3-2006

Discrete event simulation Model of the Ground Maintenance Operations Cycle of a Reusable Launch Vehicle

John T. Pope III

DISCRETE EVENT SIMULATION MODEL OF THE GROUND MAINTENANCE OPERATIONS CYCLE OF A REUSABLE LAUNCH VEHICLE

THESIS

John T. Pope III, Captain, USAF

AFIT/GLM/ENS/06-14

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

AFIT/GLM/ENS/06-14

DISCRETE EVENT SIMULATION MODEL OF THE GROUND MAINTENANCE OPERATIONS CYCLE OF A REUSABLE LAUNCH VEHICLE

THESIS

Presented to the Faculty

Department of Operational Sciences

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Logistics Management

John T. Pope III, BS

Captain, USAF

March 2006

AFIT/GLM/ENS/06-14

DISCRETE EVENT SIMULATION MODEL OF THE GROUND MAINTENANCE OPERATIONS CYCLE OF A REUSABLE LAUNCH VEHICLE

John T. Pope III, BS

Captain, USAF

Approved:

_____	_____
Alan W Johnson, Ph.D. (Chairman)	Date
_____	_____
Stephan P Brady, Ph.D. (Member)	Date

Abstract

The Air Force uses a family of expendable launch vehicles to meet its spacelift needs. Unfortunately, this method is not responsive: months of preparation are typically required and launch costs are high. Consequently, the Air Force seeks a reusable military launch vehicle that can be launched inexpensively and quickly regenerated between flights. Air Force Research Laboratory personnel desire a tool to help evaluate candidate designs and perform tradeoff studies necessary to acquire a launch vehicle that will achieve Air Force goals. The objective of this research was first to develop a conceptual model of maintenance operations needed to regenerate a launch vehicle between flights, and then to translate this conceptual model into a discrete event simulation tool. This research was accomplished concurrently with Stiegelmeier, who focused on vehicle prelaunch operations.

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Dr. Alan Johnson, for his guidance and support throughout the course of this thesis effort. His insight and experience was certainly appreciated. I would, also, like to thank my sponsor, Bruce Thieman, from the Air Force Research Labs for both the support and latitude provided to me in this endeavor. I would also like to thank Adam Stiegelmeier who was instrumental in helping code the Graphical User Interface as well as finding errors in my code and model. I know we both learned a lot working on our model together. Last, I'd like to thank my family. My wife is the most understanding person I know. She never hesitated to give me alone time when I needed it. There were a lot of late nights and conversations that she did not quite understand but she was there through it all. Thank you.

John T. Pope III

Table of Contents

Page

List of Figures

Page

List of Tables

Page

DISCRETE EVENT SIMULATION MODEL OF THE GROUND MAINTENANCE OPERATIONS CYCLE OF A REUSABLE LAUNCH VEHICLE

I. Introduction

Background

The Space shuttle system of reusable launch vehicles is reaching the end of its service life. To sustain our efforts in space and to encourage growth in the area, a reusable space launch vehicle must be developed that has a shorter down time between launches. In order to shorten the time that a proposed system is on the ground, a method should be devised that can improve the ground maintenance flow of the system as well as the time to integrate a payload and actually launch the vehicle. Space has become a critical part of the United States' warfighting capability and requires that future space systems become more responsive than the current systems of reusable and expendable launch vehicles (Brown 2003). This launch vehicle would be considered Stage1 of a two stage craft similar to the Space Shuttle. The tasks that the Air Force must increase it's capabilities to perform include, but are not limited to, GPS satellite launch to cover areas with pinpoint powerful GPS that is less likely to be jammed by the enemy, communications satellites, and tactical response to unwanted aggression with conventional projectile placement on demand.

> Research in the area of space transportation systems has focused primarily in the design and manufacture of the vehicle components: propulsion, materials, thermal protection, and controls to mention a few. In most cases, the operation of the vehicle and all phases of the facility/equipment component were ignored early on in design or had very little

> consideration. However, experience with previous systems has shown NASA and industry, that operations has the most significant effect in the life cycle cost and performance of a space transportation system. (Zapata and Ruiz-Torres 2000)

Problem Statement

There has been much written about the complete cycle of flight time and regeneration time for the Space Shuttle and other launch vehicles. However, there has not been much focus on the maintenance time which is one of the most important aspects of the operation cycle of a space craft. Before another reusable launch vehicle is developed, the Air Force must address maintenance ground time to determine how often a vehicle can be expected to launch. Based on that information, fleet size, manpower, and capabilities of launch facilities can be determined during the design phase, eliminating waste and unrealistic expectations. A model of the expected maintenance tasks will provide a dynamic method to determine where improvements can be made and where the money should be spent to have the biggest impact on decreasing the time between launches.

Investigative Questions

No models currently can assess in detail the maintenance time required between launches for an Air Force-developed booster. The closest data that can be found to probable maintenance activities is limited to either the Space shuttle program, one of the expendable launch vehicle programs, or an aircraft program. From this perspective, there are several questions that will be addressed in this paper:

1. What generic functions, or sequence of actions, describe Reusable Military Launch Vehicle (RMLV) maintenance?

2. How do these RMLV maintenance operation functions compare to aircraft, Expendable Launch Vehicle, and Intercontinental Ballistic Missile (ICBM) maintenance operation functions?

3. What are the RMLV design drivers that will influence RMLV maintenance operations, and how will these drivers affect the relationships, number, type, and duration of RMLV maintenance operations activities?

4. How can these RMLV design drivers and maintenance operation activities be incorporated into a discrete-event simulation model that captures a baseline RMLV maintenance operations sequence?

Research Focus

The focus of my research will be to develop a method of showing the maintenance flow of a launch vehicle that can be integrated with post-flight, pre-flight, and space operations. Keeping a focus on compatibility will give the user a valuable tool in assessing mission capabilities in regards to operations tempo. This will allow for the planning of fleet size, manpower requirements, and, to a lesser extent, facility requirements.

Methodology

This research will assess the current maintenance process in the Space shuttle system as well as that of aircraft maintenance organizations. After examining several maintenance flows, this research will then will determine which actions most likely represent the tempo, size, and complexity of the launch vehicle under consideration. This will enable a model to be constructed that will give a good measure of the time that should be expected for completion of the maintenance cycle during mission operations. A Delphi panel will be constructed utilizing expertise from a range of relevant fields that

will ensure that the model captures the best maintenance flow representing a reusable launch vehicle maintenance cycle.

The model will be exercised using process times from experience, estimated times from experts in the Delphi panel, and reasonable estimates where unknown processes leave gaps in information available. A selection of modules will be randomly selected, and will be set to extreme values both high and low to ensure the model responds reasonably to a change in input. Also, the model will be exercised by several maintainers to ensure that it produces a reasonable output based on their similar experiences in maintenance and that it works in a predictable manner.

Assumptions/Limitations

The biggest limitation faced is that of information. Little data exists on the maintenance cycle of existing systems. Since the launch vehicle shares some commonality with the Space shuttle, but is being designed to be much simpler, the information available on Space shuttle maintenance operations will be of limited use. However, there is a lot of information about the Space Shuttle and the maintenance problems that go with it. The Air Force is attempting to develop a launch vehicle that must be simpler to maintain.

As the final design of the launch vehicle remains to be completed, I will use several assumptions for the model that I build. One assumption is that the launch vehicle will have vertical takeoff and horizontal landing capability. Since the landing is the most important aspect of the process in regard to maintenance, this assumption is important. In comparing aircraft maintenance to the launch vehicle maintenance, I will assume that the

same type of maintenance that can be performed on an aircraft can also be performed on the launch vehicle in a horizontal state. That is, the launch vehicle does not have to be rotated to a vertical position to perform the same maintenance that would be expected to be performed on an aircraft.

When the launch vehicle arrives at the maintenance facility, I am assuming that it will be rolling on its own gear, towed by a wheeled vehicle. The safety pins are installed prior to arrival as part of the post-flight cycle. Also, the tanks and motors will be dried prior to entering the maintenance cycle. The vehicle will have cooled to a temperature such that maintainers will be able to work on it. Maintenance ends when the launch vehicle is ready for prelaunch activities. The payload and all integration of that payload will be handled after the maintenance cycle has been completed. That is to say that maintenance does not include mating a payload to the craft or fueling and launching it. The vehicle will be ready for integration or storage upon completion of maintenance.

Another assumption that I make is that the Air Force will treat this vehicle as they treat regular aircraft. That is, the maintenance practices seen on the flightline will be very similar to the maintenance practices utilized on the launch vehicle.

Implications

A military vehicle, once proven, becomes a vital part of the Air Force inventory. To have a space vehicle that has an unpredictable maintenance cycle is unacceptable. “The use of discrete event simulation to model the Space Shuttle began as early as 1970 before the shuttle was approved for development.”(Schlagheck and Byers 1971). That initial work suffered from a lack of an established baseline for what the shuttle

architecture would actually be (Cates, Mollaghasemi et al. 2002). The current Space shuttle has too much variability in the time it takes to prepare for launch. Models showed after testing of the Space Shuttle that it would have a much longer maintenance cycle than initially thought (McCleskey 2005). This was due to a much higher maintenance time than was anticipated. In addition, it has far too much down time to be relied on by the military. Understanding the maintenance cycle of a launch vehicle is vital in understanding the possible contributions such a craft can make in any capacity. This maintenance cycle, once fully understood, will pinpoint areas for further study and improved efficiencies.

II. Literature Review

Chapter Overview

The purpose of this chapter is to provide some background information on the literature that is available and relevant to the topic of this thesis. This literature review will touch on several systems that should give some insight to the maintenance practices of various systems. The Space shuttle maintenance will be evaluated for comparisons to the expected maintenance flow of a launch vehicle. Large-aircraft maintenance practices are also looked into. Existing models of space launch vehicles are explored for relevance.

Introduction to Maintenance

Since the inception of air travel, maintainability has been increasingly important as aircraft have become more expensive. The Wright brothers performed their own repairs and were not concerned with high sortie rates. If the pilot wasn't the mechanic, he had knowledge of the entire aircraft and was able to make repairs as required. The Wright brothers designed, flew, regenerated, and repaired their aircraft. Maintaining the Space Shuttle is too complicated for one person to perform alone. Further, the Space Shuttle was designed to be as safe as technology could make it. This necessitated compromises in the maintainability and serviceability of the system. During the height of Shuttle flights, it still took several months for maintenance between flights. The overlooked maintainability of the Shuttle during the design phase has made the Shuttle too expensive and too time intensive for it to meet Air Force space launch needs.

A new system is in order. The system should be dynamic and responsive. In order to be responsive and able to support fast paced operations, changes must be made. During the design phase of the next launch vehicle, the maintenance cycle must be defined and carefully reviewed to limit the time and manpower requirements during this time of increased operation tempo and ever decreasing pool of available manpower. In order to understand what constitutes the maintenance cycle, we must know all the processes that go into launch vehicle operations. The preflight operations cycle includes mating of the payload, final testing of connections, fueling, and launch operations. The flight operations cycle includes all actions that occur after preflight and end with landing operations. That is, everything where the launch vehicle is above the earth. The post flight operations cycle includes operations such as placing the craft in a safe-for-maintenance condition. Safety pins, lanyards, grounding cords, and de-fueling operations are contained in this portion.

What is left over from the other operations listed is the maintenance cycle. This cycle will include all scheduled maintenance in the form of inspections and periodic replacement and checking of various components and parts. The scheduled maintenance will also include replacing consumable items with the exception of fuel. The maintenance cycle also includes unscheduled maintenance which has the largest level of variability. The unscheduled maintenance includes repairing damage from flight, replacing damaged components, and repairing the Thermal Protective System (TPS) as shown in Figure 1 for the Space Shuttle.

Figure 1 Thermal Protection Repair Activity around Orbiter Elevon Aero surfaces McCleskey, C. M. (2005)

Shuttle Maintenance Practices

Figure 2 depicts the originally planned vision for the Space Shuttle. It was evident when the Space Shuttle program was set to double its flight requirements that a study would be required to find out what resources would be strained. At the time, the external fuel tanks could be manufactured at the rate of 7 per year (Cates, Mollaghasemi et al. 2002).

> In 1999, at a time when NASA was considering plans to increase the flight rate from 7 flights per year to as many as 15 flights per year, the Kennedy Space Center began discussions with the University of Central Florida to develop a simulation model of Space Shuttle processing. (Cates, Mollaghasemi et al. 2002)

The Space Shuttle's flight requirements changed a lot from initial speculation where it was believed that the Shuttle could be regenerated in less than a week. The reality is that "…the Space Shuttle has demonstrated less than 20 percent of this capability, or an 80-percent shortfall in payload delivery performance."(McCleskey 2005)

Upon landing, the Orbiter has safety gear installed and the vehicle is evaluated. If this landing ended the eighth flight, the orbiter is prepared for transport to Palmdale, CA. This was to perform depot type of chronologically based maintenance that the facilities in Florida could not handle (McCleskey 2005). Now that the shuttle is near the end of its life-cycle, it is not cost effective to build new maintenance facilities at Kennedy Space Center to eliminate the need for ferrying the shuttle to Palmdale (McCleskey 2005). After being serviced at Palmdale, the shuttle has to be ferried back to Kennedy Space Center (McCleskey 2005).

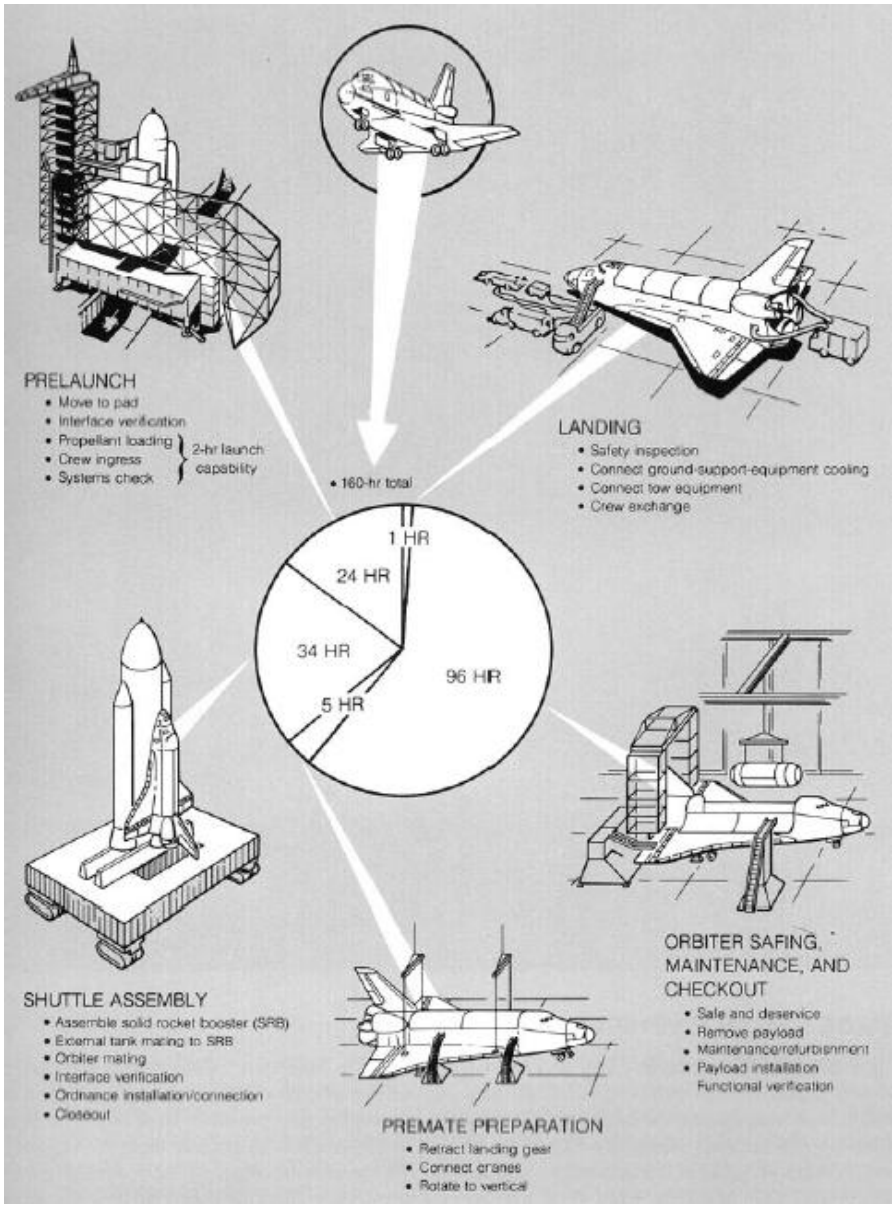


Figure 2 Original Concept of Shuttle Operations (Ca. 1976) McCleskey, C. M. (2005)

The Orbiter is then parked until mission requirements necessitate that maintenance be started. The Orbiter is then taken to the maintenance facility where it is placed on jacks to facilitate access to the lower surfaces. Maintenance stands are placed in key areas and the maintenance platforms built into the facility are moved into place.

At this point, an assessment is made on the TPS, tile by tile. This information drives the need to construct, individually, replacement tiles from stock that is kept on hand. These tiles are formed in-house. As the tiles are being worked on, other maintenance is going on around the same area. The gear is looked at, the tires are checked, or replaced as in the case of the rear landing gear tires which are replaced after every landing. The engine portion of maintenance requires removal of the engine for several reasons, according to (Rooney and Hartong 2004):

> The components that cause engine removal are:
>
> 1. Nozzle
> 2. Pre-burners
> 3. Hot Gas Manifold
> 4. Main Injection
> 5. Main Combustion Chamber
> 6. All pumps
> 7. Heat Exchanger
> 8. Other causes

Any TPS surface that covers an access panel must be replaced if the panel is opened. During this same time, all the fluid subsystems have to be checked and serviced. The payload area is fitted with adapters to mate the next payload to the orbiter. This takes place over several days. There are weight plates that must be installed to ensure

proper balance in the bay (McCleskey 2005). The required adapters and configurations are cumbersome.

Eventually, through improved software and improved maintenance practices, there could be a great reduction in the costs to analyze performance data and identify maintenance needs that would enable the system to generate maintenance requests while still in flight. That way, various maintenance activities could be staged to ensure rapid employment upon landing.

The recent NASA technical paper on Space Shuttle operations really showed where maintenance time was being spent:

> The highest concentration of Orbiter turnaround work was found in the Unplanned Troubleshooting and Repair category with 66,018 task-hours accumulated for the eight flows examined. In addition to the 30-percent contribution of thermal protection tile work on the vehicle, troubleshooting and replacement of system components, or line replaceable units (LRUs), also contributed quite heavily (22 percent). (McCleskey 2005)

Aircraft Maintenance Cycle

When a military aircraft lands, different things happen depending on the aircraft type. I have seen many different sequences of maintenance operations in my nine years of experience in aircraft maintenance. For F-15s and A-10s, the aircraft is met at the end of the runway where all the ordnance systems are pinned and the landing gear is checked and pinned. At that point, the aircraft is taxied to the flight line where it sits while maintenance is performed. The maintenance on these aircraft can be very wide-ranging. However, the aircraft is often merely prepared for the next mission. This usually includes

fueling, cleaning, and checking the external surface for damage. The avionics are not checked, nor are the flight controls. Also, the surface rarely needs maintenance. The aircraft forms are checked for needed maintenance activities such as aircrew reported problems. The F-22 uses a system where an aircraft in flight can notify personnel on the ground of problems so that the appropriate maintenance processes can be activated before the aircraft lands.

The bigger aircraft differ in several areas. They are not met at the end of the runway unless they have ordnance discrepancies. This is usually caused by aircrew error and is taken care of by weapons personnel who visually check the aircraft bay and pin the weapons to prevent inadvertent dropping. The aircraft taxies to the flight line where the maintenance is performed. The maintenance is quite a bit slower on these large frame aircraft as the stands and equipment used to perform maintenance are much heavier. The B-52 typically flies long missions with plenty of associated downtime to perform maintenance. Thus, maintenance can be scheduled to effectively use very few people. The isochronal inspections can take up to a month to complete. The flight line type of inspections can usually be completed during a 10 hour shift.

The B-2 program adds to the mix due to its unique surface handling requirements. Changing a landing gear tire can take from two to four hours. Changing Line Replaceable Units (LRU) can take several hours because an engine run is required to check out the avionics system once the swap is done. It then takes up to 30 hours to repair the surface that was damaged during panel access.

Engine changes on the B-2 also have some time intensive surface procedures. A block sealant, used to restore the firewall, has a 24 hour cure check. Overall, an engine change takes about four days to accomplish (Lee and Schmierer 1994).

The big maintenance task with the B-2 involves the special surface material. This material covers the aircraft completely enough that it must be damaged to access panels during maintenance. Further, normal flight damages parts of the surface routinely. Overall, about half of maintenance time is spent inspecting, repairing, and renewing this surface material (Lee and Schmierer 1994).

Differences in Aircraft and expected launch vehicle Maintenance Operations

Aircraft and the new launch vehicle have a lot in common in that they both move through the same atmospheric conditions. Of course, there are quite a few differences as well. The effect of maintenance differences is what concerns the direction of this research.

The (smaller) aircraft do not require much in the way of maintenance before the next flight and are typically placed into flight with only cursory inspections, fueling, and mission data input. Bombers, on the other hand, require more interaction by maintenance personnel. The larger bomber aircraft equipment are more similar to what is expected in a launch vehicle application. One difference is that the launch vehicle will not be fueled as part of the maintenance cycle. It will instead be fueled prior to flight during the prelaunch cycle (Stiegelmeier 2006). Since the landings will probably be much faster with the launch vehicle and use fewer, lighter tires, tire changes will become a larger portion of the maintenance footprint than in the bomber maintenance cycle (O'Malley

2006). Also, life support and all the maintenance and associated checks that go into making sure the B-2 will support an aircrew are unnecessary in the launch vehicle which is assumed to be uninhabited.

Shuttle Operations

The Space Shuttle has long been studied using simulation.

> The use of discrete event simulation to model the Space Shuttle began as early as 1970 before the shuttle was approved for development. That initial work suffered from a lack of an established baseline for what the shuttle architecture would actually be. (Cates, Mollaghasemi et al. 2002)

These early attempts at simulating the Space Shuttle were not very accurate in that the actual process they were modeling had not yet been developed. Further, the complications increased when NASA considered plans to increase the flight rate.

> In 1999, at a time when NASA was considering plans to increase the flight rate from 7 flights per year to as many as 15 flights per year, the Kennedy Space Center began discussions with the University of Central Florida to develop a simulation model of Space Shuttle processing. The doubling of the flight rate was expected to strain the existing workforce, facilities, ground support equipment, and flight hardware elements. The question was which parts would be strained and how much? (Cates, Mollaghasemi et al. 2002)

GEM-FLO

The Generic Environment for Modeling Future Launch Operations (GEM-FLO) is a model that provides an upper-level view of Reusable Launch Vehicles.

> The issues precipitating the need for a generic RLV simulation model were to analyze the operations performance of several architectures in a timely manner, and to provide feedback to the design community as to the operational ramifications of design decisions. To this end, the Generic Simulation Environment for Modeling Future Launch Operations (GEMFLO) was developed. (Steele, Mollaghasemi et al. 2002)

GEMFLO does not model a particular vehicle, but rather offers a high-level view of a generic vehicle's operational steps. Specific models do provide an easier path to higher fidelity analyses and more representative animation, which can be crucial to obtaining face validity (Steele, Mollaghasemi et al. 2002). GEMFLO is not as detailed as AFRL personnel require to make decisions about maintenance practices.

SOVOCS

SOVOCS data is restricted by International Traffic in Arms Regulations (ITAR) and/or Export Administration Regulations (EAR) and is subject to the export control laws of the United States. Performed by the Boeing Corporation, SOVOCS was a commissioned study by the Air Force Research Laboratory (AFRL) to determine the launch vehicle configuration. This study included modeling software to determine the rate at which a vehicle could be regenerated based on its configuration. In short, the study found that a launch vehicle could be regenerated in the 2-3 day time frame. The simulation pointed out that TPS and propulsion are the two critical tasks that drive the regeneration time. SOVOCS also found that aircraft had the desired operability. The study is a static. AFRL personnel cannot use it to examine how maintenance changes could affect the overall launch vehicle regeneration time.

Summary

Because of the special surface handling requirements and its size similarities, the B-2 offers an excellent opportunity to create a baseline model of the launch vehicle's maintenance cycle. The differences have been noted and can be dealt with in the model.

Further, the B-2 has a similar mission to the launch vehicle. It puts a payload in place during a longer duration mission, returns to be refueled and prepared for the next flight, and repeats as necessary. This makes the two systems a good match for building a model that can be checked with actual maintenance practices. Existing models are not robust enough or are too mired in detail and technical barriers to be useful for AFRL personnel to use. This lack of a good method of evaluating maintenance practices drives the need to create a model that works better in this capacity.

III. Building the Model

Process Overview

This chapter will explain why a simulation was used and cover building the model in Arena software as well as the process to selecting modules to populate the model. Arena was chosen since the sponsor of this research has access to the software and it is widely used by Department of Defense analysts. It has also been used in previous NASA research(Cates, Mollaghasemi et al. 2002). Arena offers a great amount of freedom to build models with flexibility for future expansion. This chapter will also describe the Delphi panel that was used to validate the conceptual maintenance model. Building the Graphical User Interface (GUI) will be explained on in this chapter as well.

Why Simulation?

A method was chosen to represent the maintenance cycle that would enable changes to be introduced and tested easily. "Simulation enables the study of, and experimentation with, the internal interactions of a complex system or of a subsystem within a complex system… Simulation can be used to experiment with new designs or policies before implementation, so as to prepare for what might happen."(Banks, II et al. 2005) Simulation refers to a broad collection of methods and applications to mimic the behavior of real systems, usually on a computer with appropriate software (Kelton, Sadowski et al. 2004). It is this ability to mimic a real system that I wanted to be able to present to my sponsor. Simulations can be detailed enough to provide actionable output

yet general enough to allow for changes to be introduced as technology or requirements change. It is this capability that makes simulation the best method to use in this case.

Beginning the Process

The Air Force needs a way to evaluate the early design alternatives for a reusable launch vehicle. Since the vehicle and its operations have many aspects that have yet to be determined, a simulation model offers the flexibility of making changes quickly to evaluate improved methodology.

Little literature exists on the ground maintenance portion of space launch vehicles. Sea Launch, a good place to look for integration, launch, and recovery, basically, has no maintenance of the type that a reusable launch vehicle would have (Boeing 2000). The same goes for any of the non-reusable assets like Intercontinental Ballistic Missiles. Further, there is no single person to turn to who knows the entire spectrum of ground maintenance for an as of yet undetermined vehicle. Building a model covering the entire ground maintenance operations cycle requires a lot of expertise. This expertise is spread over multiple career fields and throughout the civilian ranks. The best way to tap a geographically separated pool of experts is through the use of a Delphi panel (H. Murat Gunaydin 1998).

The Model Building Process

"The simulation-model building process… can be broken down into four phases."(Banks, II et al. 2005) Figure 3 shows the four phases. Phase one covers the

first 2 steps and is where the formulation and setting of objectives takes place. This phase is completed when the research question and investigative questions are settled upon. The second phase covers steps 3 through 7. It is this phase that constitutes the bulk of this methodology chapter. This is where verification and validation takes place as well as where the Delphi panel is utilized. The third phase covers steps 8 through 10 and is where the model is run. Finally, phase 4 is implementation, which is in the hands of the sponsor of this research.

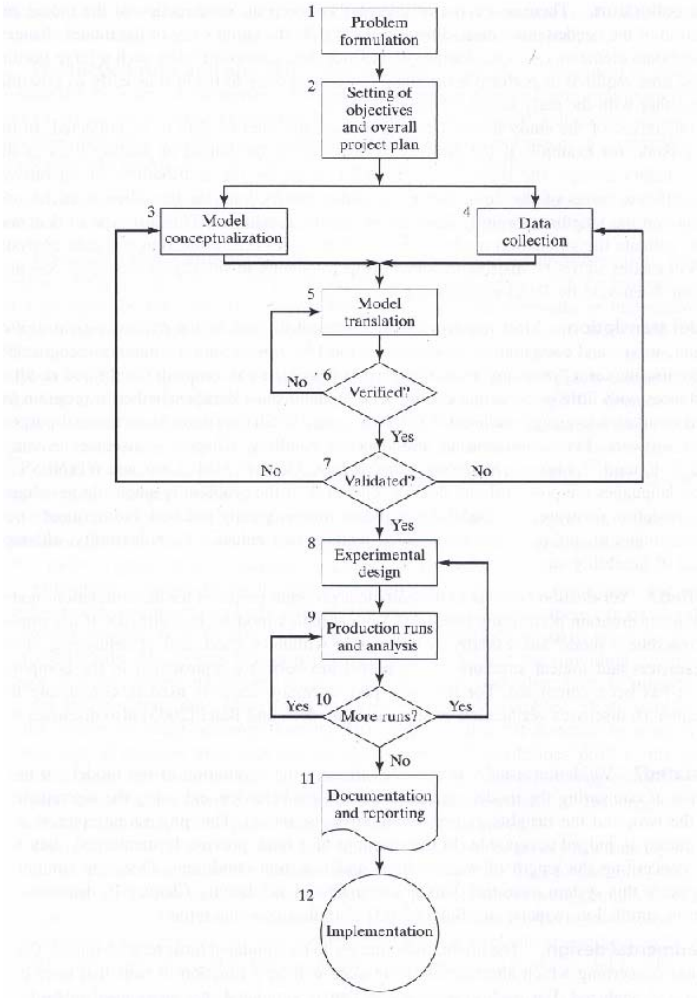


Figure 3 Steps in a Simulation Study

The Delphi Panel

A Delphi panel is a group of experts who meet on a regular basis to come to an agreed upon solution to a problem (H. Murat Gunaydin 1998). In our case, it is to develop a maintenance model. Stiegelmeier developed the prelaunch portion of the model (Stiegelmeier 2006). He and I worked together on many parts of the model and code together. We also set up the Delphi panel together and used it to refine and validate our respective prelaunch and maintenance portions of the vehicle regeneration model.

The Delphi panel begins with a rough stab at what processes are thought to belong to the area of interest. In this aspect, I developed a flowchart of the activities and their sequencing that followed aircraft maintenance methods that I have observed in 9 years working the flight line of various aircraft, including the A-10, F-15C, and the B-52H. I spoke with experts on the B-2 to ascertain the maintenance practices that differ from that of the B-52. The B-2 aircraft is similar in many ways to the launch vehicle in that it is a complicated system that has similar flight parameters and maintenance hindered by a special surface coating. By utilizing the B-2 maintenance practices and noting the probable differences, a reasonably accurate model could be constructed. Once the initial model was built, experts were found in various fields that could add experience from their areas of expertise to enhance and validate the model. Confidentiality was assured to get unbiased input without fear of reprisal.

Two experts in intercontinental ballistic missile maintenance were chosen. This brought vertical integration skills, heavy equipment movement, connection to boosters/motors expertise, as well as more than 20 years of experience to the process.

Two aircraft maintainers were included from Air Combat Command. One has experience with fighter aircraft, the other with heavy aircraft. The fighter maintainer brought rapid regeneration times, expedited maintenance practices, and quick motor change to the table with his 20+ years of experience. The other aircraft maintainer is equally experienced, but on the U-2, C141, C130, C135, SR-71, and B-52 aircraft. There were experts on the Space Shuttle on the panel as well. The five Shuttle people combined to deliver more than 80 years of space vehicle maintenance experience to the Delphi panel. There were several modeling and analysis people involved in the panel as well. These people ranged from the Air Force's Operationally Response Space program to component level model experts. The Delphi panel had 19 members.

With missile, small and large aircraft, Space Shuttle, and modeling experts involved, the Delphi panel was truly a well rounded panel with experts in all associated fields. The panel was presented with the proposed conceptual flows in the Delphi's first round -- see Appendix A for the model's Visio representation. The Appendix shows only pages 1 through 8 of the Visio representation. This is because pages 9 through 14 represented the prelaunch operations that Stiegelmeier covered in his thesis (Stiegelmeier 2006). The model was improved with each of three Delphi rounds, after which the panel recommended no further changes. The individual rounds and panel comments are shown in Appendices B, C, and D. Our responses to comments are italicized. The comments are listed by Visio page. This enabled panel members to refer to the Visio document to clarify comments. Also, notice that the comments are prefaced with a number. A number was randomly assigned to each respondent to mask his or her identity from other

panel members. The responses by these experts ensured that the model was built using expertise from all related areas. One of the challenges with using a Delphi panel is that over time, the panel members may become unable to devote further attention to the process. After it was apparent that the changes we were receiving were minor, we put in a final request that all members respond to either confirm that the model was a good representation, or recommend further changes. We received confirmation from most of the panel, and no suggested changes. At that point, we declared the panel complete and moved on to the next step which was building the model in Arena.

Building the Model in Arena

> Simulation refers to a broad collection of methods and applications to mimic the behavior of real systems, usually on a computer with appropriate software. In fact, “simulation” can be an extremely general term since the idea applies across many fields, industries, and applications. These days, simulation is more popular and powerful than ever since computers and software are better than ever. (Kelton, Sadowski et al. 2004)

The software that was used to build a model of the maintenance cycle is Arena 7.01.00 and is available off the shelf. The Arena software offers that “Arena is an easy-to-use, powerful tool that allows you to create and run experiments on models of your systems. By testing out ideas in this computer "laboratory," you can predict the future with confidence ... and without disrupting your current business environment.” (Kelton, Sadowski et al. 2004) Arena will allow some flexibility with maintenance times and distributions. This is important in modeling a system with no historical data to back up the results.

> The typical scenario for a simulation study entails developing a specific model of an existing system for the purpose of analysis. This begins with the capturing of knowledge by the simulation analysis from the system expert, including information of system structure and data, and continues to the modeling of that information and data at some level of abstraction, the running of model scenarios, and the subsequent analysis of the resulting model output data. Typically, this is accomplished of a system that is in existence, or nearly so, that has the requisite details known by which to construct and run a simulation. (Steele, Mollaghasemi et al. 2002)

Arena uses modules to represent processes and decisions. These processes can be populated with values ranging from a single value to most types of distributions. The many processes involved in the maintenance cycle of a notional launch vehicle lack historical data to quantify them. "There are three distributions that have application to incomplete or limited data. These are the uniform, triangular, and beta distributions" (Banks, II et al. 2005). The uniform distribution assumes that the process is random but that nothing else is known about process times. Experience in processes similar to most of the events that will occur during launch vehicle maintenance allows me to assign a likely minimum, maximum, and mode value. The triangular distribution can be used when assumptions are made about the minimum, maximum, and modal values of the random variable (Banks, II et al. 2005). In the following pages, the final model will be explained.

There are several modules in the front part of the model that will not be discussed in this paper. Those modules are used to determine the flow through the model past the maintenance portion. The paths involve the integration portion of the overall model, and are presented by Stiegelmeier (Stiegelmeier 2006). Figure 4 shows the first main part of the maintenance portion of the model. The first module assigns a value to the number of

motors variable. This prevents data from a previous run of the model from interfering with the current run. The next module is the connection of the tow vehicle to the launch vehicle. The connections are to tow the launch vehicle on its own landing gear as regular aircraft are towed. Following that module is the transportation time from the safing area to the maintenance facility. The fourth module seizes the maintenance bay. This prevents multiple launch vehicles from occupying the same space. The next module is the positioning module. This includes positional moves as well as hanger door openings and closings that may be necessary. The electrical grounding procedures follow that. The last module in Figure 4 is the disconnection from the launch vehicle.

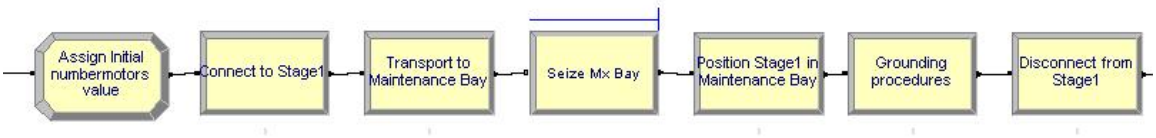


Figure 4 Arena Model

Figure 5 is the next section of the model. The first module is the process of interrogating the launch vehicle's maintenance data reporter to determine system health. This information will allow for maintainers to prepare for maintenance tasks down the line. The current generation of Integrated Vehicle Health Monitoring systems has the capabilities to report maintenance status prior to landing (John B. Shroeder 2006). In that case, this module would simply have a value of zero because the user would enter a constant value of zero to show that no time is used during the maintenance cycle to get the information. The positioning of all the various maintenance stands follows. At this point, aircraft style maintenance stands are assumed to be used. The module can be used

to represent the time it would take to position built-in stands as well, though. There will be electrical connections required to power up the various systems in the launch vehicle. The third module represents the time that is required for these connections. Again, this module is set up assuming a single connection point for external power. The separate module follows. This allows the model to depict multiple paths of parallel processes. When there is a chance that the processes can not be done in parallel, they were put in a serial branch. This ensures that if there is an error made in the ability to perform tasks in parallel sequence, the actual maintenance procedures will take less time than the model predicts. Battery testing is the last module in Figure 5. It is on the upper leg of a parallel branch.

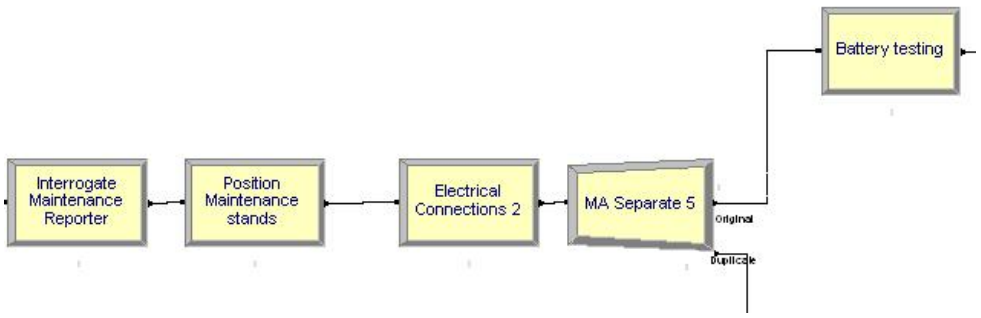


Figure 5 Arena Model (cont.)

Figure 6 depicts the first decision module of the maintenance part of the model. This module is controlled by a probability argument. If the batteries are good, they are charged in the far right module. If they are not good, they are replaced in the lower module and then charged in the far right module.

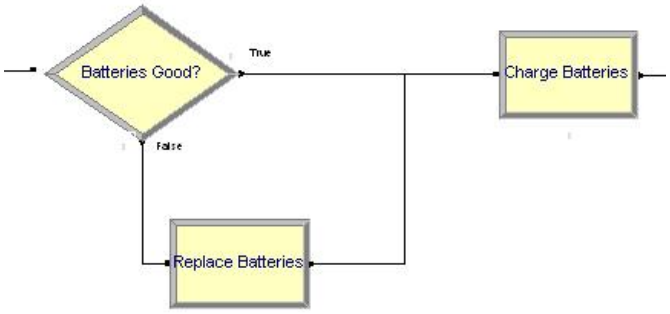


Figure 6 Arena Model (cont.)

Figure 7 depicts the lower branch portion of Figure 5 that is in parallel with the modules in Figure 6. The first and last modules are the split and recombining modules. They are the beginning and end of a set of processes in a parallel configuration. The second module allows for time to perform all electrical avionics testing. This is the testing to ensure the avionics package is communicating appropriately. The flight controls module follows. This is to test that the avionics system controls the flight surfaces correctly. The lower module allows for the removal of experimental data or telemetry information. The information that these modules represent can be adjusted to represent different processes as long as they fall into the general categories depicted.

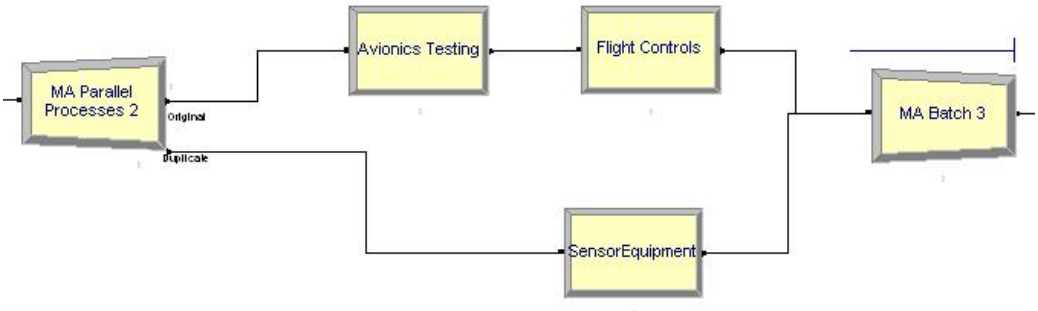


Figure 7 Arena Model (cont.)

Figure 8 shows the end of the first parallel branches as well as starting two more. The second module is the testing of the electrical connections that will go between the first and second stages. This involves connecting a tester or dummy load that will allow the internal checks to be completed.

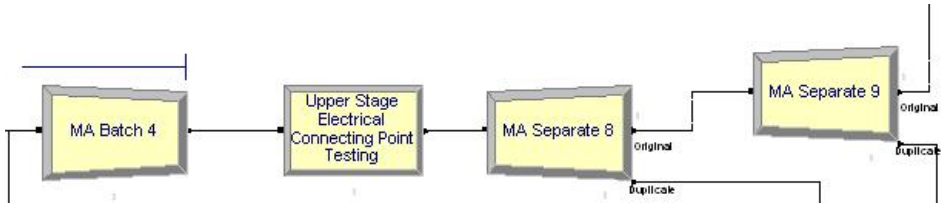


Figure 8 Arena Model (cont.)

Figure 9 shows some tasks that can be performed in parallel. The lower leg involves stage 2 connections and hardware. The third module is for a buffer plug removal and installation. This method uses a plug with pins on both sides. During quick turns on fighter aircraft when speed is important, the buffer plug offers a secure connection that allows for separation between two vehicles in motion. The top portion of this figure has a module for replacing the drag chute if the vehicle uses one. The final three modules involve the Thermal Protection System (TPS). The TPS modules can be minimized if the launch vehicle does not require extensive TPS work. This would be the case if the vehicle had a low atmospheric re-entry speed.

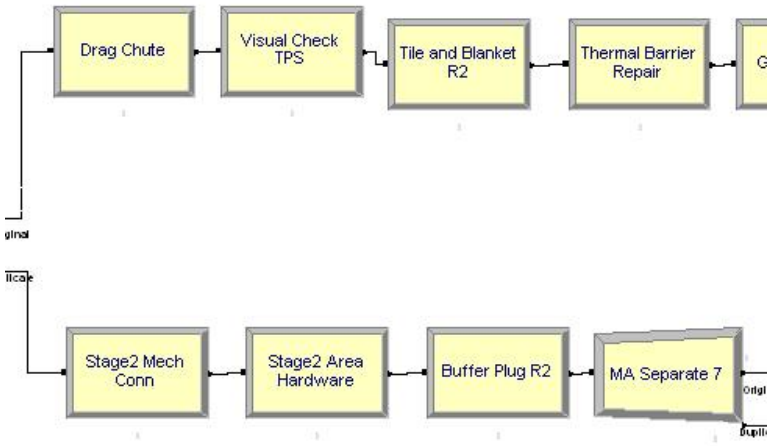


Figure 9 Arena Model (cont.)

Figure 10 connects to Figure 9. The top line is a continuation of the TPS process. The surface would be checked after all systems checked out good. This will preclude multiple surface inspections as well as ensuring that the surface, once repaired, will not be damaged from other maintenance being performed. The total time that is spent on TPS work can be high. The B-2, while using a surface treatment for stealth versus thermal protection purposes, can take 30 hours to repair after panel access (O'Malley 2006). It is important that minimal surface damage occur during the other phases of maintenance.

The final module on the top line accommodates time to perform a full system check. This module, because of the time that the preceding modules require, will be the last process completed before TPS waterproofing. The parallel group allows for the checking of fluid systems. Any line replaceable unit replacements are modeled in the final block of the parallel portion. The last module assigns a value to the number-of -

motors variable. This value is additive every time the process path goes through the module. This sets up a limited loop.

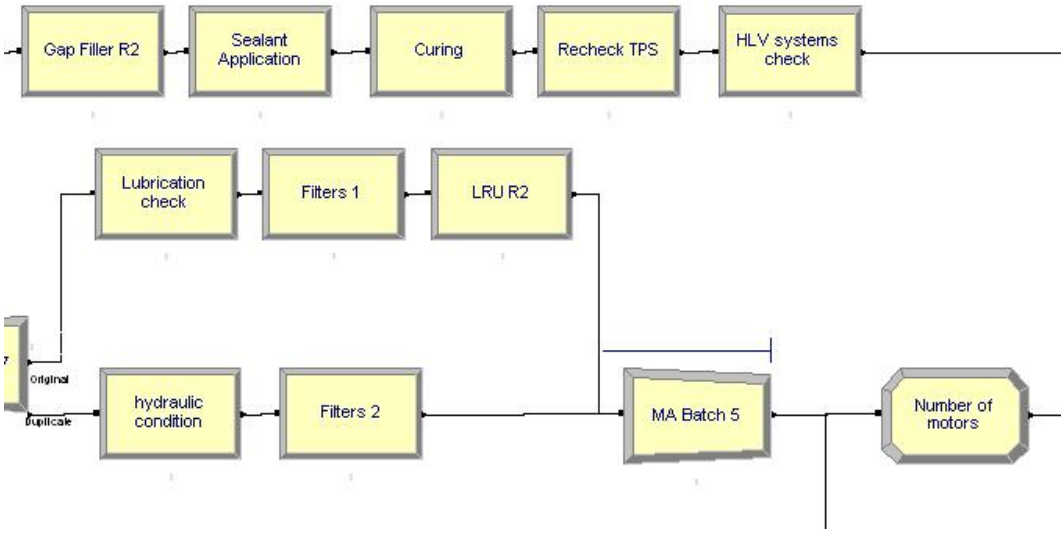


Figure 10 Arena Model (cont.)

One aspect of the launch vehicle that will differ from aircraft maintenance is the fact that the engine will require certain tasks to be performed after every flight. These checks will include tank checks, valve operation, nozzle controls, and linkage inspections. The design of the engine will greatly affect how maintenance is performed in this area. Since the engine is not required for testing of the other avionics systems, it can be inspected while the other avionics are being tested. This parallel processing will enable the maintenance footprint to be quite a bit smaller. A modular engine assembly that could be quickly removed and replaced would be an alternative to repairing the engine while attached to the launch vehicle. These alternatives are captured in the first

module of Figure 11 which is a decide node. If the motor is modular, the upper path will be utilized. Otherwise, the lower path will be followed.

The first module of the top path is where the motor stand is connected to the launch vehicle. The stand is rolled into place where it is pinned to the launch vehicle to hold it in place. A cradle is connected to the motor that will enable it to be rolled back onto the stand. The second and third modules are where the motor is disconnected from the launch vehicle in preparation of being removed to the stand. Then, it is removed. The last module is where the stand is disconnected and moved out of the way. At this time, it could be taken to the motor shop which would be where motors are refurbished and prepared for further use.

The lower path is where maintenance tasks are performed without removing the motor. The first module is the diagnostics module. This enables the connection of test equipment to determine the extent of repairs to the individual motors. The next three modules separate out the tasks performed during motor checkout and refurbishing. The linkage check and repair module is shown in Figure 12.

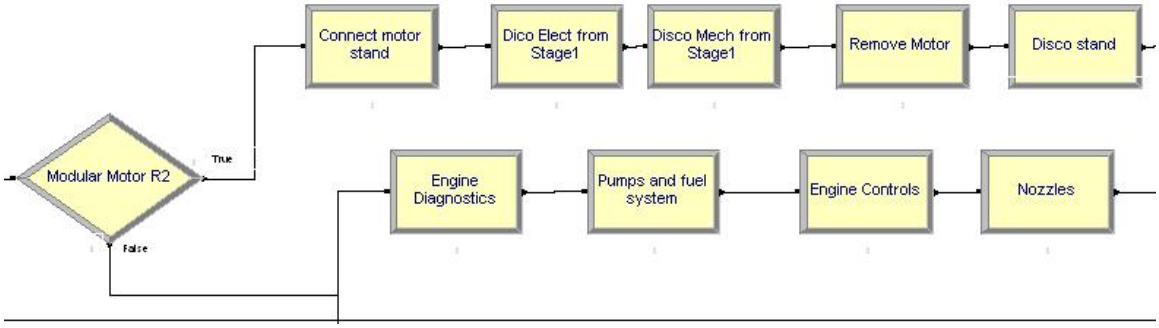


Figure 11 Arena Model (cont.)

Figure 12 continues the motor replacement process. The placement of a stand with a good motor on it is what the first module represents. Just like in the removal portion, this module is where the stand is connected to the launch vehicle to stabilize the stand for rolling the motor into the launch vehicle. The motor is then connected mechanically and then electrically. A connection test is performed prior to removing the stand. If there is a problem with the connections, leaving the stand in place facilitates removal of the motor. Finally, the stand is disconnected and removed from the area.

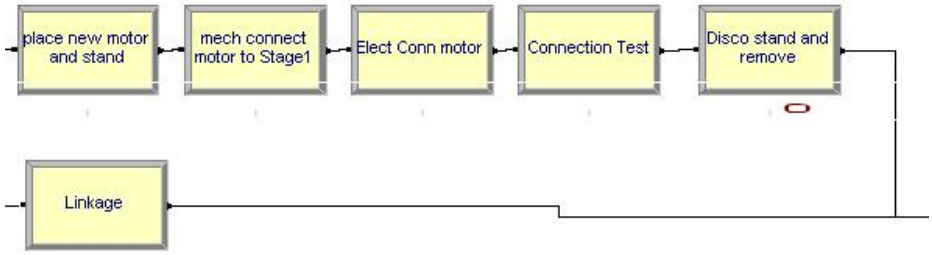


Figure 12 Arena Model (cont.)

Figure 14 shows just three process modules. The top module is the application of a TPS waterproof coating. This coating could probably be applied via a stationary spray device that the vehicle is towed through like a large car wash booth. However, presently, the waterproofing is applied by hand sprayers while the vehicle is located in the maintenance facility. This module allows either method to be utilized.

There is, in the second row, a “motor decide” module. This module looks at the entity (which represents the current task from a vehicle’s collective regeneration

activities prior to a particular flight) going through the model. If the entity has cycled through the motor repair loop for as many times as the vehicle has motors, then it passes through. Otherwise, the entity is forced back through the motor loop to allow for time to check and repair the next motor. This module allows the model to represent launch vehicle designs with different numbers of motors.

The last module on the second row is the engine checkout module. This last module, another decide node, asks if the motor check was good. If the motor check was not good, the entity is forced back through the engine diagnostics and repair loop. If the check is good, the entity proceeds out of the maintenance section of the model and into the prelaunch model coded by Stiegelmeier. Figure 13's last line contains the "Landing Gear and Tires" module which allows for time to check and repair as necessary the tires, brakes and landing gear assembly.

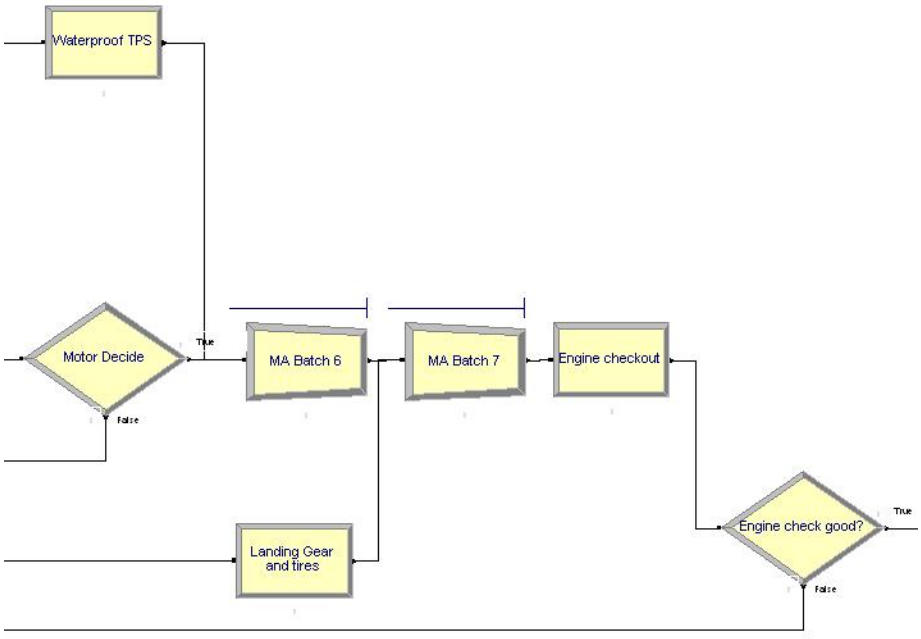


Figure 13 Arena Model (cont.)

Scheduled maintenance includes any Time Compliant Technical Order (TCTO) work and any pre-planned maintenance. The TCTOs are the Air Force equivalent of a factory recall and includes isochronal inspections. TCTOs correct conditions that could result in damage to equipment, injury to personnel, or destruction of the system. These inspections can be planned ahead of time to take advantage of scheduled down time. It should be noted that scheduled maintenance can be performed during lulls in the sortie generation. That is, when the launch vehicle is not needed to be launched for an amount of time that is greater than the time the maintenance will require, scheduled maintenance should be performed. Preplanned maintenance tasks may include any maintenance that was not the result of operations.

The last module was included in Figure 13 as well. See the section on Figure 13 for an explanation of this module.

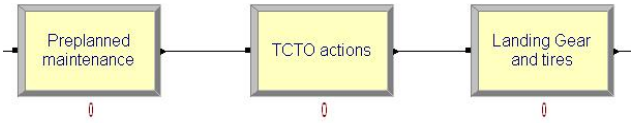


Figure 14 Arena Model (cont.)

Building the Graphical User Interface (GUI)

The sponsor of this research requested that the model be built in such a way that a person with limited experience in Arena or simulation could use it. To simplify the

process of running the model, a GUI was developed that is simple to use and self explanatory. Experience with regular computer software is all that is necessary to run the model. Stiegelmeier and I teamed up to develop MILePOST (Stiegelmeier 2006). This software starts automatically upon selecting the appropriate Arena model. Further, this software allows the user to save a current modeling run as a uniquely named file to simplify the ability to run multiple models and access past runs.

The GUI is made up of several successive screens that are dynamically linked to the Arena model. Changes on the page are updated to the Arena model as the screens are navigated. The opening screen is shown in Figure 15.

Figure 15 MILePOST welcome screen

The Hierarchy screen allows the user to navigate directly to the portion of the model that requires updating. MILePOST always displays the most current run model values. This feature allows subsequent runs to be made quickly while requiring only the

changes made that differ from the last model's inputs, as shown in Figure 16. The program opens an Excel file that reads each run's regeneration time--putting the data in an easy to use format that can be saved and manipulated, and eliminating the need to understand Arena's output analysis software. Each successive page in MILePOST asks a series of questions that can be answered either with experimental data or populated with the default distributions that the literature and discussions with the Delphi experts suggest are reasonable representations of processes.

