

Name: RTI_HOME

Path: /opt/portico/portico-1.0.2

Figure B - 35: Adding RTI_HOME classpath variable in OMNeT++

E) Enter the variable shown in the table below. Substitute $C2WTROOT, $JAVA_HOME, and $RTI_HOME with their actual values as in the initial section on environment variables.

Table B - 8: Classpath variable values for OMNeT++ configuration

Variable Name	Variable Value
GME_ROOT	$C2WTROOT/3rdParty/java/gme
JAVA_HOME	$JAVA_HOME
RTI_HOME	$RTI_HOME

B.4.12 INETMANET FRAMEWORK

Inetmanet is an add-on project to OMNeT++ that allows computer network simulation within OMNeT++. Inetmanet must be both installed and configured for the C2 Wind Tunnel in the Ubuntu Linux environment.

B.4.12.1 Installing INETMANET. Following are the steps to install Inetmanet:

1) Using your browser, go to https://github.com/aarizaq/inetmanet-2.0/zipball/1898946af9578b4a48581dcc0d4643d7fb893280

2) Select "Save File" in the resulting pop-up window, and save "aarizaq-inetmanet-2.0-1898946.zip" to your Ubuntu environment.
3) Extract the contents of aarizaq-inetmanet-2.0-1898946.zip at an install location (e.g., /opt/inetmanet), and give read, write, and execute permissions to everyone. This can be done by entering the following commands:

```
cd /opt
sudo mkdir inetmanet
cd /opt/inetmanet
sudo unzip <PATH>
mv aarizaq-inetmanet-2.0-1898946 inetmanet-2.0
sudo chmod -R 777 .
```

   where <PATH> is the path of the aarizaq-inetmanet-2.0-1898946.zip file, i.e. where it was saved by your browser. THE SPACE AND PERIOD ' .' AT THE END OF THE LAST COMMAND ARE VERY IMPORTANT. These commands will result in the creation of the directory "/opt/inetmanet/inetmanet-2.0".
   From this point forward, we will refer to this directory as <INETDIR>.
4) Set a persistent environment variable INET_HOME to <INETDIR> (see section B.1.3.1 for how to set a persistent environment variable in the Ubuntu Linux environment).

B.4.12.2 Configuring INETMANET. Following are the steps to configure Inetmanet:

1) Execute OMNeT++ as in steps 1) – 3) in section B.4.6.2.1
2) Under "File" select "Import ..."

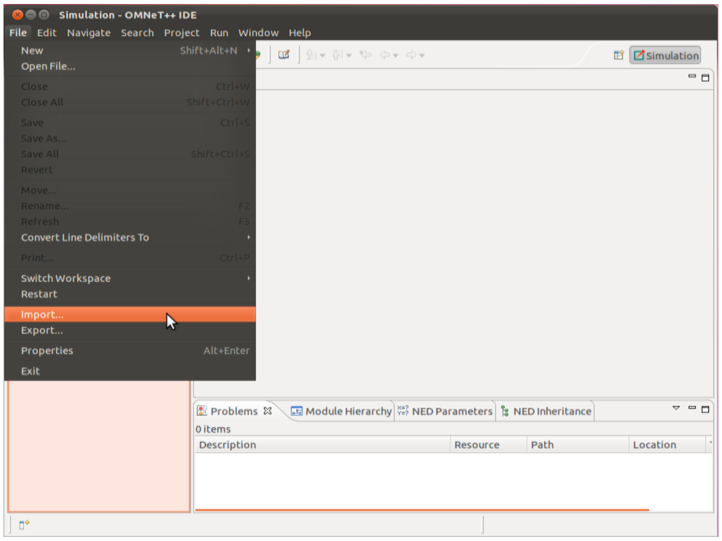


Figure B - 36: Importing Inetmanet project in OMNeT++ Eclipse IDE

3) In the resulting window, click on the small triangle next to “General”. A sub-menu should appear under “General”. Select “Existing Projects into Workspace” from this sub-menu, and click “Next”.

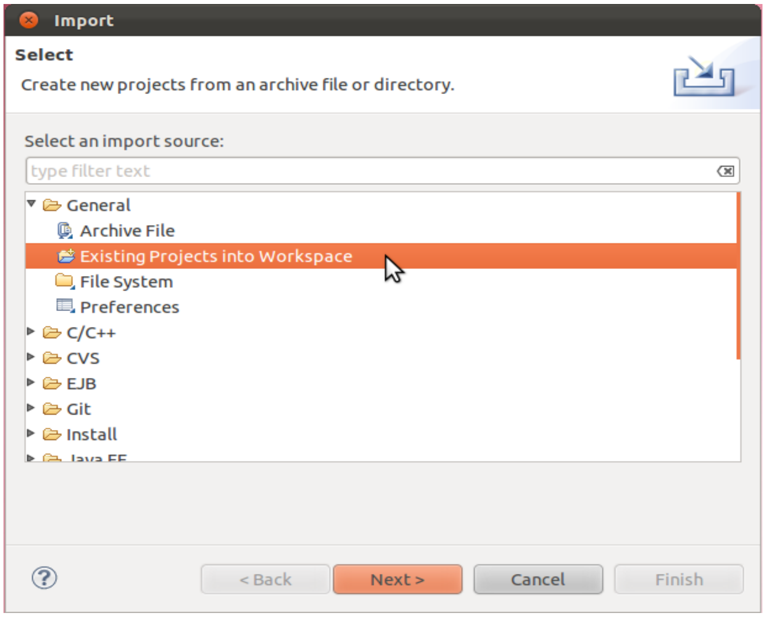
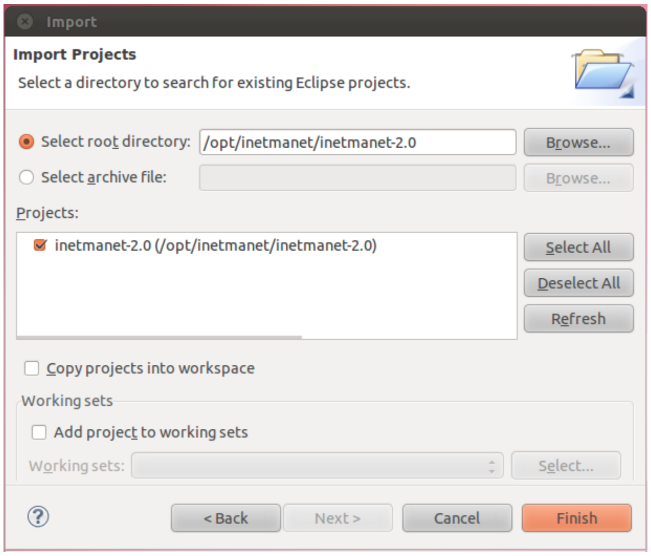


Figure B - 37: Choosing to import project from existing sources

4) In the resulting window, type <INETDIR> into the text field next to “Select root directory:”, and then click in the “Projects” subwindow. A project should appear in this subwindow. Make sure the checkbox next to the project is checked, and click “Finish”.



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: /opt/inetmanet/inetmanet-2.0

○ Select archive file:

Projects:

☑ inetmanet-2.0 (/opt/inetmanet/inetmanet-2.0)

□ Copy projects into workspace

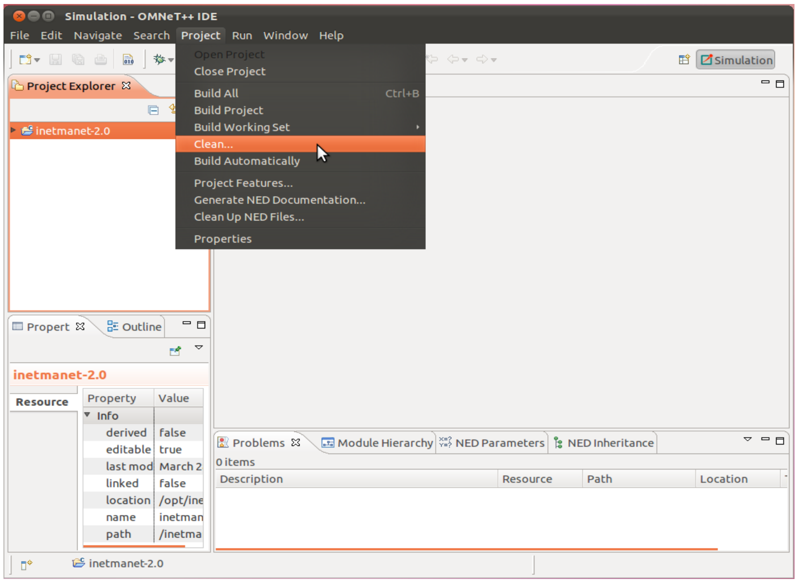
Working sets

□ Add project to working sets

Working sets:

Figure B - 38: Finalizing Inetmanet project import OMNeT++ Eclipse IDE

5) In the main OMNeT++ window, the project "inetmanet-2.0" should appear in the upper left-hand sub-window. Click on this project to select it, then click on "Project → Clean..." from the menu bar.



Simulation - OMNeT++ IDE

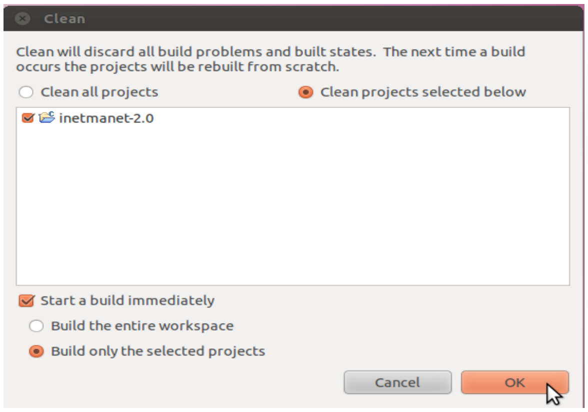
Property	Value
Info	
derived	false
editable	true
last mod	March 2
linked	false
location	/opt/ine
name	inetman
path	/inetma

Figure B - 39: Cleaning the Inetmanet project for recompilation

6) In the resulting window:

A) Select "Clean projects selected below"
B) Check the "Start a build immediately" checkbox.
C) Select "Build only the selected projects"

D) Click “OK”.



Clean

Clean will discard all build problems and built states. The next time a build occurs the projects will be rebuilt from scratch.

○ Clean all projects ● Clean projects selected below

☑ inetmanet-2.0

☑ Start a build immediately

○ Build the entire workspace

● Build only the selected projects

Figure B - 40: Choosing to rebuild the entire project after clean

The inetmanet-2.0 project will build – this may take a while. After hitting OK, you might get an error dialog about missing sources. This is just due to a bug in OMNeT++ indexer and is okay. Just press OK button to continue building.

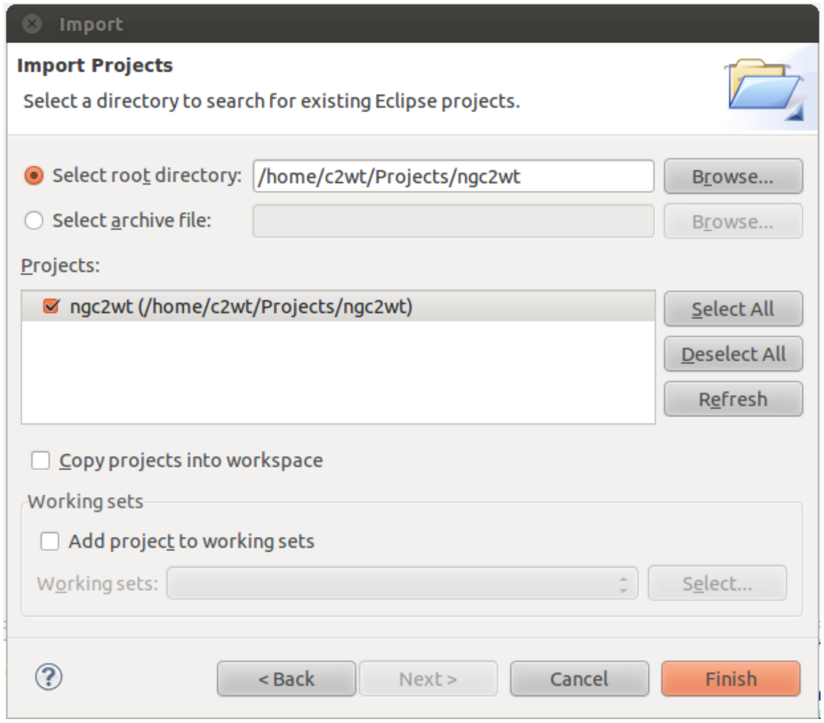
B.4.13 IMPORTING C2 WIND TUNNEL PROJECTS INTO OMNET++

The C2 Wind Tunnel software consists of both Java and C++ projects in OMNeT++. The section covers how to import and compile them in OMNeT++.

B.4.13.1 Importing the C2 Wind Tunnel Java project. The Java project in the C2 Wind Tunnel encompasses not only federate code, but also code for the GME Java-based interpreters of section B.4.2.1.3.2.

B.4.13.1.1 Importing the Java project. Following are the steps to import the Java project:

1) Execute OMNeT++ and get to the project import window as in steps 1) – 3) in section B.4.7.1.
2) In the resulting “Import Projects” window, type the $C2WTROOT directory into the text field next to “Select root directory:”, and then click in the “Projects” sub-window. A project should appear in this sub-window. Make sure the checkbox next to the project is checked, and click “Finish”.



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: /home/c2wt/Projects/ngc2wt

○ Select archive file:

Projects:

☑ ngc2wt (/home/c2wt/Projects/ngc2wt)

☐ Copy projects into workspace

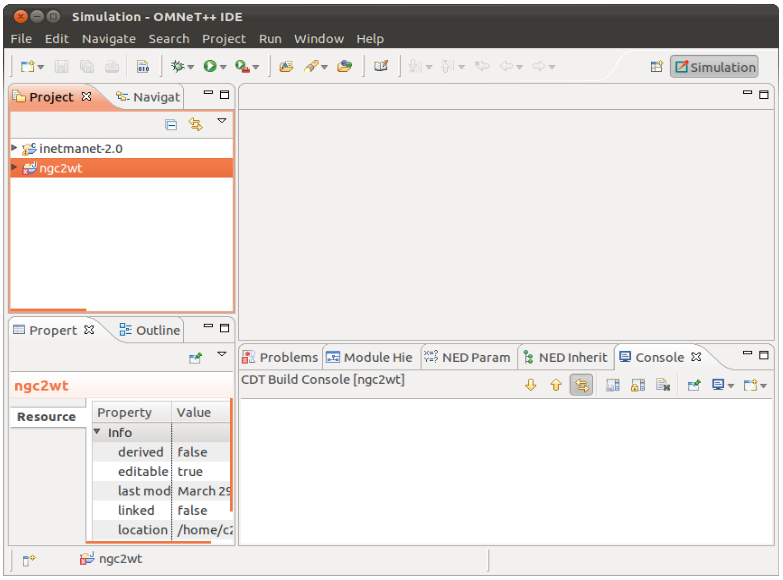
Working sets

☐ Add project to working sets

Working sets:

Figure B - 41: Importing ngc2wt project in OMNeT++ Eclipse IDE

3) The “ngc2wt” project should now appear in the main OMNeT++ window.



Property	Value
Info	
derived	false
editable	true
last mod	March 2
linked	false
location	/home/c

Figure B - 42: Imported view of the ngc2wt project in OMNeT++ Eclipse IDE

B.4.13.1.2 Building the Java project. Large portions of the C2 Wind Tunnel Java Project (ngc2wt), will not build (i.e. compile) at this time. This is because necessary code for the project has not yet been generated (via GME, see section B.5). *What will build are the Java-based interpreters that GME was configured to use back in section B.4.2.1.3.2, in particular the one that can generate this code.*

1) To build the ngc2wt project in OMNeT++, first, in a terminal run the following command. It will create missing directories whose absence prevent's the project from building.

   ```
   bash $C2WTROOT/src/c2w/scripts/generated_dirs.sh
   ```

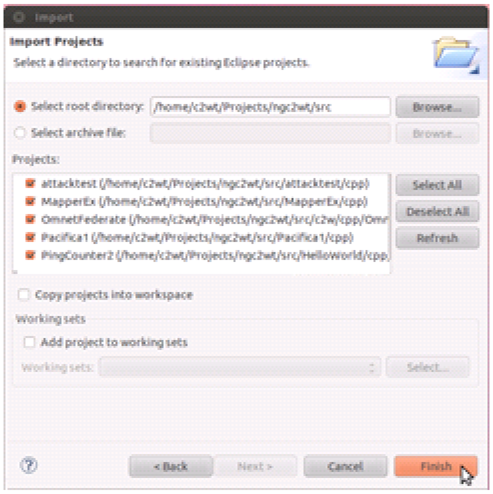
2) Now, to build the ngc2wt project, select the project in the left-hand sub-window, and follow steps 5) – 6) in section B.4.7.1 as was done for the inetmanet-2.0 project.

At this point, the java-based interpreters for the CASIM paradigm in GME should run on a CASIM model in GME.

B.4.13.2 Registering the C2 Wind Tunnel C++ projects. The C++ projects for the C2 Wind Tunnel are imported as follows. As with portions of the “ngc2wt” Java project, these projects will not build until necessary code is generated for them via GME (described in the next section).

1) Follow steps 1) – 3) in section B.4.7.1,
2) In the resulting “Import Projects” window, type the “$C2WTROOT/src” directory into the text field next to “Select root directory:”, and then click in the “Projects” sub-window. Several projects should appear in this sub-window. Make sure the checkbox next to each

project is checked, and click “Finish”.



Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: /home/c2wt/Projects/ngc2wt/src

○ Select archive file:

Projects:

- ☑ attacktest (/home/c2wt/Projects/ngc2wt/src/attacktest/cpp)
- ☑ MapperEx (/home/c2wt/Projects/ngc2wt/src/MapperEx/cpp)
- ☑ OmnetFederate (/home/c2wt/Projects/ngc2wt/src/c2w/cpp/Omr
- ☑ Pacifica1 (/home/c2wt/Projects/ngc2wt/src/Pacifica1/cpp)
- ☑ PingCounter2 (/home/c2wt/Projects/ngc2wt/src/HelloWorld/cpp

□ Copy projects into workspace

Working sets

□ Add project to working sets

Working sets:

Figure B - 43: Importing other projects in OMNeT++ Eclipse IDE

These projects will appear in the OMNeT++ main window.

B.5 GENERATING CODE FOR C2 WIND TUNNEL PROJECTS

In OMNeT++, a given project will not compile unless code has been generated for it in GME, in particular from the GME model pertaining to the project. This section does not describe how to build a model for a project. Rather, it explains how to generate code from an already existing model.

Generally, the model for any project should be stored in the file (in the Ubuntu Linux environment “$C2WTROOT/src/<PROJECT NAME>/models/gme/<PROJECT NAME>.mga”, where <PROJECT NAME> is the name of the project for which you wish to generate code. To generate code for a project, do the following. We'll use the “HelloWorld” project as as example.

1) Execute GME in your Windows environment:
   - For WINE, execute the following command in a terminal (the single quotes are very important):
     ```
     wine 'C:\Program Files\GME\Bin\GME.exe'
     ```
   - For a Windows OS, execute the “C:\Program Files\GME\Bin\GME.exe” using a command-prompt or explorer window.
2) Select “File → Open Project ...” from the top menu bar.

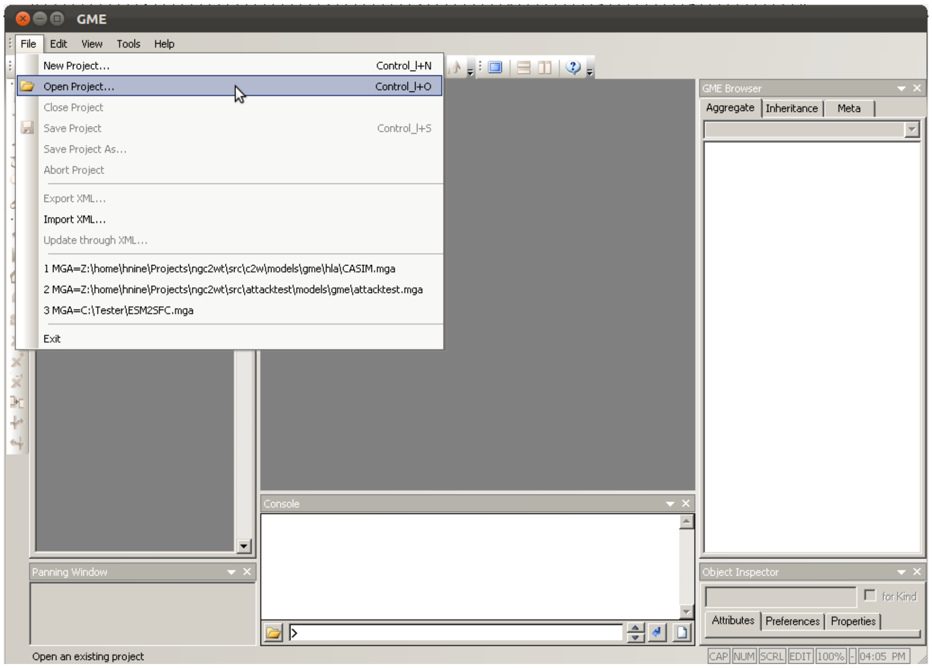
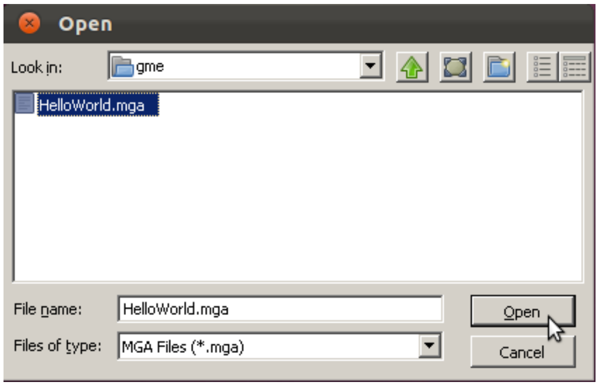


Figure B - 44: Opening a project in GME

3) In the resulting window, navigate to and open the file <PROJECT NAME>.mga.
   - In WINE, this file will be in "Z:\$C2WTROOT\src\<PROJECT NAME>\models\gme\<PROJECT NAME>.mga (see section B.2.1).
   - In a Windows OS, assuming you have set up a share as described in section B.2.2, the file will be in "Z:\src\<PROJECT NAME>\models\gme\<PROJECT NAME>.mga



Look in: gme

File name: HelloWorld.mga

Files of type: MGA Files (*.mga)

Figure B - 45: Selecting the project to open in GME

IF THIS FILE <PROJECT NAME>.mga DOES NOT CURRENTLY EXIST, it will need to be imported from the file <PROJECT NAME>.xme, which should also be in the "$C2WTROOT/src/<PROJECT NAME>/models/gme" directory:

A) Instead of “File → Open Project …”, select “File → Import XML ...” from the top menu bar:

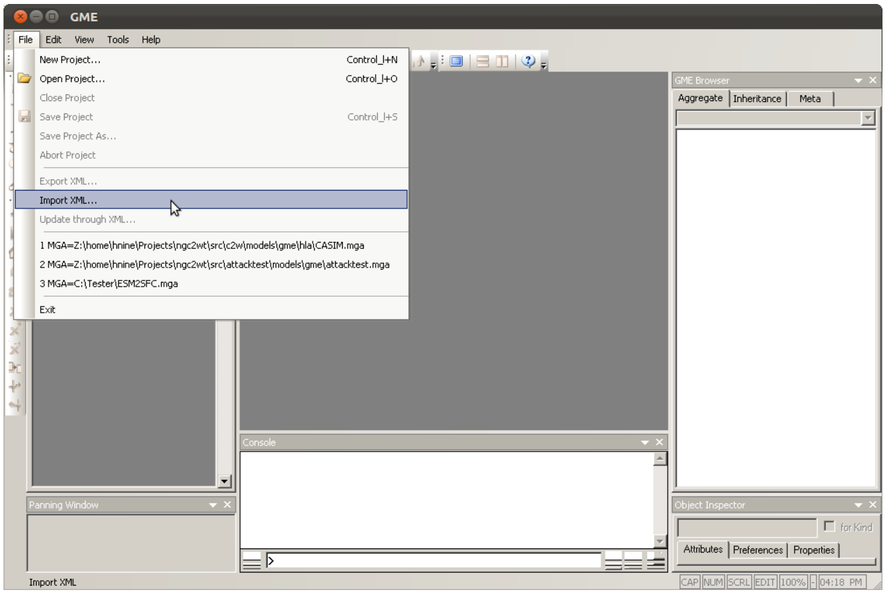
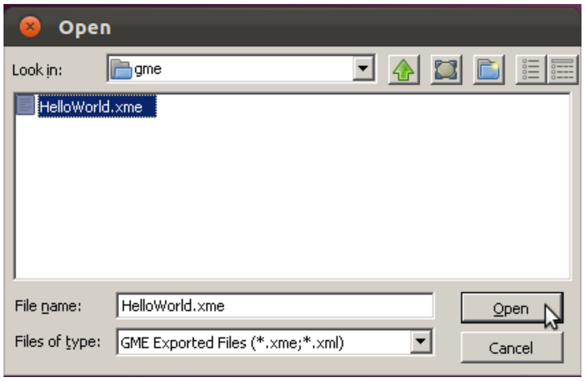


Figure B - 46: Importing an XML file for the project to open in GME

B) In the resulting window, navigate to and open the file <PROJECT NAME>.xme.

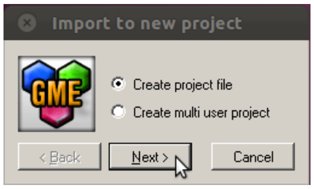


File name: HelloWorld.xme

Files of type: GME Exported Files (*.xme;*.xml)

Figure B - 47: Selecting XME file to open in GME

C) In the resulting pop-up window, select “Create project file” and click “Next”.

