Georgia Tech Research Corporation

Georgia Institute of Technology
Office of Sponsored Programs
Atlanta, Georgia 30332-0420 U.S.A.

January 4, 2010
In reply refer to: 112048

Director, Naval Research Lab
Attn: Code 5596
4555 Overlook Avenue, SW
Washington , DC 20375-5320

Subject: Progress Report
Project Director(s): Mavris Dimitri
Telephone No.: 404-894-1557
Contract No.: N00014-09-C-0394
Prime No: N/A
"INTEGRATED RECONFIGURABLE INTELLIGENT SYSTEMS (IRIS) FOR COMPEX NAVAL...
Period Covered: 10/1/2009 – 12/31/2009

The subject report is forwarded in conformance with the contract/grant specifications.

Should you have any questions or comments regarding this report(s), please contact the Project Director.

Sincerely,

Kamie Cunningham
Date Entry Specialist

Addressee: 1 copy
DTIC 2 copies

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS

1. REPORT DATE (DD-MM-YYY)	2. REPORT TYPE	3. DATES COVERED *(From – To)*
31-December-2009	Quarterly Technical & Cost Expenditure Status Report	01-Oct-2009 to 31-Dec-2009

4. TITLE AND SUBTITLE

Integrated Reconfigurable Intelligent Systems (IRIS) for Complex Naval Systems

5a. CONTRACT NUMBER

N00014-09-C-0394

5b. GRANT NUMBER

N/A

5c. PROGRAM ELEMENT NUMBER

N/A

6. Author(s)

Dr. Dimitri N. Mavris

Dr. Yongchang Li

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Georgia Institute of Technology
School of Aerospace Engineer
Atlanta, GA 30332-0150

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
875 North Randolph Street
Arlington, VA 22203-1995

10. SPONSOR/MONITOR'S ACRONYM(S)

ONR

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

N/A

12. DISTRIBUTION AVAILABILITY STATEMENT

Unlimited Distribution

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The report details the progress that has been made by ASDL in developing and applying the IRIS concept for the period of October 1 to December 31, 2009. The team worked on refining the UML diagrams created. In addition, two initial Paramrine configurations for a ship product model (PM) have been developed; integration structure of the M&S environment was improved; high level controller has been implemented and integrated for resource allocation; inference capability of the mid level controller was tested; graph-based model of the notional-YP cooling system was constructed and the reference damage controller was tested; the model of human in the loop control was improved and documented, and the script for integration was developed; further study on theoretical framework for survivability design was performed.

15. SUBJECT TERMS

Modeling & Simulation, Reconfigurability, Integrated & Intelligent, Naval Systems

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Mr. Anthony J. Seman
U	U	U	SAR	40	19b. TELEPHONE NUMBER *(include area code)* 703-696-5992

QUARTERLY REPORT

(October 1, 2009 – December 31, 2009)

"Integrated Reconfigurable Intelligent Systems (IRIS) for Complex Naval Systems"

Contract #: N00014-09-C-0394

SUBMITTED TO:
Mr. Anthony J. Seman, Office of Naval Research
(email: Anthony_Seman@onr.navy.mil)

Office of Naval Research
875 North Randolph Street
Arlington, VA 22203-1995

SUBMITTED BY:
Georgia Institute of Technology
School of Aerospace Engineering
Aerospace Systems Design Laboratory (ASDL)
Atlanta, Georgia 30332-0150

AWARD PERIOD:
July 1, 2009 to February 21, 2009

December 31, 2009

Principal Investigator:
Professor Dimitri N. Mavris
Director
Aerospace Systems Design Laboratory
School of Aerospace Engineering
Georgia Institute of Technology
Phone: (404) 894-1557
dimitri.mavris@ae.gatech.edu

Summary

The following report details the progress that has been made by ASDL in developing and applying the IRIS concept for the period of October 1 to December 31, 2009. The team worked on refining the UML diagrams created and attempted to integrate the diagrams to represent the new design process for intelligent systems. In addition, progress is made on individual tasks: two initial Paramrine configurations for a ship product model (PM) have been developed; integration structure of the Modeling and Simulation (M&S) environment was improved and model validation was conducted; high level controller has been implemented and integrated for resource allocation; inference capability of the mid level controller was tested by studying a various of scenarios; graph-based model of the notional-YP cooling system was constructed and the reference damage controller was developed and tested; the model of human in the loop control was improved and documented, and the script for integrating with other models was developed; further study on theoretical framework for survivability design was performed, a baseline naval architecture was developed and used to demonstrate the proposed methodology for survivability study.

Task 1: Design of Integrated Heterogeneous Systems

Subtask 1.1: Design Process Development Using System Engineering Approaches

Subtask 1.1.1: Method Development for Complex System Design

Introduction

United Modeling Language (UML) has been found useful in specifying, visualizing, constructing and documenting the work products of a software system and representing a business process. ASDL proposed to develop a new design process for intelligent systems, such as IRIS designed system with assess, predict, plan and execute functions, using UML. This is due to the fact that the use of UML can help designer to address all the requirements of an intelligent complex system. In addition, the use of UML can also provide the essential information for each design activity and modeling aspects required for developing tools for design purposes, such as identifying who will be involved in the design process and complete what design activities, what information/resources are needed for each design step, and what tool/methods are required in order to complete the design process.

Progress

In previous quarterly report, the preliminary UML diagrams created by the team was presented and described, including use case diagrams, activity diagrams and communication diagrams. Based on the developed diagrams, the team is working on refining the diagrams and makes them more consistent and comprehensive. Intensive discussions have been conducted on modifying and improving the original diagrams. In addition, the team is working on integrating the diagrams together to represent the entire design process. The integration is based on the use case diagram, as shown in Figure 1.

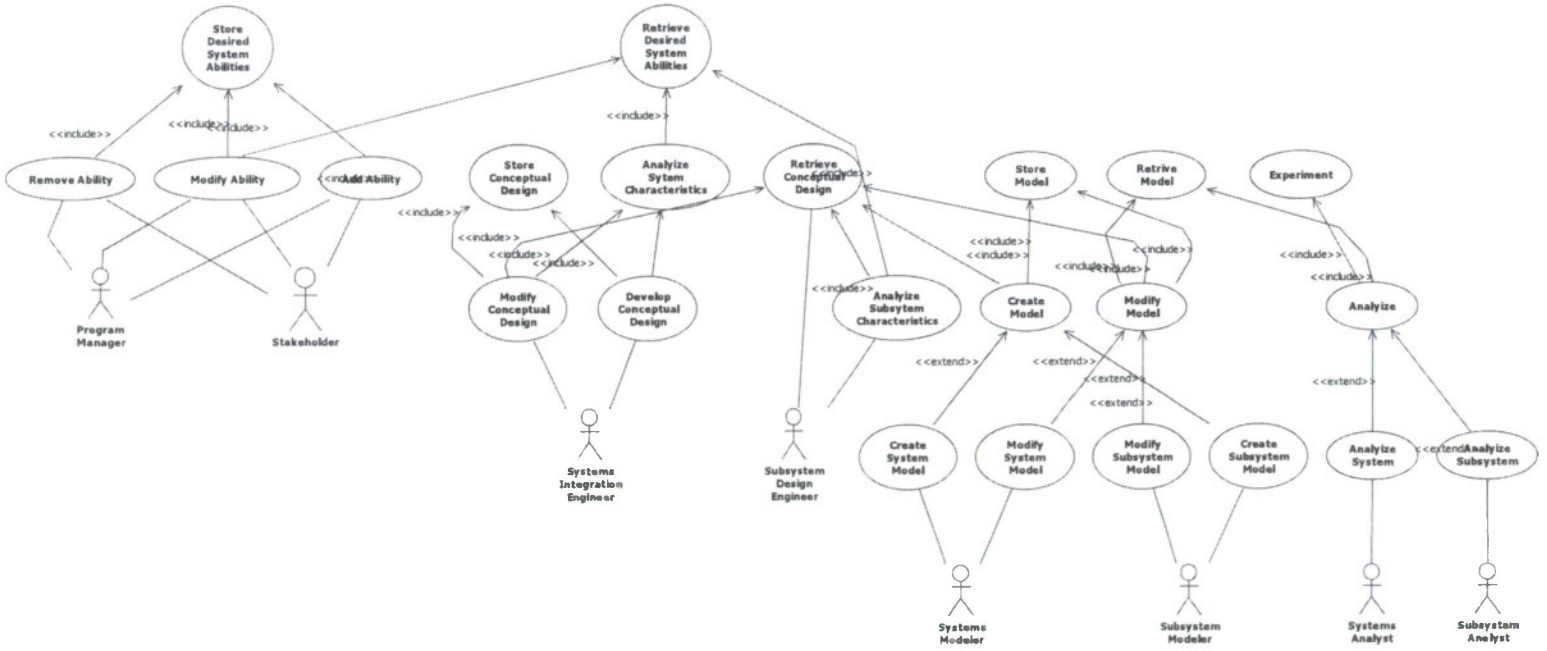


Figure 1: Use Case Diagram for Design Process

Furthermore, it is decided that a class diagram will be helpful for the designer to understand what objects (methods, tools) need to be identified in order to fully describe and represent the design process. This is an ongoing task and the solid results will be presented in the final report.

Future Work

Future work regarding this task will be to further improve and integrate the UML diagrams and develop new diagrams such as class diagram if it is necessary. Sequentially, the integrated diagrams will be used to describe and represent the new advanced design method for intelligent systems. With the employment of UML diagram, the new design process will help designer with identifying the key design requirements and activities, developing the necessary tools and methods that are required to complete each activity, and understanding the interactions among the design activities.

Subtask 1.1.2: Notional Ship Development

Introduction

A critical element for conducting survivability studies, as well as developing and testing the proposed design process, is a sizing and visualization environment. This environment is the ship geometry, including the architecture along with the internal subsystem distribution. Over the last three months, two different configurations for a ship product model (PM) have been developed, a Yard Patrol craft YP-679 and a naval destroyer DDG-51. Both notional ships would be the geometric baselines for a set of studies that will support the development of the survivability-based design method.

Progress

Starting from the initial vision for this subtask, the objective was to generate a computer geometry model of a notional ship, with a dynamic simulation environment of the ship engineering systems to be built around it for analysis of operations. The taxonomy of the subsystem components would be predefined and the architecture of the engineering systems would be similar, yet scalable to match the corresponding ship architecture in terms of size and mission requirements.

Meanwhile, Paramarine, a ship sizing and design tool has been recently acquired by ASDL. Besides its strong capability as a CAD visualization tool, Paramarine carries many possibilities for conducting various types of analysis related to naval architecting. The most common of them are stability analysis, ship weight estimation and sizing, system health monitoring, finite element analysis, etc. Given the analysis possibilities of Paramarine, a decision has been made to implement a ship baseline in this new environment.

Yard Patrol Craft YP-679

In order to for an IRIS demonstration model to be implemented, the original proposed idea was to create a baseline notional ship that would be heavily based on a YP-679 configuration. This direction has been proposed by ASDL and encouraged by the feedback advice given by ONR. Moreover, the available engineering system models were

sized for a YP-like configuration and therefore, choosing this baseline was very straightforward.

As mentioned in the previous quarter report, basic information around the geometry and the dimensions of the YP-679 was sparse. The only avenue for locating information around the YP geometry could only be found from publicly available resources (web search for reports, schematics and fact sheets). That info has been imported to the Paramarine PM as reference information and as starting point, given that even this information was not entirely complete. Notional information has been added where required information cannot be available. In Figure 2, a screenshot of the early stages of hull generation is given, where except from the centerline and the deck edge curve, all other lines were notional and had been adjusted to the available visual information on existing YP ships.

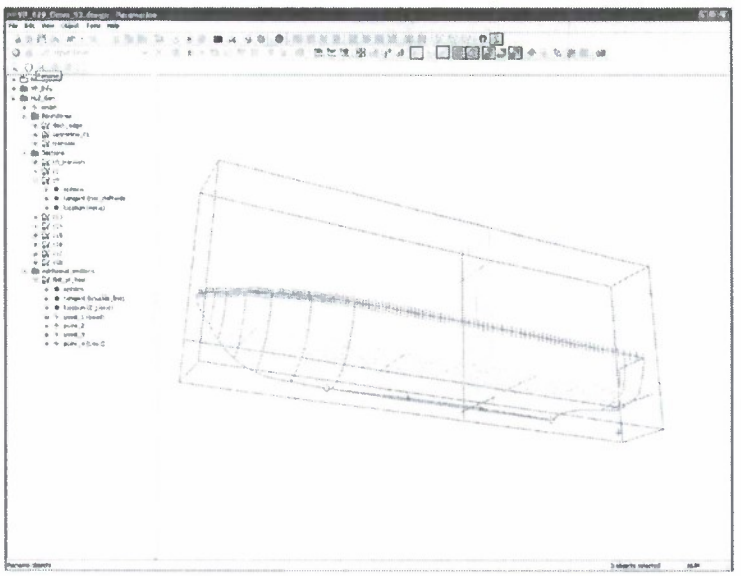


Figure 2: Early Stages of a Yard Patrol YP-679 Hull Generation in Paramarine

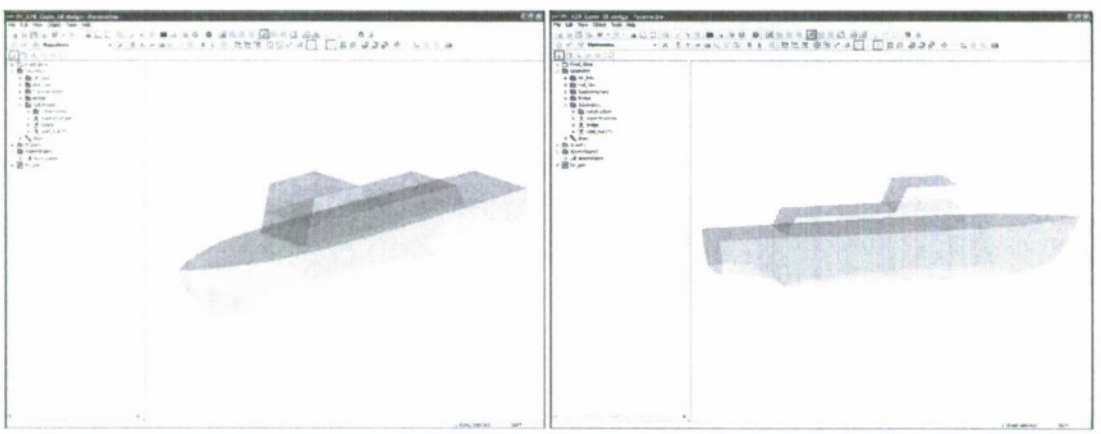


Figure 3: Final Design of a Yard Patrol YP-679 in Paramarine

The final external design of the YP ship is shown in Figure 3. While this design includes the hull, superstructure, the bridge and the internal compartmentation, the overall design will be finalized when the engineering systems representation (power generation and distribution, cooling, propulsion, etc.) is included.