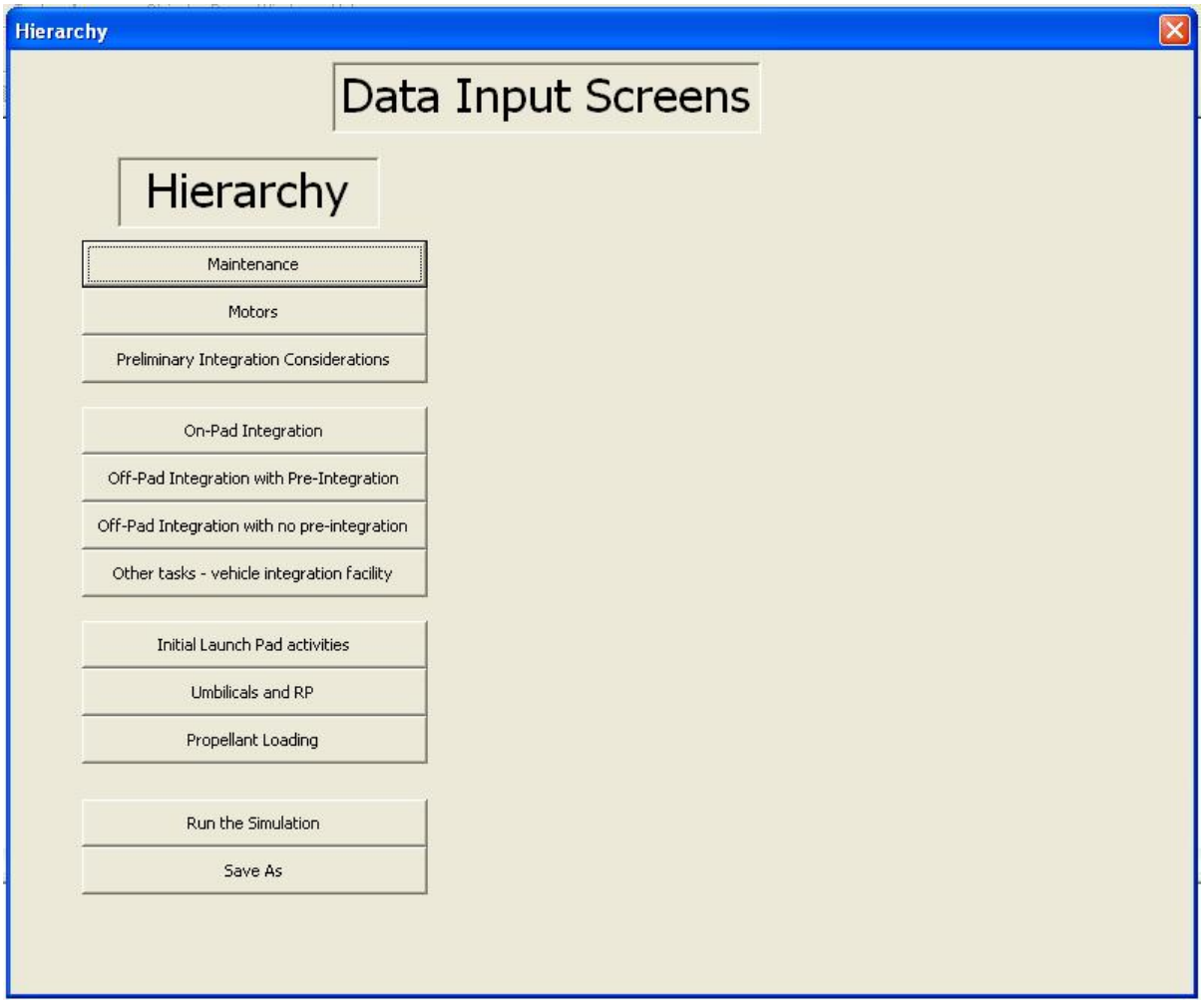


Figure 16 MILePOST Hierarchy screen

The Milepost GUI consists of more than 196 pages of Virtual Basic for Application (VBA) code, as shown in Appendix E. Of those, 67 pages cover the maintenance portion. The remaining code was coded by Stiegelmeier and covers the prelaunch portion of the model (Stiegelmeier 2006).

IV. Analysis and Results

Chapter Overview

The model was first tested to ensure that all paths are possible. That is, the test ensured that there are no logic errors that would prevent an outcome from being possible, given the right inputs. After assuring that all paths are possible, the model was exercised to provide insights and detect sensitivities. This was done by selecting expected values for all modules except for the three with the longest possible processing time. Those three were manually changed to range from the minimum expected value to their highest expected value to ensure the model reacts predictably.

Checking the model paths

Model assumptions fall into two general classes: structural assumptions and data assumptions. Structural assumptions involve questions of how the system operates and usually involve simplifications and abstractions of reality (Banks, II et al. 2005).

To facilitate the checking of the model structure, a simple methodology was followed. First, all values were set to zero for every process module. Next, the model was forced to represent each of eight possible outcomes through manipulation of the decision nodes. See Table 1 for the tests.

To begin the testing process, all values were set to zero. The expected outcome for 10 runs would be a regeneration time of zero minutes for each run since the distribution used each time was a constant zero. Zero was the outcome for each of 10

simulation runs. This ensures that no process module has an affect to these tests with their current value. That is, to affect the test, their value must be moved from zero.

Table 1 Model Tests

Model Tests				
Test #	Batteries Good?	Modular Motor?	Good Motor Test?	Expected Outcome
1	Yes	Yes	Yes	10
2	Yes	Yes	No	20
3	Yes	No	Yes	100
4	Yes	No	No	200
5	No	Yes	Yes	11
6	No	Yes	No	21
7	No	No	Yes	101
8	No	No	No	201

The first four tests require that the batteries are good. To ensure that they are, I set the probability that the batteries would test good, to 100%. The only module that comes into play from this decide module is the “Replace Batteries” module. By setting this module to ‘one’, any outcome that includes it will end in a ‘one’ as shown in Table 1 So, the first four tests in Table 1 will NOT have a ‘one’. That outcome is only possible if the “Batteries Good” decide module is false.

Next, I set “Connect Motor Stand” process module equal to 10. This way, any test that has a value in the tens place would indicate that the Modular path was used. This should be the case in tests 1,2,5, and 6 of Table 1.

Setting the "Nozzles" process module to 100 allows the "Motor Decide" module, which checks for a good motor, to be tested. If a motor is bad, the system loops back. If the Non-Modular path had been picked, this will add 100 to the outcome for every pass where a motor tested bad. Setting the probability to zero would ensure that the entity would have no chance of getting out of the loop. This would make testing in this manner impossible. By setting the probability low and running 10 tests, some of the outcomes would have multiples of either 10 (if Modular) or 100 (if Non-Modular) but some could beat the probability and come through without multiples. Table 1 also shows the expected outcomes of the tests in terms of multiples of 1, 10, or 100.

Table 2 shows the actual test outcomes. The tests were successful. Note that the battery passed its check in the 7th run of tests 5, 6, 7, and 8. Most battery checks resulted in battery failure, which is consistent with the 90% failure rate coded into the model. After running the model 100 times, Arena produced only 8 outcomes where the batteries passed their check. Each run of the model resets the stream of numbers that are randomly generated. Since the model has no change in the order in which the numbers are utilized, it stands to reason that the same number would fall on the same simulation (Kelton, Sadowski et al. 2004).

Sensitivity

The tests so far proved the model can follow differing paths depending upon user inputs. Sensitivity analysis was addressed next.

> For most large-scale simulation models, there are many input variables and thus many possible sensitivity tests. The model builder must attempt to choose the most

critical input variables for testing if it is too expensive or time consuming to vary all input variables. (Banks, II et al. 2005)

Table 2 Verification Tests

Verification Tests								
	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8
Run 1	10	20	100	200	11	21	101	201
Run 2	10	20	100	200	11	21	101	201
Run 3	10	20	100	200	11	21	101	201
Run 4	10	20	100	200	11	21	101	201
Run 5	10	20	100	200	11	21	101	201
Run 6	10	20	100	200	11	21	101	201
Run 7	10	20	100	200	10	20	100	200
Run 8	10	20	100	200	11	21	101	201
Run 9	10	20	100	200	11	21	101	201
Run 10	10	20	100	200	11	21	101	201
Expected	10	20	100	200	11	21	101	201

In the case of MILePOST, the model's output is the total time to perform maintenance and prelaunch activities between launch vehicle flights. Since time is the most important factor, instead of utilization rates or resource utilization, the variables with the highest possible time constraints are the ones that are of interest.

> …Sensitivity analysis can be used even very early in a project to assess the impact of changes in data on the model results. If you can't easily obtain good data about some aspect of your system, run the model with a range of values to see if the system's performance changes significantly. If it doesn't, you may not need to invest in collecting data and still can have good confidence in your conclusions. If it does, then you'll either need to find a way to obtain reliable data or your results and recommendations will be coarser. (Kelton, Sadowski et al. 2004)

Three branches

There are three main parallel branches in the maintenance section of MILePOST. These branches represent the majority of time that is in the maintenance portion. One

branch represents the TPS processes, one represents Motor maintenance, and the final branch represents other maintenance that can be accomplished in parallel with the first two.

The flow time will be plotted with the average time for a specific job with 95% confidence intervals utilizing 30 replications. A replication is one entity, or launch vehicle, entering the maintenance cycle. No other entity enters the maintenance cycle until the previous entity enters preflight operations.

Results of Simulation Scenarios

Three configurations were checked. These were the two motor, four motor, and six motor configurations. The values that were used to exercise each branch are shown in Table 3. The first data point on each chart depicts the default values for each module using the decision selections for that test. Data point 2 always represents the branch with a total processing time of zero. After that, the data points rise incrementally to show the effect that value change has on that particular branch of the model. No change should be detected on a branch that is not on the critical path.

Table 3 Values used for branch tests.

Branch		2 Motors	4 Motors
TPS	1	Defaults	Defaults
	2	0	0
	3	300	600
	4	600	1200
	5	900	1800
	6	1200	2400
Motor	1	Defaults	Defaults
	2	0	0
	3	300	300
	4	600	600
	5	900	900
	6	1200	1200
Other Maint	1	Defaults	Defaults
	2	0	0
	3	600	1200
	4	1200	2400
	5	1800	3600
	6	2400	4800

The first set of tests were completed for launch vehicles with two installed motors. To run these tests, the value of each branch was altered in turn to determine the point that each branch becomes the longest processing time of the three. Figure 18 shows the results of the test.

When the value of the branch was dropped to zero in the second run, the overall length of time for regeneration dropped. This shows that the TPS branch is, when two motors are selected, on the critical path. Further, Figure 19 shows the other two branches when the two motor test is performed. Data point two remains statistically the same as data point one. This shows that lowering the branches does not affect the model's overall processing time. The TPS branch time affects the overall time of the model as soon as the value of the branch approaches 300 minutes.

The Motor branch does not affect the model here until the value of the branch approaches 600 minutes. Further, the Other Maintenance branch does not affect the model until the branch value approaches 1800 minutes.

These first tests show that when there are only two motors selected, effort is best placed at reducing the TPS branch as lowering its overall time also lowers the overall regeneration time.

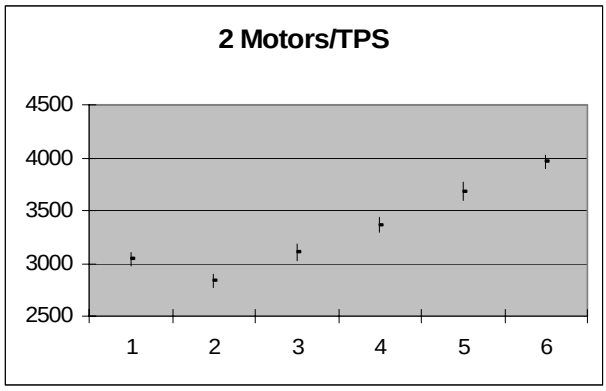


Figure 17 Test of 2 Motor TPS branch

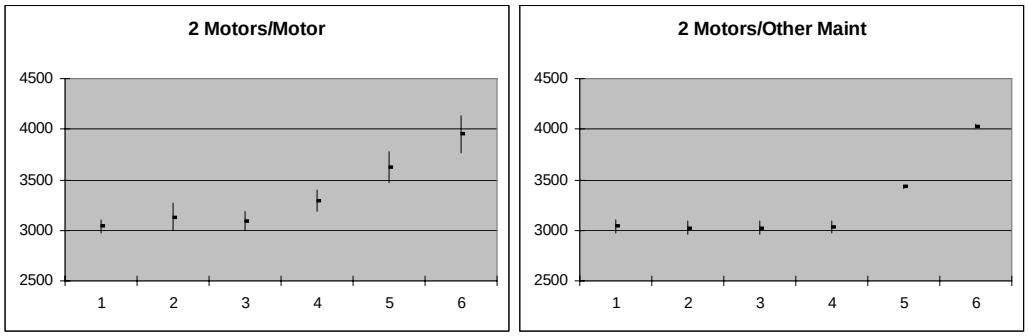


Figure 18 Test of 2 motor selection Motor and Other Maintenance branches

The set of tests for 6 motors echoed the test for 4 motors. The motor leg is on the critical path. TPS becomes the critical path when the branch value approaches 1200

minutes. Figure 21 shows the Motor and Other Maintenance branch when 4 motors are selected. Notice that there is no significant difference in the Other Maintenance branch when it is set to an overall value of zero. The Motor branch is the only branch that shows a decrease in overall time when it is set to zero. This is on the critical path. The Motor branch rises with any value above zero. The other two branches do not rise until their processing time is increased to more than 1200 for TPS or 2400 for Other Maintenance.

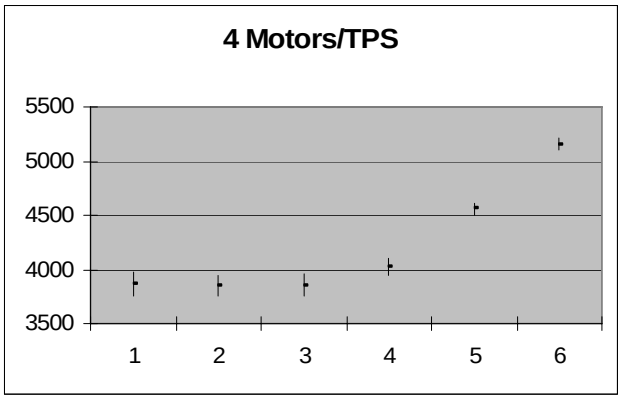


Figure 19 Test of 4 motors TPS branch

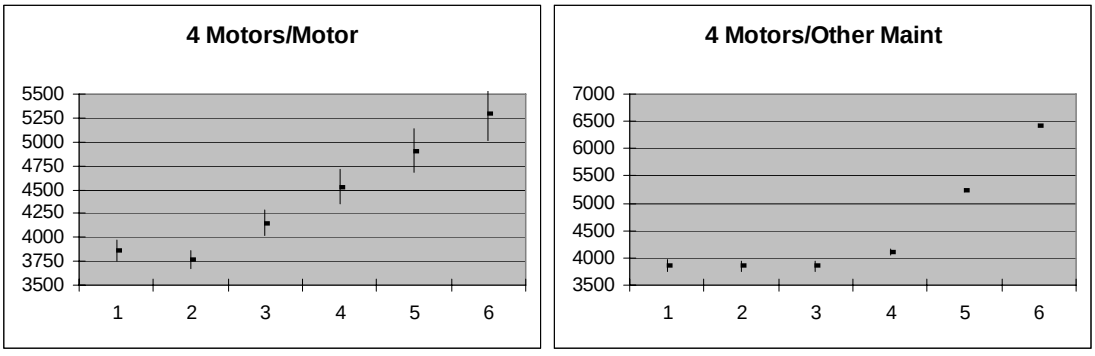


Figure 20 Test of 4 motors selected Motor branch and Other Maintenance branch

There were some issues with using the above methods to verify and validate the model. As with any model, these issues are sometimes introduced by the methods used by the tester or test methods. The issues that were introduced by the methodology used to build MILePOST can be put into two classes: those associated with the Delphi process, those issues relating to the model.

The Delphi panel introduces several issues. The panel members were aware of each other. While their responses were masked, the rounds were emailed to the members who then knew who the other respondents were. The data gathered from the Delphi panel relies on self-reporting which introduces potential bias to their input. The generalizability of results may be limited. This is because the Delphi panel that was used for this research lacked experts from civilian industries such as the airlines and commercial space booster programs. Furthermore, the Delphi Panel exhibited a diminishing response rate with each round, which was expected as the members closed in on consensus.

The model also introduces several issues. One of those issues was the limited availability of actual maintenance processing times. Most of the data that was used was notional. The model is also resource unconstrained. Since MILePOST only modeled one launch vehicle per run, the unlimited number of technicians used to perform maintenance is the only resource that was seized excessively.

Investigative Questions Answered

At this point, the original investigative questions have all been answered.

1. *What generic functions, or sequence of actions, describe Reusable Military Launch Vehicle (RMLV) maintenance?*

The functions are listed in the modules of the model. Figure 4 through Figure 14 represent all of these functions as well as the sequence that describes RMLV maintenance.

2. *How do these RMLV maintenance operation functions compare to aircraft, Expendable Launch Vehicle, and Intercontinental Ballistic Missile (ICBM) maintenance operation functions?*

The maintenance functions are based mostly on large-aircraft maintenance procedures, with the exception of the thermal protection system. The TPS flow is is based on Shuttle maintenance and on data from the B-2's stealth coating regeneration.

3. *What are the RMLV design drivers that will influence RMLV maintenance operations, and how will these drivers affect the relationships, number, type, and duration of RMLV maintenance operations activities?*

Based on plausible, but notional input data, the design drivers that influence RMLV maintenance are the number of motors and the length of time that TPS maintenance will take. The verification tests above highlight these drivers.

4. *How can these RMLV design drivers and maintenance operation activities be incorporated into a discrete-event simulation model that captures a baseline RMLV maintenance operations sequence?*

MILePOST, co-developed in this research and by parallel work by Stiegelmeier (2006), models an RMLV maintenance operations sequence. The

model can accommodate design alternatives including engines quantity and degree of modularity, thermal protection system support, process times and distributions, and other maintenance processes.

V. Conclusions and Recommendations

The Air Force must better understand Reusable Military Launch Vehicle maintenance operations. This has become increasingly important as the Air Force needs a responsive, affordable launch system for space access. The Space Shuttle takes months of preparation before a flight and is expensive. Current expendable launch vehicles require weeks to prepare for flight, with launch costs that are deemed to be too expensive by at least an order of magnitude. Before the Air Force can develop its own reusable system, a better understanding of the maintenance process must be understood.

To tackle the problem, MILePOST was developed. Experts in multiple fields participated in a Delphi panel that formed the model. Multiple rounds were used to form a consensus on the maintenance activities and their relationships.

The model was verified through two processes. One process was to watch an animated version of the model in action while each of the 8 possible sample paths was observed. The other method that was used was to get a base run time of the model when two, four, and six-motor design configurations were selected, and then to change the values of each of MILePOST's three parallel maintenance branches to examine the model's output sensitivity. With the two-motor design, the model indicated that the TPS processing time was the longest of the three branches. This was proven when the TPS branch time was lowered and a lower overall time resulted. Further, with the two-motor design, lowering either of the remaining two maintenance branches to zero had no effect on the regeneration time. When the processing time was raised high enough in the other two branches, they became part of the critical path.

Considering the four-motor design, the motor maintenance branch becomes the longest of the three maintenance branches. This was observed when lowering the motor branch time lowered the entire model's time. Lowering either of the other branches had no effect on the model's reprocessing time. Only when the other maintenance and thermal protection system maintenance branches were raised significantly did they become part of the critical path.

Conclusions of Research

MILePOST pointed out that effort can be directed to different paths based on the number of motors used in an RMLV design. With two motors, the thermal protection system was identified as the process where increased efficiency would impact the overall processing time. When more than two motors are used, MILePOST indicated that the motor maintenance process impacted the overall maintenance time the most. Efforts to lower the overall regeneration time would have been best spent on improving motor ease of maintenance or its reliability.

Significance of Research

This research will allow the Air Force to evaluate maintenance strategies of an RMLV prior to developing the system. This will ensure that attention is put into the areas that will provide the largest payback in terms of overall regeneration time. The default input values of this model are plausible but notional, and are based on aircraft and Shuttle maintenance experience. Further, the proportion of total regeneration time that a

vehicle spends in maintenance with MILePOST is similar to the proportion of time that the Space Shuttle spends in maintenance.

Recommendations for Action

AFRL should use MILePOST to aid in the design of the RMLV. Further, attention should be paid to the processes that will provide the largest impact on overall time reduction for the RMLV. Details and data should be incorporated as feasible into this model.

Thermal protection system maintenance after each flight should be limited or eliminated if motor maintenance time can be reduced. Aircraft maintenance practices should be incorporated in RMLV maintenance strategies wherever possible. Health monitoring systems analogous to the F-22's on-board self diagnosis capability could help reduce maintenance time between flights. Finally, the Space Shuttle's depot maintenance philosophy will not work for the operations tempos desired in an RMLV.

Recommendations for Future Research

1. The model should be expanded to include post-flight operations. This will make MILePOST's ground regeneration modeling complete, by capturing all significant regeneration activities from vehicle touch-down to launch for the next flight.
2. The model should also be populated with better quality data as it becomes available. This will allow the model to better represent the ground operations cycle time.

3. Current aircraft sortie generation philosophies should be compared to proposed RMLV operations.
4. The sensitivity of resource levels and capabilities versus sortie production, should be examined. How would the number and skills of maintenance personnel affect sortie production? MILePOST now assumes that all technicians are qualified for all jobs and that these personnel are unlimited in quantity. It would be of interest to the Air Force to know what minimal manning is, to achieve sortie generation rates.

Summary

MILePOST is capable of modeling a RMLV to the level of detail that allows designers to identify areas of improvement in maintenance. Further, the Air Force can identify where money should be spent in order to improve the processing time of an RMLV. MILePOST is capable of being updated and expanded as data becomes available or as plans are finalized.

Appendix A. Delphi Panel Visio Document

Introduction

Diagram Notes

The following pages contain a conceptual flow diagram that represents a more detailed simulation model that is being developed. The actual simulation model is being developed in Arena software.

This diagram includes only maintenance, integration, and launch pad operations. Other portions of RMLV operations (post-flight recovery and in-flight operations) are not included.

The diagram does not include many support processes that go on outside of the critical path. Only major regeneration events are included.

No extensive scheduled maintenance activities (depot maintenance or phase/periodic inspection) are depicted in this diagram.

Assumptions

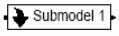
1. Unmanned vehicle
2. Reusable first stage (Space operations vehicle (SOV)) and expendable upper stage.
3. Vertical takeoff, horizontal landing (VTHL)
4. Vehicle will incorporate integrated vehicle health management (IVHM) capability.
5. Vehicle maintenance will be more similar to current aircraft maintenance practices than shuttle maintenance practices.
6. Liquid propellants.

Module Key

Create Module: The starting point of the entire diagram.

Batch Module: Indicates the end of parallel processes. All processes prior to this module must be complete before the next process can occur.

Process Module: An action occurs that takes a certain amount of time.



Submodel Module: Used to provide a clear overview of the entire diagram. Each submodel is a grouping of other modules, for organization purposes.

Decision Module: A decision is made that determines the route to follow.

Off-page reference: Indicates where the path connects on a separate page.

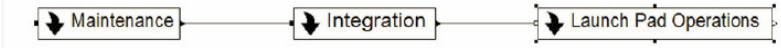
Separate Module: Indicates the start of parallel processes. The diagram path "splits" at this point, and the processes in each path occur simultaneously.

Dispose Module: The ending point of the entire diagram.

01

RMLV Ground Operations Conceptual Flow
Top-level Diagram View

The ground operations cycle of the RMLV is composed of three main segments, or submodels. These submodels are: Maintenance, which includes all mechanical and electrical operations that are required to maintain the systems' flight capabilities; Integration, which involves the assembly of vehicle stages and mating of the payload; and Launch Pad Operations, which involves launch pad activities. This is the top-level view of the entire diagram, and some of these submodels contain submodels within them.



The maintenance flow is shown on pages 3-8.

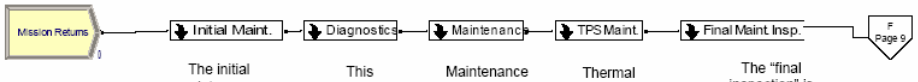
The integration flow is shown on pages 9-11.

Launch pad operations are shown on pages 12-14.

02

Maintenance Flow Overview

Mission returns from Post-Flight operations



The initial maintenance portion includes positioning the RMLV as well as any maintenance stands that are to be in place during every maintenance flow. (Page 4)

This diagnostics module will include using an IVHM system similar to the maintenance system used in the B2 and F22 programs (Page 5)

Maintenance includes unscheduled maintenance where workers are fixing problems found during flight or in subsequent testing and also scheduled, or pre-planned maintenance. These can be worked simultaneously. (Page 6)

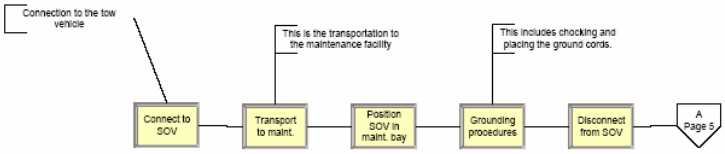
Thermal Protection System maintenance (Page 7)

The "final inspection" is merely the testing to ensure that the craft should be mated with the payload. This gives us the opportunity to use a back-up SOV if there are maintenance issues that are unresolved. (Page 8)

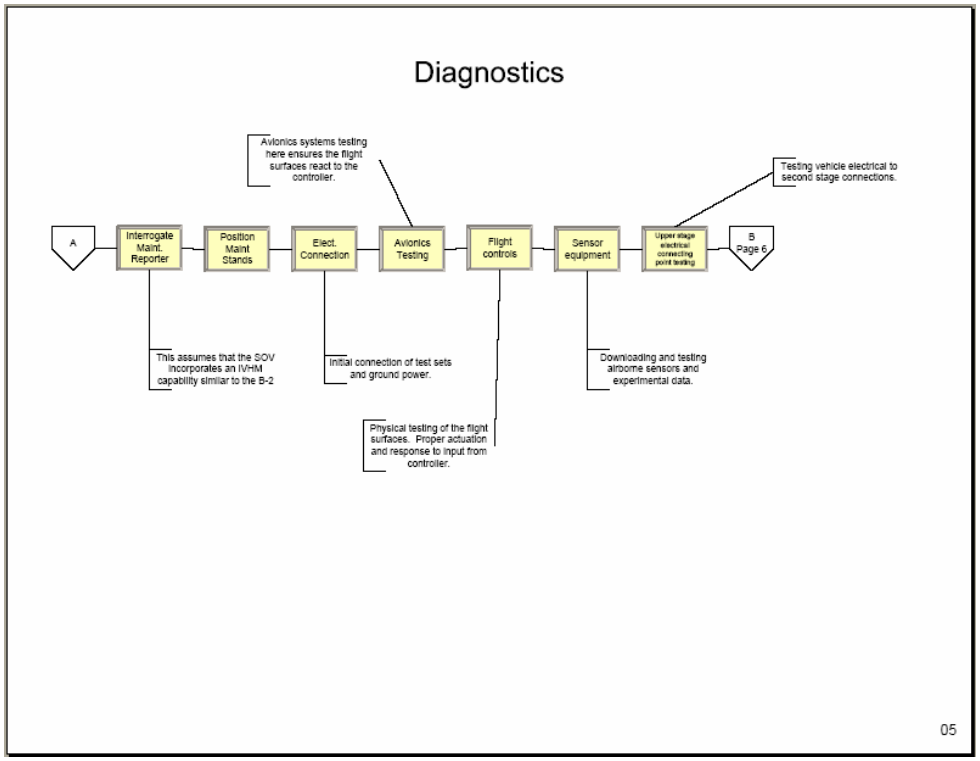
At this point, the RMLV is released for integration or intermediate storage.

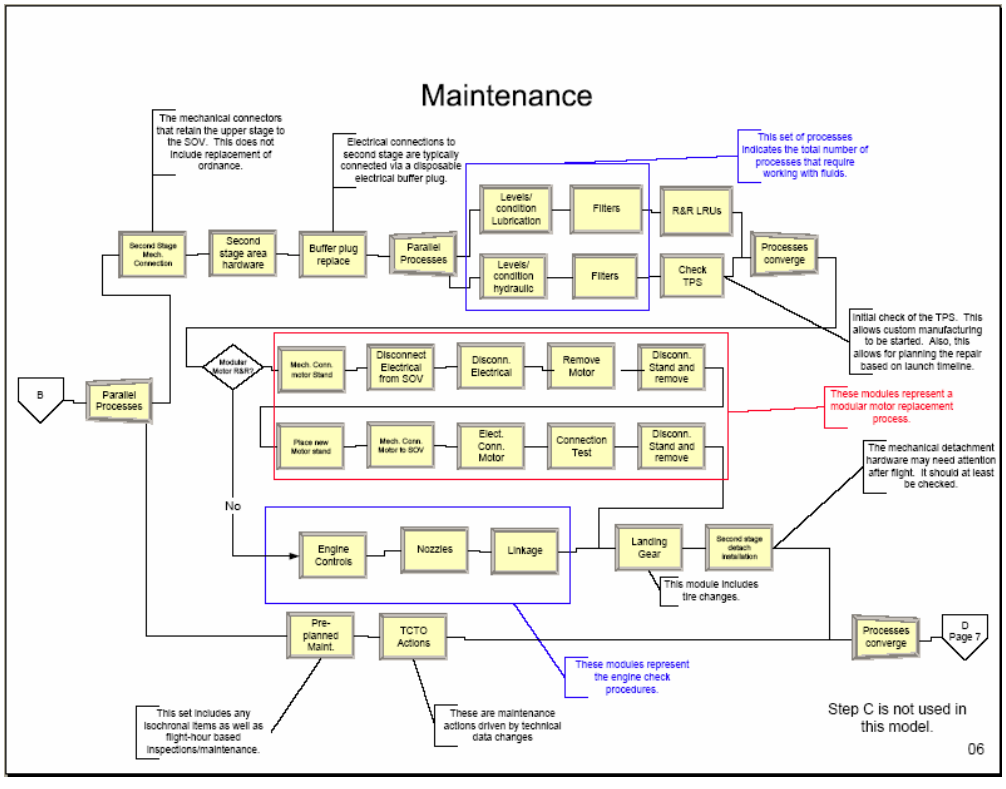
03

Initial Maintenance



04

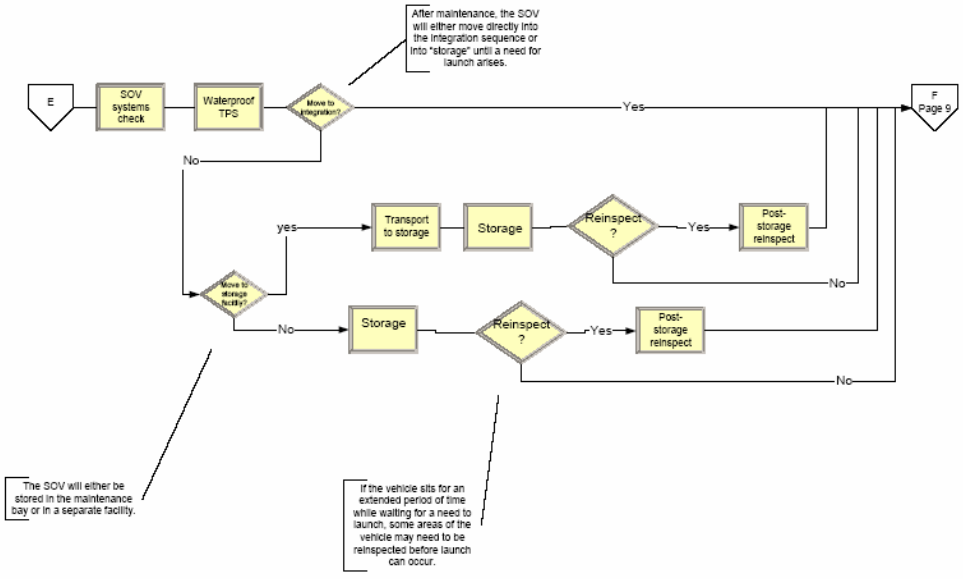

Diagnostics
Avionics systems testing here ensures the flight surfaces react to the controller.
Testing vehicle electrical to second stage connections.
A
Interrogate Maint. Reporter
Position Maint Stands
Elect. Connection
Avionics Testing
Flight controls
Sensor equipment
Upper stage electrical connecting point testing
B Page 6
This assumes that the SOV incorporates an IVHM capability similar to the B-2
Initial connection of test sets and ground power.
Downloading and testing airborne sensors and experimental data.
Physical testing of the flight surfaces. Proper actuation and response to input from controller.
05


Maintenance
The mechanical connectors that retain the upper stage to the SOV. This does not include replacement of ordnance.
Electrical connections to second stage are typically connected via a disposable electrical buffer plug.
This set of processes indicates the total number of processes that require working with fluids.
Second Stage Mech. Connection
Second stage area hardware
Buffer plug replace
Parallel Processes
Levels/ condition Lubrication
Filters
R&R LRUs
Levels/ condition hydraulic
Filters
Check TPS
Processes converge
Initial check of the TPS. This allows custom manufacturing to be started. Also, this allows for planning the repair based on launch timeline.
B
Parallel Processes
Modular Motor R&R?
Mech. Conn. motor Stand
Disconnect Electrical from SOV
Disconn. Electrical
Remove Motor
Disconn. Stand and remove
Place new Motor stand
Mech. Conn. Motor to SOV
Elect. Conn. Motor
Connection Test
Disconn. Stand and remove
These modules represent a modular motor replacement process.
The mechanical detachment hardware may need attention after flight. It should at least be checked.
No
Engine Controls
Nozzles
Linkage
Landing Gear
Second stage detach installation
This module includes tire changes.
Pre-planned Maint.
TCTO Actions
Processes converge
D Page 7
These modules represent the engine check procedures.
This set includes any isochronal items as well as flight-hour based inspections/maintenance.
These are maintenance actions driven by technical data changes
Step C is not used in this model.
06

Thermal Protection System

07

Final Maintenance Inspection



08

Appendix B. Delphi Panel Round One Comments

Delphi Panel First Round Comments by page of Visio Document that the panel were given. Our responses are italicized.

PAGE 1

PAGE 2

#9 You may want to consider adding an in-parallel flow for post-flight engineering assessments, mission call-up, preflight flight engineering, major flight readiness reviews, and ultimately authority/authorization to launch.

PAGE 3

#9 The TPS Maintenance can probably be shown as being in parallel with Maintenance. At least some of the TPS maintenance activities will likely be doable in parallel with other maintenance activities.

TPS maintenance is now shown in parallel. The TPS was initially shown to be a serial process after maintenance in the hopes that an accelerated maintenance cycle would allow the surface to be worked on after maintenance is complete to prevent re-damaging the treatment. However, scheduling the TPS and maintenance work could result in most of the TPS work being done in parallel without slowing maintenance or creating future obstacles.

#5 Use F-35 IVHM instead of B2 and F22.

The F-35 system is the newest system available. Actual data is unavailable. However, the capabilities that go with the system outweighs the data that was available on IVHM with the B-2. The F-35 information and capabilities will be used from here on.

PAGE 4

#2 For the Initial Maintenance Module, is the RMLV on a trailer, or towed on its own landing gear? Either way, you probably need a step to check that the trailer or RMLV is safe to tow (i.e. brake accumulator pressure within limits, etc.)

The RMLV will be towed on its own gear. The flow is modeled after a B-2 maintenance cycle. The safing process is part of the connection phase. Both the towing vehicle and the RMLV will be safed during the Post-flight portion of the overall model. It is not part of the scope of this model and will be handled by a separate treatment.

#9 Add a Process block prior to "Connect to SOV" for post landing safing and inspections on the runway or ramp. Recommend that the vehicle be designed such that it can taxi off the active runway prior to shutting down.

Post-flight activities are not handled in this model.

PAGE 5

#2 Between the Diagnostics and Maintenance modules, what happens to the data downloaded from the maintenance reporter? If there are fault codes, they need to be addressed.

The Diagnostics module drives the Maintenance requirements. This allows maintenance activities to be planned to minimize the maintenance flow time.

#9 Depending upon the vehicle design, some of the diagnostic activities can be accomplished in parallel rather than in the serial fashion shown. In particular, the sensor/experiment activity should be designed such that it can be worked on in parallel with avionics and flight controls.

You are correct. It has been changed to reflect your suggestion.

#14 "Interrogate Maintenance reporter" what exactly does this mean? Are you downloading information from the IVHM system prior to checking the rest of the systems. After a flight, there should be some kind of data download - if that's what this event is - then great, if not, then an event including data download/diagnostics would be important to include here.

This module is mostly a space holder that allows maintenance to take time organizing the information gathered from the IVHM into a logical flow of maintenance tasks. Optimally, the RMLV will broadcast its health, operating parameters, telemetry, and such during flight-completely eliminating the need for time to be spent at the "Interrogate Maintenance reporter" block.

#14 Further explain Avionics Testing and Flight Controls. I agree they should be separate events, but your descriptions sound very similar.

The Avionics Testing is a check to ensure that the software has reacted in an expected manner. There are no pilots to let us know that it was sluggish to the right. The Flight Controls block is a physical check to ensure that limit switches are functioning and that the controls are doing what the software thinks they are.

PAGE 6

#9 I would like to see more detailed information for the process blocks titled "Pre-planned maintenance" and "TCTO Actions." Perhaps a notes page describing what is thought to be in these blocks.

These blocks are residue from my flight line experience. The IVHM should render both blocks obsolete as it will report ALL maintenance due. We are trying to move away from time based maintenance and into capability based maintenance.

#9 Along with Pre-Planned Maintenance and TCTO Actions, there needs to be a process block for unplanned maintenance activities. This will likely be a combination of in-parallel and serial work.

Unplanned maintenance will show up in the distributions of other maintenance tasks and can't be planned into the model as they can happen at any point.

#9 Some landing gear activities can probably be performed in parallel with engine work and other pre-planned/TCTO maintenance activities.

That is correct and has been changed on the model.

#14 -What is the difference between SECOND STAGE MECH. CONNECTION and SECOND STAGE DETACH INSTALLATION?

"Second Stage Mech Connection" is the physical connection that creates a connection point to the RMLV. The "Second Stage Detach Installation" has been changed to "Second Stage Detachment mechanism" and refers to the ordnance or electro-mechanical locking mechanism. This should clear it up a little.

#13 Modular engine replacement - Would the output from this need to go into the engine check procedures after installation or are the controls and linkages all part of the modular engine design?

The engine would be completely modular if this path is followed. This capability is already being developed for the NK-33.

#5 TPS should be addressed earlier as IVHM may incorporate early assessment capabilities.

It is and it will.

#5 Combine TCTO Actions and Pre-planned Maintenance if prognostic capabilities exist.

The limits of the prognostic capabilities have not been fully explored. Rather than create a model that will only work with really good IVHM, blocks have been added that can have distributions tuned based on actual capabilities. If the IVHM can report all maintenance tasks, the time for these two blocks will be zeroed out.

PAGE 7

#9 The TPS work shown on this page should be moved to page 6 and shown in parallel with the other maintenance activities.

Good catch. It is now in parallel but remains on it's own page as it gets more difficult to follow the "eye chart" that is maintenance.

#14 -you mentioned custom manufacturing on p.6 - but then do not have it on this page. Is it included in TILE AND BLANKET R&R? If so, I think this block needs to be expanded to cover more detail.

Yes it does. The block has been explained on the model.

#5 Move TPS into a parallel process as much as possible.

TPS has been moved in parallel with most of maintenance.

PAGE 8

#9 An alternative to the serial work to perform "Waterproof TPS" being performed all at the end would be to perform the waterproofing work incrementally as you go. This may

result in a quicker ground turnaround. However, if you can come up with a processing station/system that allows the entire vehicle to be quickly waterproofed (e.g. the way deicing is done on aircraft) then that would probably be best.

TPS waterproofing will have more in common with de-icing than any of the more common methods in use. A drive-through spray section could suffice.

#9 This page shows a storage option for the first stage SOV. On subsequent pages, it may be beneficial to show a similar storage/hold option for the integrated SOV/2nd stage as well as the SOV/2nd Stage/Payload. These options would allow you to minimize the time between mission call-up and launch. The questions then become how long can you hold in these various configurations and what needs to be done to continue subsequent processing towards launch.

We have now added similar storage options for the integrated SOV/2nd stage as well as the SOV/2nd Stage/Payload.

#14 -WATERPROOF TPS - this is an assumption that the TPS requires waterproofing

Currently, TPS requires waterproofing. If waterproofing is not required in the future, this section could be removed or bypassed or set to a time of zero.

#14 -REINSPECT - is there a time that this should not occur? If no reinspection should take place what is the time limit?

A reinspection, at least to some extent, will almost always occur after storage. In the Air Force, an aircraft preflight inspection is good for 24, 48, or 72 hours, depending upon MAJCOM instructions or airframe specific requirements. We are assuming RMLV maintenance will follow similar guidelines.

#14 -MOVE TO STORAGE FACILITY - this question is confusing because your 'yes' and 'no' both lead to storage facility options. Perhaps a rephrasing of the question would help.

We have now simplified this section. Hopefully it is no longer confusing.

#13 Is the Re-inspect decision block and Post Storage re-inspect significantly different between the two storage areas?

No, they would have been similar. We have simplified this section so that there is only one re-inspection module entitled "Re-accomplish Preflight."

PAGE 9
PAGE 10

#4 Can the upper stage and payload be mated together while the SOV is going through it's turnaround maintenance then mate the upper stage and payload to the SOV?

We have now added a preintegration section, occurring in parallel with SOV maintenance operations.

#14 PREP CLEAN ROOM - this is good. Don't you have to transport the vehicle/payload TO the clean room? This could be quite a process. (you also have this option listed on p. 11 and p.12)

We are assuming that the clean room is either at the integration facility or at the launch pad.

#13 Seems as though some combination of the blocks could happen here as well. Seems to me that the only difference from the pre-integration of the payload path and the no pre-integration path is the fact that the payload is already in the second stage on the upper flow. Would there be a significant change in these distributions for this reason?

There may not be a significant change in the distributions, so some of the modules could be combined. We left them separate to keep from confusing people. We will combine if we get more similar feedback.

#13 Also, wouldn't there be a 1st and 2nd stage integration check for the pre-integration flow? I think there may be some differences hare since there may be some integration checks to the installed payload if the payload is already on the 2nd stage.

This is taken care of with the "Entire vehicle integration check" that takes place after the payload and 2nd stage are attached to the SOV. 2nd stage to payload integration check is covered in the preintegration section.

#13 Pre-integration during decision on waiting for launch - Why was the pre-integration process not included on the flow after maintenance inspection? Are you assuming that if the payload is not pre-integrated and the need for launch arises that the payload will "Only" be processed on the No Pre-integration path? This is OK and you mention that the pre-integration takes place to save time (Waiting for launch requirement and assuming the mission ie payload requirement is known). This although precludes the time it takes (Value-Added time?) to pre-integrate the payload into the 2nd stage. My thought was that this may need to be included if a launch requirement came before the payload was pre-integrated. Since the launch requirement decision is not modeled you could assume that you know that you have sufficient time to pre-integrate the payload before the decision to launch comes. I know this is long winded and I'm not sure if this/these process should be modeled. I do think you should address these to see if they should be included.

We have now added a preintegration section in the model, occurring in parallel with SOV maintenance. If preintegration does not take place, then it is assumed that integration takes place in the traditional way: 1st stage to 2nd stage and then payload.

PAGE 11

#4 Very busy – can it be broken out? Bottom process – you are loading hypergolics before ordnance install – I believe that the fuel loading should be the last thing done – after ordnance install. Recommend doing all fueling on pad as last and final step – you don't want to move anything with fuels loaded. Also - the pad/payload is purged with nitrogen (inert gas) prior to fuel load - nitrogen atmosphere is unfriendly to humans :-)

Hopefully this page is easier to follow now. We have been told by other sources that hypergolics can be loaded before the vehicle gets to the pad. Shuttle hypergolics are loaded on the pad but still days before launch. Should hypergolics really be moved to last step? Is hypergolic loading prior to arrival at launch pad a possibility?

We're looking for more info on the nitrogen gas purge prior to fuel load. Is this always done, regardless of the propellant type? Is it done for cryogenics only? What does it involve? Is it done only once before propellant loading, or is it done several times, perhaps at the beginning and then before a switch is made to a different type of propellant?

#13 Very thorough but should there be a "Clean Room" on the "Install Payload Now Vertical integration" path?

Yes, good catch, but now OBE as this path was removed.

#11 Why are pages 10 and 11 so different? All tasks must be completed wether on pad or off...

The main reason these pages look so different is that off-pad integration could be done horizontally or vertically, while on-pad integration will obviously be done vertically. Page 11 accounts for both the horizontal or vertical possibilities. In addition, page 11 includes the option of installing the payload in the integration facility or waiting to install the payload at the pad (Delta IV). Page 11 also includes the option to do several other activities in the integration facility, like install ordnance. For on-pad integration, these other activities are accounted for on page 13.

PAGE 12

#5 The "no" option for "Install payload on pad?" should join the other path before the "entire vehicle integration check" module. Always perform integration check before fueling.

If the payload was installed previously on page 11, then the integration check was already done, in the integration facility before the vehicle got to the pad, so the integration check still gets done before refueling.

PAGE 13

#4 Umbilical connections. These should be KISS – keep it simple stupid. Connections on the Titan IV were numerous and had connections on almost every stage. These connections get caught on the tower when you move it as well – this I know from experience. Use a race way to carry all connections on the vehicle to one or two concentrated connection points (similar to MMIII) with electrical, fuel, comm, etc. connections.

Agreed—umbilical connections should be as simple and as few as possible. Any recommendations for simplifying the umbilical connection section on page 13, from a modeling perspective?

PAGE 14

#4 Another note – are you going to use squibbed batteries? If you blow the battery and don't launch then you have to R&R the batteries = lost time.

We are not sure about battery specifics. The probability of a scrubbed launch (and other similar pass/fail scenarios) will be added to future iterations of the simulation program.

We are interested in any perspectives or suggestions concerning propellant loading sequence or parallel propellant loading operations.

Appendix C. Delphi Panel Round Two Comments

Delphi Panel second round responses by Visio page

PAGE 1
 # 19

Need to change SOV to ORS or HLV (hybrid launch vehicle). The SOV is a very narrow interpretation of a solution which is no longer part of the ARES plan.

All references to SOV have been changed to HLV.

#9

What are your assumptions regarding the electrical power source for the reusable first stage during flight. Will it be powered by on-board batteries or fuel cells? Depending upon the choice, additional blocks should be shown on page 6 for the maintenance of the electrical power system. If batteries, will they be reusable or require R&R between missions?

At this point, it is my assumption that a generator from the main engine will supply power during the powered lift phase and batteries will supply power during space operations and return. These batteries, at present are reusable. The model now reflects battery maintenance/charging and a removal block.

What are your assumptions regarding power for actuating flight control surfaces, landing gear deployment, steering, and brakes on the reusable first stage? Will they be powered by a hydraulic system or electrical servo actuators? If a hydraulic system is used, what will provide the power source to pressurize the hydraulic fluid?

Control surfaces will be powered by either electronic controls or a hydraulic-electric hybrid. Fluid would be pressurized locally. There will be no main hydraulic pump.

Will the reusable first stage have a drag-chute for landing?

The drag chute, among other things, is one of those things that I have a lot of experience with but still omitted them accidentally. The model now shows chute installation. The removal would be during post-flight operations usually at the end of the runway.

What type of system will be used to manage waste heat from avionics on the reusable first stage?

The waste heat during the powered portion of the launch will be handled by heat exchangers in the main fuel tank.

Acronym List: TPS and R&R should be added to the acronym list.

 # 2

I read the comments from the first round and am happy with the responses to my questions. I don't have enough system knowledge to provide any more input on the highlighted comments. The revised model appears to be logical to me.

PAGE 2

 # 7

Instead of "Initial maint." call it "Maintenance Preparation" since there is no actual maintenance happening in this submodel.

Done.

PAGE 3

 # 13

Will there be a clean room prep requirement for Pre-integration?

We have now added an option for clean room prep on the preintegration page.

PAGE 4

#13

It seems like the "Post Flight" checks and safeing would be an important process to capture. If the "Post Flight" process was not part of the initial scope of your thesis could it be added or at least commented on in your report for future efforts? Are there other known areas that would need to be worked in future efforts to capture them as well?

There is a GRP (mini-thesis) that is in the pipeline that will capture the post-flight portion of the overall model. It will be constructed so as to fit with the model we are developing now.

PAGE 5

PAGE 6

#14

It is difficult to account for unplanned maintenance. Is there a way you can account for error your analysis or possibly a range of times so that you have the best and worst case scenario? Or is there a way to insert into your model that an engine will need to be replaced unexpectedly?

Yes. Arena software allows for distributions and anomalies that will account for catastrophic failures as well as unplanned maintenance events. These numbers

will be able to be tuned as maintenance and reliability data is collected on the components of the RMLV.

#7

For the "R&R LRUs" and "Pre-planned maint" modules will the most maintenance intensive subsytems be considered(RCS/OMS, Power, actuation, etc)? Are these details to be worked out later or would these systems be considered only as depot maintenance and hence not in the model?

Most maintenance actions to be performed on the craft itself will be of the box swap variety. Back shops will be utilized to perform most repairs.

#13

Now that this [maintenance and TPS maintenance] has been split up as a parallel maintenance (As they should be) there will probably be interactions between these two sub-models that will impact the processing times. For example, if maintenance of engines, LRU's, landing gear, etc is still being done, is it understood how and where (in the process) these processes would impact the TSP processing? A major portion of the TPS could be accomplished in parallel but I believe that there will be open access panels, gear door seals, etc. that will require the final close out for the TPS to wait until after the maintenance has reached a certain point in its process.

The model has been corrected to show two points at which the TPS system is repaired. Notice that page 6 has two points that point to "D" of page 7. The actual model (as opposed to this Visio representation) will allow us to track whether or not maintenance actions would impact the TPS system. Thank you for pointing out that I had originally had the engine portion of maintenance skip the TPS system. This would have allowed the TPS to be finalized while the engines, TCTO items, and landing gear were possible still being worked on. Further, these are the type on interactions that it will be vital that we maintain in the Arena model.

PAGE 7

 # 7

What is considered to be sealant? RTV? Gap fillers and thermal barriers should also be considered, these can require a lot manhours.

I've included the fillers and barriers in the model. Advances in TPS protection will help lower the manhours required and the frequency of repair. Having said that, I've included the two modules you mention in case advances are slow in coming.

PAGE 8

#13

General comment/clarification. [The below are] not recommended changes unless the following comment is not what was intended in the model.
The way this is modeled I assume that vehicles are maintained in a hanger and maintenance equipment is brought to the vehicle (As opposed to how it is done in the shuttle processing facilities). Storage of the vehicle waiting for a mission will be in the place where the vehicle was maintained so no transport time is required.

If re-inspection identifies additional maintenance, you would need another process module with a preceding decision module to account for this occurrence.

For this effort, we are assuming that any additional maintenance required will be captured in the Reinspection module. We have changed the title of these modules to "Reinspection and additional mx." There is obviously more detail that could be included here, but this will be fleshed out later.

PAGE 9
PAGE 10

#14

I think you should keep them separate{Two modules that we separated to prevent confusion}. It's possible that a buyer that wants to launch a payload could provide a second stage with the payload already integrated. This option could account for that. Depending on your payload, integrating it on the pad will add the task of moving the payload to the pad and all the logistics that go along with that.

#13

The two separate vehicle erection process flows still seems redundant. If the Pre-integration decision module is moved to before the 1st, 2nd stage integration check (If this is even needed since it is done again at the vehicle integration check) process, then the affirmative path is directly to the Entire vehicle integration check. The negative path will be to integrate the payload prior to the Entire vehicle integration check.

Disagreement here between #13 and #14. We realize that either option is feasible, but we will leave the model as is, since having two separate paths with similar modules will not affect model output. The only real disadvantage to having two separate paths is more clutter in the model. The processing time required to leave both options available is minimal as is the extra work in including them. The reality will determine the direction the model is used.

PAGE 11
#4

There is usually some type of cooled air supplied to the payload/vehicle while it sits on the pad. It's usually air until the fueling is done and then they switch to nitrogen. The fuel used was cryo so it may be that it is not needed for the hypergolics. The shuttle hypergolics on the shuttle are only a fraction of the total fuel. Also - we transport the PSREs with hypergolics in them but it's a pain.

Transporting a fully fueled vehicle to the pad is a BAD idea - to much risk. What about the added weight of the fuel - will that make your transporter requirements unattainable.

Supplying conditioned air/nitrogen is something that will happen continuously as other events on the pad are taking place. To supply the air or nitrogen, hoses/umbilicals will need to be attached, and we have accounted for that in the "Attach payload handling equipment" and umbilical connection modules.

Hopefully we'll avoid hypergolics altogether, but just in case, we've left the option to include them and to load them either in the integration facility or on the pad.

The only fuel on the vehicle during transport would be hypergolics, if any; and the amount of hypergolics would likely be quite small.

#13

You have a vertical and a horizontal process combining into the 1st & 2nd stage integration check. Should the down stream process be considered different for payload integration in the vertical or horizontal configuration? These were different paths on the first iteration but now they are combined paths although there was no mention of this in the comments for this page.

Yes, they were different paths on the first iteration, but they were very similar. The only difference between the two downstream options in the first iteration was that the vertical integration path had a module entitled "Lift and align payload," and the horizontal integration path had a corresponding module entitled "Position and align payload." Since these two paths were so similar, we combined them, and assumed that the obvious differences between vertical and horizontal payload integration would be captured in the distributions that are put into Arena.

Need to extend the "No" path for Load Hypergols Decision module around Load Hypergolic fuel process.

Thank-you. This was corrected.

Global Comment on re-inspections. What happens if a problem is found during inspections? Is there information on likelihood of occurrence? Seems that the

farther you are in the process the worse the process time might be to fix. Say the TPS is damaged on erecting the completely integrated vehicle on the pad. How this would be repaired might require longer delays than if the damage occurred in the maintenance facility. This may be beyond the scope of this effort but a processing time hit will result if additional maintenance activities occur.

For this effort, we are assuming that any additional maintenance required will be captured in the Reinspection module. We have changed the title of these modules to "Reinspection and additional mx." There is obviously more detail that could be included here, but this will be fleshed out later.

PAGE 12
PAGE 13

#4

Keep [umbilical connections] as is - I'm probably getting into the weeds on this one :-)

#14

The connections should be simple, but they are not always simple. I would leave it as is, and if when (if?) we ever build something the tool can be validated with correct times and correct operations. You always have the option of zeroing out time, but I think it would be difficult to take into account an even that did not exist in the model.

The connections can be very complicated if, for instance, a pin by pin check is required prior to mating and then a full electrical check required afterwards. With Arena, this process will be able to modeled after real data captured from actual connection times of similar connectors. Once the connector type is finalized and the check requirements are known, it will be fairly simple to change the distribution in Arena to model that type.

 # 7

If the Final TPS Inspection done manually it should be performed before the RP-1 fueling to reduce the number of personnel near an already fueled vehicle. If the inspection is automated this would not be required.

Our understanding was that it is not exceptionally hazardous to work around RP-1, since it is similar to jet fuel. (Aircraft operations include maintenance around fully fueled aircraft all the time, with certain restrictions.)