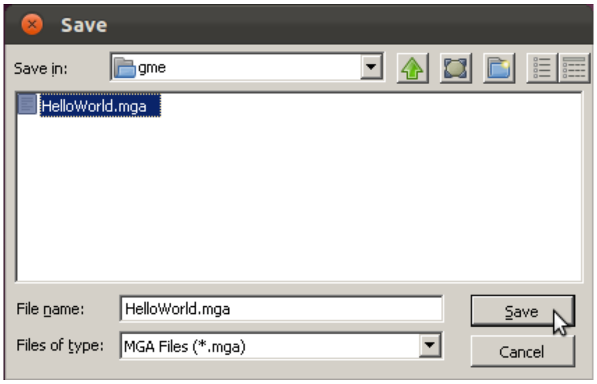


● Create project file

○ Create multi user project

Figure B - 48: Choosing to create a project file for the opened XME file in GME

D) In the resulting window, you'll be prompted to save the model as "<PROJECT NAME>.mga". Please do so by clicking "Save".



Save in: gme

File name: HelloWorld.mga

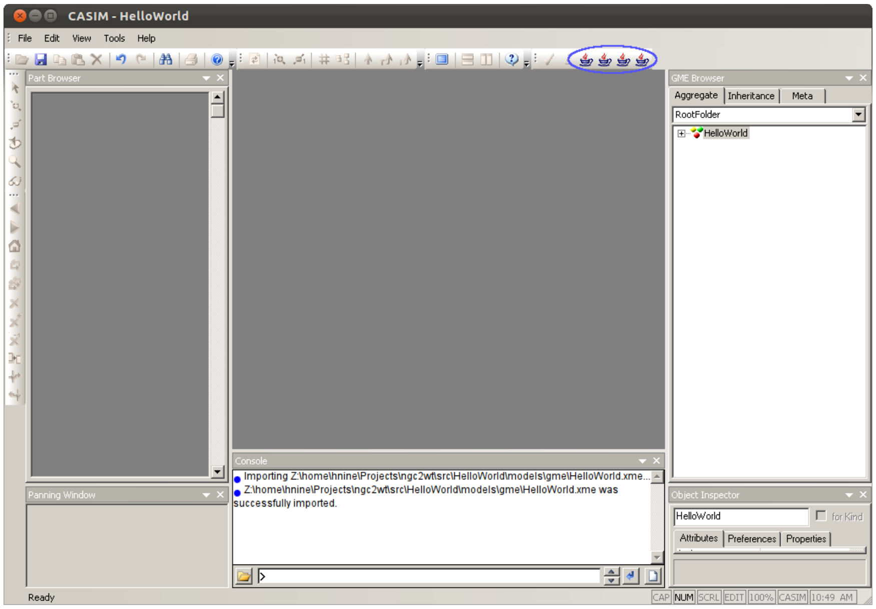
Files of type: MGA Files (*.mga)

Figure B - 49: Specifying filename to save the project file in GME

From this point onward, you can load the <PROJECT NAME>.mga file as in steps 1) – 3) above, i.e. without importing <PROJECT NAME>.xme.

4) Now that the model is loaded, and provided GME was configured as in section B.4.2.2, code can be generated for the model. This is done in two ways:
   - The C2W Main Interpreter will generate all code, (e.g. Java, C++) needed to integrate all modeling platforms used by the project into the HLA (i.e. portico).
   - The C2W Deployment Interpreter will generate shell scripts to allow you to run the Federation modeled by the GME model of the project using the HLA.

Each interpreter is executed by pressing the appropriate button at the top of the GME window. The buttons should have the Java logo (a steaming cup of coffee) on them, and there should be four of them (circled in the figure below). You can determine which is which by centering the mouse over them: a tooltip will appear indicating which interpreter will be executed by pressing the button.



```
● Importing Z:\home\hnine\Projects\ngc2wt\src\HelloWorld\models\gme\HelloWorld.xme...
● Z:\home\hnine\Projects\ngc2wt\src\HelloWorld\models\gme\HelloWorld.xme was
successfully imported.
```

Figure B - 50: Interpreter buttons when a project is opened in GME

The two of the interpreters you'll be using the most are:

- The C2W Main Interpreter (or "C2WInterpreter") will generate all code (e.g. Java, C++) needed needed in order to build the OMNeT++ projects associated with this model. When these projects build successfully, they generally produce programs that act as federates in an HLA federation simulation.
- The C2W Deployment Interpreter will generate shell scripts for the model to allow you to run this HLA federation.

NOTE: If you get an error while saving.MGA project files in GME via Wine, just export the project as an XME file and re-import the new XME file to save the updated .MGA project file.

B.6 CREATING SECONDARY MACHINES FOR THE C2 WIND TUNNEL BY CLONING THE MAIN VIRTUAL MACHINE

The C2 Wind Tunnel can use multiple machines (usually virtual machines) for a given simulation. In doing so, some federates may run on the "main" machine (i.e. one that is set up according to sections B.1 – B.4 of this Appendix), while other federates run on the secondary machines. You may wish to do this if, for instance:

- Having all of the federates run on the "main" machine overwhelms the processor of this machine or otherwise causes it to run unacceptably slowly. In this case, some federates are "off-loaded" onto secondary machines so that the processing power needed to run all of the federates is distributed amongst multiple machines.
- The simulation uses multiple CPN federates (based on the CPN Tools of section B.4.10). Due to limitations of the software used to integrate CPN models into the C2 Wind Tunnel as federates, only one CPN federate can run on a single machine during a simulation. Consequently, if the simulation uses multiple CPN models, each must run on a separate machine.

The easiest means of creating a secondary machine is, provided the main machine is a virtual machine (VM), to clone the main machine VM.

B.6.1 Preparing main VM before cloning. *It is very important that you perform the below tasks before you cloning the main VM to make secondary VM's.*

B.6.1.1 Configuring ssh. It is important that the main machine be able to log in to the secondary machines via ssh without being queried for a password. The following procedure will make this possible.

1. In a shell, execute the following command:
   ```
   ssh-keygen -t dsa
   ```
   ● Accept the default for the "file in which to save the key."
   ● Just press <ENTER> when queried for a passphrase (i.e. enter an empty passphrase).
2. In the same shell, execute the following command:
   ```
   cd ~/.ssh
   cat id_dsa.pub >> authorized_keys
   ```
3. In the same shell, execute to following command:
   ```
   ssh localhost
   ```
   When queried whether to "continue connecting", respond with "yes". You will be logged in to the main machine via an ssh session. In the resulting shell, enter the following command:
   ```
   exit
   ```

B.6.1.2 Setting up a mount of the Windows share on the main machine by the secondary machine. A secondary machine will need to access the C2WTROOT directory of the primary machine. This is done by setting up a share on the main machine that is subsequently mounted on the secondary machine.

1. Set up a share on the main machine (if you haven't already) by executing step 1) of section B.2.2 of this Appendix.
2. Obtain the IP address of the main machine as in step 2.A) of section B.2.2 of this Appendix. Then enter the following commands (:
   ```
   sudo sh -c 'echo >> /etc/fstab'
   sudo sh -c 'echo "//<IPADDRESS>/<SHARENAME>/<SPACE><C2WTROOT><SPACE>
   cifs<SPACE>users,exec,username=<USERNAME>,password=<PASSWORD>,rw<SPACE>0
   <SPACE>0" >> /etc/fstab'
   ```
   Where the meaning of the words in chevrons "<>" are shown in the table below:

Table B - 9: Table for values to be used in setting up machine mount

<SPACE>	One or more spaces
<IPADDRESS>	The IP address of the main machine
<SHARENAME>	The name you gave the share when you created it following setup 1) of section B.2.2 of this Appendix (e.g. "ngc2wt").
<C2WTROOT>	The value of the C2WTROOT environment variable, i.e. the directory that is described at the beginning of section B.2 of this Appendix (e.g. "/home/c2wt/Projects/ngc2wt").
<USERNAME>	Your user account name (e.g. "c2wt")
<PASSWORD>	Your user account password

Once you have performed these tasks, you can clone the main VM to create secondary VM's.

B.6.1.3 Disable the share on the main VM. If you created a share on the main machine, i.e. steps 1) and 2) of section B.2.2, disable the share temporarily as follows:

1. Execute nautilus as in step 1B of section B.2.2, and navigate to the C2WTROOT folder (e.g. "/home/c2wt/Projects/ngc2wt"). Then, right click on this folder and select "Sharing Options":

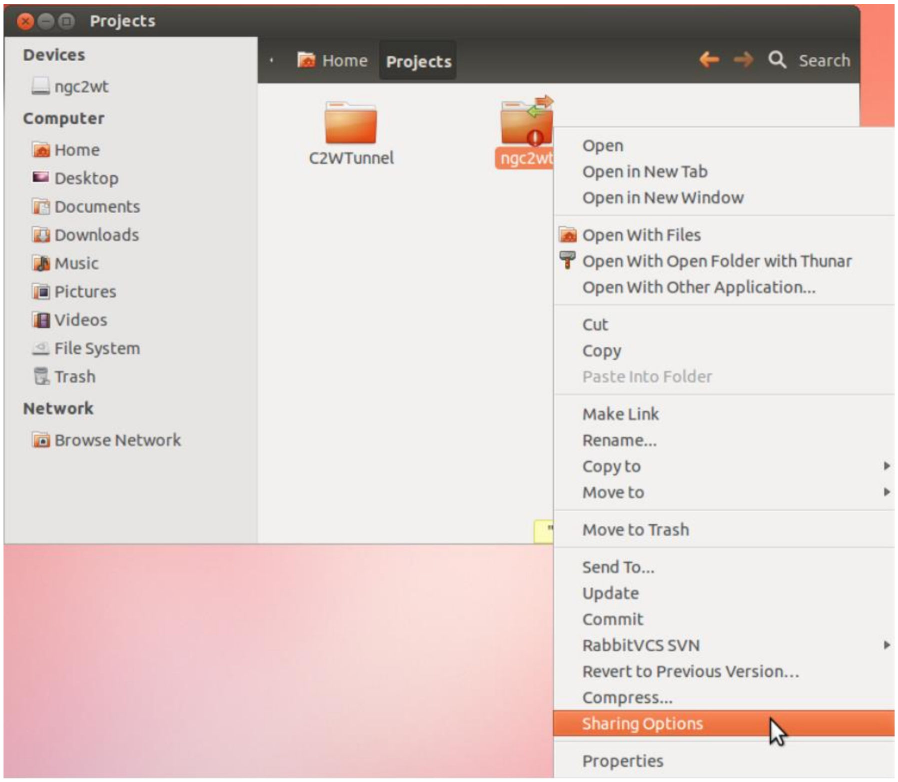
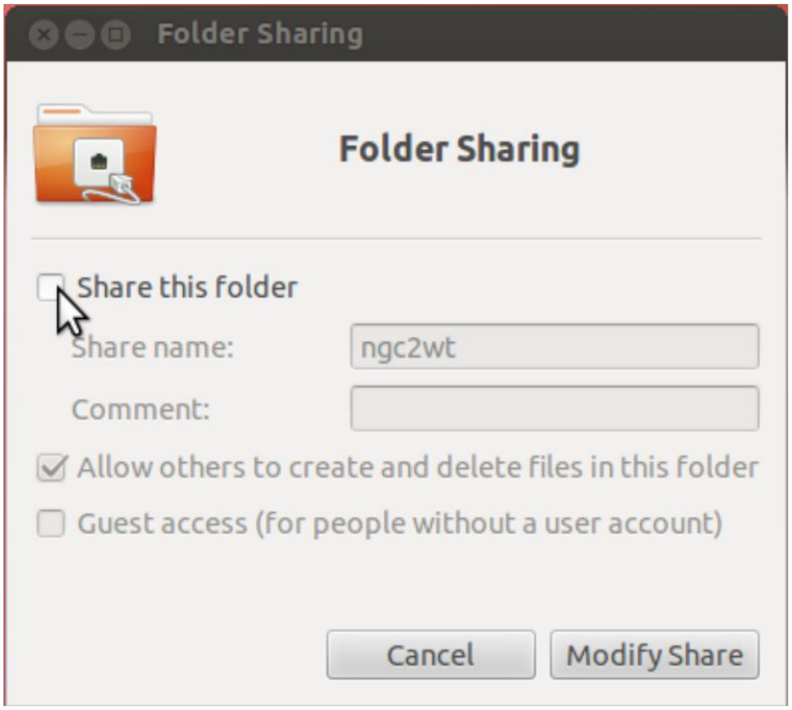


Figure B - 51: Selecting options to share a folder

2. In the resulting pop-up window, uncheck the "Share this folder" checkbox, and then click on the "Modify Share" button:



□ Share this folder

Share name: ngc2wt

Comment:

☑ Allow others to create and delete files in this folder

□ Guest access (for people without a user account)

3.

Figure B - 52: Selecting to modify share

Once you have completed these tasks, clone the main VM to create secondary VM's.

B.6.2 Cloning the VM

Once the secondary VM's have been created, **for the main machine**:

1. BOOT THE MAIN MACHINE FIRST. This should be the case whenever you are booting all of the machines that are going to be used for a simulation in the C2 Wind Tunnel.
2. Establish or (re-establish) the share on the main machine by executing step 1) in section B.2.2 of this Appendix.

Then, **for each secondary machine**:

1. After booting the machine, change the IP address of the secondary machine (please see Ubuntu documentation for how to do this).
2. Make any ssh-login from the main machine to the secondary machine be completely automatic, i.e. with no prompts. To do this, from the main machine, log in to the secondary machine using ssh. That is:
    A. In a terminal, type:
    ```
    ssh <IPADDRESS>
    ```
    where <IPADDRESS> is the IP address of the secondary machine.
    B. If the ssh command prompts to continue connecting, respond with “yes”.
    C. Once you are logged in, just enter the “exit” command and press <ENTER>.

D. Try logging in again as in A). ssh should not give another prompt like that in B) when logging in from the main machine to this secondary machine. Again, once you are logged in, just enter “exit”.

This completes the process of creating secondary machines.

APPENDIX C – LIST OF SOFTWARE USED IN CASIM

In CASIM, the main development work is done on a Ubuntu machine (or a VM). All of the sources exist here on the main/development machine. The version we use currently is **Ubuntu 12.04 LTS (precise) (32-bit OS)**.

We use GME for modeling and code generation. GME is a Windows application. We use and support GME under Windows XP and Windows 7. Also, we use GME inside the Ubuntu Virtual Machine using **wine** Windows Emulator. In general, user should not need a Windows machine for GME use. As long the user can run the supplied Ubuntu Virtual Machine, which contains all necessary software, including GME, pre-installed and pre-configured, it should be sufficient.

Windows machine can be used to natively compile a couple of C++ GME components (HLADecorator and HLAHelper), the sources for which are supplied with CASIM. However, for user's convenience, pre-compiled binaries are also supplied with CASIM, so that user can work completely in the Ubuntu Virtual Machine. If there are changes user wants to make or want to re-compile these GME components for their Windows, the following software is needed:
1) **GME 12.3.13**
2) **UDM 3.2.10** (needed only to convert UDM based script XML file to GME model)
3) **JDK 1.6.0_37**
4) **MS Visual Studio 2010**

Below are the tools that must be installed on the Ubuntu machine to develop and execute CASIM scenarios:
1) **JDK 1.6.0_37**
2) **Java 3D 1.5.2**
3) **Boost 1.46.0**
4) **Portico 1.0.2**
5) **CPN Tools 2.2.0**
6) **OMNeT++ 4.2.1**
7) **MySQL Server 5.5.28**
8) **MySQL C++ Connector 1.1.0**
9) **INETMANET aarizaq-inetmanet-2.0-1898946** (Commit ID: 1898946af9578b4a48581dcc0d4643d7fb893280 on github for aarizaq/inetmanet-2.0 repository)

Apart from above, there are several other third-party libraries that come with the CASIM sources when you download the sources from its SVN. These are:
1) **BRITNeY 2.2.0**
2) **GetOpt 1.0.13**
3) **JGraphX 1.7.1.10**
4) **MySQL Java Connector 5.1.14**
5) **StringTemplate 3.2** (which uses **ANTLR 3.1.3**)
6) **XOM 1.2.6**

APPENDIX D – CASIM KEY FEATURES

This Appendix provides a list of new features added and improvements made to the C2 Wind Tunnel technology during the project. The Appendix also lists the range of cyber effects that have been implemented thus far.

D.1 GENERIC OMNET++ FOM

A FOM for the network simulation OMNET++ federate that is defined generically in one place as a static library that can be re-used in multiple scenarios without modification/customization. It describes the generic interactions the OMNeT++ federate works with. The figure below shows the FOM model:

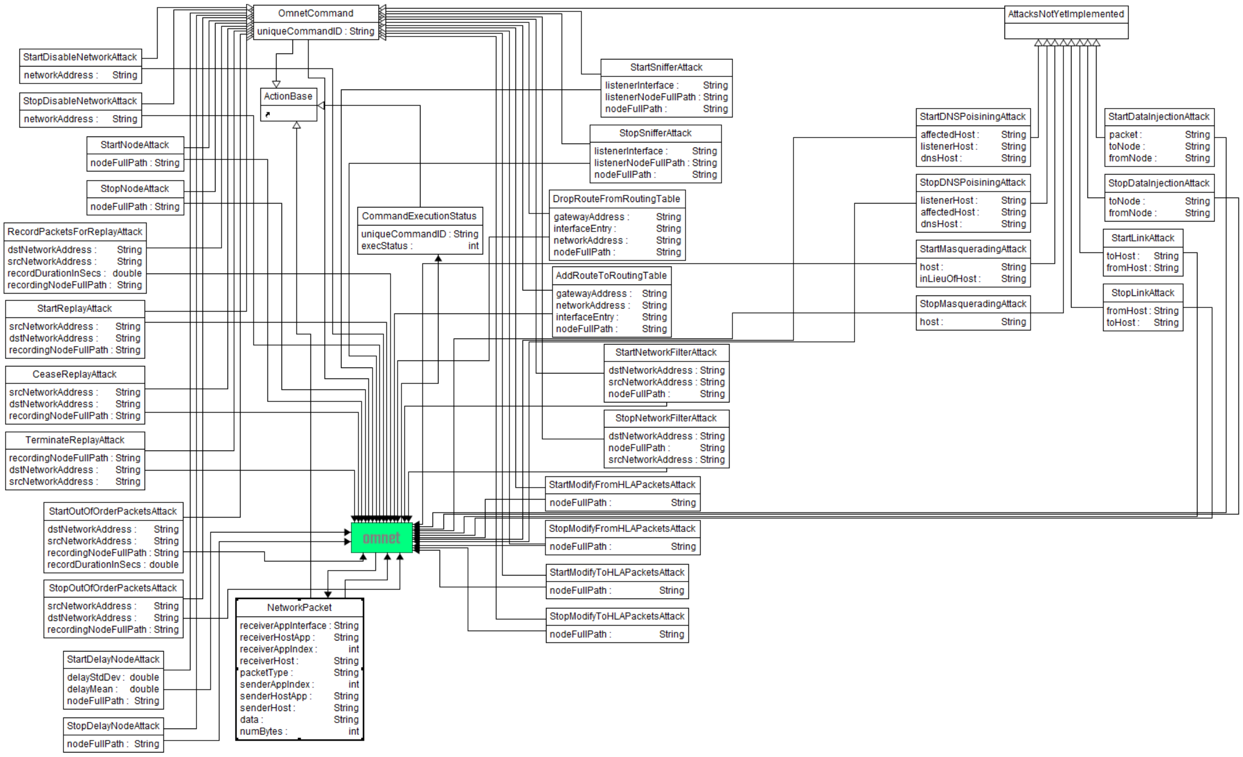
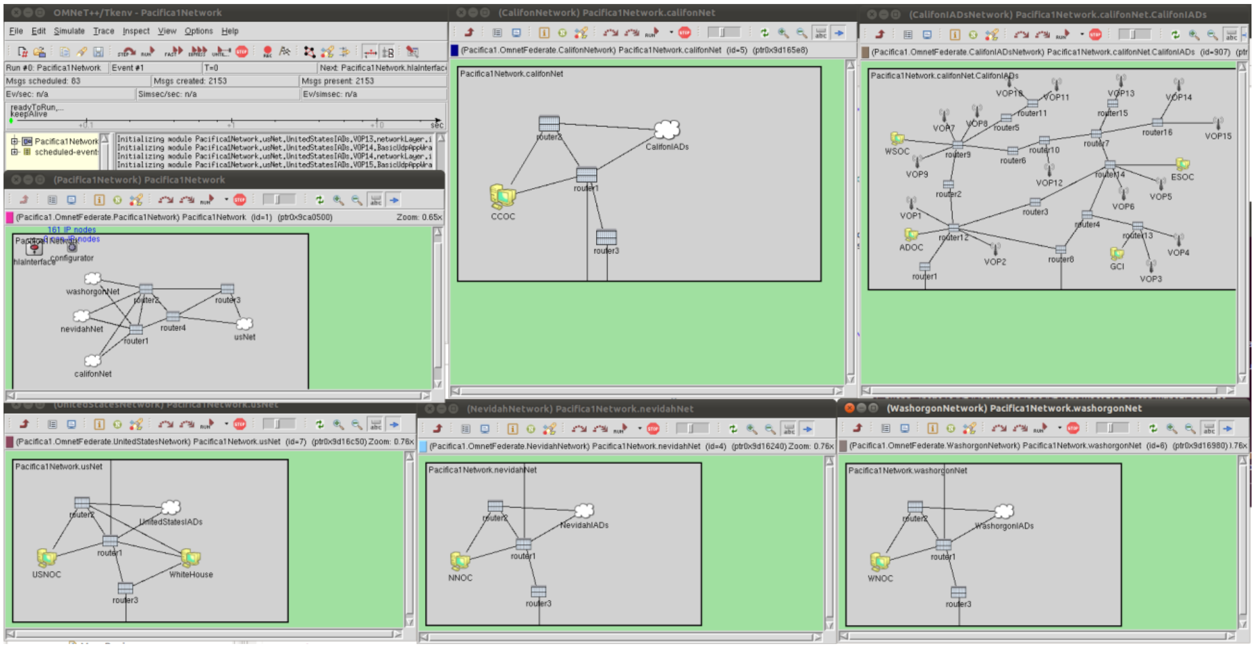


Figure D - 1: OMNeT++ FOM

D.2 GENERIC OMNET++ SIMULATOR:

It supplies all C2 Wind Tunnel integration and cyber effects facilities for integrating scenario-specific network models. The integration hooks enable the network model to run as a C2 Wind Tunnel federate with time synchronization. The cyber effects layers provide OMNeT++ module implementations for a variety of cyber effects as described below. The generic simulator is supplied as a static library that can be easily incorporated in various scenarios. The figure below shows OMNeT++ simulation for an example scenario:



```
Run #0: Pacifica1Network   Event #1   T=0   Next: Pacifica1Network.hlaInterfac
Msgs scheduled: 83   Msgs created: 2153   Msgs present: 2153
Ev/sec: n/a   Simsec/sec: n/a   Ev/simsec: n/a
Initializing module Pacifica1Network.usNet.UnitedStatesIADs.VOP13.networkLayer.i
Initializing module Pacifica1Network.usNet.UnitedStatesIADs.VOP14.BasicUdpAppWra
Initializing module Pacifica1Network.usNet.UnitedStatesIADs.VOP14.networkLayer.i
Initializing module Pacifica1Network.usNet.UnitedStatesIADs.VOP15.BasicUdpAppWra
```

OMNeT++/Tkenv - Pacifica1Network
File Edit Simulate Trace Inspect View Options Help
readyToRun,...
keepAlive
Pacifica1Network
scheduled-events
(Pacifica1Network) Pacifica1Network
(Pacifica1.OmnetFederate.Pacifica1Network) Pacifica1Network (id=1) (ptr0x9ca0500) Zoom: 0.65x
161 IP nodes
0 non-IP nodes
hlaInterface
configurator
washorgonNet
nevidahNet
califonNet
usNet
router1
router2
router3
router4
(CalifonNetwork) Pacifica1Network.califonNet
(Pacifica1.OmnetFederate.CalifonNetwork) Pacifica1Network.califonNet (id=5) (ptr0x9d165e8)
Pacifica1Network.califonNet
CalifonIADs
CCOC
(CalifonIADsNetwork) Pacifica1Network.califonNet.CalifonIADs
(Pacifica1.OmnetFederate.CalifonIADsNetwork) Pacifica1Network.califonNet.CalifonIADs (id=907)
Pacifica1Network.califonNet.CalifonIADs
WSOC
ESOC
ADOC
GCI
VOP1
VOP2
VOP3
VOP4
VOP5
VOP6
VOP7
VOP8
VOP9
VOP10
VOP11
VOP12
VOP13
VOP14
VOP15
router5
router6
router7
router8
router9
router10
router11
router12
router13
router14
router15
router16
(UnitedStatesNetwork) Pacifica1Network.usNet
(Pacifica1.OmnetFederate.UnitedStatesNetwork) Pacifica1Network.usNet (id=7) (ptr0x9d16c50) Zoom: 0.76x
Pacifica1Network.usNet
UnitedStatesIADs
USNOC
WhiteHouse
(NevidahNetwork) Pacifica1Network.nevidahNet
(Pacifica1.OmnetFederate.NevidahNetwork) Pacifica1Network.nevidahNet (id=4) (ptr0x9d16240) Zoom: 0.76x
Pacifica1Network.nevidahNet
NevidahIADs
NNOC
(WashorgonNetwork) Pacifica1Network.washorgonNet
(Pacifica1.OmnetFederate.WashorgonNetwork) Pacifica1Network.washorgonNet (id=6) (ptr0x9d16980) 1.76x
Pacifica1Network.washorgonNet
WashorgonIADs
WNOC

Figure D - 2: Generic OMNeT++ Simulator

D.3 FLEXIBLE, DETAILED, AND EFFICIENT CYBER EFFECTS LIBRARY

CASIM comes with a library of cyber effects in the generic OMNeT++ simulator. This library has been written such that it can be easily incorporated in any scenario/demonstration. It is fairly detailed and uses a number of parameters to configure the effects as needed in a particular scenario. Also, this has been implemented with efficient code so as to not introduce a significant overhead for the network simulator. The efficiency and modularity are the key strengths of this library. Before this library, the cyber effects were very rudimentary and needed significant effort to make them work for a new scenario and also those were highly computation intensive so much that the entire federation simulation was slowed down manifolds during the period of cyber effects. With the new design in the Cyber Effects Library those problems have been eliminated.

The library comes as part of the generic OMNeT++ simulator (a static library). Using this library, users can use scenario-specific network models and simulated networked applications without needing to change the cyber effects implementation. As the cyber effects are based on the interaction models defined in OMNeT++ FOM, the cyber effects can be *exercised* and *controlled* (turned on/off, varied) at run-time using appropriate scenario sequence models (COAs).