Notional Naval Destroyer DDG-51

Despite the availability of the YP ship model, including its two alternative configurations that are currently under work, there have been some thoughts for a larger baseline ship architecture design. Regarding the task (Task 5) of developing a survivability-based design method, it appears that a small scale ship, such as the YP might not be sufficient for conducting adequate survivability studies. A larger architecture is expected to offer more meaningful results when running a typical damage scenario, with damage propagation extended throughout the ship to a certain extent, allowing for cases where the ship can still remain partially intact. There are definitely doubts that a YP architecture design might just suffer a total catastrophic failure from a single missile attack, given the fewer subsystem zones and limited available reconfigurability strategies for improving survival.

Thus, a decision had been made to initiate the development of a larger ship design, in order to work with a larger design space while improving the engineering systems design for reconfigurability. A notional destroyer ship design has been selected, heavily influenced by a DDG-51 ship design, and originally obtained by Anteon Corporation.

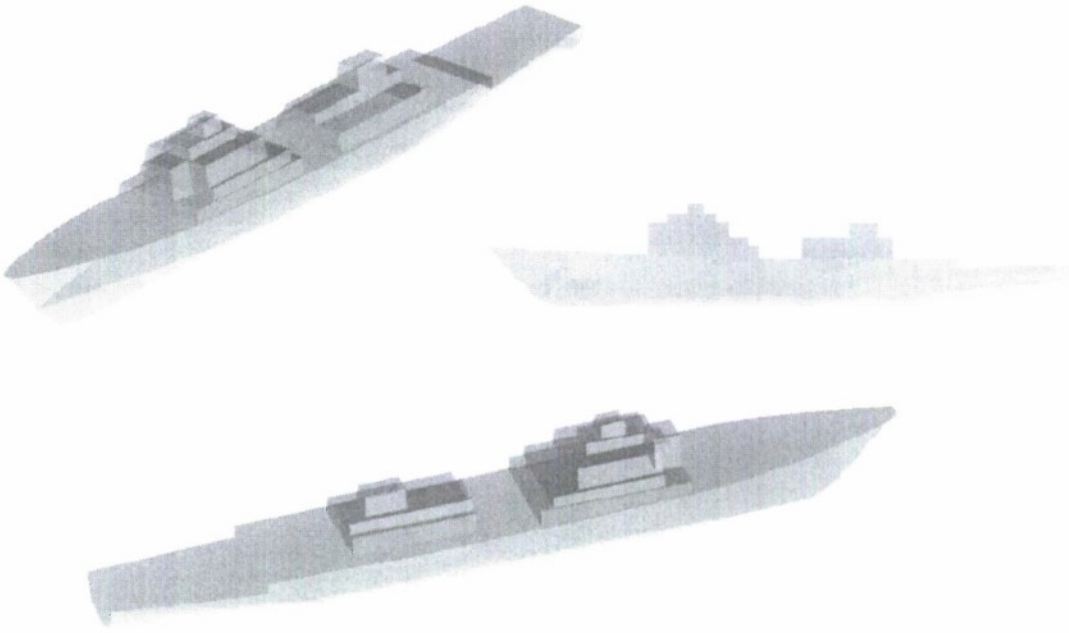


Figure 4: Finalized external design of the notional DDG-51 in Paramarine

Similarly to the progress status of the YP ship, the DDG-51 is complete in terms of its hull and external design, as shown in Figure 4. It follows the dimensions of the original

CAD designs that were made available to ASDL, yet other elements of the design are sourced from publicly available information regarding this particular type of destroyer.

Future Work

According to the latest technical feedback from ONR, the version of the YP ship that ONR will be using for their own in-house studies, will include an internal systems architecture based off the Tabletop systems simulator. On the other hand, ASDL's own version will use an alternative architecture, mainly based on a reduced and scaled down version of the RSAD cooling systems simulation and an in-house developed power system model. This final task on completing the YP ship model is currently active and is expected to be concluded within the first few weeks of 2010.

Regarding the notional DDG-51, there are tasks similar to the YP that are pending. Besides the subdivision, the engineering systems architecture has been recently decided and bound to be implemented in the near future. It is an extended IPS architecture, based on the DC electrical distribution system, as described by Fireman and Doerry. It includes five instead of four zones, with more AC and DC lads per zone. For more information on the systems architecture, please refer to section 5.2 regarding the modeling & simulation environment.

Subtask 1.2: Integration of heterogeneous dynamic systems

Introduction

Based on the previously described findings regarding co-simulation of third party proprietary dynamic sub-systems, the current work is now focused on two main objectives: verifying the execution order of the sub-models, and validating the simulation results. First, the co-simulation of the notional YP model must be completed. The sub-models exist, but some do not yet run as expected. The validation of the co-simulation results may not be possible as previously suggested. However, a method is currently programmed that can help to ensure that the simulation results stay within given error bounds, and takes corrective action if the error bounds are exceeded during simulation execution.

Progress

Co-simulation setup for notional YP simulation

One major issue that was found in the notional YP co-simulation setup was that the data exchange between the mid level agent based controller ("ABCtrl" in the model) and the low level controller ("metaVDCS11222009" in the model) was not timed properly. The issue was that in the previous simulation setup, all models were executed in parallel, before they were stopped and the data was exchanged between them. This lead to the situation that the mid level controller generated valve setting signals for the low level controller that corresponded to the current system state. However, due to the nature of the setup, the low level controller received these signals one time step later, when their validity was not necessarily given any more. The corrected setup now takes this issue into account and changes have been made to the initial setup to address this issue. The new

simulation setup allows the mid level controller to be executed first. Then the generated signals and outputs for the low level controller are fed into the low level controller, and the low level controller is executed after that. Only then are all data fed into the scheduler, exchanged as needed, and the next time step executed. This ensures that the low level controller received the correct data of the current time step from the mid level controller, and can react accordingly. It also ensures that the other sub-models, especially the mid level controller, have the correct values delivered at the new time step. Figure 5 shows the previous setup and connection of the co-simulation setup, and Figure 6 depicts the new setup with the corrected execution scheme. Please note that Figure 5 and Figure 6 represent notional setups to show how the sub models are linked within the co-simulation environment. In the actual simulation, the sub-models are not actually linked using ModelCenter's link editor, as shown, but rather using a scheduler script that takes care of model run schedule and data exchange. This has been described in earlier project reports, and allows for great freedom in the model execution schedule.

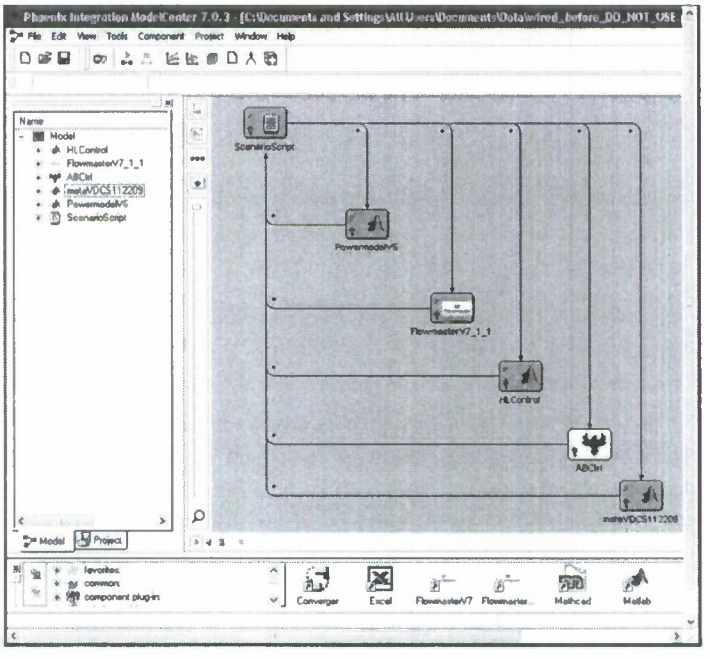


Figure 5: Initial setup of notional YP co-simulation

The co-simulation was also given a primitive variable output and visualization interface. This is an intermediate solution until the HMI (Human Machine Interface) is completed and integrated. The final HMI will allow for more sophisticated data visualization and user inputs into the simulation during run time. The interface presented here is based on simple Excel tables, graphs, and control objects. The first table allows for graphical output of time series data for given variables within the simulation.

Figure 7 shows a sample output, with service load temperatures, rupture information, and valve states as outputs. These outputs can be chosen freely, as can be the amount of

graphs for visualization. Since the table is a common Excel sheet, all the Excel graphic formatting options are open and useable.

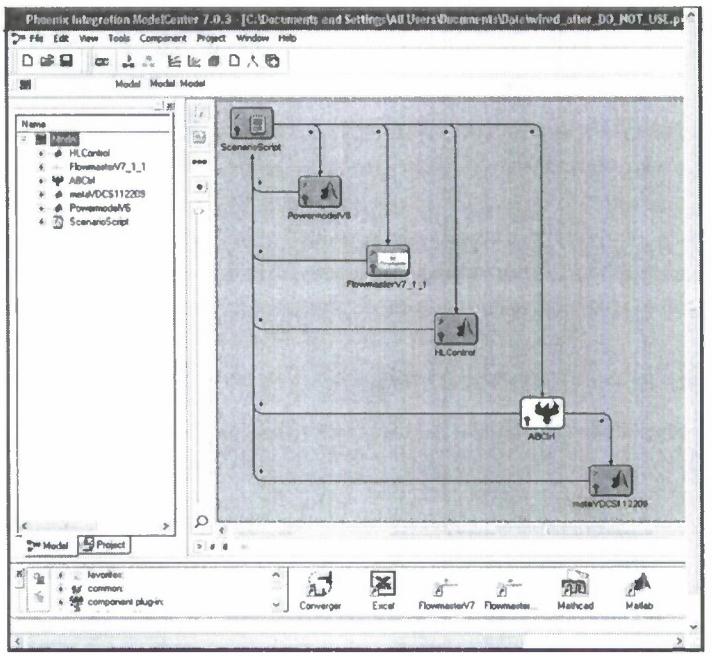


Figure 6: New setup of notional YP co-simulation

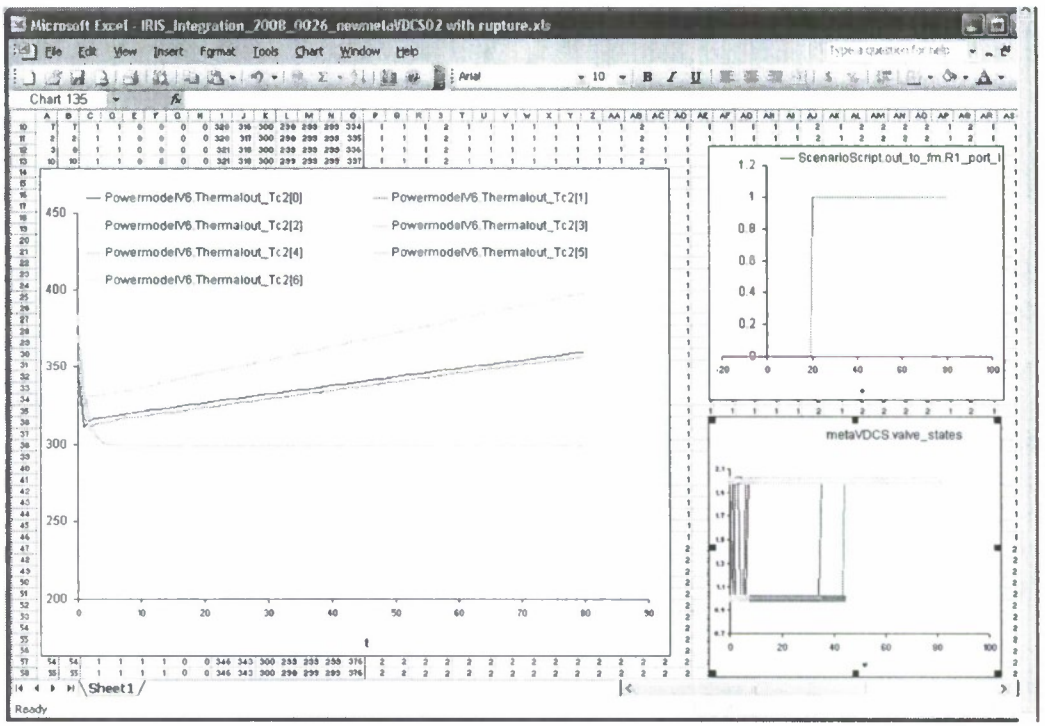


Figure 7: Excel output of time series for given variables

In order to track valve behavior during the simulation, and to be able to better assign which valve within the notional YP fluid network has what status, another Excel sheet was developed. It represents the notional YP fluid network using a diagram. It uses Excel control objects to represent the states of valves (both valve opening state and flow rate), pumps (on or off), and service load temperatures. This makes it easier to understand the current system state, and proves itself useful to debug errors that currently still exist within the rupture identification algorithm of the low level controller. Figure 8 show a notional sample output for the overview.