PAGE 14
#4

If you use hypergolics you can work on the vehicle. If you use cryo - once you fuel then you can't go near it. Plus with all the losses while it sits on the pad you would want to fuel then launch as quick as possible.

I think your only limitation on parallel loading is the infrastructure (pump and pipe size) and the vehicles ability to load multiple tanks at once (structural loading, etc.)

We are expecting a cryogenic fuel to be utilized to lessen the hazards associated with using hypergolic fuels. Until we know piping and pumping limitations and vehicle structural limitations, we will not be able to add more specifics to the parallel propellant loading portion of the model. The model will retain the capability of parallel fueling in case that capability needs to be utilized.

#14

Is there any way you can create an option to have serial or parallel propellant loading?

We have accounted for these possibilities on page 14. The decision nodes at the beginning of page 14 will direct the model appropriately, allowing for either parallel or serial loading operations. Until we know piping and pumping limitations and vehicle structural limitations, we will not be able to add more specifics to the parallel propellant loading portion of the model.

Appendix D. Delphi Panel Round Three Comments

GENERAL COMMENTS

#14 I don't mean to be knit-picky here, but I think HLV makes an assumption that may not be true. The first space operating vehicle (SOV) may or may not a hybrid. It could be, but it also could be fully resuable or completely expendable. I thought SOV was the best way to name the vehicle because it does not assume or imply anything except that the vehicle can operate in space. Operationally Responsive Space Lift/Acess (ORS) does not really denote a vehicle, but really describes a *type* vehicle. I would have kept it: SOV. This is probably minor, though.

#2 From my perspective, no missing events, paths make sense, model appears sound, no recommended changes/deletions/additions/comments.

#1 I reviewed the model and have no additional input.

#9 I have no further comments or questions. I enjoyed participating in the development process.

PAGE 3

#10 Several instances of using a clean room as a decision block in the payload processing flow diagrams. If the payload is so sensitive to contamination at this stage of the processing, then the flow diagram needs to include encapsulating the payload once your connections have been made and verified. Encapsulated payloads require constant monitoring and a constant supply of clean, dry, regulated air or nitrogen purge. A more realistic approach might be to assume that the payload comes pre-serviced and encapsulated. The Launch team is only responsible for power, comm, and mechanical connections with no clean room required. The EELV payloads usually arrive in this manner, hypergolic propellants are already loaded (but they do have a limited shelf life once they are loaded (30 days??).

Appendix E. Graphical User Interface Code

Project/Hierarchy

```
Private Sub CommandButton1_Click()
  Me.Hide

  Maintenance.Show
End Sub

Private Sub CommandButton10_Click()
  'c = ahtCommonFileOpenSave()
  Arena.Application.ActiveModel.SaveAs (Module1.ahtCommonFileOpenSave)

End Sub

Private Sub CommandButton11_Click()
  Me.Hide
  po8propellant.Show

End Sub

Private Sub CommandButton12_Click()
  Me.Hide
  Run.Show
End Sub

Private Sub CommandButton2_Click()
  Me.Hide

  motors.Show
End Sub

Private Sub CommandButton3_Click()
  Me.Hide
  po1prelim.Show
End Sub

Private Sub CommandButton4_Click()
  Me.Hide
  po2on.Show
End Sub

Private Sub CommandButton5_Click()
  Me.Hide
  po3offpreint.Show

End Sub

Private Sub CommandButton6_Click()
  Me.Hide
  po4offnopreint.Show

End Sub

Private Sub CommandButton7_Click()
  Me.Hide
  po5offhyper.Show

End Sub

Private Sub CommandButton8_Click()
  Me.Hide
  po6erect.Show
```

```
44   End Sub

45   Private Sub CommandButton9_Click()
46     Me.Hide
47     po7umbilical.Show

48   End Sub

49   Private Sub done01_Click()

50   End Sub

51   Sub HideArena()
52     Application.Visible = True
53   End Sub

54   Private Sub Label3_Click()

55   End Sub

56   Private Sub TextBox1_Change()

57   End Sub

58   Private Sub UserForm_Click()

59   End Sub

60   Private Sub UserForm_Initialize()
61     done01.Visible = False
62     done02.Visible = False
63     done03.Visible = False
64     done04.Visible = False
65     done05.Visible = False
66     done06.Visible = False
67     done07.Visible = False
68     done08.Visible = False
69     done09.Visible = False
70     done10.Visible = False

71   End Sub
```

Project/Maintenance

```
 1   Private Sub ComboBox1_Change()

 2   End Sub

 3   Private Sub ComboBox2_Change()

 4   End Sub

 5   Private Sub ComboBox31_Change()

 6   End Sub

 7   Private Sub CommandButton2_Click()
 8     Me.Hide
 9     Hierarchy.Show
10   End Sub

11   Private Sub CommandButton3_Click()

12     Dim m As Model
13     Set m = ThisDocument.Model

14     'Dim ma99 As Module
```

```
    'Dim ma99i As Long
    'ma99i = m.Modules.Find(smFindTag, "ma99")
    'Set ma99 = m.Modules(pop99i)
    'pop1.Data("Expression") = mad99.Text
    'pop1.Data("Units") = mau99.Text

    Dim ma01 As Module
    Dim ma01i As Long
    ma01i = m.Modules.Find(smFindTag, "ma01")
    Set ma01 = m.Modules(ma01i)
    ma01.Data("Expression") = mad01.Text
    ma01.Data("Units") = mau01.Text

    Dim ma02 As Module
    Dim ma02i As Long
    ma02i = m.Modules.Find(smFindTag, "ma02")
    Set ma02 = m.Modules(ma02i)
    ma02.Data("Expression") = mad02.Text
    ma02.Data("Units") = mau02.Text

    Dim ma03 As Module
    Dim ma03i As Long
    ma03i = m.Modules.Find(smFindTag, "ma03")
    Set ma03 = m.Modules(ma03i)
    ma03.Data("Expression") = mad03.Text
    ma03.Data("Units") = mau03.Text

    Dim ma04 As Module
    Dim ma04i As Long
    ma04i = m.Modules.Find(smFindTag, "ma04")
    Set ma04 = m.Modules(ma04i)
    ma04.Data("Expression") = mad04.Text
    ma04.Data("Units") = mau04.Text

    Dim ma05 As Module
    Dim ma05i As Long
    ma05i = m.Modules.Find(smFindTag, "ma05")
    Set ma05 = m.Modules(ma05i)
    ma05.Data("Expression") = mad05.Text
    ma05.Data("Units") = mau05.Text

    'This is to turn on the deleted question
    'Dim ma06 As Module
    'Dim ma06i As Long
    'ma06i = m.Modules.Find(smFindTag, "ma06")
    'Set ma06 = m.Modules(ma06i)
    'ma06.Data("Expression") = mad06.Text
    'ma06.Data("Units") = mau06.Text

    Dim ma07 As Module
    Dim ma07i As Long
    ma07i = m.Modules.Find(smFindTag, "ma07")
    Set ma07 = m.Modules(ma07i)
    ma07.Data("Expression") = mad07.Text
    ma07.Data("Units") = mau07.Text

    Dim ma08 As Module
    Dim ma08i As Long
    ma08i = m.Modules.Find(smFindTag, "ma08")
    Set ma08 = m.Modules(ma08i)
    ma08.Data("Expression") = mad08.Text
    ma08.Data("Units") = mau08.Text

    Dim ma09 As Module
    Dim ma09i As Long
    ma09i = m.Modules.Find(smFindTag, "ma09")
    Set ma09 = m.Modules(ma09i)
    ma09.Data("Expression") = mad09.Text
    ma09.Data("Units") = mau09.Text
```

```
    Dim ma10 As Module
    Dim ma10i As Long
    ma10i = m.Modules.Find(smFindTag, "ma10")
    Set ma10 = m.Modules(ma10i)
    ma10.Data("Expression") = mad10.Text
    ma10.Data("Units") = mau10.Text

    Dim ma11 As Module
    Dim ma11i As Long
    ma11i = m.Modules.Find(smFindTag, "ma11")
    Set ma11 = m.Modules(ma11i)
    ma11.Data("Expression") = mad11.Text
    ma11.Data("Units") = mau11.Text

    Dim ma12 As Module
    Dim ma12i As Long
    ma12i = m.Modules.Find(smFindTag, "ma12")
    Set ma12 = m.Modules(ma12i)
    ma12.Data("Expression") = mad12.Text
    ma12.Data("Units") = mau12.Text

    Dim ma13 As Module
    Dim ma13i As Long
    ma13i = m.Modules.Find(smFindTag, "ma13")
    Set ma13 = m.Modules(ma13i)
    ma13.Data("Expression") = mad13.Text
    ma13.Data("Units") = mau13.Text

    Dim ma14 As Module
    Dim ma14i As Long
    ma14i = m.Modules.Find(smFindTag, "ma14")
    Set ma14 = m.Modules(ma14i)
    ma14.Data("Expression") = mad14.Text
    ma14.Data("Units") = mau14.Text

    Dim ma14a As Module
    Dim ma14ai As Long
    ma14ai = m.Modules.Find(smFindTag, "ma14a")
    Set ma14a = m.Modules(ma14ai)
    ma14a.Data("Percent True") = map14a.Text

    Dim ma15 As Module
    Dim ma15i As Long
    ma15i = m.Modules.Find(smFindTag, "ma15")
    Set ma15 = m.Modules(ma15i)
    ma15.Data("Expression") = mad15.Text
    ma15.Data("Units") = mau15.Text

    Dim ma16 As Module
    Dim ma16i As Long
    ma16i = m.Modules.Find(smFindTag, "ma16")
    Set ma16 = m.Modules(ma16i)
    ma16.Data("Expression") = mad16.Text
    ma16.Data("Units") = mau16.Text

    Dim ma17 As Module
    Dim ma17i As Long
    ma17i = m.Modules.Find(smFindTag, "ma17")
    Set ma17 = m.Modules(ma17i)
    ma17.Data("Expression") = mad17.Text
    ma17.Data("Units") = mau17.Text

    Dim ma18 As Module
    Dim ma18i As Long
    ma18i = m.Modules.Find(smFindTag, "ma18")
    Set ma18 = m.Modules(ma18i)
    ma18.Data("Expression") = mad18.Text
    ma18.Data("Units") = mau18.Text
```

```
136     Dim ma19 As Module
137     Dim ma19i As Long
138     ma19i = m.Modules.Find(smFindTag, "ma19")
139     Set ma19 = m.Modules(ma19i)
140     ma19.Data("Expression") = mad19.Text
141     ma19.Data("Units") = mau19.Text

142     Dim ma20 As Module
143     Dim ma20i As Long
144     ma20i = m.Modules.Find(smFindTag, "ma20")
145     Set ma20 = m.Modules(ma20i)
146     ma20.Data("Expression") = mad20.Text
147     ma20.Data("Units") = mau20.Text

148     Dim ma21 As Module
149     Dim ma21i As Long
150     ma21i = m.Modules.Find(smFindTag, "ma21")
151     Set ma21 = m.Modules(ma21i)
152     ma21.Data("Expression") = mad21.Text
153     ma21.Data("Units") = mau21.Text

154     Dim ma22 As Module
155     Dim ma22i As Long
156     ma22i = m.Modules.Find(smFindTag, "ma22")
157     Set ma22 = m.Modules(ma22i)
158     ma22.Data("Expression") = mad22.Text
159     ma22.Data("Units") = mau22.Text

160     Dim ma23 As Module
161     Dim ma23i As Long
162     ma23i = m.Modules.Find(smFindTag, "ma23")
163     Set ma23 = m.Modules(ma23i)
164     ma23.Data("Expression") = mad23.Text
165     ma23.Data("Units") = mau23.Text

166     Dim ma24 As Module
167     Dim ma24i As Long
168     ma24i = m.Modules.Find(smFindTag, "ma24")
169     Set ma24 = m.Modules(ma24i)
170     ma24.Data("Expression") = mad24.Text
171     ma24.Data("Units") = mau24.Text

172     Dim ma25 As Module
173     Dim ma25i As Long
174     ma25i = m.Modules.Find(smFindTag, "ma25")
175     Set ma25 = m.Modules(ma25i)
176     ma25.Data("Expression") = mad25.Text
177     ma25.Data("Units") = mau25.Text

178     Dim ma26 As Module
179     Dim ma26i As Long
180     ma26i = m.Modules.Find(smFindTag, "ma26")
181     Set ma26 = m.Modules(ma26i)
182     ma26.Data("Expression") = mad26.Text
183     ma26.Data("Units") = mau26.Text

184     Dim ma27 As Module
185     Dim ma27i As Long
186     ma27i = m.Modules.Find(smFindTag, "ma27")
187     Set ma27 = m.Modules(ma27i)
188     ma27.Data("Expression") = mad27.Text
189     ma27.Data("Units") = mau27.Text

190     Dim ma28 As Module
191     Dim ma28i As Long
192     ma28i = m.Modules.Find(smFindTag, "ma28")
193     Set ma28 = m.Modules(ma28i)
194     ma28.Data("Expression") = mad28.Text
```

```
195     ma28.Data("Units") = mau28.Text

196     Dim ma29 As Module
197     Dim ma29i As Long
198     ma29i = m.Modules.Find(smFindTag, "ma29")
199     Set ma29 = m.Modules(ma29i)
200     ma29.Data("Expression") = mad29.Text
201     ma29.Data("Units") = mau29.Text

202     Dim ma30 As Module
203     Dim ma30i As Long
204     ma30i = m.Modules.Find(smFindTag, "ma30")
205     Set ma30 = m.Modules(ma30i)
206     ma30.Data("Expression") = mad30.Text
207     ma30.Data("Units") = mau30.Text

208     Dim ma31 As Module
209     Dim ma31i As Long
210     ma31i = m.Modules.Find(smFindTag, "ma31")
211     Set ma31 = m.Modules(ma31i)
212     ma31.Data("Expression") = mad31.Text
213     ma31.Data("Units") = mau31.Text

214     Dim ma32 As Module
215     Dim ma32i As Long
216     ma32i = m.Modules.Find(smFindTag, "ma32")
217     Set ma32 = m.Modules(ma32i)
218     ma32.Data("Expression") = mad32.Text
219     ma32.Data("Units") = mau32.Text

220     Dim ma33 As Module
221     Dim ma33i As Long
222     ma33i = m.Modules.Find(smFindTag, "ma33")
223     Set ma33 = m.Modules(ma33i)
224     ma33.Data("Expression") = mad33.Text
225     ma33.Data("Units") = mau33.Text

226     Dim ma34 As Module
227     Dim ma34i As Long
228     ma34i = m.Modules.Find(smFindTag, "ma34")
229     Set ma34 = m.Modules(ma34i)
230     ma34.Data("Expression") = mad34.Text
231     ma34.Data("Units") = mau34.Text

232     Dim ma35 As Module
233     Dim ma35i As Long
234     ma35i = m.Modules.Find(smFindTag, "ma35")
235     Set ma35 = m.Modules(ma35i)
236     ma35.Data("Expression") = mad35.Text
237     ma35.Data("Units") = mau35.Text

238     Dim ma36 As Module
239     Dim ma36i As Long
240     ma36i = m.Modules.Find(smFindTag, "ma36")
241     Set ma36 = m.Modules(ma36i)
242     ma36.Data("Expression") = mad36.Text
243     ma36.Data("Units") = mau36.Text

244     Dim ma37 As Module
245     Dim ma37i As Long
246     ma37i = m.Modules.Find(smFindTag, "ma37")
247     Set ma37 = m.Modules(ma37i)
248     ma37.Data("Expression") = mad37.Text
249     ma37.Data("Units") = mau37.Text

253     'Dim ma38 As Module
254     'Dim ma38i As Long
255     'ma38i = m.Modules.Find(smFindTag, "ma38")
256     'Set ma38 = m.Modules(ma38i)
```

```
    'ma38.Data("Expression") = map38.Text

    'Dim ma39 As Module
    'Dim ma39i As Long
    'ma39i = m.Modules.Find(smFindTag, "ma39")
    'Set ma39 = m.Modules(ma39i)
    'ma39.Data("Expression") = mad39.Text
    'ma39.Data("Units") = mau39.Text

    Hierarchy.done01.Visible = True
    Me.Hide
    motors.Show
  End Sub

  Private Sub CommandButton4_Click()
    Dim m As Model
    Set m = ThisDocument.Model

    'Dim ma99 As Module
    'Dim ma99i As Long
    'ma99i = m.Modules.Find(smFindTag, "ma99")
    'Set ma99 = m.Modules(pop99i)
    'pop1.Data("Expression") = mad99.Text
    'pop1.Data("Units") = mau99.Text

    Dim ma01 As Module
    Dim ma01i As Long
    ma01i = m.Modules.Find(smFindTag, "ma01")
    Set ma01 = m.Modules(ma01i)
    ma01.Data("Expression") = mad01.Text
    ma01.Data("Units") = mau01.Text

    Dim ma02 As Module
    Dim ma02i As Long
    ma02i = m.Modules.Find(smFindTag, "ma02")
    Set ma02 = m.Modules(ma02i)
    ma02.Data("Expression") = mad02.Text
    ma02.Data("Units") = mau02.Text

    Dim ma03 As Module
    Dim ma03i As Long
    ma03i = m.Modules.Find(smFindTag, "ma03")
    Set ma03 = m.Modules(ma03i)
    ma03.Data("Expression") = mad03.Text
    ma03.Data("Units") = mau03.Text

    Dim ma04 As Module
    Dim ma04i As Long
    ma04i = m.Modules.Find(smFindTag, "ma04")
    Set ma04 = m.Modules(ma04i)
    ma04.Data("Expression") = mad04.Text
    ma04.Data("Units") = mau04.Text

    Dim ma05 As Module
    Dim ma05i As Long
    ma05i = m.Modules.Find(smFindTag, "ma05")
    Set ma05 = m.Modules(ma05i)
    ma05.Data("Expression") = mad05.Text
    ma05.Data("Units") = mau05.Text

    'This is to turn on the deleted question
    'Dim ma06 As Module
    'Dim ma06i As Long
    'ma06i = m.Modules.Find(smFindTag, "ma06")
    'Set ma06 = m.Modules(ma06i)
    'ma06.Data("Expression") = mad06.Text
    'ma06.Data("Units") = mau06.Text

    Dim ma07 As Module
```

```
318     Dim ma07i As Long
319     ma07i = m.Modules.Find(smFindTag, "ma07")
320     Set ma07 = m.Modules(ma07i)
321     ma07.Data("Expression") = mad07.Text
322     ma07.Data("Units") = mau07.Text

323     Dim ma08 As Module
324     Dim ma08i As Long
325     ma08i = m.Modules.Find(smFindTag, "ma08")
326     Set ma08 = m.Modules(ma08i)
327     ma08.Data("Expression") = mad08.Text
328     ma08.Data("Units") = mau08.Text

329     Dim ma09 As Module
330     Dim ma09i As Long
331     ma09i = m.Modules.Find(smFindTag, "ma09")
332     Set ma09 = m.Modules(ma09i)
333     ma09.Data("Expression") = mad09.Text
334     ma09.Data("Units") = mau09.Text

335     Dim ma10 As Module
336     Dim ma10i As Long
337     ma10i = m.Modules.Find(smFindTag, "ma10")
338     Set ma10 = m.Modules(ma10i)
339     ma10.Data("Expression") = mad10.Text
340     ma10.Data("Units") = mau10.Text

341     Dim ma11 As Module
342     Dim ma11i As Long
343     ma11i = m.Modules.Find(smFindTag, "ma11")
344     Set ma11 = m.Modules(ma11i)
345     ma11.Data("Expression") = mad11.Text
346     ma11.Data("Units") = mau11.Text

347     Dim ma12 As Module
348     Dim ma12i As Long
349     ma12i = m.Modules.Find(smFindTag, "ma12")
350     Set ma12 = m.Modules(ma12i)
351     ma12.Data("Expression") = mad12.Text
352     ma12.Data("Units") = mau12.Text

353     Dim ma13 As Module
354     Dim ma13i As Long
355     ma13i = m.Modules.Find(smFindTag, "ma13")
356     Set ma13 = m.Modules(ma13i)
357     ma13.Data("Expression") = mad13.Text
358     ma13.Data("Units") = mau13.Text

359     Dim ma14 As Module
360     Dim ma14i As Long
361     ma14i = m.Modules.Find(smFindTag, "ma14")
362     Set ma14 = m.Modules(ma14i)
363     ma14.Data("Expression") = mad14.Text
364     ma14.Data("Units") = mau14.Text

365     Dim ma15 As Module
366     Dim ma15i As Long
367     ma15i = m.Modules.Find(smFindTag, "ma15")
368     Set ma15 = m.Modules(ma15i)
369     ma15.Data("Expression") = mad15.Text
370     ma15.Data("Units") = mau15.Text

371     Dim ma16 As Module
372     Dim ma16i As Long
373     ma16i = m.Modules.Find(smFindTag, "ma16")
374     Set ma16 = m.Modules(ma16i)
375     ma16.Data("Expression") = mad16.Text
376     ma16.Data("Units") = mau16.Text
```

```
377     Dim ma17 As Module
378     Dim ma17i As Long
379     ma17i = m.Modules.Find(smFindTag, "ma17")
380     Set ma17 = m.Modules(ma17i)
381     ma17.Data("Expression") = mad17.Text
382     ma17.Data("Units") = mau17.Text

383     Dim ma18 As Module
384     Dim ma18i As Long
385     ma18i = m.Modules.Find(smFindTag, "ma18")
386     Set ma18 = m.Modules(ma18i)
387     ma18.Data("Expression") = mad18.Text
388     ma18.Data("Units") = mau18.Text

389     Dim ma19 As Module
390     Dim ma19i As Long
391     ma19i = m.Modules.Find(smFindTag, "ma19")
392     Set ma19 = m.Modules(ma19i)
393     ma19.Data("Expression") = mad19.Text
394     ma19.Data("Units") = mau19.Text

395     Dim ma20 As Module
396     Dim ma20i As Long
397     ma20i = m.Modules.Find(smFindTag, "ma20")
398     Set ma20 = m.Modules(ma20i)
399     ma20.Data("Expression") = mad20.Text
400     ma20.Data("Units") = mau20.Text

401     Dim ma21 As Module
402     Dim ma21i As Long
403     ma21i = m.Modules.Find(smFindTag, "ma21")
404     Set ma21 = m.Modules(ma21i)
405     ma21.Data("Expression") = mad21.Text
406     ma21.Data("Units") = mau21.Text

407     Dim ma22 As Module
408     Dim ma22i As Long
409     ma22i = m.Modules.Find(smFindTag, "ma22")
410     Set ma22 = m.Modules(ma22i)
411     ma22.Data("Expression") = mad22.Text
412     ma22.Data("Units") = mau22.Text

413     Dim ma23 As Module
414     Dim ma23i As Long
415     ma23i = m.Modules.Find(smFindTag, "ma23")
416     Set ma23 = m.Modules(ma23i)
417     ma23.Data("Expression") = mad23.Text
418     ma23.Data("Units") = mau23.Text

419     Dim ma24 As Module
420     Dim ma24i As Long
421     ma24i = m.Modules.Find(smFindTag, "ma24")
422     Set ma24 = m.Modules(ma24i)
423     ma24.Data("Expression") = mad24.Text
424     ma24.Data("Units") = mau24.Text

425     Dim ma25 As Module
426     Dim ma25i As Long
427     ma25i = m.Modules.Find(smFindTag, "ma25")
428     Set ma25 = m.Modules(ma25i)
429     ma25.Data("Expression") = mad25.Text
430     ma25.Data("Units") = mau25.Text

431     Dim ma26 As Module
432     Dim ma26i As Long
433     ma26i = m.Modules.Find(smFindTag, "ma26")
434     Set ma26 = m.Modules(ma26i)
435     ma26.Data("Expression") = mad26.Text
436     ma26.Data("Units") = mau26.Text
```

```
437     Dim ma27 As Module
438     Dim ma27i As Long
439     ma27i = m.Modules.Find(smFindTag, "ma27")
440     Set ma27 = m.Modules(ma27i)
441     ma27.Data("Expression") = mad27.Text
442     ma27.Data("Units") = mau27.Text

443     Dim ma28 As Module
444     Dim ma28i As Long
445     ma28i = m.Modules.Find(smFindTag, "ma28")
446     Set ma28 = m.Modules(ma28i)
447     ma28.Data("Expression") = mad28.Text
448     ma28.Data("Units") = mau28.Text

449     Dim ma29 As Module
450     Dim ma29i As Long
451     ma29i = m.Modules.Find(smFindTag, "ma29")
452     Set ma29 = m.Modules(ma29i)
453     ma29.Data("Expression") = mad29.Text
454     ma29.Data("Units") = mau29.Text

455     Dim ma30 As Module
456     Dim ma30i As Long
457     ma30i = m.Modules.Find(smFindTag, "ma30")
458     Set ma30 = m.Modules(ma30i)
459     ma30.Data("Expression") = mad30.Text
460     ma30.Data("Units") = mau30.Text

461     Dim ma31 As Module
462     Dim ma31i As Long
463     ma31i = m.Modules.Find(smFindTag, "ma31")
464     Set ma31 = m.Modules(ma31i)
465     ma31.Data("Expression") = mad31.Text
466     ma31.Data("Units") = mau31.Text

467     Dim ma32 As Module
468     Dim ma32i As Long
469     ma32i = m.Modules.Find(smFindTag, "ma32")
470     Set ma32 = m.Modules(ma32i)
471     ma32.Data("Expression") = mad32.Text
472     ma32.Data("Units") = mau32.Text

473     Dim ma33 As Module
474     Dim ma33i As Long
475     ma33i = m.Modules.Find(smFindTag, "ma33")
476     Set ma33 = m.Modules(ma33i)
477     ma33.Data("Expression") = mad33.Text
478     ma33.Data("Units") = mau33.Text

479     Dim ma34 As Module
480     Dim ma34i As Long
481     ma34i = m.Modules.Find(smFindTag, "ma34")
482     Set ma34 = m.Modules(ma34i)
483     ma34.Data("Expression") = mad34.Text
484     ma34.Data("Units") = mau34.Text

485     Dim ma35 As Module
486     Dim ma35i As Long
487     ma35i = m.Modules.Find(smFindTag, "ma35")
488     Set ma35 = m.Modules(ma35i)
489     ma35.Data("Expression") = mad35.Text
490     ma35.Data("Units") = mau35.Text

491     Dim ma36 As Module
492     Dim ma36i As Long
493     ma36i = m.Modules.Find(smFindTag, "ma36")
494     Set ma36 = m.Modules(ma36i)
495     ma36.Data("Expression") = mad36.Text
```

```
496     ma36.Data("Units") = mau36.Text

497     Dim ma37 As Module
498     Dim ma37i As Long
499     ma37i = m.Modules.Find(smFindTag, "ma37")
500     Set ma37 = m.Modules(ma37i)
501     ma37.Data("Expression") = mad37.Text
502     ma37.Data("Units") = mau37.Text

506     Dim ma38 As Module
507     Dim ma38i As Long
508     ma38i = m.Modules.Find(smFindTag, "ma38")
509     Set ma38 = m.Modules(ma38i)
510     'ma38.Data("Expression") = map38.Text

514     Dim ma39 As Module
515     Dim ma39i As Long
516     ma39i = m.Modules.Find(smFindTag, "ma39")
517     Set ma39 = m.Modules(ma39i)
518     ma39.Data("Expression") = mad39.Text
519     ma39.Data("Units") = mau39.Text

520     Hierarchy.done01.Visible = True
521     Me.Hide
522     Hierarchy.Show
523   End Sub

524   Private Sub Frame2_Click()

525   End Sub

526   Private Sub Frame3_Click()

527   End Sub

528   Private Sub Label1_Click()

529   End Sub

530   Private Sub Label10_Click()

531   End Sub

532   Private Sub Label15_Click()

533   End Sub

534   Private Sub Label2_Click()

535   End Sub

536   Private Sub Label24_Click()

537   End Sub

538   Private Sub Label27_Click()

539   End Sub

540   Private Sub Label3_Click()

541   End Sub

542   Private Sub Label31_Click()

543   End Sub

544   Private Sub Label32_Click()
```

```
545   End Sub
546   Private Sub Label33_Click()
547   End Sub
548   Private Sub Label35_Click()
549   End Sub
550   Private Sub Label38_Click()
551   End Sub
552   Private Sub Label39_Click()
553   End Sub
554   Private Sub Label40_Click()
555   End Sub
556   Private Sub Label42_Click()
557   End Sub
558   Private Sub Label43_Click()
559   End Sub
560   Private Sub ma14a_Change()
561   End Sub
562   Private Sub mad01_Change()
563   End Sub
564   Private Sub mad04_Change()
565   End Sub
566   Private Sub mad07_Change()
567   End Sub
568   Private Sub mad14_Change()
569   End Sub
570   Private Sub mad34_Change()
571   End Sub
572   Private Sub MaHelp_Click()
573     Me.Hide
574     MaHelp.Show
575   End Sub
576   Private Sub MaintenanceHelp_Click()
577     Me.Hide
578     MaHelp.Show
579   End Sub
580   Private Sub MaintenancePrevious_Click()
581     Me.Hide
```

```
582   End Sub
583   Private Sub map38_Change()
584   End Sub
585   Private Sub mau06_Change()
586   End Sub
587   Private Sub UserForm_Click()
588   End Sub
589   Private Sub UserForm_Initialize()
590     'Code below populates all questions
591     Dim m As Model
592     Set m = ThisDocument.Model
593     Dim mad01 As Module
594     Dim mad01i As Long
595     Dim mad01v As String
596     mad01i = m.Modules.Find(smFindTag, "ma01")
597     Set mad01 = m.Modules(mad01i)
598     mad01v = mad01.Data("Expression")
599     Maintenance.mad01.value = mad01v
600     Maintenance.mad01.AddItem "TRIA ( 15, 30, 42 )", 0
601     Maintenance.mad01.AddItem "TRIA ( Min, Mode, Max )", 1
602     Maintenance.mad01.AddItem "NORM ( Mean, StdDev )", 2
603     Maintenance.mad01.AddItem "EXPO ( Mean )", 3
604     Maintenance.mad01.AddItem "UNIF ( Min, Max )", 4
605     Dim mau01u As Module
606     Dim mau01ui As Long
607     Dim mau01uv As String
608     mau01ui = m.Modules.Find(smFindTag, "ma01")
609     Set mau01u = m.Modules(mau01ui)
610     mau01uv = mau01u.Data("Units")
611     Maintenance.mau01.value = mau01uv
612     Maintenance.mau01.AddItem "Seconds", 0
613     Maintenance.mau01.AddItem "Minutes", 1
614     Maintenance.mau01.AddItem "Hours", 2
615     Maintenance.mau01.AddItem "Days", 3
616     Dim mad02 As Module
617     Dim mad02i As Long
618     Dim mad02v As String
619     mad02i = m.Modules.Find(smFindTag, "ma02")
620     Set mad02 = m.Modules(mad02i)
621     mad02v = mad02.Data("Expression")
622     Maintenance.mad02.value = mad02v
623     Maintenance.mad02.AddItem "TRIA ( 17, 30, 36 )", 0
624     Maintenance.mad02.AddItem "TRIA ( Min, Mode, Max )", 1
625     Maintenance.mad02.AddItem "NORM ( Mean, StdDev )", 2
626     Maintenance.mad02.AddItem "EXPO ( Mean )", 3
627     Maintenance.mad02.AddItem "UNIF ( Min, Max )", 4
628     Dim mau02u As Module
629     Dim mau02ui As Long
630     Dim mau02uv As String
631     mau02ui = m.Modules.Find(smFindTag, "ma02")
632     Set mau02u = m.Modules(mau02ui)
633     mau02uv = mau02u.Data("Units")
634     Maintenance.mau02.value = mau02uv
635     Maintenance.mau02.AddItem "Seconds", 0
636     Maintenance.mau02.AddItem "Minutes", 1
637     Maintenance.mau02.AddItem "Hours", 2
638     Maintenance.mau02.AddItem "Days", 3
```

```
639     Dim mad03 As Module
640     Dim mad03i As Long
641     Dim mad03v As String
642     mad03i = m.Modules.Find(smFindTag, "ma03")
643     Set mad03 = m.Modules(mad03i)
644     mad03v = mad03.Data("Expression")
645     Maintenance.mad03.value = mad03v
646     Maintenance.mad03.AddItem "TRIA ( 54, 60, 84 )", 0
647     Maintenance.mad03.AddItem "TRIA ( Min, Mode, Max )", 1
648     Maintenance.mad03.AddItem "NORM ( Mean, StdDev )", 2
649     Maintenance.mad03.AddItem "EXPO ( Mean )", 3
650     Maintenance.mad03.AddItem "UNIF ( Min, Max )", 4
651     Dim mau03u As Module
652     Dim mau03ui As Long
653     Dim mau03uv As String
654     mau03ui = m.Modules.Find(smFindTag, "ma03")
655     Set mau03u = m.Modules(mau03ui)
656     mau03uv = mau03u.Data("Units")
657     Maintenance.mau03.value = mau03uv
658     Maintenance.mau03.AddItem "Seconds", 0
659     Maintenance.mau03.AddItem "Minutes", 1
660     Maintenance.mau03.AddItem "Hours", 2
661     Maintenance.mau03.AddItem "Days", 3

662     Dim mad04 As Module
663     Dim mad04i As Long
664     Dim mad04v As String
665     mad04i = m.Modules.Find(smFindTag, "ma04")
666     Set mad04 = m.Modules(mad04i)
667     mad04v = mad04.Data("Expression")
668     Maintenance.mad04.value = mad04v
669     Maintenance.mad04.AddItem "TRIA ( 16, 20, 24 )", 0
670     Maintenance.mad04.AddItem "TRIA ( Min, Mode, Max )", 1
671     Maintenance.mad04.AddItem "NORM ( Mean, StdDev )", 2
672     Maintenance.mad04.AddItem "EXPO ( Mean )", 3
673     Maintenance.mad04.AddItem "UNIF ( Min, Max )", 4
674     Dim mau04u As Module
675     Dim mau04ui As Long
676     Dim mau04uv As String
677     mau04ui = m.Modules.Find(smFindTag, "ma04")
678     Set mau04u = m.Modules(mau04ui)
679     mau04uv = mau04u.Data("Units")
680     Maintenance.mau04.value = mau04uv
681     Maintenance.mau04.AddItem "Seconds", 0
682     Maintenance.mau04.AddItem "Minutes", 1
683     Maintenance.mau04.AddItem "Hours", 2
684     Maintenance.mau04.AddItem "Days", 3

685     Dim mad05 As Module
686     Dim mad05i As Long
687     Dim mad05v As String
688     mad05i = m.Modules.Find(smFindTag, "ma05")
689     Set mad05 = m.Modules(mad05i)
690     mad05v = mad05.Data("Expression")
691     Maintenance.mad05.value = mad05v
692     Maintenance.mad05.AddItem "TRIA ( 8, 10, 14 )", 0
693     Maintenance.mad05.AddItem "TRIA ( Min, Mode, Max )", 1
694     Maintenance.mad05.AddItem "NORM ( Mean, StdDev )", 2
695     Maintenance.mad05.AddItem "EXPO ( Mean )", 3
696     Maintenance.mad05.AddItem "UNIF ( Min, Max )", 4
697     Dim mau05u As Module
698     Dim mau05ui As Long
699     Dim mau05uv As String
700     mau05ui = m.Modules.Find(smFindTag, "ma05")
701     Set mau05u = m.Modules(mau05ui)
702     mau05uv = mau05u.Data("Units")
703     Maintenance.mau05.value = mau05uv
704     Maintenance.mau05.AddItem "Seconds", 0
705     Maintenance.mau05.AddItem "Minutes", 1
```

```
706     Maintenance.mau05.AddItem "Hours", 2
707     Maintenance.mau05.AddItem "Days", 3

708     'To add a module to the Arena software just add it in and then right click
709     'the process module and change the tag to "mad06" without the quotes
710     'remove the "'" from the following statements to activate this question
711     'Dim m As Model
712     'Set m = ThisDocument.Model
713     'Dim mad06 As Module
714     'Dim mad06i As Long
715     'Dim mad06v As String
716     'mad06i = m.Modules.Find(smFindTag, "mad06")
717     'Set mad06 = m.Modules(mad06i)
718     'mad06v = mad06.Data("Expression")
719     'Maintenance.mad06.value = mad06v
720     'Maintenance.mad06.AddItem "TRIA ( 27, 30, 42 )", 0
721     'Maintenance.mad06.AddItem "TRIA ( Min, Mode, Max )", 1
722     'Maintenance.mad06.AddItem "NORM ( Mean, StdDev )", 2
723     'Maintenance.mad06.AddItem "EXPO ( Mean )", 3
724     'Maintenance.mad06.AddItem "UNIF ( Min, Max )", 4
725     'Dim mau06u As Module
726     'Dim mau06ui As Long
727     'Dim mau06uv As String
728     'mau06ui = m.Modules.Find(smFindTag, "ma06")
729     'Set mau06u = m.Modules(mau06ui)
730     'mau06uv = mau06u.Data("Units")
731     'Maintenance.mau06.value = mau06uv
732     'Maintenance.mau06.AddItem "Seconds", 0
733     'Maintenance.mau06.AddItem "Minutes", 1
734     'Maintenance.mau06.AddItem "Hours", 2
735     'Maintenance.mau06.AddItem "Days", 3

736     Dim mad07 As Module
737     Dim mad07i As Long
738     Dim mad07v As String
739     mad07i = m.Modules.Find(smFindTag, "ma07")
740     Set mad07 = m.Modules(mad07i)
741     mad07v = mad07.Data("Expression")
742     Maintenance.mad07.value = mad07v
743     Maintenance.mad07.AddItem "TRIA ( 8, 10, 14 )", 0
744     Maintenance.mad07.AddItem "TRIA ( Min, Mode, Max )", 1
745     Maintenance.mad07.AddItem "NORM ( Mean, StdDev )", 2
746     Maintenance.mad07.AddItem "EXPO ( Mean )", 3
747     Maintenance.mad07.AddItem "UNIF ( Min, Max )", 4
748     Dim mau07u As Module
749     Dim mau07ui As Long
750     Dim mau07uv As String
751     mau07ui = m.Modules.Find(smFindTag, "ma07")
752     Set mau07u = m.Modules(mau07ui)
753     mau07uv = mau07u.Data("Units")
754     Maintenance.mau07.value = mau07uv
755     Maintenance.mau07.AddItem "Seconds", 0
756     Maintenance.mau07.AddItem "Minutes", 1
757     Maintenance.mau07.AddItem "Hours", 2
758     Maintenance.mau07.AddItem "Days", 3

759     Dim mad08 As Module
760     Dim mad08i As Long
761     Dim mad08v As String
762     mad08i = m.Modules.Find(smFindTag, "ma08")
763     Set mad08 = m.Modules(mad08i)
764     mad08v = mad08.Data("Expression")
765     Maintenance.mad08.value = mad08v
766     Maintenance.mad08.AddItem "TRIA ( 54, 60, 72 )", 0
767     Maintenance.mad08.AddItem "TRIA ( Min, Mode, Max )", 1
768     Maintenance.mad08.AddItem "NORM ( Mean, StdDev )", 2
769     Maintenance.mad08.AddItem "EXPO ( Mean )", 3
770     Maintenance.mad08.AddItem "UNIF ( Min, Max )", 4
771     Dim mau08u As Module
```

```
772     Dim mau08ui As Long
773     Dim mau08uv As String
774     mau08ui = m.Modules.Find(smFindTag, "ma08")
775     Set mau08u = m.Modules(mau08ui)
776     mau08uv = mau08u.Data("Units")
777     Maintenance.mau08.value = mau08uv
778     Maintenance.mau08.AddItem "Seconds", 0
779     Maintenance.mau08.AddItem "Minutes", 1
780     Maintenance.mau08.AddItem "Hours", 2
781     Maintenance.mau08.AddItem "Days", 3

782     Dim mad09 As Module
783     Dim mad09i As Long
784     Dim mad09v As String
785     mad09i = m.Modules.Find(smFindTag, "ma09")
786     Set mad09 = m.Modules(mad09i)
787     mad09v = mad09.Data("Expression")
788     Maintenance.mad09.value = mad09v
789     Maintenance.mad09.AddItem "TRIA ( 8, 10, 14 )", 0
790     Maintenance.mad09.AddItem "TRIA ( Min, Mode, Max )", 1
791     Maintenance.mad09.AddItem "NORM ( Mean, StdDev )", 2
792     Maintenance.mad09.AddItem "EXPO ( Mean )", 3
793     Maintenance.mad09.AddItem "UNIF ( Min, Max )", 4
794     Dim mau09u As Module
795     Dim mau09ui As Long
796     Dim mau09uv As String
797     mau09ui = m.Modules.Find(smFindTag, "ma09")
798     Set mau09u = m.Modules(mau09ui)
799     mau09uv = mau09u.Data("Units")
800     Maintenance.mau09.value = mau09uv
801     Maintenance.mau09.AddItem "Seconds", 0
802     Maintenance.mau09.AddItem "Minutes", 1
803     Maintenance.mau09.AddItem "Hours", 2
804     Maintenance.mau09.AddItem "Days", 3

805     Dim mad10 As Module
806     Dim mad10i As Long
807     Dim mad10v As String
808     mad10i = m.Modules.Find(smFindTag, "ma10")
809     Set mad10 = m.Modules(mad10i)
810     mad10v = mad10.Data("Expression")
811     Maintenance.mad10.value = mad10v
812     Maintenance.mad10.AddItem "TRIA ( 54, 60, 84 )", 0
813     Maintenance.mad10.AddItem "TRIA ( Min, Mode, Max )", 1
814     Maintenance.mad10.AddItem "NORM ( Mean, StdDev )", 2
815     Maintenance.mad10.AddItem "EXPO ( Mean )", 3
816     Maintenance.mad10.AddItem "UNIF ( Min, Max )", 4
817     Dim mau10u As Module
818     Dim mau10ui As Long
819     Dim mau10uv As String
820     mau10ui = m.Modules.Find(smFindTag, "ma10")
821     Set mau10u = m.Modules(mau10ui)
822     mau10uv = mau10u.Data("Units")
823     Maintenance.mau10.value = mau10uv
824     Maintenance.mau10.AddItem "Seconds", 0
825     Maintenance.mau10.AddItem "Minutes", 1
826     Maintenance.mau10.AddItem "Hours", 2
827     Maintenance.mau10.AddItem "Days", 3

828     Dim mad11 As Module
829     Dim mad11i As Long
830     Dim mad11v As String
831     mad11i = m.Modules.Find(smFindTag, "ma11")
832     Set mad11 = m.Modules(mad11i)
833     mad11v = mad11.Data("Expression")
834     Maintenance.mad11.value = mad11v
835     Maintenance.mad11.AddItem "TRIA ( 27, 30, 42 )", 0
836     Maintenance.mad11.AddItem "TRIA ( Min, Mode, Max )", 1
837     Maintenance.mad11.AddItem "NORM ( Mean, StdDev )", 2
```

```
838     Maintenance.mad11.AddItem "EXPO ( Mean )", 3
839     Maintenance.mad11.AddItem "UNIF ( Min, Max )", 4
840     Dim mau11u As Module
841     Dim mau11ui As Long
842     Dim mau11uv As String
843     mau11ui = m.Modules.Find(smFindTag, "ma11")
844     Set mau11u = m.Modules(mau11ui)
845     mau11uv = mau11u.Data("Units")
846     Maintenance.mau11.value = mau11uv
847     Maintenance.mau11.AddItem "Seconds", 0
848     Maintenance.mau11.AddItem "Minutes", 1
849     Maintenance.mau11.AddItem "Hours", 2
850     Maintenance.mau11.AddItem "Days", 3

851     Dim mad12 As Module
852     Dim mad12i As Long
853     Dim mad12v As String
854     mad12i = m.Modules.Find(smFindTag, "ma12")
855     Set mad12 = m.Modules(mad12i)
856     mad12v = mad12.Data("Expression")
857     Maintenance.mad12.value = mad12v
858     Maintenance.mad12.AddItem "TRIA ( 54, 60, 84 )", 0
859     Maintenance.mad12.AddItem "TRIA ( Min, Mode, Max )", 1
860     Maintenance.mad12.AddItem "NORM ( Mean, StdDev )", 2
861     Maintenance.mad12.AddItem "EXPO ( Mean )", 3
862     Maintenance.mad12.AddItem "UNIF ( Min, Max )", 4
863     Dim mau12u As Module
864     Dim mau12ui As Long
865     Dim mau12uv As String
866     mau12ui = m.Modules.Find(smFindTag, "ma12")
867     Set mau12u = m.Modules(mau12ui)
868     mau12uv = mau12u.Data("Units")
869     Maintenance.mau12.value = mau12uv
870     Maintenance.mau12.AddItem "Seconds", 0
871     Maintenance.mau12.AddItem "Minutes", 1
872     Maintenance.mau12.AddItem "Hours", 2
873     Maintenance.mau12.AddItem "Days", 3

874     Dim mad13 As Module
875     Dim mad13i As Long
876     Dim mad13v As String
877     mad13i = m.Modules.Find(smFindTag, "ma13")
878     Set mad13 = m.Modules(mad13i)
879     mad13v = mad13.Data("Expression")
880     Maintenance.mad13.value = mad13v
881     Maintenance.mad13.AddItem "TRIA ( 24, 60, 84 )", 0
882     Maintenance.mad13.AddItem "TRIA ( Min, Mode, Max )", 1
883     Maintenance.mad13.AddItem "NORM ( Mean, StdDev )", 2
884     Maintenance.mad13.AddItem "EXPO ( Mean )", 3
885     Maintenance.mad13.AddItem "UNIF ( Min, Max )", 4
886     Dim mau13u As Module
887     Dim mau13ui As Long
888     Dim mau13uv As String
889     mau13ui = m.Modules.Find(smFindTag, "ma13")
890     Set mau13u = m.Modules(mau13ui)
891     mau13uv = mau13u.Data("Units")
892     Maintenance.mau13.value = mau13uv
893     Maintenance.mau13.AddItem "Seconds", 0
894     Maintenance.mau13.AddItem "Minutes", 1
895     Maintenance.mau13.AddItem "Hours", 2
896     Maintenance.mau13.AddItem "Days", 3

897     Dim mad14 As Module
898     Dim mad14i As Long
899     Dim mad14v As String
900     mad14i = m.Modules.Find(smFindTag, "ma14")
901     Set mad14 = m.Modules(mad14i)
902     mad14v = mad14.Data("Expression")
903     Maintenance.mad14.value = mad14v
```

```
Maintenance.mad14.AddItem "TRIA ( 24, 30, 42 )", 0
Maintenance.mad14.AddItem "TRIA ( Min, Mode, Max )", 1
Maintenance.mad14.AddItem "NORM ( Mean, StdDev )", 2
Maintenance.mad14.AddItem "EXPO ( Mean )", 3
Maintenance.mad14.AddItem "UNIF ( Min, Max )", 4
Dim mau14u As Module
Dim mau14ui As Long
Dim mau14uv As String
mau14ui = m.Modules.Find(smFindTag, "ma14")
Set mau14u = m.Modules(mau14ui)
mau14uv = mau14u.Data("Units")
Maintenance.mau14.value = mau14uv
Maintenance.mau14.AddItem "Seconds", 0
Maintenance.mau14.AddItem "Minutes", 1
Maintenance.mau14.AddItem "Hours", 2
Maintenance.mau14.AddItem "Days", 3

Dim ma14a As Module
Dim ma14ai As Long
Dim ma14av As String
ma14ai = m.Modules.Find(smFindTag, "ma14a")
Set ma14a = m.Modules(ma14ai)
ma14av = ma14a.Data("Percent True")
Maintenance.map14a.value = ma14av
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
```

```
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
Maintenance.map14a.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
```

```
1041     Maintenance.map14a.AddItem countnumber, countnumber
1042     If countnumber < 100 Then countnumber = countnumber + 1
1043     Maintenance.map14a.AddItem countnumber, countnumber
1044     If countnumber < 100 Then countnumber = countnumber + 1
1045     Maintenance.map14a.AddItem countnumber, countnumber
1046     If countnumber < 100 Then countnumber = countnumber + 1
1047     Maintenance.map14a.AddItem countnumber, countnumber
1048     If countnumber < 100 Then countnumber = countnumber + 1
1049     Maintenance.map14a.AddItem countnumber, countnumber
1050     If countnumber < 100 Then countnumber = countnumber + 1
1051     Maintenance.map14a.AddItem countnumber, countnumber
1052     If countnumber < 100 Then countnumber = countnumber + 1
1053     Maintenance.map14a.AddItem countnumber, countnumber
1054     If countnumber < 100 Then countnumber = countnumber + 1
1055     Maintenance.map14a.AddItem countnumber, countnumber
1056     If countnumber < 100 Then countnumber = countnumber + 1
1057     Maintenance.map14a.AddItem countnumber, countnumber
1058     If countnumber < 100 Then countnumber = countnumber + 1
1059     Maintenance.map14a.AddItem countnumber, countnumber
1060     If countnumber < 100 Then countnumber = countnumber + 1
1061     Maintenance.map14a.AddItem countnumber, countnumber
1062     If countnumber < 100 Then countnumber = countnumber + 1
1063     Maintenance.map14a.AddItem countnumber, countnumber
1064     If countnumber < 100 Then countnumber = countnumber + 1
1065     Maintenance.map14a.AddItem countnumber, countnumber
1066     If countnumber < 100 Then countnumber = countnumber + 1
1067     Maintenance.map14a.AddItem countnumber, countnumber
1068     If countnumber < 100 Then countnumber = countnumber + 1
1069     Maintenance.map14a.AddItem countnumber, countnumber
1070     If countnumber < 100 Then countnumber = countnumber + 1
1071     Maintenance.map14a.AddItem countnumber, countnumber
1072     If countnumber < 100 Then countnumber = countnumber + 1
1073     Maintenance.map14a.AddItem countnumber, countnumber
1074     If countnumber < 100 Then countnumber = countnumber + 1
1075     Maintenance.map14a.AddItem countnumber, countnumber
1076     If countnumber < 100 Then countnumber = countnumber + 1
1077     Maintenance.map14a.AddItem countnumber, countnumber
1078     If countnumber < 100 Then countnumber = countnumber + 1
1079     Maintenance.map14a.AddItem countnumber, countnumber
1080     If countnumber < 100 Then countnumber = countnumber + 1
1081     Maintenance.map14a.AddItem countnumber, countnumber
1082     If countnumber < 100 Then countnumber = countnumber + 1
1083     Maintenance.map14a.AddItem countnumber, countnumber
1084     If countnumber < 100 Then countnumber = countnumber + 1
1085     Maintenance.map14a.AddItem countnumber, countnumber
1086     If countnumber < 100 Then countnumber = countnumber + 1
1087     Maintenance.map14a.AddItem countnumber, countnumber
1088     If countnumber < 100 Then countnumber = countnumber + 1
1089     Maintenance.map14a.AddItem countnumber, countnumber
1090     If countnumber < 100 Then countnumber = countnumber + 1
1091     Maintenance.map14a.AddItem countnumber, countnumber
1092     If countnumber < 100 Then countnumber = countnumber + 1
1093     Maintenance.map14a.AddItem countnumber, countnumber
1094     If countnumber < 100 Then countnumber = countnumber + 1
1095     Maintenance.map14a.AddItem countnumber, countnumber
1096     If countnumber < 100 Then countnumber = countnumber + 1
1097     Maintenance.map14a.AddItem countnumber, countnumber
1098     If countnumber < 100 Then countnumber = countnumber + 1
1099     Maintenance.map14a.AddItem countnumber, countnumber
1100     If countnumber < 100 Then countnumber = countnumber + 1
1101     Maintenance.map14a.AddItem countnumber, countnumber
1102     If countnumber < 100 Then countnumber = countnumber + 1
1103     Maintenance.map14a.AddItem countnumber, countnumber
1104     If countnumber < 100 Then countnumber = countnumber + 1
1105     Maintenance.map14a.AddItem countnumber, countnumber
1106     If countnumber < 100 Then countnumber = countnumber + 1
1107     Maintenance.map14a.AddItem countnumber, countnumber
1108     If countnumber < 100 Then countnumber = countnumber + 1
1109     Maintenance.map14a.AddItem countnumber, countnumber
```

```
1110     If countnumber < 100 Then countnumber = countnumber + 1
1111     Maintenance.map14a.AddItem countnumber, countnumber
1112     If countnumber < 100 Then countnumber = countnumber + 1
1113     Maintenance.map14a.AddItem countnumber, countnumber
1114     If countnumber < 100 Then countnumber = countnumber + 1
1115     Maintenance.map14a.AddItem countnumber, countnumber
1116     If countnumber < 100 Then countnumber = countnumber + 1
1117     Maintenance.map14a.AddItem countnumber, countnumber
1118     If countnumber < 100 Then countnumber = countnumber + 1
1119     Maintenance.map14a.AddItem countnumber, countnumber
1120     If countnumber < 100 Then countnumber = countnumber + 1
1121     Maintenance.map14a.AddItem countnumber, countnumber
1122     If countnumber < 100 Then countnumber = countnumber + 1
1123     Maintenance.map14a.AddItem countnumber, countnumber
1124     If countnumber < 100 Then countnumber = countnumber + 1
1125     Maintenance.map14a.AddItem countnumber, countnumber
1126     If countnumber < 100 Then countnumber = countnumber + 1
1127     Maintenance.map14a.AddItem countnumber, countnumber

1128     'Dim ma14a As Module
1129     'Dim ma14ai As Long
1130     'Dim ma14av As String
1131     'ma14ai = m.Modules.Find(smFindTag, "ma14a")
1132     'Set ma14a = m.Modules(ma14ai)
1133     'ma14av = ma14a.Data("expression")
1134     'Maintenance.ma14a.value = mad14av

1135     Dim mad15 As Module
1136     Dim mad15i As Long
1137     Dim mad15v As String
1138     mad15i = m.Modules.Find(smFindTag, "ma15")
1139     Set mad15 = m.Modules(mad15i)
1140     mad15v = mad15.Data("Expression")
1141     Maintenance.mad15.value = mad15v
1142     Maintenance.mad15.AddItem "TRIA ( 81, 90, 126 )", 0
1143     Maintenance.mad15.AddItem "TRIA ( Min, Mode, Max )", 1
1144     Maintenance.mad15.AddItem "NORM ( Mean, StdDev )", 2
1145     Maintenance.mad15.AddItem "EXPO ( Mean )", 3
1146     Maintenance.mad15.AddItem "UNIF ( Min, Max )", 4
1147     Dim mau15u As Module
1148     Dim mau15ui As Long
1149     Dim mau15uv As String
1150     mau15ui = m.Modules.Find(smFindTag, "ma15")
1151     Set mau15u = m.Modules(mau15ui)
1152     mau15uv = mau15u.Data("Units")
1153     Maintenance.mau15.value = mau15uv
1154     Maintenance.mau15.AddItem "Seconds", 0
1155     Maintenance.mau15.AddItem "Minutes", 1
1156     Maintenance.mau15.AddItem "Hours", 2
1157     Maintenance.mau15.AddItem "Days", 3

1158     Dim mad16 As Module
1159     Dim mad16i As Long
1160     Dim mad16v As String
1161     mad16i = m.Modules.Find(smFindTag, "ma16")
1162     Set mad16 = m.Modules(mad16i)
1163     mad16v = mad16.Data("Expression")
1164     Maintenance.mad16.value = mad16v
1165     Maintenance.mad16.AddItem "TRIA ( 162, 180, 252 )", 0
1166     Maintenance.mad16.AddItem "TRIA ( Min, Mode, Max )", 1
1167     Maintenance.mad16.AddItem "NORM ( Mean, StdDev )", 2
1168     Maintenance.mad16.AddItem "EXPO ( Mean )", 3
1169     Maintenance.mad16.AddItem "UNIF ( Min, Max )", 4
1170     Dim mau16u As Module
1171     Dim mau16ui As Long
1172     Dim mau16uv As String
1173     mau16ui = m.Modules.Find(smFindTag, "ma16")
1174     Set mau16u = m.Modules(mau16ui)
1175     mau16uv = mau16u.Data("Units")
```

```
1176     Maintenance.mau16.value = mau16uv
1177     Maintenance.mau16.AddItem "Seconds", 0
1178     Maintenance.mau16.AddItem "Minutes", 1
1179     Maintenance.mau16.AddItem "Hours", 2
1180     Maintenance.mau16.AddItem "Days", 3

1181     Dim mad17 As Module
1182     Dim mad17i As Long
1183     Dim mad17v As String
1184     mad17i = m.Modules.Find(smFindTag, "ma17")
1185     Set mad17 = m.Modules(mad17i)
1186     mad17v = mad17.Data("Expression")
1187     Maintenance.mad17.value = mad17v
1188     Maintenance.mad17.AddItem "TRIA ( 108, 120, 168 )", 0
1189     Maintenance.mad17.AddItem "TRIA ( Min, Mode, Max )", 1
1190     Maintenance.mad17.AddItem "NORM ( Mean, StdDev )", 2
1191     Maintenance.mad17.AddItem "EXPO ( Mean )", 3
1192     Maintenance.mad17.AddItem "UNIF ( Min, Max )", 4
1193     Dim mau17u As Module
1194     Dim mau17ui As Long
1195     Dim mau17uv As String
1196     mau17ui = m.Modules.Find(smFindTag, "ma17")
1197     Set mau17u = m.Modules(mau17ui)
1198     mau17uv = mau17u.Data("Units")
1199     Maintenance.mau17.value = mau17uv
1200     Maintenance.mau17.AddItem "Seconds", 0
1201     Maintenance.mau17.AddItem "Minutes", 1
1202     Maintenance.mau17.AddItem "Hours", 2
1203     Maintenance.mau17.AddItem "Days", 3

1204     Dim mad18 As Module
1205     Dim mad18i As Long
1206     Dim mad18v As String
1207     mad18i = m.Modules.Find(smFindTag, "ma18")
1208     Set mad18 = m.Modules(mad18i)
1209     mad18v = mad18.Data("Expression")
1210     Maintenance.mad18.value = mad18v
1211     Maintenance.mad18.AddItem "TRIA ( 54, 60, 84 )", 0
1212     Maintenance.mad18.AddItem "TRIA ( Min, Mode, Max )", 1
1213     Maintenance.mad18.AddItem "NORM ( Mean, StdDev )", 2
1214     Maintenance.mad18.AddItem "EXPO ( Mean )", 3
1215     Maintenance.mad18.AddItem "UNIF ( Min, Max )", 4
1216     Dim mau18u As Module
1217     Dim mau18ui As Long
1218     Dim mau18uv As String
1219     mau18ui = m.Modules.Find(smFindTag, "ma18")
1220     Set mau18u = m.Modules(mau18ui)
1221     mau18uv = mau18u.Data("Units")
1222     Maintenance.mau18.value = mau18uv
1223     Maintenance.mau18.AddItem "Seconds", 0
1224     Maintenance.mau18.AddItem "Minutes", 1
1225     Maintenance.mau18.AddItem "Hours", 2
1226     Maintenance.mau18.AddItem "Days", 3

1227     Dim mad19 As Module
1228     Dim mad19i As Long
1229     Dim mad19v As String
1230     mad19i = m.Modules.Find(smFindTag, "ma19")
1231     Set mad19 = m.Modules(mad19i)
1232     mad19v = mad19.Data("Expression")
1233     Maintenance.mad19.value = mad19v
1234     Maintenance.mad19.AddItem "TRIA ( 8, 10, 14 )", 0
1235     Maintenance.mad19.AddItem "TRIA ( Min, Mode, Max )", 1
1236     Maintenance.mad19.AddItem "NORM ( Mean, StdDev )", 2
1237     Maintenance.mad19.AddItem "EXPO ( Mean )", 3
1238     Maintenance.mad19.AddItem "UNIF ( Min, Max )", 4
1239     Dim mau19u As Module
1240     Dim mau19ui As Long
1241     Dim mau19uv As String
```

```
1242     mau19ui = m.Modules.Find(smFindTag, "ma19")
1243     Set mau19u = m.Modules(mau19ui)
1244     mau19uv = mau19u.Data("Units")
1245     Maintenance.mau19.value = mau19uv
1246     Maintenance.mau19.AddItem "Seconds", 0
1247     Maintenance.mau19.AddItem "Minutes", 1
1248     Maintenance.mau19.AddItem "Hours", 2
1249     Maintenance.mau19.AddItem "Days", 3

1250     Dim mad20 As Module
1251     Dim mad20i As Long
1252     Dim mad20v As String
1253     mad20i = m.Modules.Find(smFindTag, "ma20")
1254     Set mad20 = m.Modules(mad20i)
1255     mad20v = mad20.Data("Expression")
1256     Maintenance.mad20.value = mad20v
1257     Maintenance.mad20.AddItem "TRIA ( 24, 30, 42 )", 0
1258     Maintenance.mad20.AddItem "TRIA ( Min, Mode, Max )", 1
1259     Maintenance.mad20.AddItem "NORM ( Mean, StdDev )", 2
1260     Maintenance.mad20.AddItem "EXPO ( Mean )", 3
1261     Maintenance.mad20.AddItem "UNIF ( Min, Max )", 4
1262     Dim mau20u As Module
1263     Dim mau20ui As Long
1264     Dim mau20uv As String
1265     mau20ui = m.Modules.Find(smFindTag, "ma20")
1266     Set mau20u = m.Modules(mau20ui)
1267     mau20uv = mau20u.Data("Units")
1268     Maintenance.mau20.value = mau20uv
1269     Maintenance.mau20.AddItem "Seconds", 0
1270     Maintenance.mau20.AddItem "Minutes", 1
1271     Maintenance.mau20.AddItem "Hours", 2
1272     Maintenance.mau20.AddItem "Days", 3

1273     Dim mad21 As Module
1274     Dim mad21i As Long
1275     Dim mad21v As String
1276     mad21i = m.Modules.Find(smFindTag, "ma21")
1277     Set mad21 = m.Modules(mad21i)
1278     mad21v = mad21.Data("Expression")
1279     Maintenance.mad21.value = mad21v
1280     Maintenance.mad21.AddItem "TRIA ( 81, 90, 126 )", 0
1281     Maintenance.mad21.AddItem "TRIA ( Min, Mode, Max )", 1
1282     Maintenance.mad21.AddItem "NORM ( Mean, StdDev )", 2
1283     Maintenance.mad21.AddItem "EXPO ( Mean )", 3
1284     Maintenance.mad21.AddItem "UNIF ( Min, Max )", 4
1285     Dim mau21u As Module
1286     Dim mau21ui As Long
1287     Dim mau21uv As String
1288     mau21ui = m.Modules.Find(smFindTag, "ma21")
1289     Set mau21u = m.Modules(mau21ui)
1290     mau21uv = mau21u.Data("Units")
1291     Maintenance.mau21.value = mau21uv
1292     Maintenance.mau21.AddItem "Seconds", 0
1293     Maintenance.mau21.AddItem "Minutes", 1
1294     Maintenance.mau21.AddItem "Hours", 2
1295     Maintenance.mau21.AddItem "Days", 3

1296     Dim mad22 As Module
1297     Dim mad22i As Long
1298     Dim mad22v As String
1299     mad22i = m.Modules.Find(smFindTag, "ma22")
1300     Set mad22 = m.Modules(mad22i)
1301     mad22v = mad22.Data("Expression")
1302     Maintenance.mad22.value = mad22v
1303     Maintenance.mad22.AddItem "TRIA ( 24, 30, 42 )", 0
1304     Maintenance.mad22.AddItem "TRIA ( Min, Mode, Max )", 1
1305     Maintenance.mad22.AddItem "NORM ( Mean, StdDev )", 2
1306     Maintenance.mad22.AddItem "EXPO ( Mean )", 3
1307     Maintenance.mad22.AddItem "UNIF ( Min, Max )", 4
```

```
    Dim mau22u As Module
    Dim mau22ui As Long
    Dim mau22uv As String
    mau22ui = m.Modules.Find(smFindTag, "ma22")
    Set mau22u = m.Modules(mau22ui)
    mau22uv = mau22u.Data("Units")
    Maintenance.mau22.value = mau22uv
    Maintenance.mau22.AddItem "Seconds", 0
    Maintenance.mau22.AddItem "Minutes", 1
    Maintenance.mau22.AddItem "Hours", 2
    Maintenance.mau22.AddItem "Days", 3

    Dim mad23 As Module
    Dim mad23i As Long
    Dim mad23v As String
    mad23i = m.Modules.Find(smFindTag, "ma23")
    Set mad23 = m.Modules(mad23i)
    mad23v = mad23.Data("Expression")
    Maintenance.mad23.value = mad23v
    Maintenance.mad23.AddItem "TRIA ( 81, 90, 126 )", 0
    Maintenance.mad23.AddItem "TRIA ( Min, Mode, Max )", 1
    Maintenance.mad23.AddItem "NORM ( Mean, StdDev )", 2
    Maintenance.mad23.AddItem "EXPO ( Mean )", 3
    Maintenance.mad23.AddItem "UNIF ( Min, Max )", 4
    Dim mau23u As Module
    Dim mau23ui As Long
    Dim mau23uv As String
    mau23ui = m.Modules.Find(smFindTag, "ma23")
    Set mau23u = m.Modules(mau23ui)
    mau23uv = mau23u.Data("Units")
    Maintenance.mau23.value = mau23uv
    Maintenance.mau23.AddItem "Seconds", 0
    Maintenance.mau23.AddItem "Minutes", 1
    Maintenance.mau23.AddItem "Hours", 2
    Maintenance.mau23.AddItem "Days", 3

    Dim mad24 As Module
    Dim mad24i As Long
    Dim mad24v As String
    mad24i = m.Modules.Find(smFindTag, "ma24")
    Set mad24 = m.Modules(mad24i)
    mad24v = mad24.Data("Expression")
    Maintenance.mad24.value = mad24v
    Maintenance.mad24.AddItem "TRIA ( 81, 90, 126 )", 0
    Maintenance.mad24.AddItem "TRIA ( Min, Mode, Max )", 1
    Maintenance.mad24.AddItem "NORM ( Mean, StdDev )", 2
    Maintenance.mad24.AddItem "EXPO ( Mean )", 3
    Maintenance.mad24.AddItem "UNIF ( Min, Max )", 4
    Dim mau24u As Module
    Dim mau24ui As Long
    Dim mau24uv As String
    mau24ui = m.Modules.Find(smFindTag, "ma24")
    Set mau24u = m.Modules(mau24ui)
    mau24uv = mau24u.Data("Units")
    Maintenance.mau24.value = mau24uv
    Maintenance.mau24.AddItem "Seconds", 0
    Maintenance.mau24.AddItem "Minutes", 1
    Maintenance.mau24.AddItem "Hours", 2
    Maintenance.mau24.AddItem "Days", 3

    Dim mad25 As Module
    Dim mad25i As Long
    Dim mad25v As String
    mad25i = m.Modules.Find(smFindTag, "ma25")
    Set mad25 = m.Modules(mad25i)
    mad25v = mad25.Data("Expression")
    Maintenance.mad25.value = mad25v
    Maintenance.mad25.AddItem "TRIA ( 0, 60, 120 )", 0
    Maintenance.mad25.AddItem "TRIA ( Min, Mode, Max )", 1
```

```
    Maintenance.mad25.AddItem "NORM ( Mean, StdDev )", 2
    Maintenance.mad25.AddItem "EXPO ( Mean )", 3
    Maintenance.mad25.AddItem "UNIF ( Min, Max )", 4
    Dim mau25u As Module
    Dim mau25ui As Long
    Dim mau25uv As String
    mau25ui = m.Modules.Find(smFindTag, "ma25")
    Set mau25u = m.Modules(mau25ui)
    mau25uv = mau25u.Data("Units")
    Maintenance.mau25.value = mau25uv
    Maintenance.mau25.AddItem "Seconds", 0
    Maintenance.mau25.AddItem "Minutes", 1
    Maintenance.mau25.AddItem "Hours", 2
    Maintenance.mau25.AddItem "Days", 3

    Dim mad26 As Module
    Dim mad26i As Long
    Dim mad26v As String
    mad26i = m.Modules.Find(smFindTag, "ma26")
    Set mad26 = m.Modules(mad26i)
    mad26v = mad26.Data("Expression")
    Maintenance.mad26.value = mad26v
    Maintenance.mad26.AddItem "TRIA ( 0, 60, 120 )", 0
    Maintenance.mad26.AddItem "TRIA ( Min, Mode, Max )", 1
    Maintenance.mad26.AddItem "NORM ( Mean, StdDev )", 2
    Maintenance.mad26.AddItem "EXPO ( Mean )", 3
    Maintenance.mad26.AddItem "UNIF ( Min, Max )", 4
    Dim mau26u As Module
    Dim mau26ui As Long
    Dim mau26uv As String
    mau26ui = m.Modules.Find(smFindTag, "ma26")
    Set mau26u = m.Modules(mau26ui)
    mau26uv = mau26u.Data("Units")
    Maintenance.mau26.value = mau26uv
    Maintenance.mau26.AddItem "Seconds", 0
    Maintenance.mau26.AddItem "Minutes", 1
    Maintenance.mau26.AddItem "Hours", 2
    Maintenance.mau26.AddItem "Days", 3

    Dim mad27 As Module
    Dim mad27i As Long
    Dim mad27v As String
    mad27i = m.Modules.Find(smFindTag, "ma27")
    Set mad27 = m.Modules(mad27i)
    mad27v = mad27.Data("Expression")
    Maintenance.mad27.value = mad27v
    Maintenance.mad27.AddItem "TRIA ( 30, 180, 360 )", 0
    Maintenance.mad27.AddItem "TRIA ( Min, Mode, Max )", 1
    Maintenance.mad27.AddItem "NORM ( Mean, StdDev )", 2
    Maintenance.mad27.AddItem "EXPO ( Mean )", 3
    Maintenance.mad27.AddItem "UNIF ( Min, Max )", 4
    Dim mau27u As Module
    Dim mau27ui As Long
    Dim mau27uv As String
    mau27ui = m.Modules.Find(smFindTag, "ma27")
    Set mau27u = m.Modules(mau27ui)
    mau27uv = mau27u.Data("Units")
    Maintenance.mau27.value = mau27uv
    Maintenance.mau27.AddItem "Seconds", 0
    Maintenance.mau27.AddItem "Minutes", 1
    Maintenance.mau27.AddItem "Hours", 2
    Maintenance.mau27.AddItem "Days", 3

    Dim mad28 As Module
    Dim mad28i As Long
    Dim mad28v As String
    mad28i = m.Modules.Find(smFindTag, "ma28")
    Set mad28 = m.Modules(mad28i)
    mad28v = mad28.Data("Expression")
```

```
1440     Maintenance.mad28.value = mad28v
1441     Maintenance.mad28.AddItem "TRIA ( 108, 120, 168 )", 0
1442     Maintenance.mad28.AddItem "TRIA ( Min, Mode, Max )", 1
1443     Maintenance.mad28.AddItem "NORM ( Mean, StdDev )", 2
1444     Maintenance.mad28.AddItem "EXPO ( Mean )", 3
1445     Maintenance.mad28.AddItem "UNIF ( Min, Max )", 4
1446     Dim mau28u As Module
1447     Dim mau28ui As Long
1448     Dim mau28uv As String
1449     mau28ui = m.Modules.Find(smFindTag, "ma28")
1450     Set mau28u = m.Modules(mau28ui)
1451     mau28uv = mau28u.Data("Units")
1452     Maintenance.mau28.value = mau28uv
1453     Maintenance.mau28.AddItem "Seconds", 0
1454     Maintenance.mau28.AddItem "Minutes", 1
1455     Maintenance.mau28.AddItem "Hours", 2
1456     Maintenance.mau28.AddItem "Days", 3

1457     Dim mad29 As Module
1458     Dim mad29i As Long
1459     Dim mad29v As String
1460     mad29i = m.Modules.Find(smFindTag, "ma29")
1461     Set mad29 = m.Modules(mad29i)
1462     mad29v = mad29.Data("Expression")
1463     Maintenance.mad29.value = mad29v
1464     Maintenance.mad29.AddItem "TRIA ( 108, 120, 168 )", 0
1465     Maintenance.mad29.AddItem "TRIA ( Min, Mode, Max )", 1
1466     Maintenance.mad29.AddItem "NORM ( Mean, StdDev )", 2
1467     Maintenance.mad29.AddItem "EXPO ( Mean )", 3
1468     Maintenance.mad29.AddItem "UNIF ( Min, Max )", 4
1469     Dim mau29u As Module
1470     Dim mau29ui As Long
1471     Dim mau29uv As String
1472     mau29ui = m.Modules.Find(smFindTag, "ma29")
1473     Set mau29u = m.Modules(mau29ui)
1474     mau29uv = mau29u.Data("Units")
1475     Maintenance.mau29.value = mau29uv
1476     Maintenance.mau29.AddItem "Seconds", 0
1477     Maintenance.mau29.AddItem "Minutes", 1
1478     Maintenance.mau29.AddItem "Hours", 2
1479     Maintenance.mau29.AddItem "Days", 3

1480     Dim mad30 As Module
1481     Dim mad30i As Long
1482     Dim mad30v As String
1483     mad30i = m.Modules.Find(smFindTag, "ma30")
1484     Set mad30 = m.Modules(mad30i)
1485     mad30v = mad30.Data("Expression")
1486     Maintenance.mad30.value = mad30v
1487     Maintenance.mad30.AddItem "TRIA ( 0, 120, 180 )", 0
1488     Maintenance.mad30.AddItem "TRIA ( Min, Mode, Max )", 1
1489     Maintenance.mad30.AddItem "NORM ( Mean, StdDev )", 2
1490     Maintenance.mad30.AddItem "EXPO ( Mean )", 3
1491     Maintenance.mad30.AddItem "UNIF ( Min, Max )", 4
1492     Dim mau30u As Module
1493     Dim mau30ui As Long
1494     Dim mau30uv As String
1495     mau30ui = m.Modules.Find(smFindTag, "ma30")
1496     Set mau30u = m.Modules(mau30ui)
1497     mau30uv = mau30u.Data("Units")
1498     Maintenance.mau30.value = mau30uv
1499     Maintenance.mau30.AddItem "Seconds", 0
1500     Maintenance.mau30.AddItem "Minutes", 1
1501     Maintenance.mau30.AddItem "Hours", 2
1502     Maintenance.mau30.AddItem "Days", 3

1503     Dim mad31 As Module
1504     Dim mad31i As Long
1505     Dim mad31v As String
```

```
1506     mad31i = m.Modules.Find(smFindTag, "ma31")
1507     Set mad31 = m.Modules(mad31i)
1508     mad31v = mad31.Data("Expression")
1509     Maintenance.mad31.value = mad31v
1510     Maintenance.mad31.AddItem "TRIA ( 0, 120, 180 )", 0
1511     Maintenance.mad31.AddItem "TRIA ( Min, Mode, Max )", 1
1512     Maintenance.mad31.AddItem "NORM ( Mean, StdDev )", 2
1513     Maintenance.mad31.AddItem "EXPO ( Mean )", 3
1514     Maintenance.mad31.AddItem "UNIF ( Min, Max )", 4
1515     Dim mau31u As Module
1516     Dim mau31ui As Long
1517     Dim mau31uv As String
1518     mau31ui = m.Modules.Find(smFindTag, "ma31")
1519     Set mau31u = m.Modules(mau31ui)
1520     mau31uv = mau31u.Data("Units")
1521     Maintenance.mau31.value = mau31uv
1522     Maintenance.mau31.AddItem "Seconds", 0
1523     Maintenance.mau31.AddItem "Minutes", 1
1524     Maintenance.mau31.AddItem "Hours", 2
1525     Maintenance.mau31.AddItem "Days", 3

1526     Dim mad32 As Module
1527     Dim mad32i As Long
1528     Dim mad32v As String
1529     mad32i = m.Modules.Find(smFindTag, "ma32")
1530     Set mad32 = m.Modules(mad32i)
1531     mad32v = mad32.Data("Expression")
1532     Maintenance.mad32.value = mad32v
1533     Maintenance.mad32.AddItem "TRIA ( 0, 120, 180 )", 0
1534     Maintenance.mad32.AddItem "TRIA ( Min, Mode, Max )", 1
1535     Maintenance.mad32.AddItem "NORM ( Mean, StdDev )", 2
1536     Maintenance.mad32.AddItem "EXPO ( Mean )", 3
1537     Maintenance.mad32.AddItem "UNIF ( Min, Max )", 4
1538     Dim mau32u As Module
1539     Dim mau32ui As Long
1540     Dim mau32uv As String
1541     mau32ui = m.Modules.Find(smFindTag, "ma32")
1542     Set mau32u = m.Modules(mau32ui)
1543     mau32uv = mau32u.Data("Units")
1544     Maintenance.mau32.value = mau32uv
1545     Maintenance.mau32.AddItem "Seconds", 0
1546     Maintenance.mau32.AddItem "Minutes", 1
1547     Maintenance.mau32.AddItem "Hours", 2
1548     Maintenance.mau32.AddItem "Days", 3

1549     Dim mad33 As Module
1550     Dim mad33i As Long
1551     Dim mad33v As String
1552     mad33i = m.Modules.Find(smFindTag, "ma33")
1553     Set mad33 = m.Modules(mad33i)
1554     mad33v = mad33.Data("Expression")
1555     Maintenance.mad33.value = mad33v
1556     Maintenance.mad33.AddItem "TRIA ( 0, 120,180 )", 0
1557     Maintenance.mad33.AddItem "TRIA ( Min, Mode, Max )", 1
1558     Maintenance.mad33.AddItem "NORM ( Mean, StdDev )", 2
1559     Maintenance.mad33.AddItem "EXPO ( Mean )", 3
1560     Maintenance.mad33.AddItem "UNIF ( Min, Max )", 4
1561     Dim mau33u As Module
1562     Dim mau33ui As Long
1563     Dim mau33uv As String
1564     mau33ui = m.Modules.Find(smFindTag, "ma33")
1565     Set mau33u = m.Modules(mau33ui)
1566     mau33uv = mau33u.Data("Units")
1567     Maintenance.mau33.value = mau33uv
1568     Maintenance.mau33.AddItem "Seconds", 0
1569     Maintenance.mau33.AddItem "Minutes", 1
1570     Maintenance.mau33.AddItem "Hours", 2
1571     Maintenance.mau33.AddItem "Days", 3
```

```
1572     Dim mad34 As Module
1573     Dim mad34i As Long
1574     Dim mad34v As String
1575     mad34i = m.Modules.Find(smFindTag, "ma34")
1576     Set mad34 = m.Modules(mad34i)
1577     mad34v = mad34.Data("Expression")
1578     Maintenance.mad34.value = mad34v
1579     Maintenance.mad34.AddItem "TRIA ( 0, 240,600 )", 0
1580     Maintenance.mad34.AddItem "TRIA ( Min, Mode, Max )", 1
1581     Maintenance.mad34.AddItem "NORM ( Mean, StdDev )", 2
1582     Maintenance.mad34.AddItem "EXPO ( Mean )", 3
1583     Maintenance.mad34.AddItem "UNIF ( Min, Max )", 4
1584     Dim mau34u As Module
1585     Dim mau34ui As Long
1586     Dim mau34uv As String
1587     mau34ui = m.Modules.Find(smFindTag, "ma34")
1588     Set mau34u = m.Modules(mau34ui)
1589     mau34uv = mau34u.Data("Units")
1590     Maintenance.mau34.value = mau34uv
1591     Maintenance.mau34.AddItem "Seconds", 0
1592     Maintenance.mau34.AddItem "Minutes", 1
1593     Maintenance.mau34.AddItem "Hours", 2
1594     Maintenance.mau34.AddItem "Days", 3

1595     Dim mad35 As Module
1596     Dim mad35i As Long
1597     Dim mad35v As String
1598     mad35i = m.Modules.Find(smFindTag, "ma35")
1599     Set mad35 = m.Modules(mad35i)
1600     mad35v = mad35.Data("Expression")
1601     Maintenance.mad35.value = mad35v
1602     Maintenance.mad35.AddItem "TRIA ( 81, 90, 126 )", 0
1603     Maintenance.mad35.AddItem "TRIA ( Min, Mode, Max )", 1
1604     Maintenance.mad35.AddItem "NORM ( Mean, StdDev )", 2
1605     Maintenance.mad35.AddItem "EXPO ( Mean )", 3
1606     Maintenance.mad35.AddItem "UNIF ( Min, Max )", 4
1607     Dim mau35u As Module
1608     Dim mau35ui As Long
1609     Dim mau35uv As String
1610     mau35ui = m.Modules.Find(smFindTag, "ma35")
1611     Set mau35u = m.Modules(mau35ui)
1612     mau35uv = mau35u.Data("Units")
1613     Maintenance.mau35.value = mau35uv
1614     Maintenance.mau35.AddItem "Seconds", 0
1615     Maintenance.mau35.AddItem "Minutes", 1
1616     Maintenance.mau35.AddItem "Hours", 2
1617     Maintenance.mau35.AddItem "Days", 3

1618     Dim mad36 As Module
1619     Dim mad36i As Long
1620     Dim mad36v As String
1621     mad36i = m.Modules.Find(smFindTag, "ma36")
1622     Set mad36 = m.Modules(mad36i)
1623     mad36v = mad36.Data("Expression")
1624     Maintenance.mad36.value = mad36v
1625     Maintenance.mad36.AddItem "TRIA ( 162, 180, 252 )", 0
1626     Maintenance.mad36.AddItem "TRIA ( Min, Mode, Max )", 1
1627     Maintenance.mad36.AddItem "NORM ( Mean, StdDev )", 2
1628     Maintenance.mad36.AddItem "EXPO ( Mean )", 3
1629     Maintenance.mad36.AddItem "UNIF ( Min, Max )", 4
1630     Dim mau36u As Module
1631     Dim mau36ui As Long
1632     Dim mau36uv As String
1633     mau36ui = m.Modules.Find(smFindTag, "ma36")
1634     Set mau36u = m.Modules(mau36ui)
1635     mau36uv = mau36u.Data("Units")
1636     Maintenance.mau36.value = mau36uv
1637     Maintenance.mau36.AddItem "Seconds", 0
1638     Maintenance.mau36.AddItem "Minutes", 1
```

```
1639     Maintenance.mau36.AddItem "Hours", 2
1640     Maintenance.mau36.AddItem "Days", 3

1641     Dim mad37 As Module
1642     Dim mad37i As Long
1643     Dim mad37v As String
1644     mad37i = m.Modules.Find(smFindTag, "ma37")
1645     Set mad37 = m.Modules(mad37i)
1646     mad37v = mad37.Data("Expression")
1647     Maintenance.mad37.value = mad37v
1648     Maintenance.mad37.AddItem "TRIA ( 81, 90, 126 )", 0
1649     Maintenance.mad37.AddItem "TRIA ( Min, Mode, Max )", 1
1650     Maintenance.mad37.AddItem "NORM ( Mean, StdDev )", 2
1651     Maintenance.mad37.AddItem "EXPO ( Mean )", 3
1652     Maintenance.mad37.AddItem "UNIF ( Min, Max )", 4
1653     Dim mau37u As Module
1654     Dim mau37ui As Long
1655     Dim mau37uv As String
1656     mau37ui = m.Modules.Find(smFindTag, "ma37")
1657     Set mau37u = m.Modules(mau37ui)
1658     mau37uv = mau37u.Data("Units")
1659     Maintenance.mau37.value = mau37uv
1660     Maintenance.mau37.AddItem "Seconds", 0
1661     Maintenance.mau37.AddItem "Minutes", 1
1662     Maintenance.mau37.AddItem "Hours", 2
1663     Maintenance.mau37.AddItem "Days", 3

1667     Dim map38 As Module
1668     Dim map38i As Long
1669     Dim map38v As String
1670     map38i = m.Modules.Find(smFindTag, "ma38")
1671     Set map38 = m.Modules(map38i)
1672     'map38v = map38.Data("Percent True (0-100)")
1673     'Maintenance.map38.value = map38v

1677     Maintenance.map38.AddItem countnumber1 = countnumber1
1678     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1679     Maintenance.map38.AddItem countnumber1, countnumber1
1680     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1681     Maintenance.map38.AddItem countnumber1, countnumber1
1682     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1683     Maintenance.map38.AddItem countnumber1, countnumber1
1684     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1685     Maintenance.map38.AddItem countnumber1, countnumber1
1686     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1687     Maintenance.map38.AddItem countnumber1, countnumber1
1688     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1689     Maintenance.map38.AddItem countnumber1, countnumber1
1690     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1691     Maintenance.map38.AddItem countnumber1, countnumber1
1692     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1693     Maintenance.map38.AddItem countnumber1, countnumber1
1694     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1695     Maintenance.map38.AddItem countnumber1, countnumber1
1696     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1697     Maintenance.map38.AddItem countnumber1, countnumber1
1698     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1699     Maintenance.map38.AddItem countnumber1, countnumber1
1700     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1701     Maintenance.map38.AddItem countnumber1, countnumber1
1702     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1703     Maintenance.map38.AddItem countnumber1, countnumber1
1704     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1705     Maintenance.map38.AddItem countnumber1, countnumber1
1706     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1707     Maintenance.map38.AddItem countnumber1, countnumber1
1708     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1709     Maintenance.map38.AddItem countnumber1, countnumber1
1710     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
```

```
1711     Maintenance.map38.AddItem countnumber1, countnumber1
1712     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1713     Maintenance.map38.AddItem countnumber1, countnumber1
1714     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1715     Maintenance.map38.AddItem countnumber1, countnumber1
1716     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1717     Maintenance.map38.AddItem countnumber1, countnumber1
1718     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1719     Maintenance.map38.AddItem countnumber1, countnumber1
1720     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1721     Maintenance.map38.AddItem countnumber1, countnumber1
1722     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1723     Maintenance.map38.AddItem countnumber1, countnumber1
1724     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1725     Maintenance.map38.AddItem countnumber1, countnumber1
1726     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1727     Maintenance.map38.AddItem countnumber1, countnumber1
1728     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1729     Maintenance.map38.AddItem countnumber1, countnumber1
1730     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1731     Maintenance.map38.AddItem countnumber1, countnumber1
1732     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1733     Maintenance.map38.AddItem countnumber1, countnumber1
1734     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1735     Maintenance.map38.AddItem countnumber1, countnumber1
1736     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1737     Maintenance.map38.AddItem countnumber1, countnumber1
1738     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1739     Maintenance.map38.AddItem countnumber1, countnumber1
1740     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1741     Maintenance.map38.AddItem countnumber1, countnumber1
1742     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1743     Maintenance.map38.AddItem countnumber1, countnumber1
1744     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1745     Maintenance.map38.AddItem countnumber1, countnumber1
1746     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1747     Maintenance.map38.AddItem countnumber1, countnumber1
1748     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1749     Maintenance.map38.AddItem countnumber1, countnumber1
1750     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1751     Maintenance.map38.AddItem countnumber1, countnumber1
1752     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1753     Maintenance.map38.AddItem countnumber1, countnumber1
1754     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1755     Maintenance.map38.AddItem countnumber1, countnumber1
1756     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1757     Maintenance.map38.AddItem countnumber1, countnumber1
1758     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1759     Maintenance.map38.AddItem countnumber1, countnumber1
1760     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1761     Maintenance.map38.AddItem countnumber1, countnumber1
1762     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1763     Maintenance.map38.AddItem countnumber1, countnumber1
1764     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1765     Maintenance.map38.AddItem countnumber1, countnumber1
1766     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1767     Maintenance.map38.AddItem countnumber1, countnumber1
1768     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1769     Maintenance.map38.AddItem countnumber1, countnumber1
1770     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1771     Maintenance.map38.AddItem countnumber1, countnumber1
1772     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1773     Maintenance.map38.AddItem countnumber1, countnumber1
1774     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1775     Maintenance.map38.AddItem countnumber1, countnumber1
1776     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1777     Maintenance.map38.AddItem countnumber1, countnumber1
1778     If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1779     Maintenance.map38.AddItem countnumber1, countnumber1
```

```
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
Maintenance.map38.AddItem countnumber1, countnumber1
If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
```

```
1849      Maintenance.map38.AddItem countnumber1, countnumber1
1850      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1851      Maintenance.map38.AddItem countnumber1, countnumber1
1852      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1853      Maintenance.map38.AddItem countnumber1, countnumber1
1854      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1855      Maintenance.map38.AddItem countnumber1, countnumber1
1856      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1857      Maintenance.map38.AddItem countnumber1, countnumber1
1858      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1859      Maintenance.map38.AddItem countnumber1, countnumber1
1860      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1861      Maintenance.map38.AddItem countnumber1, countnumber1
1862      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1863      Maintenance.map38.AddItem countnumber1, countnumber1
1864      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1865      Maintenance.map38.AddItem countnumber1, countnumber1
1866      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1867      Maintenance.map38.AddItem countnumber1, countnumber1
1868      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1869      Maintenance.map38.AddItem countnumber1, countnumber1
1870      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1871      Maintenance.map38.AddItem countnumber1, countnumber1
1872      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1873      Maintenance.map38.AddItem countnumber1, countnumber1
1874      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1875      Maintenance.map38.AddItem countnumber1, countnumber1
1876      If countnumber1 < 100 Then countnumber1 = countnumber1 + 1
1877      Maintenance.map38.AddItem countnumber1, countnumber1

1878      Dim mad39 As Module
1879      Dim mad39i As Long
1880      Dim mad39v As String
1881      mad39i = m.Modules.Find(smFindTag, "ma39")
1882      Set mad39 = m.Modules(mad39i)
1883      mad39v = mad39.Data("Expression")
1884      Maintenance.mad39.value = mad39v
1885      Maintenance.mad39.AddItem "TRIA ( 27, 30, 42 )", 0
1886      Maintenance.mad39.AddItem "TRIA ( Min, Mode, Max )", 1
1887      Maintenance.mad39.AddItem "NORM ( Mean, StdDev )", 2
1888      Maintenance.mad39.AddItem "EXPO ( Mean )", 3
1889      Maintenance.mad39.AddItem "UNIF ( Min, Max )", 4
1890      Dim mau39u As Module
1891      Dim mau39ui As Long
1892      Dim mau39uv As String
1893      mau39ui = m.Modules.Find(smFindTag, "ma39")
1894      Set mau39u = m.Modules(mau39ui)
1895      mau39uv = mau39u.Data("Units")
1896      Maintenance.mau39.value = mau39uv
1897      Maintenance.mau39.AddItem "Seconds", 0
1898      Maintenance.mau39.AddItem "Minutes", 1
1899      Maintenance.mau39.AddItem "Hours", 2
1900      Maintenance.mau39.AddItem "Days", 3

1901      'End of code for this question

1902    End Sub
```