Please refer to the Appendix F on ‘CyberEffects’ for a detailed description of the cyber effects that are part of the generic library that comes with CASIM.

D.4 FOM MAPPING LANGUAGE AND EXAMPLE MAPPER FEDERATES

CASIM intends to make use of C2 Wind Tunnel much simpler and flexible. It also intends to support external HLA-based simulation tools that have pre-defined FOMs. In order to support federates that have FOM pre-defined, it is necessary to device mechanisms that can translate messages defined in one FOM to the other. If there were such a facility, then different federates with their fixed FOMs can be easily integrated and made to work together. However, this is a non-trivial task and is an active research area in both industries and academia.

In CASIM, we created a FOM mapping language that can be used to map messages defined in one FOM to messages defined in other FOMs. Using this language, users can specify in the GME models how the message must be translated in other FOM such that federates using that FOM can interpret the message correctly. In addition to simple relations that involve parameter mapping and conversions, we also support custom Java code that can be embedded in the models to support highly involved mappings.

We have created several Mapper federate examples that demonstrate the successful use of the FOM mapping technology. The Mapper code is completely automatically generated.

Please refer to the Appendix G on ‘Mapping Technology’ for a detailed discussion of mapping capabilities in CASIM.

D.5 COA SEQUENCE EXECUTION LANGUAGE

In CASIM, we designed a language for describing Coarse-Of-Action (COA) to be used in various scenarios. For example, COAs are commonly used in creating war-gaming scenarios. The figure below shows an example COA model:

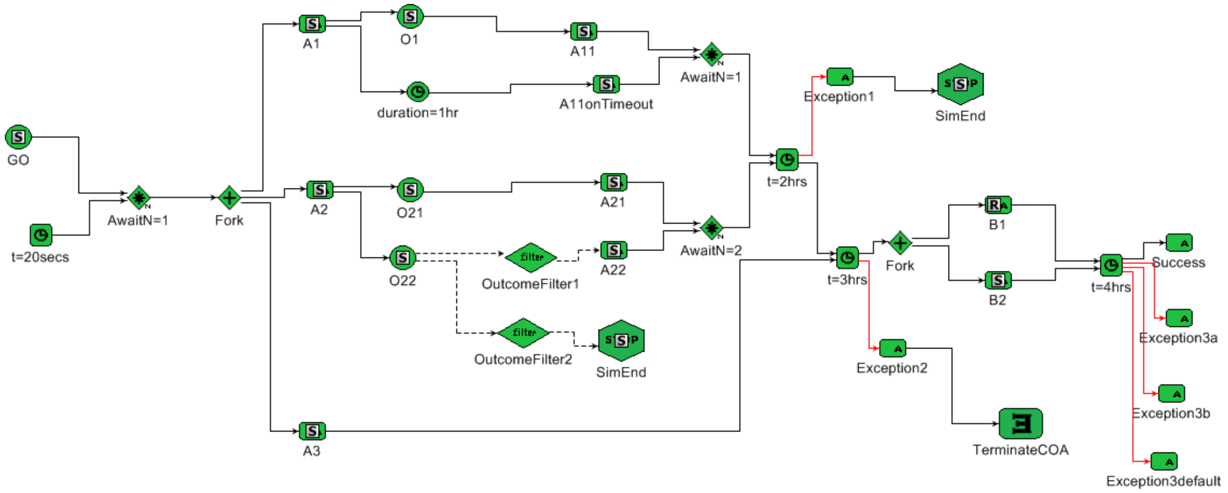


Figure D - 3: COA sequence model example

As can be seen in the above figure, the COA language supports a number of the COA sequence elements that are used to describe a COA. Here are the sequence elements that are in the COA Sequence Execution Language:

Table D - 1: Elements of the sequence models

COA Sequence Element	ICON	Description
Synchronization Point (and Exceptions/timeouts)	🕒	This represents the absolute time point from the beginning of the simulation. The semantics is that all incoming branches must wait until the time-point represented by the synchronization point has been reached. If all incoming branches have been finished, the success following branch is taken. Also, depending on the number of incoming branches that have finished till the synchronization point, the exception branches can be taken according to the model. For example, in the figure at synchronization point (t=2hrs), if both the branches have succeeded, then the branch with synchronization point (t=3hrs) is activated, otherwise Exception1 branch is triggered.
Action	A	An action is basically an interaction type of an interaciton that must be sent out by the COA Sequence Executor as soon as the action point is reached in COA sequence execution. The parameters of the interactions can be specified.
Outcome	🟢	An outcome basically represents the type of an interaction that the COA sequence executor must wait for to arrive before it can proceed. For example, this can represent a planned or expected activity in a scenario that when occurs, an interaction of this type is sent to HLA.
Fork	✚	A Fork element is a branching element. Its semantics are that all branches following a fork element are executed in parallel as soon as the fork element is reached.
Probabilistic Choice	🎲 ◈	A Probabilistic Choice element chooses only a single succeeding branch. Different branches have a probability specified for their selection. The probabilities of all branches are normalized to 1 if they already do not add to 1. At run-time a random value between 0 and 1 is chosen and depending on that value the appropriate branch is selected for execution.
AwaitN	✱ N	A simple specifier to must wait on a given number of incoming branches to finish, before letting the COA sequence execution to proceed.

Duration	🕒	A duration specifier represents the time the COA sequence executor delays the execution once the duration element is reached. This time-period is relative to the time when the duration element is reached.
Random Duration	🎲 🕒	A duration specifier represents a duration that is randomly distributed using a uniform distribution. For uniform distribution a range is provided and the orchestrator automatically chooses a value according to a seed value. The seed value can be configured for an experiment for repeatability of experiment purpose. If no seed is provided, a default 0 is used as the seed value. Once a value for the random duration is selected, the execution semantics is the same as a regular duration.
OutcomeFilter	filter	This is an advanced concept and can be used to filter based on the values of the parameters of the received interaction (as an Outcome). Different outgoing branches can be executed based on different values of parameters.
Terminate COA	Ǝ	When reached the COA execution is terminated.
SimEnd	STOP	When reached the entire simulation/federation is terminated.

D.6 COA SEQUENCE EXECUTOR (ORCHESTRATOR):

Once a number of COAs are modeled in a scenario, depending on the gaming requirements (if any), we need a COA sequence executor that can parse the COA models and execute them while matching the intended semantics. We extended the Federation Manager Java federate in C2 Wind Tunnel to handle this. The COA sequence executor parses all of the COA models designed and builds a graph data structure from it with COA sequence elements as nodes of the graph and appropriate dependencies between COA sequence elements as edges of the graph. It then executes the graph structures step-by-step while matching the intended semantics.

The COA models are first translated in the form of an XML file that can also be manually changed (or if needed created using a gaming script for advanced applications and variability). This XML file is then parsed to create the graph data structures – which are then executed by the COA sequence executor.

Multiple COA models can be executed in parallel.

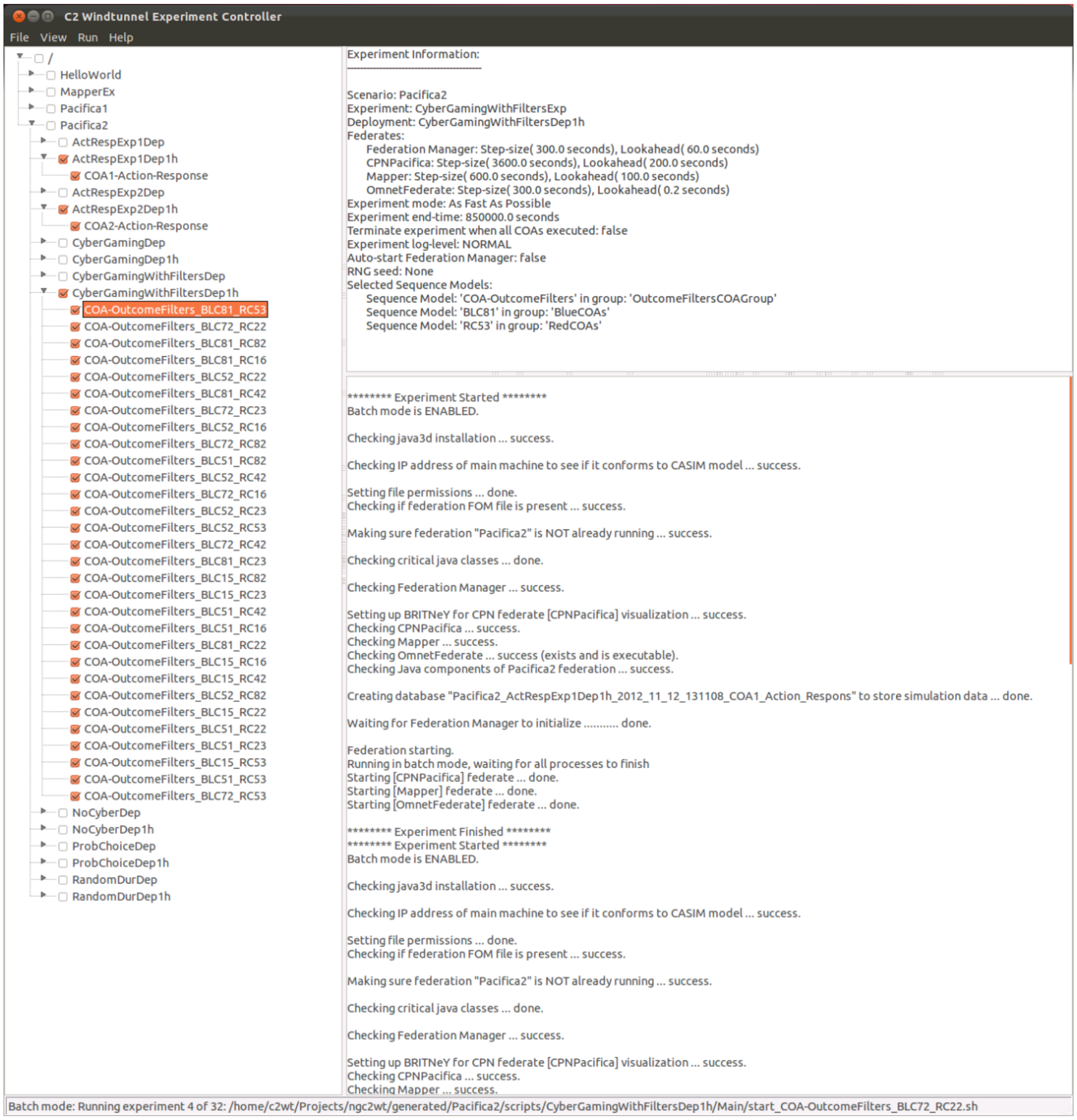
Note: As Outcome represents the arrival of a specified type of interaction, if multiple COA models are waiting on the same interaction type at the same time, all of them will be activated when such an interaction is received by the federation manager.

D.7 COA SIMULATION DISPLAY:

The COA Sequence Executor informs about the nodes of then COA models as it executes them in the form of console log messages. However, we wanted to create a graphical display that replicates the COA model visually and displays the execution of the COA model visually at run-time as the COA models are executed. Therefore, we extended the federation manager to also include a COA simulation display GUI (using an open-source JGraph library), to automatically generate a graphical GUI from the COA data structures and then make it stateful using icons for the three different states of COA nodes, viz. INACTIVE (not yet reached in COA execution), ACTIVE (currently being executed), and EXECUTED (already finished executing). The result is that the COA simulation can be visually seen and intuitively shows what is happening in the simulation and why!

D.8 C2WT EXPERIMENT CONTROLLER:

One of the most requested requirements among users of C2 Wind Tunnel is to create a front-end graphical user-interface that is easy and intuitive to use. The C2 Wind Tunnel Experiment Controller, written as a python application serves this purpose. It can be used to automatically detect the available experiments for various scenarios users have created and generated and compiled code for. It can be used to execute any set of the selected experiments. Apart from this basic functionality, the C2 Wind Tunnel Experiment Controller supports a number of user-friendly features that together provide a very useful tool for experimentation with C2 Wind Tunnel.



```
□ /
  □ HelloWorld
  □ MapperEx
  □ Pacifica1
  □ Pacifica2
    □ ActRespExp1Dep
    ☑ ActRespExp1Dep1h
      ☑ COA1-Action-Response
    □ ActRespExp2Dep
    ☑ ActRespExp2Dep1h
      ☑ COA2-Action-Response
    □ CyberGamingDep
    □ CyberGamingDep1h
    □ CyberGamingWithFiltersDep
    ☑ CyberGamingWithFiltersDep1h
      ☑ COA-OutcomeFilters_BLC81_RC53
      ☑ COA-OutcomeFilters_BLC72_RC22
      ☑ COA-OutcomeFilters_BLC81_RC82
      ☑ COA-OutcomeFilters_BLC81_RC16
      ☑ COA-OutcomeFilters_BLC52_RC22
      ☑ COA-OutcomeFilters_BLC81_RC42
      ☑ COA-OutcomeFilters_BLC72_RC23
      ☑ COA-OutcomeFilters_BLC52_RC16
      ☑ COA-OutcomeFilters_BLC72_RC82
      ☑ COA-OutcomeFilters_BLC51_RC82
      ☑ COA-OutcomeFilters_BLC52_RC42
      ☑ COA-OutcomeFilters_BLC72_RC16
      ☑ COA-OutcomeFilters_BLC52_RC23
      ☑ COA-OutcomeFilters_BLC52_RC53
      ☑ COA-OutcomeFilters_BLC72_RC42
      ☑ COA-OutcomeFilters_BLC81_RC23
      ☑ COA-OutcomeFilters_BLC15_RC82
      ☑ COA-OutcomeFilters_BLC15_RC23
      ☑ COA-OutcomeFilters_BLC51_RC42
      ☑ COA-OutcomeFilters_BLC51_RC16
      ☑ COA-OutcomeFilters_BLC81_RC22
      ☑ COA-OutcomeFilters_BLC15_RC16
      ☑ COA-OutcomeFilters_BLC15_RC42
      ☑ COA-OutcomeFilters_BLC52_RC82
      ☑ COA-OutcomeFilters_BLC15_RC22
      ☑ COA-OutcomeFilters_BLC51_RC22
      ☑ COA-OutcomeFilters_BLC51_RC23
      ☑ COA-OutcomeFilters_BLC15_RC53
      ☑ COA-OutcomeFilters_BLC51_RC53
      ☑ COA-OutcomeFilters_BLC72_RC53
    □ NoCyberDep
    □ NoCyberDep1h
    □ ProbChoiceDep
    □ ProbChoiceDep1h
    □ RandomDurDep
    □ RandomDurDep1h
```

```
Experiment Information:
----------------------------------------

Scenario: Pacifica2
Experiment: CyberGamingWithFiltersExp
Deployment: CyberGamingWithFiltersDep1h
Federates:
    Federation Manager: Step-size( 300.0 seconds), Lookahead( 60.0 seconds)
    CPNPacifica: Step-size( 3600.0 seconds), Lookahead( 200.0 seconds)
    Mapper: Step-size( 600.0 seconds), Lookahead( 100.0 seconds)
    OmnetFederate: Step-size( 300.0 seconds), Lookahead( 0.2 seconds)
Experiment mode: As Fast As Possible
Experiment end-time: 850000.0 seconds
Terminate experiment when all COAs executed: false
Experiment log-level: NORMAL
Auto-start Federation Manager: false
RNG seed: None
Selected Sequence Models:
    Sequence Model: 'COA-OutcomeFilters' in group: 'OutcomeFiltersCOAGroup'
    Sequence Model: 'BLC81' in group: 'BlueCOAs'
    Sequence Model: 'RC53' in group: 'RedCOAs'
```

```
******** Experiment Started ********
Batch mode is ENABLED.

Checking java3d installation ... success.

Checking IP address of main machine to see if it conforms to CASIM model ... success.

Setting file permissions ... done.
Checking if federation FOM file is present ... success.

Making sure federation "Pacifica2" is NOT already running ... success.

Checking critical java classes ... done.

Checking Federation Manager ... success.

Setting up BRITNeY for CPN federate [CPNPacifica] visualization ... success.
Checking CPNPacifica ... success.
Checking Mapper ... success.
Checking OmnetFederate ... success (exists and is executable).
Checking Java components of Pacifica2 federation ... success.

Creating database "Pacifica2_ActRespExp1Dep1h_2012_11_12_131108_COA1_Action_Respons" to store simulation data ... done.

Waiting for Federation Manager to initialize ........... done.

Federation starting.
Running in batch mode, waiting for all processes to finish
Starting [CPNPacifica] federate ... done.
Starting [Mapper] federate ... done.
Starting [OmnetFederate] federate ... done.

******** Experiment Finished ********
******** Experiment Started ********
Batch mode is ENABLED.

Checking java3d installation ... success.

Checking IP address of main machine to see if it conforms to CASIM model ... success.

Setting file permissions ... done.
Checking if federation FOM file is present ... success.

Making sure federation "Pacifica2" is NOT already running ... success.

Checking critical java classes ... done.

Checking Federation Manager ... success.

Setting up BRITNeY for CPN federate [CPNPacifica] visualization ... success.
Checking CPNPacifica ... success.
Checking Mapper ... success.
```

```
Batch mode: Running experiment 4 of 32: /home/c2wt/Projects/ngc2wt/generated/Pacifica2/scripts/CyberGamingWithFiltersDep1h/Main/start_COA-OutcomeFilters_BLC72_RC22.sh
```

Figure D - 4: Experiment Controller

Please see the Appendix I on ‘C2WT Experiment Controller’ for a detailed user’s manual of this tool.

D.9 BRITNEY CPN MODEL VISUALIZATION:

BRITNeY Java library is used in C2 Wind Tunnel to connect to the CPN models from Java and execute and control their simulation as a C2WT federate. This new feature uses the visualization capabilities of the BRITNeY Java library. The default library needed to be extended in-house to support this functionality. With the visualization support, the CPN federate shows the complete state of each of its places and transitions as the simulation is proceeding. It shows what tokens are generated at which places and how they move from place to place. It also shows the hooks of the CPN model with HLA (input and output places) and what input tokens come in from HLA and what goes back to HLA. This is a major addition to the C2 Wind Tunnel infrastructure. A picture of an example scenario with BRITNeY visualization is shown below:

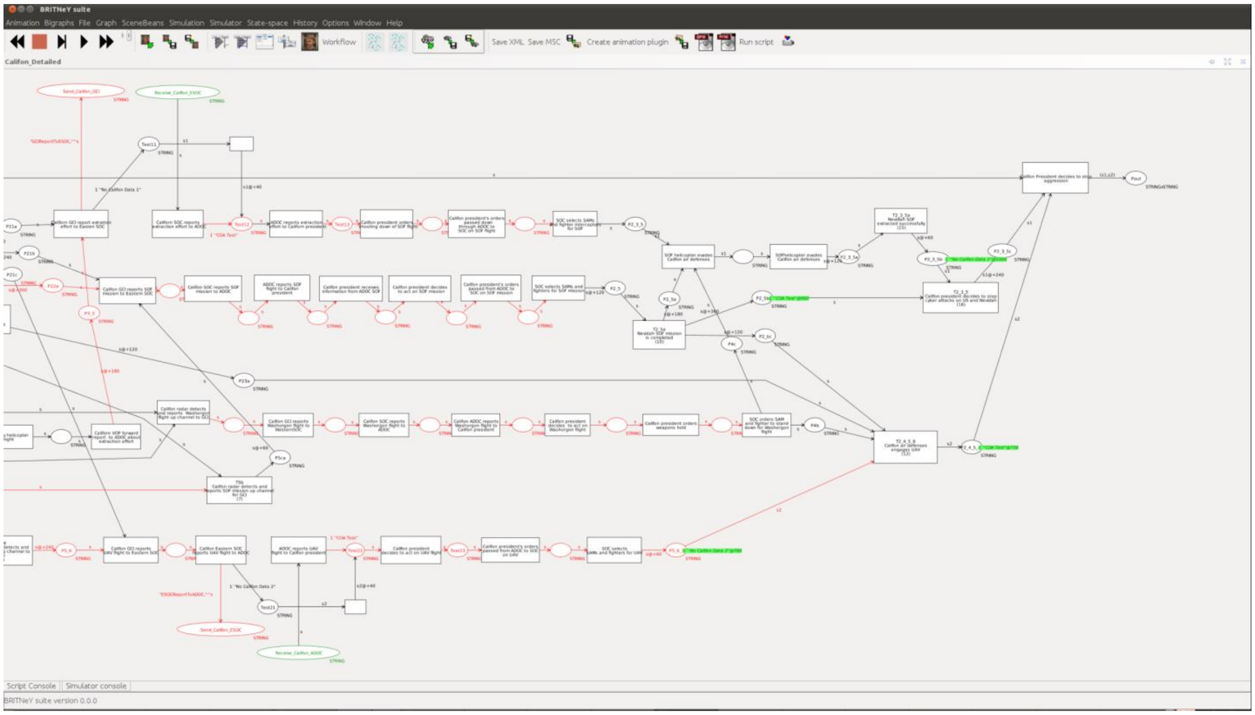


Figure D - 5: BRITNeY CPN Model execution

D.10 SCENEBEANS ANIMATION FOR CPN MODELS:

The Scenebeans animations provide an additional and intuitive way of looking into CPN model execution. The BRITNeY visualization is highly detailed as it shows the exact CPN model with exactly how and which tokens are flowing from which place to which place. However, to provide an easier to parse and intuitive display, the team at George Mason University extended the CPN simulation by incorporating a variety of scenebeans animations.

APPENDIX E – INTEGRATING CPN MODELS WITH C2WT

The purpose of this Appendix is to describe how we configure a CPN model in GME model and configure it so that it can run as a federate in C2WT. It also provides the interface description of CPN federate and HLA.

This Appendix does not describe installation and configuration of CPNTools, BRITNeY software, or C2WT. User can use Appendix B for the details of installation and configuration of C2WT.

Below are the steps to configure a CPN model to run as a federate in C2WT:

Step 1: Design a CPN model for a scenario. Create separate input and output places for input to it from HLA and output to HLA. Make sure that names of all output places start with "Send" (e.g., Send_Command, SendCoordinateMessage) and names of all input places start with "Receive" (e.g., ReceiveOpsFeedback, Receive_Ops_Report).

Step 2a: The output tokens to be sent to HLA must arrive at output places in the following format:

```
1`"InteractionName,Comma-separated-Parameters"@Timestamp
```

For example: `` 1`"Instruction,ISRD,TET,REVISE_ITNL"@200.5 ``

Above example shows an interaction "Instruction" to be sent to HLA with a timestamp of 200.5. This interaction has 3 parameters:

- Name of the organization issuing a command (type String)
- Name of the organization receiving a command (type String)
- The command string

Note: The parameter values themselves cannot contain a comma.

Step 2b: The input tokens that come from HLA arrive on the input places in the following format:

```
1`"InteractionName,Comma-separated-Parameters"@Timestamp
```

For example: `` 1`"Response,TET,ISRD,true"@250.2 ``

Above example shows an interaction "Response" that arrived from HLA to CPN with a timestamp of 250.2. This interaction has 3 parameters:

- Name of the organization sending the response (type String)
- Name of the organization receiving the response (type String)
- Flag Indicating command execution success (type Boolean)

Note: Again the parameter values themselves can't contain comma.

IMPORTANT NOTE ABOUT Step 2 and 3 above:

The time-stamp above is always in the CPN time. By default, CPN uses milliseconds, but that's modifiable in CPN. Whatever unit is chosen in CPN, that's what becomes the units of these timestamps. This unit needs to be configured in GME model to appropriately generate CPN federate related sources.

Step 3: Once the places for input and output are marked/designed as above, the CPN model needs to be annotated so that its clock can be controlled generically by C2WT. For this, at the top-level of the CPN model, add a place named "null_p" of type INT (i.e., int timed), a transition named "null_t" and an arc from the place "null_p" to the transition "null_t" with a label "m" on the arc.

Step 4: Once we have a CPN model, with input and output places defined as above, the next step is to create a CPN FOM using GME (i.e. the corresponding interaction model with parameter types, etc. For example, for above two examples, we would create two interactions in GME model for the scenario and define parameters for those two interactions with data type selected as above.

Step 5: We create a CPNFederate model in GME and define its properties. One of the properties is 'CPNTimeRatio' which is the ratio of time units assumed in CPN model over the federate time unit (which is always 1 second). So, for example, if CPN model assumes time delays in minutes, the CPNTimeRatio should be set to 60. On the other hand if the time delays in CPN were assumed to be in hours, the CPNTimeRatio should be set to 3600.

IMPORTANT: The minimum step in CPNTools is 1, so the step size of CPN federate cannot be less than the CPNTimeRatio.

Step 6: We need to add the step size (MaxStep) and Lookahead parameters of the CPN federate, which are normally set as 1.0 and 0.1 seconds respectively. However, this can be changed depending on the simulation fidelity requirements.

Step 7: Next we specify the name of the CPN model file (.CPN). The convention in CASIM is that the CPN model file is located at <project_root>\src\<demo_name>\models\cpn directory and the GME model at <project_root>\src\<demo_name>\models\gme directory. For example, for Pacifica1 demo, the CPN model is located at directory <project_root>\src\Pacifica1\models\cpn and the GME model is located at <project_root>\src\Pacifica1\models\gme. Please note, however, that the name to be specified in the 'CPNFile' attribute of the CPN federate is just the name of the CPN model (e.g., Pacifica1.cpn) without any paths. The attribute 'ConfigFile' is not used anymore, because the configuration file is now generated by the C2W Main Interpreter.

Step 8: Next we double-click the CPNFederate to open the model and then run the CPNImportInterpreter to import the CPN model in GME, such that the input and output places become places in the CPN federate model in GME. The convention of naming input and output places to begin with "Receive" and "Send" respectively helps here because only those places which have their names beginning with "Send" or "Receive" are imported as ports of the CPN federate.

Step 9: Next, the publish and subscribe relations of the CPN federate needs to be specified. This means connecting the incoming interactions to appropriate input port of the CPN federate (beginning with "Receive") and outgoing interactions to appropriate output port of the CPN federate (beginning with "Send").

Step 10: The last step involves running the C2W Main Interpreter, which generates all the source code and configuration files needed to run the CPN federate!