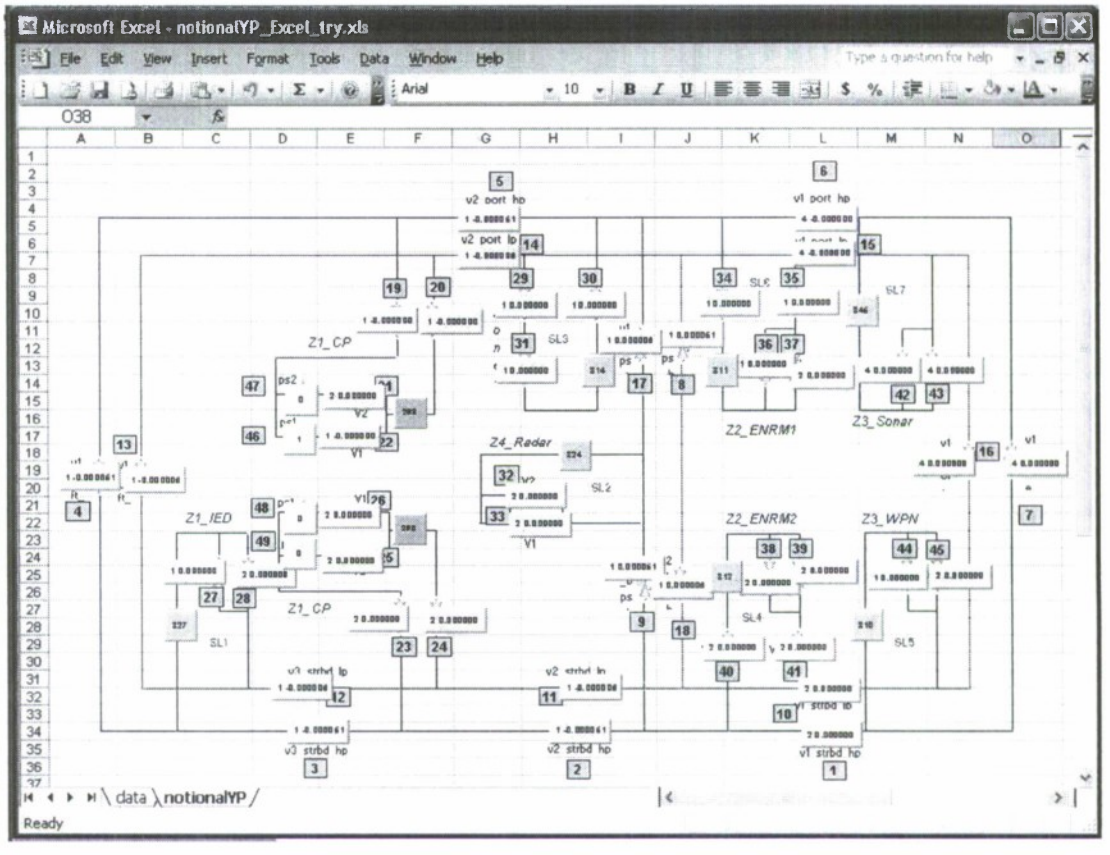


Figure 8: Excel output of valve data in a notional YP fluid network sketch

Model validation and error bounding

It has previously been proposed that methods of numerical integration for ordinary differential equations are being investigated for the validation of co-simulation results. After more investigation into the problem, it has been determined that a true validation of a co-simulation is not possible, at least not with the initially proposed approach. Nevertheless, the general idea can still be employed to validate whether a co-simulation runs within certain given error bounds. For this, advanced methods of numerical integration of ordinary differential equations are modified such that their underlying algorithms and principles are adapted towards a use in co-simulation error bounding. The algorithms have the properties that they use both future and past data, perform a

predictor-corrector approach of the variables under observation, and adapt the simulation time step according to a given error limit. Various such methods exist, and will be carefully weighed against each other to find an optimum trade-off between simulation execution speed and accuracy of the results. The programming of such a method is currently ongoing. It will be tested with a simple dynamic model which consists of two sub models. The monolithic model behavior for this simple co-simulation is known, and hence the algorithm output can be compared with the "true" system behavior to see how well this implementation works. A more complicated model exists in the ASDL lab, and will be used for further studies on the subject once the initial implementation and testing has been successful.

Future Work

Within the next three months, the mathematical principles of the suggested approach will be collected, refined, and synthesized to an overall approach for co-simulation. A simple model to test the approach already exists, and will be modified to include the new mathematical methods. Further down the road, a more complicated and realistic model will be developed, which will represent a ship system and which will serve to evaluate the approach and make sure that it is applicable beyond a "lab" environment. Further investigations will address issues with time steps and data exchange schedules in stiff systems, as these issues may cause further problems in a real world simulation. Also, the upper and lower bounds for time steps, and the algorithms for time step settings will be evaluated and tested.

Task 2: Intelligent Autonomous System

Subtask 2.1: High Level Control

Introduction

In the operation of complex systems such as naval ship, the control systems are hierarchical in nature. The controllers at each level has their own objectives and they collaborate together to achieve the overall operation goals. The effectiveness of an intelligent system depends on the functionalities that the controllers at different levels provide, which results from the objective decomposition and the use of control techniques at each control level. In the resource allocation process, since the resources are limited, a well designed control systems are needed to make right decisions and control commands in order to efficiently utilize the resources. IRIS control systems consist of three levels of controls: high-level control, mid-level control and low-level control. Each control level has different control objectives and employs a varied method or technique to fulfill its functionalities.

The high level control of the M&S environment determines the priorities of the ship systems for resource allocation based on the systems' importance to the mission, their status and environmental state. That is, how much resource a system can get depends on its importance and priority in order to operate properly and maximize the mission effectiveness.

Progress

Various advanced decision and control methods are investigated for the resource allocation problem. It turns out that the rule-based method is effective enough to accomplish the high level control task. This is due to the fact that high level decisions about the system priorities are made by the controller, which is effectively to be realized by a rule based reasoning process. In addition, the rule based controller is easy to be implemented and developed for this purpose, and is flexible in modification and extension for dealing with multiple-resource allocation problem. For example, in this study the high level control is applied to reallocate the cooling resource by prioritizing the ship systems. This prioritization is determined by the evaluation of environmental state, system status and mission being performed. The developed high level controller possesses intelligence when deciding on the prioritization for the ships systems. The priority not only depends on the importance of a system to the overall mission, but also depends on its urgency to require the resource. The output from the high level controller is the percentage of maximum flow rate that a system can get based on the evaluation of its importance to the mission and urgency to get resource. This is given by Equation (1).

$$Perc = \frac{w_p * p + w_u * U}{w_p + w_u} \tag{1}$$

where p and U represents the importance and urgency, and w_p and w_u are their weighting factors, respectively.

From the equation, it can be seen that the percentage is a normalized weighted sum of the importance and the urgency of the system. If a system has high importance because it is critical to the current mission, but this system has a large margin before it reaches a critical stage (e.g. not extremely hot but running at a regular operating temperature), then it makes sense to save some of the resources and not provide this load with further resource. This formulation implemented in the high level controller has been proved to increase the efficiency of resource usage and mission effectiveness. The high level controller will give this percentage value to the mid and low level controllers which will control the corresponding valves and find a optimal route to distribute the required resource to the ship systems.

Future Work

Further work will be done to implement the high level controller to effectively allocate multiple resources for the ship systems. The interactions between multiple resources will be addressed as well when allocating the resources.

Subtask 2.2: Multi-Agent Based Mid-level Control with Dynamic Inference Engine

Introduction

In the last quarterly report, the detailed description about Multiple Sectioned Bayesian Networks (MSBNs) had been presented and the sketch of the implementation has been

formulated. Currently, the integrated model works properly. Several different scenarios are performed to test the integrated model and the simulation results are analyzed.

As described before, agents are established in JADE which is completely implemented in Java, while Flowmaster is thermo-fluid simulation software. The integration task is to make the Flowmaster model capable of accepting control commands from the control agents and the control agents receiving necessary information from the Flowmaster model. Although establishing direct communications between a Flowmaster model and JADE agents is time-consuming, they can be connected through some intermediate tools. Flowmaster model supports COM objects such as Controllers and Gauges. The Controllers can accept information from model coded in Visual Basic while the Gauges can send simulation results to Visual Basic programs. JADE supports Java Beans, and ModelCenter of Phoenix Integration as an integration tool supports both Visual Basic Plug-Ins and Java Bean Plug-Ins, therefore, Flowmaster model and agents in JADE are communicating with each other through Visual Basic program and Java program in ModelCenter. The entire test model in ModelCenter analysis view was presented in previous quarterly report and the integrated model for the application is complete. A script scheduler written in VBScript manages the simulation determining the running order of each model and making sure right information is exchanged between models at right time. All of the tested scenarios are running over a predetermined time span [0 1200*sec*] and are defined in different Excel worksheets. All of the prior distributions and conditional distributions for the Bayesian networks are notional and not presented here due to space limit. For each scenario, it runs about 50 minutes on a computer with Intel(R) Core(TM) 2 Duo CPU, E7200 @ 2.53GHz and 1.96 GB of RAM.

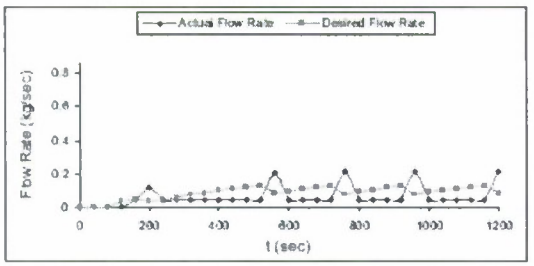
Scenario Study and Result Analysis

Scenario 1 (Nominal Condition)

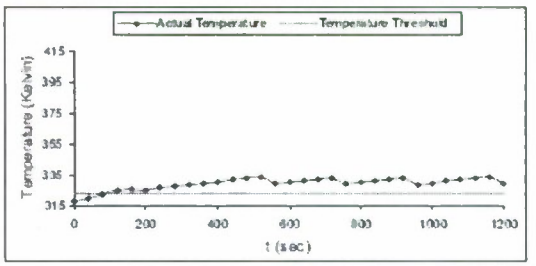
Conditions: all of the components are not damaged; every flow rate point listed in the Bayesian network is observable; every component open degree is observable; resource capacity is 0.8*kg/sec*; the initial temperatures of service load 1 and service load 2 are 317*kelvin* and 400*kelvin* respectively. After the scenario is run, the monitored outputs are created and shown in Figure 9.

Initially, service load 1 temperature is below $T_{threshold}$ = 323*kelvin* while service load 2 temperature is above the threshold. Service load 2 requires cooling water. Thus, Pump1 and valve1 in resource center are open to provide cooling water to service load 2. Pump2 and valve 2 as a redundant set of pump1 and valve 1 in resource center keep shutdown. Valve 11 in service load 2 is open to accept cooling water from resource center. Since service load 1 as a power component has 50*kw* incoming power and the component efficiency is 0.7, 30 percent of the incoming power becomes heat of service load 1 and causes its temperature to increase. Thermo-Electrical System (TES) CA calculates the required cooling fluid flow rate of each service load according to its properties and its current temperature every 40 seconds. At time t = 120*sec*, the temperature of service load 1 is above the temperature threshold as shown in

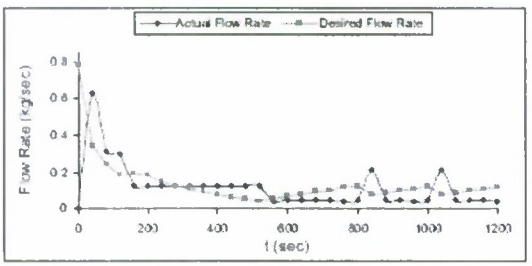
Figure 9 (b) and TES CA gets a required flow rate greater than 0 and sends it to the Agent-Based Control (ABCtrl) CA. ABCtrl CA gets the information and tries to redistribute the resource to each service load. Valve7 is Open with probability close to 1 at time $t = 200sec$ to accept cooling water for service load 1 from resource center. From Figure 9 (a) and Figure 9 (c), we can see that the actual flow rate and the required flow rate do not match exactly all the time, because the flow rate is controlled by adjusting one valve open degree at one time in one service load in a discrete way and the valve open degree in another service load also affects the flow rate in this service load. In the application, adjust valve open degree in service load 1 to satisfy its flow rate requirement firstly, and then adjust valve open degree in service load 2 to satisfy its flow rate requirement, which affects the flow rate in service load 1. That is why the difference between the actual flow rate and the desired flow rate in service load 1 shown in Figure 9 (a) is slightly bigger than that in service load 2 shown in Figure 9 (c). This argument also explains why the temperature of service load 1 fluctuates slightly more heavily than the temperature of service load 2 does as shown in Figure 9 (b) and Figure 9 (d) respectively. This problem can be solved by adjusting all of the valves "simultaneously". By "simultaneously", it means to refine the time step of adjusting each valve open degree further. That is, in each small time step, make smaller adjustment sequentially for all of the valves. However, under current situation, the temperatures fluctuate is in a tolerable range so it does not need to get through this practice. In summary, for the nominal case, the control system with MSBNs inference engine can make the right decisions and distribute the resource to different service loads accordingly.