Project/Motors

```
1   Private Sub ComboBox2_Change()

2   End Sub

3   Private Sub ComboBox31_Change()

4   End Sub
```

```
5   Private Sub CommandButton2_Click()
6     Me.Hide

7     Maintenance.Show
8   End Sub

9   Private Sub CommandButton3_Click()

10    'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
11    Dim msgResult As Integer
12    If (Modular.value = False And NotModular.value = False) Then
13      msgResult = MsgBox("Is the motor modular?", vbYesNo)
14      If msgResult = vbYes Then
15        Modular.value = True
16      Else
17        NotModular.value = True
18      End If
19    End If

20    Dim m As Model
21    Set m = ThisDocument.Model

22    Dim mon02 As Module
23    Dim mon02i As Long
24    mon02i = m.Modules.Find(smFindTag, "mo02")
25    Set mon02 = m.Modules(mon02i)
26    mon02.Data("Value") = mot02.Text

27    Dim mo03 As Module
28    Dim mo03i As Long
29    mo03i = m.Modules.Find(smFindTag, "mo03")
30    Set mo03 = m.Modules(mo03i)
31    mo03.Data("Expression") = mod03.Text
32    mo03.Data("Units") = mou03.Text

33    Dim mo04 As Module
34    Dim mo04i As Long
35    mo04i = m.Modules.Find(smFindTag, "mo04")
36    Set mo04 = m.Modules(mo04i)
37    mo04.Data("Expression") = mod04.Text
38    mo04.Data("Units") = mou04.Text

39    Dim mo05 As Module
40    Dim mo05i As Long
41    mo05i = m.Modules.Find(smFindTag, "mo05")
42    Set mo05 = m.Modules(mo05i)
43    mo05.Data("Expression") = mod05.Text
44    mo05.Data("Units") = mou05.Text

45    Dim mo06 As Module
46    Dim mo06i As Long
47    mo06i = m.Modules.Find(smFindTag, "mo06")
48    Set mo06 = m.Modules(mo06i)
49    mo06.Data("Expression") = mod06.Text
50    mo06.Data("Units") = mou06.Text

54    Dim mo07 As Module
55    Dim mo07i As Long
56    mo07i = m.Modules.Find(smFindTag, "mo07")
57    Set mo07 = m.Modules(mo07i)
58    mo07.Data("Percent True") = mop07.Text

59    Dim mo07b As Module
60    Dim mo07bi As Long
61    mo07bi = m.Modules.Find(smFindTag, "mo07b")
62    Set mo07b = m.Modules(mo07bi)
```

```
63      mo07b.Data("Expression") = mod07b.Text
64      mo07b.Data("Units") = mou07b.Text

68      Dim mo08 As Module
69      Dim mo08i As Long
70      mo08i = m.Modules.Find(smFindTag, "mo08")
71      Set mo08 = m.Modules(mo08i)
72      mo08.Data("Expression") = mod08.Text
73      mo08.Data("Units") = mou08.Text

74      Dim mo09 As Module
75      Dim mo09i As Long
76      mo09i = m.Modules.Find(smFindTag, "mo09")
77      Set mo09 = m.Modules(mo09i)
78      mo09.Data("Expression") = mod09.Text
79      mo09.Data("Units") = mou09.Text

80      Dim mo10 As Module
81      Dim mo10i As Long
82      mo10i = m.Modules.Find(smFindTag, "mo10")
83      Set mo10 = m.Modules(mo10i)
84      mo10.Data("Expression") = mod10.Text
85      mo10.Data("Units") = mou10.Text

86      Dim mo11 As Module
87      Dim mo11i As Long
88      mo11i = m.Modules.Find(smFindTag, "mo11")
89      Set mo11 = m.Modules(mo11i)
90      mo11.Data("Expression") = mod11.Text
91      mo11.Data("Units") = mou11.Text

92      Dim mo12 As Module
93      Dim mo12i As Long
94      mo12i = m.Modules.Find(smFindTag, "mo12")
95      Set mo12 = m.Modules(mo12i)
96      mo12.Data("Expression") = mod12.Text
97      mo12.Data("Units") = mou12.Text

98      Dim mo13 As Module
99      Dim mo13i As Long
100     mo13i = m.Modules.Find(smFindTag, "mo13")
101     Set mo13 = m.Modules(mo13i)
102     mo13.Data("Expression") = mod13.Text
103     mo13.Data("Units") = mou13.Text

104     Dim mo14 As Module
105     Dim mo14i As Long
106     mo14i = m.Modules.Find(smFindTag, "mo14")
107     Set mo14 = m.Modules(mo14i)
108     mo14.Data("Expression") = mod14.Text
109     mo14.Data("Units") = mou14.Text

110     Dim mo15 As Module
111     Dim mo15i As Long
112     mo15i = m.Modules.Find(smFindTag, "mo15")
113     Set mo15 = m.Modules(mo15i)
114     mo15.Data("Expression") = mod15.Text
115     mo15.Data("Units") = mou15.Text

116     Dim mo16 As Module
117     Dim mo16i As Long
118     mo16i = m.Modules.Find(smFindTag, "mo16")
119     Set mo16 = m.Modules(mo16i)
120     mo16.Data("Expression") = mod16.Text
121     mo16.Data("Units") = mou16.Text

122     Dim mo17 As Module
123     Dim mo17i As Long
124     mo17i = m.Modules.Find(smFindTag, "mo17")
```

```
125     Set mo17 = m.Modules(mo17i)
126     mo17.Data("Expression") = mod17.Text
127     mo17.Data("Units") = mou17.Text

128     Dim mo18 As Module
129     Dim mo18i As Long
130     mo18i = m.Modules.Find(smFindTag, "mo18")
131     Set mo18 = m.Modules(mo18i)
132     mo18.Data("Expression") = mod18.Text
133     mo18.Data("Units") = mou18.Text

134     Dim modular1 As Module
135     Dim modular1i As Long
136     modular1i = m.Modules.Find(smFindTag, "modular1")
137     Set modular1 = m.Modules(modular1i)
138     If Modular.value = True Then
139       modular1.Data("Initial Value") = "1"
140     Else
141       modular1.Data("Initial Value") = "0"
142     End If

143     Hierarchy.done02.Visible = True
144     Me.Hide
145     po1prelim.Show

146   End Sub

147   Private Sub CommandButton4_Click()
148     Me.Hide

149     Mohelp.Show

150   End Sub

151   Private Sub CommandButton5_Click()

152     'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
153     Dim msgResult As Integer
154     If (Modular.value = False And NotModular.value = False) Then
155       msgResult = MsgBox("Is the motor modular?", vbYesNo)
156       If msgResult = vbYes Then
157         Modular.value = True
158       Else
159         NotModular.value = True
160       End If
161     End If

162     Dim m As Model
163     Set m = ThisDocument.Model

164     Dim mon02 As Module
165     Dim mon02i As Long
166     mon02i = m.Modules.Find(smFindTag, "mo02")
167     Set mon02 = m.Modules(mon02i)
168     mon02.Data("Value") = mot02.Text

169     Dim mo03 As Module
170     Dim mo03i As Long
171     mo03i = m.Modules.Find(smFindTag, "mo03")
172     Set mo03 = m.Modules(mo03i)
173     mo03.Data("Expression") = mod03.Text
174     mo03.Data("Units") = mou03.Text

175     Dim mo04 As Module
176     Dim mo04i As Long
177     mo04i = m.Modules.Find(smFindTag, "mo04")
178     Set mo04 = m.Modules(mo04i)
```

```
179     mo04.Data("Expression") = mod04.Text
180     mo04.Data("Units") = mou04.Text

181     Dim mo05 As Module
182     Dim mo05i As Long
183     mo05i = m.Modules.Find(smFindTag, "mo05")
184     Set mo05 = m.Modules(mo05i)
185     mo05.Data("Expression") = mod05.Text
186     mo05.Data("Units") = mou05.Text

187     Dim mo06 As Module
188     Dim mo06i As Long
189     mo06i = m.Modules.Find(smFindTag, "mo06")
190     Set mo06 = m.Modules(mo06i)
191     mo06.Data("Expression") = mod06.Text
192     mo06.Data("Units") = mou06.Text

193     Dim mo07 As Module
194     Dim mo07i As Long
195     mo07i = m.Modules.Find(smFindTag, "mo07")
196     Set mo07 = m.Modules(mo07i)
197     mo07.Data("Percent True") = mop07.Text

198     Dim mo07b As Module
199     Dim mo07bi As Long
200     mo07bi = m.Modules.Find(smFindTag, "mo07b")
201     Set mo07b = m.Modules(mo07bi)
202     mo07b.Data("Expression") = mod07b.Text
203     mo07b.Data("Units") = mou07b.Text

204     Dim mo08 As Module
205     Dim mo08i As Long
206     mo08i = m.Modules.Find(smFindTag, "mo08")
207     Set mo08 = m.Modules(mo08i)
208     mo08.Data("Expression") = mod08.Text
209     mo08.Data("Units") = mou08.Text

210     Dim mo09 As Module
211     Dim mo09i As Long
212     mo09i = m.Modules.Find(smFindTag, "mo09")
213     Set mo09 = m.Modules(mo09i)
214     mo09.Data("Expression") = mod09.Text
215     mo09.Data("Units") = mou09.Text

216     Dim mo10 As Module
217     Dim mo10i As Long
218     mo10i = m.Modules.Find(smFindTag, "mo10")
219     Set mo10 = m.Modules(mo10i)
220     mo10.Data("Expression") = mod10.Text
221     mo10.Data("Units") = mou10.Text

222     Dim mo11 As Module
223     Dim mo11i As Long
224     mo11i = m.Modules.Find(smFindTag, "mo11")
225     Set mo11 = m.Modules(mo11i)
226     mo11.Data("Expression") = mod11.Text
227     mo11.Data("Units") = mou11.Text

228     Dim mo12 As Module
229     Dim mo12i As Long
230     mo12i = m.Modules.Find(smFindTag, "mo12")
231     Set mo12 = m.Modules(mo12i)
232     mo12.Data("Expression") = mod12.Text
233     mo12.Data("Units") = mou12.Text

234     Dim mo13 As Module
235     Dim mo13i As Long
236     mo13i = m.Modules.Find(smFindTag, "mo13")
237     Set mo13 = m.Modules(mo13i)
```

```
238     mo13.Data("Expression") = mod13.Text
239     mo13.Data("Units") = mou13.Text

240     Dim mo14 As Module
241     Dim mo14i As Long
242     mo14i = m.Modules.Find(smFindTag, "mo14")
243     Set mo14 = m.Modules(mo14i)
244     mo14.Data("Expression") = mod14.Text
245     mo14.Data("Units") = mou14.Text

246     Dim mo15 As Module
247     Dim mo15i As Long
248     mo15i = m.Modules.Find(smFindTag, "mo15")
249     Set mo15 = m.Modules(mo15i)
250     mo15.Data("Expression") = mod15.Text
251     mo15.Data("Units") = mou15.Text

252     Dim mo16 As Module
253     Dim mo16i As Long
254     mo16i = m.Modules.Find(smFindTag, "mo16")
255     Set mo16 = m.Modules(mo16i)
256     mo16.Data("Expression") = mod16.Text
257     mo16.Data("Units") = mou16.Text

258     Dim mo17 As Module
259     Dim mo17i As Long
260     mo17i = m.Modules.Find(smFindTag, "mo17")
261     Set mo17 = m.Modules(mo17i)
262     mo17.Data("Expression") = mod17.Text
263     mo17.Data("Units") = mou17.Text

264     Dim mo18 As Module
265     Dim mo18i As Long
266     mo18i = m.Modules.Find(smFindTag, "mo18")
267     Set mo18 = m.Modules(mo18i)
268     mo18.Data("Expression") = mod18.Text
269     mo18.Data("Units") = mou18.Text

270     Dim modular1 As Module
271     Dim modular1i As Long
272     modular1i = m.Modules.Find(smFindTag, "modular1")
273     Set modular1 = m.Modules(modular1i)
274     If Modular.value = True Then
275       modular1.Data("Initial Value") = "1"
276     Else
277       modular1.Data("Initial Value") = "0"
278     End If

279     Hierarchy.done02.Visible = True
280     Me.Hide
281     Hierarchy.Show

282   End Sub

283   Private Sub Frame2_Click()

284   End Sub

285   Private Sub Label11_Click()

286   End Sub

287   Private Sub Label17_Click()

288   End Sub

289   Private Sub Label22_Click()

290   End Sub
```

```
291   Private Sub Label23_Click()
292   End Sub
293   Private Sub Label24_Click()
294   End Sub
295   Private Sub mo01_Change()
296   End Sub
297   Private Sub mo01_Enter()
298   End Sub
299   Private Sub mo07_Change()
300   End Sub
301   Private Sub mod03_Change()
302   End Sub
303   Private Sub mod06_Change()
304   End Sub
305   Private Sub Mod07a_Change()
306   End Sub
307   Private Sub mod08_Change()
308   End Sub
309   Private Sub mod10_Change()
310   End Sub
311   Private Sub mod11_Change()
312   End Sub
313   Private Sub mod13_Change()
314   End Sub
315   Private Sub mod15_Change()
316   End Sub
317   Private Sub mod18_Change()
318   End Sub
319   Private Sub Modular_Click()
320     Frame1.Visible = False
321     Frame2.Visible = True
322     Frame2.Top = 100
323     Frame2.Left = 100
324   End Sub
325   Private Sub mon02_Change()
326   End Sub
```

```
327   Private Sub mop07_Change()

328   End Sub

329   Private Sub mou09_Change()

330   End Sub

331   Private Sub NotModular_Click()
332     Frame1.Visible = True
333     Frame2.Visible = False
334     Frame1.Top = 100
335     Frame1.Left = 100
336   End Sub

337   Private Sub OptionButton1_Click()

338   End Sub

339   Private Sub ToggleButton1_Click()

340   End Sub

341   Private Sub UserForm_Click()

342   End Sub

343   Private Sub UserForm_Initialize()

 350     Dim m As Model
351     Set m = ThisDocument.Model

352     Dim mon02 As Module
353     Dim mon02i As Long
354     Dim mon02v As String
355     mon02i = m.Modules.Find(smFindTag, "mo02")
356     Set mon02 = m.Modules(mon02i)
357     mon02v = mon02.Data("Value")
358     motors.mot02.value = mon02v
359     motors.mot02.AddItem "1", 0
360     motors.mot02.AddItem "2", 1
361     motors.mot02.AddItem "3", 2
362     motors.mot02.AddItem "4", 3
363     motors.mot02.AddItem "5", 4
364     motors.mot02.AddItem "6", 5

369     Dim mod03 As Module
370     Dim mod03i As Long
371     Dim mod03v As String
372     mod03i = m.Modules.Find(smFindTag, "mo03")
373     Set mod03 = m.Modules(mod03i)
374     mod03v = mod03.Data("Expression")
375     motors.mod03.value = mod03v
376     motors.mod03.AddItem "TRIA ( 108, 120, 168 )", 0
377     motors.mod03.AddItem "TRIA ( Min, Mode, mox )", 1
378     motors.mod03.AddItem "NORM ( Mean, StdDev )", 2
379     motors.mod03.AddItem "EXPO ( Mean )", 3
380     motors.mod03.AddItem "UNIF ( Min, mox )", 4
381     Dim mou03u As Module
382     Dim mou03ui As Long
383     Dim mou03uv As String
384     mou03ui = m.Modules.Find(smFindTag, "mo03")
385     Set mou03u = m.Modules(mou03ui)
386     mou03uv = mou03u.Data("Units")
387     motors.mou03.value = mou03uv
388     motors.mou03.AddItem "Seconds", 0
389     motors.mou03.AddItem "Minutes", 1
390     motors.mou03.AddItem "Hours", 2
391     motors.mou03.AddItem "Days", 3
```

```
392     Dim mod04 As Module
393     Dim mod04i As Long
394     Dim mod04v As String
395     mod04i = m.Modules.Find(smFindTag, "mo04")
396     Set mod04 = m.Modules(mod04i)
397     mod04v = mod04.Data("Expression")
398     motors.mod04.value = mod04v
399     motors.mod04.AddItem "TRIA ( 54, 60, 84 )", 0
400     motors.mod04.AddItem "TRIA ( Min, Mode, mox )", 1
401     motors.mod04.AddItem "NORM ( Mean, StdDev )", 2
402     motors.mod04.AddItem "EXPO ( Mean )", 3
403     motors.mod04.AddItem "UNIF ( Min, mox )", 4
404     Dim mou04u As Module
405     Dim mou04ui As Long
406     Dim mou04uv As String
407     mou04ui = m.Modules.Find(smFindTag, "mo04")
408     Set mou04u = m.Modules(mou04ui)
409     mou04uv = mou04u.Data("Units")
410     motors.mou04.value = mou04uv
411     motors.mou04.AddItem "Seconds", 0
412     motors.mou04.AddItem "Minutes", 1
413     motors.mou04.AddItem "Hours", 2
414     motors.mou04.AddItem "Days", 3

415     Dim mod05 As Module
416     Dim mod05i As Long
417     Dim mod05v As String
418     mod05i = m.Modules.Find(smFindTag, "mo05")
419     Set mod05 = m.Modules(mod05i)
420     mod05v = mod05.Data("Expression")
421     motors.mod05.value = mod05v
422     motors.mod05.AddItem "TRIA ( 54, 60, 84 )", 0
423     motors.mod05.AddItem "TRIA ( Min, Mode, mox )", 1
424     motors.mod05.AddItem "NORM ( Mean, StdDev )", 2
425     motors.mod05.AddItem "EXPO ( Mean )", 3
426     motors.mod05.AddItem "UNIF ( Min, mox )", 4
427     Dim mou05u As Module
428     Dim mou05ui As Long
429     Dim mou05uv As String
430     mou05ui = m.Modules.Find(smFindTag, "mo05")
431     Set mou05u = m.Modules(mou05ui)
432     mou05uv = mou05u.Data("Units")
433     motors.mou05.value = mou05uv
434     motors.mou05.AddItem "Seconds", 0
435     motors.mou05.AddItem "Minutes", 1
436     motors.mou05.AddItem "Hours", 2
437     motors.mou05.AddItem "Days", 3

438     Dim mod06 As Module
439     Dim mod06i As Long
440     Dim mod06v As String
441     mod06i = m.Modules.Find(smFindTag, "mo06")
442     Set mod06 = m.Modules(mod06i)
443     mod06v = mod06.Data("Expression")
444     motors.mod06.value = mod06v
445     motors.mod06.AddItem "TRIA ( 108, 120, 168 )", 0
446     motors.mod06.AddItem "TRIA ( Min, Mode, mox )", 1
447     motors.mod06.AddItem "NORM ( Mean, StdDev )", 2
448     motors.mod06.AddItem "EXPO ( Mean )", 3
449     motors.mod06.AddItem "UNIF ( Min, mox )", 4
450     Dim mou06u As Module
451     Dim mou06ui As Long
452     Dim mou06uv As String
453     mou06ui = m.Modules.Find(smFindTag, "mo06")
454     Set mou06u = m.Modules(mou06ui)
455     mou06uv = mou06u.Data("Units")
456     motors.mou06.value = mou06uv
457     motors.mou06.AddItem "Seconds", 0
```

```
motors.mou06.AddItem "Minutes", 1
motors.mou06.AddItem "Hours", 2
motors.mou06.AddItem "Days", 3

Dim mo07 As Module
Dim mo07i As Long
Dim mo07v As String
mo07i = m.Modules.Find(smFindTag, "mo07")
Set mo07 = m.Modules(mo07i)
mo07v = mo07.Data("Percent True")
motors.mop07.value = mo07v
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
```

```
526     motors.mop07.AddItem countnumber, countnumber
527     If countnumber < 100 Then countnumber = countnumber + 1
528     motors.mop07.AddItem countnumber, countnumber
529     If countnumber < 100 Then countnumber = countnumber + 1
530     motors.mop07.AddItem countnumber, countnumber
531     If countnumber < 100 Then countnumber = countnumber + 1
532     motors.mop07.AddItem countnumber, countnumber
533     If countnumber < 100 Then countnumber = countnumber + 1
534     motors.mop07.AddItem countnumber, countnumber
535     If countnumber < 100 Then countnumber = countnumber + 1
536     motors.mop07.AddItem countnumber, countnumber
537     If countnumber < 100 Then countnumber = countnumber + 1
538     motors.mop07.AddItem countnumber, countnumber
539     If countnumber < 100 Then countnumber = countnumber + 1
540     motors.mop07.AddItem countnumber, countnumber
541     If countnumber < 100 Then countnumber = countnumber + 1
542     motors.mop07.AddItem countnumber, countnumber
543     If countnumber < 100 Then countnumber = countnumber + 1
544     motors.mop07.AddItem countnumber, countnumber
545     If countnumber < 100 Then countnumber = countnumber + 1
546     motors.mop07.AddItem countnumber, countnumber
547     If countnumber < 100 Then countnumber = countnumber + 1
548     motors.mop07.AddItem countnumber, countnumber
549     If countnumber < 100 Then countnumber = countnumber + 1
550     motors.mop07.AddItem countnumber, countnumber
551     If countnumber < 100 Then countnumber = countnumber + 1
552     motors.mop07.AddItem countnumber, countnumber
553     If countnumber < 100 Then countnumber = countnumber + 1
554     motors.mop07.AddItem countnumber, countnumber
555     If countnumber < 100 Then countnumber = countnumber + 1
556     motors.mop07.AddItem countnumber, countnumber
557     If countnumber < 100 Then countnumber = countnumber + 1
558     motors.mop07.AddItem countnumber, countnumber
559     If countnumber < 100 Then countnumber = countnumber + 1
560     motors.mop07.AddItem countnumber, countnumber
561     If countnumber < 100 Then countnumber = countnumber + 1
562     motors.mop07.AddItem countnumber, countnumber
563     If countnumber < 100 Then countnumber = countnumber + 1
564     motors.mop07.AddItem countnumber, countnumber
565     If countnumber < 100 Then countnumber = countnumber + 1
566     motors.mop07.AddItem countnumber, countnumber
567     If countnumber < 100 Then countnumber = countnumber + 1
568     motors.mop07.AddItem countnumber, countnumber
569     If countnumber < 100 Then countnumber = countnumber + 1
570     motors.mop07.AddItem countnumber, countnumber
571     If countnumber < 100 Then countnumber = countnumber + 1
572     motors.mop07.AddItem countnumber, countnumber
573     If countnumber < 100 Then countnumber = countnumber + 1
574     motors.mop07.AddItem countnumber, countnumber
575     If countnumber < 100 Then countnumber = countnumber + 1
576     motors.mop07.AddItem countnumber, countnumber
577     If countnumber < 100 Then countnumber = countnumber + 1
578     motors.mop07.AddItem countnumber, countnumber
579     If countnumber < 100 Then countnumber = countnumber + 1
580     motors.mop07.AddItem countnumber, countnumber
581     If countnumber < 100 Then countnumber = countnumber + 1
582     motors.mop07.AddItem countnumber, countnumber
583     If countnumber < 100 Then countnumber = countnumber + 1
584     motors.mop07.AddItem countnumber, countnumber
585     If countnumber < 100 Then countnumber = countnumber + 1
586     motors.mop07.AddItem countnumber, countnumber
587     If countnumber < 100 Then countnumber = countnumber + 1
588     motors.mop07.AddItem countnumber, countnumber
589     If countnumber < 100 Then countnumber = countnumber + 1
590     motors.mop07.AddItem countnumber, countnumber
591     If countnumber < 100 Then countnumber = countnumber + 1
592     motors.mop07.AddItem countnumber, countnumber
593     If countnumber < 100 Then countnumber = countnumber + 1
594     motors.mop07.AddItem countnumber, countnumber
```

```
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
motors.mop07.AddItem countnumber, countnumber
If countnumber < 100 Then countnumber = countnumber + 1
```

```
664     motors.mop07.AddItem countnumber, countnumber
665     If countnumber < 100 Then countnumber = countnumber + 1
666     motors.mop07.AddItem countnumber, countnumber
667     If countnumber < 100 Then countnumber = countnumber + 1
668     motors.mop07.AddItem countnumber, countnumber

669     Dim mod07b As Module
670     Dim mod07bi As Long
671     Dim mod07bv As String
672     mod07bi = m.Modules.Find(smFindTag, "mo07b")
673     Set mod07b = m.Modules(mod07bi)
674     mod07bv = mod07b.Data("Expression")
675     motors.mod07b.value = mod07bv
676     motors.mod07b.AddItem "TRIA ( 108, 120, 168 )", 0
677     motors.mod07b.AddItem "TRIA ( Min, Mode, mox )", 1
678     motors.mod07b.AddItem "NORM ( Mean, StdDev )", 2
679     motors.mod07b.AddItem "EXPO ( Mean )", 3
680     motors.mod07b.AddItem "UNIF ( Min, mox )", 4
681     Dim mou07bu As Module
682     Dim mou07bui As Long
683     Dim mou07buv As String
684     mou07bui = m.Modules.Find(smFindTag, "mo07b")
685     Set mou07bu = m.Modules(mou07bui)
686     mou07buv = mou07bu.Data("Units")
687     motors.mou07b.value = mou07buv
688     motors.mou07b.AddItem "Seconds", 0
689     motors.mou07b.AddItem "Minutes", 1
690     motors.mou07b.AddItem "Hours", 2
691     motors.mou07b.AddItem "Days", 3

692     Dim mod08 As Module
693     Dim mod08i As Long
694     Dim mod08v As String
695     mod08i = m.Modules.Find(smFindTag, "mo08")
696     Set mod08 = m.Modules(mod08i)
697     mod08v = mod08.Data("Expression")
698     motors.mod08.value = mod08v
699     motors.mod08.AddItem "TRIA ( 54, 60, 84 )", 0
700     motors.mod08.AddItem "TRIA ( Min, Mode, mox )", 1
701     motors.mod08.AddItem "NORM ( Mean, StdDev )", 2
702     motors.mod08.AddItem "EXPO ( Mean )", 3
703     motors.mod08.AddItem "UNIF ( Min, mox )", 4
704     Dim mou08u As Module
705     Dim mou08ui As Long
706     Dim mou08uv As String
707     mou08ui = m.Modules.Find(smFindTag, "mo08")
708     Set mou08u = m.Modules(mou08ui)
709     mou08uv = mou08u.Data("Units")
710     motors.mou08.value = mou08uv
711     motors.mou08.AddItem "Seconds", 0
712     motors.mou08.AddItem "Minutes", 1
713     motors.mou08.AddItem "Hours", 2
714     motors.mou08.AddItem "Days", 3

715     Dim mod09 As Module
716     Dim mod09i As Long
717     Dim mod09v As String
718     mod09i = m.Modules.Find(smFindTag, "mo09")
719     Set mod09 = m.Modules(mod09i)
720     mod09v = mod09.Data("Expression")
721     motors.mod09.value = mod09v
722     motors.mod09.AddItem "TRIA ( 54, 60, 84 )", 0
723     motors.mod09.AddItem "TRIA ( Min, Mode, mox )", 1
724     motors.mod09.AddItem "NORM ( Mean, StdDev )", 2
725     motors.mod09.AddItem "EXPO ( Mean )", 3
726     motors.mod09.AddItem "UNIF ( Min, mox )", 4
727     Dim mou09u As Module
728     Dim mou09ui As Long
729     Dim mou09uv As String
```

```
730     mou09ui = m.Modules.Find(smFindTag, "mo09")
731     Set mou09u = m.Modules(mou09ui)
732     mou09uv = mou09u.Data("Units")
733     motors.mou09.value = mou09uv
734     motors.mou09.AddItem "Seconds", 0
735     motors.mou09.AddItem "Minutes", 1
736     motors.mou09.AddItem "Hours", 2
737     motors.mou09.AddItem "Days", 3

738     Dim mod10 As Module
739     Dim mod10i As Long
740     Dim mod10v As String
741     mod10i = m.Modules.Find(smFindTag, "mo10")
742     Set mod10 = m.Modules(mod10i)
743     mod10v = mod10.Data("Expression")
744     motors.mod10.value = mod10v
745     motors.mod10.AddItem "TRIA ( 54, 60, 84 )", 0
746     motors.mod10.AddItem "TRIA ( Min, Mode, mox )", 1
747     motors.mod10.AddItem "NORM ( Mean, StdDev )", 2
748     motors.mod10.AddItem "EXPO ( Mean )", 3
749     motors.mod10.AddItem "UNIF ( Min, mox )", 4
750     Dim mou10u As Module
751     Dim mou10ui As Long
752     Dim mou10uv As String
753     mou10ui = m.Modules.Find(smFindTag, "mo10")
754     Set mou10u = m.Modules(mou10ui)
755     mou10uv = mou10u.Data("Units")
756     motors.mou10.value = mou10uv
757     motors.mou10.AddItem "Seconds", 0
758     motors.mou10.AddItem "Minutes", 1
759     motors.mou10.AddItem "Hours", 2
760     motors.mou10.AddItem "Days", 3

761     Dim mod11 As Module
762     Dim mod11i As Long
763     Dim mod11v As String
764     mod11i = m.Modules.Find(smFindTag, "mo11")
765     Set mod11 = m.Modules(mod11i)
766     mod11v = mod11.Data("Expression")
767     motors.mod11.value = mod11v
768     motors.mod11.AddItem "TRIA ( 108, 120, 168 )", 0
769     motors.mod11.AddItem "TRIA ( Min, Mode, mox )", 1
770     motors.mod11.AddItem "NORM ( Mean, StdDev )", 2
771     motors.mod11.AddItem "EXPO ( Mean )", 3
772     motors.mod11.AddItem "UNIF ( Min, mox )", 4
773     Dim mou11u As Module
774     Dim mou11ui As Long
775     Dim mou11uv As String
776     mou11ui = m.Modules.Find(smFindTag, "mo11")
777     Set mou11u = m.Modules(mou11ui)
778     mou11uv = mou11u.Data("Units")
779     motors.mou11.value = mou11uv
780     motors.mou11.AddItem "Seconds", 0
781     motors.mou11.AddItem "Minutes", 1
782     motors.mou11.AddItem "Hours", 2
783     motors.mou11.AddItem "Days", 3

784     Dim mod12 As Module
785     Dim mod12i As Long
786     Dim mod12v As String
787     mod12i = m.Modules.Find(smFindTag, "mo12")
788     Set mod12 = m.Modules(mod12i)
789     mod12v = mod12.Data("Expression")
790     motors.mod12.value = mod12v
791     motors.mod12.AddItem "TRIA ( 24, 30, 42 )", 0
792     motors.mod12.AddItem "TRIA ( Min, Mode, mox )", 1
793     motors.mod12.AddItem "NORM ( Mean, StdDev )", 2
794     motors.mod12.AddItem "EXPO ( Mean )", 3
795     motors.mod12.AddItem "UNIF ( Min, mox )", 4
```

```
    Dim mou12u As Module
    Dim mou12ui As Long
    Dim mou12uv As String
    mou12ui = m.Modules.Find(smFindTag, "mo12")
    Set mou12u = m.Modules(mou12ui)
    mou12uv = mou12u.Data("Units")
    motors.mou12.value = mou12uv
    motors.mou12.AddItem "Seconds", 0
    motors.mou12.AddItem "Minutes", 1
    motors.mou12.AddItem "Hours", 2
    motors.mou12.AddItem "Days", 3

    Dim mod13 As Module
    Dim mod13i As Long
    Dim mod13v As String
    mod13i = m.Modules.Find(smFindTag, "mo13")
    Set mod13 = m.Modules(mod13i)
    mod13v = mod13.Data("Expression")
    motors.mod13.value = mod13v
    motors.mod13.AddItem "TRIA ( 54, 60, 84 )", 0
    motors.mod13.AddItem "TRIA ( Min, Mode, mox )", 1
    motors.mod13.AddItem "NORM ( Mean, StdDev )", 2
    motors.mod13.AddItem "EXPO ( Mean )", 3
    motors.mod13.AddItem "UNIF ( Min, mox )", 4
    Dim mou13u As Module
    Dim mou13ui As Long
    Dim mou13uv As String
    mou13ui = m.Modules.Find(smFindTag, "mo13")
    Set mou13u = m.Modules(mou13ui)
    mou13uv = mou13u.Data("Units")
    motors.mou13.value = mou13uv
    motors.mou13.AddItem "Seconds", 0
    motors.mou13.AddItem "Minutes", 1
    motors.mou13.AddItem "Hours", 2
    motors.mou13.AddItem "Days", 3

    Dim mod14 As Module
    Dim mod14i As Long
    Dim mod14v As String
    mod14i = m.Modules.Find(smFindTag, "mo14")
    Set mod14 = m.Modules(mod14i)
    mod14v = mod14.Data("Expression")
    motors.mod14.value = mod14v
    motors.mod14.AddItem "TRIA ( 81, 90, 126 )", 0
    motors.mod14.AddItem "TRIA ( Min, Mode, mox )", 1
    motors.mod14.AddItem "NORM ( Mean, StdDev )", 2
    motors.mod14.AddItem "EXPO ( Mean )", 3
    motors.mod14.AddItem "UNIF ( Min, mox )", 4
    Dim mou14u As Module
    Dim mou14ui As Long
    Dim mou14uv As String
    mou14ui = m.Modules.Find(smFindTag, "mo14")
    Set mou14u = m.Modules(mou14ui)
    mou14uv = mou14u.Data("Units")
    motors.mou14.value = mou14uv
    motors.mou14.AddItem "Seconds", 0
    motors.mou14.AddItem "Minutes", 1
    motors.mou14.AddItem "Hours", 2
    motors.mou14.AddItem "Days", 3

    Dim mod15 As Module
    Dim mod15i As Long
    Dim mod15v As String
    mod15i = m.Modules.Find(smFindTag, "mo15")
    Set mod15 = m.Modules(mod15i)
    mod15v = mod15.Data("Expression")
    motors.mod15.value = mod15v
    motors.mod15.AddItem "TRIA ( 54, 60, 84 )", 0
    motors.mod15.AddItem "TRIA ( Min, Mode, mox )", 1
```

```
862     motors.mod15.AddItem "NORM ( Mean, StdDev )", 2
863     motors.mod15.AddItem "EXPO ( Mean )", 3
864     motors.mod15.AddItem "UNIF ( Min, mox )", 4
865     Dim mou15u As Module
866     Dim mou15ui As Long
867     Dim mou15uv As String
868     mou15ui = m.Modules.Find(smFindTag, "mo15")
869     Set mou15u = m.Modules(mou15ui)
870     mou15uv = mou15u.Data("Units")
871     motors.mou15.value = mou15uv
872     motors.mou15.AddItem "Seconds", 0
873     motors.mou15.AddItem "Minutes", 1
874     motors.mou15.AddItem "Hours", 2
875     motors.mou15.AddItem "Days", 3

876     Dim mod16 As Module
877     Dim mod16i As Long
878     Dim mod16v As String
879     mod16i = m.Modules.Find(smFindTag, "mo16")
880     Set mod16 = m.Modules(mod16i)
881     mod16v = mod16.Data("Expression")
882     motors.mod16.value = mod16v
883     motors.mod16.AddItem "TRIA ( 54, 60, 84 )", 0
884     motors.mod16.AddItem "TRIA ( Min, Mode, mox )", 1
885     motors.mod16.AddItem "NORM ( Mean, StdDev )", 2
886     motors.mod16.AddItem "EXPO ( Mean )", 3
887     motors.mod16.AddItem "UNIF ( Min, mox )", 4
888     Dim mou16u As Module
889     Dim mou16ui As Long
890     Dim mou16uv As String
891     mou16ui = m.Modules.Find(smFindTag, "mo16")
892     Set mou16u = m.Modules(mou16ui)
893     mou16uv = mou16u.Data("Units")
894     motors.mou16.value = mou16uv
895     motors.mou16.AddItem "Seconds", 0
896     motors.mou16.AddItem "Minutes", 1
897     motors.mou16.AddItem "Hours", 2
898     motors.mou16.AddItem "Days", 3

899     Dim mod17 As Module
900     Dim mod17i As Long
901     Dim mod17v As String
902     mod17i = m.Modules.Find(smFindTag, "mo17")
903     Set mod17 = m.Modules(mod17i)
904     mod17v = mod17.Data("Expression")
905     motors.mod17.value = mod17v
906     motors.mod17.AddItem "TRIA ( 54, 60, 84 )", 0
907     motors.mod17.AddItem "TRIA ( Min, Mode, mox )", 1
908     motors.mod17.AddItem "NORM ( Mean, StdDev )", 2
909     motors.mod17.AddItem "EXPO ( Mean )", 3
910     motors.mod17.AddItem "UNIF ( Min, mox )", 4
911     Dim mou17u As Module
912     Dim mou17ui As Long
913     Dim mou17uv As String
914     mou17ui = m.Modules.Find(smFindTag, "mo17")
915     Set mou17u = m.Modules(mou17ui)
916     mou17uv = mou17u.Data("Units")
917     motors.mou17.value = mou17uv
918     motors.mou17.AddItem "Seconds", 0
919     motors.mou17.AddItem "Minutes", 1
920     motors.mou17.AddItem "Hours", 2
921     motors.mou17.AddItem "Days", 3

922     Dim mod18 As Module
923     Dim mod18i As Long
924     Dim mod18v As String
925     mod18i = m.Modules.Find(smFindTag, "mo18")
926     Set mod18 = m.Modules(mod18i)
927     mod18v = mod18.Data("Expression")
```

```
928     motors.mod18.value = mod18v
929     motors.mod18.AddItem "TRIA ( 54, 60, 84 )", 0
930     motors.mod18.AddItem "TRIA ( Min, Mode, mox )", 1
931     motors.mod18.AddItem "NORM ( Mean, StdDev )", 2
932     motors.mod18.AddItem "EXPO ( Mean )", 3
933     motors.mod18.AddItem "UNIF ( Min, mox )", 4
934     Dim mou18u As Module
935     Dim mou18ui As Long
936     Dim mou18uv As String
937     mou18ui = m.Modules.Find(smFindTag, "mo18")
938     Set mou18u = m.Modules(mou18ui)
939     mou18uv = mou18u.Data("Units")
940     motors.mou18.value = mou18uv
941     motors.mou18.AddItem "Seconds", 0
942     motors.mou18.AddItem "Minutes", 1
943     motors.mou18.AddItem "Hours", 2
944     motors.mou18.AddItem "Days", 3

945   End Sub
```