NOTE:The CPN configuration file is now automatically generated and no longer needs to be manually created.

APPENDIX F – CYBER EFFECTS AVAILABLE IN CASIM

This Appendix lists the planned cyber effect categories that can be supported by the OMNeT++ federate. This list is a list of *possible* events, *not* a Cyber COA to be integrated as it is into a scenario. Consequently, in the absence of a cyber COA, the scenario will specify when cyber events will occur and how.

When a network model is created, the appropriate simulation modules should be added to the model (e.g. an *AttackNode* module to a specific host) and that simulation module will be activated when the appropriate interaction is sent to the OMNeT++ federate (e.g. Start_AttackNode interaction). The same module will be deactivated by a similar interaction (e.g. a Stop_AttackNode interaction).

F.1 LIST OF CYBER EFFECTS IN CASIM

Terminology:

- A ‘node’ below can be a host computer, a switch, or a router.
- A ‘link’ is a direct connection between two nodes. That is a link can be a specific connection between a router and a switch.
- A ‘network’ is a graph of interconnected hosts, switchers, routers. It may be the entire network in the simulation, or any part of it.

1) Attack a node

Name: *AttackNode*
Description : Completely disable a specific node. Node does not function.
Parameters:
- nodeFullPath: The full path in the network of the node that will be disabled.

<u>*Status:*</u> It has been been fully implemented and tested.

2) Attack a network

Name: *AttackNetwork*
Description: Change the behavior of a network.
Parameters:

- network	: Address of the network to be attacked.
- type:	: Type of attack from the list: - completely disable network, not traffic - reduce the communication bandwidth on each link - increase latency on each link - increase packet loss on each link

<u>*Status:*</u> It has been partially implemented. Only first type has been completely implemented – which completely disables a given network. This first type is also tested.

3) Attack a link

Name: *AttackLink*

Description: Change the behavior of a specific network link.
Parameters:

- from node : one endpoint of the link
- to node : other endpoint of the link
- type: : Type of attack from the list – same as type in *AttackNetwork*.

Status: It has been NOT been implemented yet.

4) Network-Filter attack

Name: *NetworkFilterAttack*
Description: Filter (i.e. drop) transmission of packets flowing between a given network address to another network address via a given node.
Parameters:

- srcNetworkAddress : src network address (could be an address of a host or subnet)
- dstNetworkAddress : dst network address (could be an address of a host or subnet)
- nodeFullPath : The full network path of the node where the filter attack is deployed

Status: It has been fully implemented and tested.

5) Replay attack

Name: *ReplayAttack*
Description: A malicious node (usually a router) intercepts and buffers packets for a given duration (initiated using *RecordPacketsForReplayAttack* interaction), and when activated (using *StartReplayAttack* interaction), it 'replays' them in order until the attack is ceased (using *CeaseReplayAttack interaction*) or terminated (using *TerminateReplayAttack* interaction).
Parameters:

- SrcNetworkAddress:Source network address (could be an address of a host or subnet) from where the packets to be recorded originate from.
- DstNetworkAddress: Destination network address (could be an address of a host or subnet) to where the packets to be recorded are destined.
- RecordingNodeFullPath: The full network path of the node where the relay attack is deployed.
- RecordDurationInSecs: Duration in seconds for which the packets are recorded for the attack.

Status: It has been fully implemented and tested.

6) Packet modification attack

Name: *ModifyPackets*
Description: The attacker intercepts, inspects, modifies, and then sends out the packets.
Parameters:

- router: the router where the packet modification happens

Status: It has been NOT been implemented yet. It is trivial to implement in such a way that the network basically corrupts the data.

7) Data injection attack

Name: *InjectPacket*
Description: A packet is injected into a specific link in the network.
Parameters:

- from node	: one endpoint of the link
- to node	: other endpoint of the link
- packet	: the payload that gets injected

Status: It has been NOT been implemented yet. It is trivial to implement in such a way that the network injects dummy data.

8) Out-of-order packets attack

Name: *OutofOrderPackets*
Description: A specific node in the network is buffering and re-sequencing packets. That is, the sending order will be different from the receive order. When the attack is launched, the node records the packets for the given duration periodically (period = record duration) and replays them in random order.
Parameters:

- SrcNetworkAddress: Source network address (could be an address of a host or subnet) from where the packets to be recorded originate from.
- DstNetworkAddress: Destination network address (could be an address of a host or subnet) to where the packets to be recorded are destined.
- RecordingNodeFullPath: The full network path of the node where the relay attack is deployed.
- RecordDurationInSecs: Duration in seconds for which the packets are recorded for the attack.

Status: It has been fully implemented, but not tested.

9) Sniffing attack

Name: *SnifferAttack*
Description: A node relays all messages going through it to another listener host.
Parameters:

- NodeFullPath: The full network path of the node where the sniffer attack is deployed.
- ListenerHostIPAddress: Network address of the listener host where packets are duplicated.

Status: It has been fully implemented and tested.

10) Masquerading attack

Name: *MasqueradeAttack*

Description: A malicious host masquerades as another, legal host. All packets are intercepted by malicious host and are responded to. The legal host does not see anything from the intercepted traffic.

Parameters:

- host : the malicious host
- in-lieu-of host : the legal host the packets were meant for

<u>Status:</u> It has been NOT been implemented yet.

11) DNS poisoning attack

Name: *DNSPoisonAttack*

Description: An entry in the Domain Name Service (DNS) host is modified, such that all lookups for the 'affected' host will result in the address of the 'attacker' host. As a consequence, all subsequent traffic meant for the affected host will be relayed to the attacker host.

Parameters:

- dns host : the host that runs the DNS server
- affected host : whose entry in the DNS database is modified
- attacker host : whose name will replace the affected host in the DNS

<u>Status:</u> It has been NOT been implemented yet.

12) Routing table modification attack

Name: *ModifyRouteAttack*

Description: Change an entry in the routing tables of a node (usually a router). Subsequent network packets will be mis-routed according to the new routing table.

Parameters:

- NodeFullPath: The full network path of the node where the routing table modification attack is deployed.
- InterfaceEntry: ID of the network interface in the route entry.
- NetworkAddress: Network address of the destination in the route entry.
- GatewayAddress: Network address of the gateway node in the route entry.

<u>Status:</u> It has been implemented, but not tested.

13) Delay Node attack

Name: *DelayNodeAttack*

Description: Delay flow of packets along node in the network. This can be used for things like slowing down certain routers, switches, or hosts, in the network such that the communication in a specific path in the network gets delayed.

Parameters:

- nodeFullPath : the node to be delayed
- delayMean : mean for the delay in seconds for the node

- delayStdDev : standard deviation for the delay

Status: It has been fully implemented and tested.

14) Delay Path attack

Name: *DelayPathAttack*

Assumption: The network model uses *static* routes (no route modification during execution)

Description: Delay flow of packets along a path in the network. This can be used for communication delay along a specific path in the network.

Parameters:

- firstNodeFullPath : first node of the path to be delayed
- lastNodeFullPath : last node of the path to be delayed
- delayMean : mean for the delay in milliseconds at each node along this delayed path
- delayVariance : variance for the delay
- enabled : whether delay is enabled or disabled

Status: It has NOT been implemented, but is easy to implement.

F.2 CYBER EFFECTS AND OPTIONS TO ACHIEVE THEM IN OMNeT++ LIBRARY

Table F - 1: Cyber effects and options in CASIM's OMNeT++ generic library

#	Effects	Options to achieve the effect	Implemente d
1	Timeout in receiving Msgs	a. StartNodeAttack b. StartDisableNetworkAttack c. StartDelayNodeAttack d. StartNetworkFilterAttack e. StartLinkAttack	a. Yes b. Yes c. Yes d. Yes e. No
2	Msgs repeated in a loop (Replay attack)	a. StartReplayAttack	a. Yes
3	Msgs arriving in a different order than sent	a. StartOutOfOrderPacketsAttack	a. Yes
4	Msgs sniffed and **also** sent to listener host	a. StartSnifferAttack	a. Yes
5	Msgs routed to a diff. host than intended (and not to intended host)	a. Add/Drop RouteToRoutingTable b. StartDNSPoisiningAttack c. StartMasqueradingAttack	a. Yes b. No c. No
6	Msgs are corrupted/modified when received	a. StartModifyPacketsAttack	a. Yes
7	Unintended/Faulty Msgs received	a. StartDataInjectionAttack	a. No

APPENDIX G – AUTOMATED MAPPING TECHNOLOGY IN CASIM

G.1 INTRODUCTION

One of the core functionality of the next generation of C2 Wind Tunnel is to support easier integration of a variety of simulation tools. The aim is to be able to support the tools without user having to write code for encoding/decoding message in formats that other simulation tools understand. Additionally, if a simulation tool has already been federated to work with HLA-RTI, then the automated mapping technology should enable the use of its proprietary FOM (Federation Object Model) models and automated message encoding/decoding into FOM used by other simulator tools or a particular scenario/demo model.

The figure below illustrates the architecture of next-generation C2 Wind Tunnel and how mappers are used in order to support automated message translation among different simulation tools.

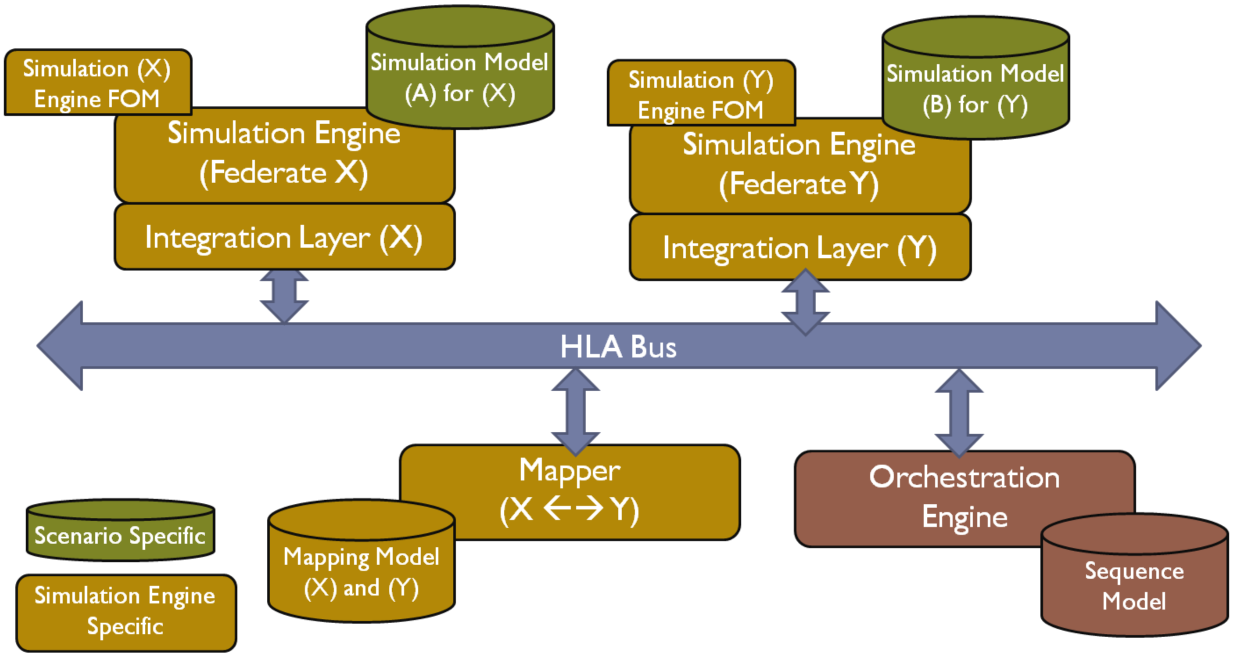

Simulation (X) Engine FOM
Simulation Model (A) for (X)
Simulation Engine (Federate X)
Integration Layer (X)
Simulation (Y) Engine FOM
Simulation Model (B) for (Y)
Simulation Engine (Federate Y)
Integration Layer (Y)
HLA Bus
Mapper (X ←→ Y)
Orchestration Engine
Scenario Specific
Simulation Engine Specific
Mapping Model (X) and (Y)
Sequence Model

Figure G - 1: Architecture of next-generation C2 Wind Tunnel

In the above figure, we can see that there are two simulation engines ‘X’ and ‘Y’ are being used in a sample federation. The simulation engine ‘X’ uses FOM/SOM definition ‘A’ and the simulation engine ‘Y’ uses the FOM definition ‘B’. As such the FOM definitions ‘A’ and ‘B’ contain quite different interaction models. In order for the two simulation engines to understand the interactions of one another, there must be a way to translate the interactions according to their own FOM/SOM definitions.

Here, we use a novel concept of a Mapper federate that subscribes and publishes interactions from FOMs both 'A' and 'B'. The mapper is configured with appropriate mapping definitions. Using these mappings, it performs the translation of interactions from one FOM to the other.

G.2 CURRENT STATUS

Currently, the mapper federate support has been completed in next-generation C2 Wind Tunnel. A special federate model with a different icon (as shown below) is integrated into the CASIM GME paradigm. The logic of the mapper federates is specified using graphical models. The mappings are specified as both graphical and textual (for complex conversions using Java like syntax). The Mapper federate code is automatically generated from the models and is fully compatible with the federation using it.

Figure G - 2: Mapper federate icon

G.3 MAPPING TYPES

CASIM supports mappings between interactions in a variety of ways. The most basic type of mapping involves conversion of those interactions that do not have any parameters. Figure F-3 shows an example of mappings for conversion of basic interactions that do not have any parameters.

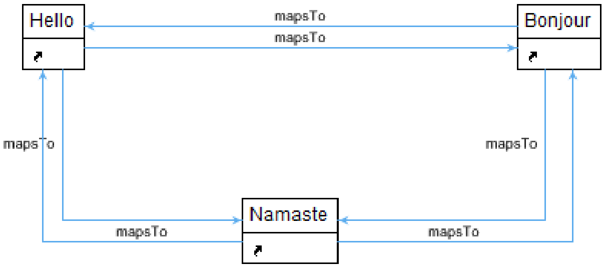


Figure G - 3: Mappings among interactions with no parameters

Another type of mapping involves interactions that contain parameters, but in both the interactions the data-type and order of these parameters are the same. Figure F-4 below shows an example of mapping between interactions EUDebtBAC and EUDebtSG. Both these interactions have an integer parameter so the mapper straightforwardly converts one interaction into other type by copying the data value.

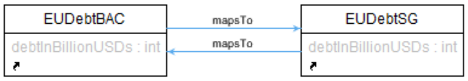


Figure G - 4: Mapping among interaction with parameters, but with no data conversion

CASIM also supports mapping among interactions that contain parameters with different data type. The next such mapping example involves conversion of one interaction into another type such that both interactions contain same number of parameters and also in the same order. The mapper in CASIM automatically figures out the data types of the parameters of both interactions, and assuming the matching order of parameters in the two interactions mapped, it translates one into another by automatically converting data from one type to the other. For example, Figure F-5 shows that when a FrenchTransferReceipt interaction is received by the mapper, which has a parameter 'id' of type 'int', it generates an equivalent interaction USTransferReceipt with no data conversion needed for the 'id' parameter, and an equivalent interaction IndianTransferReceipt with the value of 'id' converted to a 'String'.

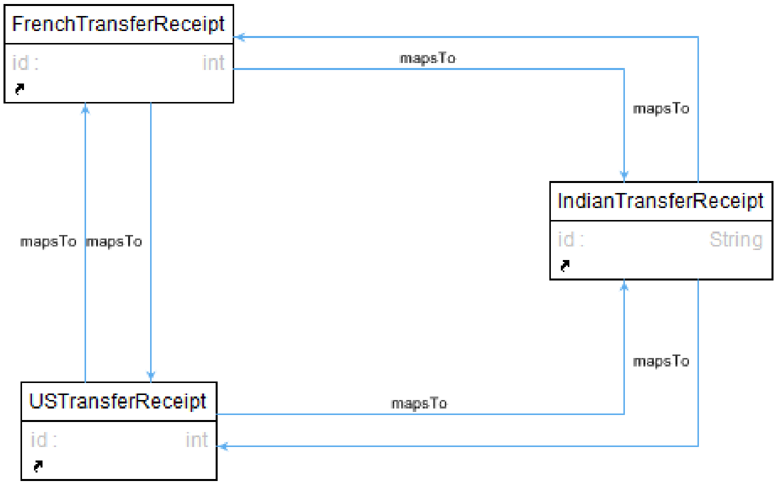
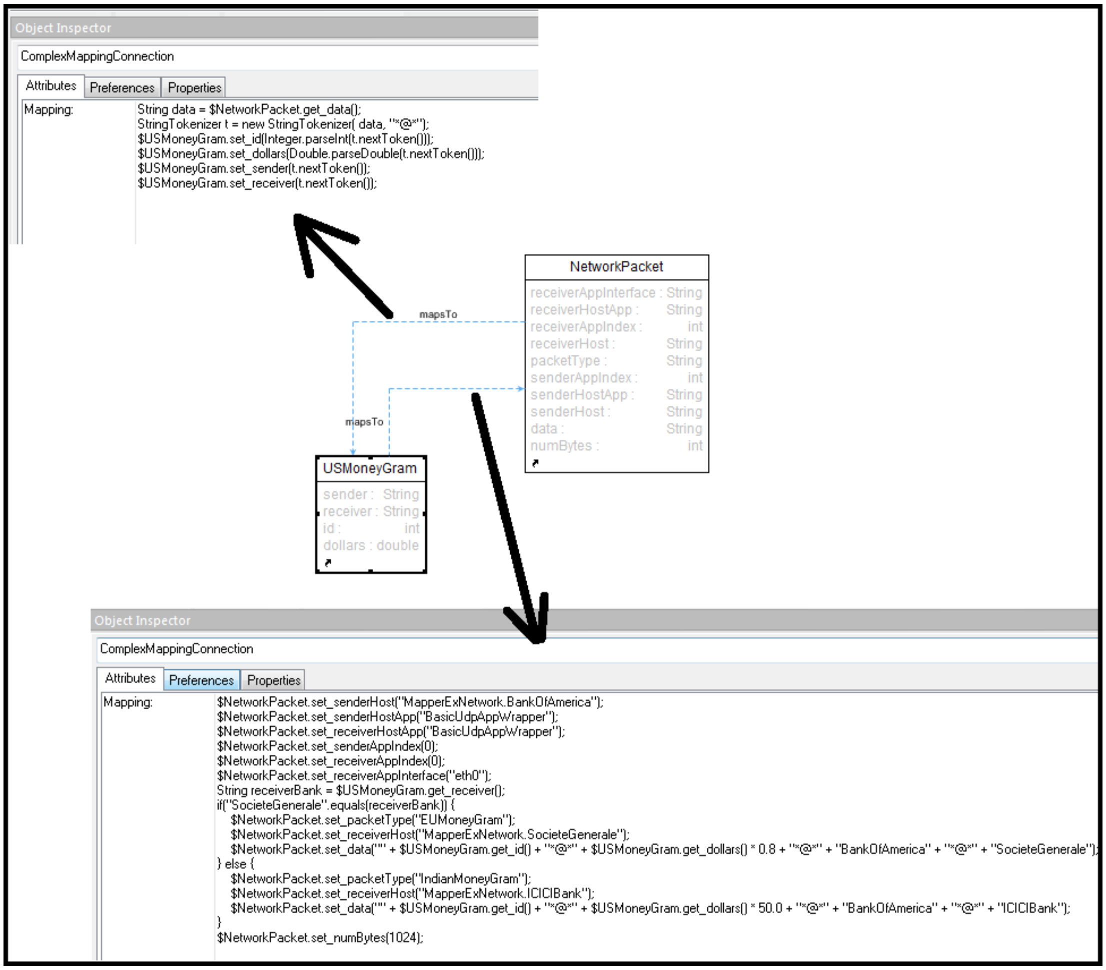


Figure G - 5: Mapping support with automatic data conversion

When a more complex set of interactions are involved that need a much more complex translation logic than the above three easier mechanisms, CASIM supports user-defined Java-like code in the mapping models. The convention is used to denote the interactions mapped using a dollar sign followed by name of the modeling element representing the interaction in the mappings. Figure F-6 shows an example of such complex mapping.

In this example user needs to translate interaction USMoneyGram into a NetworkPacket and vice versa. In the object models user has detailed parameter fields of these interactions. Creating a NetworkPacket interaction, for example, requires filling parameters like senderHost, receiverHost, etc. The way to fill these parameter values are completely domain dependent and user is free to write any conversion code as shown in the Figure. Note the use of dollar signs in the conversion code. This is a convention to refer to the participant interactions in the conversion code. When the code generator reads this code, it automatically replaces the tokens with right calls using appropriate RTI calls.



```
String data = $NetworkPacket.get_data();
StringTokenizer t = new StringTokenizer( data, "*@*");
$USMoneyGram.set_id(Integer.parseInt(t.nextToken()));
$USMoneyGram.set_dollars(Double.parseDouble(t.nextToken()));
$USMoneyGram.set_sender(t.nextToken());
$USMoneyGram.set_receiver(t.nextToken());
```

```
$NetworkPacket.set_senderHost("MapperExNetwork.BankOfAmerica");
$NetworkPacket.set_senderHostApp("BasicUdpAppWrapper");
$NetworkPacket.set_receiverHostApp("BasicUdpAppWrapper");
$NetworkPacket.set_senderAppIndex(0);
$NetworkPacket.set_receiverAppIndex(0);
$NetworkPacket.set_receiverAppInterface("eth0");
String receiverBank = $USMoneyGram.get_receiver();
if("SocieteGenerale".equals(receiverBank)) {
   $NetworkPacket.set_packetType("EUMoneyGram");
   $NetworkPacket.set_receiverHost("MapperExNetwork.SocieteGenerale");
   $NetworkPacket.set_data("" + $USMoneyGram.get_id() + "*@*" + $USMoneyGram.get_dollars() * 0.8 + "*@*" + "BankOfAmerica" + "*@*" + "SocieteGenerale");
} else {
   $NetworkPacket.set_packetType("IndianMoneyGram");
   $NetworkPacket.set_receiverHost("MapperExNetwork.ICICIBank");
   $NetworkPacket.set_data("" + $USMoneyGram.get_id() + "*@*" + $USMoneyGram.get_dollars() * 50.0 + "*@*" + "BankOfAmerica" + "*@*" + "ICICIBank");
}
$NetworkPacket.set_numBytes(1024);
```

Figure G - 6: Complex mappings with user code

APPENDIX H – FAULT MANAGEMENT IN CASIM

H.1 INTRODUCTION

One of the objectives in the next generation C2 Wind Tunnel is to provide better fault management support in the C2WT infrastructure. A wide variety of model-integration improvements and customized experimentation support scripts are already provided with the new infrastructure. However, it is also highly important to provide a better fault handling. This not only includes an error handling in the code that is exercised at run-time, but also during modeling and experimentation. In this Appendix, we attempt to list various fault handling tasks that can be developed in NGC2WT to provide a more robust usage experience.

H.2 ERROR-HANDLING FOR INVALID MODELS

Not-only do these user errors need to be detected, but an appropriate error dialog should be generated that points the user as close as possible to the actual error location in the model.

H.2.1 Modeling time error-handling (detection happens during modeling using OCL constraints)

COA models:

☒ An OutcomeFilter can have only 1 outgoing connection.
☒ An OutcomeFilter can have only 1 incoming connection and that incoming connection must originate from an Outcome type COA element.
☒ Incoming connection of an OutcomeFilter must be of type "Outcome2Filter".
☒ Outgoing connection of an OutcomeFilter must be of type "Filter2COAElement".
☒ "Outcome2Filter" connection can be used only from an Outcome node to an OutcomeFilter node.
☒ "Filter2COAElement" connection can be used only from an OutcomeFilter node.
☒ TerminateCOA cannot have any outgoing connections.
☒ All outgoing connections from "ProbabilisticChoice" node must be of type "COAFlowWithProbability".
☒ "COAFlowWithProbability" connection can only be used when the source node is of type "ProbabilisticChoice".
☒ The "minBranchesToAwait" must be less than or equal to the no. of incoming branches to the AwaitN node.
☒ Only 1 outgoing connection of type "COAFlow" is allowed from a SyncPoint node (this represents the successful achievement of the synchronization point).
☒ All outgoing connections from a "SyncPoint" node should be of type “COAFlow” or "COAException".
☒ "COAException" type connection can only be added after user has specified the default "COAFlow" type of connection.
☒ "COAException" type connection can only be used when the source node is of type "SyncPoint".

☒ The "minBranchesToSync" must be less than or equal to the no. of incoming branches to the SyncPoint node.
☒ The "branchesFinishedCondition" can be empty for only 1 "COAException" outgoing connection from a SyncPoint node (representing default handling if no other conditions succeeded).
☒ For any of the COAFlow, COAFlowWithProbability, COAException, Outcome2Filter, and Filter2COAElement connections, the source and destination cannot be the same node (i.e. no self-connections in COA models).
☒ For any set of two COAFlow, COAFlowWithProbability, COAException, Outcome2Filter, or Filter2COAElement connections, both source and destination cannot be the same.

FOM and SOM models:

☒ CPNFederate should not directly publish/subscribe an interaction (only via a port corresponding to output/input place is supported).
☒ Each port of a CPNFederate can have maximum one connection.
☒ CPN's port names must start with "Send_" and "Receive_".
☒ CPNFederate should not allow subscribe connections to ports corresponding to output places.
☒ CPNFederate should not allow publish connections to ports corresponding to input places.

H.2.2 Modeling time error-handling (detection happens during model interpretation)

☒ Check if user has properly created an environment variable $C2WTROOT.
☒ Check if the interpreters have been configured properly or not.
  ☒ Check whether GME classpath is set right.
  ☒ Check if the required jar files and C2W interpreter classes are found in classpath.
☒ Check if name of a federate ends in "Mapper" (reserved for Mapper federates).
☒ Check if name of a mapper federate doesn't end in "Mapper".
☒ Check if IP address and credentials of all network nodes have been specified. Also, make sure that the IP address is syntactically correct.
☒ Check if there are null references in the model.
☒ Check if the duration and time-points are all specified positive.
☒ Check if all of the needed parameters are specified and in syntactically correct way for (a) injected interactions (b) action elements of COAs.
☒ Injection time of injected interaction should not be negative.
☒ If any interaction in a set of two interactions has some locally declared parameters, and there is a mapping between them, then it cannot be a "direct" mapping (i.e., mapping specification cannot be empty).
☒ A federate in an experiment model should be allowed to be connected to only one computer and it must be connected.
☒ A deployment model can have only one experiment reference in it.
☒ One of the computers in a deployment must be marked as MAIN machine.
☒ All interactions must derive from C2WinteractionRoot.