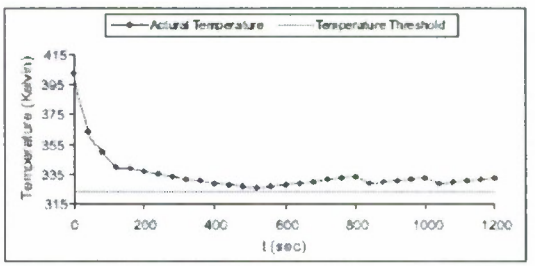
(a) Service Load 1 Flow Rate vs. Time



(b) Service Load 1 Temperature vs. Time



(c) Service Load 2 Flow Rate vs. Time

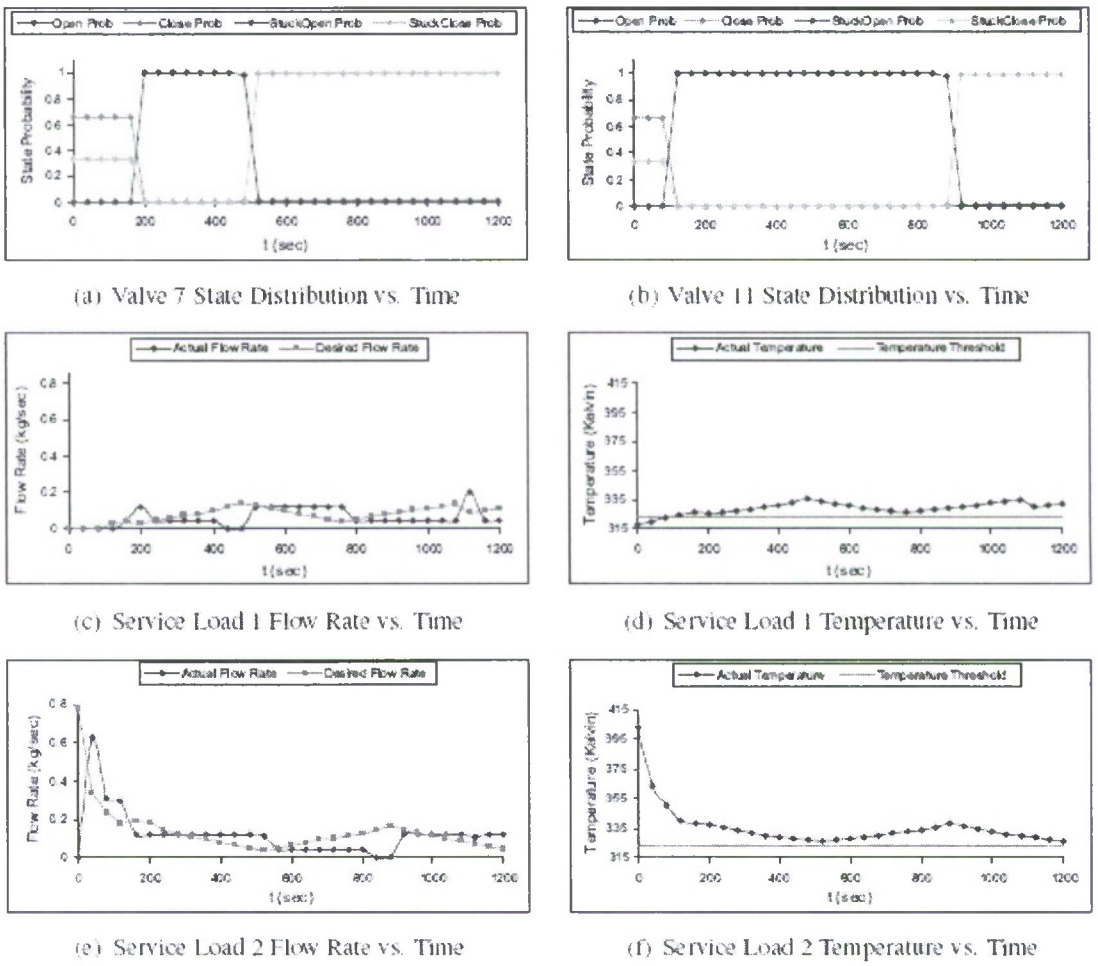


(d) Service Load 2 Temperature vs. Time

Figure 9: Scenario 1 Results

Scenario 2 (Damage Condition 1)

Initial conditions: all of the flow rates listed in the Bayesian network are not observable; every component's open degree is observable; resource capacity is 0.8*kg*/*sec*; valve7 becomes StuckClose at time $t = 440sec$ (the 11th iteration); valve11 becomes StuckClose at time $t = 840sec$(the 21st iteration); the initial temperatures of service load 1 and service load 2 are 317*kelvin* and 400*kelvin* respectively. The simulation results of this scenario are shown in Figure 10.



(a) Valve 7 State Distribution vs. Time

(b) Valve 11 State Distribution vs. Time

(c) Service Load 1 Flow Rate vs. Time

(d) Service Load 1 Temperature vs. Time

(e) Service Load 2 Flow Rate vs. Time

(f) Service Load 2 Temperature vs. Time

Figure 10: Scenario 2 Results

Before time $t = 440sec$, scenario 2 is as the same as scenario 1 except that no flow rate is observable. Compare Figure 9 with Figure 10, we can see that the estimation results and control results in scenario 2 are close to those in scenario 1. At time $t = 440sec$, valve7 becomes StuckClose. The MSBNs inference engine detects valve7 state change at time $t = 520sec$. At time $t = 440sec$, an Open command is given to valve7 from the control system; at time $t = 480sec$, the fluid network gives the valve open degree information back to the control system; through data processing, at time $t = 520sec$, the MSBNs

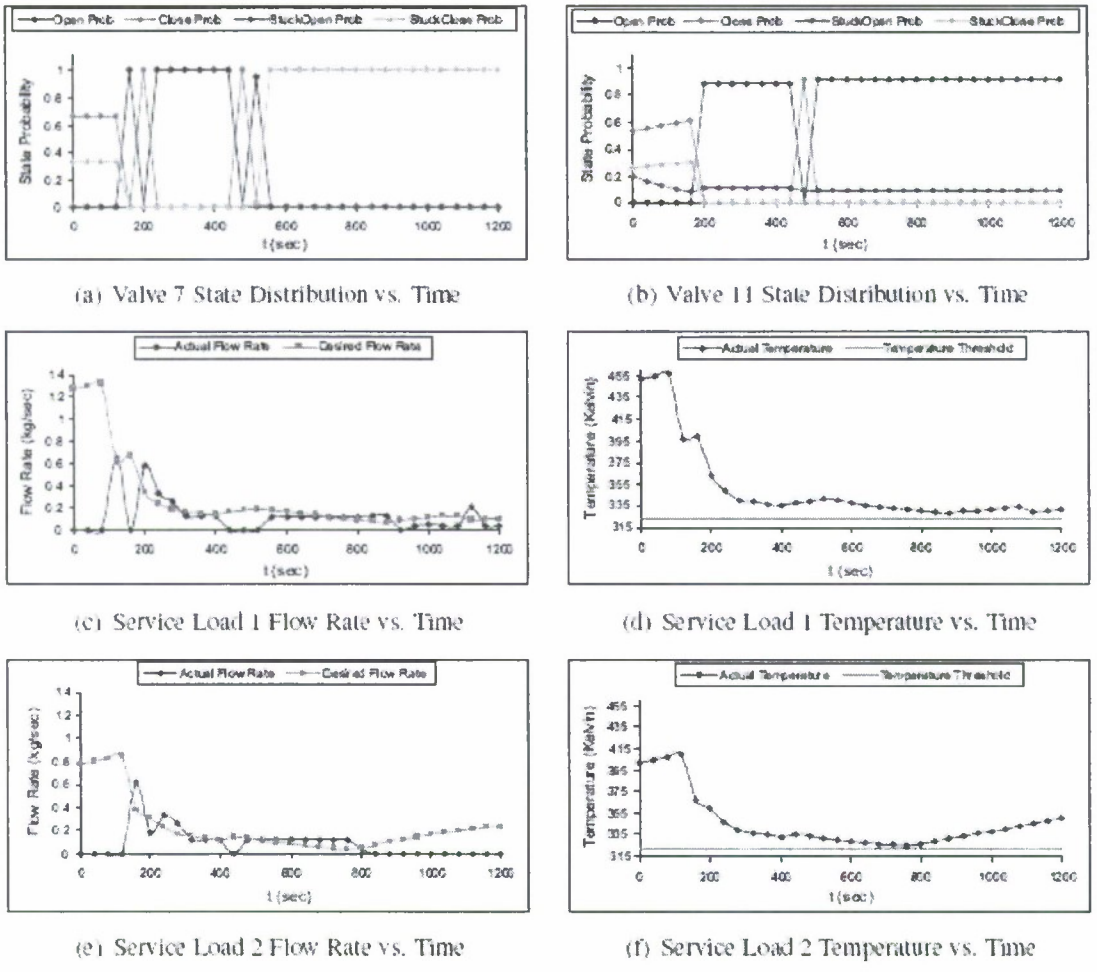
inference engine gives that valve7 is StuckClose with probability close to 1. At the same time, an Open command is sent by the control system to valve8 as a backup of valve7 in the fluid network system and valve8 state distribution is changed into Open with probability close to 1 at time $t = 520sec$. Similarly, valve11 is detected to be StuckClosed 80 seconds (two iterations) later after its state change at time $t = 840sec$ by the MSBNs inference engine; an Open command to valve12 is initiated by the control system at time $t = 920sec$ and valve12 state distribution is changed into Open with probability close to 1 at time $t = 960sec$. The detections about state changes of valve7 and valve11 are shown in Figure 10 (a) and Figure 10 (b) respectively. Due to the two iterations (80 seconds) delay of the state change detection, there is no flow through service load 1 during time $t = 440sec$ to time $t = 520sec$ as seen in Figure 10 (c). Similarly, there is no flow through service load 2 during time $t = 840sec$ to time $t = 920sec$ as shown in Figure 10 (e). The delays are also reflected in the steeper temperature increasing in service load 1 during time $t = 440sec$ to time $t = 520sec$ as shown in Figure 10 (d) and that in service load 2 during time $t = 840sec$ to time $t = 920sec$ as shown in Figure 10 (f). In summary, without any flow rate observation and only with component open degree observations, the MSBNs inference engine can detect component damages quick enough and the control system can reconfigure the whole system by switching from damaged components to their corresponding redundant ones to redistribute system resource.

Scenario 3 (Damage Condition 2)

Initial conditions: only the flow rates at the points located in the upstream of the valves in each service load and listed in the Bayesian networks are observable; valve open degrees are observable only for valve1, valve2 and valve7; resource capacity is $0.8kg/sec$; valve7 becomes StuckClose at time $t = 440sec$ (the 11th iteration); valve11 becomes StuckClose at time $t = 840sec$(the 21st iteration); the initial temperatures of service load 1 and service load 2 are $450kelvin$ and $400kelvin$ respectively. The monitored outputs are shown in Figure 11.

Initially, both of service load 1 and service load 2's temperatures are above $T_{threshold} = 323kelvin$. At time $t_0 = 0$, service load 1 and service load 2's cooling water requirements are $1.27388507kg/sec$ and $0.781790803kg/sec$ respectively. The summation of those two requirements is more than the resource capacity $0.8kg/sec$. Due to 2 time steps delay, no cooling water is supplied until at the 2nd iteration, where service load 1 temperature and service load 2 temperature increase to $457.3051948kelvin$ and $407.3051948kelvin$, while the cooling water requirements increases to $1.32181633kg/sec$ and $0.829722062kg/sec$ respectively. Service load 1 priority is superior to service load 2 priority, so the control system tries to satisfy service load 1 requirement first by giving an Open command to valve 7 in service load 1 and a Close command to valve 11 in service load 2. Service load 1 is cooled down very quickly and its temperature is decreased to $397.5332534kelvin$ at the 3rd iteration after 40 seconds cooling down by the actual flow rate $0.644990944kg/sec$, which is different from the capacity $0.8kg/sec$ of the chiller plant. The reason is that the estimation of the capacity $0.8kg/sec$ is based on the assumption that all of the valves out of the chiller plant are open. However, this difference does not have significant effect on the whole control system performance. Service load 2's temperature keeps increasing to $409.7402597kelvin$ and its corresponding cooling water requirement

increases to 0.853687692*kg/sec* while service load 2's cooling water requirement decreases to 0.629146348*kg/sec* at the 3rd iteration. The flow rate of service load 1 and service load 2 are shown in Figure 11 (c) and Figure 11 (e) respectively. Now, service load 1's requirement is less than the capacity of the chiller plant, so the control system tries to redistribute the left cooling water to service load 2 after providing enough cooling water to service load 1.



(a) Valve 7 State Distribution vs. Time

(b) Valve 11 State Distribution vs. Time

(c) Service Load 1 Flow Rate vs. Time

(d) Service Load 1 Temperature vs. Time

(e) Service Load 2 Flow Rate vs. Time

(f) Service Load 2 Temperature vs. Time

Figure 11: Scenario 3 Results

In this simulation, the control system picks up a state of a component proportionally to this state likelihood. Although a state likelihood is very small, it still has the chance to get picked up. This argument can be used to explain some sudden jumps shown in the results of the application. At time t = 440*sec*, valve7 becomes StuckClose. The MSBNs inference engine detects valve7 state change at time t = 560*sec* as shown in Figure 11 (a). Valve11 state becomes StuckClose at time t = 840*sec*. However, the MSBNs inference engine can not detect the damage of valve11 and gives a wrong state estimation as StuckClose with probability close to 0 as shown in Figure 11 (b). The control system uses the wrong state estimation of valve11 from the inference engine and keeps giving

valve11 an Open command. Therefore, the actual flow rate through service load 2 becomes zero since time t = 840*sec* as shown in Figure 11 (e) and service load 2 temperature keeps increasing since time t =840*sec* as shown in Figure 11 (f). Service load 1 flow rate and temperature become stable as shown in Figure 11 (c) and Figure 11 (d) respectively. In summary, without enough observations, MSBNs can not detect some component state changes. Another reason for the wrong estimation in this application is that the fluid network is a recycled cooling system and Bayesian network can not handle directed cycles. In the simulation, the directed cycle is broken by giving hard evidence to a possible measurable flow point. It indicates that the recycled cooling system is not the best example to show the effectiveness of MSBNs inference engine and MSBNs can perform better for non-recycled systems.

Future Work

Currently, the fluid network model for testing the proposed methodology consists of one chiller plant and two service loads. In the following two months, the small model will be extended to a chilled water system of notional YP ship which includes two chiller plants and 7 service loads. The proposed methodology will be implemented to the extended fluid network.

Task 3: Graph-Based Component Surrogate Modeling

Background

The objective of this task is to develop an M&S method that is capable of taking the topological configuration among the components of the chilled-water network as a "design variable." Then integrated with the design-space exploration or the design optimization process, this proposed M&S environment may enable a simulation-based design for resiliency and survivability study. The method is developed by combing three techniques – graph-based topological modeling, object-oriented component model definition, and surrogate modeling for representing components' behaviors. Though the development of the method is implemented to fluid systems modeling, the development approach has also considered the extension of usage of this method including the application to electrical power network, which is another most common type of networks in engineering systems, with just minor modifications.

Summery of previous progress

As mentioned in the last report, this task was divided into five phases, which were:

Phase 1: Development and test of the basic classes for a simulation environment like the solver, the data manager and the classes of the graph elements such as nodes, edges, sources, sinks, and damages.

Phase 2: Development of the damage scenario generator class. Test of the damage simulation of the M&S environment

Phase 3: Development and test of a reference damage controller.

Phase 4: Development and test of the experimental design class for the network topological space.

Phase 5: Development of the chilled-water system model of the notional YP, and demonstration of damage analyses and topological design space exploration.

As of the last report, the progress made was: for Phase 1 over 90% has been completed, Phase 2 about 70%, and Phase 4 about 40% completed (the percentage numbers are subjectively estimated). The implementation and test of those phases were with a simple fluid network that has three heat exchangers and a chiller-pump unit.

Progress

The research has been focused on the Phase3 and Phase 5 during the fourth quarter of this year. First of all, the graph-based surrogate model of the notional-YP cooling system was finally constructed with the developed modeling environment so is being used for the Phases 3 and 5 research works. For the Phase 3 research, an auto-generation algorithm of a reference damage control for a given graph plant model was developed. The preliminary test using the previous simple fluid network was quite successful (isolating an arbitrary damage), although another test with the notional-YP revealed bugs of the algorithm code and the debugging effort is currently ongoing. The corrected model is expected to be ready by the end of this year.