Project/Module1

```
  1   Type tagOPENFILENAME
  2     lStructSize As Long
  3     hwndOwner As Long
  4     hInstance As Long
  5     strFilter As String
  6     strCustomFilter As String
  7     nMaxCustFilter As Long
  8     nFilterIndex As Long
  9     strFile As String
 10     nMaxFile As Long
 11     strFileTitle As String
 12     nMaxFileTitle As Long
 13     strInitialDir As String
 14     strTitle As String
 15     Flags As Long
 16     nFileOffset As Integer
 17     nFileExtension As Integer
 18     strDefExt As String
 19     lCustData As Long
 20     lpfnHook As Long
 21     lpTemplateName As String
 22   End Type
 23   Declare Function aht_apiGetOpenFileName Lib "comdlg32.dll" _         Alias
"GetOpenFileNameA" (OFN As tagOPENFILENAME) As Boolean
 24   Declare Function aht_apiGetSaveFileName Lib "comdlg32.dll" _         Alias
"GetSaveFileNameA" (OFN As tagOPENFILENAME) As Boolean
 25   Declare Function CommDlgExtendedError Lib "comdlg32.dll" () As Long
 26   Global Const ahtOFN_READONLY = &H1
 27   Global Const ahtOFN_OVERWRITEPROMPT = &H2
 28   Global Const ahtOFN_HIDEREADONLY = &H4
 29   Global Const ahtOFN_NOCHANGEDIR = &H8
 30   Global Const ahtOFN_SHOWHELP = &H10
 31   ' You won't use these.
 32   'Global Const ahtOFN_ENABLEHOOK = &H20
 33   'Global Const ahtOFN_ENABLETEMPLATE = &H40
 34   'Global Const ahtOFN_ENABLETEMPLATEHANDLE = &H80
 35   Global Const ahtOFN_NOVALIDATE = &H100
 36   Global Const ahtOFN_ALLOWMULTISELECT = &H200
 37   Global Const ahtOFN_EXTENSIONDIFFERENT = &H400
 38   Global Const ahtOFN_PATHMUSTEXIST = &H800
 39   Global Const ahtOFN_FILEMUSTEXIST = &H1000
 40   Global Const ahtOFN_CREATEPROMPT = &H2000
 41   Global Const ahtOFN_SHAREAWARE = &H4000
 42   Global Const ahtOFN_NOREADONLYRETURN = &H8000
 43   Global Const ahtOFN_NOTESTFILECREATE = &H10000
 44   Global Const ahtOFN_NONETWORKBUTTON = &H20000
 45   Global Const ahtOFN_NOLONGNAMES = &H40000
```

```
  ' New for Windows 95
  Global Const ahtOFN_EXPLORER = &H80000
  Global Const ahtOFN_NODEREFERENCELINKS = &H100000
  Global Const ahtOFN_LONGNAMES = &H200000

  Function ahtAddFilterItem(strFilter As String, _         strDescription
As String, Optional varItem As Variant) As String
    ' Tack a new chunk onto the file filter.
    ' That is, take the old value, stick onto it the description,
    ' (like "Databases"), a null character, the skeleton
    ' (like "*.mdb;*.mda") and a final null character.

    If IsMissing(varItem) Then varItem = "*.*"
    ahtAddFilterItem = strFilter & _
 strDescription & vbNullChar & _
 varItem & vbNullChar
  End Function

  Function ahtCommonFileOpenSave( _         Optional
ByRef Flags As Variant, _         Optional
ByVal InitialDir As Variant, _         Optional
ByVal Filter As Variant, _         Optional
ByVal FilterIndex As Variant, _         Optional
ByVal DefaultExt As Variant, _         Optional
ByVal FileName As Variant, _         Optional
ByVal DialogTitle As Variant, _         Optional
ByVal hwnd As Variant, _         Optional
ByVal OpenFile As Variant) As Variant
    ' This is the entry point you'll use to call the common
    ' file open/save dialog. The parameters are listed
    ' below, and all are optional.
    '
    ' In:
    ' Flags: one or more of the ahtOFN_* constants, OR'd together.
    ' InitialDir: the directory in which to first look
    ' Filter: a set of file filters, set up by calling
    ' AddFilterItem. See examples.
    ' FilterIndex: 1-based integer indicating which filter
    ' set to use, by default (1 if unspecified)
    ' DefaultExt: Extension to use if the user doesn't enter one.
    ' Only useful on file saves.
    ' FileName: Default value for the file name text box.
    ' DialogTitle: Title for the dialog.
    ' hWnd: parent window handle
    ' OpenFile: Boolean(True=Open File/False=Save As)
    ' Out:
    ' Return Value: Either Null or the selected filename
    Dim OFN As tagOPENFILENAME
    Dim strFileName As String
    Dim strFileTitle As String
    Dim fResult As Boolean
    ' Give the dialog a caption title.
    If IsMissing(InitialDir) Then InitialDir = CurDir
    If IsMissing(Filter) Then Filter = ".doe"
    If IsMissing(FilterIndex) Then FilterIndex = 1
    If IsMissing(Flags) Then Flags = 0&
    If IsMissing(DefaultExt) Then DefaultExt = ".doe"
    If IsMissing(FileName) Then FileName = "MILePOST"
    If IsMissing(DialogTitle) Then DialogTitle = "Save your MILePOST file"
    'XXXXXXXXXX  If IsMissing(hwnd) Then hwnd = Application.hwnd.Access.App
    If IsMissing(OpenFile) Then OpenFile = False
    ' Allocate string space for the returned strings.
    strFileName = Left(FileName & String(256, 0), 256)
    strFileTitle = String(256, 0)
    ' Set up the data structure before you call the function
    With OFN
      .lStructSize = Len(OFN)
      hwndOwner = hwnd
      .strFilter = Filter
```

```
100       .nFilterIndex = FilterIndex
101       .strFile = strFileName
102       .nMaxFile = Len(strFileName)
103       .strFileTitle = strFileTitle
104       .nMaxFileTitle = Len(strFileTitle)
105       .strTitle = DialogTitle
106       .Flags = Flags
107       .strDefExt = DefaultExt
108       .strInitialDir = InitialDir
111       .hInstance = 0
112       .strCustomFilter = ""
113       .nMaxCustFilter = 0
114       .lpfnHook = 0
115       'New for NT 4.0
116       .strCustomFilter = String(255, 0)
117       .nMaxCustFilter = 255
118     End With
122     If OpenFile Then
123       fResult = aht_apiGetOpenFileName(OFN)
124     Else
125       fResult = aht_apiGetSaveFileName(OFN)
126     End If
130     If fResult Then
136       If Not IsMissing(Flags) Then Flags = OFN.Flags
137       ahtCommonFileOpenSave = TrimNull(OFN.strFile)
138     Else
139       ahtCommonFileOpenSave = vbNullString
140     End If
141   End Function

142   Function GetOpenFile(Optional varDirectory As Variant, _         Optional
varTitleForDialog As Variant) As Variant
143     ' Here's an example that gets an Access database name.
144     Dim strFilter As String
145     Dim lngFlags As Long
146     Dim varFileName As Variant
147     ' Specify that the chosen file must already exist,
148     ' don't change directories when you're done
149     ' Also, don't bother displaying
150     ' the read-only box. It'll only confuse people.
151     lngFlags = ahtOFN_FILEMUSTEXIST Or _
ahtOFN_HIDEREADONLY Or ahtOFN_NOCHANGEDIR
152     If IsMissing(varDirectory) Then
153       varDirectory = ""
154     End If
155     If IsMissing(varTitleForDialog) Then
156       varTitleForDialog = ""
157     End If

158     ' Define the filter string and allocate space in the "c"
159     ' string Duplicate this line with changes as necessary for
160     ' more file templates.
161     strFilter = ahtAddFilterItem(strFilter, _
"Access (*.mdb)", "*.MDB;*.MDA")
162     ' Now actually call to get the file name.
163     varFileName = ahtCommonFileOpenSave( _
OpenFile:=True, _
InitialDir:=varDirectory, _
Filter:=strFilter, _
Flags:=lngFlags, _
DialogTitle:=varTitleForDialog)
164     If Not IsNull(varFileName) Then
165       varFileName = TrimNull(varFileName)
166     End If
167     GetOpenFile = varFileName
168   End Function

169   Function TestIt()
170     Dim strFilter As String
```

```
  171     Dim lngFlags As Long
  172     strFilter = ahtAddFilterItem(strFilter, "Access Files (*.mda, *.mdb)", _
  "*.MDA;*.MDB")
  173     strFilter = ahtAddFilterItem(strFilter, "dBASE Files (*.dbf)", "*.DBF")
  174     strFilter = ahtAddFilterItem(strFilter, "Text Files (*.txt)", "*.TXT")
  175     strFilter = ahtAddFilterItem(strFilter, "All Files (*.*)", "*.*")
  176     MsgBox "You selected: " & ahtCommonFileOpenSave(InitialDir:="C:\", _
  Filter:=strFilter, FilterIndex:=3, Flags:=lngFlags, _
  DialogTitle:="Hello! Open Me!")
  179     Debug.Print Hex(lngFlags)
  180   End Function

  181   Private Function TrimNull(ByVal strItem As String) As String
  182     Dim intPos As Integer
  183     intPos = InStr(strItem, vbNullChar)
  184     If intPos > 0 Then
  185       TrimNull = Left(strItem, intPos - 1)
  186     Else
  187       TrimNull = strItem
  188     End If
  189   End Function
```

Project/po1prelim

```
    1   Private Sub CommandButton3_Click()
    2     Me.Hide
    3     motors.Show
    4   End Sub

    5   Private Sub CommandButton4_Click()
    6     Hierarchy.done03.Visible = True

    7     'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
    8     Dim msgResult As Integer
    9     If (po1opt1.value = False And po1opt2.value = False) Then
   10       msgResult = MsgBox("You must make a preintegration choice.  Will the 2nd
stage and payload be preintegrated?", vbYesNo)
   11       If msgResult = vbYes Then
   12         po1opt1.value = True
   13       Else
   14         po1opt2.value = True
   15       End If
   16     End If
   17     If (po1opt3.value = False And po1opt4.value = False) Then
   18       msgResult = MsgBox("You must make an integration location decision.  Click
Yes for stage 1 and stage 2 integration on the launch pad.  Click No for stage 1 and
stage 2 integration off the launch pad.", vbYesNo)
   19       If msgResult = vbYes Then
   20         po1opt3.value = True
   21       Else
   22         po1opt4.value = True
   23       End If
   24     End If
   25     If (po1opt4.value = True And po1opt5.value = False And po1opt6.value = False)
Then
   26       msgResult = MsgBox("You must make an off-pad integration location decision.
Click Yes if integration will take place in the maintenance bay.  Click No if integration
will take place in a separate integration facility.", vbYesNo)
   27       If msgResult = vbYes Then
   28         po1opt5.value = True
   29       Else
   30         po1opt6.value = True
   31       End If
   32     End If

   33     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10
```

```
34      Dim m As Model
35      Set m = ThisDocument.Model

36      Dim pop1 As Module
37      Dim pop1i As Long
38      pop1i = m.Modules.Find(smFindTag, "pop1")
39      Set pop1 = m.Modules(pop1i)
40      pop1.Data("Expression") = po1com1.Text
41      pop1.Data("Units") = po1com2.Text

42      Dim pop2 As Module
43      Dim pop2i As Long
44      pop2i = m.Modules.Find(smFindTag, "pop2")
45      Set pop2 = m.Modules(pop2i)
46      pop2.Data("Expression") = po1com3.Text
47      pop2.Data("Units") = po1com4.Text

48      Dim pop3 As Module
49      Dim pop3i As Long
50      pop3i = m.Modules.Find(smFindTag, "pop3")
51      Set pop3 = m.Modules(pop3i)
52      pop3.Data("Expression") = po1com5.Text
53      pop3.Data("Units") = po1com6.Text

54      Dim pop4 As Module
55      Dim pop4i As Long
56      pop4i = m.Modules.Find(smFindTag, "pop4")
57      Set pop4 = m.Modules(pop4i)
58      pop4.Data("Expression") = po1com7.Text
59      pop4.Data("Units") = po1com8.Text

60      Dim pop5 As Module
61      Dim pop5i As Long
62      pop5i = m.Modules.Find(smFindTag, "pop5")
63      Set pop5 = m.Modules(pop5i)
64      pop5.Data("Expression") = po1com9.Text
65      pop5.Data("Units") = po1com10.Text

66      Dim pop6 As Module
67      Dim pop6i As Long
68      pop6i = m.Modules.Find(smFindTag, "pop6")
69      Set pop6 = m.Modules(pop6i)
70      pop6.Data("Expression") = po1com11.Text
71      pop6.Data("Units") = po1com12.Text

72      Dim pop7 As Module
73      Dim pop7i As Long
74      pop7i = m.Modules.Find(smFindTag, "pop7")
75      Set pop7 = m.Modules(pop7i)
76      pop7.Data("Expression") = po1com13.Text
77      pop7.Data("Units") = po1com14.Text

78      'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
79      Dim pov1 As Module
80      Dim pov1i As Long
81      pov1i = m.Modules.Find(smFindTag, "pov1")
82      Set pov1 = m.Modules(pov1i)
83      If po1opt1.value = True Then
84        pov1.Data("Initial Value") = "1"
85      Else
86        pov1.Data("Initial Value") = "0"
87      End If

88      Dim pov2 As Module
89      Dim pov2i As Long
90      pov2i = m.Modules.Find(smFindTag, "pov2")
91      Set pov2 = m.Modules(pov2i)
```

```
    If po1opt3.value = True Then
      pov2.Data("Initial Value") = "1"
    Else
      pov2.Data("Initial Value") = "0"
    End If

    Dim pov3 As Module
    Dim pov3i As Long
    pov3i = m.Modules.Find(smFindTag, "pov3")
    Set pov3 = m.Modules(pov3i)
    If po1opt5.value = True Then
      pov3.Data("Initial Value") = "1"
    Else
      pov3.Data("Initial Value") = "0"
    End If

    'Code below checks to see which form to show next and then shows the
appropriate form
    Me.Hide
    If po1opt3.value = True Then
      po2on.Show
    ElseIf po1opt4.value = True And po1opt1.value = True Then
      po3offpreint.Show
    Else
      po4offnopreint.Show
    End If

  End Sub

  Private Sub CommandButton5_Click()

  End Sub

  Private Sub CommandButton6_Click()
    Hierarchy.done03.Visible = True

    'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
    Dim msgResult As Integer
    If (po1opt1.value = False And po1opt2.value = False) Then
      msgResult = MsgBox("You must make a preintegration choice.  Will the 2nd
stage and payload be preintegrated?", vbYesNo)
      If msgResult = vbYes Then
        po1opt1.value = True
      Else
        po1opt2.value = True
      End If
    End If
    If (po1opt3.value = False And po1opt4.value = False) Then
      msgResult = MsgBox("You must make an integration location decision.  Click
Yes for stage 1 and stage 2 integration on the launch pad.  Click No for stage 1 and
stage 2 integration off the launch pad.", vbYesNo)
      If msgResult = vbYes Then
        po1opt3.value = True
      Else
        po1opt4.value = True
      End If
    End If
    If (po1opt4.value = True And po1opt5.value = False And po1opt6.value = False)
Then
      msgResult = MsgBox("You must make an off-pad integration location decision.
Click Yes if integration will take place in the maintenance bay.  Click No if integration
will take place in a separate integration facility.", vbYesNo)
      If msgResult = vbYes Then
        po1opt5.value = True
      Else
        po1opt6.value = True
      End If
```

```
145     End If

146     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10

147     Dim m As Model
148     Set m = ThisDocument.Model

149     Dim pop1 As Module
150     Dim pop1i As Long
151     pop1i = m.Modules.Find(smFindTag, "pop1")
152     Set pop1 = m.Modules(pop1i)
153     pop1.Data("Expression") = po1com1.Text
154     pop1.Data("Units") = po1com2.Text

155     Dim pop2 As Module
156     Dim pop2i As Long
157     pop2i = m.Modules.Find(smFindTag, "pop2")
158     Set pop2 = m.Modules(pop2i)
159     pop2.Data("Expression") = po1com3.Text
160     pop2.Data("Units") = po1com4.Text

161     Dim pop3 As Module
162     Dim pop3i As Long
163     pop3i = m.Modules.Find(smFindTag, "pop3")
164     Set pop3 = m.Modules(pop3i)
165     pop3.Data("Expression") = po1com5.Text
166     pop3.Data("Units") = po1com6.Text

167     Dim pop4 As Module
168     Dim pop4i As Long
169     pop4i = m.Modules.Find(smFindTag, "pop4")
170     Set pop4 = m.Modules(pop4i)
171     pop4.Data("Expression") = po1com7.Text
172     pop4.Data("Units") = po1com8.Text

173     Dim pop5 As Module
174     Dim pop5i As Long
175     pop5i = m.Modules.Find(smFindTag, "pop5")
176     Set pop5 = m.Modules(pop5i)
177     pop5.Data("Expression") = po1com9.Text
178     pop5.Data("Units") = po1com10.Text

179     Dim pop6 As Module
180     Dim pop6i As Long
181     pop6i = m.Modules.Find(smFindTag, "pop6")
182     Set pop6 = m.Modules(pop6i)
183     pop6.Data("Expression") = po1com11.Text
184     pop6.Data("Units") = po1com12.Text

185     Dim pop7 As Module
186     Dim pop7i As Long
187     pop7i = m.Modules.Find(smFindTag, "pop7")
188     Set pop7 = m.Modules(pop7i)
189     pop7.Data("Expression") = po1com13.Text
190     pop7.Data("Units") = po1com14.Text

191     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
192     Dim pov1 As Module
193     Dim pov1i As Long
194     pov1i = m.Modules.Find(smFindTag, "pov1")
195     Set pov1 = m.Modules(pov1i)
196     If po1opt1.value = True Then
197       pov1.Data("Initial Value") = "1"
198     Else
199       pov1.Data("Initial Value") = "0"
200     End If
```

```
    Dim pov2 As Module
    Dim pov2i As Long
    pov2i = m.Modules.Find(smFindTag, "pov2")
    Set pov2 = m.Modules(pov2i)
    If po1opt3.value = True Then
      pov2.Data("Initial Value") = "1"
    Else
      pov2.Data("Initial Value") = "0"
    End If

    Dim pov3 As Module
    Dim pov3i As Long
    pov3i = m.Modules.Find(smFindTag, "pov3")
    Set pov3 = m.Modules(pov3i)
    If po1opt5.value = True Then
      pov3.Data("Initial Value") = "1"
    Else
      pov3.Data("Initial Value") = "0"
    End If

    Me.Hide
    Hierarchy.Show
  End Sub

  Private Sub Label11_Click()

  End Sub

  Private Sub Label12_Click()

  End Sub

  Private Sub OptionButton1_Click()

  End Sub

  Private Sub OptionButton2_Click()

  End Sub

  Private Sub OptionButton4_Click()

  End Sub

  Private Sub OptionButton6_Click()

  End Sub

  Private Sub po1com1_Change()

  End Sub

  Private Sub po1opt1_Click()
    po1frm1.Visible = True
    po2on.po2frm1.Visible = True
    po2on.po2frm2.Visible = False
    po6erect.po6frm3.Visible = False

  End Sub

  Private Sub po1opt2_Click()
    po1frm1.Visible = False
    po2on.po2frm1.Visible = False
    po2on.po2frm2.Visible = True
    po6erect.po6frm3.Visible = True

  End Sub

  Private Sub po1opt3_Click()
```

```
249     po1frm2.Visible = True
250     po1frm3.Visible = False
251     po1frm4.Visible = False

252   End Sub

253   Private Sub po1opt4_Click()
254     po1frm2.Visible = False
255     po1frm3.Visible = True
256     po1frm4.Visible = True
257   End Sub

258   Private Sub po1opt5_Click()
259     po1frm4.Visible = False

260   End Sub

261   Private Sub po1opt6_Click()
262     po1frm4.Visible = True

263   End Sub

264   Private Sub TextBox1_Change()

265   End Sub

266   Private Sub ToggleButton1_Click()

267   End Sub

268   Private Sub UserForm_Click()

269   End Sub

270   Private Sub UserForm_Initialize()
271     Dim m As Model
272     Set m = ThisDocument.Model

273     'Code below populates large combo boxes for PI-02 thru PI-06 and PI-08 and PI-
10
274     Dim pop1 As Module
275     Dim pop1i As Long
276     Dim pop1v As String
277     pop1i = m.Modules.Find(smFindTag, "pop1")
278     Set pop1 = m.Modules(pop1i)
279     pop1v = pop1.Data("Expression")

280     po1prelim.po1com1.value = pop1v
281     po1prelim.po1com1.AddItem "TRIA ( 27, 30, 42 )", 0
282     po1prelim.po1com1.AddItem "TRIA ( Min, Mode, Max )", 1
283     po1prelim.po1com1.AddItem "NORM ( Mean, StdDev )", 2
284     po1prelim.po1com1.AddItem "EXPO ( Mean )", 3
285     po1prelim.po1com1.AddItem "UNIF ( Min, Max )", 4

286     Dim pop2 As Module
287     Dim pop2i As Long
288     Dim pop2v As String
289     pop2i = m.Modules.Find(smFindTag, "pop2")
290     Set pop2 = m.Modules(pop2i)
291     pop2v = pop2.Data("Expression")

292     po1prelim.po1com3.value = pop2v
293     po1prelim.po1com3.AddItem "TRIA ( 27, 30, 42 )", 0
294     po1prelim.po1com3.AddItem "TRIA ( Min, Mode, Max )", 1
295     po1prelim.po1com3.AddItem "NORM ( Mean, StdDev )", 2
296     po1prelim.po1com3.AddItem "EXPO ( Mean )", 3
297     po1prelim.po1com3.AddItem "UNIF ( Min, Max )", 4

298     Dim pop3 As Module
```

```
    Dim pop3i As Long
    Dim pop3v As String
    pop3i = m.Modules.Find(smFindTag, "pop3")
    Set pop3 = m.Modules(pop3i)
    pop3v = pop3.Data("Expression")

    po1prelim.po1com5.value = pop3v
    po1prelim.po1com5.AddItem "TRIA ( 18, 20, 28 )", 0
    po1prelim.po1com5.AddItem "TRIA ( Min, Mode, Max )", 1
    po1prelim.po1com5.AddItem "NORM ( Mean, StdDev )", 2
    po1prelim.po1com5.AddItem "EXPO ( Mean )", 3
    po1prelim.po1com5.AddItem "UNIF ( Min, Max )", 4

    Dim pop4 As Module
    Dim pop4i As Long
    Dim pop4v As String
    pop4i = m.Modules.Find(smFindTag, "pop4")
    Set pop4 = m.Modules(pop4i)
    pop4v = pop4.Data("Expression")

    po1prelim.po1com7.value = pop4v
    po1prelim.po1com7.AddItem "TRIA ( 18, 20, 28 )", 0
    po1prelim.po1com7.AddItem "TRIA ( Min, Mode, Max )", 1
    po1prelim.po1com7.AddItem "NORM ( Mean, StdDev )", 2
    po1prelim.po1com7.AddItem "EXPO ( Mean )", 3
    po1prelim.po1com7.AddItem "UNIF ( Min, Max )", 4

    Dim pop5 As Module
    Dim pop5i As Long
    Dim pop5v As String
    pop5i = m.Modules.Find(smFindTag, "pop5")
    Set pop5 = m.Modules(pop5i)
    pop5v = pop5.Data("Expression")

    po1prelim.po1com9.value = pop5v
    po1prelim.po1com9.AddItem "TRIA ( 27, 30, 42 )", 0
    po1prelim.po1com9.AddItem "TRIA ( Min, Mode, Max )", 1
    po1prelim.po1com9.AddItem "NORM ( Mean, StdDev )", 2
    po1prelim.po1com9.AddItem "EXPO ( Mean )", 3
    po1prelim.po1com9.AddItem "UNIF ( Min, Max )", 4

    Dim pop6 As Module
    Dim pop6i As Long
    Dim pop6v As String
    pop6i = m.Modules.Find(smFindTag, "pop6")
    Set pop6 = m.Modules(pop6i)
    pop6v = pop6.Data("Expression")

    po1prelim.po1com11.value = pop6v
    po1prelim.po1com11.AddItem "TRIA ( 27, 30, 42 )", 0
    po1prelim.po1com11.AddItem "TRIA ( Min, Mode, Max )", 1
    po1prelim.po1com11.AddItem "NORM ( Mean, StdDev )", 2
    po1prelim.po1com11.AddItem "EXPO ( Mean )", 3
    po1prelim.po1com11.AddItem "UNIF ( Min, Max )", 4

    Dim pop7 As Module
    Dim pop7i As Long
    Dim pop7v As String
    pop7i = m.Modules.Find(smFindTag, "pop7")
    Set pop7 = m.Modules(pop7i)
    pop7v = pop7.Data("Expression")

    po1prelim.po1com13.value = pop7v
    po1prelim.po1com13.AddItem "TRIA ( 13.5, 15, 21 )", 0
    po1prelim.po1com13.AddItem "TRIA ( Min, Mode, Max )", 1
    po1prelim.po1com13.AddItem "NORM ( Mean, StdDev )", 2
    po1prelim.po1com13.AddItem "EXPO ( Mean )", 3
    po1prelim.po1com13.AddItem "UNIF ( Min, Max )", 4
```

```
358     'Code below populates small combo boxes for PI-02 thru PI-06 and PI-08 and PI-
10
359     Dim pop1u As Module
360     Dim pop1ui As Long
361     Dim pop1uv As String
362     pop1ui = m.Modules.Find(smFindTag, "pop1")
363     Set pop1u = m.Modules(pop1ui)
364     pop1uv = pop1u.Data("Units")

365     po1prelim.po1com2.value = pop1uv
366     po1prelim.po1com2.AddItem "Seconds", 0
367     po1prelim.po1com2.AddItem "Minutes", 1
368     po1prelim.po1com2.AddItem "Hours", 2
369     po1prelim.po1com2.AddItem "Days", 3

370     Dim pop2u As Module
371     Dim pop2ui As Long
372     Dim pop2uv As String
373     pop2ui = m.Modules.Find(smFindTag, "pop2")
374     Set pop2u = m.Modules(pop2ui)
375     pop2uv = pop2u.Data("Units")

376     po1prelim.po1com4.value = pop2uv
377     po1prelim.po1com4.AddItem "Seconds", 0
378     po1prelim.po1com4.AddItem "Minutes", 1
379     po1prelim.po1com4.AddItem "Hours", 2
380     po1prelim.po1com4.AddItem "Days", 3

381     Dim pop3u As Module
382     Dim pop3ui As Long
383     Dim pop3uv As String
384     pop3ui = m.Modules.Find(smFindTag, "pop3")
385     Set pop3u = m.Modules(pop3ui)
386     pop3uv = pop3u.Data("Units")

387     po1prelim.po1com6.value = pop3uv
388     po1prelim.po1com6.AddItem "Seconds", 0
389     po1prelim.po1com6.AddItem "Minutes", 1
390     po1prelim.po1com6.AddItem "Hours", 2
391     po1prelim.po1com6.AddItem "Days", 3

392     Dim pop4u As Module
393     Dim pop4ui As Long
394     Dim pop4uv As String
395     pop4ui = m.Modules.Find(smFindTag, "pop4")
396     Set pop4u = m.Modules(pop4ui)
397     pop4uv = pop4u.Data("Units")

398     po1prelim.po1com8.value = pop4uv
399     po1prelim.po1com8.AddItem "Seconds", 0
400     po1prelim.po1com8.AddItem "Minutes", 1
401     po1prelim.po1com8.AddItem "Hours", 2
402     po1prelim.po1com8.AddItem "Days", 3

403     Dim pop5u As Module
404     Dim pop5ui As Long
405     Dim pop5uv As String
406     pop5ui = m.Modules.Find(smFindTag, "pop5")
407     Set pop5u = m.Modules(pop5ui)
408     pop5uv = pop5u.Data("Units")

409     po1prelim.po1com10.value = pop5uv
410     po1prelim.po1com10.AddItem "Seconds", 0
411     po1prelim.po1com10.AddItem "Minutes", 1
412     po1prelim.po1com10.AddItem "Hours", 2
413     po1prelim.po1com10.AddItem "Days", 3

414     Dim pop6u As Module
415     Dim pop6ui As Long
```

```
416     Dim pop6uv As String
417     pop6ui = m.Modules.Find(smFindTag, "pop6")
418     Set pop6u = m.Modules(pop6ui)
419     pop6uv = pop6u.Data("Units")

420     po1prelim.po1com12.value = pop6uv
421     po1prelim.po1com12.AddItem "Seconds", 0
422     po1prelim.po1com12.AddItem "Minutes", 1
423     po1prelim.po1com12.AddItem "Hours", 2
424     po1prelim.po1com12.AddItem "Days", 3

425     Dim pop7u As Module
426     Dim pop7ui As Long
427     Dim pop7uv As String
428     pop7ui = m.Modules.Find(smFindTag, "pop7")
429     Set pop7u = m.Modules(pop7ui)
430     pop7uv = pop7u.Data("Units")

431     po1prelim.po1com14.value = pop7uv
432     po1prelim.po1com14.AddItem "Seconds", 0
433     po1prelim.po1com14.AddItem "Minutes", 1
434     po1prelim.po1com14.AddItem "Hours", 2
435     po1prelim.po1com14.AddItem "Days", 3

436   End Sub
```