☒ C2WInteractionRoot must derive from InteractionRoot.
☒ The interaction hierarchy must be rooted – no more than 1 root nodes should be found.
☒ There should not be a cycle in any of the COAs (all COAs should be DAGs).
☒ Make sure that the time specified for a synchronization point is positive.
☒ Make sure that FlowID of any branch doesn't contain any whitespace characters.
☒ The "branchesFinishedCondition" must include space (or newline) separated flow-IDs of only those connections that are the SyncPoint node's incoming connections.
☒ COASequencesGroup models must contain at least one reference to a COA.
☒ A COA must be referred in maximum one COASequenceGroup in an experiment model.
☒ Make sure no COA reference in a COASequenceGroup is null.
☒ If deployment scripts are to be generated, a project must have at least one network model with at least one computer model in it.

H.2.3 Execution time error-handling

☒ Check if mappings have been correctly specified (provided detailed comments in the generated code so that users can tie compilation errors to invalid mappings in the model).
☒ Check if the expression of OutcomeFilter nodes are valid (provide detailed comments in the generated code so that users can tie compilation errors to invalid OutcomeFilter expressions in the model.
☒ Normalize the probabilities on ProbabilisticChoice branches to 1.0.

H.3 ERROR-HANDLING FOR INVALID EXECUTION/EXPERIMENTATION SETUP

Not-only do these user errors need to be detected, but an appropriate error dialog should be generated that points the user as close as possible to the actual error location in the model. If the experiment is run in the batch mode (non-GUI), then an appropriate error message should be recorded in the federation manager's log file.

H.3.1 Initialization time error-handling (before simulation is launched)

☒ Test if IP address and credentials are correct before launching.
☒ Test if any federation is already running before launching.
☒ Check if Java3D is installed.
☒ Check if C++ and OMNeT++ federate executable exist.
☒ Check if the required Java classes can be loaded using the classpath set in the shell scripts.
☒ Check if user has generated code (e.g. whether <federation-name>.fed file exists).
☒ Check if ProcessId project has been built and a shared object has been generated.
☒ Check if Main machine's IP address is same as in the deployment model.
☒ Make sure no two federates attempt to join the federation simultaneously (problem with portico).
☒ If the runAll script is used for an experiment, that experiment must specify either an end-time or the terminate-on-coa-finish flag.
☒ Make sure there are no compilation errors in byte-code of the compiled federation and generic C2WT Java classes.

H.3.2 Execution time error-handling (after simulation has been launched)

☒ Detect tight-loop in federation – caused by very small step-sizes and/or look-ahead values, or by generation of too many interactions in a very small time-period. Ideally, RTI is supposed to be able to handle this, but Portico doesn’t.
☒ Handle tight-loop and allow user to terminate entire federation from GUI. In Non-GUI mode, just log the warning message and continue running.
☒ Terminate the entire federation when user has chosen to terminate it. This means that the federation must terminate in entirety even if some exception occurs somewhere or a thread is stuck indefinitely somewhere and federates never receive SimEnd interaction.

APPENDIX I – C2 WIND TUNNEL EXPERIMENT CONTROLLER

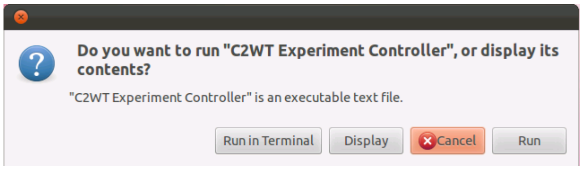
I.1 INTRODUCTION

One of the most requested requirements among users of C2 Wind Tunnel is to create a front-end graphical user-interface that is easy and intuitive to use. The C2 Wind Tunnel Experiment Controller, written as a python application serves this purpose. It can be used to automatically detect the available experiments for various scenarios users have created and generated and compiled code for. It can be used to execute any set of the selected experiments. Apart from this basic functionality, the C2 Wind Tunnel Experiment Controller supports a number of user-friendly features that together provide a very useful tool for experimentation with C2 Wind Tunnel. In this Appendix, we describe this tool in detail and discuss some future work.

I.2 TOOL OVERVIEW

The experiment controller is started with double-clicking a shortcut icon on the supplied virtual machine's desktop. This shortcut points to the python script file C2WTExperimentController.py located in the C2WT root directory (i.e., $C2WTROOT). Figure I-1 below shows the shortcut icon located on the desktop. As appropriate, user can also invoke the python script file directly to launch the experiment controller. When the shortcut icon is double-clicked, a choice is presented to the user. The 'Run' button should be pressed to launch the experiment controller.

Figure I - 1: Shortcut icon on user's desktop to launch C2WT experiment controller

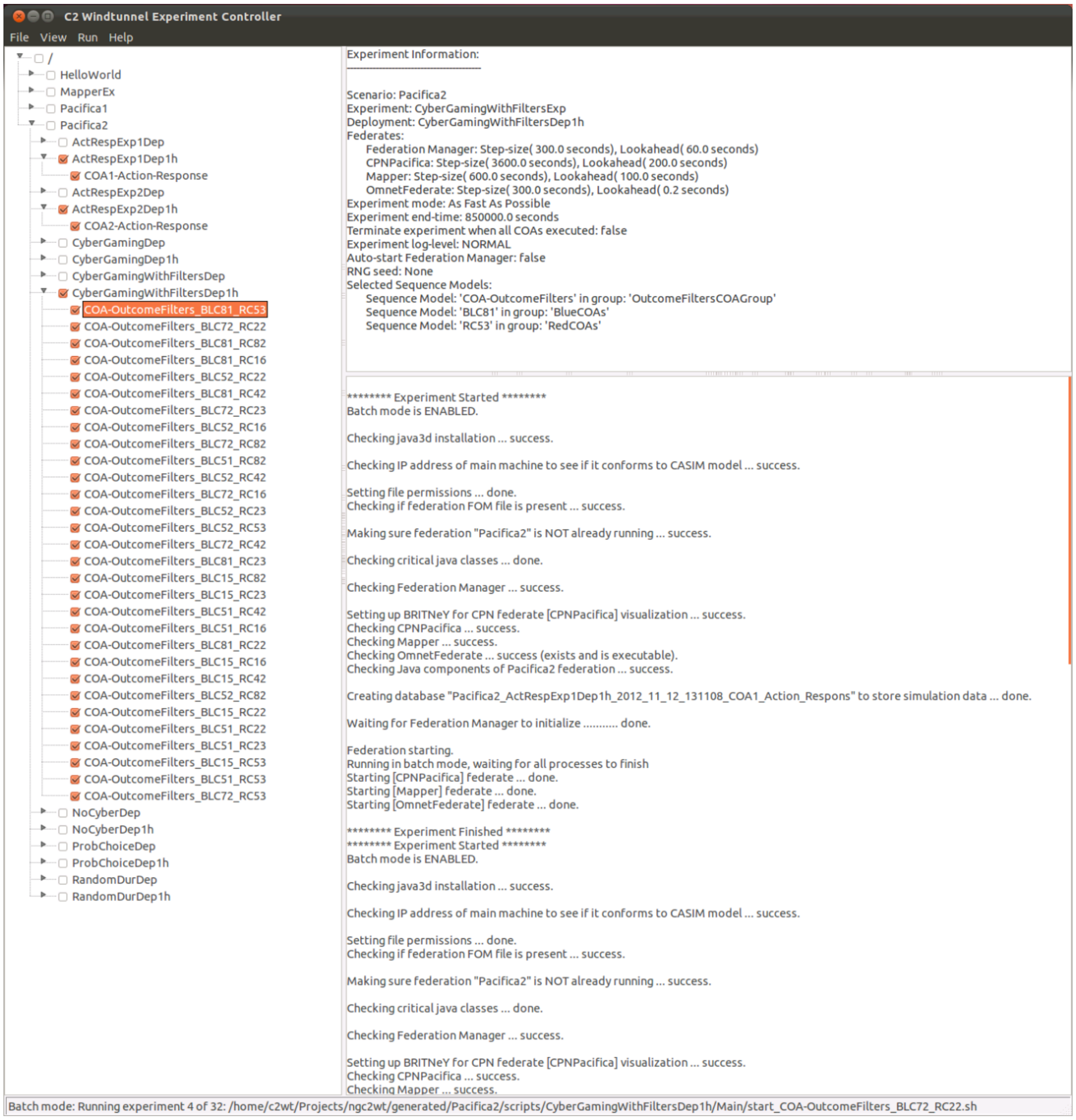

Do you want to run "C2WT Experiment Controller", or display its contents?

"C2WT Experiment Controller" is an executable text file.

Figure I - 2: Launch choice

When the experiment controller is launched, the tool scans through the $C2WTROOT directory to find out all the experiment scenarios that user has modeled and generated code for. This essentially means that after creating the scenario model in GME, including the FOM, SOM, Mapping, COAs, computation infrastructure, and deployment models, user must have run the C2WT Main Interpreter to generate the experiment sources, and C2WT Deployment Interpreter to generate the deployment scripts and configuration files. If this is the case, then the scenario is loaded properly in the experiment controller, where it can be used as part of the experiments. If the source code has not been generated or older version of interpreters was used, the experiment controller shows appropriate warning messages at load time.

Once all the scenarios and deployment scripts generated by user for various experiments are loaded by the experiment controller, the graphical user interface is displayed as shown in Figure I-3 below. The main page is divided into various sections. On the left, the tool presents all the scenarios, deployments, and experiment runs available in a tree form. Each of these scenarios, deployments, and experiment runs are selectable by checking the checkbox that appears in front of them. Note that the existence of various COA-Groups in a particular deployment may lead to more than 1 experiment run due to selection of 1 COA from each of the COA-Groups. When a checkbox is checked, all elements under that item, if any, are also automatically selected. The aggregate of the selections represents the set of experiments a user wants to run. For example, in Figure I-3, the 32 experiment runs belonging to the Pacifica2 scenario model have been selected. These correspond to three deployments ActRespExp1Dep1h, ActRespExp2Dep1h, and CyberGamingWithFilersDep1h of the Pacifica2 scenario.



```
▼ □ /
  ▶ □ HelloWorld
  ▶ □ MapperEx
  ▶ □ Pacifica1
  ▼ □ Pacifica2
    ▶ □ ActRespExp1Dep
    ▼ ☑ ActRespExp1Dep1h
        ☑ COA1-Action-Response
    ▶ □ ActRespExp2Dep
    ▼ ☑ ActRespExp2Dep1h
        ☑ COA2-Action-Response
    ▶ □ CyberGamingDep
    ▶ □ CyberGamingDep1h
    ▶ □ CyberGamingWithFiltersDep
    ▼ ☑ CyberGamingWithFiltersDep1h
        ☑ COA-OutcomeFilters_BLC81_RC53
        ☑ COA-OutcomeFilters_BLC72_RC22
        ☑ COA-OutcomeFilters_BLC81_RC82
        ☑ COA-OutcomeFilters_BLC81_RC16
        ☑ COA-OutcomeFilters_BLC52_RC22
        ☑ COA-OutcomeFilters_BLC81_RC42
        ☑ COA-OutcomeFilters_BLC72_RC23
        ☑ COA-OutcomeFilters_BLC52_RC16
        ☑ COA-OutcomeFilters_BLC72_RC82
        ☑ COA-OutcomeFilters_BLC51_RC82
        ☑ COA-OutcomeFilters_BLC52_RC42
        ☑ COA-OutcomeFilters_BLC72_RC16
        ☑ COA-OutcomeFilters_BLC52_RC23
        ☑ COA-OutcomeFilters_BLC52_RC53
        ☑ COA-OutcomeFilters_BLC72_RC42
        ☑ COA-OutcomeFilters_BLC81_RC23
        ☑ COA-OutcomeFilters_BLC15_RC82
        ☑ COA-OutcomeFilters_BLC15_RC23
        ☑ COA-OutcomeFilters_BLC51_RC42
        ☑ COA-OutcomeFilters_BLC51_RC16
        ☑ COA-OutcomeFilters_BLC81_RC22
        ☑ COA-OutcomeFilters_BLC15_RC16
        ☑ COA-OutcomeFilters_BLC15_RC42
        ☑ COA-OutcomeFilters_BLC52_RC82
        ☑ COA-OutcomeFilters_BLC15_RC22
        ☑ COA-OutcomeFilters_BLC51_RC22
        ☑ COA-OutcomeFilters_BLC51_RC23
        ☑ COA-OutcomeFilters_BLC15_RC53
        ☑ COA-OutcomeFilters_BLC51_RC53
        ☑ COA-OutcomeFilters_BLC72_RC53
    ▶ □ NoCyberDep
    ▶ □ NoCyberDep1h
    ▶ □ ProbChoiceDep
    ▶ □ ProbChoiceDep1h
    ▶ □ RandomDurDep
    ▶ □ RandomDurDep1h
```

```
Experiment Information:
-----------------------------------

Scenario: Pacifica2
Experiment: CyberGamingWithFiltersExp
Deployment: CyberGamingWithFiltersDep1h
Federates:
    Federation Manager: Step-size( 300.0 seconds), Lookahead( 60.0 seconds)
    CPNPacifica: Step-size( 3600.0 seconds), Lookahead( 200.0 seconds)
    Mapper: Step-size( 600.0 seconds), Lookahead( 100.0 seconds)
    OmnetFederate: Step-size( 300.0 seconds), Lookahead( 0.2 seconds)
Experiment mode: As Fast As Possible
Experiment end-time: 850000.0 seconds
Terminate experiment when all COAs executed: false
Experiment log-level: NORMAL
Auto-start Federation Manager: false
RNG seed: None
Selected Sequence Models:
    Sequence Model: 'COA-OutcomeFilters' in group: 'OutcomeFiltersCOAGroup'
    Sequence Model: 'BLC81' in group: 'BlueCOAs'
    Sequence Model: 'RC53' in group: 'RedCOAs'
```

```
******** Experiment Started ********
Batch mode is ENABLED.

Checking java3d installation ... success.

Checking IP address of main machine to see if it conforms to CASIM model ... success.

Setting file permissions ... done.
Checking if federation FOM file is present ... success.

Making sure federation "Pacifica2" is NOT already running ... success.

Checking critical java classes ... done.

Checking Federation Manager ... success.

Setting up BRITNeY for CPN federate [CPNPacifica] visualization ... success.
Checking CPNPacifica ... success.
Checking Mapper ... success.
Checking OmnetFederate ... success (exists and is executable).
Checking Java components of Pacifica2 federation ... success.

Creating database "Pacifica2_ActRespExp1Dep1h_2012_11_12_131108_COA1_Action_Respons" to store simulation data ... done.

Waiting for Federation Manager to initialize ........... done.

Federation starting.
Running in batch mode, waiting for all processes to finish
Starting [CPNPacifica] federate ... done.
Starting [Mapper] federate ... done.
Starting [OmnetFederate] federate ... done.

******** Experiment Finished ********
******** Experiment Started ********
Batch mode is ENABLED.

Checking java3d installation ... success.

Checking IP address of main machine to see if it conforms to CASIM model ... success.

Setting file permissions ... done.
Checking if federation FOM file is present ... success.

Making sure federation "Pacifica2" is NOT already running ... success.

Checking critical java classes ... done.

Checking Federation Manager ... success.

Setting up BRITNeY for CPN federate [CPNPacifica] visualization ... success.
Checking CPNPacifica ... success.
Checking Mapper ... success.
```

```
Batch mode: Running experiment 4 of 32: /home/c2wt/Projects/ngc2wt/generated/Pacifica2/scripts/CyberGamingWithFiltersDep1h/Main/start_COA-OutcomeFilters_BLC72_RC22.sh
```

Figure I - 3: C2 Wind Tunnel Experiment Controller

The right section of the display is further divided into two parts. The top-right section provides information on the selected scenario, experiment, or the experiment run. For example, in Figure I-3, when the experiment run ‘COA-OutcomeFilters_BLCB1_RC53’ is selected, the top-right panel shows the detailed information about that experiment. It shows the name of the experiment model, scenario model, and the deployment model for the experiment. It shows which federates are being run in this particular experiment including their step-size and lookahead values. Also, it shows all the sequence models used in the experiment, and various configuration parameters of the experiment.

If only a single experiment run is selected, it can be executed either in Interactive or Batch mode. Under Batch mode, no GUIs or dialogs are shown during execution. When more than 1 experiment is selected in the left tree panel for execution, they need to be run in the Batch mode. The console messages of the running experiments are captured and presented in the bottom-right section. Figure I-3 shows the sample log messages of running the selected experiments in Batch mode. The log messages do not overwrite experiment-by-experiment during a batch execution. However, when a new set of experiments are executed (in Batch or Interactive mode), old log messages are scratched and new log messages corresponding to the currently selected experiments appear in the section.

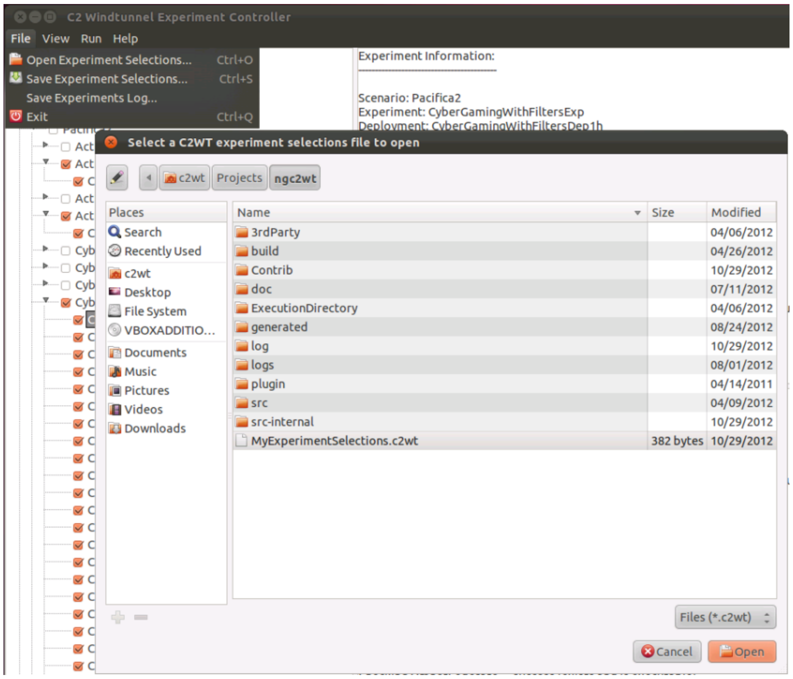
At the bottom of the experiment controller is a status bar. The status bar identifies the currently running experiment and a running count of how many experiments from the selected batch have been executed so far. Also, it provides the path of the executable shell script that the experiment controller is using to run the experiment.

I.3 MENUS

This section describes various menus of the experiment controller and their functions.

I.3.1 File Menu. Figure I-4 below shows the File menu. As shown, the first two menu items can be used to save and load a set of experiment selections. When the currently selected experiments are saved as a file for later use, they are saved with a .c2wt extension. This file can subsequently be opened in the experiment controller to restore the experiment selections. During the load time, the experiment controller reconstructs the tree-view by scanning the users $C2WTROOT directory for scenarios, deployments, and experiments. After loading the experiments, the experiment controller reads the .c2wt experiment selections file to mark the saved selections as checked. If any of the saved selections can no longer be found in the directory or they are found non-executable due to some errors, they are ignored.

Also, the ‘Save Experiments Log...’ menuitem can be used to save the console messages that appear in the bottom-right section of the experiment controller. The log messages are saved as a text file with .log extension.



Experiment Information:

Scenario: Pacifica2
Experiment: CyberGamingWithFiltersExp
Deployment: CyberGamingWithFiltersDep1h

Name	Size	Modified
3rdParty		04/06/2012
build		04/26/2012
Contrib		10/29/2012
doc		07/11/2012
ExecutionDirectory		04/06/2012
generated		08/24/2012
log		10/29/2012
logs		08/01/2012
plugin		04/14/2011
src		04/09/2012
src-internal		10/29/2012
MyExperimentSelections.c2wt	382 bytes	10/29/2012

Figure I - 4: File menu

The last menu item 'Exit' can be used to close the experiment controller.

I.3.2 View Menu.

Figure I-5 below shows the View menu.

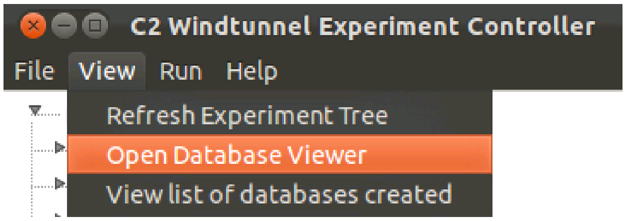

C2 Windtunnel Experiment Controller
File View Run Help
Refresh Experiment Tree
Open Database Viewer
View list of databases created

Figure I - 5: View menu

The 'Refresh Experiment Tree' forces the experiment controller to re-scan the $CWTROOT directory and recreate the scenarios, deployments, and experiments tree in the left section. This is useful if while the experiment controller is loaded, user re-generates code and deployment scripts of a scenario or has recently added a new scenario in the experiments set. Note that refreshing the experiment tree also clears all experiment selections that user might have made in the left tree. If these experiment selections are to be preserved, user can save them as a file and reload the selections after refreshing the experiment tree.

The 'Open Database Viewer' menu item is used to launch an external tool named 'DBVisualizer', which is a database client that is pre-installed in the virtual machine.

DBVisualizer is used to view the log of experiment data persisted by the infrastructure during experiment runs. These data-sets are used for extensive analysis.

The 'View list of databases created' menu item is used to show the list of databases that were created for the currently selected experiments in the last executed batch of experiments. C2 Wind Tunnel creates new databases for each experiment. The names of the databases not only use the name of the scenario, but also a date and time stamp and the names of the sequence models used in that experiment (up to a total of '64' characters as support by MySQL). When selected a dialog box is shown with the list of database names, which can be copied to the clipboard if user wants to save them as a text file.

I.3.3 Run Menu. Figure I-6 below shows the Run menu.

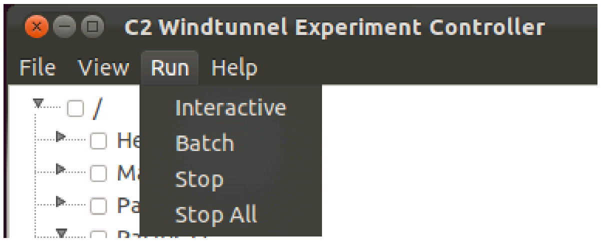


Figure I - 6: Run menu

Run menu is the key menu for experiment control. When a single experiment has been selected, the menu item 'Interactive' can be used to run the selected experiment in Interactive mode. This mode shows Federation Manager and other simulation tool graphical windows. The Federation Manager can be used to terminate the experiment manually. Also, the menu item 'Stop' of the experiment controller can be used to stop the currently executing experiment.

IMPORTANT: The 'Stop' menu can be used for a currently executing experiment only after the experiment has started executing successfully. When this happens, the log window shows that the Federation Manager has been successfully initialized and all federates that are part of the federation have started successfully.

The 'Batch' menu item starts the batch execution of currently selected experiment(s). Note that when more than one experiment is selected, only Batch mode of execution is permitted. When a batch of experiments is running, user can terminate the currently running experiment using the 'Stop' menu item. If the entire batch needs to be terminated, user can use the 'Stop All' menu item. Again, as mentioned above, 'Stop' menu item should be used only after the current experiment has started executing successfully.

I.3.4 Help Menu. Figure I-7 shows the Help menu. The menu item 'About' shows a brief information message about the experiment controller. The 'Help Contents' menu item opens a help file that contain information about how to use the experiment controller.

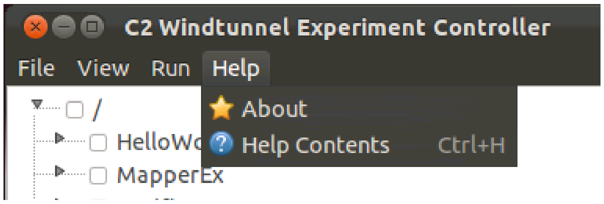


Figure I - 7: Help menu

I.4 CONCLUSION

The experiment controller provides a very user-friendly tool for the users of C2 Wind Tunnel. It can be used to not only, run various experiments in interactive and batch mode, but it also provides a rich set of utilities to control their execution. The tool provides various measures to save and store useful experiment information. A user can also execute a batch of experiments that selects experiments various scenario models simultaneously. A detailed log of currently running experiments is presented and a status bar captures information about currently executing experiment.

I.5 FUTURE WORK

Currently the experiment controller can execute experiments only one at a time. In the future, we plan to extend experiment controller to support execution of experiments across a computation cluster. This will involve executing several experiments in parallel, which will greatly reduce the overall computation time for running a large set of C2 experiments. In the future, we also plan to add new functionality to enhance experiment parameter modification right from within the tool without a need to update the GME model and regenerate the sources and scripts. Also, we plan to extend the experiment controller with the addition of a timeline view of currently executing experiments, which can present message flows up to the level of detail desired in a graphical, easy-to-visualize view.