The developed simulation environment was tested and the simulation ran with a single pump turned on at the speed of 400 rad/s (about 4000 rpm), and a damage was triggered at t_{sim} = 2 sec creating rupture at the location shown in Figure 12. After the rupture happens, the pipe model components immersed in a damage bubble was reconfigured. During the five seconds of simulation time, the flow rates and the pressure values of the two ends of all seven heat exchangers in the notional YP model were measured. The plots presented in Figure 13 are the simple demonstration output from the simulation of the graph-based model of the notional-YP cooling system.

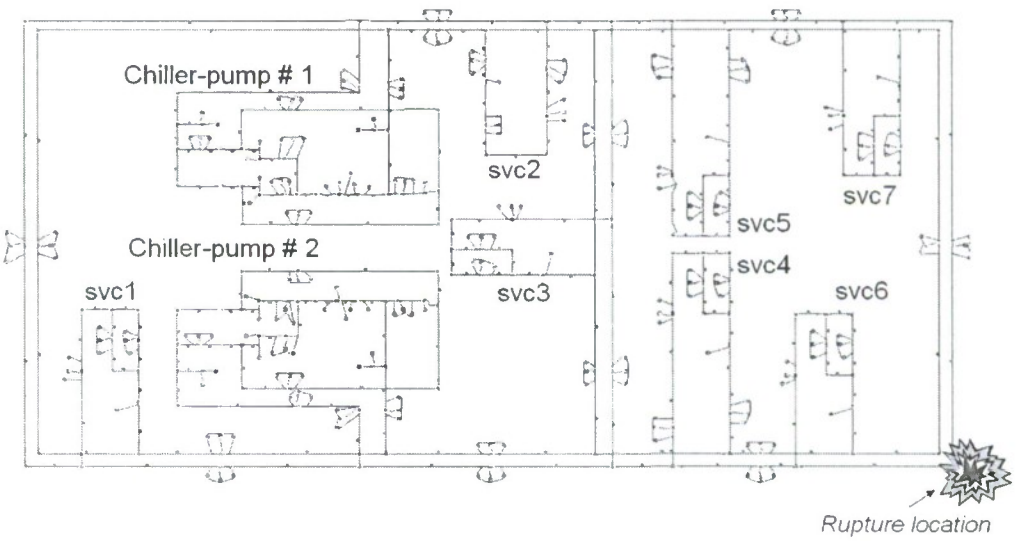


Figure 12: Notional-YP cooling system model and rupture location

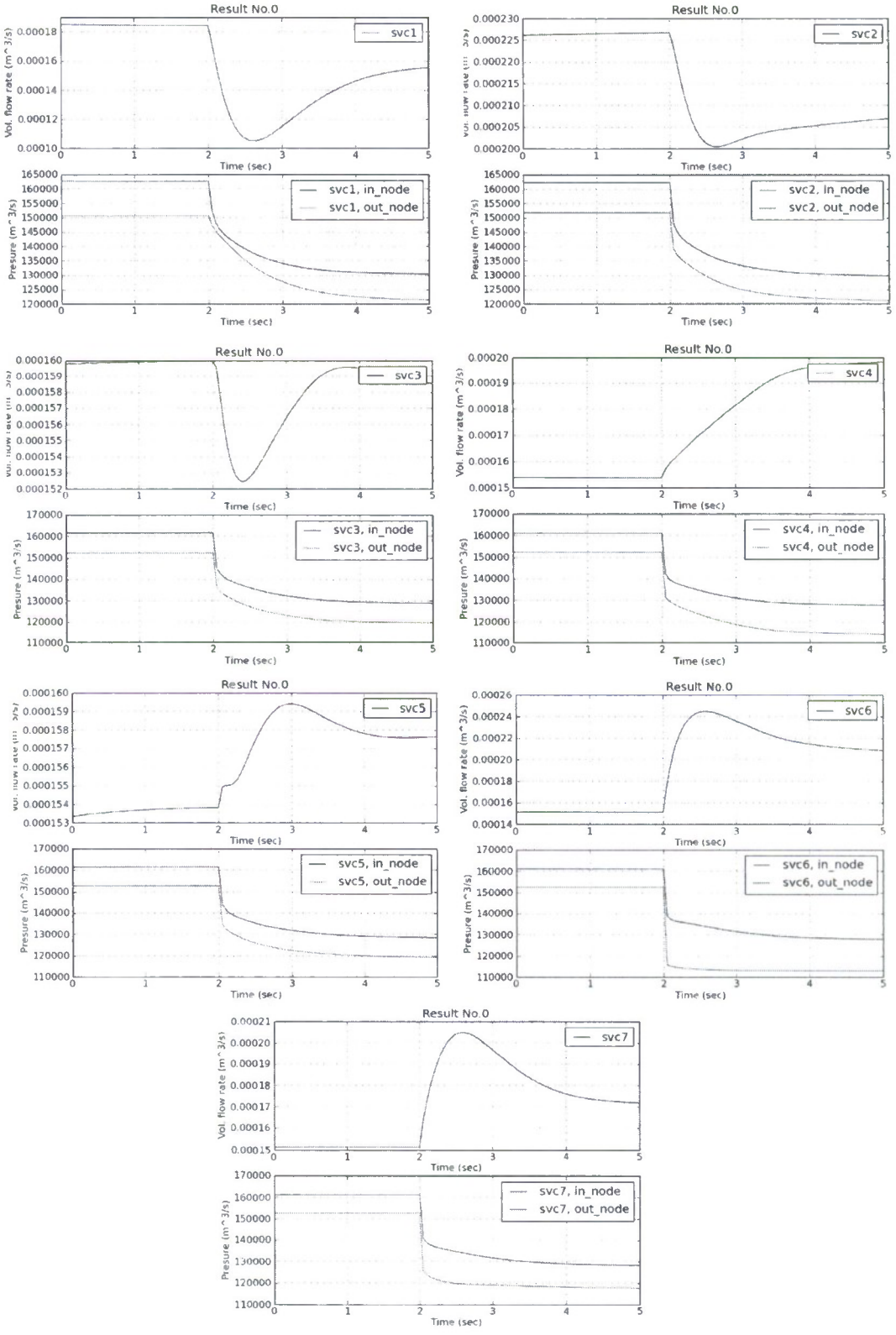


Figure 13: Result of the damage simulation

As it can be seen from the simulation results, the pressure responses were monotonically dropped – this behavior is very predictable and consistent with engineering intuitions, but the flow behavior, on the other hands, was a lot more complicated so the simulation will be very helpful in the engineering decision and design analyses.

The current research progress status is, by a subjective estimation, about 80% completion of the Phase 3 research and 70% completion of the Phase 5 research.

Future Work

After the Phase 3 research gets done, the M&S environment will finally be capable of performing the closed-loop damage analysis. The Phase 5 research will be moved forward, focusing on the development of topological analysis capability. As soon as it is achieved, the post-processing algorithm will be constructed for the design-oriented, comprehensive and rigorous computer experiment environment, which will be the final delivery of this research.

Task 4: Human in the Loop Control

Introduction

The Human Machine Interface (HMI) started as a simple visualization tool for observers to identify and understand emergent behaviors of the complex system, such as IRIS type systems, as the system consists of multiple subsystems. The tool allows for rudimentary user interaction to see how behaviors might be influenced by human interaction. Since the HMIs conception, the tool has spurred many other ideas and questions. These questions challenged how engineering tools are designed and used. They challenged the nature of useful engineering knowledge for design purposes. In the process of investigating solutions to these challenges, the HMI is transforming into a design tool seeking to increase the accessibility of engineering knowledge by pooling the strengths of distributive systems. This is referring to the distributive nature of the design process of large complex systems.

By default large complex systems must be decomposed in hopes that the system may be understood by its parts. Engineers attempt to maintain both the macro and the micro perspectives of the system. Nonetheless, experience teaches us that these attempts are generally expensive and risky. Risk is introduced as decisions are made. Knowledge may mitigate risk, thus the more that is known when a decision is made the less risk the decision might incur. The catch is that knowledge is built upon decision. One cannot understand the system as a whole until it can be understood by it causes, which are determined by the interacting parts.

The HMI might be able to contribute to a solution. From the beginning the HMI was design as a web-based tool providing a level of abstraction between the user and the services. The service in this case is a remote simulation environment. This abstraction would enable engineers outside the IRIS project to study a complex system in a hands-on manner, interacting with the simulation, and observing behaviors. By introducing key

levels of abstractions between users and services, and services and other services, the complexities of knowledge building and designing might be controlled. Thus reducing risk during the design process.

Fundamentally IRIS is an exploration: it is an activity to build engineering knowledge for a specific class of systems. In its conceptions every decision requires a rational process to substantiate it. However, the design needed a starting point, a beginning to form a foundation for knowledge. This prerequisite transformed the project into an infrastructure project and an experimental plan designed to act as a first step to understanding. What was hopped to be learned was a true appreciation of the underline causes behind the behaviors of the integration of intelligent systems.

Objectives

The vision behind the HMI could not be realized without some guiding objectives. Each objective has some roots from observations of projects from a variety of disciplines. The notion is that there existed a generalized solution to each phenomenon that project hoped to address. The objectives broke down into four categories:

1. Real-Time Operation
2. Data Fusion
3. Visual Analytics
4. Dynamic Data Sets

Many aspects of the objectives are being tackled in small steps. The following sections will reveal more as to why these objectives have been chosen and how they are being addressed.

Real Time

In order to properly present a simulation intended for human interaction, the simulation environment and the HMI must be able to maintain a real time performance requirement. This helps human operators obtain a feel for the responsiveness of the system to external stimulus. This burden is heaviest on the simulation environment, but it does mandate that the HMI must have the feel of a locally executed application on the desktop.

The HMI client takes advantage of asynchronous communications and data pre-fetching to achieve this goal. The theory being that it will be network lags that are the most encumbering. In general this belief has held true. However it has been noticed that older computers do show some lags do to the load of a full screen flash application. Often older computers are coupled with new monitors with higher resolutions, or for some other reasons they do not meet the minimum video performance required. This will lead to a hardware requirement specification that will be released with each version of the client.

Data Fusion

The development of the HMI has created an interesting opportunity in the realm of design science. On the one hand the HMI has a very tight integration with simulation environments, specifically those utilizing Model Center. On the other is an application

framework for analyzing data. In between is a database. Collaboration systems for design purposes have been a long standing interest at ASDL

- Distributive Design
- Distributive Modeling
- Distributive Simulations
- Etc.

Data fusion is a step beyond data integration. Data integration is the process of merging two or more data sets into one, while data fusion suggests that more can be learned from merged data sets after a reduction process. People perform data fusion when they abstract meaning from multiple data sets and then determine meaning or consequence from the combined abstracts. A system with a service orientated architecture can be utilized to perform data fusion tasks in a distributed fashion. Only this fusion process does not need to be limited to raw data. It could be applied to designs, models, and simulation environments. The key is to maximize the potential benefits of any effort by keeping modularity and reusability in mind. Object orientated programming achieved this at the application level. The next step is service orientated, network centric architectures of both data and software.

Historically applications and data have been treated as static entities. Unfortunately the reality is that these entities change frequently. The applications change. The models change. Software in general will change. Change in fact has become the problem. The problem was created after computer systems became decentralized during the beginning of the era of the personal computer. Decades later with the advent of the internet computer systems are beginning to resemble a hybrid between centralized and distributed systems. This hybrid if realized can bring the information ages into a useful collaboration environment.

The vision of this objective is to explore the nature of the hybrid centralized and distributed system model and the implementations it may have on the design paradigm for complex physical systems.

Visual analytics

Visual analytics has been described as the sciencc of analytical reasoning facilitated by interactive visual interfaces. Recently attention in this area has been on the rise due to a strong interest in the subject from the department of homeland security. Visual analytics has the potential of enabling the processing of an overwhelming amount of disparate and conflicting data.

The design process is an act of analytical reasoning and the dimensionality of complex physical designs is overwhelming without the proper tools and techniques. The HMI is a visual analytics tool in that it was design to facilitate the design process using an interactive visual interface. The server further supports this role by supporting tools such as JMP from SAS and Microsoft's Excel.

Dynamic Data Sets

This aspect refers to the use of tools designed to aid decision making, i.e. the so call calculator tools or sometimes call dashboard tools that are often utilized to help decision maker with making wise strategic decisions. These tools attempt to present information in such a way that it might be meaningful to a decision maker. This information is often based on some collected data that if not refreshed becomes very static. The HMI as an objective hopes to produce an environment for decision makers with real-time data.

Progress

Documentation System

Based on MediaWiki Project

The MediaWiki Project has gained a strong presents in the web community. Not just within open source circles but the general public at large. This is primarily due to the success of Wikipedia which is based on the MediaWiki project. The underline concepts are based on the basic world-wide-web with one exception. MediaWiki opens the doors for contributors to provide content to the system.

User-based and Developer-based Contributions

The HMI server has been equipped with a wiki system similar to that found at Wikipedia, as shown in Figure 14. Only here the wiki's purpose is to document the HMI. It has always been the intention for this project to enable users to further develop the capabilities of the HMI. So it only makes sense that users should also be able to contribute to the documentation.

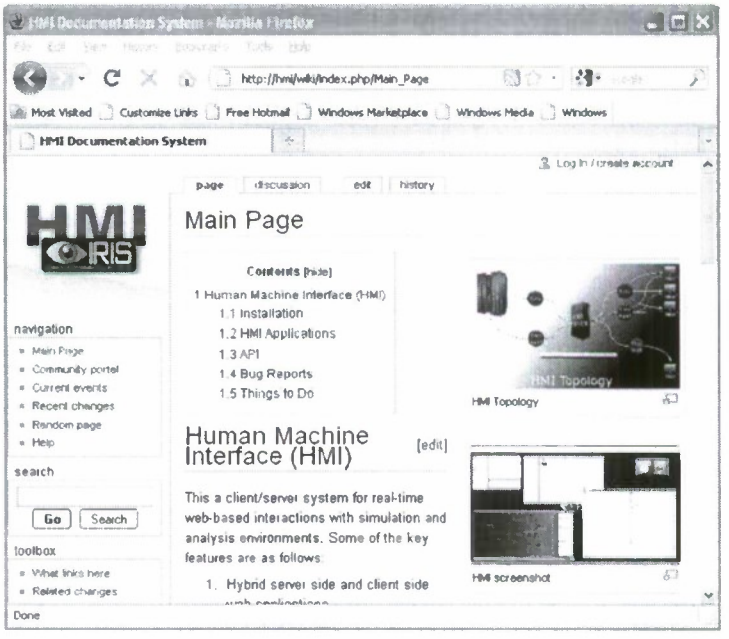


Figure 14: HMI Wiki

Virtual Machine Based Builds

Based on Virtual Box (Open Source)

Virtual Box is an open source x86 virtualization package. In essence one could create a machine (computer) virtually, configure it and redistribute the machine with few strings attached. This creates an ideal platform for engineers to build within, without having to have too much regard for the environment the virtual machine will be running in. Server virtualization is become quickly adopted in general for many advantages they afford, including but not limited to the ease of distribution, backup, and management.