Project/po1prelim

```
1   Private Sub CommandButton3_Click()
2     Me.Hide
3     motors.Show
4   End Sub

5   Private Sub CommandButton4_Click()
6     Hierarchy.done03.Visible = True

7     'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
8     Dim msgResult As Integer
9     If (po1opt1.value = False And po1opt2.value = False) Then
10      msgResult = MsgBox("You must make a preintegration choice.  Will the 2nd
stage and payload be preintegrated?", vbYesNo)
11      If msgResult = vbYes Then
12        po1opt1.value = True
13      Else
14        po1opt2.value = True
15      End If
16    End If
17    If (po1opt3.value = False And po1opt4.value = False) Then
18      msgResult = MsgBox("You must make an integration location decision.  Click
Yes for stage 1 and stage 2 integration on the launch pad.  Click No for stage 1 and
stage 2 integration off the launch pad.", vbYesNo)
19      If msgResult = vbYes Then
20        po1opt3.value = True
21      Else
22        po1opt4.value = True
23      End If
24    End If
25    If (po1opt4.value = True And po1opt5.value = False And po1opt6.value = False)
Then
26      msgResult = MsgBox("You must make an off-pad integration location decision.
Click Yes if integration will take place in the maintenance bay.  Click No if integration
will take place in a separate integration facility.", vbYesNo)
27      If msgResult = vbYes Then
28        po1opt5.value = True
29      Else
30        po1opt6.value = True
31      End If
```

```
32      End If

33      'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10

34      Dim m As Model
35      Set m = ThisDocument.Model

36      Dim pop1 As Module
37      Dim pop1i As Long
38      pop1i = m.Modules.Find(smFindTag, "pop1")
39      Set pop1 = m.Modules(pop1i)
40      pop1.Data("Expression") = po1com1.Text
41      pop1.Data("Units") = po1com2.Text

42      Dim pop2 As Module
43      Dim pop2i As Long
44      pop2i = m.Modules.Find(smFindTag, "pop2")
45      Set pop2 = m.Modules(pop2i)
46      pop2.Data("Expression") = po1com3.Text
47      pop2.Data("Units") = po1com4.Text

48      Dim pop3 As Module
49      Dim pop3i As Long
50      pop3i = m.Modules.Find(smFindTag, "pop3")
51      Set pop3 = m.Modules(pop3i)
52      pop3.Data("Expression") = po1com5.Text
53      pop3.Data("Units") = po1com6.Text

54      Dim pop4 As Module
55      Dim pop4i As Long
56      pop4i = m.Modules.Find(smFindTag, "pop4")
57      Set pop4 = m.Modules(pop4i)
58      pop4.Data("Expression") = po1com7.Text
59      pop4.Data("Units") = po1com8.Text

60      Dim pop5 As Module
61      Dim pop5i As Long
62      pop5i = m.Modules.Find(smFindTag, "pop5")
63      Set pop5 = m.Modules(pop5i)
64      pop5.Data("Expression") = po1com9.Text
65      pop5.Data("Units") = po1com10.Text

66      Dim pop6 As Module
67      Dim pop6i As Long
68      pop6i = m.Modules.Find(smFindTag, "pop6")
69      Set pop6 = m.Modules(pop6i)
70      pop6.Data("Expression") = po1com11.Text
71      pop6.Data("Units") = po1com12.Text

72      Dim pop7 As Module
73      Dim pop7i As Long
74      pop7i = m.Modules.Find(smFindTag, "pop7")
75      Set pop7 = m.Modules(pop7i)
76      pop7.Data("Expression") = po1com13.Text
77      pop7.Data("Units") = po1com14.Text

78      'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
79      Dim pov1 As Module
80      Dim pov1i As Long
81      pov1i = m.Modules.Find(smFindTag, "pov1")
82      Set pov1 = m.Modules(pov1i)
83      If po1opt1.value = True Then
84        pov1.Data("Initial Value") = "1"
85      Else
86        pov1.Data("Initial Value") = "0"
87      End If
```

```
    Dim pov2 As Module
    Dim pov2i As Long
    pov2i = m.Modules.Find(smFindTag, "pov2")
    Set pov2 = m.Modules(pov2i)
    If po1opt3.value = True Then
      pov2.Data("Initial Value") = "1"
    Else
      pov2.Data("Initial Value") = "0"
    End If

    Dim pov3 As Module
    Dim pov3i As Long
    pov3i = m.Modules.Find(smFindTag, "pov3")
    Set pov3 = m.Modules(pov3i)
    If po1opt5.value = True Then
      pov3.Data("Initial Value") = "1"
    Else
      pov3.Data("Initial Value") = "0"
    End If

    'Code below checks to see which form to show next and then shows the
appropriate form
    Me.Hide
    If po1opt3.value = True Then
      po2on.Show
    ElseIf po1opt4.value = True And po1opt1.value = True Then
      po3offpreint.Show
    Else
      po4offnopreint.Show
    End If

  End Sub

  Private Sub CommandButton5_Click()

  End Sub

  Private Sub CommandButton6_Click()
    Hierarchy.done03.Visible = True

    'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
    Dim msgResult As Integer
    If (po1opt1.value = False And po1opt2.value = False) Then
      msgResult = MsgBox("You must make a preintegration choice.  Will the 2nd
stage and payload be preintegrated?", vbYesNo)
      If msgResult = vbYes Then
        po1opt1.value = True
      Else
        po1opt2.value = True
      End If
    End If
    If (po1opt3.value = False And po1opt4.value = False) Then
      msgResult = MsgBox("You must make an integration location decision.  Click
Yes for stage 1 and stage 2 integration on the launch pad.  Click No for stage 1 and
stage 2 integration off the launch pad.", vbYesNo)
      If msgResult = vbYes Then
        po1opt3.value = True
      Else
        po1opt4.value = True
      End If
    End If
    If (po1opt4.value = True And po1opt5.value = False And po1opt6.value = False)
Then
      msgResult = MsgBox("You must make an off-pad integration location decision.
Click Yes if integration will take place in the maintenance bay.  Click No if integration
will take place in a separate integration facility.", vbYesNo)
      If msgResult = vbYes Then
```

```
141         po1opt5.value = True
142       Else
143         po1opt6.value = True
144       End If
145     End If

146     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10

147     Dim m As Model
148     Set m = ThisDocument.Model

149     Dim pop1 As Module
150     Dim pop1i As Long
151     pop1i = m.Modules.Find(smFindTag, "pop1")
152     Set pop1 = m.Modules(pop1i)
153     pop1.Data("Expression") = po1com1.Text
154     pop1.Data("Units") = po1com2.Text

155     Dim pop2 As Module
156     Dim pop2i As Long
157     pop2i = m.Modules.Find(smFindTag, "pop2")
158     Set pop2 = m.Modules(pop2i)
159     pop2.Data("Expression") = po1com3.Text
160     pop2.Data("Units") = po1com4.Text

161     Dim pop3 As Module
162     Dim pop3i As Long
163     pop3i = m.Modules.Find(smFindTag, "pop3")
164     Set pop3 = m.Modules(pop3i)
165     pop3.Data("Expression") = po1com5.Text
166     pop3.Data("Units") = po1com6.Text

167     Dim pop4 As Module
168     Dim pop4i As Long
169     pop4i = m.Modules.Find(smFindTag, "pop4")
170     Set pop4 = m.Modules(pop4i)
171     pop4.Data("Expression") = po1com7.Text
172     pop4.Data("Units") = po1com8.Text

173     Dim pop5 As Module
174     Dim pop5i As Long
175     pop5i = m.Modules.Find(smFindTag, "pop5")
176     Set pop5 = m.Modules(pop5i)
177     pop5.Data("Expression") = po1com9.Text
178     pop5.Data("Units") = po1com10.Text

179     Dim pop6 As Module
180     Dim pop6i As Long
181     pop6i = m.Modules.Find(smFindTag, "pop6")
182     Set pop6 = m.Modules(pop6i)
183     pop6.Data("Expression") = po1com11.Text
184     pop6.Data("Units") = po1com12.Text

185     Dim pop7 As Module
186     Dim pop7i As Long
187     pop7i = m.Modules.Find(smFindTag, "pop7")
188     Set pop7 = m.Modules(pop7i)
189     pop7.Data("Expression") = po1com13.Text
190     pop7.Data("Units") = po1com14.Text

191     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
192     Dim pov1 As Module
193     Dim pov1i As Long
194     pov1i = m.Modules.Find(smFindTag, "pov1")
195     Set pov1 = m.Modules(pov1i)
196     If po1opt1.value = True Then
197       pov1.Data("Initial Value") = "1"
```

```
198     Else
199       pov1.Data("Initial Value") = "0"
200     End If

201     Dim pov2 As Module
202     Dim pov2i As Long
203     pov2i = m.Modules.Find(smFindTag, "pov2")
204     Set pov2 = m.Modules(pov2i)
205     If po1opt3.value = True Then
206       pov2.Data("Initial Value") = "1"
207     Else
208       pov2.Data("Initial Value") = "0"
209     End If

210     Dim pov3 As Module
211     Dim pov3i As Long
212     pov3i = m.Modules.Find(smFindTag, "pov3")
213     Set pov3 = m.Modules(pov3i)
214     If po1opt5.value = True Then
215       pov3.Data("Initial Value") = "1"
216     Else
217       pov3.Data("Initial Value") = "0"
218     End If

219     Me.Hide
220     Hierarchy.Show
221   End Sub

222   Private Sub Label11_Click()

223   End Sub

224   Private Sub Label12_Click()

225   End Sub

226   Private Sub OptionButton1_Click()

227   End Sub

228   Private Sub OptionButton2_Click()

229   End Sub

230   Private Sub OptionButton4_Click()

231   End Sub

232   Private Sub OptionButton6_Click()

233   End Sub

234   Private Sub po1com1_Change()

235   End Sub

236   Private Sub po1opt1_Click()
237     po1frm1.Visible = True
238     po2on.po2frm1.Visible = True
239     po2on.po2frm2.Visible = False
240     po6erect.po6frm3.Visible = False

241   End Sub

242   Private Sub po1opt2_Click()
243     po1frm1.Visible = False
244     po2on.po2frm1.Visible = False
245     po2on.po2frm2.Visible = True
246     po6erect.po6frm3.Visible = True
```

```
247   End Sub

248   Private Sub po1opt3_Click()
249     po1frm2.Visible = True
250     po1frm3.Visible = False
251     po1frm4.Visible = False

252   End Sub

253   Private Sub po1opt4_Click()
254     po1frm2.Visible = False
255     po1frm3.Visible = True
256     po1frm4.Visible = True
257   End Sub

258   Private Sub po1opt5_Click()
259     po1frm4.Visible = False

260   End Sub

261   Private Sub po1opt6_Click()
262     po1frm4.Visible = True

263   End Sub

264   Private Sub TextBox1_Change()

265   End Sub

266   Private Sub ToggleButton1_Click()

267   End Sub

268   Private Sub UserForm_Click()

269   End Sub

270   Private Sub UserForm_Initialize()
271     Dim m As Model
272     Set m = ThisDocument.Model

273     'Code below populates large combo boxes for PI-02 thru PI-06 and PI-08 and PI-
10
274     Dim pop1 As Module
275     Dim pop1i As Long
276     Dim pop1v As String
277     pop1i = m.Modules.Find(smFindTag, "pop1")
278     Set pop1 = m.Modules(pop1i)
279     pop1v = pop1.Data("Expression")

280     po1prelim.po1com1.value = pop1v
281     po1prelim.po1com1.AddItem "TRIA ( 27, 30, 42 )", 0
282     po1prelim.po1com1.AddItem "TRIA ( Min, Mode, Max )", 1
283     po1prelim.po1com1.AddItem "NORM ( Mean, StdDev )", 2
284     po1prelim.po1com1.AddItem "EXPO ( Mean )", 3
285     po1prelim.po1com1.AddItem "UNIF ( Min, Max )", 4

286     Dim pop2 As Module
287     Dim pop2i As Long
288     Dim pop2v As String
289     pop2i = m.Modules.Find(smFindTag, "pop2")
290     Set pop2 = m.Modules(pop2i)
291     pop2v = pop2.Data("Expression")

292     po1prelim.po1com3.value = pop2v
293     po1prelim.po1com3.AddItem "TRIA ( 27, 30, 42 )", 0
294     po1prelim.po1com3.AddItem "TRIA ( Min, Mode, Max )", 1
295     po1prelim.po1com3.AddItem "NORM ( Mean, StdDev )", 2
```

```
296     po1prelim.po1com3.AddItem "EXPO ( Mean )", 3
297     po1prelim.po1com3.AddItem "UNIF ( Min, Max )", 4

298     Dim pop3 As Module
299     Dim pop3i As Long
300     Dim pop3v As String
301     pop3i = m.Modules.Find(smFindTag, "pop3")
302     Set pop3 = m.Modules(pop3i)
303     pop3v = pop3.Data("Expression")

304     po1prelim.po1com5.value = pop3v
305     po1prelim.po1com5.AddItem "TRIA ( 18, 20, 28 )", 0
306     po1prelim.po1com5.AddItem "TRIA ( Min, Mode, Max )", 1
307     po1prelim.po1com5.AddItem "NORM ( Mean, StdDev )", 2
308     po1prelim.po1com5.AddItem "EXPO ( Mean )", 3
309     po1prelim.po1com5.AddItem "UNIF ( Min, Max )", 4

310     Dim pop4 As Module
311     Dim pop4i As Long
312     Dim pop4v As String
313     pop4i = m.Modules.Find(smFindTag, "pop4")
314     Set pop4 = m.Modules(pop4i)
315     pop4v = pop4.Data("Expression")

316     po1prelim.po1com7.value = pop4v
317     po1prelim.po1com7.AddItem "TRIA ( 18, 20, 28 )", 0
318     po1prelim.po1com7.AddItem "TRIA ( Min, Mode, Max )", 1
319     po1prelim.po1com7.AddItem "NORM ( Mean, StdDev )", 2
320     po1prelim.po1com7.AddItem "EXPO ( Mean )", 3
321     po1prelim.po1com7.AddItem "UNIF ( Min, Max )", 4

322     Dim pop5 As Module
323     Dim pop5i As Long
324     Dim pop5v As String
325     pop5i = m.Modules.Find(smFindTag, "pop5")
326     Set pop5 = m.Modules(pop5i)
327     pop5v = pop5.Data("Expression")

328     po1prelim.po1com9.value = pop5v
329     po1prelim.po1com9.AddItem "TRIA ( 27, 30, 42 )", 0
330     po1prelim.po1com9.AddItem "TRIA ( Min, Mode, Max )", 1
331     po1prelim.po1com9.AddItem "NORM ( Mean, StdDev )", 2
332     po1prelim.po1com9.AddItem "EXPO ( Mean )", 3
333     po1prelim.po1com9.AddItem "UNIF ( Min, Max )", 4

334     Dim pop6 As Module
335     Dim pop6i As Long
336     Dim pop6v As String
337     pop6i = m.Modules.Find(smFindTag, "pop6")
338     Set pop6 = m.Modules(pop6i)
339     pop6v = pop6.Data("Expression")

340     po1prelim.po1com11.value = pop6v
341     po1prelim.po1com11.AddItem "TRIA ( 27, 30, 42 )", 0
342     po1prelim.po1com11.AddItem "TRIA ( Min, Mode, Max )", 1
343     po1prelim.po1com11.AddItem "NORM ( Mean, StdDev )", 2
344     po1prelim.po1com11.AddItem "EXPO ( Mean )", 3
345     po1prelim.po1com11.AddItem "UNIF ( Min, Max )", 4

346     Dim pop7 As Module
347     Dim pop7i As Long
348     Dim pop7v As String
349     pop7i = m.Modules.Find(smFindTag, "pop7")
350     Set pop7 = m.Modules(pop7i)
351     pop7v = pop7.Data("Expression")

352     po1prelim.po1com13.value = pop7v
353     po1prelim.po1com13.AddItem "TRIA ( 13.5, 15, 21 )", 0
354     po1prelim.po1com13.AddItem "TRIA ( Min, Mode, Max )", 1
```

```
355     po1prelim.po1com13.AddItem "NORM ( Mean, StdDev )", 2
356     po1prelim.po1com13.AddItem "EXPO ( Mean )", 3
357     po1prelim.po1com13.AddItem "UNIF ( Min, Max )", 4

358     'Code below populates small combo boxes for PI-02 thru PI-06 and PI-08 and PI-
10
359     Dim pop1u As Module
360     Dim pop1ui As Long
361     Dim pop1uv As String
362     pop1ui = m.Modules.Find(smFindTag, "pop1")
363     Set pop1u = m.Modules(pop1ui)
364     pop1uv = pop1u.Data("Units")

365     po1prelim.po1com2.value = pop1uv
366     po1prelim.po1com2.AddItem "Seconds", 0
367     po1prelim.po1com2.AddItem "Minutes", 1
368     po1prelim.po1com2.AddItem "Hours", 2
369     po1prelim.po1com2.AddItem "Days", 3

370     Dim pop2u As Module
371     Dim pop2ui As Long
372     Dim pop2uv As String
373     pop2ui = m.Modules.Find(smFindTag, "pop2")
374     Set pop2u = m.Modules(pop2ui)
375     pop2uv = pop2u.Data("Units")

376     po1prelim.po1com4.value = pop2uv
377     po1prelim.po1com4.AddItem "Seconds", 0
378     po1prelim.po1com4.AddItem "Minutes", 1
379     po1prelim.po1com4.AddItem "Hours", 2
380     po1prelim.po1com4.AddItem "Days", 3

381     Dim pop3u As Module
382     Dim pop3ui As Long
383     Dim pop3uv As String
384     pop3ui = m.Modules.Find(smFindTag, "pop3")
385     Set pop3u = m.Modules(pop3ui)
386     pop3uv = pop3u.Data("Units")

387     po1prelim.po1com6.value = pop3uv
388     po1prelim.po1com6.AddItem "Seconds", 0
389     po1prelim.po1com6.AddItem "Minutes", 1
390     po1prelim.po1com6.AddItem "Hours", 2
391     po1prelim.po1com6.AddItem "Days", 3

392     Dim pop4u As Module
393     Dim pop4ui As Long
394     Dim pop4uv As String
395     pop4ui = m.Modules.Find(smFindTag, "pop4")
396     Set pop4u = m.Modules(pop4ui)
397     pop4uv = pop4u.Data("Units")

398     po1prelim.po1com8.value = pop4uv
399     po1prelim.po1com8.AddItem "Seconds", 0
400     po1prelim.po1com8.AddItem "Minutes", 1
401     po1prelim.po1com8.AddItem "Hours", 2
402     po1prelim.po1com8.AddItem "Days", 3

403     Dim pop5u As Module
404     Dim pop5ui As Long
405     Dim pop5uv As String
406     pop5ui = m.Modules.Find(smFindTag, "pop5")
407     Set pop5u = m.Modules(pop5ui)
408     pop5uv = pop5u.Data("Units")

409     po1prelim.po1com10.value = pop5uv
410     po1prelim.po1com10.AddItem "Seconds", 0
411     po1prelim.po1com10.AddItem "Minutes", 1
412     po1prelim.po1com10.AddItem "Hours", 2
```

```
413     po1prelim.po1com10.AddItem "Days", 3

414     Dim pop6u As Module
415     Dim pop6ui As Long
416     Dim pop6uv As String
417     pop6ui = m.Modules.Find(smFindTag, "pop6")
418     Set pop6u = m.Modules(pop6ui)
419     pop6uv = pop6u.Data("Units")

420     po1prelim.po1com12.value = pop6uv
421     po1prelim.po1com12.AddItem "Seconds", 0
422     po1prelim.po1com12.AddItem "Minutes", 1
423     po1prelim.po1com12.AddItem "Hours", 2
424     po1prelim.po1com12.AddItem "Days", 3

425     Dim pop7u As Module
426     Dim pop7ui As Long
427     Dim pop7uv As String
428     pop7ui = m.Modules.Find(smFindTag, "pop7")
429     Set pop7u = m.Modules(pop7ui)
430     pop7uv = pop7u.Data("Units")

431     po1prelim.po1com14.value = pop7uv
432     po1prelim.po1com14.AddItem "Seconds", 0
433     po1prelim.po1com14.AddItem "Minutes", 1
434     po1prelim.po1com14.AddItem "Hours", 2
435     po1prelim.po1com14.AddItem "Days", 3

436   End Sub
```

Project/po2on

```
    1   Private Sub CommandButton3_Click()

    2   End Sub

    3   Private Sub CommandButton6_Click()
    4     Me.Hide
    5     po1prelim.Show
    6   End Sub

    7   Private Sub CommandButton7_Click()
    8     Hierarchy.done04.Visible = True

    9     'Code below checks if any option button sets are not clicked, and if so, forces
the user to make a decision
   10     Dim msgResult As Integer
   11     If (po2opt1.value = False And po2opt2.value = False) Then
   12       msgResult = MsgBox("You must make a hypergolic fuels decision.  Are
hypergolic fuels required?", vbYesNo)
   13       If msgResult = vbYes Then
   14         po2opt1.value = True
   15       Else
   16         po2opt2.value = True
   17       End If
   18     End If
   19     If (po2opt1.value = True And po2opt3.value = False And po2opt4.value = False)
Then
   20       msgResult = MsgBox("You must make a hypergolic fuels loading decision.  Click
Yes if hypergolics are loaded now, in the integration facility.  Click No if hypergolics
are loaded later, on the launch pad.", vbYesNo)
   21       If msgResult = vbYes Then
   22         po2opt3.value = True
   23       Else
   24         po2opt4.value = True
   25       End If
   26     End If
   27     If (po2opt5.value = False And po2opt6.value = False) Then
   28       msgResult = MsgBox("You must make an ordnance decision.  Is ordnance
required?", vbYesNo)
   29       If msgResult = vbYes Then
   30         po2opt5.value = True
   31       Else
   32         po2opt6.value = True
   33       End If
   34     End If
   35     If (po2opt5.value = True And po2opt7.value = False And po2opt8.value = False)
Then
   36       msgResult = MsgBox("You must make an ordnance installation location decision.
Click Yes if ordnance is loaded now, in the integration facility.  Click No if ordnance
is loaded later, on the launch pad.", vbYesNo)
   37       If msgResult = vbYes Then
   38         po2opt7.value = True
   39       Else
   40         po2opt8.value = True
   41       End If
   42     End If

   43     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10
   44     Dim m As Model
   45     Set m = ThisDocument.Model

   46     Dim pop47 As Module
   47     Dim pop47i As Long
   48     pop47i = m.Modules.Find(smFindTag, "pop47")
   49     Set pop47 = m.Modules(pop47i)
   50     pop47.Data("Expression") = po2com1.Text
```

```
    pop47.Data("Units") = po2com2.Text

    Dim pop48 As Module
    Dim pop48i As Long
    pop48i = m.Modules.Find(smFindTag, "pop48")
    Set pop48 = m.Modules(pop48i)
    pop48.Data("Expression") = po2com3.Text
    pop48.Data("Units") = po2com4.Text

    Dim pop49 As Module
    Dim pop49i As Long
    pop49i = m.Modules.Find(smFindTag, "pop49")
    Set pop49 = m.Modules(pop49i)
    pop49.Data("Expression") = po2com5.Text
    pop49.Data("Units") = po2com6.Text

    Dim pop50 As Module
    Dim pop50i As Long
    pop50i = m.Modules.Find(smFindTag, "pop50")
    Set pop50 = m.Modules(pop50i)
    pop50.Data("Expression") = po2com7.Text
    pop50.Data("Units") = po2com8.Text

    Dim pop51 As Module
    Dim pop51i As Long
    pop51i = m.Modules.Find(smFindTag, "pop51")
    Set pop51 = m.Modules(pop51i)
    pop51.Data("Expression") = po2com9.Text
    pop51.Data("Units") = po2com10.Text

    Dim pop52 As Module
    Dim pop52i As Long
    pop52i = m.Modules.Find(smFindTag, "pop52")
    Set pop52 = m.Modules(pop52i)
    pop52.Data("Expression") = po2com11.Text
    pop52.Data("Units") = po2com12.Text

    Dim pop64 As Module
    Dim pop64i As Long
    pop64i = m.Modules.Find(smFindTag, "pop64")
    Set pop64 = m.Modules(pop64i)
    If po2frm1.Visible = True Then
      pop64.Data("Expression") = po2com13.Text
      pop64.Data("Units") = po2com14.Text
    Else
      pop64.Data("Expression") = po2com37.Text
      pop64.Data("Units") = po2com38.Text
    End If

    Dim pop53 As Module
    Dim pop53i As Long
    pop53i = m.Modules.Find(smFindTag, "pop53")
    Set pop53 = m.Modules(pop53i)
    pop53.Data("Expression") = po2com15.Text
    pop53.Data("Units") = po2com16.Text

    Dim pop54 As Module
    Dim pop54i As Long
    pop54i = m.Modules.Find(smFindTag, "pop54")
    Set pop54 = m.Modules(pop54i)
    pop54.Data("Expression") = po2com17.Text
    pop54.Data("Units") = po2com18.Text

    Dim pop55 As Module
    Dim pop55i As Long
    pop55i = m.Modules.Find(smFindTag, "pop55")
    Set pop55 = m.Modules(pop55i)
    pop55.Data("Expression") = po2com19.Text
    pop55.Data("Units") = po2com20.Text
```

```
111     Dim pop56 As Module
112     Dim pop56i As Long
113     pop56i = m.Modules.Find(smFindTag, "pop56")
114     Set pop56 = m.Modules(pop56i)
115     pop56.Data("Expression") = po2com21.Text
116     pop56.Data("Units") = po2com22.Text

117     Dim pop57 As Module
118     Dim pop57i As Long
119     pop57i = m.Modules.Find(smFindTag, "pop57")
120     Set pop57 = m.Modules(pop57i)
121     pop57.Data("Expression") = po2com23.Text
122     pop57.Data("Units") = po2com24.Text

123     Dim pop58 As Module
124     Dim pop58i As Long
125     pop58i = m.Modules.Find(smFindTag, "pop58")
126     Set pop58 = m.Modules(pop58i)
127     pop58.Data("Expression") = po2com25.Text
128     pop58.Data("Units") = po2com26.Text

129     Dim pop59 As Module
130     Dim pop59i As Long
131     pop59i = m.Modules.Find(smFindTag, "pop59")
132     Set pop59 = m.Modules(pop59i)
133     pop59.Data("Expression") = po2com27.Text
134     pop59.Data("Units") = po2com28.Text

135     Dim pop60 As Module
136     Dim pop60i As Long
137     pop60i = m.Modules.Find(smFindTag, "pop60")
138     Set pop60 = m.Modules(pop60i)
139     pop60.Data("Expression") = po2com29.Text
140     pop60.Data("Units") = po2com30.Text

141     Dim pop61 As Module
142     Dim pop61i As Long
143     pop61i = m.Modules.Find(smFindTag, "pop61")
144     Set pop61 = m.Modules(pop61i)
145     pop61.Data("Expression") = po2com31.Text
146     pop61.Data("Units") = po2com32.Text

147     Dim pop62 As Module
148     Dim pop62i As Long
149     pop62i = m.Modules.Find(smFindTag, "pop62")
150     Set pop62 = m.Modules(pop62i)
151     pop62.Data("Expression") = po2com33.Text
152     pop62.Data("Units") = po2com34.Text

153     Dim pop63 As Module
154     Dim pop63i As Long
155     pop63i = m.Modules.Find(smFindTag, "pop63")
156     Set pop63 = m.Modules(pop63i)
157     pop63.Data("Expression") = po2com35.Text
158     pop63.Data("Units") = po2com36.Text

159     Dim pop34 As Module
160     Dim pop34i As Long
161     pop34i = m.Modules.Find(smFindTag, "pop34")
162     Set pop34 = m.Modules(pop34i)
163     pop34.Data("Expression") = po2com39.Text
164     pop34.Data("Units") = po2com40.Text

165     Dim pop71 As Module
166     Dim pop71i As Long
167     pop71i = m.Modules.Find(smFindTag, "pop71")
168     Set pop71 = m.Modules(pop71i)
169     pop71.Data("Expression") = po2com39.Text
```

```
  170     pop71.Data("Units") = po2com40.Text

  171     Dim pop35 As Module
  172     Dim pop35i As Long
  173     pop35i = m.Modules.Find(smFindTag, "pop35")
  174     Set pop35 = m.Modules(pop35i)
  175     pop35.Data("Expression") = po2com41.Text
  176     pop35.Data("Units") = po2com42.Text

  177     Dim pop77 As Module
  178     Dim pop77i As Long
  179     pop77i = m.Modules.Find(smFindTag, "pop77")
  180     Set pop77 = m.Modules(pop77i)
  181     pop77.Data("Expression") = po2com41.Text
  182     pop77.Data("Units") = po2com42.Text

  183     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
  184     Dim pov6 As Module
  185     Dim pov6i As Long
  186     pov6i = m.Modules.Find(smFindTag, "pov6")
  187     Set pov6 = m.Modules(pov6i)
  188     If po2opt2.value = True Then
  189       pov6.Data("Initial Value") = "0"
  190     ElseIf po2opt3.value = True Then
  191       pov6.Data("Initial Value") = "1"
  192     Else
  193       pov6.Data("Initial Value") = "2"
  194     End If

  195     Dim pov7 As Module
  196     Dim pov7i As Long
  197     pov7i = m.Modules.Find(smFindTag, "pov7")
  198     Set pov7 = m.Modules(pov7i)
  199     If po2opt6.value = True Then
  200       pov7.Data("Initial Value") = "0"
  201     ElseIf po2opt7.value = True Then
  202       pov7.Data("Initial Value") = "1"
  203     Else
  204       pov7.Data("Initial Value") = "2"
  205     End If

  206     'code below hides the current form and shows the next form
  207     Me.Hide
  208     po7umbilical.Show

  209   End Sub

  210   Private Sub CommandButton9_Click()
  211     Hierarchy.done04.Visible = True

  212     'Code below checks if any option button sets are not clicked, and if so, forces
the user to make a decision
  213     Dim msgResult As Integer
  214     If (po2opt1.value = False And po2opt2.value = False) Then
  215       msgResult = MsgBox("You must make a hypergolic fuels decision.  Are
hypergolic fuels required?", vbYesNo)
  216       If msgResult = vbYes Then
  217         po2opt1.value = True
  218       Else
  219         po2opt2.value = True
  220       End If
  221     End If
  222     If (po2opt1.value = True And po2opt3.value = False And po2opt4.value = False)
Then
  223       msgResult = MsgBox("You must make a hypergolic fuels loading decision.  Click
Yes if hypergolics are loaded now, in the integration facility.  Click No if hypergolics
are loaded later, on the launch pad.", vbYesNo)
  224       If msgResult = vbYes Then
```

```
  225         po2opt3.value = True
  226       Else
  227         po2opt4.value = True
  228       End If
  229     End If
  230     If (po2opt5.value = False And po2opt6.value = False) Then
  231       msgResult = MsgBox("You must make an ordnance decision.  Is ordnance
required?", vbYesNo)
  232       If msgResult = vbYes Then
  233         po2opt5.value = True
  234       Else
  235         po2opt6.value = True
  236       End If
  237     End If
  238     If (po2opt5.value = True And po2opt7.value = False And po2opt8.value = False)
Then
  239       msgResult = MsgBox("You must make an ordnance installation location decision.
Click Yes if ordnance is loaded now, in the integration facility.  Click No if ordnance
is loaded later, on the launch pad.", vbYesNo)
  240       If msgResult = vbYes Then
  241         po2opt7.value = True
  242       Else
  243         po2opt8.value = True
  244       End If
  245     End If

  246     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10
  247     Dim m As Model
  248     Set m = ThisDocument.Model

  249     Dim pop47 As Module
  250     Dim pop47i As Long
  251     pop47i = m.Modules.Find(smFindTag, "pop47")
  252     Set pop47 = m.Modules(pop47i)
  253     pop47.Data("Expression") = po2com1.Text
  254     pop47.Data("Units") = po2com2.Text

  255     Dim pop48 As Module
  256     Dim pop48i As Long
  257     pop48i = m.Modules.Find(smFindTag, "pop48")
  258     Set pop48 = m.Modules(pop48i)
  259     pop48.Data("Expression") = po2com3.Text
  260     pop48.Data("Units") = po2com4.Text

  261     Dim pop49 As Module
  262     Dim pop49i As Long
  263     pop49i = m.Modules.Find(smFindTag, "pop49")
  264     Set pop49 = m.Modules(pop49i)
  265     pop49.Data("Expression") = po2com5.Text
  266     pop49.Data("Units") = po2com6.Text

  267     Dim pop50 As Module
  268     Dim pop50i As Long
  269     pop50i = m.Modules.Find(smFindTag, "pop50")
  270     Set pop50 = m.Modules(pop50i)
  271     pop50.Data("Expression") = po2com7.Text
  272     pop50.Data("Units") = po2com8.Text

  273     Dim pop51 As Module
  274     Dim pop51i As Long
  275     pop51i = m.Modules.Find(smFindTag, "pop51")
  276     Set pop51 = m.Modules(pop51i)
  277     pop51.Data("Expression") = po2com9.Text
  278     pop51.Data("Units") = po2com10.Text

  279     Dim pop52 As Module
  280     Dim pop52i As Long
  281     pop52i = m.Modules.Find(smFindTag, "pop52")
```

```
282     Set pop52 = m.Modules(pop52i)
283     pop52.Data("Expression") = po2com11.Text
284     pop52.Data("Units") = po2com12.Text

285     Dim pop64 As Module
286     Dim pop64i As Long
287     pop64i = m.Modules.Find(smFindTag, "pop64")
288     Set pop64 = m.Modules(pop64i)
289     If po2frm1.Visible = True Then
290       pop64.Data("Expression") = po2com13.Text
291       pop64.Data("Units") = po2com14.Text
292     Else
293       pop64.Data("Expression") = po2com37.Text
294       pop64.Data("Units") = po2com38.Text
295     End If

296     Dim pop53 As Module
297     Dim pop53i As Long
298     pop53i = m.Modules.Find(smFindTag, "pop53")
299     Set pop53 = m.Modules(pop53i)
300     pop53.Data("Expression") = po2com15.Text
301     pop53.Data("Units") = po2com16.Text

302     Dim pop54 As Module
303     Dim pop54i As Long
304     pop54i = m.Modules.Find(smFindTag, "pop54")
305     Set pop54 = m.Modules(pop54i)
306     pop54.Data("Expression") = po2com17.Text
307     pop54.Data("Units") = po2com18.Text

308     Dim pop55 As Module
309     Dim pop55i As Long
310     pop55i = m.Modules.Find(smFindTag, "pop55")
311     Set pop55 = m.Modules(pop55i)
312     pop55.Data("Expression") = po2com19.Text
313     pop55.Data("Units") = po2com20.Text

314     Dim pop56 As Module
315     Dim pop56i As Long
316     pop56i = m.Modules.Find(smFindTag, "pop56")
317     Set pop56 = m.Modules(pop56i)
318     pop56.Data("Expression") = po2com21.Text
319     pop56.Data("Units") = po2com22.Text

320     Dim pop57 As Module
321     Dim pop57i As Long
322     pop57i = m.Modules.Find(smFindTag, "pop57")
323     Set pop57 = m.Modules(pop57i)
324     pop57.Data("Expression") = po2com23.Text
325     pop57.Data("Units") = po2com24.Text

326     Dim pop58 As Module
327     Dim pop58i As Long
328     pop58i = m.Modules.Find(smFindTag, "pop58")
329     Set pop58 = m.Modules(pop58i)
330     pop58.Data("Expression") = po2com25.Text
331     pop58.Data("Units") = po2com26.Text

332     Dim pop59 As Module
333     Dim pop59i As Long
334     pop59i = m.Modules.Find(smFindTag, "pop59")
335     Set pop59 = m.Modules(pop59i)
336     pop59.Data("Expression") = po2com27.Text
337     pop59.Data("Units") = po2com28.Text

338     Dim pop60 As Module
339     Dim pop60i As Long
340     pop60i = m.Modules.Find(smFindTag, "pop60")
341     Set pop60 = m.Modules(pop60i)
```

```
342     pop60.Data("Expression") = po2com29.Text
343     pop60.Data("Units") = po2com30.Text

344     Dim pop61 As Module
345     Dim pop61i As Long
346     pop61i = m.Modules.Find(smFindTag, "pop61")
347     Set pop61 = m.Modules(pop61i)
348     pop61.Data("Expression") = po2com31.Text
349     pop61.Data("Units") = po2com32.Text

350     Dim pop62 As Module
351     Dim pop62i As Long
352     pop62i = m.Modules.Find(smFindTag, "pop62")
353     Set pop62 = m.Modules(pop62i)
354     pop62.Data("Expression") = po2com33.Text
355     pop62.Data("Units") = po2com34.Text

356     Dim pop63 As Module
357     Dim pop63i As Long
358     pop63i = m.Modules.Find(smFindTag, "pop63")
359     Set pop63 = m.Modules(pop63i)
360     pop63.Data("Expression") = po2com35.Text
361     pop63.Data("Units") = po2com36.Text

362     Dim pop34 As Module
363     Dim pop34i As Long
364     pop34i = m.Modules.Find(smFindTag, "pop34")
365     Set pop34 = m.Modules(pop34i)
366     pop34.Data("Expression") = po2com39.Text
367     pop34.Data("Units") = po2com40.Text

368     Dim pop71 As Module
369     Dim pop71i As Long
370     pop71i = m.Modules.Find(smFindTag, "pop71")
371     Set pop71 = m.Modules(pop71i)
372     pop71.Data("Expression") = po2com39.Text
373     pop71.Data("Units") = po2com40.Text

374     Dim pop35 As Module
375     Dim pop35i As Long
376     pop35i = m.Modules.Find(smFindTag, "pop35")
377     Set pop35 = m.Modules(pop35i)
378     pop35.Data("Expression") = po2com41.Text
379     pop35.Data("Units") = po2com42.Text

380     Dim pop77 As Module
381     Dim pop77i As Long
382     pop77i = m.Modules.Find(smFindTag, "pop77")
383     Set pop77 = m.Modules(pop77i)
384     pop77.Data("Expression") = po2com41.Text
385     pop77.Data("Units") = po2com42.Text

386     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
387     Dim pov6 As Module
388     Dim pov6i As Long
389     pov6i = m.Modules.Find(smFindTag, "pov6")
390     Set pov6 = m.Modules(pov6i)
391     If po2opt2.value = True Then
392       pov6.Data("Initial Value") = "0"
393     ElseIf po2opt3.value = True Then
394       pov6.Data("Initial Value") = "1"
395     Else
396       pov6.Data("Initial Value") = "2"
397     End If

398     Dim pov7 As Module
399     Dim pov7i As Long
400     pov7i = m.Modules.Find(smFindTag, "pov7")
```

```
401     Set pov7 = m.Modules(pov7i)
402     If po2opt6.value = True Then
403       pov7.Data("Initial Value") = "0"
404     ElseIf po2opt7.value = True Then
405       pov7.Data("Initial Value") = "1"
406     Else
407       pov7.Data("Initial Value") = "2"
408     End If

409     'code below hides the current form and shows the main form
410     Me.Hide
411     Hierarchy.Show

412   End Sub

413   Private Sub Label11_Click()

414   End Sub

415   Private Sub Label12_Click()

416   End Sub

417   Private Sub Label23_Click()

418   End Sub

419   Private Sub Label6_Click()

420   End Sub

421   Private Sub OptionButton1_Click()

422   End Sub

423   Private Sub OptionButton2_Click()

424   End Sub

425   Private Sub OptionButton4_Click()

426   End Sub

427   Private Sub OptionButton6_Click()

428   End Sub

429   Private Sub po2opt1_Click()
430     po2frm3.Visible = True

431   End Sub

432   Private Sub po2opt2_Click()
433     po2frm3.Visible = False

434   End Sub

435   Private Sub po2opt5_Click()
436     po2frm4.Visible = True

437   End Sub

438   Private Sub po2opt6_Click()
439     po2frm4.Visible = False

440   End Sub

441   Private Sub TextBox23_Change()
```

```
442   End Sub

443   Private Sub ToggleButton1_Click()

444   End Sub

445   Private Sub UserForm_Click()

446   End Sub

447   Private Sub UserForm_Initialize()
448     Dim m As Model
449     Set m = ThisDocument.Model

450     'Code below populates large combo boxes for OP-01 thru OP-25
451     Dim pop47 As Module
452     Dim pop47i As Long
453     Dim pop47v As String
454     pop47i = m.Modules.Find(smFindTag, "pop47")
455     Set pop47 = m.Modules(pop47i)
456     pop47v = pop47.Data("Expression")

457     po2on.po2com1.value = pop47v
458     po2on.po2com1.AddItem "TRIA ( 54, 60, 84 )", 0
459     po2on.po2com1.AddItem "TRIA ( Min, Mode, Max )", 1
460     po2on.po2com1.AddItem "NORM ( Mean, StdDev )", 2
461     po2on.po2com1.AddItem "EXPO ( Mean )", 3
462     po2on.po2com1.AddItem "UNIF ( Min, Max )", 4

463     Dim pop48 As Module
464     Dim pop48i As Long
465     Dim pop48v As String
466     pop48i = m.Modules.Find(smFindTag, "pop48")
467     Set pop48 = m.Modules(pop48i)
468     pop48v = pop48.Data("Expression")

469     po2on.po2com3.value = pop48v
470     po2on.po2com3.AddItem "TRIA ( 108, 120, 168 )", 0
471     po2on.po2com3.AddItem "TRIA ( Min, Mode, Max )", 1
472     po2on.po2com3.AddItem "NORM ( Mean, StdDev )", 2
473     po2on.po2com3.AddItem "EXPO ( Mean )", 3
474     po2on.po2com3.AddItem "UNIF ( Min, Max )", 4

475     Dim pop49 As Module
476     Dim pop49i As Long
477     Dim pop49v As String
478     pop49i = m.Modules.Find(smFindTag, "pop49")
479     Set pop49 = m.Modules(pop49i)
480     pop49v = pop49.Data("Expression")

481     po2on.po2com5.value = pop49v
482     po2on.po2com5.AddItem "TRIA ( 27, 30, 42 )", 0
483     po2on.po2com5.AddItem "TRIA ( Min, Mode, Max )", 1
484     po2on.po2com5.AddItem "NORM ( Mean, StdDev )", 2
485     po2on.po2com5.AddItem "EXPO ( Mean )", 3
486     po2on.po2com5.AddItem "UNIF ( Min, Max )", 4

487     Dim pop50 As Module
488     Dim pop50i As Long
489     Dim pop50v As String
490     pop50i = m.Modules.Find(smFindTag, "pop50")
491     Set pop50 = m.Modules(pop50i)
492     pop50v = pop50.Data("Expression")

493     po2on.po2com7.value = pop50v
494     po2on.po2com7.AddItem "TRIA ( 81, 90, 126 )", 0
495     po2on.po2com7.AddItem "TRIA ( Min, Mode, Max )", 1
496     po2on.po2com7.AddItem "NORM ( Mean, StdDev )", 2
497     po2on.po2com7.AddItem "EXPO ( Mean )", 3
```

```
498     po2on.po2com7.AddItem "UNIF ( Min, Max )", 4

499     Dim pop51 As Module
500     Dim pop51i As Long
501     Dim pop51v As String
502     pop51i = m.Modules.Find(smFindTag, "pop51")
503     Set pop51 = m.Modules(pop51i)
504     pop51v = pop51.Data("Expression")

505     po2on.po2com9.value = pop51v
506     po2on.po2com9.AddItem "TRIA ( 36, 40, 56 )", 0
507     po2on.po2com9.AddItem "TRIA ( Min, Mode, Max )", 1
508     po2on.po2com9.AddItem "NORM ( Mean, StdDev )", 2
509     po2on.po2com9.AddItem "EXPO ( Mean )", 3
510     po2on.po2com9.AddItem "UNIF ( Min, Max )", 4

511     Dim pop52 As Module
512     Dim pop52i As Long
513     Dim pop52v As String
514     pop52i = m.Modules.Find(smFindTag, "pop52")
515     Set pop52 = m.Modules(pop52i)
516     pop52v = pop52.Data("Expression")

517     po2on.po2com11.value = pop52v
518     po2on.po2com11.AddItem "TRIA ( 36, 40, 56 )", 0
519     po2on.po2com11.AddItem "TRIA ( Min, Mode, Max )", 1
520     po2on.po2com11.AddItem "NORM ( Mean, StdDev )", 2
521     po2on.po2com11.AddItem "EXPO ( Mean )", 3
522     po2on.po2com11.AddItem "UNIF ( Min, Max )", 4

523     Dim pop64 As Module
524     Dim pop64i As Long
525     Dim pop64v As String
526     pop64i = m.Modules.Find(smFindTag, "pop64")
527     Set pop64 = m.Modules(pop64i)
528     pop64v = pop64.Data("Expression")

529     po2on.po2com13.value = pop64v
530     po2on.po2com13.AddItem "TRIA ( 27, 30, 42 )", 0
531     po2on.po2com13.AddItem "TRIA ( Min, Mode, Max )", 1
532     po2on.po2com13.AddItem "NORM ( Mean, StdDev )", 2
533     po2on.po2com13.AddItem "EXPO ( Mean )", 3
534     po2on.po2com13.AddItem "UNIF ( Min, Max )", 4

535     po2on.po2com37.value = pop64v
536     po2on.po2com37.AddItem "TRIA ( 27, 30, 42 )", 0
537     po2on.po2com37.AddItem "TRIA ( Min, Mode, Max )", 1
538     po2on.po2com37.AddItem "NORM ( Mean, StdDev )", 2
539     po2on.po2com37.AddItem "EXPO ( Mean )", 3
540     po2on.po2com37.AddItem "UNIF ( Min, Max )", 4

541     Dim pop53 As Module
542     Dim pop53i As Long
543     Dim pop53v As String
544     pop53i = m.Modules.Find(smFindTag, "pop53")
545     Set pop53 = m.Modules(pop53i)
546     pop53v = pop53.Data("Expression")

547     po2on.po2com15.value = pop53v
548     po2on.po2com15.AddItem "TRIA ( 54, 60, 84 )", 0
549     po2on.po2com15.AddItem "TRIA ( Min, Mode, Max )", 1
550     po2on.po2com15.AddItem "NORM ( Mean, StdDev )", 2
551     po2on.po2com15.AddItem "EXPO ( Mean )", 3
552     po2on.po2com15.AddItem "UNIF ( Min, Max )", 4

553     Dim pop54 As Module
554     Dim pop54i As Long
555     Dim pop54v As String
556     pop54i = m.Modules.Find(smFindTag, "pop54")
```

```
    Set pop54 = m.Modules(pop54i)
    pop54v = pop54.Data("Expression")

    po2on.po2com17.value = pop54v
    po2on.po2com17.AddItem "TRIA ( 108, 120, 168 )", 0
    po2on.po2com17.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com17.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com17.AddItem "EXPO ( Mean )", 3
    po2on.po2com17.AddItem "UNIF ( Min, Max )", 4

    Dim pop55 As Module
    Dim pop55i As Long
    Dim pop55v As String
    pop55i = m.Modules.Find(smFindTag, "pop55")
    Set pop55 = m.Modules(pop55i)
    pop55v = pop55.Data("Expression")

    po2on.po2com19.value = pop55v
    po2on.po2com19.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com19.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com19.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com19.AddItem "EXPO ( Mean )", 3
    po2on.po2com19.AddItem "UNIF ( Min, Max )", 4

    Dim pop56 As Module
    Dim pop56i As Long
    Dim pop56v As String
    pop56i = m.Modules.Find(smFindTag, "pop56")
    Set pop56 = m.Modules(pop56i)
    pop56v = pop56.Data("Expression")

    po2on.po2com21.value = pop56v
    po2on.po2com21.AddItem "TRIA ( 81, 90, 126 )", 0
    po2on.po2com21.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com21.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com21.AddItem "EXPO ( Mean )", 3
    po2on.po2com21.AddItem "UNIF ( Min, Max )", 4

    Dim pop57 As Module
    Dim pop57i As Long
    Dim pop57v As String
    pop57i = m.Modules.Find(smFindTag, "pop57")
    Set pop57 = m.Modules(pop57i)
    pop57v = pop57.Data("Expression")

    po2on.po2com23.value = pop57v
    po2on.po2com23.AddItem "TRIA ( 36, 40, 56 )", 0
    po2on.po2com23.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com23.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com23.AddItem "EXPO ( Mean )", 3
    po2on.po2com23.AddItem "UNIF ( Min, Max )", 4

    Dim pop58 As Module
    Dim pop58i As Long
    Dim pop58v As String
    pop58i = m.Modules.Find(smFindTag, "pop58")
    Set pop58 = m.Modules(pop58i)
    pop58v = pop58.Data("Expression")

    po2on.po2com25.value = pop58v
    po2on.po2com25.AddItem "TRIA ( 36, 40, 56 )", 0
    po2on.po2com25.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com25.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com25.AddItem "EXPO ( Mean )", 3
    po2on.po2com25.AddItem "UNIF ( Min, Max )", 4

    Dim pop59 As Module
    Dim pop59i As Long
    Dim pop59v As String
```

```
    pop59i = m.Modules.Find(smFindTag, "pop59")
    Set pop59 = m.Modules(pop59i)
    pop59v = pop59.Data("Expression")

    po2on.po2com27.value = pop59v
    po2on.po2com27.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com27.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com27.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com27.AddItem "EXPO ( Mean )", 3
    po2on.po2com27.AddItem "UNIF ( Min, Max )", 4

    Dim pop60 As Module
    Dim pop60i As Long
    Dim pop60v As String
    pop60i = m.Modules.Find(smFindTag, "pop60")
    Set pop60 = m.Modules(pop60i)
    pop60v = pop60.Data("Expression")

    po2on.po2com29.value = pop60v
    po2on.po2com29.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com29.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com29.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com29.AddItem "EXPO ( Mean )", 3
    po2on.po2com29.AddItem "UNIF ( Min, Max )", 4

    Dim pop61 As Module
    Dim pop61i As Long
    Dim pop61v As String
    pop61i = m.Modules.Find(smFindTag, "pop61")
    Set pop61 = m.Modules(pop61i)
    pop61v = pop61.Data("Expression")

    po2on.po2com31.value = pop61v
    po2on.po2com31.AddItem "TRIA ( 81, 90, 126 )", 0
    po2on.po2com31.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com31.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com31.AddItem "EXPO ( Mean )", 3
    po2on.po2com31.AddItem "UNIF ( Min, Max )", 4

    Dim pop62 As Module
    Dim pop62i As Long
    Dim pop62v As String
    pop62i = m.Modules.Find(smFindTag, "pop62")
    Set pop62 = m.Modules(pop62i)
    pop62v = pop62.Data("Expression")

    po2on.po2com33.value = pop62v
    po2on.po2com33.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com33.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com33.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com33.AddItem "EXPO ( Mean )", 3
    po2on.po2com33.AddItem "UNIF ( Min, Max )", 4

    Dim pop63 As Module
    Dim pop63i As Long
    Dim pop63v As String
    pop63i = m.Modules.Find(smFindTag, "pop63")
    Set pop63 = m.Modules(pop63i)
    pop63v = pop63.Data("Expression")

    po2on.po2com35.value = pop63v
    po2on.po2com35.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com35.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com35.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com35.AddItem "EXPO ( Mean )", 3
    po2on.po2com35.AddItem "UNIF ( Min, Max )", 4

    Dim pop34 As Module
    Dim pop34i As Long
```

```
675     Dim pop34v As String
676     pop34i = m.Modules.Find(smFindTag, "pop34")
677     Set pop34 = m.Modules(pop34i)
678     pop34v = pop34.Data("Expression")

679     po2on.po2com39.value = pop34v
680     po2on.po2com39.AddItem "TRIA ( 756, 840, 1176 )", 0
681     po2on.po2com39.AddItem "TRIA ( Min, Mode, Max )", 1
682     po2on.po2com39.AddItem "NORM ( Mean, StdDev )", 2
683     po2on.po2com39.AddItem "EXPO ( Mean )", 3
684     po2on.po2com39.AddItem "UNIF ( Min, Max )", 4

685     Dim pop35 As Module
686     Dim pop35i As Long
687     Dim pop35v As String
688     pop35i = m.Modules.Find(smFindTag, "pop35")
689     Set pop35 = m.Modules(pop35i)
690     pop35v = pop35.Data("Expression")

691     po2on.po2com41.value = pop35v
692     po2on.po2com41.AddItem "TRIA ( 324, 360, 504 )", 0
693     po2on.po2com41.AddItem "TRIA ( Min, Mode, Max )", 1
694     po2on.po2com41.AddItem "NORM ( Mean, StdDev )", 2
695     po2on.po2com41.AddItem "EXPO ( Mean )", 3
696     po2on.po2com41.AddItem "UNIF ( Min, Max )", 4

697     'Code below populates small combo boxes for OP-01 thru OP-25
698     Dim pop47u As Module
699     Dim pop47ui As Long
700     Dim pop47uv As String
701     pop47ui = m.Modules.Find(smFindTag, "pop47")
702     Set pop47u = m.Modules(pop47ui)
703     pop47uv = pop47u.Data("Units")

704     po2on.po2com2.value = pop47uv
705     po2on.po2com2.AddItem "Seconds", 0
706     po2on.po2com2.AddItem "Minutes", 1
707     po2on.po2com2.AddItem "Hours", 2
708     po2on.po2com2.AddItem "Days", 3

709     Dim pop48u As Module
710     Dim pop48ui As Long
711     Dim pop48uv As String
712     pop48ui = m.Modules.Find(smFindTag, "pop48")
713     Set pop48u = m.Modules(pop48ui)
714     pop48uv = pop48u.Data("Units")

715     po2on.po2com4.value = pop48uv
716     po2on.po2com4.AddItem "Seconds", 0
717     po2on.po2com4.AddItem "Minutes", 1
718     po2on.po2com4.AddItem "Hours", 2
719     po2on.po2com4.AddItem "Days", 3

720     Dim pop49u As Module
721     Dim pop49ui As Long
722     Dim pop49uv As String
723     pop49ui = m.Modules.Find(smFindTag, "pop49")
724     Set pop49u = m.Modules(pop49ui)
725     pop49uv = pop49u.Data("Units")

726     po2on.po2com6.value = pop49uv
727     po2on.po2com6.AddItem "Seconds", 0
728     po2on.po2com6.AddItem "Minutes", 1
729     po2on.po2com6.AddItem "Hours", 2
730     po2on.po2com6.AddItem "Days", 3

731     Dim pop50u As Module
732     Dim pop50ui As Long
733     Dim pop50uv As String
```

```
734     pop50ui = m.Modules.Find(smFindTag, "pop50")
735     Set pop50u = m.Modules(pop50ui)
736     pop50uv = pop50u.Data("Units")

737     po2on.po2com8.value = pop50uv
738     po2on.po2com8.AddItem "Seconds", 0
739     po2on.po2com8.AddItem "Minutes", 1
740     po2on.po2com8.AddItem "Hours", 2
741     po2on.po2com8.AddItem "Days", 3

742     Dim pop51u As Module
743     Dim pop51ui As Long
744     Dim pop51uv As String
745     pop51ui = m.Modules.Find(smFindTag, "pop51")
746     Set pop51u = m.Modules(pop51ui)
747     pop51uv = pop51u.Data("Units")

748     po2on.po2com10.value = pop51uv
749     po2on.po2com10.AddItem "Seconds", 0
750     po2on.po2com10.AddItem "Minutes", 1
751     po2on.po2com10.AddItem "Hours", 2
752     po2on.po2com10.AddItem "Days", 3

753     Dim pop52u As Module
754     Dim pop52ui As Long
755     Dim pop52uv As String
756     pop52ui = m.Modules.Find(smFindTag, "pop52")
757     Set pop52u = m.Modules(pop52ui)
758     pop52uv = pop52u.Data("Units")

759     po2on.po2com12.value = pop52uv
760     po2on.po2com12.AddItem "Seconds", 0
761     po2on.po2com12.AddItem "Minutes", 1
762     po2on.po2com12.AddItem "Hours", 2
763     po2on.po2com12.AddItem "Days", 3

764     Dim pop64u As Module
765     Dim pop64ui As Long
766     Dim pop64uv As String
767     pop64ui = m.Modules.Find(smFindTag, "pop64")
768     Set pop64u = m.Modules(pop64ui)
769     pop64uv = pop64u.Data("Units")

770     po2on.po2com14.value = pop64uv
771     po2on.po2com14.AddItem "Seconds", 0
772     po2on.po2com14.AddItem "Minutes", 1
773     po2on.po2com14.AddItem "Hours", 2
774     po2on.po2com14.AddItem "Days", 3

775     po2on.po2com38.value = pop64uv
776     po2on.po2com38.AddItem "Seconds", 0
777     po2on.po2com38.AddItem "Minutes", 1
778     po2on.po2com38.AddItem "Hours", 2
779     po2on.po2com38.AddItem "Days", 3

780     Dim pop53u As Module
781     Dim pop53ui As Long
782     Dim pop53uv As String
783     pop53ui = m.Modules.Find(smFindTag, "pop53")
784     Set pop53u = m.Modules(pop53ui)
785     pop53uv = pop53u.Data("Units")

786     po2on.po2com16.value = pop53uv
787     po2on.po2com16.AddItem "Seconds", 0
788     po2on.po2com16.AddItem "Minutes", 1
789     po2on.po2com16.AddItem "Hours", 2
790     po2on.po2com16.AddItem "Days", 3

791     Dim pop54u As Module
```

```
792     Dim pop54ui As Long
793     Dim pop54uv As String
794     pop54ui = m.Modules.Find(smFindTag, "pop54")
795     Set pop54u = m.Modules(pop54ui)
796     pop54uv = pop54u.Data("Units")

797     po2on.po2com18.value = pop54uv
798     po2on.po2com18.AddItem "Seconds", 0
799     po2on.po2com18.AddItem "Minutes", 1
800     po2on.po2com18.AddItem "Hours", 2
801     po2on.po2com18.AddItem "Days", 3

802     Dim pop55u As Module
803     Dim pop55ui As Long
804     Dim pop55uv As String
805     pop55ui = m.Modules.Find(smFindTag, "pop55")
806     Set pop55u = m.Modules(pop55ui)
807     pop55uv = pop55u.Data("Units")

808     po2on.po2com20.value = pop55uv
809     po2on.po2com20.AddItem "Seconds", 0
810     po2on.po2com20.AddItem "Minutes", 1
811     po2on.po2com20.AddItem "Hours", 2
812     po2on.po2com20.AddItem "Days", 3

813     Dim pop56u As Module
814     Dim pop56ui As Long
815     Dim pop56uv As String
816     pop56ui = m.Modules.Find(smFindTag, "pop56")
817     Set pop56u = m.Modules(pop56ui)
818     pop56uv = pop56u.Data("Units")

819     po2on.po2com22.value = pop56uv
820     po2on.po2com22.AddItem "Seconds", 0
821     po2on.po2com22.AddItem "Minutes", 1
822     po2on.po2com22.AddItem "Hours", 2
823     po2on.po2com22.AddItem "Days", 3

824     Dim pop57u As Module
825     Dim pop57ui As Long
826     Dim pop57uv As String
827     pop57ui = m.Modules.Find(smFindTag, "pop57")
828     Set pop57u = m.Modules(pop57ui)
829     pop57uv = pop57u.Data("Units")

830     po2on.po2com24.value = pop57uv
831     po2on.po2com24.AddItem "Seconds", 0
832     po2on.po2com24.AddItem "Minutes", 1
833     po2on.po2com24.AddItem "Hours", 2
834     po2on.po2com24.AddItem "Days", 3

835     Dim pop58u As Module
836     Dim pop58ui As Long
837     Dim pop58uv As String
838     pop58ui = m.Modules.Find(smFindTag, "pop58")
839     Set pop58u = m.Modules(pop58ui)
840     pop58uv = pop58u.Data("Units")

841     po2on.po2com26.value = pop58uv
842     po2on.po2com26.AddItem "Seconds", 0
843     po2on.po2com26.AddItem "Minutes", 1
844     po2on.po2com26.AddItem "Hours", 2
845     po2on.po2com26.AddItem "Days", 3

846     Dim pop59u As Module
847     Dim pop59ui As Long
848     Dim pop59uv As String
849     pop59ui = m.Modules.Find(smFindTag, "pop59")
850     Set pop59u = m.Modules(pop59ui)
```

```
851     pop59uv = pop59u.Data("Units")

852     po2on.po2com28.value = pop59uv
853     po2on.po2com28.AddItem "Seconds", 0
854     po2on.po2com28.AddItem "Minutes", 1
855     po2on.po2com28.AddItem "Hours", 2
856     po2on.po2com28.AddItem "Days", 3

857     Dim pop60u As Module
858     Dim pop60ui As Long
859     Dim pop60uv As String
860     pop60ui = m.Modules.Find(smFindTag, "pop60")
861     Set pop60u = m.Modules(pop60ui)
862     pop60uv = pop60u.Data("Units")

863     po2on.po2com30.value = pop60uv
864     po2on.po2com30.AddItem "Seconds", 0
865     po2on.po2com30.AddItem "Minutes", 1
866     po2on.po2com30.AddItem "Hours", 2
867     po2on.po2com30.AddItem "Days", 3

868     Dim pop61u As Module
869     Dim pop61ui As Long
870     Dim pop61uv As String
871     pop61ui = m.Modules.Find(smFindTag, "pop61")
872     Set pop61u = m.Modules(pop61ui)
873     pop61uv = pop61u.Data("Units")

874     po2on.po2com32.value = pop61uv
875     po2on.po2com32.AddItem "Seconds", 0
876     po2on.po2com32.AddItem "Minutes", 1
877     po2on.po2com32.AddItem "Hours", 2
878     po2on.po2com32.AddItem "Days", 3

879     Dim pop62u As Module
880     Dim pop62ui As Long
881     Dim pop62uv As String
882     pop62ui = m.Modules.Find(smFindTag, "pop62")
883     Set pop62u = m.Modules(pop62ui)
884     pop62uv = pop62u.Data("Units")

885     po2on.po2com34.value = pop62uv
886     po2on.po2com34.AddItem "Seconds", 0
887     po2on.po2com34.AddItem "Minutes", 1
888     po2on.po2com34.AddItem "Hours", 2
889     po2on.po2com34.AddItem "Days", 3

890     Dim pop63u As Module
891     Dim pop63ui As Long
892     Dim pop63uv As String
893     pop63ui = m.Modules.Find(smFindTag, "pop63")
894     Set pop63u = m.Modules(pop63ui)
895     pop63uv = pop63u.Data("Units")

896     po2on.po2com36.value = pop63uv
897     po2on.po2com36.AddItem "Seconds", 0
898     po2on.po2com36.AddItem "Minutes", 1
899     po2on.po2com36.AddItem "Hours", 2
900     po2on.po2com36.AddItem "Days", 3

901     Dim pop34u As Module
902     Dim pop34ui As Long
903     Dim pop34uv As String
904     pop34ui = m.Modules.Find(smFindTag, "pop34")
905     Set pop34u = m.Modules(pop34ui)
906     pop34uv = pop34u.Data("Units")

907     po2on.po2com40.value = pop34uv
908     po2on.po2com40.AddItem "Seconds", 0
```

```
    po2on.po2com40.AddItem "Minutes", 1
    po2on.po2com40.AddItem "Hours", 2
    po2on.po2com40.AddItem "Days", 3

    Dim pop35u As Module
    Dim pop35ui As Long
    Dim pop35uv As String
    pop35ui = m.Modules.Find(smFindTag, "pop35")
    Set pop35u = m.Modules(pop35ui)
    pop35uv = pop35u.Data("Units")

    po2on.po2com42.value = pop35uv
    po2on.po2com42.AddItem "Seconds", 0
    po2on.po2com42.AddItem "Minutes", 1
    po2on.po2com42.AddItem "Hours", 2
    po2on.po2com42.AddItem "Days", 3

  End Sub
```