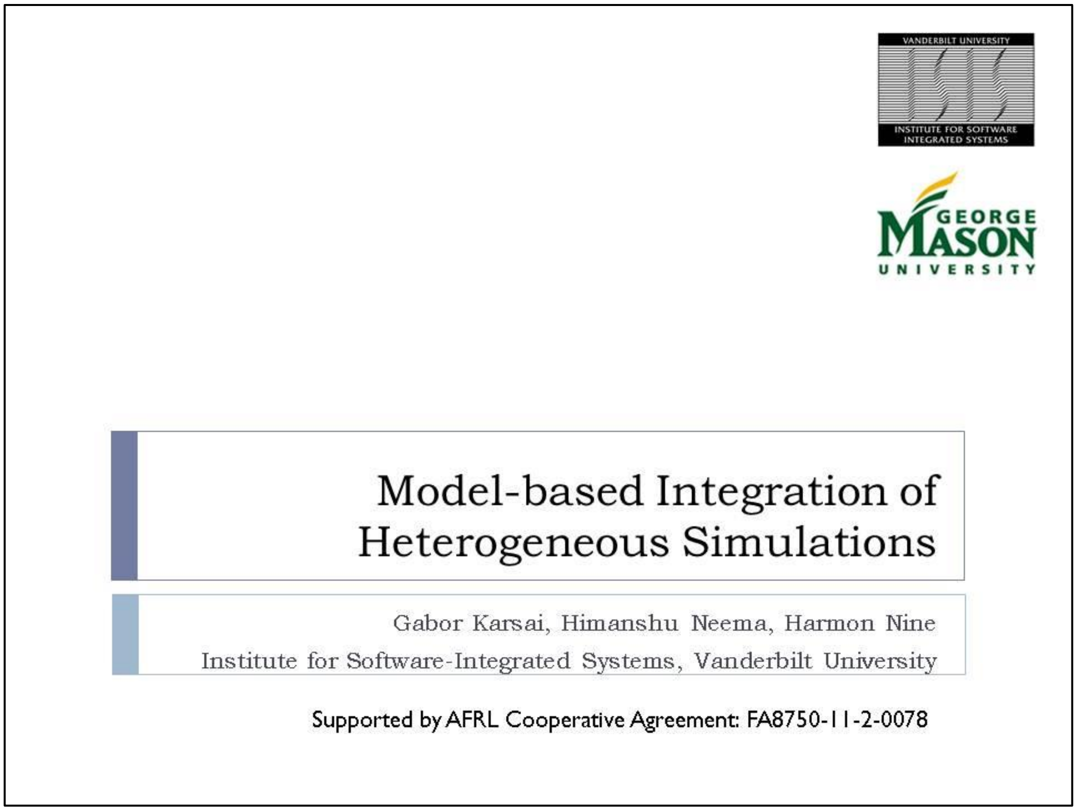
APPENDIX J – RESEARCH PUBLICATION: "MODEL-BASED INTEGRATION OF HETEROGENEOUS SIMULATIONS"

Conference:	Quantitative Methods in Defense and National Security
Venue:	George Mason University, Fairfax, VA
Date:	May 1, 2012
Authors:	Gabor Karsai, Himanshu Neema, Harmon Nine
Affiliation:	Institute for Software Integrated Systems, Vanderbilt University

J.1 ABSTRACT

Simulation-based evaluation of systems and plans is an essential tool for designers and planners. However, often heterogeneous, specialized simulation tools and models need to be integrated in an overarching framework such that meaningful results can be generated. For instance, in a C2 setting, various domains, including vehicle dynamics, communication networks, computing nodes, and human organizations and decision making processes need to be co-simulated. There is no single simulation package that can perform all of the above, but there are well-developed simulation models for the individual domains. The challenge is the semantically correct composition of models in an effective way. The talk will present a model-based approach developed in the course of the 'Command and Control Wind Tunnel' (C2WT) project and further developed during the 'Course of Action Simulation' (CASIM) project that is based on the precise modeling of simulation interactions and the mapping between those, as well as operational sequences that drive the simulation execution framework. The integration models are used to automatically generate the interfaces and translation processes, and configure an instance of the C2WT to execute the simulations in a coordinated manner. The talk will highlight the technical issues of simulation composition and present our solutions.

J.2 SLIDES



VANDERBILT UNIVERSITY
INSTITUTE FOR SOFTWARE INTEGRATED SYSTEMS

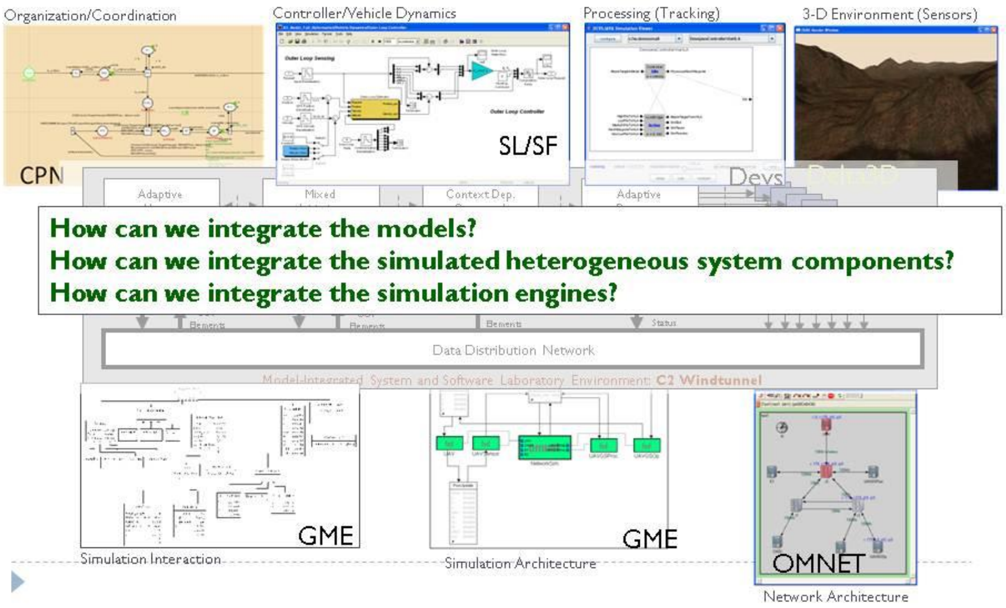
GEORGE MASON UNIVERSITY

Model-based Integration of Heterogeneous Simulations

Gabor Karsai, Himanshu Neema, Harmon Nine
Institute for Software-Integrated Systems, Vanderbilt University

Supported by AFRL Cooperative Agreement: FA8750-11-2-0078

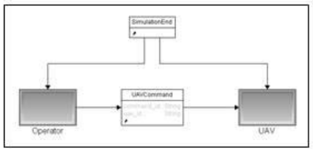
Challenges for Model and Simulation Integration



How can we integrate the models?
How can we integrate the simulated heterogeneous system components?
How can we integrate the simulation engines?

Command and Control Wind-Tunnel
Model-based Integration of Dynamic Simulations

C2W **integration** models
(data flow, timing, parameters)



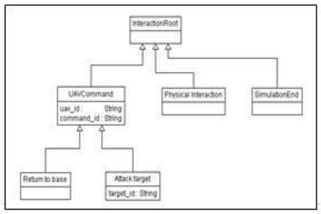
configuration

Based on C2WT models configuration files are generated for the various simulation components.
Configure how the component is connected to the simulation (input-output binding)

Domain specific
C2W simulation components

OMNET component
CPN component
Simulink component
Delta3D component

C2W **data** models
(interaction and object models)

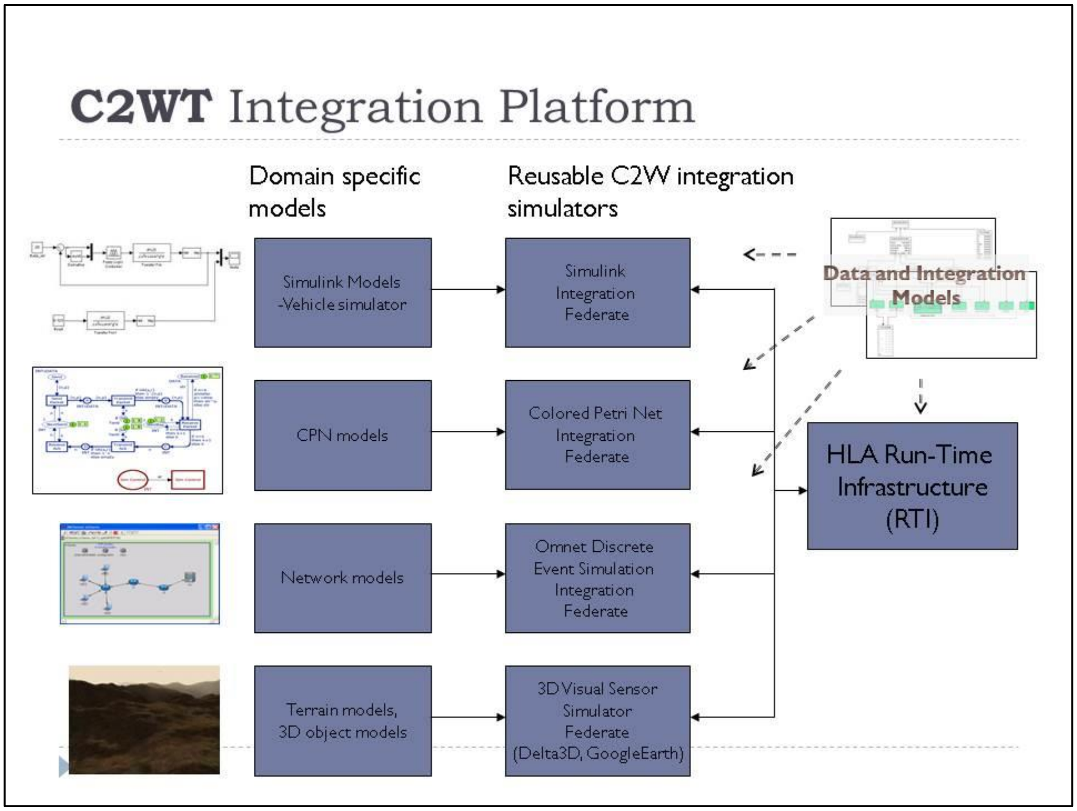


transformation

Federates have to have a common data model to be able to share data.
- Data model can be imported from domain specific models
- Domain specific models can be generated from data models

Domain specific
simulation models

Omnet models
CPN models
Simulink models
…



C2WT Integration Platform

Domain specific models

Reusable C2W integration simulators

Simulink Models -Vehicle simulator

Simulink Integration Federate

CPN models

Colored Petri Net Integration Federate

Network models

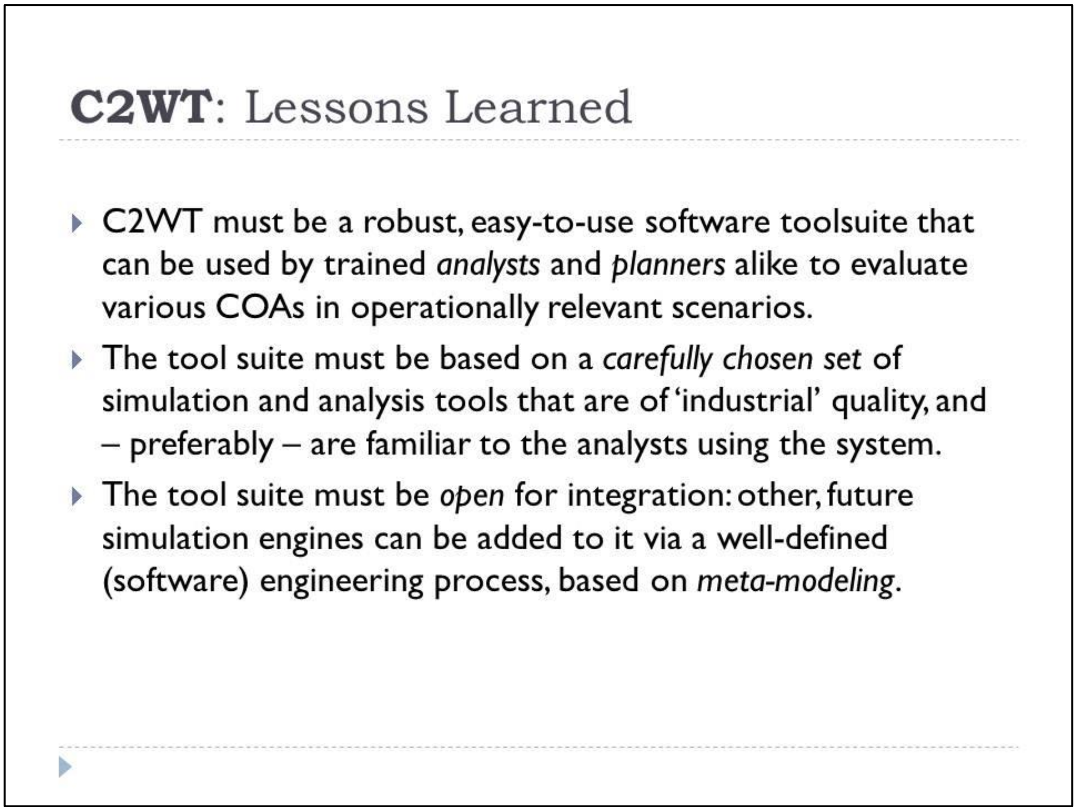
Omnet Discrete Event Simulation Integration Federate

Terrain models, 3D object models

3D Visual Sensor Simulator Federate (Delta3D, GoogleEarth)

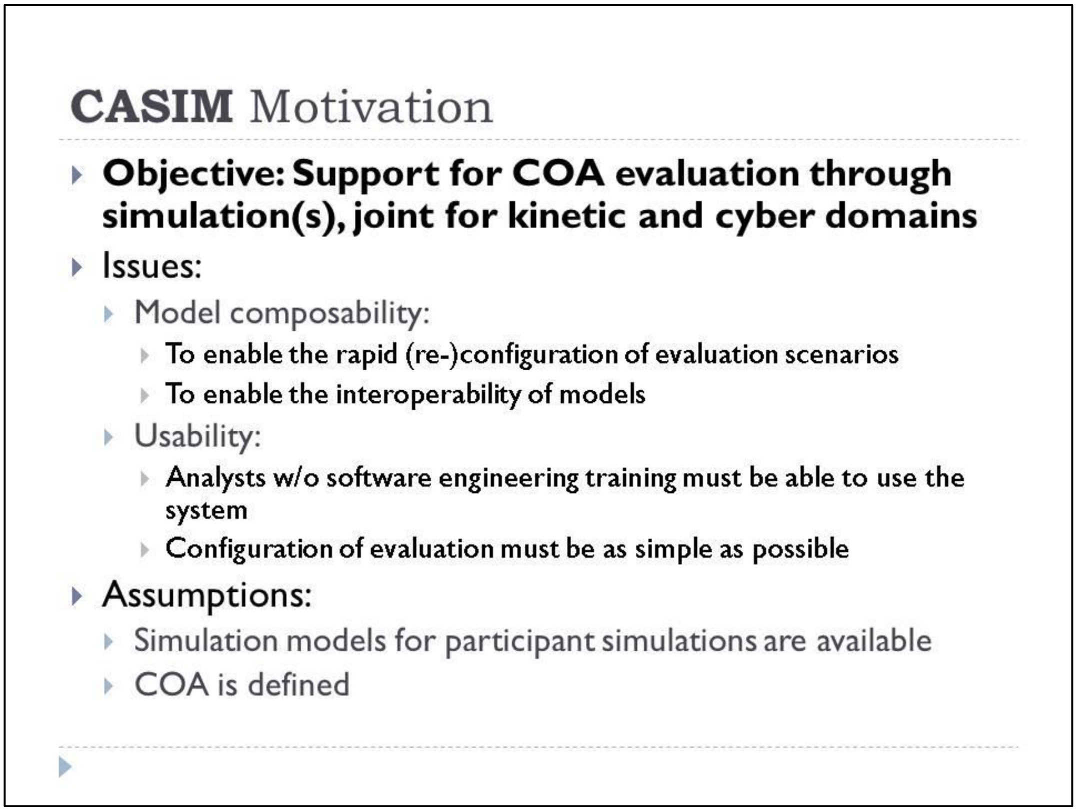
Data and Integration Models

HLA Run-Time Infrastructure (RTI)



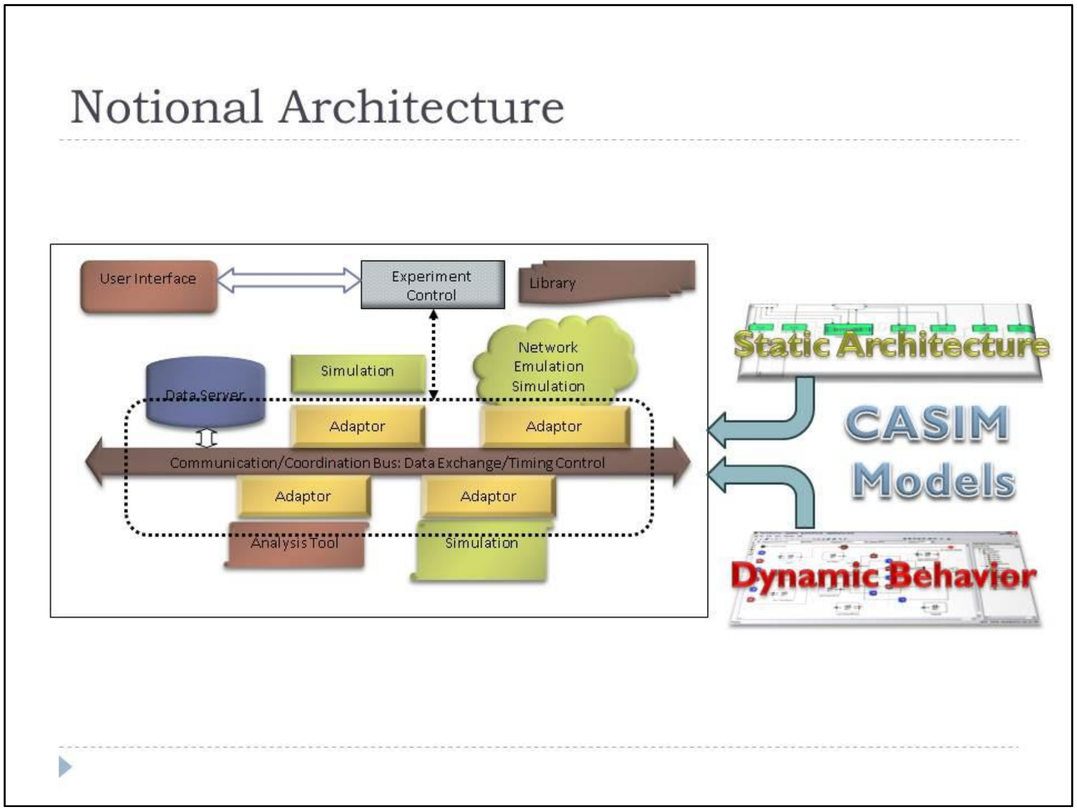
C2WT: Lessons Learned

- C2WT must be a robust, easy-to-use software toolsuite that can be used by trained *analysts* and *planners* alike to evaluate various COAs in operationally relevant scenarios.
- The tool suite must be based on a *carefully chosen set* of simulation and analysis tools that are of 'industrial' quality, and – preferably – are familiar to the analysts using the system.
- The tool suite must be *open* for integration: other, future simulation engines can be added to it via a well-defined (software) engineering process, based on *meta-modeling*.



CASIM Motivation

- **Objective: Support for COA evaluation through simulation(s), joint for kinetic and cyber domains**
- Issues:
  - Model composability:
    - To enable the rapid (re-)configuration of evaluation scenarios
    - To enable the interoperability of models
  - Usability:
    - Analysts w/o software engineering training must be able to use the system
    - Configuration of evaluation must be as simple as possible
- Assumptions:
  - Simulation models for participant simulations are available
  - COA is defined

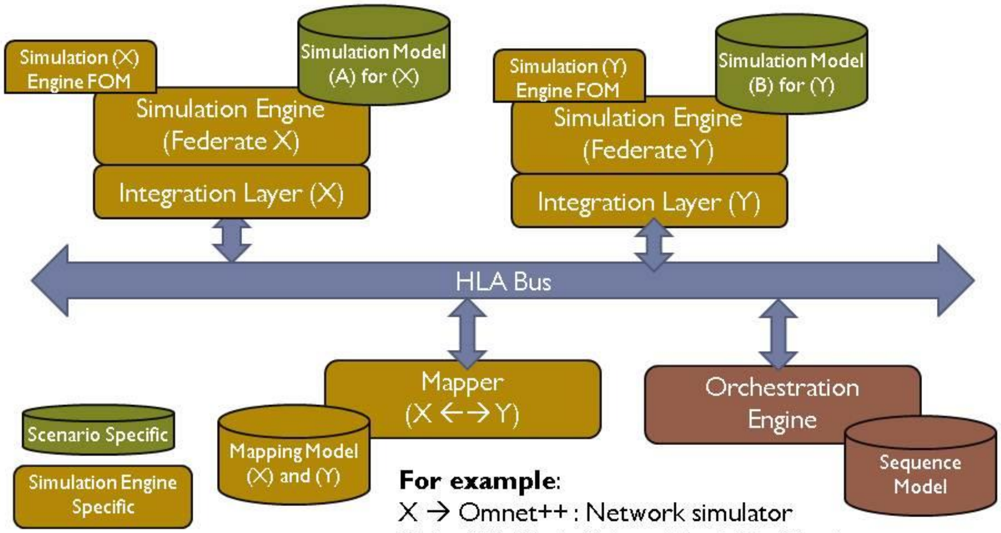


Notional Architecture

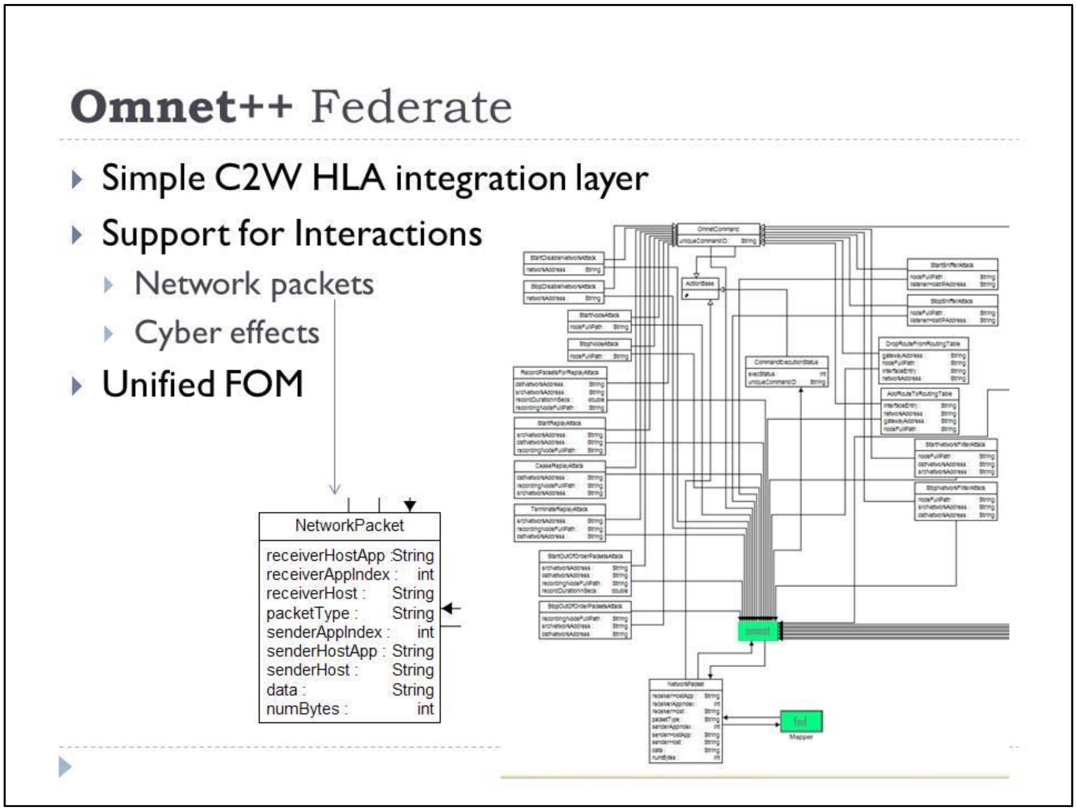
Next Generation C2WT

- *Uniform architecture* that allows using any simulation engine once it is integrated
  - A simulation = Simulation Engine + Simulation Model
    - Example: CPN Engine + Pacifica scenario model
  - Once the engine is integrated → no custom (model-specific) code development (the simulation model may have to adjusted)
  - Fixed Federation Object Model (FOMs) for simulation engines
  - ***Mapper*** federates translate messages between simulation engines
    - Mapper is generated from models that describe how to map the messages of one simulator into the messages of another one
- CASIM Design Language models *operational sequences* derived from COA
  - Sequencing and timing of interactions that are injected into the federation
  - Sequences are executed by the orchestration engine
  - Response messages the in COA 'orchestration engine' waits for during execution

Software Infrastructure and Platform

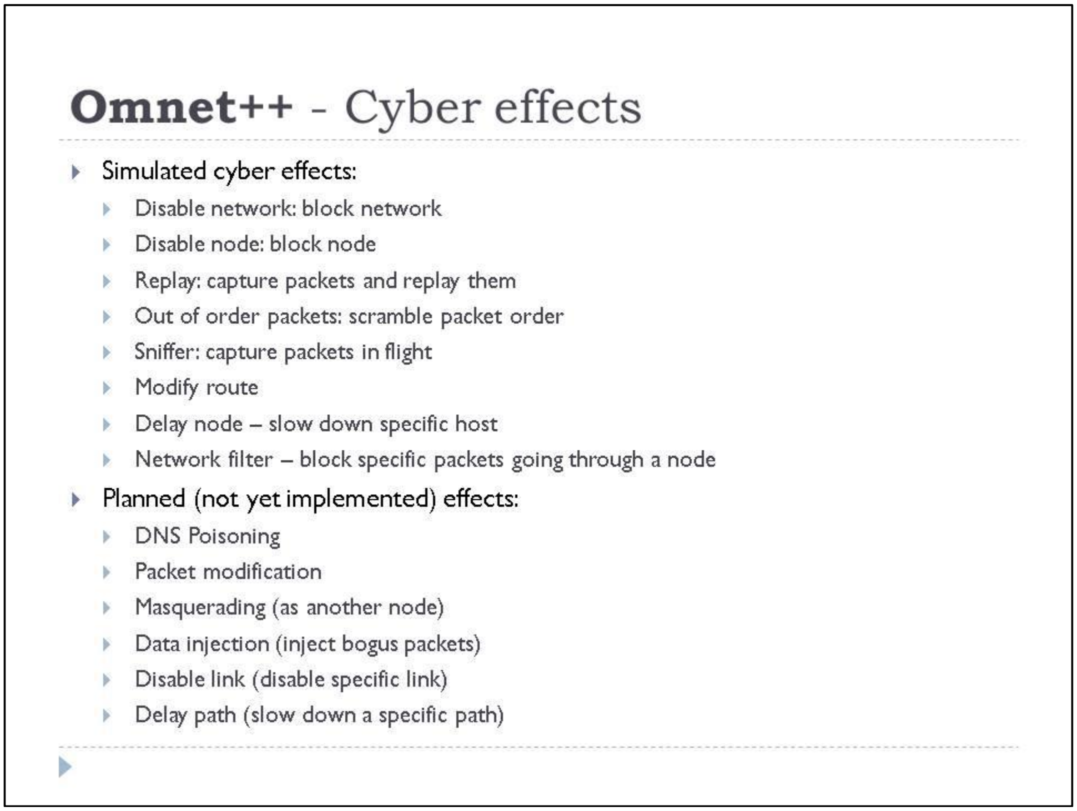

Simulation (X) Engine FOM
Simulation Model (A) for (X)
Simulation Engine (Federate X)
Integration Layer (X)
Simulation (Y) Engine FOM
Simulation Model (B) for (Y)
Simulation Engine (Federate Y)
Integration Layer (Y)
HLA Bus
Mapper (X ←→ Y)
Orchestration Engine
Scenario Specific
Simulation Engine Specific
Mapping Model (X) and (Y)
Sequence Model