Easy Distribution

Once a virtual machine is configured and running it is a simple process to export the machine and burn to a disc or USB drive for easy distribution. The virtual machine encapsulates all the software required for the servers operations. The installation process is again another two step process. Download and install the virtual box software, and them import the virtual machine. Once the machine has been imported, press the start button and the system is up and running.

Easy Snapshot Backups

Since the actual virtual machine is little more than a few files the entire machine, backups are simple. Snapshots are an easy way to protect the system from changes. At anytime (even while the system is running) a snapshot can be taken of the system. At which the system could be rolled back to any given snapshot at anytime. To protect against a hardware failure simply shutdown the virtual machine and export it to a backup location.

Low Resource Requirements

Currently the virtual machine is running Ubuntu server 9.10 with a standard LAMP (Linux, Apache, XML, and PHP) installation. The HMI is then installed on top of the LAMP configuration. The virtual machine is configured for 512 megabytes of RAM and 30 gigabytes of disc space. This configuration was design for some growing room. Currently the actual disc space being used is less that 900 megabytes and the server will run with less than 128 megabytes of RAM.

Model Center Plug-in

Two-way communication

The new Model Center plug-in was only one way until now. Since the first version of the HMI the structure of the information sent between the simulation environment and server has changed significantly thus requiring new parsers and encoders to be written. The new implementation supports the XML dialect called dashML. The plug-in is written in java script using a flexible extension of the object class to handle all the encoding, decoding, and communications with the HMI server.

Built-in logger

The new plug-in now has a new logging capability. Each event is logged and reported back to model center for verification purposes. This feature allows a user watching the

simulation environment to quickly troubleshoot issues with either integration or networking. In addition to the logs the plug-in also reports the actual data being sent to the HMI server and the actual data received before parsing.

Built-in error handler

The plug-in also has an error handler which should prevent the plug-in from ever crashing the simulation environment. Once an error is detected it is logged with an explanation, and the plug-in attempts to exit gracefully.

Stream-line work flow

The work flow is condensed to simply specifying which variables are to be sent and which are expected back from the server. No more details are required from the user, making communication with a remote service as easy (if not easier) as working with a completely local environment.

Fully Documented

The plug-in API is completely documented on the HMI wiki system. The documentation includes examples and explanations for each method and attribute currently in use. As mentioned earlier the wiki allows for the user to augment the documentation as appropriate to clarify for future use or customizations.

Future Work

The items in this section have been reported in the past quarterly report in detail. The details will not be repeated here, however a short list is being carried over as a reminder of task being purposed for future work. The list is as follows:

- More advanced error handling
- Categorize errors
- Hybrid logging system (between the client and the server)
- Scalable vector graphics renderer
- Server side SVG compiler

It is notable that these points are having an influence on work that has taken place this past quarter. The ModelCenter plug-in which has been described above includes both the error handling and the logging capabilities.

Task 5: A Methodology for Improving System Effectiveness in Resilient Systems Design

Introduction

Traditional design approaches are based on optimizing naval systems for performance, based on a very limited number of mission scenarios. While the traditional design approach is conceptually fairly simple and straightforward, it does not really address any issues regarding the uncertainty around naval system mission requirements,

environmental condition or even the capability of the system to perform as designed under real operations. A robust solution will represent a system that in theory would be better prepared to perform multiple mission acts and withstand a larger spectrum of unexpected events. At the same time, prescribed design performance might not be optimal, in order to compensate for the multi-mission capability (e.g., preferred extra weight for redundant systems over maneuverability).

One of the main objectives of IRIS is to deliver a conceptual design methodology for more survivable and mission effective ships. However, the question that remains at this point is how exactly the multi-mission capability and the enhanced survivability are enabled. Typical survivability enhancement features, such as component redundancy, separation and shielding are immediate techniques that can be properly applied to the design based on conceptual sizing. Yet, there is no standard design method that determines the extent of such enhancements and the type of survivability that each enhancement would seek to improve (susceptibility, vulnerability or recoverability). The current United States Navy standard is primarily determined by the Survivability Design Handbook for Surface Ships (OPNAV P-86-4-99) and according to this procedure, survivability is improved by focusing only on vulnerability, implying that susceptibility reduction and recoverability improvement are expected consequences of the former.

In order to extend the state of the art to address all types of survivability, there is a modern philosophy recently emerging and seeks to address the aforementioned concerns. Resilience engineering is a philosophical framework that encompasses a collection of concepts around safety management and engineering. Some of them focus on understanding threats, accident and damage propagation, as well as how a system should be designed to conform to changes that occur around it, for the purpose of withstanding adverse effects and maintaining its mission effectiveness.

The fundamental research question regarding this initiative would be how to improve the design the system, so that system effectiveness through survivability is maximized for a given set of scenarios, which will include system damage and/or restoration events. Moreover, it can extend to consider how the philosophy of resilience engineering can translate into a systems engineering method, involving various aspects, such as accident and damage modeling or system functionality and possible enablers, in order to fit into the bigger picture of more survivable systems in a highly uncertain mission environment.

Progress

Three main research areas have been identified as supporting work to this task. The first thrust involves the clarification of key words and concepts and the theoretical framework, which the methodology will be built upon. The second thrust includes all the efforts for obtaining a suitable modeling and simulation environment, given the revolutionary nature of the underlying concepts and premises. The third is the development and the demonstration of the methodology, using a baseline naval architecture.

Design for Resilience: The need for a theoretical framework

Resilience engineering can offer insight and research directions that may lead to answers regarding the design of more safe and survivable complex systems. According to the systemic view of how accidents occur, one can infer that a resilient response by the system would include the ability to efficiently adjust to non-favorable influences rather than to resist them. Such ability could be embedded as collection of internal functionalities and be the basis for certain active features for susceptibility/vulnerability reduction and recoverability increase. Automation and networks of sensing grids and information distribution might be possible enablers for enhanced reconfigurability and would lead to the essential functionality of a resilient system. In other words, a resilient system is expected to adjust its functioning prior to or following changes and disturbances so that it can go on working even after a major mishap or in the presence of continuous stress.

However, the traditional definition of resilience, as given by Holling (1973), contains a limited perspective on how a system can be safer:
"Resilience is a measure of the persistence of systems and their ability to absorb change and disturbance and still maintain the same relationship between populations or state variables"

It is implied that a system would be less vulnerable and more recoverable by being capable of absorbing adverse changes that affect its normal operating conditions. In other words, resilience at this point is presented as a measure of robustness. However, the vision of a resilient system hints to a set of expected system features and characteristics that go beyond the characteristics of a robust system. Thus, a resilient system is a robust system, yet a robust system is not necessarily resilient.

Robust vs. Resilient

As a first attempt to distinguish between robust and resilient systems, the following table has been constructed. For three conceptually different designs of the same system (baseline safe, robust and resilient) a breakdown of expected system functionality is provided, according to existing definitions for all three types. While safe systems mainly aim towards preventing system (including human life) without any provision on the mission by avoiding or resisting threats, robust designs additionally seek ways to partially recover the mission after assuring system survival. Instead of employing actions to withstand the adverse effects of a threat, robust systems should just be inherently insensitive to change by design, thus without requiring any particular attention for avoiding/resisting the threat.

For the resilient system, it is additionally expected that it is possible for threats and system status to be sensed, with the ultimate purpose to actively recover the mission after system loss is prevented. Adaptability to change is key distinctive feature of the resilient system compared to safe and robust designs.

Taking the above into account, there has been an attempt to extend the definition of resilience to include all three types of survivability. Table 2 contains a list of the

proposed characteristics of a system that can be referred to as resilient, along with the corresponding functionality of two example systems, a naval system and the human body.

Table 1: System functionality for all three types of survivability

	Architecture Capability		
	Type I Survivability **Susceptibility**	Type II Survivability **Vulnerability**	Type III Survivability **Recoverability**
Survivable/Safe	Avoid	Withstand/Resist	Prevent system loss
Robust	Expect	Mitigate/Neutralize	Prevent loss and partially recover mission (passive response)
Resilient	Sense	Adapt	Prevent loss and fully recover mission (active response)

Table 2: Proposed extension of resilience definition, along with two illustrating examples

Survivability type	Resilience definition	Characteristic	Engineering System (e.g. threat missile attack)	Human Body (e.g. threat = flu)
Susceptibility	Extended to systems engineering	Ability to monitor threat	Use Radar detection to locate origin of missile and direction	Be aware of people with flu symptoms around you
		Ability to sense threat	Estimate speed of missile and direction	Detect for flu symptoms on your body (sore throat, intense cough, etc)
		Ability to warn about threat	Activate warning indications of display panel if missile headed towards ship	Change body temperature, body fluids, muscle aches, headache, fatigue
		Ability to adapt to change for more effective system persistence to (adverse) change and system recoverability	Reposition ship to threat, maneuver away, activate anti-missile defense systems	Change of body temperature, sweating,
vulnerability	Basic Resilience definition	Ability to persist to change	Activate missile defense systems, utilize performance for avoiding or mitigating change	Activate mechanisms that fight against virus effects and spreading, enhancing immunity
		Ability to absorb change	Shielding, material absorbing ability, compartmentation, fault isolation	Virus isolation, flu shot effects, finite recovery time
Recoverability	Basic Resilience definition	Ability to maintain subsystem connectivity and integrity	Apply system redundancy, reconfigurability	Basic functions maintained through redundant organs (e.g., if cannot breath from nose, then use mouth)
	Extended to systems engineering	Ability to prevent system loss and fully recover the system's mission	Recollect and redefine mission objectives that are achievable through the new configuration (mission flexibility)	Be able to perform basic functions, vital body functions unaffected by virus spread,

Main Hypothesis Revisited

The central hypothesis of this task has been formulated and states the following:

A more resilient system demonstrates improved survivability than a robust system, mainly by incorporating engineering system reconfigurability, if subject to the same intelligent or natural events that affect system operations.

Improved safety and survivability come at some expense in overall system performance, acquisition and maintenance costs.

The objective is to develop and optimize two system architectures, one to exhibit the features of a robust system design and the other to be the resilient system. In order for this hypothesis to be tested and supported, not only a design approach is required for the acquisition of the two solutions, but also a complete evaluation framework that will include the necessary metrics for confirming that a solution is resilient, according to the previously discussed concepts and premises of resilience engineering.

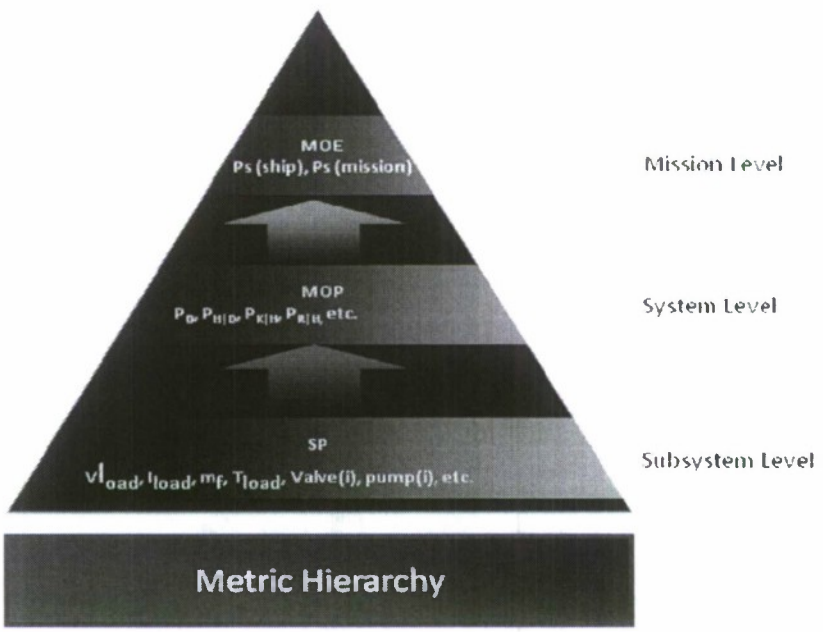
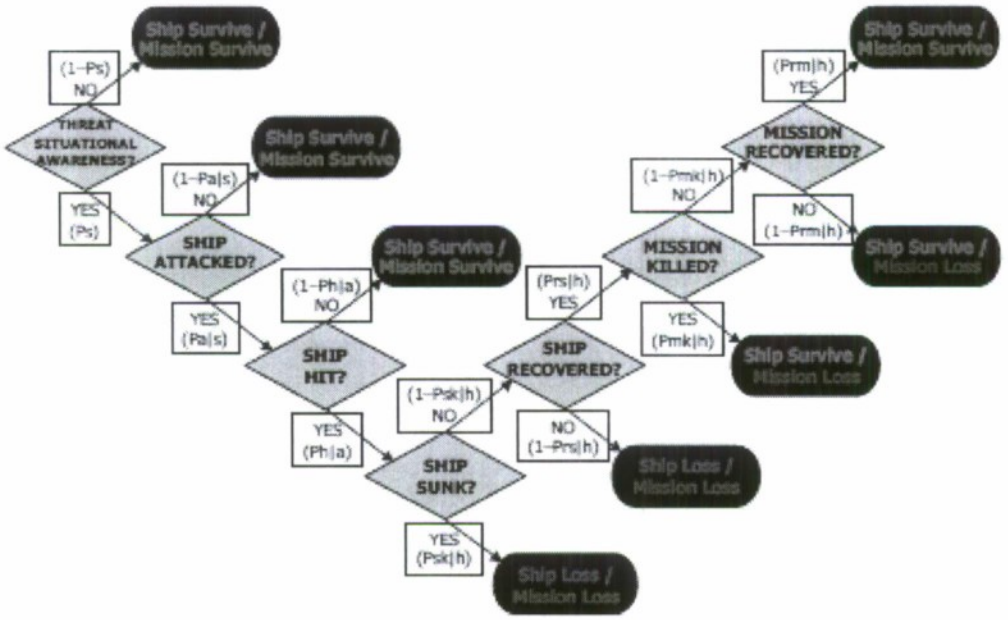


Figure 15: Metric hierarchy for survivability assessment framework

The first steps to quantitative framework development for Resilience Engineering

Given that the ultimate objective of developing resilient systems is to improve overall survivability and mission effectiveness, a convenient starting point for the framework development would be one of the existing state-of practice methods for survivability assessment. One of the most common is the Total Ship Survivability Assessment Method (TSSA) (Yarbrough & Kupferer, 2002). There is a certain hierarchical distribution of metrics that allow for total ship and mission probability of survival calculations (Figure

15). At the lower level there are the subsystem metrics, or *System Parameters*. The next level includes the measures of performance, or MOPs that involve conditional probability calculations based upon the values of the SPs. The higher level is represented by the MOEs, including the aggregate metrics for high level mission and system survival assessment.