Project/po1prelim

```
  Private Sub CommandButton3_Click()
    Me.Hide
    motors.Show
  End Sub

  Private Sub CommandButton4_Click()
    Hierarchy.done03.Visible = True

    'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
    Dim msgResult As Integer
    If (po1opt1.value = False And po1opt2.value = False) Then
      msgResult = MsgBox("You must make a preintegration choice.  Will the 2nd
stage and payload be preintegrated?", vbYesNo)
      If msgResult = vbYes Then
        po1opt1.value = True
      Else
        po1opt2.value = True
      End If
    End If
    If (po1opt3.value = False And po1opt4.value = False) Then
      msgResult = MsgBox("You must make an integration location decision.  Click
Yes for stage 1 and stage 2 integration on the launch pad.  Click No for stage 1 and
stage 2 integration off the launch pad.", vbYesNo)
      If msgResult = vbYes Then
        po1opt3.value = True
      Else
        po1opt4.value = True
      End If
    End If
    If (po1opt4.value = True And po1opt5.value = False And po1opt6.value = False)
Then
      msgResult = MsgBox("You must make an off-pad integration location decision.
Click Yes if integration will take place in the maintenance bay.  Click No if integration
will take place in a separate integration facility.", vbYesNo)
      If msgResult = vbYes Then
        po1opt5.value = True
      Else
        po1opt6.value = True
      End If
    End If

    'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10

    Dim m As Model
    Set m = ThisDocument.Model
```

```
36      Dim pop1 As Module
37      Dim pop1i As Long
38      pop1i = m.Modules.Find(smFindTag, "pop1")
39      Set pop1 = m.Modules(pop1i)
40      pop1.Data("Expression") = po1com1.Text
41      pop1.Data("Units") = po1com2.Text

42      Dim pop2 As Module
43      Dim pop2i As Long
44      pop2i = m.Modules.Find(smFindTag, "pop2")
45      Set pop2 = m.Modules(pop2i)
46      pop2.Data("Expression") = po1com3.Text
47      pop2.Data("Units") = po1com4.Text

48      Dim pop3 As Module
49      Dim pop3i As Long
50      pop3i = m.Modules.Find(smFindTag, "pop3")
51      Set pop3 = m.Modules(pop3i)
52      pop3.Data("Expression") = po1com5.Text
53      pop3.Data("Units") = po1com6.Text

54      Dim pop4 As Module
55      Dim pop4i As Long
56      pop4i = m.Modules.Find(smFindTag, "pop4")
57      Set pop4 = m.Modules(pop4i)
58      pop4.Data("Expression") = po1com7.Text
59      pop4.Data("Units") = po1com8.Text

60      Dim pop5 As Module
61      Dim pop5i As Long
62      pop5i = m.Modules.Find(smFindTag, "pop5")
63      Set pop5 = m.Modules(pop5i)
64      pop5.Data("Expression") = po1com9.Text
65      pop5.Data("Units") = po1com10.Text

66      Dim pop6 As Module
67      Dim pop6i As Long
68      pop6i = m.Modules.Find(smFindTag, "pop6")
69      Set pop6 = m.Modules(pop6i)
70      pop6.Data("Expression") = po1com11.Text
71      pop6.Data("Units") = po1com12.Text

72      Dim pop7 As Module
73      Dim pop7i As Long
74      pop7i = m.Modules.Find(smFindTag, "pop7")
75      Set pop7 = m.Modules(pop7i)
76      pop7.Data("Expression") = po1com13.Text
77      pop7.Data("Units") = po1com14.Text

78      'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
79      Dim pov1 As Module
80      Dim pov1i As Long
81      pov1i = m.Modules.Find(smFindTag, "pov1")
82      Set pov1 = m.Modules(pov1i)
83      If po1opt1.value = True Then
84        pov1.Data("Initial Value") = "1"
85      Else
86        pov1.Data("Initial Value") = "0"
87      End If

88      Dim pov2 As Module
89      Dim pov2i As Long
90      pov2i = m.Modules.Find(smFindTag, "pov2")
91      Set pov2 = m.Modules(pov2i)
92      If po1opt3.value = True Then
93        pov2.Data("Initial Value") = "1"
94      Else
```

```
95        pov2.Data("Initial Value") = "0"
96      End If

97      Dim pov3 As Module
98      Dim pov3i As Long
99      pov3i = m.Modules.Find(smFindTag, "pov3")
100     Set pov3 = m.Modules(pov3i)
101     If po1opt5.value = True Then
102       pov3.Data("Initial Value") = "1"
103     Else
104       pov3.Data("Initial Value") = "0"
105     End If

106     'Code below checks to see which form to show next and then shows the
appropriate form
107     Me.Hide
108     If po1opt3.value = True Then
109       po2on.Show
110     ElseIf po1opt4.value = True And po1opt1.value = True Then
111       po3offpreint.Show
112     Else
113       po4offnopreint.Show
114     End If

115   End Sub

116   Private Sub CommandButton5_Click()

117   End Sub

118   Private Sub CommandButton6_Click()
119     Hierarchy.done03.Visible = True

120     'The following code checks to see if the user forgot to click any option
buttons and then displays message boxes forcing the user to make a choice on decisions
they skipped in the form
121     Dim msgResult As Integer
122     If (po1opt1.value = False And po1opt2.value = False) Then
123       msgResult = MsgBox("You must make a preintegration choice.  Will the 2nd
stage and payload be preintegrated?", vbYesNo)
124       If msgResult = vbYes Then
125         po1opt1.value = True
126       Else
127         po1opt2.value = True
128       End If
129     End If
130     If (po1opt3.value = False And po1opt4.value = False) Then
131       msgResult = MsgBox("You must make an integration location decision.  Click
Yes for stage 1 and stage 2 integration on the launch pad.  Click No for stage 1 and
stage 2 integration off the launch pad.", vbYesNo)
132       If msgResult = vbYes Then
133         po1opt3.value = True
134       Else
135         po1opt4.value = True
136       End If
137     End If
138     If (po1opt4.value = True And po1opt5.value = False And po1opt6.value = False)
Then
139       msgResult = MsgBox("You must make an off-pad integration location decision.
Click Yes if integration will take place in the maintenance bay.  Click No if integration
will take place in a separate integration facility.", vbYesNo)
140       If msgResult = vbYes Then
141         po1opt5.value = True
142       Else
143         po1opt6.value = True
144       End If
145     End If

146     'Code below populates the appropriate arena modules with the distributions the
```

```
user put into the combo boxes for PI-02 thru PI-10

    Dim m As Model
    Set m = ThisDocument.Model

    Dim pop1 As Module
    Dim pop1i As Long
    pop1i = m.Modules.Find(smFindTag, "pop1")
    Set pop1 = m.Modules(pop1i)
    pop1.Data("Expression") = po1com1.Text
    pop1.Data("Units") = po1com2.Text

    Dim pop2 As Module
    Dim pop2i As Long
    pop2i = m.Modules.Find(smFindTag, "pop2")
    Set pop2 = m.Modules(pop2i)
    pop2.Data("Expression") = po1com3.Text
    pop2.Data("Units") = po1com4.Text

    Dim pop3 As Module
    Dim pop3i As Long
    pop3i = m.Modules.Find(smFindTag, "pop3")
    Set pop3 = m.Modules(pop3i)
    pop3.Data("Expression") = po1com5.Text
    pop3.Data("Units") = po1com6.Text

    Dim pop4 As Module
    Dim pop4i As Long
    pop4i = m.Modules.Find(smFindTag, "pop4")
    Set pop4 = m.Modules(pop4i)
    pop4.Data("Expression") = po1com7.Text
    pop4.Data("Units") = po1com8.Text

    Dim pop5 As Module
    Dim pop5i As Long
    pop5i = m.Modules.Find(smFindTag, "pop5")
    Set pop5 = m.Modules(pop5i)
    pop5.Data("Expression") = po1com9.Text
    pop5.Data("Units") = po1com10.Text

    Dim pop6 As Module
    Dim pop6i As Long
    pop6i = m.Modules.Find(smFindTag, "pop6")
    Set pop6 = m.Modules(pop6i)
    pop6.Data("Expression") = po1com11.Text
    pop6.Data("Units") = po1com12.Text

    Dim pop7 As Module
    Dim pop7i As Long
    pop7i = m.Modules.Find(smFindTag, "pop7")
    Set pop7 = m.Modules(pop7i)
    pop7.Data("Expression") = po1com13.Text
    pop7.Data("Units") = po1com14.Text

    'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
    Dim pov1 As Module
    Dim pov1i As Long
    pov1i = m.Modules.Find(smFindTag, "pov1")
    Set pov1 = m.Modules(pov1i)
    If po1opt1.value = True Then
      pov1.Data("Initial Value") = "1"
    Else
      pov1.Data("Initial Value") = "0"
    End If

    Dim pov2 As Module
    Dim pov2i As Long
    pov2i = m.Modules.Find(smFindTag, "pov2")
```

```
204     Set pov2 = m.Modules(pov2i)
205     If po1opt3.value = True Then
206       pov2.Data("Initial Value") = "1"
207     Else
208       pov2.Data("Initial Value") = "0"
209     End If

210     Dim pov3 As Module
211     Dim pov3i As Long
212     pov3i = m.Modules.Find(smFindTag, "pov3")
213     Set pov3 = m.Modules(pov3i)
214     If po1opt5.value = True Then
215       pov3.Data("Initial Value") = "1"
216     Else
217       pov3.Data("Initial Value") = "0"
218     End If

219     Me.Hide
220     Hierarchy.Show
221   End Sub

222   Private Sub Label11_Click()

223   End Sub

224   Private Sub Label12_Click()

225   End Sub

226   Private Sub OptionButton1_Click()

227   End Sub

228   Private Sub OptionButton2_Click()

229   End Sub

230   Private Sub OptionButton4_Click()

231   End Sub

232   Private Sub OptionButton6_Click()

233   End Sub

234   Private Sub po1com1_Change()

235   End Sub

236   Private Sub po1opt1_Click()
237     po1frm1.Visible = True
238     po2on.po2frm1.Visible = True
239     po2on.po2frm2.Visible = False
240     po6erect.po6frm3.Visible = False

241   End Sub

242   Private Sub po1opt2_Click()
243     po1frm1.Visible = False
244     po2on.po2frm1.Visible = False
245     po2on.po2frm2.Visible = True
246     po6erect.po6frm3.Visible = True

247   End Sub

248   Private Sub po1opt3_Click()
249     po1frm2.Visible = True
250     po1frm3.Visible = False
251     po1frm4.Visible = False
```

```
252   End Sub

253   Private Sub po1opt4_Click()
254     po1frm2.Visible = False
255     po1frm3.Visible = True
256     po1frm4.Visible = True
257   End Sub

258   Private Sub po1opt5_Click()
259     po1frm4.Visible = False

260   End Sub

261   Private Sub po1opt6_Click()
262     po1frm4.Visible = True

263   End Sub

264   Private Sub TextBox1_Change()

265   End Sub

266   Private Sub ToggleButton1_Click()

267   End Sub

268   Private Sub UserForm_Click()

269   End Sub

270   Private Sub UserForm_Initialize()
271     Dim m As Model
272     Set m = ThisDocument.Model

273     'Code below populates large combo boxes for PI-02 thru PI-06 and PI-08 and PI-
10
274     Dim pop1 As Module
275     Dim pop1i As Long
276     Dim pop1v As String
277     pop1i = m.Modules.Find(smFindTag, "pop1")
278     Set pop1 = m.Modules(pop1i)
279     pop1v = pop1.Data("Expression")

280     po1prelim.po1com1.value = pop1v
281     po1prelim.po1com1.AddItem "TRIA ( 27, 30, 42 )", 0
282     po1prelim.po1com1.AddItem "TRIA ( Min, Mode, Max )", 1
283     po1prelim.po1com1.AddItem "NORM ( Mean, StdDev )", 2
284     po1prelim.po1com1.AddItem "EXPO ( Mean )", 3
285     po1prelim.po1com1.AddItem "UNIF ( Min, Max )", 4

286     Dim pop2 As Module
287     Dim pop2i As Long
288     Dim pop2v As String
289     pop2i = m.Modules.Find(smFindTag, "pop2")
290     Set pop2 = m.Modules(pop2i)
291     pop2v = pop2.Data("Expression")

292     po1prelim.po1com3.value = pop2v
293     po1prelim.po1com3.AddItem "TRIA ( 27, 30, 42 )", 0
294     po1prelim.po1com3.AddItem "TRIA ( Min, Mode, Max )", 1
295     po1prelim.po1com3.AddItem "NORM ( Mean, StdDev )", 2
296     po1prelim.po1com3.AddItem "EXPO ( Mean )", 3
297     po1prelim.po1com3.AddItem "UNIF ( Min, Max )", 4

298     Dim pop3 As Module
299     Dim pop3i As Long
300     Dim pop3v As String
301     pop3i = m.Modules.Find(smFindTag, "pop3")
```

```
302     Set pop3 = m.Modules(pop3i)
303     pop3v = pop3.Data("Expression")

304     po1prelim.po1com5.value = pop3v
305     po1prelim.po1com5.AddItem "TRIA ( 18, 20, 28 )", 0
306     po1prelim.po1com5.AddItem "TRIA ( Min, Mode, Max )", 1
307     po1prelim.po1com5.AddItem "NORM ( Mean, StdDev )", 2
308     po1prelim.po1com5.AddItem "EXPO ( Mean )", 3
309     po1prelim.po1com5.AddItem "UNIF ( Min, Max )", 4

310     Dim pop4 As Module
311     Dim pop4i As Long
312     Dim pop4v As String
313     pop4i = m.Modules.Find(smFindTag, "pop4")
314     Set pop4 = m.Modules(pop4i)
315     pop4v = pop4.Data("Expression")

316     po1prelim.po1com7.value = pop4v
317     po1prelim.po1com7.AddItem "TRIA ( 18, 20, 28 )", 0
318     po1prelim.po1com7.AddItem "TRIA ( Min, Mode, Max )", 1
319     po1prelim.po1com7.AddItem "NORM ( Mean, StdDev )", 2
320     po1prelim.po1com7.AddItem "EXPO ( Mean )", 3
321     po1prelim.po1com7.AddItem "UNIF ( Min, Max )", 4

322     Dim pop5 As Module
323     Dim pop5i As Long
324     Dim pop5v As String
325     pop5i = m.Modules.Find(smFindTag, "pop5")
326     Set pop5 = m.Modules(pop5i)
327     pop5v = pop5.Data("Expression")

328     po1prelim.po1com9.value = pop5v
329     po1prelim.po1com9.AddItem "TRIA ( 27, 30, 42 )", 0
330     po1prelim.po1com9.AddItem "TRIA ( Min, Mode, Max )", 1
331     po1prelim.po1com9.AddItem "NORM ( Mean, StdDev )", 2
332     po1prelim.po1com9.AddItem "EXPO ( Mean )", 3
333     po1prelim.po1com9.AddItem "UNIF ( Min, Max )", 4

334     Dim pop6 As Module
335     Dim pop6i As Long
336     Dim pop6v As String
337     pop6i = m.Modules.Find(smFindTag, "pop6")
338     Set pop6 = m.Modules(pop6i)
339     pop6v = pop6.Data("Expression")

340     po1prelim.po1com11.value = pop6v
341     po1prelim.po1com11.AddItem "TRIA ( 27, 30, 42 )", 0
342     po1prelim.po1com11.AddItem "TRIA ( Min, Mode, Max )", 1
343     po1prelim.po1com11.AddItem "NORM ( Mean, StdDev )", 2
344     po1prelim.po1com11.AddItem "EXPO ( Mean )", 3
345     po1prelim.po1com11.AddItem "UNIF ( Min, Max )", 4

346     Dim pop7 As Module
347     Dim pop7i As Long
348     Dim pop7v As String
349     pop7i = m.Modules.Find(smFindTag, "pop7")
350     Set pop7 = m.Modules(pop7i)
351     pop7v = pop7.Data("Expression")

352     po1prelim.po1com13.value = pop7v
353     po1prelim.po1com13.AddItem "TRIA ( 13.5, 15, 21 )", 0
354     po1prelim.po1com13.AddItem "TRIA ( Min, Mode, Max )", 1
355     po1prelim.po1com13.AddItem "NORM ( Mean, StdDev )", 2
356     po1prelim.po1com13.AddItem "EXPO ( Mean )", 3
357     po1prelim.po1com13.AddItem "UNIF ( Min, Max )", 4

358     'Code below populates small combo boxes for PI-02 thru PI-06 and PI-08 and PI-
10
359     Dim pop1u As Module
```

```
360     Dim pop1ui As Long
361     Dim pop1uv As String
362     pop1ui = m.Modules.Find(smFindTag, "pop1")
363     Set pop1u = m.Modules(pop1ui)
364     pop1uv = pop1u.Data("Units")

365     po1prelim.po1com2.value = pop1uv
366     po1prelim.po1com2.AddItem "Seconds", 0
367     po1prelim.po1com2.AddItem "Minutes", 1
368     po1prelim.po1com2.AddItem "Hours", 2
369     po1prelim.po1com2.AddItem "Days", 3

370     Dim pop2u As Module
371     Dim pop2ui As Long
372     Dim pop2uv As String
373     pop2ui = m.Modules.Find(smFindTag, "pop2")
374     Set pop2u = m.Modules(pop2ui)
375     pop2uv = pop2u.Data("Units")

376     po1prelim.po1com4.value = pop2uv
377     po1prelim.po1com4.AddItem "Seconds", 0
378     po1prelim.po1com4.AddItem "Minutes", 1
379     po1prelim.po1com4.AddItem "Hours", 2
380     po1prelim.po1com4.AddItem "Days", 3

381     Dim pop3u As Module
382     Dim pop3ui As Long
383     Dim pop3uv As String
384     pop3ui = m.Modules.Find(smFindTag, "pop3")
385     Set pop3u = m.Modules(pop3ui)
386     pop3uv = pop3u.Data("Units")

387     po1prelim.po1com6.value = pop3uv
388     po1prelim.po1com6.AddItem "Seconds", 0
389     po1prelim.po1com6.AddItem "Minutes", 1
390     po1prelim.po1com6.AddItem "Hours", 2
391     po1prelim.po1com6.AddItem "Days", 3

392     Dim pop4u As Module
393     Dim pop4ui As Long
394     Dim pop4uv As String
395     pop4ui = m.Modules.Find(smFindTag, "pop4")
396     Set pop4u = m.Modules(pop4ui)
397     pop4uv = pop4u.Data("Units")

398     po1prelim.po1com8.value = pop4uv
399     po1prelim.po1com8.AddItem "Seconds", 0
400     po1prelim.po1com8.AddItem "Minutes", 1
401     po1prelim.po1com8.AddItem "Hours", 2
402     po1prelim.po1com8.AddItem "Days", 3

403     Dim pop5u As Module
404     Dim pop5ui As Long
405     Dim pop5uv As String
406     pop5ui = m.Modules.Find(smFindTag, "pop5")
407     Set pop5u = m.Modules(pop5ui)
408     pop5uv = pop5u.Data("Units")

409     po1prelim.po1com10.value = pop5uv
410     po1prelim.po1com10.AddItem "Seconds", 0
411     po1prelim.po1com10.AddItem "Minutes", 1
412     po1prelim.po1com10.AddItem "Hours", 2
413     po1prelim.po1com10.AddItem "Days", 3

414     Dim pop6u As Module
415     Dim pop6ui As Long
416     Dim pop6uv As String
417     pop6ui = m.Modules.Find(smFindTag, "pop6")
418     Set pop6u = m.Modules(pop6ui)
```

```
419     pop6uv = pop6u.Data("Units")

420     po1prelim.po1com12.value = pop6uv
421     po1prelim.po1com12.AddItem "Seconds", 0
422     po1prelim.po1com12.AddItem "Minutes", 1
423     po1prelim.po1com12.AddItem "Hours", 2
424     po1prelim.po1com12.AddItem "Days", 3

425     Dim pop7u As Module
426     Dim pop7ui As Long
427     Dim pop7uv As String
428     pop7ui = m.Modules.Find(smFindTag, "pop7")
429     Set pop7u = m.Modules(pop7ui)
430     pop7uv = pop7u.Data("Units")

431     po1prelim.po1com14.value = pop7uv
432     po1prelim.po1com14.AddItem "Seconds", 0
433     po1prelim.po1com14.AddItem "Minutes", 1
434     po1prelim.po1com14.AddItem "Hours", 2
435     po1prelim.po1com14.AddItem "Days", 3

436   End Sub
```

Project/po2on

```
    1   Private Sub CommandButton3_Click()

    2   End Sub

    3   Private Sub CommandButton6_Click()
    4     Me.Hide
    5     po1prelim.Show
    6   End Sub

    7   Private Sub CommandButton7_Click()
    8     Hierarchy.done04.Visible = True

    9     'Code below checks if any option button sets are not clicked, and if so, forces
the user to make a decision
   10     Dim msgResult As Integer
   11     If (po2opt1.value = False And po2opt2.value = False) Then
   12       msgResult = MsgBox("You must make a hypergolic fuels decision.  Are
hypergolic fuels required?", vbYesNo)
   13       If msgResult = vbYes Then
   14         po2opt1.value = True
   15       Else
   16         po2opt2.value = True
   17       End If
   18     End If
   19     If (po2opt1.value = True And po2opt3.value = False And po2opt4.value = False)
Then
   20       msgResult = MsgBox("You must make a hypergolic fuels loading decision.  Click
Yes if hypergolics are loaded now, in the integration facility.  Click No if hypergolics
are loaded later, on the launch pad.", vbYesNo)
   21       If msgResult = vbYes Then
   22         po2opt3.value = True
   23       Else
   24         po2opt4.value = True
   25       End If
   26     End If
   27     If (po2opt5.value = False And po2opt6.value = False) Then
   28       msgResult = MsgBox("You must make an ordnance decision.  Is ordnance
required?", vbYesNo)
   29       If msgResult = vbYes Then
   30         po2opt5.value = True
   31       Else
   32         po2opt6.value = True
   33       End If
   34     End If
   35     If (po2opt5.value = True And po2opt7.value = False And po2opt8.value = False)
Then
   36       msgResult = MsgBox("You must make an ordnance installation location decision.
Click Yes if ordnance is loaded now, in the integration facility.  Click No if ordnance
is loaded later, on the launch pad.", vbYesNo)
   37       If msgResult = vbYes Then
   38         po2opt7.value = True
   39       Else
   40         po2opt8.value = True
   41       End If
   42     End If

   43     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10
   44     Dim m As Model
   45     Set m = ThisDocument.Model

   46     Dim pop47 As Module
   47     Dim pop47i As Long
   48     pop47i = m.Modules.Find(smFindTag, "pop47")
   49     Set pop47 = m.Modules(pop47i)
   50     pop47.Data("Expression") = po2com1.Text
```

```
51      pop47.Data("Units") = po2com2.Text

52      Dim pop48 As Module
53      Dim pop48i As Long
54      pop48i = m.Modules.Find(smFindTag, "pop48")
55      Set pop48 = m.Modules(pop48i)
56      pop48.Data("Expression") = po2com3.Text
57      pop48.Data("Units") = po2com4.Text

58      Dim pop49 As Module
59      Dim pop49i As Long
60      pop49i = m.Modules.Find(smFindTag, "pop49")
61      Set pop49 = m.Modules(pop49i)
62      pop49.Data("Expression") = po2com5.Text
63      pop49.Data("Units") = po2com6.Text

64      Dim pop50 As Module
65      Dim pop50i As Long
66      pop50i = m.Modules.Find(smFindTag, "pop50")
67      Set pop50 = m.Modules(pop50i)
68      pop50.Data("Expression") = po2com7.Text
69      pop50.Data("Units") = po2com8.Text

70      Dim pop51 As Module
71      Dim pop51i As Long
72      pop51i = m.Modules.Find(smFindTag, "pop51")
73      Set pop51 = m.Modules(pop51i)
74      pop51.Data("Expression") = po2com9.Text
75      pop51.Data("Units") = po2com10.Text

76      Dim pop52 As Module
77      Dim pop52i As Long
78      pop52i = m.Modules.Find(smFindTag, "pop52")
79      Set pop52 = m.Modules(pop52i)
80      pop52.Data("Expression") = po2com11.Text
81      pop52.Data("Units") = po2com12.Text

82      Dim pop64 As Module
83      Dim pop64i As Long
84      pop64i = m.Modules.Find(smFindTag, "pop64")
85      Set pop64 = m.Modules(pop64i)
86      If po2frm1.Visible = True Then
87        pop64.Data("Expression") = po2com13.Text
88        pop64.Data("Units") = po2com14.Text
89      Else
90        pop64.Data("Expression") = po2com37.Text
91        pop64.Data("Units") = po2com38.Text
92      End If

93      Dim pop53 As Module
94      Dim pop53i As Long
95      pop53i = m.Modules.Find(smFindTag, "pop53")
96      Set pop53 = m.Modules(pop53i)
97      pop53.Data("Expression") = po2com15.Text
98      pop53.Data("Units") = po2com16.Text

99      Dim pop54 As Module
100     Dim pop54i As Long
101     pop54i = m.Modules.Find(smFindTag, "pop54")
102     Set pop54 = m.Modules(pop54i)
103     pop54.Data("Expression") = po2com17.Text
104     pop54.Data("Units") = po2com18.Text

105     Dim pop55 As Module
106     Dim pop55i As Long
107     pop55i = m.Modules.Find(smFindTag, "pop55")
108     Set pop55 = m.Modules(pop55i)
109     pop55.Data("Expression") = po2com19.Text
110     pop55.Data("Units") = po2com20.Text
```

```
111     Dim pop56 As Module
112     Dim pop56i As Long
113     pop56i = m.Modules.Find(smFindTag, "pop56")
114     Set pop56 = m.Modules(pop56i)
115     pop56.Data("Expression") = po2com21.Text
116     pop56.Data("Units") = po2com22.Text

117     Dim pop57 As Module
118     Dim pop57i As Long
119     pop57i = m.Modules.Find(smFindTag, "pop57")
120     Set pop57 = m.Modules(pop57i)
121     pop57.Data("Expression") = po2com23.Text
122     pop57.Data("Units") = po2com24.Text

123     Dim pop58 As Module
124     Dim pop58i As Long
125     pop58i = m.Modules.Find(smFindTag, "pop58")
126     Set pop58 = m.Modules(pop58i)
127     pop58.Data("Expression") = po2com25.Text
128     pop58.Data("Units") = po2com26.Text

129     Dim pop59 As Module
130     Dim pop59i As Long
131     pop59i = m.Modules.Find(smFindTag, "pop59")
132     Set pop59 = m.Modules(pop59i)
133     pop59.Data("Expression") = po2com27.Text
134     pop59.Data("Units") = po2com28.Text

135     Dim pop60 As Module
136     Dim pop60i As Long
137     pop60i = m.Modules.Find(smFindTag, "pop60")
138     Set pop60 = m.Modules(pop60i)
139     pop60.Data("Expression") = po2com29.Text
140     pop60.Data("Units") = po2com30.Text

141     Dim pop61 As Module
142     Dim pop61i As Long
143     pop61i = m.Modules.Find(smFindTag, "pop61")
144     Set pop61 = m.Modules(pop61i)
145     pop61.Data("Expression") = po2com31.Text
146     pop61.Data("Units") = po2com32.Text

147     Dim pop62 As Module
148     Dim pop62i As Long
149     pop62i = m.Modules.Find(smFindTag, "pop62")
150     Set pop62 = m.Modules(pop62i)
151     pop62.Data("Expression") = po2com33.Text
152     pop62.Data("Units") = po2com34.Text

153     Dim pop63 As Module
154     Dim pop63i As Long
155     pop63i = m.Modules.Find(smFindTag, "pop63")
156     Set pop63 = m.Modules(pop63i)
157     pop63.Data("Expression") = po2com35.Text
158     pop63.Data("Units") = po2com36.Text

159     Dim pop34 As Module
160     Dim pop34i As Long
161     pop34i = m.Modules.Find(smFindTag, "pop34")
162     Set pop34 = m.Modules(pop34i)
163     pop34.Data("Expression") = po2com39.Text
164     pop34.Data("Units") = po2com40.Text

165     Dim pop71 As Module
166     Dim pop71i As Long
167     pop71i = m.Modules.Find(smFindTag, "pop71")
168     Set pop71 = m.Modules(pop71i)
169     pop71.Data("Expression") = po2com39.Text
```

```
  170     pop71.Data("Units") = po2com40.Text

  171     Dim pop35 As Module
  172     Dim pop35i As Long
  173     pop35i = m.Modules.Find(smFindTag, "pop35")
  174     Set pop35 = m.Modules(pop35i)
  175     pop35.Data("Expression") = po2com41.Text
  176     pop35.Data("Units") = po2com42.Text

  177     Dim pop77 As Module
  178     Dim pop77i As Long
  179     pop77i = m.Modules.Find(smFindTag, "pop77")
  180     Set pop77 = m.Modules(pop77i)
  181     pop77.Data("Expression") = po2com41.Text
  182     pop77.Data("Units") = po2com42.Text

  183     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
  184     Dim pov6 As Module
  185     Dim pov6i As Long
  186     pov6i = m.Modules.Find(smFindTag, "pov6")
  187     Set pov6 = m.Modules(pov6i)
  188     If po2opt2.value = True Then
  189       pov6.Data("Initial Value") = "0"
  190     ElseIf po2opt3.value = True Then
  191       pov6.Data("Initial Value") = "1"
  192     Else
  193       pov6.Data("Initial Value") = "2"
  194     End If

  195     Dim pov7 As Module
  196     Dim pov7i As Long
  197     pov7i = m.Modules.Find(smFindTag, "pov7")
  198     Set pov7 = m.Modules(pov7i)
  199     If po2opt6.value = True Then
  200       pov7.Data("Initial Value") = "0"
  201     ElseIf po2opt7.value = True Then
  202       pov7.Data("Initial Value") = "1"
  203     Else
  204       pov7.Data("Initial Value") = "2"
  205     End If

  206     'code below hides the current form and shows the next form
  207     Me.Hide
  208     po7umbilical.Show

  209   End Sub

  210   Private Sub CommandButton9_Click()
  211     Hierarchy.done04.Visible = True

  212     'Code below checks if any option button sets are not clicked, and if so, forces
the user to make a decision
  213     Dim msgResult As Integer
  214     If (po2opt1.value = False And po2opt2.value = False) Then
  215       msgResult = MsgBox("You must make a hypergolic fuels decision.  Are
hypergolic fuels required?", vbYesNo)
  216       If msgResult = vbYes Then
  217         po2opt1.value = True
  218       Else
  219         po2opt2.value = True
  220       End If
  221     End If
  222     If (po2opt1.value = True And po2opt3.value = False And po2opt4.value = False)
Then
  223       msgResult = MsgBox("You must make a hypergolic fuels loading decision.  Click
Yes if hypergolics are loaded now, in the integration facility.  Click No if hypergolics
are loaded later, on the launch pad.", vbYesNo)
  224       If msgResult = vbYes Then
```

```
  225         po2opt3.value = True
  226       Else
  227         po2opt4.value = True
  228       End If
  229     End If
  230     If (po2opt5.value = False And po2opt6.value = False) Then
  231       msgResult = MsgBox("You must make an ordnance decision.  Is ordnance
required?", vbYesNo)
  232       If msgResult = vbYes Then
  233         po2opt5.value = True
  234       Else
  235         po2opt6.value = True
  236       End If
  237     End If
  238     If (po2opt5.value = True And po2opt7.value = False And po2opt8.value = False)
Then
  239       msgResult = MsgBox("You must make an ordnance installation location decision.
Click Yes if ordnance is loaded now, in the integration facility.  Click No if ordnance
is loaded later, on the launch pad.", vbYesNo)
  240       If msgResult = vbYes Then
  241         po2opt7.value = True
  242       Else
  243         po2opt8.value = True
  244       End If
  245     End If

  246     'Code below populates the appropriate arena modules with the distributions the
user put into the combo boxes for PI-02 thru PI-10
  247     Dim m As Model
  248     Set m = ThisDocument.Model

  249     Dim pop47 As Module
  250     Dim pop47i As Long
  251     pop47i = m.Modules.Find(smFindTag, "pop47")
  252     Set pop47 = m.Modules(pop47i)
  253     pop47.Data("Expression") = po2com1.Text
  254     pop47.Data("Units") = po2com2.Text

  255     Dim pop48 As Module
  256     Dim pop48i As Long
  257     pop48i = m.Modules.Find(smFindTag, "pop48")
  258     Set pop48 = m.Modules(pop48i)
  259     pop48.Data("Expression") = po2com3.Text
  260     pop48.Data("Units") = po2com4.Text

  261     Dim pop49 As Module
  262     Dim pop49i As Long
  263     pop49i = m.Modules.Find(smFindTag, "pop49")
  264     Set pop49 = m.Modules(pop49i)
  265     pop49.Data("Expression") = po2com5.Text
  266     pop49.Data("Units") = po2com6.Text

  267     Dim pop50 As Module
  268     Dim pop50i As Long
  269     pop50i = m.Modules.Find(smFindTag, "pop50")
  270     Set pop50 = m.Modules(pop50i)
  271     pop50.Data("Expression") = po2com7.Text
  272     pop50.Data("Units") = po2com8.Text

  273     Dim pop51 As Module
  274     Dim pop51i As Long
  275     pop51i = m.Modules.Find(smFindTag, "pop51")
  276     Set pop51 = m.Modules(pop51i)
  277     pop51.Data("Expression") = po2com9.Text
  278     pop51.Data("Units") = po2com10.Text

  279     Dim pop52 As Module
  280     Dim pop52i As Long
  281     pop52i = m.Modules.Find(smFindTag, "pop52")
```

```
282     Set pop52 = m.Modules(pop52i)
283     pop52.Data("Expression") = po2com11.Text
284     pop52.Data("Units") = po2com12.Text

285     Dim pop64 As Module
286     Dim pop64i As Long
287     pop64i = m.Modules.Find(smFindTag, "pop64")
288     Set pop64 = m.Modules(pop64i)
289     If po2frm1.Visible = True Then
290       pop64.Data("Expression") = po2com13.Text
291       pop64.Data("Units") = po2com14.Text
292     Else
293       pop64.Data("Expression") = po2com37.Text
294       pop64.Data("Units") = po2com38.Text
295     End If

296     Dim pop53 As Module
297     Dim pop53i As Long
298     pop53i = m.Modules.Find(smFindTag, "pop53")
299     Set pop53 = m.Modules(pop53i)
300     pop53.Data("Expression") = po2com15.Text
301     pop53.Data("Units") = po2com16.Text

302     Dim pop54 As Module
303     Dim pop54i As Long
304     pop54i = m.Modules.Find(smFindTag, "pop54")
305     Set pop54 = m.Modules(pop54i)
306     pop54.Data("Expression") = po2com17.Text
307     pop54.Data("Units") = po2com18.Text

308     Dim pop55 As Module
309     Dim pop55i As Long
310     pop55i = m.Modules.Find(smFindTag, "pop55")
311     Set pop55 = m.Modules(pop55i)
312     pop55.Data("Expression") = po2com19.Text
313     pop55.Data("Units") = po2com20.Text

314     Dim pop56 As Module
315     Dim pop56i As Long
316     pop56i = m.Modules.Find(smFindTag, "pop56")
317     Set pop56 = m.Modules(pop56i)
318     pop56.Data("Expression") = po2com21.Text
319     pop56.Data("Units") = po2com22.Text

320     Dim pop57 As Module
321     Dim pop57i As Long
322     pop57i = m.Modules.Find(smFindTag, "pop57")
323     Set pop57 = m.Modules(pop57i)
324     pop57.Data("Expression") = po2com23.Text
325     pop57.Data("Units") = po2com24.Text

326     Dim pop58 As Module
327     Dim pop58i As Long
328     pop58i = m.Modules.Find(smFindTag, "pop58")
329     Set pop58 = m.Modules(pop58i)
330     pop58.Data("Expression") = po2com25.Text
331     pop58.Data("Units") = po2com26.Text

332     Dim pop59 As Module
333     Dim pop59i As Long
334     pop59i = m.Modules.Find(smFindTag, "pop59")
335     Set pop59 = m.Modules(pop59i)
336     pop59.Data("Expression") = po2com27.Text
337     pop59.Data("Units") = po2com28.Text

338     Dim pop60 As Module
339     Dim pop60i As Long
340     pop60i = m.Modules.Find(smFindTag, "pop60")
341     Set pop60 = m.Modules(pop60i)
```

```
342     pop60.Data("Expression") = po2com29.Text
343     pop60.Data("Units") = po2com30.Text

344     Dim pop61 As Module
345     Dim pop61i As Long
346     pop61i = m.Modules.Find(smFindTag, "pop61")
347     Set pop61 = m.Modules(pop61i)
348     pop61.Data("Expression") = po2com31.Text
349     pop61.Data("Units") = po2com32.Text

350     Dim pop62 As Module
351     Dim pop62i As Long
352     pop62i = m.Modules.Find(smFindTag, "pop62")
353     Set pop62 = m.Modules(pop62i)
354     pop62.Data("Expression") = po2com33.Text
355     pop62.Data("Units") = po2com34.Text

356     Dim pop63 As Module
357     Dim pop63i As Long
358     pop63i = m.Modules.Find(smFindTag, "pop63")
359     Set pop63 = m.Modules(pop63i)
360     pop63.Data("Expression") = po2com35.Text
361     pop63.Data("Units") = po2com36.Text

362     Dim pop34 As Module
363     Dim pop34i As Long
364     pop34i = m.Modules.Find(smFindTag, "pop34")
365     Set pop34 = m.Modules(pop34i)
366     pop34.Data("Expression") = po2com39.Text
367     pop34.Data("Units") = po2com40.Text

368     Dim pop71 As Module
369     Dim pop71i As Long
370     pop71i = m.Modules.Find(smFindTag, "pop71")
371     Set pop71 = m.Modules(pop71i)
372     pop71.Data("Expression") = po2com39.Text
373     pop71.Data("Units") = po2com40.Text

374     Dim pop35 As Module
375     Dim pop35i As Long
376     pop35i = m.Modules.Find(smFindTag, "pop35")
377     Set pop35 = m.Modules(pop35i)
378     pop35.Data("Expression") = po2com41.Text
379     pop35.Data("Units") = po2com42.Text

380     Dim pop77 As Module
381     Dim pop77i As Long
382     pop77i = m.Modules.Find(smFindTag, "pop77")
383     Set pop77 = m.Modules(pop77i)
384     pop77.Data("Expression") = po2com41.Text
385     pop77.Data("Units") = po2com42.Text

386     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
387     Dim pov6 As Module
388     Dim pov6i As Long
389     pov6i = m.Modules.Find(smFindTag, "pov6")
390     Set pov6 = m.Modules(pov6i)
391     If po2opt2.value = True Then
392       pov6.Data("Initial Value") = "0"
393     ElseIf po2opt3.value = True Then
394       pov6.Data("Initial Value") = "1"
395     Else
396       pov6.Data("Initial Value") = "2"
397     End If

398     Dim pov7 As Module
399     Dim pov7i As Long
400     pov7i = m.Modules.Find(smFindTag, "pov7")
```

```
401     Set pov7 = m.Modules(pov7i)
402     If po2opt6.value = True Then
403       pov7.Data("Initial Value") = "0"
404     ElseIf po2opt7.value = True Then
405       pov7.Data("Initial Value") = "1"
406     Else
407       pov7.Data("Initial Value") = "2"
408     End If

409     'code below hides the current form and shows the main form
410     Me.Hide
411     Hierarchy.Show

412   End Sub

413   Private Sub Label11_Click()

414   End Sub

415   Private Sub Label12_Click()

416   End Sub

417   Private Sub Label23_Click()

418   End Sub

419   Private Sub Label6_Click()

420   End Sub

421   Private Sub OptionButton1_Click()

422   End Sub

423   Private Sub OptionButton2_Click()

424   End Sub

425   Private Sub OptionButton4_Click()

426   End Sub

427   Private Sub OptionButton6_Click()

428   End Sub

429   Private Sub po2opt1_Click()
430     po2frm3.Visible = True

431   End Sub

432   Private Sub po2opt2_Click()
433     po2frm3.Visible = False

434   End Sub

435   Private Sub po2opt5_Click()
436     po2frm4.Visible = True

437   End Sub

438   Private Sub po2opt6_Click()
439     po2frm4.Visible = False

440   End Sub

441   Private Sub TextBox23_Change()
```

```
End Sub

Private Sub ToggleButton1_Click()

End Sub

Private Sub UserForm_Click()

End Sub

Private Sub UserForm_Initialize()
  Dim m As Model
  Set m = ThisDocument.Model

  'Code below populates large combo boxes for OP-01 thru OP-25
  Dim pop47 As Module
  Dim pop47i As Long
  Dim pop47v As String
  pop47i = m.Modules.Find(smFindTag, "pop47")
  Set pop47 = m.Modules(pop47i)
  pop47v = pop47.Data("Expression")

  po2on.po2com1.value = pop47v
  po2on.po2com1.AddItem "TRIA ( 54, 60, 84 )", 0
  po2on.po2com1.AddItem "TRIA ( Min, Mode, Max )", 1
  po2on.po2com1.AddItem "NORM ( Mean, StdDev )", 2
  po2on.po2com1.AddItem "EXPO ( Mean )", 3
  po2on.po2com1.AddItem "UNIF ( Min, Max )", 4

  Dim pop48 As Module
  Dim pop48i As Long
  Dim pop48v As String
  pop48i = m.Modules.Find(smFindTag, "pop48")
  Set pop48 = m.Modules(pop48i)
  pop48v = pop48.Data("Expression")

  po2on.po2com3.value = pop48v
  po2on.po2com3.AddItem "TRIA ( 108, 120, 168 )", 0
  po2on.po2com3.AddItem "TRIA ( Min, Mode, Max )", 1
  po2on.po2com3.AddItem "NORM ( Mean, StdDev )", 2
  po2on.po2com3.AddItem "EXPO ( Mean )", 3
  po2on.po2com3.AddItem "UNIF ( Min, Max )", 4

  Dim pop49 As Module
  Dim pop49i As Long
  Dim pop49v As String
  pop49i = m.Modules.Find(smFindTag, "pop49")
  Set pop49 = m.Modules(pop49i)
  pop49v = pop49.Data("Expression")

  po2on.po2com5.value = pop49v
  po2on.po2com5.AddItem "TRIA ( 27, 30, 42 )", 0
  po2on.po2com5.AddItem "TRIA ( Min, Mode, Max )", 1
  po2on.po2com5.AddItem "NORM ( Mean, StdDev )", 2
  po2on.po2com5.AddItem "EXPO ( Mean )", 3
  po2on.po2com5.AddItem "UNIF ( Min, Max )", 4

  Dim pop50 As Module
  Dim pop50i As Long
  Dim pop50v As String
  pop50i = m.Modules.Find(smFindTag, "pop50")
  Set pop50 = m.Modules(pop50i)
  pop50v = pop50.Data("Expression")

  po2on.po2com7.value = pop50v
  po2on.po2com7.AddItem "TRIA ( 81, 90, 126 )", 0
  po2on.po2com7.AddItem "TRIA ( Min, Mode, Max )", 1
  po2on.po2com7.AddItem "NORM ( Mean, StdDev )", 2
  po2on.po2com7.AddItem "EXPO ( Mean )", 3
```

```
498     po2on.po2com7.AddItem "UNIF ( Min, Max )", 4

499     Dim pop51 As Module
500     Dim pop51i As Long
501     Dim pop51v As String
502     pop51i = m.Modules.Find(smFindTag, "pop51")
503     Set pop51 = m.Modules(pop51i)
504     pop51v = pop51.Data("Expression")

505     po2on.po2com9.value = pop51v
506     po2on.po2com9.AddItem "TRIA ( 36, 40, 56 )", 0
507     po2on.po2com9.AddItem "TRIA ( Min, Mode, Max )", 1
508     po2on.po2com9.AddItem "NORM ( Mean, StdDev )", 2
509     po2on.po2com9.AddItem "EXPO ( Mean )", 3
510     po2on.po2com9.AddItem "UNIF ( Min, Max )", 4

511     Dim pop52 As Module
512     Dim pop52i As Long
513     Dim pop52v As String
514     pop52i = m.Modules.Find(smFindTag, "pop52")
515     Set pop52 = m.Modules(pop52i)
516     pop52v = pop52.Data("Expression")

517     po2on.po2com11.value = pop52v
518     po2on.po2com11.AddItem "TRIA ( 36, 40, 56 )", 0
519     po2on.po2com11.AddItem "TRIA ( Min, Mode, Max )", 1
520     po2on.po2com11.AddItem "NORM ( Mean, StdDev )", 2
521     po2on.po2com11.AddItem "EXPO ( Mean )", 3
522     po2on.po2com11.AddItem "UNIF ( Min, Max )", 4

523     Dim pop64 As Module
524     Dim pop64i As Long
525     Dim pop64v As String
526     pop64i = m.Modules.Find(smFindTag, "pop64")
527     Set pop64 = m.Modules(pop64i)
528     pop64v = pop64.Data("Expression")

529     po2on.po2com13.value = pop64v
530     po2on.po2com13.AddItem "TRIA ( 27, 30, 42 )", 0
531     po2on.po2com13.AddItem "TRIA ( Min, Mode, Max )", 1
532     po2on.po2com13.AddItem "NORM ( Mean, StdDev )", 2
533     po2on.po2com13.AddItem "EXPO ( Mean )", 3
534     po2on.po2com13.AddItem "UNIF ( Min, Max )", 4

535     po2on.po2com37.value = pop64v
536     po2on.po2com37.AddItem "TRIA ( 27, 30, 42 )", 0
537     po2on.po2com37.AddItem "TRIA ( Min, Mode, Max )", 1
538     po2on.po2com37.AddItem "NORM ( Mean, StdDev )", 2
539     po2on.po2com37.AddItem "EXPO ( Mean )", 3
540     po2on.po2com37.AddItem "UNIF ( Min, Max )", 4

541     Dim pop53 As Module
542     Dim pop53i As Long
543     Dim pop53v As String
544     pop53i = m.Modules.Find(smFindTag, "pop53")
545     Set pop53 = m.Modules(pop53i)
546     pop53v = pop53.Data("Expression")

547     po2on.po2com15.value = pop53v
548     po2on.po2com15.AddItem "TRIA ( 54, 60, 84 )", 0
549     po2on.po2com15.AddItem "TRIA ( Min, Mode, Max )", 1
550     po2on.po2com15.AddItem "NORM ( Mean, StdDev )", 2
551     po2on.po2com15.AddItem "EXPO ( Mean )", 3
552     po2on.po2com15.AddItem "UNIF ( Min, Max )", 4

553     Dim pop54 As Module
554     Dim pop54i As Long
555     Dim pop54v As String
556     pop54i = m.Modules.Find(smFindTag, "pop54")
```

```
    Set pop54 = m.Modules(pop54i)
    pop54v = pop54.Data("Expression")

    po2on.po2com17.value = pop54v
    po2on.po2com17.AddItem "TRIA ( 108, 120, 168 )", 0
    po2on.po2com17.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com17.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com17.AddItem "EXPO ( Mean )", 3
    po2on.po2com17.AddItem "UNIF ( Min, Max )", 4

    Dim pop55 As Module
    Dim pop55i As Long
    Dim pop55v As String
    pop55i = m.Modules.Find(smFindTag, "pop55")
    Set pop55 = m.Modules(pop55i)
    pop55v = pop55.Data("Expression")

    po2on.po2com19.value = pop55v
    po2on.po2com19.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com19.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com19.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com19.AddItem "EXPO ( Mean )", 3
    po2on.po2com19.AddItem "UNIF ( Min, Max )", 4

    Dim pop56 As Module
    Dim pop56i As Long
    Dim pop56v As String
    pop56i = m.Modules.Find(smFindTag, "pop56")
    Set pop56 = m.Modules(pop56i)
    pop56v = pop56.Data("Expression")

    po2on.po2com21.value = pop56v
    po2on.po2com21.AddItem "TRIA ( 81, 90, 126 )", 0
    po2on.po2com21.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com21.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com21.AddItem "EXPO ( Mean )", 3
    po2on.po2com21.AddItem "UNIF ( Min, Max )", 4

    Dim pop57 As Module
    Dim pop57i As Long
    Dim pop57v As String
    pop57i = m.Modules.Find(smFindTag, "pop57")
    Set pop57 = m.Modules(pop57i)
    pop57v = pop57.Data("Expression")

    po2on.po2com23.value = pop57v
    po2on.po2com23.AddItem "TRIA ( 36, 40, 56 )", 0
    po2on.po2com23.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com23.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com23.AddItem "EXPO ( Mean )", 3
    po2on.po2com23.AddItem "UNIF ( Min, Max )", 4

    Dim pop58 As Module
    Dim pop58i As Long
    Dim pop58v As String
    pop58i = m.Modules.Find(smFindTag, "pop58")
    Set pop58 = m.Modules(pop58i)
    pop58v = pop58.Data("Expression")

    po2on.po2com25.value = pop58v
    po2on.po2com25.AddItem "TRIA ( 36, 40, 56 )", 0
    po2on.po2com25.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com25.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com25.AddItem "EXPO ( Mean )", 3
    po2on.po2com25.AddItem "UNIF ( Min, Max )", 4

    Dim pop59 As Module
    Dim pop59i As Long
    Dim pop59v As String
```

```
    pop59i = m.Modules.Find(smFindTag, "pop59")
    Set pop59 = m.Modules(pop59i)
    pop59v = pop59.Data("Expression")

    po2on.po2com27.value = pop59v
    po2on.po2com27.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com27.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com27.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com27.AddItem "EXPO ( Mean )", 3
    po2on.po2com27.AddItem "UNIF ( Min, Max )", 4

    Dim pop60 As Module
    Dim pop60i As Long
    Dim pop60v As String
    pop60i = m.Modules.Find(smFindTag, "pop60")
    Set pop60 = m.Modules(pop60i)
    pop60v = pop60.Data("Expression")

    po2on.po2com29.value = pop60v
    po2on.po2com29.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com29.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com29.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com29.AddItem "EXPO ( Mean )", 3
    po2on.po2com29.AddItem "UNIF ( Min, Max )", 4

    Dim pop61 As Module
    Dim pop61i As Long
    Dim pop61v As String
    pop61i = m.Modules.Find(smFindTag, "pop61")
    Set pop61 = m.Modules(pop61i)
    pop61v = pop61.Data("Expression")

    po2on.po2com31.value = pop61v
    po2on.po2com31.AddItem "TRIA ( 81, 90, 126 )", 0
    po2on.po2com31.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com31.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com31.AddItem "EXPO ( Mean )", 3
    po2on.po2com31.AddItem "UNIF ( Min, Max )", 4

    Dim pop62 As Module
    Dim pop62i As Long
    Dim pop62v As String
    pop62i = m.Modules.Find(smFindTag, "pop62")
    Set pop62 = m.Modules(pop62i)
    pop62v = pop62.Data("Expression")

    po2on.po2com33.value = pop62v
    po2on.po2com33.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com33.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com33.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com33.AddItem "EXPO ( Mean )", 3
    po2on.po2com33.AddItem "UNIF ( Min, Max )", 4

    Dim pop63 As Module
    Dim pop63i As Long
    Dim pop63v As String
    pop63i = m.Modules.Find(smFindTag, "pop63")
    Set pop63 = m.Modules(pop63i)
    pop63v = pop63.Data("Expression")

    po2on.po2com35.value = pop63v
    po2on.po2com35.AddItem "TRIA ( 27, 30, 42 )", 0
    po2on.po2com35.AddItem "TRIA ( Min, Mode, Max )", 1
    po2on.po2com35.AddItem "NORM ( Mean, StdDev )", 2
    po2on.po2com35.AddItem "EXPO ( Mean )", 3
    po2on.po2com35.AddItem "UNIF ( Min, Max )", 4

    Dim pop34 As Module
    Dim pop34i As Long
```

```
675     Dim pop34v As String
676     pop34i = m.Modules.Find(smFindTag, "pop34")
677     Set pop34 = m.Modules(pop34i)
678     pop34v = pop34.Data("Expression")

679     po2on.po2com39.value = pop34v
680     po2on.po2com39.AddItem "TRIA ( 756, 840, 1176 )", 0
681     po2on.po2com39.AddItem "TRIA ( Min, Mode, Max )", 1
682     po2on.po2com39.AddItem "NORM ( Mean, StdDev )", 2
683     po2on.po2com39.AddItem "EXPO ( Mean )", 3
684     po2on.po2com39.AddItem "UNIF ( Min, Max )", 4

685     Dim pop35 As Module
686     Dim pop35i As Long
687     Dim pop35v As String
688     pop35i = m.Modules.Find(smFindTag, "pop35")
689     Set pop35 = m.Modules(pop35i)
690     pop35v = pop35.Data("Expression")

691     po2on.po2com41.value = pop35v
692     po2on.po2com41.AddItem "TRIA ( 324, 360, 504 )", 0
693     po2on.po2com41.AddItem "TRIA ( Min, Mode, Max )", 1
694     po2on.po2com41.AddItem "NORM ( Mean, StdDev )", 2
695     po2on.po2com41.AddItem "EXPO ( Mean )", 3
696     po2on.po2com41.AddItem "UNIF ( Min, Max )", 4

697     'Code below populates small combo boxes for OP-01 thru OP-25
698     Dim pop47u As Module
699     Dim pop47ui As Long
700     Dim pop47uv As String
701     pop47ui = m.Modules.Find(smFindTag, "pop47")
702     Set pop47u = m.Modules(pop47ui)
703     pop47uv = pop47u.Data("Units")

704     po2on.po2com2.value = pop47uv
705     po2on.po2com2.AddItem "Seconds", 0
706     po2on.po2com2.AddItem "Minutes", 1
707     po2on.po2com2.AddItem "Hours", 2
708     po2on.po2com2.AddItem "Days", 3

709     Dim pop48u As Module
710     Dim pop48ui As Long
711     Dim pop48uv As String
712     pop48ui = m.Modules.Find(smFindTag, "pop48")
713     Set pop48u = m.Modules(pop48ui)
714     pop48uv = pop48u.Data("Units")

715     po2on.po2com4.value = pop48uv
716     po2on.po2com4.AddItem "Seconds", 0
717     po2on.po2com4.AddItem "Minutes", 1
718     po2on.po2com4.AddItem "Hours", 2
719     po2on.po2com4.AddItem "Days", 3

720     Dim pop49u As Module
721     Dim pop49ui As Long
722     Dim pop49uv As String
723     pop49ui = m.Modules.Find(smFindTag, "pop49")
724     Set pop49u = m.Modules(pop49ui)
725     pop49uv = pop49u.Data("Units")

726     po2on.po2com6.value = pop49uv
727     po2on.po2com6.AddItem "Seconds", 0
728     po2on.po2com6.AddItem "Minutes", 1
729     po2on.po2com6.AddItem "Hours", 2
730     po2on.po2com6.AddItem "Days", 3

731     Dim pop50u As Module
732     Dim pop50ui As Long
733     Dim pop50uv As String
```

```
734     pop50ui = m.Modules.Find(smFindTag, "pop50")
735     Set pop50u = m.Modules(pop50ui)
736     pop50uv = pop50u.Data("Units")

737     po2on.po2com8.value = pop50uv
738     po2on.po2com8.AddItem "Seconds", 0
739     po2on.po2com8.AddItem "Minutes", 1
740     po2on.po2com8.AddItem "Hours", 2
741     po2on.po2com8.AddItem "Days", 3

742     Dim pop51u As Module
743     Dim pop51ui As Long
744     Dim pop51uv As String
745     pop51ui = m.Modules.Find(smFindTag, "pop51")
746     Set pop51u = m.Modules(pop51ui)
747     pop51uv = pop51u.Data("Units")

748     po2on.po2com10.value = pop51uv
749     po2on.po2com10.AddItem "Seconds", 0
750     po2on.po2com10.AddItem "Minutes", 1
751     po2on.po2com10.AddItem "Hours", 2
752     po2on.po2com10.AddItem "Days", 3

753     Dim pop52u As Module
754     Dim pop52ui As Long
755     Dim pop52uv As String
756     pop52ui = m.Modules.Find(smFindTag, "pop52")
757     Set pop52u = m.Modules(pop52ui)
758     pop52uv = pop52u.Data("Units")

759     po2on.po2com12.value = pop52uv
760     po2on.po2com12.AddItem "Seconds", 0
761     po2on.po2com12.AddItem "Minutes", 1
762     po2on.po2com12.AddItem "Hours", 2
763     po2on.po2com12.AddItem "Days", 3

764     Dim pop64u As Module
765     Dim pop64ui As Long
766     Dim pop64uv As String
767     pop64ui = m.Modules.Find(smFindTag, "pop64")
768     Set pop64u = m.Modules(pop64ui)
769     pop64uv = pop64u.Data("Units")

770     po2on.po2com14.value = pop64uv
771     po2on.po2com14.AddItem "Seconds", 0
772     po2on.po2com14.AddItem "Minutes", 1
773     po2on.po2com14.AddItem "Hours", 2
774     po2on.po2com14.AddItem "Days", 3

775     po2on.po2com38.value = pop64uv
776     po2on.po2com38.AddItem "Seconds", 0
777     po2on.po2com38.AddItem "Minutes", 1
778     po2on.po2com38.AddItem "Hours", 2
779     po2on.po2com38.AddItem "Days", 3

780     Dim pop53u As Module
781     Dim pop53ui As Long
782     Dim pop53uv As String
783     pop53ui = m.Modules.Find(smFindTag, "pop53")
784     Set pop53u = m.Modules(pop53ui)
785     pop53uv = pop53u.Data("Units")

786     po2on.po2com16.value = pop53uv
787     po2on.po2com16.AddItem "Seconds", 0
788     po2on.po2com16.AddItem "Minutes", 1
789     po2on.po2com16.AddItem "Hours", 2
790     po2on.po2com16.AddItem "Days", 3

791     Dim pop54u As Module
```

```
792     Dim pop54ui As Long
793     Dim pop54uv As String
794     pop54ui = m.Modules.Find(smFindTag, "pop54")
795     Set pop54u = m.Modules(pop54ui)
796     pop54uv = pop54u.Data("Units")

797     po2on.po2com18.value = pop54uv
798     po2on.po2com18.AddItem "Seconds", 0
799     po2on.po2com18.AddItem "Minutes", 1
800     po2on.po2com18.AddItem "Hours", 2
801     po2on.po2com18.AddItem "Days", 3

802     Dim pop55u As Module
803     Dim pop55ui As Long
804     Dim pop55uv As String
805     pop55ui = m.Modules.Find(smFindTag, "pop55")
806     Set pop55u = m.Modules(pop55ui)
807     pop55uv = pop55u.Data("Units")

808     po2on.po2com20.value = pop55uv
809     po2on.po2com20.AddItem "Seconds", 0
810     po2on.po2com20.AddItem "Minutes", 1
811     po2on.po2com20.AddItem "Hours", 2
812     po2on.po2com20.AddItem "Days", 3

813     Dim pop56u As Module
814     Dim pop56ui As Long
815     Dim pop56uv As String
816     pop56ui = m.Modules.Find(smFindTag, "pop56")
817     Set pop56u = m.Modules(pop56ui)
818     pop56uv = pop56u.Data("Units")

819     po2on.po2com22.value = pop56uv
820     po2on.po2com22.AddItem "Seconds", 0
821     po2on.po2com22.AddItem "Minutes", 1
822     po2on.po2com22.AddItem "Hours", 2
823     po2on.po2com22.AddItem "Days", 3

824     Dim pop57u As Module
825     Dim pop57ui As Long
826     Dim pop57uv As String
827     pop57ui = m.Modules.Find(smFindTag, "pop57")
828     Set pop57u = m.Modules(pop57ui)
829     pop57uv = pop57u.Data("Units")

830     po2on.po2com24.value = pop57uv
831     po2on.po2com24.AddItem "Seconds", 0
832     po2on.po2com24.AddItem "Minutes", 1
833     po2on.po2com24.AddItem "Hours", 2
834     po2on.po2com24.AddItem "Days", 3

835     Dim pop58u As Module
836     Dim pop58ui As Long
837     Dim pop58uv As String
838     pop58ui = m.Modules.Find(smFindTag, "pop58")
839     Set pop58u = m.Modules(pop58ui)
840     pop58uv = pop58u.Data("Units")

841     po2on.po2com26.value = pop58uv
842     po2on.po2com26.AddItem "Seconds", 0
843     po2on.po2com26.AddItem "Minutes", 1
844     po2on.po2com26.AddItem "Hours", 2
845     po2on.po2com26.AddItem "Days", 3

846     Dim pop59u As Module
847     Dim pop59ui As Long
848     Dim pop59uv As String
849     pop59ui = m.Modules.Find(smFindTag, "pop59")
850     Set pop59u = m.Modules(pop59ui)
```

```
851     pop59uv = pop59u.Data("Units")

852     po2on.po2com28.value = pop59uv
853     po2on.po2com28.AddItem "Seconds", 0
854     po2on.po2com28.AddItem "Minutes", 1
855     po2on.po2com28.AddItem "Hours", 2
856     po2on.po2com28.AddItem "Days", 3

857     Dim pop60u As Module
858     Dim pop60ui As Long
859     Dim pop60uv As String
860     pop60ui = m.Modules.Find(smFindTag, "pop60")
861     Set pop60u = m.Modules(pop60ui)
862     pop60uv = pop60u.Data("Units")

863     po2on.po2com30.value = pop60uv
864     po2on.po2com30.AddItem "Seconds", 0
865     po2on.po2com30.AddItem "Minutes", 1
866     po2on.po2com30.AddItem "Hours", 2
867     po2on.po2com30.AddItem "Days", 3

868     Dim pop61u As Module
869     Dim pop61ui As Long
870     Dim pop61uv As String
871     pop61ui = m.Modules.Find(smFindTag, "pop61")
872     Set pop61u = m.Modules(pop61ui)
873     pop61uv = pop61u.Data("Units")

874     po2on.po2com32.value = pop61uv
875     po2on.po2com32.AddItem "Seconds", 0
876     po2on.po2com32.AddItem "Minutes", 1
877     po2on.po2com32.AddItem "Hours", 2
878     po2on.po2com32.AddItem "Days", 3

879     Dim pop62u As Module
880     Dim pop62ui As Long
881     Dim pop62uv As String
882     pop62ui = m.Modules.Find(smFindTag, "pop62")
883     Set pop62u = m.Modules(pop62ui)
884     pop62uv = pop62u.Data("Units")

885     po2on.po2com34.value = pop62uv
886     po2on.po2com34.AddItem "Seconds", 0
887     po2on.po2com34.AddItem "Minutes", 1
888     po2on.po2com34.AddItem "Hours", 2
889     po2on.po2com34.AddItem "Days", 3

890     Dim pop63u As Module
891     Dim pop63ui As Long
892     Dim pop63uv As String
893     pop63ui = m.Modules.Find(smFindTag, "pop63")
894     Set pop63u = m.Modules(pop63ui)
895     pop63uv = pop63u.Data("Units")

896     po2on.po2com36.value = pop63uv
897     po2on.po2com36.AddItem "Seconds", 0
898     po2on.po2com36.AddItem "Minutes", 1
899     po2on.po2com36.AddItem "Hours", 2
900     po2on.po2com36.AddItem "Days", 3

901     Dim pop34u As Module
902     Dim pop34ui As Long
903     Dim pop34uv As String
904     pop34ui = m.Modules.Find(smFindTag, "pop34")
905     Set pop34u = m.Modules(pop34ui)
906     pop34uv = pop34u.Data("Units")

907     po2on.po2com40.value = pop34uv
908     po2on.po2com40.AddItem "Seconds", 0
```

```
909     po2on.po2com40.AddItem "Minutes", 1
910     po2on.po2com40.AddItem "Hours", 2
911     po2on.po2com40.AddItem "Days", 3

912     Dim pop35u As Module
913     Dim pop35ui As Long
914     Dim pop35uv As String
915     pop35ui = m.Modules.Find(smFindTag, "pop35")
916     Set pop35u = m.Modules(pop35ui)
917     pop35uv = pop35u.Data("Units")

918     po2on.po2com42.value = pop35uv
919     po2on.po2com42.AddItem "Seconds", 0
920     po2on.po2com42.AddItem "Minutes", 1
921     po2on.po2com42.AddItem "Hours", 2
922     po2on.po2com42.AddItem "Days", 3

923   End Sub
```