For example:
X → Omnet++ : Network simulator
Y → CPN Tool: Colored Petri Net Simulator



Omnet++ Federate

- Simple C2W HLA integration layer
- Support for Interactions
  - Network packets
  - Cyber effects
- Unified FOM



Omnet++ - Cyber effects

- Simulated cyber effects:
  - Disable network: block network
  - Disable node: block node
  - Replay: capture packets and replay them
  - Out of order packets: scramble packet order
  - Sniffer: capture packets in flight
  - Modify route
  - Delay node – slow down specific host
  - Network filter – block specific packets going through a node
- Planned (not yet implemented) effects:
  - DNS Poisoning
  - Packet modification
  - Masquerading (as another node)
  - Data injection (inject bogus packets)
  - Disable link (disable specific link)
  - Delay path (slow down a specific path)

CPN Federate

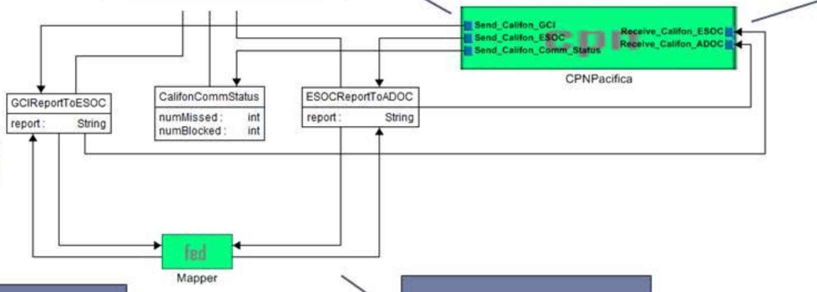
- No predefined FOM for the CPN federate
  - The CPN FOM is simulation model dependent
  - The integration layer is reusable + partially generated from the GME model
- Integration steps:
  - Develop CPN simulation model
  - Identify input and output places ('ports') in the model where tokens 'enter' and 'leave' the simulation model
  - Define interaction models that match the token 'colors' for those places
  - Import the CPN simulation model into a modeling tool
  - Wire the ports to interaction models

CPN Federate

- Example: (from simplified Pacifica scenario)

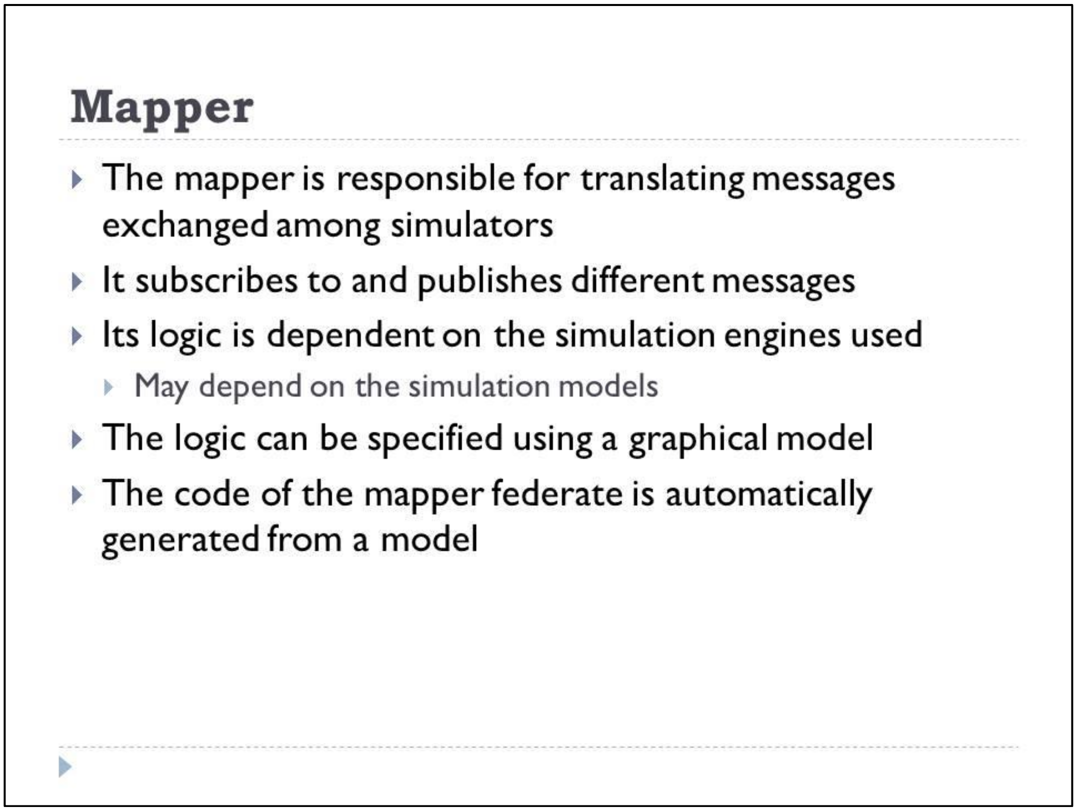
Output places:
•GCI sends report
•ESOC sends report

Input places:
•ESOC receives report
•ADOC receives report



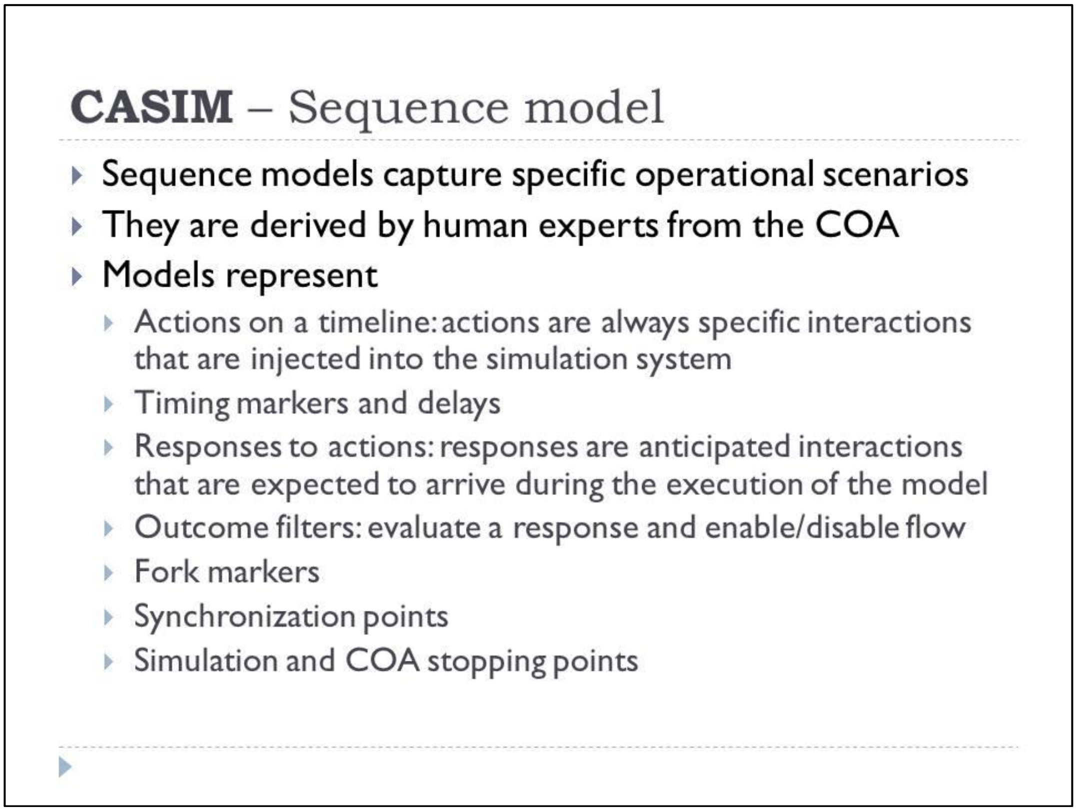
GCI Reports to ESOC
Content: report text

ESOC Reports to ADOC
Content: report text



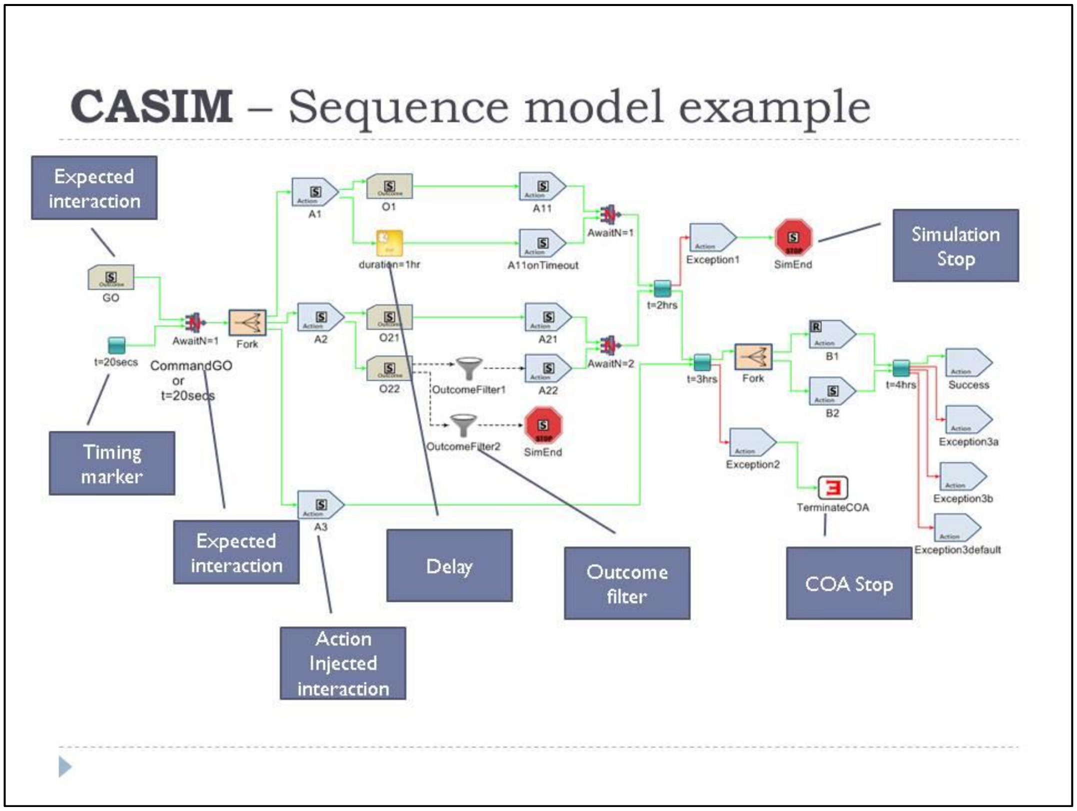
Mapper

- The mapper is responsible for translating messages exchanged among simulators
- It subscribes to and publishes different messages
- Its logic is dependent on the simulation engines used
  - May depend on the simulation models
- The logic can be specified using a graphical model
- The code of the mapper federate is automatically generated from a model



CASIM – Sequence model

- Sequence models capture specific operational scenarios
- They are derived by human experts from the COA
- Models represent
  - Actions on a timeline: actions are always specific interactions that are injected into the simulation system
  - Timing markers and delays
  - Responses to actions: responses are anticipated interactions that are expected to arrive during the execution of the model
  - Outcome filters: evaluate a response and enable/disable flow
  - Fork markers
  - Synchronization points
  - Simulation and COA stopping points



CASIM – Sequence model example

Expected interaction

Timing marker

Expected interaction

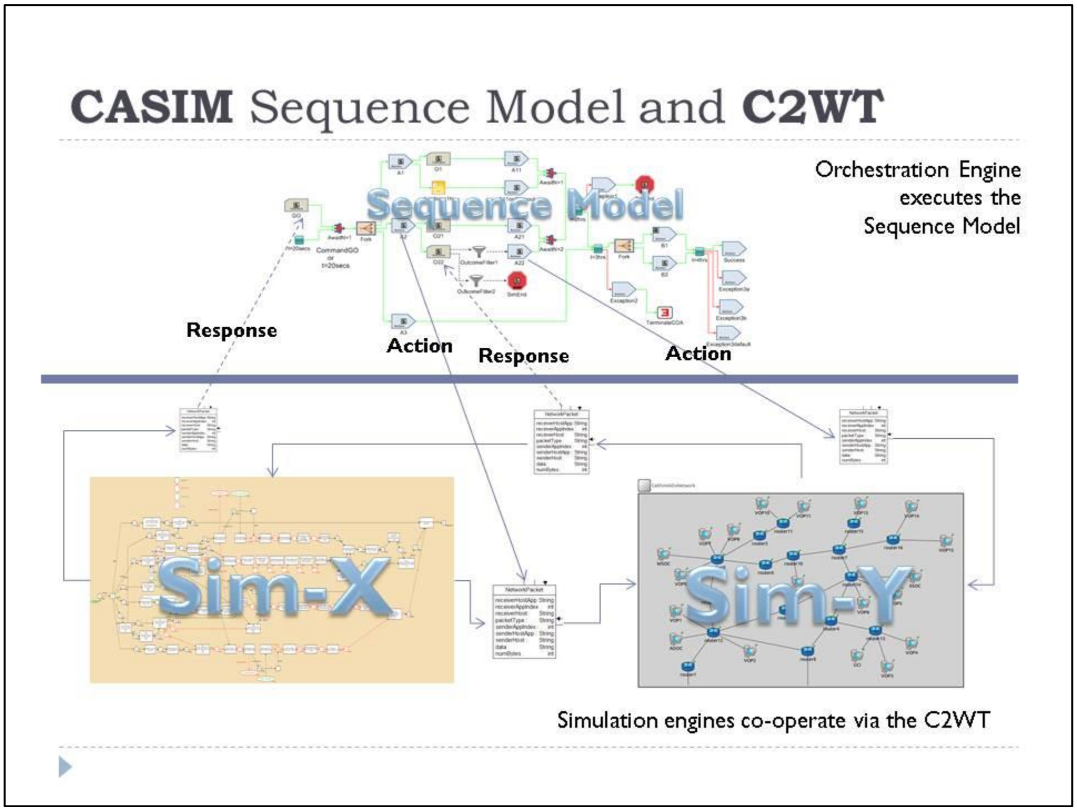
Action Injected interaction

Delay

Outcome filter

COA Stop

Simulation Stop



CASIM Sequence Model and **C2WT**

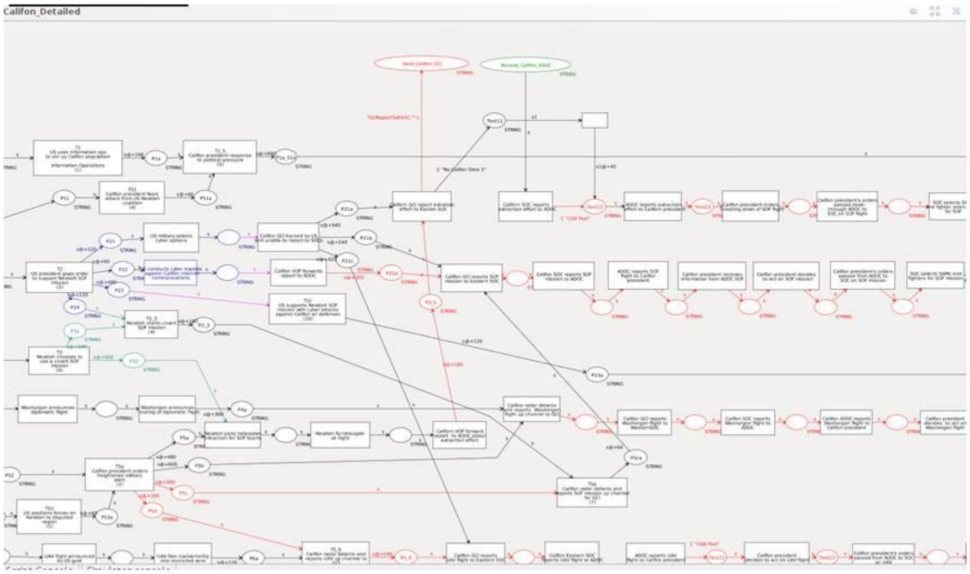
Orchestration Engine executes the Sequence Model

Simulation engines co-operate via the C2WT

Integration example: Pacifica scenario

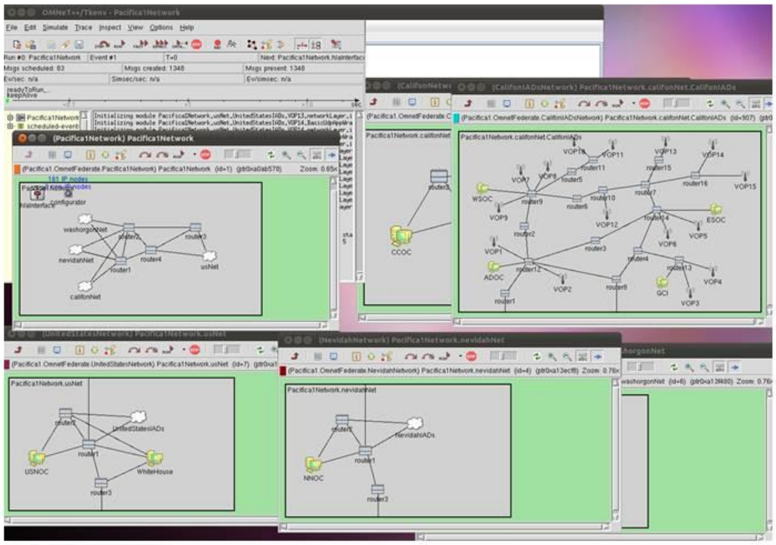
- CPN model:
  - Organizations and decision making processes
- Omnet++ model:
  - 3+1 national networks+ links between
- Integration model
  - CPN (+) Omnet++
- Mapper: mapping 'report' messages to network packets and vice versa
  - Generated from a mapping model
- COA: cyber effects on specific node(s) in a network

CPN Model – Events of the scenario


Califon_Detailed
STRING
US uses information ops to stir up Califon population Information Operations
P1a
P51
P51a
P21a
P21b
P21c
P22
P23
P24
P31
P32
Test11
US military selects cyber options
US president gives order to support Nevaleh SOF mission
Califon SOF forwards report to ADOC
Califon GCI reports SOF mission to Eastern SOC

Network model

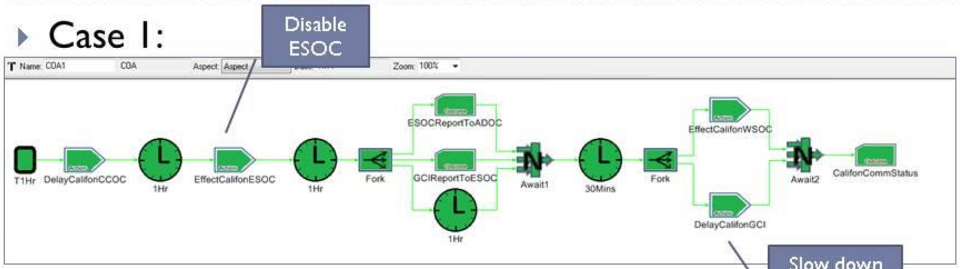
- 3+1 national networks+ links between


OMNeT++/Tkenv - Pacifica1Network
(Pacifica1Network) Pacifica1Network
washorgonNet
nevidahNet
californNet
usNet
router1
router3
router4
configurator
(CalifonIADsNetwork) Pacifica1Network.califonNet.CalifonIADs
WSOC
ESOC
ADOC
GCI
CCOC
VOP1
VOP2
VOP3
VOP4
VOP5
VOP6
VOP11
VOP12
VOP13
VOP14
VOP15
(UnitedStatesNetwork) Pacifica1Network.usNet
UnitedStatesIADs
USNOC
WhiteHouse
(NevidahNetwork) Pacifica1Network.nevidahNet
NevidahIADs
NNOC

Example: Sequence models

- Case 1:

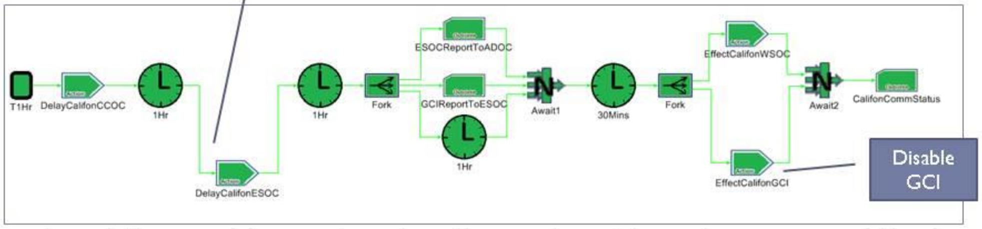
Disable ESOC



Slow down GCI

- Case 2:

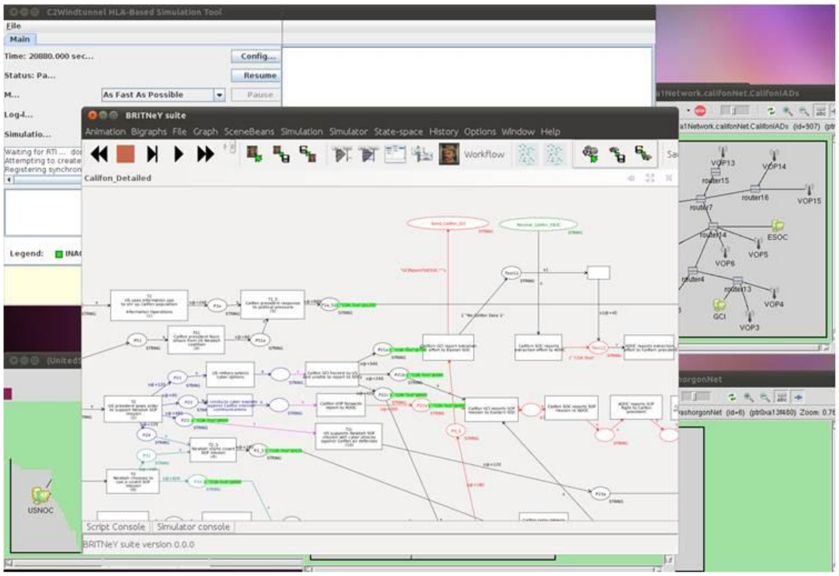
Slow down ESOC



Disable GCI

- *Caveat:* These models are *only* to show functionality of the tool – were not validated with the scenario.

Network + CPN model executing in sync



C2Windtunnel HLA-Based Simulation Tool

Time: 20880.000 sec...

Status: Pa...

M... As Fast As Possible

Log-l...

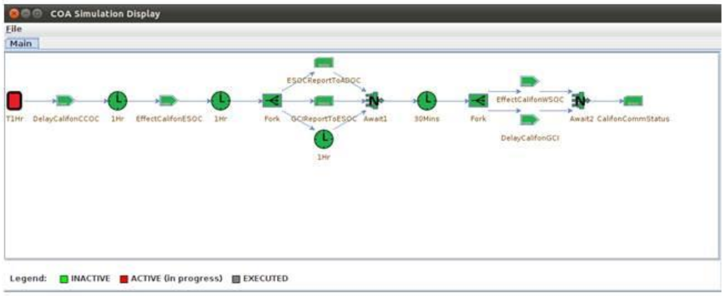
Simulatio...

Waiting for RTI ... do
Attempting to create
Registering synchron

Legend: 🟩 INA

Case-1 Sequence model execution

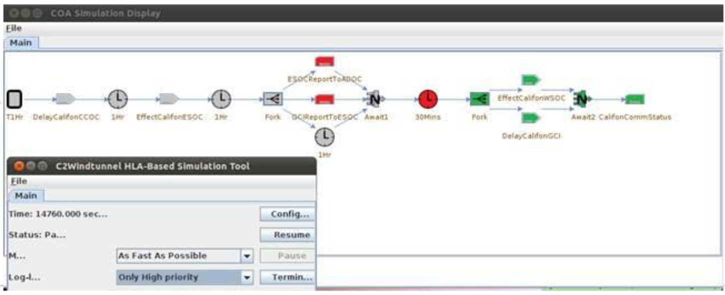
Initial state:



COA Simulation Display

Legend: 🟩 INACTIVE 🟥 ACTIVE (in progress) ▨ EXECUTED

Executing



COA Simulation Display

C2Windtunnel HLA-Based Simulation Tool

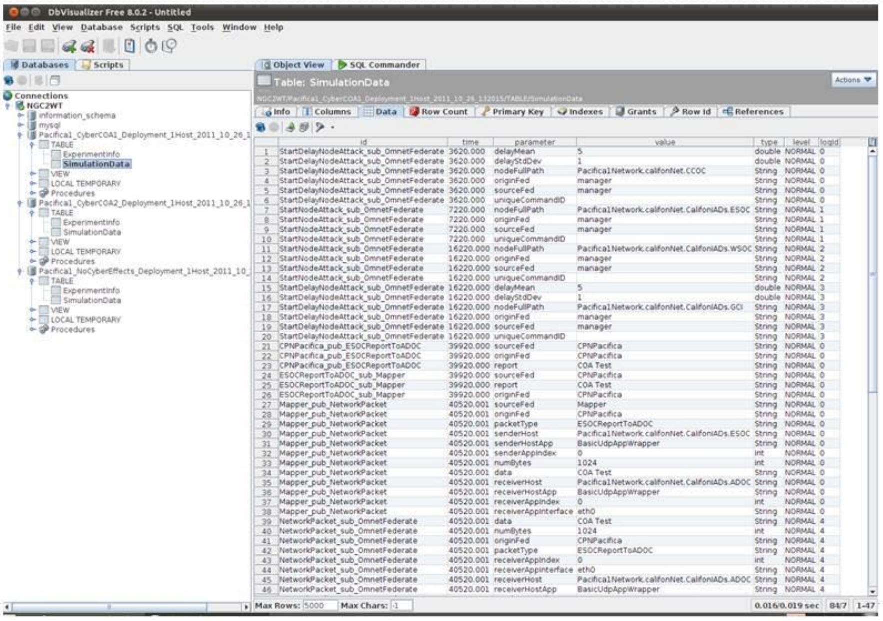
Time: 14760.000 sec...