Equation 1.

$$P(ShipLoss) = 1 - P_{s} P_{a|s} \left[1 - \sum_{i=0}^{i_{max}} P_{ih|a} \left(1 - P_{sk|ih}\right) P_{rs|ih} \right]$$

Equation 2.

$$P(MissionLoss) = 1 - P_{s} P_{a|s} \left[1 - \sum_{i=0}^{i_{max}} P_{ih|a} P_{rs|ih} \left(1 - P_{sk|ih}\right) P_{rm|ih} \right]$$

Figure 16: Kill Chain and linking of the MOPs to the MOEs

TSSA relies upon the concept of the Kill Chain. The entire incident is broken down into subsequent time epochs, at the end of each one; there is always an event with a set of possible outcomes. Calculation of the conditional probabilities for each event outcome is necessary for the MOE aggregate metric estimation for the total probabilities of mission *P(MissionLoss)* and system *P(SystemLoss)*survival. An example of a kill chain is depicted in Figure 16. As the transition from MOP to MOE seems more straightforward, the real challenge at this time is to develop, evaluate and select the appropriate SPs and effectively convert them into the corresponding MOPs.

Subsystem level metric development

The basic system information that this framework is called to be able to describe, should contain the following:

- System Geometry and Specifications
- Engineering Subsystems
    - Performance ratings
    - Cost
    - Connectivity
    - System states
- Mission profiles
    - Goals and objectives
    - Figures of merit
    - Time frames
- Threats and hazards
    - Types
    - Impact data and models
    - Failure rates and modes
    - Response and recovery times

The above requirements for metrics development are quite generic and there is a need at this point for a systematic procedure that will lead to a set of metric alternatives. This process will be based on associating the requirements to the metrics. One possible approach is to apply the "Goal-Question-Metric" process. Ideally, stakeholders and consumers would pose the goal and the question, and engineers would define the metric.

Here is a simple example on how the GQM process can be applied: Let's assume that the goal imposed by the stakeholder is to "increase product reliability". As a consequence, a relevant question can be formulated as follows: "What is the current fault removal rate compared to earlier releases of this product"? One metric that answers to the previous question is the current percent and number of faults removed by lifecycle phase and fault severity for this product release. Another possible metric is the previous percent and number of faults removed by lifecycle phase and fault severity for earlier releases.

There are four main categories under which every metric can be classified. According to its mathematical description, a metric can be at least one of the following:

- **Ratio**: We divide one quantity over the other, with the numerator and denominator are mutually exclusive
- **Proportion**: We divide one quantity over another, with the numerator and denominator are not mutually exclusive and numerator is part of the denominator
- **Percentage**: It is a conversion of a proportion in terms of –per hundred- units
- **Rate**: A rate represents the dynamic rate of change of the phenomena of interest over time

Returning to the dynamic behavior of a system with respect to its survivability, a generalized response can be captured by Figure 17:

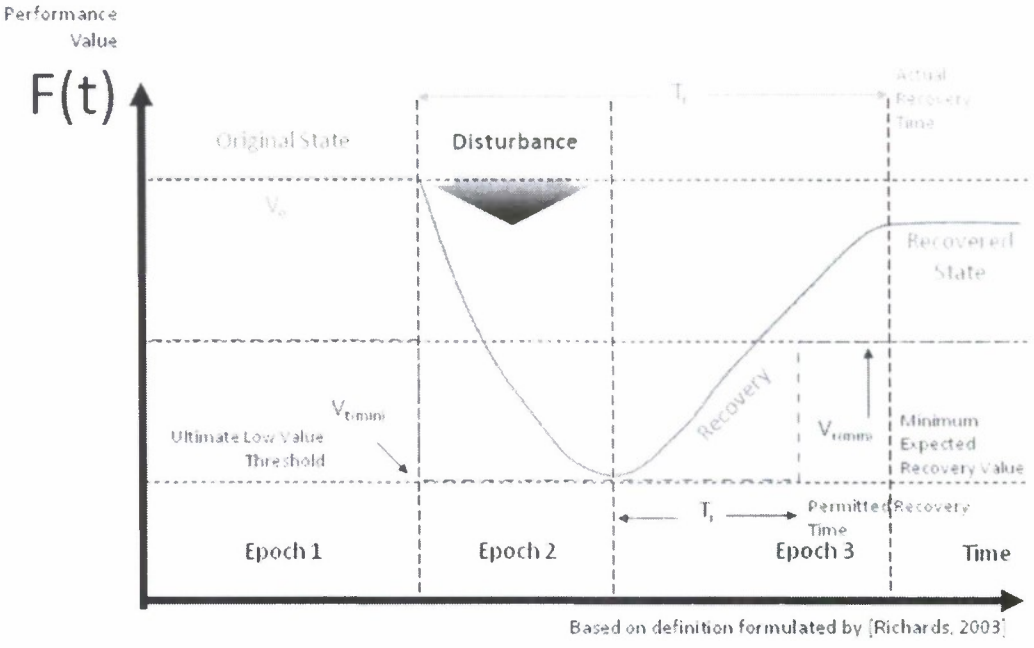


Figure 17: Generalized value-based survivability definition

According to this definition as provided by the above figure, there can be at least two metrics of interest derived. Following the GQM approach, the two goals of the system can be:

- G1: Improve ability to minimize utility loss
- G2: Improve ability to meet critical utility thresholds

Two questions can be formulated as an attempt to explore possible metrics for the measuring of the system's ability to satisfy the two previous goals:

- G1→ Q1.1: What is the utility loss due to performance degradation
- G1→ Q1.2: What is a time dependent average measure of the utility loss due to performance degradation
- G2 → Q2: To what extent is the threshold satisfied, even after significant performance degradation?

At this point, metrics can be formulated as possible answers to the goals and questions:

- G1→ Q1.1→ M1.1: The utility loss due to performance degradation can be expressed as

$$U_L = U_0 - U(t)$$

- G1→ Q1.2: → M1.2: The time weighted average utility can provide a cumulative time dependent measure of the system's response due to performance degradation

$$\bar{U}_T = \frac{1}{T}\int U(t)dt \text{ with } \bar{U}_L = U_0 - \bar{U}_T \text{ as the time weighted average utility loss}$$

- G2 → Q2 → M2: Threshold availability A_T, where, TAT is the total Time Above Threshold

$$A_T = \frac{TAT}{T}$$

These metrics can also be attributed to describe how robust is the design, given that robustness is the Insensitivity of system value delivery to changing contexts (Richards, 2009). However, there are other characteristics of a resilient system that cannot be captured by metrics that mostly refer to system robustness. For instance, adaptability to changing mission requirements is not explicitly reflected, thus there is need for an extended metric set that will seek to address all the other features that can make a robust design to become more resilient.

One thought is to distinguish effects based on their origin. While the system is suffering from sudden performance degradation, immediately there can be two types of change identified: Change due to disturbance against value delivery (adversary) and change due mission updates or system reconfigurability (favorable). A change can be also described as a time dependent rate; therefore in this case there can be two time dependent rates of change that describe opposing actions. A total rate will capture total system ability to absorb change.

Except for balanced rates and time weighted value differences, the resilience characteristic of maintaining system shape and status can command for metrics that display a count of available entities or their health status. For instance:

$$S = \frac{N_{Existing\ Connections}}{N_{Commanded\ Connections}}$$

can provide a ratio that works as a shape factor, which describes both component multitude and availability. The framework development is currently under construction, with more resilience characteristics to be addressed and with more attempts of utilizing the GQM method to formulate metrics for their quantification.

Formulation of analysis tools and integration into a M&S environment

As stated in earlier reports, for a complete survivability assessment, the original plan was to combine individual models of engineering subsystems to produce integrated models for dynamic simulation. The most significant part of this particular effort will be a routine that models and investigates damage propagation on a naval system. The damage model engine will analyze (damage prediction) and visualize (on the Paramarine ship PM) the damage propagation throughout the particular architecture. In this task, a total ship systems operations M&S environment is the desired outcome, including an investigation of damage generation and propagation. An overview of this attempt is given by Figure 18.

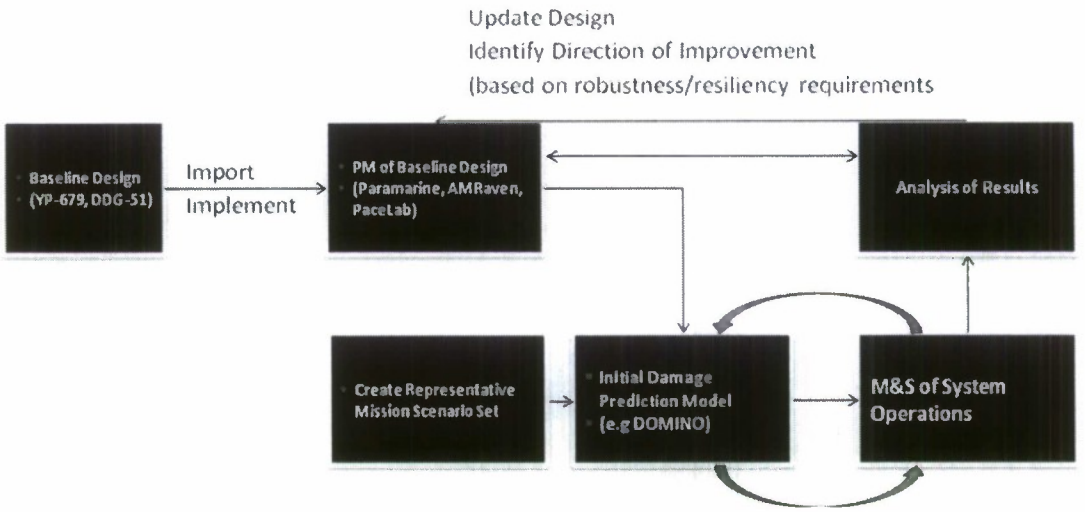


Figure 18: Modeling &Simulation Environment for Survivability Analysis

As a baseline, a notional naval ship design is required to be the starting point for the implementation of the method. At this point, the baseline design has been finalized. In earlier sections of this report, two geometry baseline designs were completed in Paramarine, however for survivability-based design method; the notional destroyer design has been ultimately selected.

It would require a certain amount of information for the creation of a ship baseline, such as ship geometry, engineering subsystems, acquisition and operations cost breakdown, mission profiles, threats and hazards and local environmental conditions. Most of this information is not available, therefore a large amount of assumptions is being made and configurations are being formulated according to prior knowledge about similar ships or engineering intuition. This will be the case for the engineering plant that is a modified version of an IPS configuration.

Before looking into the engineering systems, there have been updates concerning the damage prediction model scheme. The initial damage prediction module has been provided by the Navy (DOMINO) and it is now understood how this enabler can be integrated into the modeling process. Within DOMINO a connectivity schematic of the systems architecture is being created, in order to specify damage propagation due to initial connectivity. At the end of every time step of the systems simulation, DOMINO will be exchanging information with the dynamic simulation module and feeding this back to Paramarine, in order to conduct other static analyses of interest.