Status: Pa...

M... As Fast As Possible

Log-l... Only High priority

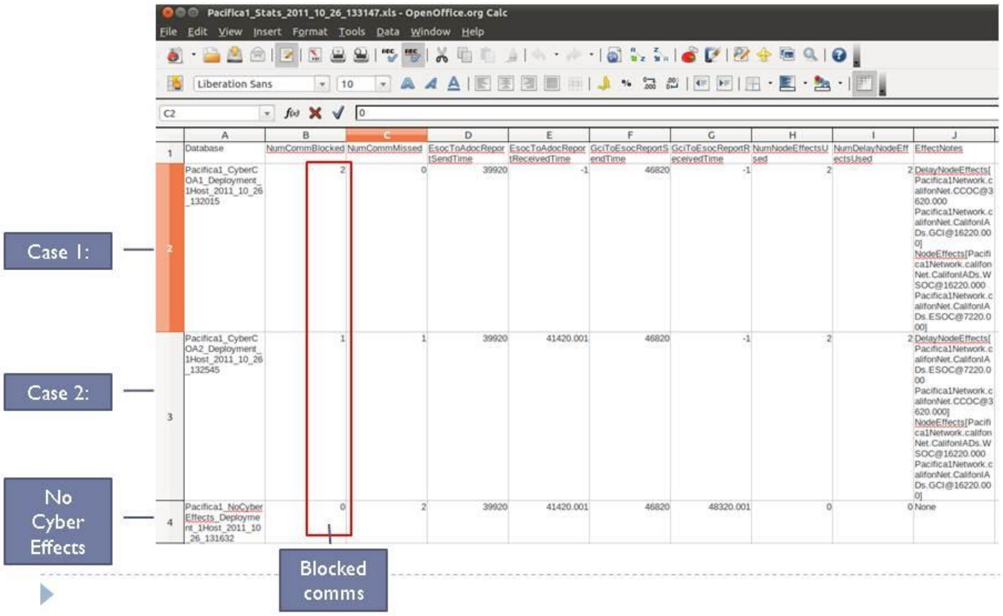
Raw results – Message logs



	id	time	parameter	value	type	level	logId
1	StartDelayNodeAttack_sub_OmnetFederate	3620.000	delayMean	5	double	NORMAL	0
2	StartDelayNodeAttack_sub_OmnetFederate	3620.000	delayStdDev	1	double	NORMAL	0
3	StartDelayNodeAttack_sub_OmnetFederate	3620.000	nodeFullPath	Pacifical Network.califonNet.CCOC	String	NORMAL	0
4	StartDelayNodeAttack_sub_OmnetFederate	3620.000	originFed	manager	String	NORMAL	0
5	StartDelayNodeAttack_sub_OmnetFederate	3620.000	sourceFed	manager	String	NORMAL	0
6	StartDelayNodeAttack_sub_OmnetFederate	3620.000	uniqueCommandID		String	NORMAL	0
7	StartNodeAttack_sub_OmnetFederate	7220.000	nodeFullPath	Pacifical Network.califonNet.CalifonIADs.ESOC	String	NORMAL	1
8	StartNodeAttack_sub_OmnetFederate	7220.000	originFed	manager	String	NORMAL	1
9	StartNodeAttack_sub_OmnetFederate	7220.000	sourceFed	manager	String	NORMAL	1
10	StartNodeAttack_sub_OmnetFederate	7220.000	uniqueCommandID		String	NORMAL	1
11	StartNodeAttack_sub_OmnetFederate	16220.000	nodeFullPath	Pacifical Network.califonNet.CalifonIADs.WSOC	String	NORMAL	2
12	StartNodeAttack_sub_OmnetFederate	16220.000	originFed	manager	String	NORMAL	2
13	StartNodeAttack_sub_OmnetFederate	16220.000	sourceFed	manager	String	NORMAL	2
14	StartNodeAttack_sub_OmnetFederate	16220.000	uniqueCommandID		String	NORMAL	2
15	StartDelayNodeAttack_sub_OmnetFederate	16220.000	delayMean	5	double	NORMAL	3
16	StartDelayNodeAttack_sub_OmnetFederate	16220.000	delayStdDev	1	double	NORMAL	3
17	StartDelayNodeAttack_sub_OmnetFederate	16220.000	nodeFullPath	Pacifical Network.califonNet.CalifonIADs.GCI	String	NORMAL	3
18	StartDelayNodeAttack_sub_OmnetFederate	16220.000	originFed	manager	String	NORMAL	3
19	StartDelayNodeAttack_sub_OmnetFederate	16220.000	sourceFed	manager	String	NORMAL	3
20	StartDelayNodeAttack_sub_OmnetFederate	16220.000	uniqueCommandID		String	NORMAL	3
21	CPNPacifica_pub_ESOCReportToADOC	39920.000	sourceFed	CPNPacifica	String	NORMAL	0
22	CPNPacifica_pub_ESOCReportToADOC	39920.000	originFed	CPNPacifica	String	NORMAL	0
23	CPNPacifica_pub_ESOCReportToADOC	39920.000	report	COA Test	String	NORMAL	0
24	ESOCReportToADOC_sub_Mapper	39920.000	sourceFed	CPNPacifica	String	NORMAL	0
25	ESOCReportToADOC_sub_Mapper	39920.000	report	COA Test	String	NORMAL	0
26	ESOCReportToADOC_sub_Mapper	39920.000	originFed	CPNPacifica	String	NORMAL	0
27	Mapper_pub_NetworkPacket	40520.001	sourceFed	Mapper	String	NORMAL	0
28	Mapper_pub_NetworkPacket	40520.001	originFed	CPNPacifica	String	NORMAL	0
29	Mapper_pub_NetworkPacket	40520.001	packetType	ESOCReportToADOC	String	NORMAL	0
30	Mapper_pub_NetworkPacket	40520.001	senderHost	Pacifical Network.califonNet.CalifonIADs.ESOC	String	NORMAL	0
31	Mapper_pub_NetworkPacket	40520.001	senderHostApp	BasicUdpAppWrapper	String	NORMAL	0
32	Mapper_pub_NetworkPacket	40520.001	senderAppIndex	0	int	NORMAL	0
33	Mapper_pub_NetworkPacket	40520.001	numBytes	1024	int	NORMAL	0
34	Mapper_pub_NetworkPacket	40520.001	data	COA Test	String	NORMAL	0
35	Mapper_pub_NetworkPacket	40520.001	receiverHost	Pacifical Network.califonNet.CalifonIADs.ADOC	String	NORMAL	0
36	Mapper_pub_NetworkPacket	40520.001	receiverHostApp	BasicUdpAppWrapper	String	NORMAL	0
37	Mapper_pub_NetworkPacket	40520.001	receiverAppIndex	0	int	NORMAL	0
38	Mapper_pub_NetworkPacket	40520.001	receiverAppInterface	eth0	String	NORMAL	0
39	NetworkPacket_sub_OmnetFederate	40520.001	data	COA Test	String	NORMAL	4
40	NetworkPacket_sub_OmnetFederate	40520.001	numBytes	1024	int	NORMAL	4
41	NetworkPacket_sub_OmnetFederate	40520.001	originFed	CPNPacifica	String	NORMAL	4
42	NetworkPacket_sub_OmnetFederate	40520.001	packetType	ESOCReportToADOC	String	NORMAL	4
43	NetworkPacket_sub_OmnetFederate	40520.001	receiverAppIndex	0	int	NORMAL	4
44	NetworkPacket_sub_OmnetFederate	40520.001	receiverAppInterface	eth0	String	NORMAL	4
45	NetworkPacket_sub_OmnetFederate	40520.001	receiverHost	Pacifical Network.califonNet.CalifonIADs.ADOC	String	NORMAL	4
46	NetworkPacket_sub_OmnetFederate	40520.001	receiverHostApp	BasicUdpAppWrapper	String	NORMAL	4

Max Rows: 5000 Max Chars: -1 0.016/0.019 sec 84/7 1-47

Processed results

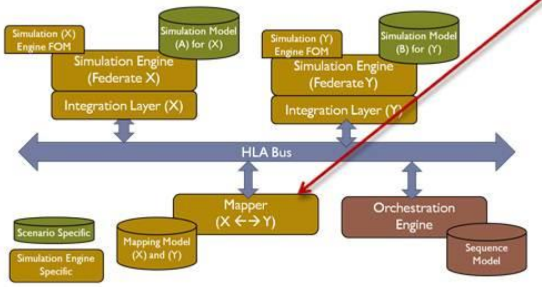


Pacifica1_Stats_2011_10_26_133147.xls - OpenOffice.org Calc
File Edit View Insert Format Tools Data Window Help
Liberation Sans
10
C2
0
Case 1:
Case 2:
No Cyber Effects
Blocked comms

	A	B	C	D	E	F	G	H	I	J
1	Database	NumCommBlocked	NumCommMissed	EsocToAdocReportSendTime	EsocToAdocReportReceivedTime	GciToEsocReportSendTime	GciToEsocReportReceivedTime	NumNodeEffectsUsed	NumDelayNodeEffectsUsed	EffectNotes
2	Pacifica1_CyberCOA1_Deployment_1Host_2011_10_26_132015	2	0	39920	-1	46820	-1	2	2	DelayNodeEffects[Pacifica1Network.califonNet.CCOC@3620.000 Pacifica1Network.califonNet.CalifonIADs.GCI@16220.000] NodeEffects[Pacifica1Network.califonNet.CalifonIADs.WSOC@16220.000 Pacifica1Network.califonNet.CalifonIADs.ESOC@7220.000]
3	Pacifica1_CyberCOA2_Deployment_1Host_2011_10_26_132545	1	1	39920	41420.001	46820	-1	2	2	DelayNodeEffects[Pacifica1Network.califonNet.CalifonIADs.ESOC@7220.000 Pacifica1Network.califonNet.CCOC@3620.000] NodeEffects[Pacifica1Network.califonNet.CalifonIADs.WSOC@16220.000 Pacifica1Network.califonNet.CalifonIADs.GCI@16220.000]
4	Pacifica1_NoCyberEffects_Deployment_1Host_2011_10_26_131632	0	2	39920	41420.001	46820	48320.001	0	0	None

Mapper: Addressing Integration Issues

- Assumption: Simulation models and their specific interfaces are defined
- *Rapid integration problem*: Composition of the specific simulation models and 'joining' their interfaces
  - Technical, but important detail: mapping between the interfaces


Simulation (X) Engine FOM
Simulation Model (A) for (X)
Simulation Engine (Federate X)
Integration Layer (X)
Simulation (Y) Engine FOM
Simulation Model (B) for (Y)
Simulation Engine (Federate Y)
Integration Layer (Y)
HLA Bus
Mapper (X ←→ Y)
Orchestration Engine
Scenario Specific
Simulation Engine Specific
Mapping Model (X) and (Y)
Sequence Model

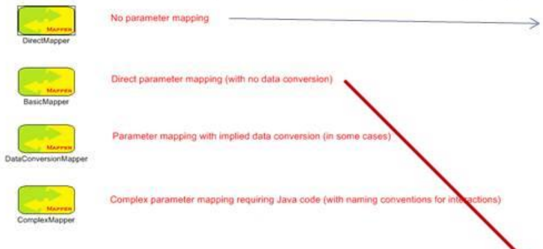
Preferably: no programming experience should be required for building a mapper

•Simple cases are easy, but for complex mappings (e.g. coordinate conversions) custom code is needed.

Mapper ~ message converter

Mapper: Modeling the mapper (1)

A Mapper (federate) consists of a set of *mappings* between interactions



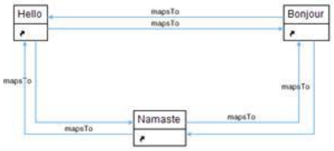
No parameter mapping

Direct parameter mapping (with no data conversion)

Parameter mapping with implied data conversion (in some cases)

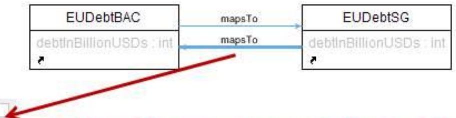
Complex parameter mapping requiring Java code (with naming conventions for interactions)

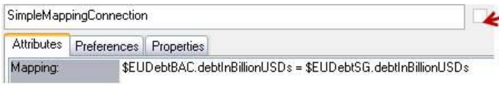
Case #1: Trivial mapping



No parameter mapping

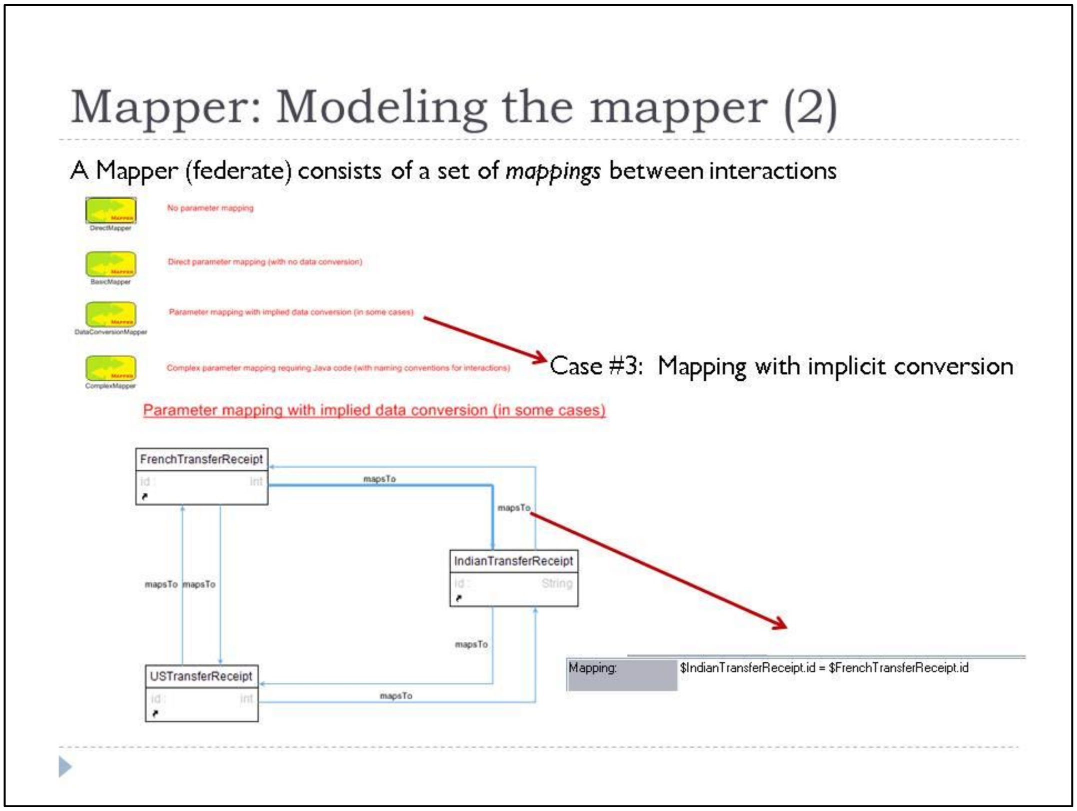
Case #2: Map the fields




SimpleMappingConnection

Mapping: $EUDebtBAC.debtInBillionUSDs = $EUDebtSG.debtInBillionUSDs

Direct parameter mapping (with no data conversion)



Mapper: Modeling the mapper (2)

A Mapper (federate) consists of a set of *mappings* between interactions

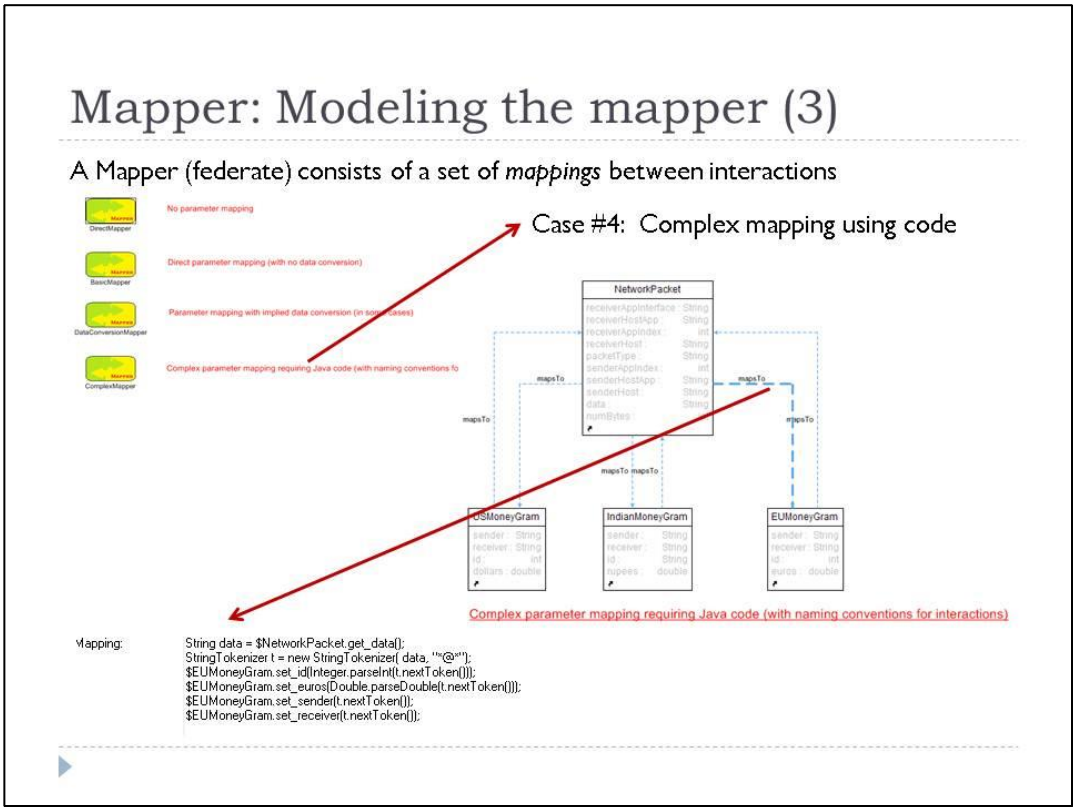
DirectMapper
No parameter mapping
BasicMapper
Direct parameter mapping (with no data conversion)
DataConversionMapper
Parameter mapping with implied data conversion (in some cases)
ComplexMapper
Complex parameter mapping requiring Java code (with naming conventions for interactions)

Case #3: Mapping with implicit conversion

Parameter mapping with implied data conversion (in some cases)

FrenchTransferReceipt
id : int
IndianTransferReceipt
id : String
USTransferReceipt
id : int
mapsTo
mapsTo
mapsTo
mapsTo
mapsTo
mapsTo

Mapping: $IndianTransferReceipt.id = $FrenchTransferReceipt.id



Mapper: Modeling the mapper (3)

A Mapper (federate) consists of a set of *mappings* between interactions

DirectMapper
No parameter mapping
BasicMapper
Direct parameter mapping (with no data conversion)
DataConversionMapper
Parameter mapping with implied data conversion (in some cases)
ComplexMapper
Complex parameter mapping requiring Java code (with naming conventions fo

Case #4: Complex mapping using code

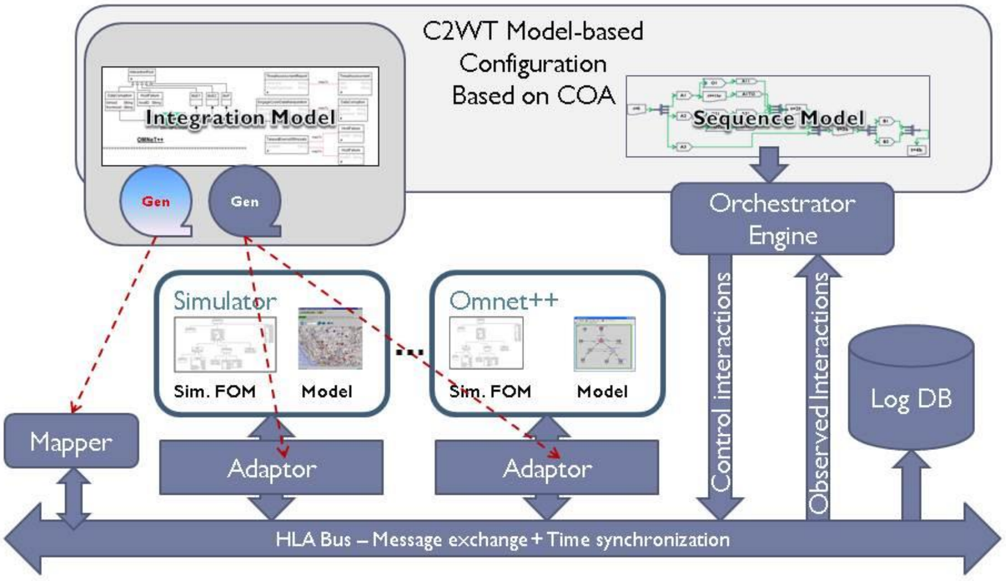
Complex parameter mapping requiring Java code (with naming conventions for interactions)

Mapping:

```
String data = $NetworkPacket.get_data();
StringTokenizer t = new StringTokenizer( data, "*@*");
$EUMoneyGram.set_id(Integer.parseInt(t.nextToken()));
$EUMoneyGram.set_euros(Double.parseDouble(t.nextToken()));
$EUMoneyGram.set_sender(t.nextToken());
$EUMoneyGram.set_receiver(t.nextToken());
```

Summary

Architecture Details - Recap



LIST OF ACRONYMS, ABBREVIATIONS, AND SYMBOLS

ADOC	Air Defense Operations Center
AFOSR	Air Force Office of Scientific Research
AFRL	Air Force Research Laboratory
BM aircraft	Bartel Maryla aircraft
BPMN	Business Process Modeling Notation
CASIM	COA Analysis Simulation
C2	Command and Control
C2WT	Command and Control Wind Tunnel
C4I	Command, Control, Communications, Computers and Intelligence
CC-C2	Counter-Counter Command and Control
CC-Cyber	Counter-Counter Cyber
CCOC	Califon Cyber Ops Center
COA	Course of Action
COC	Cyber Ops Center
CPN	Colored Petri Net
Cyber-IO	Cyber Information Operations
DB	Database
DNS	Domain Name Service
DoD	Department of Defense
DOS	Denial of Service
ELINT	Electronic Intelligence
EW	Electronic Warfare
FOM	Federation Object Model
GB	Giga Bytes
GCI	Ground Control Intercept
GME	Generic Modeling Environment
GMU	George Mason University
GUI	Graphical User Interface
HLA	High-Level Architecture
IADS	Integrated Air Defense System
IDE	Integrated Development Environment
IO	Information Operations
IP	Intetnet Protocol
ISIS	Institute for Software Integrated Systems
ISR	Intelligence, Surveillance, and Reconnaissance
JDK	Java Development Kit
JDT	Java Development Tools
JNI	Java Native Interface
JRE	Java Runtime Environment
MB	Mega Bytes
Mil-Dec	Military Deception
MoE	Measures of Effectiveness
NAT	Network Address Translation
NGC2WT	Next Generation Command and Control Wind Tunnel

NNOC	Nevidah Network Operations Center
NW	Network Warfare
OCL	Object Constraint Language
OE	Orchestration Engine
OS	Operating System
OSM	Operational Sequence Model
PRET	Partnership for Research Excellence and Transition
RPV	Remotely Piloted Vehicle
SAM	Surface-to-Air Missile
SCADA	Supervisory Control And Data Acquisition
SOC	Sector Operations Center
SOF	Special Operations Forces
SOM	Simulation Object Model
SQL	Structured Query Language
SVN (svn)	Subversion
TIN	Timed Influence Net
TTPs	Tactics, Techniques, and Procedures
UAV	Unmanned Aerial Vehicle
UC	University of California
VM	Virtual Machine
VOP	Visual Observation Point
WINE	Windows Emulator
XML	Extensible Markup Language

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

1. REPORT DATE *(DD-MM-YYYY)* 16-01-2013	**2. REPORT TYPE** Final Technical Report	**3. DATES COVERED** *(From - To)* 11-22-2010 11-21-2012

4. TITLE AND SUBTITLE
Next Generation C2WT For COA Simulation

5a. CONTRACT NUMBER
FA8750-11-2-0078

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Gabor Karsai
Alexander Levis
Himanshu Neema

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Vanderbilt University
110 21ST AVENUE S STE 937, NASHVILLE TN 37203-2416
George Mason University
4400 University Drive, MSN 4CG, Fairfax, VA 22030-4422

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
USAF, AFMC
AIR FORCE RESEARCH LABORATORY
26 ELECTRONIC PARKWAY
ROME NY 13441-4514

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RIKF

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

12. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT The objective of the CASIM program was to develop an integrated, composable modeling and simulation environment to create realistic cyber warfare operational scenarios for course of action (COA) exploration. The project has built such using a new generation of the “C2 Wind Tunnel” software platform that consists of (1) a modeling environment to compose heterogeneous simulations, and (2) a model-driven run-time framework for the coordinated and controlled execution of integrated simulations. The platform introduces novel capabilities to enhance usability including reusable simulation interfaces; a generic ‘mapper’ for the syntactical translation of messages between simulators, and an orchestration engine that executes operational sequence models. The project has evaluated the framework through a set of hypothetical scenarios from the open-source Pacifica domain.

15. SUBJECT TERMS
Modeling and simulation, COA analysis, integration of heterogeneous simulations, workflow model

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Gabor Karsai
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	U	138	**19b. TELEPONE NUMBER (*Include area code*)** +1 (615) 343 771