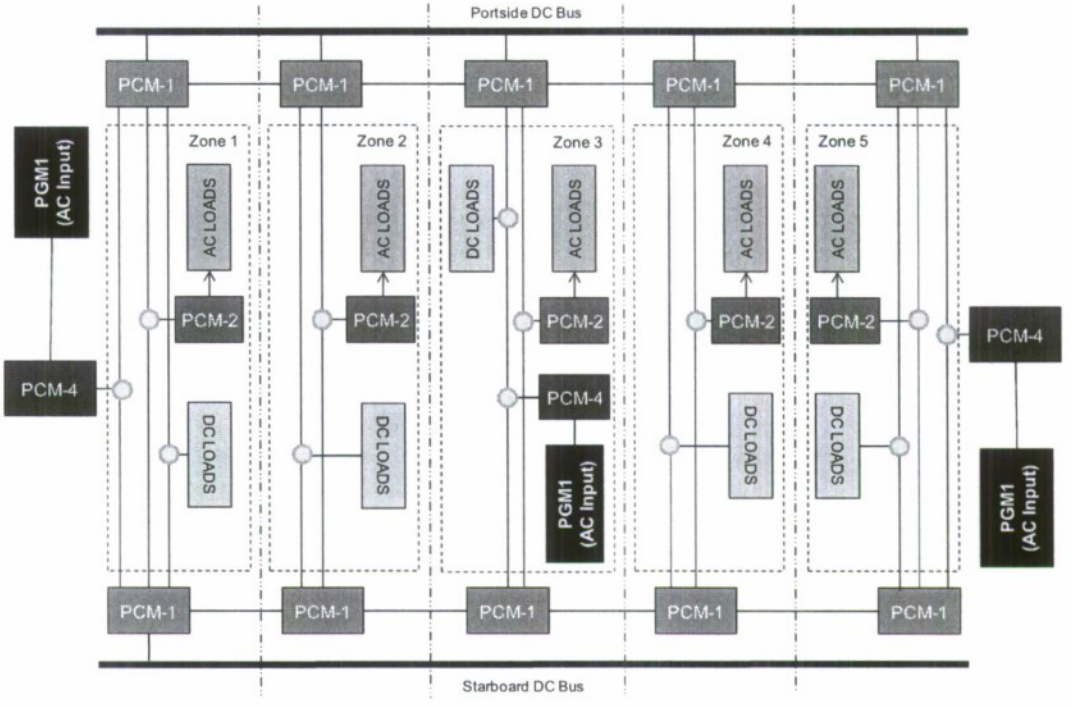


Figure 19: Configuration for Modeling &Simulation of ship engineering systems

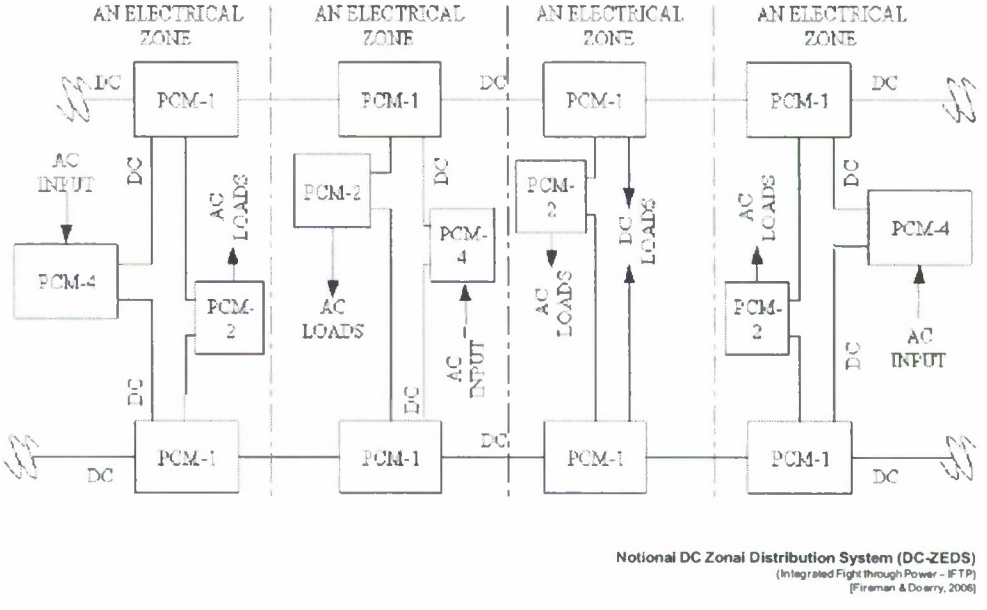


Figure 20: IPS architecture for DC electrical system (Fireman & Doerry)

The engineering system architecture is heavily based upon the IPS architecture, as presented by Fireman & Doerry (2006) and is shown in Figure 19. The original IPS

architecture with four zones refers to a DC electrical power distribution system and is described in Figure 20. The most significant changes on this IPS original architecture were the addition of an extra zone to conform to the size of the notional destroyer ship and the addition of DC or AC load placeholders to all zones in order to be able to size accordingly at the time that the design method will return the most resilient solution.

Sample stability study for the YP-679 hull

As a sample analysis demonstration, a stability analysis study is presented at the following, to demonstrate some of the capabilities of Paramarine and one of the steps of the complete damage and survivability assessment process. After the completion of the ship's hull, the hull had been subdivided into compartments. The following types of contents for each storage compartment were specified:

- Sea water, fresh water
- Diesel fuel
- Waste fluids
- Lube oil

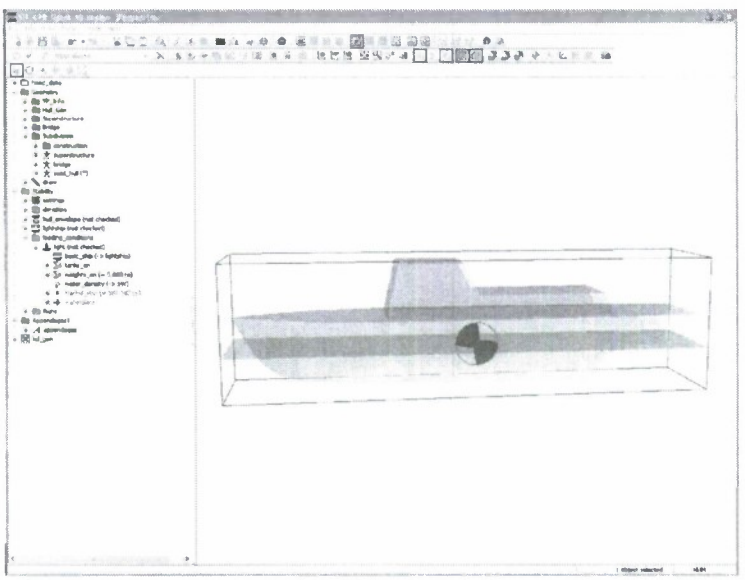


Figure 21: Waterplane of the intact ship and CG location

The next step was to specify the fullness of volume (e.g. fluid extends to 95% of compartment volume) and then the total weight of compartment content was calculated

$$\text{Weight} = (\text{Comp Volume})*(\text{Fullness})*(\text{density})$$

Summing all weights from all compartments the total weight of the contents of the ship has been estimated. The following two cases for stability analysis have been examined:

For the intact ship

This is the normal operating state of the ship, under the given loading conditions. CG is calculated based on the total weight from all compartments. According to loading conditions and CG location, the waterplane of ship equilibrium is calculated.

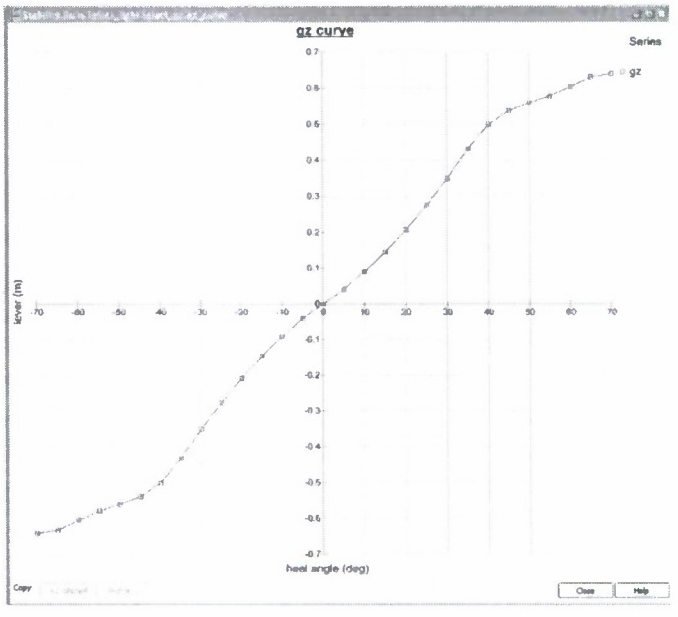


Figure 22: GZ stability curves for intact ship

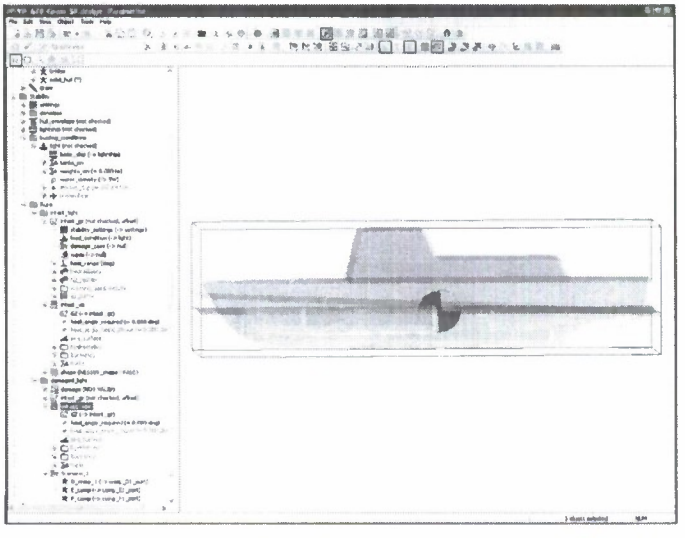


Figure 23: Waterplane of the damaged ship and CG location

Based on the same loading condition, GZ-curves for describing ship stability can be obtained directly from Paramarine. For a given heel angle (0 to 70 deg), the GZ distance can be estimated and is representative of the tendency of the ship to become unstable. It

is pretty obvious that if some compartments are flooded, then their contribution to building up the necessary buoyancy is lost, therefore affecting ship's ability to return to an equilibrium position after an angular perturbation.

For the damaged ship

Assuming a missile attack, three compartments on the port side are flooded. At this condition, it is assumed that the flooding has been caused by the immediate damage effects of the missile attack along with the latent effects of damage propagation that had affected the neighboring compartments. This set of changes will change the loading conditions. The waterplane of ship equilibrium is recalculated and shifted, since the equilibrium position depends on the new loading conditions. That results to an inclination of the waterplane, in a "nose-down" response as shown in Figure 23

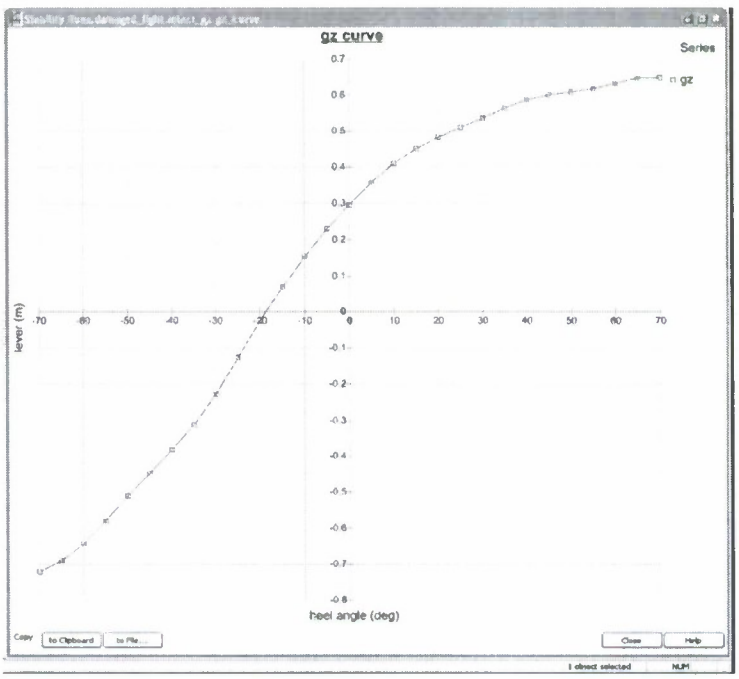


Figure 24: GZ stability curves for damaged ship

Shift of the waterplane also affects the stability of the ship under the updated loading conditions. The GZ curve in Figure 24 indicates that for a small positive heel angle, the GZ distance is quite larger, implying the larger magnitude of moment that can be generated and be responsible for rotating the ship away from its equilibrium position. Thus, not only the equilibrium state of the ship has shifted, but also the ship's ability to absorb any possible perturbations from its equilibrium state.

Future Work

Regarding the framework development, the identification of possible metrics for resilience is still under progress. However, there is also a need for supporting a process of down selecting metrics for final use. The basic criterion for this task is the sensitivity of

the identified metrics, with respect to resilience characteristics, as these have been clarified by the updated resiliency definition. A simple thermo electrical system can serve as the dynamic platform where a sensitivity analysis can be conducted. The final list would become the framework that will be used for the subsequent research questions on the larger scale system simulation.

Concerning the M&S environment, the most important task to be completed in the near future is the engineering systems simulation model in Simulink. Most of the power generation and distribution system is currently finalized, yet the cooling system that will be adjusted for the number of loads of the power system is still under construction. Next task after this is the integration of DOMINO in the M&S environment, and its interfacing to the DDG-51 Paramarine model for updating the damage propagation status. This total simulation environment will be eventually used for the method development and demonstration.

Cost Expenditure Status Report

Figure 25 shows the projected expense (based on the total budget of $155,305) over the contract period and actual expense for the month of October, November and December. We are trying to keep our cost on the target. The difference between the projected and actual expenses is due to that the potential travels have not been scheduled.

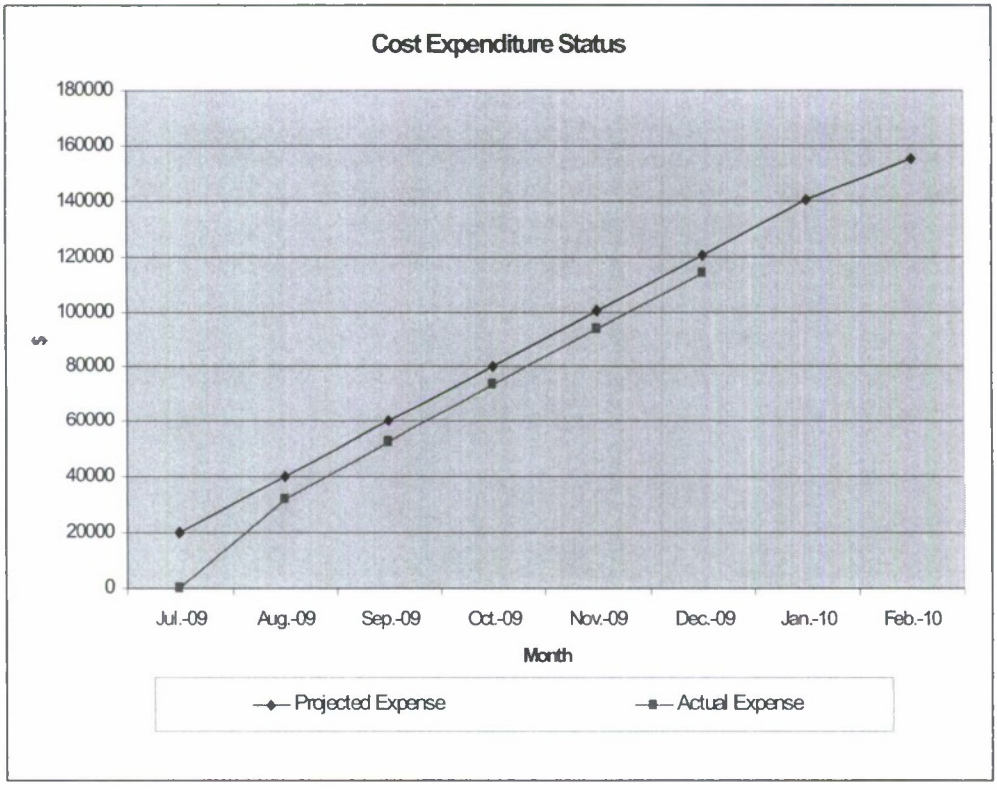


Figure 25: Cost Expenditure Status