Project/po3offpreint

```
1   Private Sub CommandButton6_Click()
2     Me.Hide
3     po1prelim.Show

4   End Sub

5   Private Sub CommandButton7_Click()
6     Hierarchy.done05.Visible = True

7     'code below checks to see if any option button sets were not clicked, and if
so, forces the user to make a choice.
8     Dim msgResult As Integer
9     If (po3opt1.value = False And po3opt2.value = False) Then
10      msgResult = MsgBox("You must make an integration orientation choice.  Click
Yes if integration takes place horizontally.  Click No if integration takes place
vertically.", vbYesNo)
11      If msgResult = vbYes Then
12        po3opt1.value = True
13      Else
14        po3opt2.value = True
15      End If
16    End If

17    'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
18    Dim m As Model
19    Set m = ThisDocument.Model

20    Dim pop8 As Module
21    Dim pop8i As Long
22    pop8i = m.Modules.Find(smFindTag, "pop8")
23    Set pop8 = m.Modules(pop8i)
24    pop8.Data("Expression") = po3com1.Text
25    pop8.Data("Units") = po3com2.Text

26    Dim pop9 As Module
27    Dim pop9i As Long
28    pop9i = m.Modules.Find(smFindTag, "pop9")
29    Set pop9 = m.Modules(pop9i)
30    pop9.Data("Expression") = po3com3.Text
31    pop9.Data("Units") = po3com4.Text

32    Dim pop10 As Module
33    Dim pop10i As Long
34    pop10i = m.Modules.Find(smFindTag, "pop10")
35    Set pop10 = m.Modules(pop10i)
36    pop10.Data("Expression") = po3com5.Text
37    pop10.Data("Units") = po3com6.Text

38    Dim pop11 As Module
39    Dim pop11i As Long
40    pop11i = m.Modules.Find(smFindTag, "pop11")
41    Set pop11 = m.Modules(pop11i)
42    pop11.Data("Expression") = po3com7.Text
43    pop11.Data("Units") = po3com8.Text

44    Dim pop12 As Module
45    Dim pop12i As Long
46    pop12i = m.Modules.Find(smFindTag, "pop12")
47    Set pop12 = m.Modules(pop12i)
48    pop12.Data("Expression") = po3com9.Text
49    pop12.Data("Units") = po3com10.Text

50    Dim pop13 As Module
51    Dim pop13i As Long
52    pop13i = m.Modules.Find(smFindTag, "pop13")
```

```
53      Set pop13 = m.Modules(pop13i)
54      pop13.Data("Expression") = po3com11.Text
55      pop13.Data("Units") = po3com12.Text

56      Dim pop33 As Module
57      Dim pop33i As Long
58      pop33i = m.Modules.Find(smFindTag, "pop33")
59      Set pop33 = m.Modules(pop33i)
60      If po3frm1.Visible = True Then
61        pop33.Data("Expression") = po3com13.Text
62        pop33.Data("Units") = po3com14.Text
63      Else
64        pop33.Data("Expression") = po3com23.Text
65        pop33.Data("Units") = po3com24.Text
66      End If

67      Dim pop14 As Module
68      Dim pop14i As Long
69      pop14i = m.Modules.Find(smFindTag, "pop14")
70      Set pop14 = m.Modules(pop14i)
71      pop14.Data("Expression") = po3com15.Text
72      pop14.Data("Units") = po3com16.Text

73      Dim pop15 As Module
74      Dim pop15i As Long
75      pop15i = m.Modules.Find(smFindTag, "pop15")
76      Set pop15 = m.Modules(pop15i)
77      pop15.Data("Expression") = po3com17.Text
78      pop15.Data("Units") = po3com18.Text

79      Dim pop16 As Module
80      Dim pop16i As Long
81      pop16i = m.Modules.Find(smFindTag, "pop16")
82      Set pop16 = m.Modules(pop16i)
83      pop16.Data("Expression") = po3com19.Text
84      pop16.Data("Units") = po3com20.Text

85      Dim pop17 As Module
86      Dim pop17i As Long
87      pop17i = m.Modules.Find(smFindTag, "pop17")
88      Set pop17 = m.Modules(pop17i)
89      pop17.Data("Expression") = po3com21.Text
90      pop17.Data("Units") = po3com22.Text

91      'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
92      Dim pov4 As Module
93      Dim pov4i As Long
94      pov4i = m.Modules.Find(smFindTag, "pov4")
95      Set pov4 = m.Modules(pov4i)
96      If po3opt2.value = True Then
97        pov4.Data("Initial Value") = "1"
98      Else
99        pov4.Data("Initial Value") = "0"
100     End If

101     'code below hides the current form and then shows the next form in the sequence
102     Me.Hide
103     po5offhyper.Show

104   End Sub

105   Private Sub CommandButton9_Click()
106     Hierarchy.done05.Visible = True

107     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
108     Dim m As Model
109     Set m = ThisDocument.Model
```

```
110     Dim pop8 As Module
111     Dim pop8i As Long
112     pop8i = m.Modules.Find(smFindTag, "pop8")
113     Set pop8 = m.Modules(pop8i)
114     pop8.Data("Expression") = po3com1.Text
115     pop8.Data("Units") = po3com2.Text

116     Dim pop9 As Module
117     Dim pop9i As Long
118     pop9i = m.Modules.Find(smFindTag, "pop9")
119     Set pop9 = m.Modules(pop9i)
120     pop9.Data("Expression") = po3com3.Text
121     pop9.Data("Units") = po3com4.Text

122     Dim pop10 As Module
123     Dim pop10i As Long
124     pop10i = m.Modules.Find(smFindTag, "pop10")
125     Set pop10 = m.Modules(pop10i)
126     pop10.Data("Expression") = po3com5.Text
127     pop10.Data("Units") = po3com6.Text

128     Dim pop11 As Module
129     Dim pop11i As Long
130     pop11i = m.Modules.Find(smFindTag, "pop11")
131     Set pop11 = m.Modules(pop11i)
132     pop11.Data("Expression") = po3com7.Text
133     pop11.Data("Units") = po3com8.Text

134     Dim pop12 As Module
135     Dim pop12i As Long
136     pop12i = m.Modules.Find(smFindTag, "pop12")
137     Set pop12 = m.Modules(pop12i)
138     pop12.Data("Expression") = po3com9.Text
139     pop12.Data("Units") = po3com10.Text

140     Dim pop13 As Module
141     Dim pop13i As Long
142     pop13i = m.Modules.Find(smFindTag, "pop13")
143     Set pop13 = m.Modules(pop13i)
144     pop13.Data("Expression") = po3com11.Text
145     pop13.Data("Units") = po3com12.Text

146     Dim pop33 As Module
147     Dim pop33i As Long
148     pop33i = m.Modules.Find(smFindTag, "pop33")
149     Set pop33 = m.Modules(pop33i)
150     If po3frm1.Visible = True Then
151       pop33.Data("Expression") = po3com13.Text
152       pop33.Data("Units") = po3com14.Text
153     Else
154       pop33.Data("Expression") = po3com23.Text
155       pop33.Data("Units") = po3com24.Text
156     End If

157     Dim pop14 As Module
158     Dim pop14i As Long
159     pop14i = m.Modules.Find(smFindTag, "pop14")
160     Set pop14 = m.Modules(pop14i)
161     pop14.Data("Expression") = po3com15.Text
162     pop14.Data("Units") = po3com16.Text

163     Dim pop15 As Module
164     Dim pop15i As Long
165     pop15i = m.Modules.Find(smFindTag, "pop15")
166     Set pop15 = m.Modules(pop15i)
167     pop15.Data("Expression") = po3com17.Text
168     pop15.Data("Units") = po3com18.Text
```

```
169     Dim pop16 As Module
170     Dim pop16i As Long
171     pop16i = m.Modules.Find(smFindTag, "pop16")
172     Set pop16 = m.Modules(pop16i)
173     pop16.Data("Expression") = po3com19.Text
174     pop16.Data("Units") = po3com20.Text

175     Dim pop17 As Module
176     Dim pop17i As Long
177     pop17i = m.Modules.Find(smFindTag, "pop17")
178     Set pop17 = m.Modules(pop17i)
179     pop17.Data("Expression") = po3com21.Text
180     pop17.Data("Units") = po3com22.Text

181     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
182     Dim pov4 As Module
183     Dim pov4i As Long
184     pov4i = m.Modules.Find(smFindTag, "pov4")
185     Set pov4 = m.Modules(pov4i)
186     If po3opt2.value = True Then
187       pov4.Data("Initial Value") = "1"
188     Else
189       pov4.Data("Initial Value") = "0"
190     End If

191     Me.Hide
192     Hierarchy.Show

193   End Sub

194   Private Sub Label1_Click()

195   End Sub

196   Private Sub Label11_Click()

197   End Sub

198   Private Sub Label12_Click()

199   End Sub

200   Private Sub Label15_Click()

201   End Sub

202   Private Sub Label23_Click()

203   End Sub

204   Private Sub OptionButton1_Click()

205   End Sub

206   Private Sub OptionButton2_Click()

207   End Sub

208   Private Sub OptionButton4_Click()

209   End Sub

210   Private Sub OptionButton6_Click()

211   End Sub

212   Private Sub po3opt1_Click()
213     po3frm2.Visible = True
```

```
214     po3frm1.Visible = False
215     po6erect.po6frm1.Visible = False
216     po6erect.po6frm2.Visible = True

217   End Sub

218   Private Sub po3opt2_Click()
219     po3frm2.Visible = False
220     po3frm1.Visible = True
221     po6erect.po6frm1.Visible = True
222     po6erect.po6frm2.Visible = False

223   End Sub

224   Private Sub ToggleButton1_Click()

225   End Sub

226   Private Sub UserForm_Click()

227   End Sub

228   Private Sub UserForm_Initialize()
229     Dim m As Model
230     Set m = ThisDocument.Model

231     'Code below populates large combo boxes for OW-02 thru OW-13
232     Dim pop8 As Module
233     Dim pop8i As Long
234     Dim pop8v As String
235     pop8i = m.Modules.Find(smFindTag, "pop8")
236     Set pop8 = m.Modules(pop8i)
237     pop8v = pop8.Data("Expression")

238     po3offpreint.po3com1.value = pop8v
239     po3offpreint.po3com1.AddItem "TRIA ( 54, 60, 84 )", 0
240     po3offpreint.po3com1.AddItem "TRIA ( Min, Mode, Max )", 1
241     po3offpreint.po3com1.AddItem "NORM ( Mean, StdDev )", 2
242     po3offpreint.po3com1.AddItem "EXPO ( Mean )", 3
243     po3offpreint.po3com1.AddItem "UNIF ( Min, Max )", 4

244     Dim pop9 As Module
245     Dim pop9i As Long
246     Dim pop9v As String
247     pop9i = m.Modules.Find(smFindTag, "pop9")
248     Set pop9 = m.Modules(pop9i)
249     pop9v = pop9.Data("Expression")

250     po3offpreint.po3com3.value = pop9v
251     po3offpreint.po3com3.AddItem "TRIA ( 108, 120, 168 )", 0
252     po3offpreint.po3com3.AddItem "TRIA ( Min, Mode, Max )", 1
253     po3offpreint.po3com3.AddItem "NORM ( Mean, StdDev )", 2
254     po3offpreint.po3com3.AddItem "EXPO ( Mean )", 3
255     po3offpreint.po3com3.AddItem "UNIF ( Min, Max )", 4

256     Dim pop10 As Module
257     Dim pop10i As Long
258     Dim pop10v As String
259     pop10i = m.Modules.Find(smFindTag, "pop10")
260     Set pop10 = m.Modules(pop10i)
261     pop10v = pop10.Data("Expression")

262     po3offpreint.po3com5.value = pop10v
263     po3offpreint.po3com5.AddItem "TRIA ( 27, 30, 42 )", 0
264     po3offpreint.po3com5.AddItem "TRIA ( Min, Mode, Max )", 1
265     po3offpreint.po3com5.AddItem "NORM ( Mean, StdDev )", 2
266     po3offpreint.po3com5.AddItem "EXPO ( Mean )", 3
267     po3offpreint.po3com5.AddItem "UNIF ( Min, Max )", 4
```

```
268     Dim pop11 As Module
269     Dim pop11i As Long
270     Dim pop11v As String
271     pop11i = m.Modules.Find(smFindTag, "pop11")
272     Set pop11 = m.Modules(pop11i)
273     pop11v = pop11.Data("Expression")

274     po3offpreint.po3com7.value = pop11v
275     po3offpreint.po3com7.AddItem "TRIA ( 81, 90, 126 )", 0
276     po3offpreint.po3com7.AddItem "TRIA ( Min, Mode, Max )", 1
277     po3offpreint.po3com7.AddItem "NORM ( Mean, StdDev )", 2
278     po3offpreint.po3com7.AddItem "EXPO ( Mean )", 3
279     po3offpreint.po3com7.AddItem "UNIF ( Min, Max )", 4

280     Dim pop12 As Module
281     Dim pop12i As Long
282     Dim pop12v As String
283     pop12i = m.Modules.Find(smFindTag, "pop12")
284     Set pop12 = m.Modules(pop12i)
285     pop12v = pop12.Data("Expression")

286     po3offpreint.po3com9.value = pop12v
287     po3offpreint.po3com9.AddItem "TRIA ( 36, 40, 56 )", 0
288     po3offpreint.po3com9.AddItem "TRIA ( Min, Mode, Max )", 1
289     po3offpreint.po3com9.AddItem "NORM ( Mean, StdDev )", 2
290     po3offpreint.po3com9.AddItem "EXPO ( Mean )", 3
291     po3offpreint.po3com9.AddItem "UNIF ( Min, Max )", 4

292     Dim pop13 As Module
293     Dim pop13i As Long
294     Dim pop13v As String
295     pop13i = m.Modules.Find(smFindTag, "pop13")
296     Set pop13 = m.Modules(pop13i)
297     pop13v = pop13.Data("Expression")

298     po3offpreint.po3com11.value = pop13v
299     po3offpreint.po3com11.AddItem "TRIA ( 36, 40, 56 )", 0
300     po3offpreint.po3com11.AddItem "TRIA ( Min, Mode, Max )", 1
301     po3offpreint.po3com11.AddItem "NORM ( Mean, StdDev )", 2
302     po3offpreint.po3com11.AddItem "EXPO ( Mean )", 3
303     po3offpreint.po3com11.AddItem "UNIF ( Min, Max )", 4

304     Dim pop33 As Module
305     Dim pop33i As Long
306     Dim pop33v As String
307     pop33i = m.Modules.Find(smFindTag, "pop33")
308     Set pop33 = m.Modules(pop33i)
309     pop33v = pop33.Data("Expression")

310     po3offpreint.po3com13.value = pop33v
311     po3offpreint.po3com13.AddItem "TRIA ( 27, 30, 42 )", 0
312     po3offpreint.po3com13.AddItem "TRIA ( Min, Mode, Max )", 1
313     po3offpreint.po3com13.AddItem "NORM ( Mean, StdDev )", 2
314     po3offpreint.po3com13.AddItem "EXPO ( Mean )", 3
315     po3offpreint.po3com13.AddItem "UNIF ( Min, Max )", 4

316     po3offpreint.po3com23.value = pop33v
317     po3offpreint.po3com23.AddItem "TRIA ( 27, 30, 42 )", 0
318     po3offpreint.po3com23.AddItem "TRIA ( Min, Mode, Max )", 1
319     po3offpreint.po3com23.AddItem "NORM ( Mean, StdDev )", 2
320     po3offpreint.po3com23.AddItem "EXPO ( Mean )", 3
321     po3offpreint.po3com23.AddItem "UNIF ( Min, Max )", 4

322     Dim pop14 As Module
323     Dim pop14i As Long
324     Dim pop14v As String
325     pop14i = m.Modules.Find(smFindTag, "pop14")
326     Set pop14 = m.Modules(pop14i)
327     pop14v = pop14.Data("Expression")
```

```
    po3offpreint.po3com15.value = pop14v
    po3offpreint.po3com15.AddItem "TRIA ( 27, 30, 42 )", 0
    po3offpreint.po3com15.AddItem "TRIA ( Min, Mode, Max )", 1
    po3offpreint.po3com15.AddItem "NORM ( Mean, StdDev )", 2
    po3offpreint.po3com15.AddItem "EXPO ( Mean )", 3
    po3offpreint.po3com15.AddItem "UNIF ( Min, Max )", 4

    Dim pop15 As Module
    Dim pop15i As Long
    Dim pop15v As String
    pop15i = m.Modules.Find(smFindTag, "pop15")
    Set pop15 = m.Modules(pop15i)
    pop15v = pop15.Data("Expression")

    po3offpreint.po3com17.value = pop15v
    po3offpreint.po3com17.AddItem "TRIA ( 54, 60, 84 )", 0
    po3offpreint.po3com17.AddItem "TRIA ( Min, Mode, Max )", 1
    po3offpreint.po3com17.AddItem "NORM ( Mean, StdDev )", 2
    po3offpreint.po3com17.AddItem "EXPO ( Mean )", 3
    po3offpreint.po3com17.AddItem "UNIF ( Min, Max )", 4

    Dim pop16 As Module
    Dim pop16i As Long
    Dim pop16v As String
    pop16i = m.Modules.Find(smFindTag, "pop16")
    Set pop16 = m.Modules(pop16i)
    pop16v = pop16.Data("Expression")

    po3offpreint.po3com19.value = pop16v
    po3offpreint.po3com19.AddItem "TRIA ( 27, 30, 42 )", 0
    po3offpreint.po3com19.AddItem "TRIA ( Min, Mode, Max )", 1
    po3offpreint.po3com19.AddItem "NORM ( Mean, StdDev )", 2
    po3offpreint.po3com19.AddItem "EXPO ( Mean )", 3
    po3offpreint.po3com19.AddItem "UNIF ( Min, Max )", 4

    Dim pop17 As Module
    Dim pop17i As Long
    Dim pop17v As String
    pop17i = m.Modules.Find(smFindTag, "pop17")
    Set pop17 = m.Modules(pop17i)
    pop17v = pop17.Data("Expression")

    po3offpreint.po3com21.value = pop17v
    po3offpreint.po3com21.AddItem "TRIA ( 27, 30, 42 )", 0
    po3offpreint.po3com21.AddItem "TRIA ( Min, Mode, Max )", 1
    po3offpreint.po3com21.AddItem "NORM ( Mean, StdDev )", 2
    po3offpreint.po3com21.AddItem "EXPO ( Mean )", 3
    po3offpreint.po3com21.AddItem "UNIF ( Min, Max )", 4

    'Code below populates small combo boxes for OW-02 thru OW-13
    Dim pop8u As Module
    Dim pop8ui As Long
    Dim pop8uv As String
    pop8ui = m.Modules.Find(smFindTag, "pop8")
    Set pop8u = m.Modules(pop8ui)
    pop8uv = pop8u.Data("Units")

    po3offpreint.po3com2.value = pop8uv
    po3offpreint.po3com2.AddItem "Seconds", 0
    po3offpreint.po3com2.AddItem "Minutes", 1
    po3offpreint.po3com2.AddItem "Hours", 2
    po3offpreint.po3com2.AddItem "Days", 3

    Dim pop9u As Module
    Dim pop9ui As Long
    Dim pop9uv As String
    pop9ui = m.Modules.Find(smFindTag, "pop9")
    Set pop9u = m.Modules(pop9ui)
```

```
387     pop9uv = pop9u.Data("Units")

388     po3offpreint.po3com4.value = pop9uv
389     po3offpreint.po3com4.AddItem "Seconds", 0
390     po3offpreint.po3com4.AddItem "Minutes", 1
391     po3offpreint.po3com4.AddItem "Hours", 2
392     po3offpreint.po3com4.AddItem "Days", 3

393     Dim pop10u As Module
394     Dim pop10ui As Long
395     Dim pop10uv As String
396     pop10ui = m.Modules.Find(smFindTag, "pop10")
397     Set pop10u = m.Modules(pop10ui)
398     pop10uv = pop10u.Data("Units")

399     po3offpreint.po3com6.value = pop10uv
400     po3offpreint.po3com6.AddItem "Seconds", 0
401     po3offpreint.po3com6.AddItem "Minutes", 1
402     po3offpreint.po3com6.AddItem "Hours", 2
403     po3offpreint.po3com6.AddItem "Days", 3

404     Dim pop11u As Module
405     Dim pop11ui As Long
406     Dim pop11uv As String
407     pop11ui = m.Modules.Find(smFindTag, "pop11")
408     Set pop11u = m.Modules(pop11ui)
409     pop11uv = pop11u.Data("Units")

410     po3offpreint.po3com8.value = pop11uv
411     po3offpreint.po3com8.AddItem "Seconds", 0
412     po3offpreint.po3com8.AddItem "Minutes", 1
413     po3offpreint.po3com8.AddItem "Hours", 2
414     po3offpreint.po3com8.AddItem "Days", 3

415     Dim pop12u As Module
416     Dim pop12ui As Long
417     Dim pop12uv As String
418     pop12ui = m.Modules.Find(smFindTag, "pop12")
419     Set pop12u = m.Modules(pop12ui)
420     pop12uv = pop12u.Data("Units")

421     po3offpreint.po3com10.value = pop12uv
422     po3offpreint.po3com10.AddItem "Seconds", 0
423     po3offpreint.po3com10.AddItem "Minutes", 1
424     po3offpreint.po3com10.AddItem "Hours", 2
425     po3offpreint.po3com10.AddItem "Days", 3

426     Dim pop13u As Module
427     Dim pop13ui As Long
428     Dim pop13uv As String
429     pop13ui = m.Modules.Find(smFindTag, "pop13")
430     Set pop13u = m.Modules(pop13ui)
431     pop13uv = pop13u.Data("Units")

432     po3offpreint.po3com12.value = pop13uv
433     po3offpreint.po3com12.AddItem "Seconds", 0
434     po3offpreint.po3com12.AddItem "Minutes", 1
435     po3offpreint.po3com12.AddItem "Hours", 2
436     po3offpreint.po3com12.AddItem "Days", 3

437     Dim pop33u As Module
438     Dim pop33ui As Long
439     Dim pop33uv As String
440     pop33ui = m.Modules.Find(smFindTag, "pop33")
441     Set pop33u = m.Modules(pop33ui)
442     pop33uv = pop33u.Data("Units")

443     po3offpreint.po3com14.value = pop33uv
444     po3offpreint.po3com14.AddItem "Seconds", 0
```

```
    po3offpreint.po3com14.AddItem "Minutes", 1
    po3offpreint.po3com14.AddItem "Hours", 2
    po3offpreint.po3com14.AddItem "Days", 3

    po3offpreint.po3com24.value = pop33uv
    po3offpreint.po3com24.AddItem "Seconds", 0
    po3offpreint.po3com24.AddItem "Minutes", 1
    po3offpreint.po3com24.AddItem "Hours", 2
    po3offpreint.po3com24.AddItem "Days", 3

    Dim pop14u As Module
    Dim pop14ui As Long
    Dim pop14uv As String
    pop14ui = m.Modules.Find(smFindTag, "pop14")
    Set pop14u = m.Modules(pop14ui)
    pop14uv = pop14u.Data("Units")

    po3offpreint.po3com16.value = pop14uv
    po3offpreint.po3com16.AddItem "Seconds", 0
    po3offpreint.po3com16.AddItem "Minutes", 1
    po3offpreint.po3com16.AddItem "Hours", 2
    po3offpreint.po3com16.AddItem "Days", 3

    Dim pop15u As Module
    Dim pop15ui As Long
    Dim pop15uv As String
    pop15ui = m.Modules.Find(smFindTag, "pop15")
    Set pop15u = m.Modules(pop15ui)
    pop15uv = pop15u.Data("Units")

    po3offpreint.po3com18.value = pop15uv
    po3offpreint.po3com18.AddItem "Seconds", 0
    po3offpreint.po3com18.AddItem "Minutes", 1
    po3offpreint.po3com18.AddItem "Hours", 2
    po3offpreint.po3com18.AddItem "Days", 3

    Dim pop16u As Module
    Dim pop16ui As Long
    Dim pop16uv As String
    pop16ui = m.Modules.Find(smFindTag, "pop16")
    Set pop16u = m.Modules(pop16ui)
    pop16uv = pop16u.Data("Units")

    po3offpreint.po3com20.value = pop16uv
    po3offpreint.po3com20.AddItem "Seconds", 0
    po3offpreint.po3com20.AddItem "Minutes", 1
    po3offpreint.po3com20.AddItem "Hours", 2
    po3offpreint.po3com20.AddItem "Days", 3

    Dim pop17u As Module
    Dim pop17ui As Long
    Dim pop17uv As String
    pop17ui = m.Modules.Find(smFindTag, "pop17")
    Set pop17u = m.Modules(pop17ui)
    pop17uv = pop17u.Data("Units")

    po3offpreint.po3com22.value = pop17uv
    po3offpreint.po3com22.AddItem "Seconds", 0
    po3offpreint.po3com22.AddItem "Minutes", 1
    po3offpreint.po3com22.AddItem "Hours", 2
    po3offpreint.po3com22.AddItem "Days", 3
  End Sub
```

Project/po4offnopreint

```
Private Sub CommandButton6_Click()
  Me.Hide
  po1prelim.Show
```

```
    4   End Sub

    5   Private Sub CommandButton7_Click()
    6     Hierarchy.done06.Visible = True

    7     'code below checks to see if any option button sets are not clicked, and if so,
forces the user to make a choice
    8     Dim msgResult As Integer
    9     If (po4opt1.value = False And po4opt2.value = False) Then
   10       msgResult = MsgBox("You must make an integration orientation choice.  Click
Yes if integration takes place horizontally.  Click No if integration takes place
vertically.", vbYesNo)
   11       If msgResult = vbYes Then
   12         po4opt1.value = True
   13       Else
   14         po4opt2.value = True
   15       End If
   16     End If
   17     If (po4opt2.value = True And po4opt3.value = False And po4opt4.value = False)
Then
   18       msgResult = MsgBox("You must make a payload integration location decision.
Click Yes if the payload is integrated now, in the integration facility.  Click No if the
payload is integrated later, on the launch pad.", vbYesNo)
   19       If msgResult = vbYes Then
   20         po4opt3.value = True
   21       Else
   22         po4opt4.value = True
   23       End If
   24     End If
   25     If (po4opt1.value = True And po4opt5.value = False And po4opt6.value = False)
Then
   26       msgResult = MsgBox("You must make a payload integration location decision.
Click Yes if the payload is integrated now, in the integration facility.  Click No if the
payload is integrated later, on the launch pad.", vbYesNo)
   27       If msgResult = vbYes Then
   28         po4opt5.value = True
   29       Else
   30         po4opt6.value = True
   31       End If
   32     End If

   33     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
   34     Dim m As Model
   35     Set m = ThisDocument.Model

   36     Dim pop18 As Module
   37     Dim pop18i As Long
   38     pop18i = m.Modules.Find(smFindTag, "pop18")
   39     Set pop18 = m.Modules(pop18i)
   40     pop18.Data("Expression") = po4com1.Text
   41     pop18.Data("Units") = po4com2.Text

   42     Dim pop100 As Module
   43     Dim pop100i As Long
   44     pop100i = m.Modules.Find(smFindTag, "pop100")
   45     Set pop100 = m.Modules(pop100i)
   46     pop100.Data("Expression") = po4com3.Text
   47     pop100.Data("Units") = po4com4.Text

   48     Dim pop19 As Module
   49     Dim pop19i As Long
   50     pop19i = m.Modules.Find(smFindTag, "pop19")
   51     Set pop19 = m.Modules(pop19i)
   52     pop19.Data("Expression") = po4com5.Text
   53     pop19.Data("Units") = po4com6.Text

   54     Dim pop20 As Module
   55     Dim pop20i As Long
```

```
    pop20i = m.Modules.Find(smFindTag, "pop20")
    Set pop20 = m.Modules(pop20i)
    pop20.Data("Expression") = po4com7.Text
    pop20.Data("Units") = po4com8.Text

    Dim pop21 As Module
    Dim pop21i As Long
    pop21i = m.Modules.Find(smFindTag, "pop21")
    Set pop21 = m.Modules(pop21i)
    pop21.Data("Expression") = po4com9.Text
    pop21.Data("Units") = po4com10.Text

    Dim pop22 As Module
    Dim pop22i As Long
    pop22i = m.Modules.Find(smFindTag, "pop22")
    Set pop22 = m.Modules(pop22i)
    pop22.Data("Expression") = po4com11.Text
    pop22.Data("Units") = po4com12.Text

    Dim pop23 As Module
    Dim pop23i As Long
    pop23i = m.Modules.Find(smFindTag, "pop23")
    Set pop23 = m.Modules(pop23i)
    pop23.Data("Expression") = po4com13.Text
    pop23.Data("Units") = po4com14.Text

    Dim pop24 As Module
    Dim pop24i As Long
    pop24i = m.Modules.Find(smFindTag, "pop24")
    Set pop24 = m.Modules(pop24i)
    pop24.Data("Expression") = po4com25.Text
    pop24.Data("Units") = po4com26.Text

    Dim pop25 As Module
    Dim pop25i As Long
    pop25i = m.Modules.Find(smFindTag, "pop25")
    Set pop25 = m.Modules(pop25i)
    pop25.Data("Expression") = po4com27.Text
    pop25.Data("Units") = po4com28.Text

    Dim pop26 As Module
    Dim pop26i As Long
    pop26i = m.Modules.Find(smFindTag, "pop26")
    Set pop26 = m.Modules(pop26i)
    pop26.Data("Expression") = po4com29.Text
    pop26.Data("Units") = po4com30.Text

    Dim pop27 As Module
    Dim pop27i As Long
    pop27i = m.Modules.Find(smFindTag, "pop27")
    Set pop27 = m.Modules(pop27i)
    pop27.Data("Expression") = po4com31.Text
    pop27.Data("Units") = po4com32.Text

    Dim pop28 As Module
    Dim pop28i As Long
    pop28i = m.Modules.Find(smFindTag, "pop28")
    Set pop28 = m.Modules(pop28i)
    pop28.Data("Expression") = po4com33.Text
    pop28.Data("Units") = po4com34.Text

    Dim pop29 As Module
    Dim pop29i As Long
    pop29i = m.Modules.Find(smFindTag, "pop29")
    Set pop29 = m.Modules(pop29i)
    If po4frm3.Visible = True Then
      pop29.Data("Expression") = po4com15.Text
      pop29.Data("Units") = po4com16.Text
    Else
```

```
116       pop29.Data("Expression") = po4com35.Text
117       pop29.Data("Units") = po4com36.Text
118     End If

119     Dim pop30 As Module
120     Dim pop30i As Long
121     pop30i = m.Modules.Find(smFindTag, "pop30")
122     Set pop30 = m.Modules(pop30i)
123     If po4frm3.Visible = True Then
124       pop30.Data("Expression") = po4com17.Text
125       pop30.Data("Units") = po4com18.Text
126     Else
127       pop30.Data("Expression") = po4com37.Text
128       pop30.Data("Units") = po4com38.Text
129     End If

130     Dim pop31 As Module
131     Dim pop31i As Long
132     pop31i = m.Modules.Find(smFindTag, "pop31")
133     Set pop31 = m.Modules(pop31i)
134     If po4frm3.Visible = True Then
135       pop31.Data("Expression") = po4com19.Text
136       pop31.Data("Units") = po4com20.Text
137     Else
138       pop31.Data("Expression") = po4com39.Text
139       pop31.Data("Units") = po4com40.Text
140     End If

141     Dim pop32 As Module
142     Dim pop32i As Long
143     pop32i = m.Modules.Find(smFindTag, "pop32")
144     Set pop32 = m.Modules(pop32i)
145     If po4frm3.Visible = True Then
146       pop32.Data("Expression") = po4com21.Text
147       pop32.Data("Units") = po4com22.Text
148     Else
149       pop32.Data("Expression") = po4com41.Text
150       pop32.Data("Units") = po4com42.Text
151     End If

152     Dim pop33 As Module
153     Dim pop33i As Long
154     pop33i = m.Modules.Find(smFindTag, "pop33")
155     Set pop33 = m.Modules(pop33i)
156     If po4frm3.Visible = True Then
157       pop33.Data("Expression") = po4com23.Text
158       pop33.Data("Units") = po4com24.Text
159     Else
160       pop31.Data("Expression") = po4com43.Text
161       pop31.Data("Units") = po4com44.Text
162     End If

163     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
164     Dim pov4 As Module
165     Dim pov4i As Long
166     pov4i = m.Modules.Find(smFindTag, "pov4")
167     Set pov4 = m.Modules(pov4i)
168     If po4opt2.value = True Then
169       pov4.Data("Initial Value") = "1"
170     Else
171       pov4.Data("Initial Value") = "0"
172     End If

173     Dim pov5 As Module
174     Dim pov5i As Long
175     pov5i = m.Modules.Find(smFindTag, "pov5")
176     Set pov5 = m.Modules(pov5i)
177     If po4opt3.value = True And po4frm2.Visible = True Then
```

```
178       pov5.Data("Initial Value") = "1"
179     ElseIf po4opt5.value = True And po4frm5.Visible = True Then
180       pov5.Data("Initial Value") = "1"
181     Else
182       pov5.Data("Initial Value") = "0"
183     End If

184     'code below hides the current form and shows the next form in the sequence
185     Me.Hide
186     po5offhyper.Show

187   End Sub

188   Private Sub CommandButton9_Click()
189     Hierarchy.done06.Visible = True

190     'code below checks to see if any option button sets are not clicked, and if so,
forces the user to make a choice
191     Dim msgResult As Integer
192     If (po4opt1.value = False And po4opt2.value = False) Then
193       msgResult = MsgBox("You must make an integration orientation choice.  Click
Yes if integration takes place horizontally.  Click No if integration takes place
vertically.", vbYesNo)
194       If msgResult = vbYes Then
195         po4opt1.value = True
196       Else
197         po4opt2.value = True
198       End If
199     End If
200     If (po4opt2.value = True And po4opt3.value = False And po4opt4.value = False)
Then
201       msgResult = MsgBox("You must make a payload integration location decision.
Click Yes if the payload is integrated now, in the integration facility.  Click No if the
payload is integrated later, on the launch pad.", vbYesNo)
202       If msgResult = vbYes Then
203         po4opt3.value = True
204       Else
205         po4opt4.value = True
206       End If
207     End If
208     If (po4opt1.value = True And po4opt5.value = False And po4opt6.value = False)
Then
209       msgResult = MsgBox("You must make a payload integration location decision.
Click Yes if the payload is integrated now, in the integration facility.  Click No if the
payload is integrated later, on the launch pad.", vbYesNo)
210       If msgResult = vbYes Then
211         po4opt5.value = True
212       Else
213         po4opt6.value = True
214       End If
215     End If

216     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
217     Dim m As Model
218     Set m = ThisDocument.Model

219     Dim pop18 As Module
220     Dim pop18i As Long
221     pop18i = m.Modules.Find(smFindTag, "pop18")
222     Set pop18 = m.Modules(pop18i)
223     pop18.Data("Expression") = po4com1.Text
224     pop18.Data("Units") = po4com2.Text

225     Dim pop100 As Module
226     Dim pop100i As Long
227     pop100i = m.Modules.Find(smFindTag, "pop100")
228     Set pop100 = m.Modules(pop100i)
229     pop100.Data("Expression") = po4com3.Text
```

```
230     pop100.Data("Units") = po4com4.Text

231     Dim pop19 As Module
232     Dim pop19i As Long
233     pop19i = m.Modules.Find(smFindTag, "pop19")
234     Set pop19 = m.Modules(pop19i)
235     pop19.Data("Expression") = po4com5.Text
236     pop19.Data("Units") = po4com6.Text

237     Dim pop20 As Module
238     Dim pop20i As Long
239     pop20i = m.Modules.Find(smFindTag, "pop20")
240     Set pop20 = m.Modules(pop20i)
241     pop20.Data("Expression") = po4com7.Text
242     pop20.Data("Units") = po4com8.Text

243     Dim pop21 As Module
244     Dim pop21i As Long
245     pop21i = m.Modules.Find(smFindTag, "pop21")
246     Set pop21 = m.Modules(pop21i)
247     pop21.Data("Expression") = po4com9.Text
248     pop21.Data("Units") = po4com10.Text

249     Dim pop22 As Module
250     Dim pop22i As Long
251     pop22i = m.Modules.Find(smFindTag, "pop22")
252     Set pop22 = m.Modules(pop22i)
253     pop22.Data("Expression") = po4com11.Text
254     pop22.Data("Units") = po4com12.Text

255     Dim pop23 As Module
256     Dim pop23i As Long
257     pop23i = m.Modules.Find(smFindTag, "pop23")
258     Set pop23 = m.Modules(pop23i)
259     pop23.Data("Expression") = po4com13.Text
260     pop23.Data("Units") = po4com14.Text

261     Dim pop24 As Module
262     Dim pop24i As Long
263     pop24i = m.Modules.Find(smFindTag, "pop24")
264     Set pop24 = m.Modules(pop24i)
265     pop24.Data("Expression") = po4com25.Text
266     pop24.Data("Units") = po4com26.Text

267     Dim pop25 As Module
268     Dim pop25i As Long
269     pop25i = m.Modules.Find(smFindTag, "pop25")
270     Set pop25 = m.Modules(pop25i)
271     pop25.Data("Expression") = po4com27.Text
272     pop25.Data("Units") = po4com28.Text

273     Dim pop26 As Module
274     Dim pop26i As Long
275     pop26i = m.Modules.Find(smFindTag, "pop26")
276     Set pop26 = m.Modules(pop26i)
277     pop26.Data("Expression") = po4com29.Text
278     pop26.Data("Units") = po4com30.Text

279     Dim pop27 As Module
280     Dim pop27i As Long
281     pop27i = m.Modules.Find(smFindTag, "pop27")
282     Set pop27 = m.Modules(pop27i)
283     pop27.Data("Expression") = po4com31.Text
284     pop27.Data("Units") = po4com32.Text

285     Dim pop28 As Module
286     Dim pop28i As Long
287     pop28i = m.Modules.Find(smFindTag, "pop28")
288     Set pop28 = m.Modules(pop28i)
```

```
289     pop28.Data("Expression") = po4com33.Text
290     pop28.Data("Units") = po4com34.Text

291     Dim pop29 As Module
292     Dim pop29i As Long
293     pop29i = m.Modules.Find(smFindTag, "pop29")
294     Set pop29 = m.Modules(pop29i)
295     If po4frm3.Visible = True Then
296       pop29.Data("Expression") = po4com15.Text
297       pop29.Data("Units") = po4com16.Text
298     Else
299       pop29.Data("Expression") = po4com35.Text
300       pop29.Data("Units") = po4com36.Text
301     End If

302     Dim pop30 As Module
303     Dim pop30i As Long
304     pop30i = m.Modules.Find(smFindTag, "pop30")
305     Set pop30 = m.Modules(pop30i)
306     If po4frm3.Visible = True Then
307       pop30.Data("Expression") = po4com17.Text
308       pop30.Data("Units") = po4com18.Text
309     Else
310       pop30.Data("Expression") = po4com37.Text
311       pop30.Data("Units") = po4com38.Text
312     End If

313     Dim pop31 As Module
314     Dim pop31i As Long
315     pop31i = m.Modules.Find(smFindTag, "pop31")
316     Set pop31 = m.Modules(pop31i)
317     If po4frm3.Visible = True Then
318       pop31.Data("Expression") = po4com19.Text
319       pop31.Data("Units") = po4com20.Text
320     Else
321       pop31.Data("Expression") = po4com39.Text
322       pop31.Data("Units") = po4com40.Text
323     End If

324     Dim pop32 As Module
325     Dim pop32i As Long
326     pop32i = m.Modules.Find(smFindTag, "pop32")
327     Set pop32 = m.Modules(pop32i)
328     If po4frm3.Visible = True Then
329       pop32.Data("Expression") = po4com21.Text
330       pop32.Data("Units") = po4com22.Text
331     Else
332       pop32.Data("Expression") = po4com41.Text
333       pop32.Data("Units") = po4com42.Text
334     End If

335     Dim pop33 As Module
336     Dim pop33i As Long
337     pop33i = m.Modules.Find(smFindTag, "pop33")
338     Set pop33 = m.Modules(pop33i)
339     If po4frm3.Visible = True Then
340       pop33.Data("Expression") = po4com23.Text
341       pop33.Data("Units") = po4com24.Text
342     Else
343       pop31.Data("Expression") = po4com43.Text
344       pop31.Data("Units") = po4com44.Text

345     End If

346     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
347     Dim pov4 As Module
348     Dim pov4i As Long
349     pov4i = m.Modules.Find(smFindTag, "pov4")
```

```
350     Set pov4 = m.Modules(pov4i)
351     If po4opt2.value = True Then
352       pov4.Data("Initial Value") = "1"
353     Else
354       pov4.Data("Initial Value") = "0"
355     End If

356     Dim pov5 As Module
357     Dim pov5i As Long
358     pov5i = m.Modules.Find(smFindTag, "pov5")
359     Set pov5 = m.Modules(pov5i)
360     If po4opt3.value = True And po4frm2.Visible = True Then
361       pov5.Data("Initial Value") = "1"
362     ElseIf po4opt5.value = True And po4frm5.Visible = True Then
363       pov5.Data("Initial Value") = "1"
364     Else
365       pov5.Data("Initial Value") = "0"
366     End If

367     Me.Hide
368     Hierarchy.Show

369   End Sub

370   Private Sub Label1_Click()

371   End Sub

372   Private Sub Label11_Click()

373   End Sub

374   Private Sub Label12_Click()

375   End Sub

376   Private Sub Label15_Click()

377   End Sub

378   Private Sub Label31_Click()

379   End Sub

380   Private Sub OptionButton1_Click()

381   End Sub

382   Private Sub OptionButton2_Click()

383   End Sub

384   Private Sub OptionButton4_Click()

385   End Sub

386   Private Sub OptionButton6_Click()

387   End Sub

388   Private Sub po4opt1_Click()
389     po4frm4.Visible = True
390     po4frm5.Visible = True
391     po4frm6.Visible = True
392     po4frm1.Visible = False
393     po4frm2.Visible = False
394     po4frm3.Visible = False
395     po6erect.po6frm2.Visible = True
396     po6erect.po6frm1.Visible = False
```

```
397   End Sub

398   Private Sub po4opt2_Click()
399     po4frm4.Visible = False
400     po4frm5.Visible = False
401     po4frm6.Visible = False
402     po4frm1.Visible = True
403     po4frm2.Visible = True
404     po4frm3.Visible = True
405     po6erect.po6frm2.Visible = False
406     po6erect.po6frm1.Visible = True

407   End Sub

408   Private Sub po4opt3_Click()
409     po4frm3.Visible = True
410     po6erect.po6frm3.Visible = False
411   End Sub

412   Private Sub po4opt4_Click()
413     po4frm3.Visible = False
414     po6erect.po6frm3.Visible = True

415   End Sub

416   Private Sub po4opt5_Click()
417     po4frm6.Visible = True
418     po6erect.po6frm3.Visible = False
419   End Sub

420   Private Sub po4opt6_Click()
421     po4frm6.Visible = False
422     po6erect.po6frm3.Visible = True

423   End Sub

424   Private Sub ToggleButton1_Click()

425   End Sub

426   Private Sub UserForm_Click()

427   End Sub

428   Private Sub UserForm_Initialize()
429     Dim m As Model
430     Set m = ThisDocument.Model

431     'Code below populates large combo boxes for ON-02 thru ON-08
432     Dim pop18 As Module
433     Dim pop18i As Long
434     Dim pop18v As String
435     pop18i = m.Modules.Find(smFindTag, "pop18")
436     Set pop18 = m.Modules(pop18i)
437     pop18v = pop18.Data("Expression")

438     po4offnopreint.po4com1.value = pop18v
439     po4offnopreint.po4com1.AddItem "TRIA ( 54, 60, 84 )", 0
440     po4offnopreint.po4com1.AddItem "TRIA ( Min, Mode, Max )", 1
441     po4offnopreint.po4com1.AddItem "NORM ( Mean, StdDev )", 2
442     po4offnopreint.po4com1.AddItem "EXPO ( Mean )", 3
443     po4offnopreint.po4com1.AddItem "UNIF ( Min, Max )", 4

444     Dim pop100 As Module
445     Dim pop100i As Long
446     Dim pop100v As String
447     pop100i = m.Modules.Find(smFindTag, "pop100")
448     Set pop100 = m.Modules(pop100i)
```

```
449     pop100v = pop100.Data("Expression")

450     po4offnopreint.po4com3.value = pop100v
451     po4offnopreint.po4com3.AddItem "TRIA ( 108, 120, 168 )", 0
452     po4offnopreint.po4com3.AddItem "TRIA ( Min, Mode, Max )", 1
453     po4offnopreint.po4com3.AddItem "NORM ( Mean, StdDev )", 2
454     po4offnopreint.po4com3.AddItem "EXPO ( Mean )", 3
455     po4offnopreint.po4com3.AddItem "UNIF ( Min, Max )", 4

456     Dim pop19 As Module
457     Dim pop19i As Long
458     Dim pop19v As String
459     pop19i = m.Modules.Find(smFindTag, "pop19")
460     Set pop19 = m.Modules(pop19i)
461     pop19v = pop19.Data("Expression")

462     po4offnopreint.po4com5.value = pop19v
463     po4offnopreint.po4com5.AddItem "TRIA ( 27, 30, 42 )", 0
464     po4offnopreint.po4com5.AddItem "TRIA ( Min, Mode, Max )", 1
465     po4offnopreint.po4com5.AddItem "NORM ( Mean, StdDev )", 2
466     po4offnopreint.po4com5.AddItem "EXPO ( Mean )", 3
467     po4offnopreint.po4com5.AddItem "UNIF ( Min, Max )", 4

468     Dim pop20 As Module
469     Dim pop20i As Long
470     Dim pop20v As String
471     pop20i = m.Modules.Find(smFindTag, "pop20")
472     Set pop20 = m.Modules(pop20i)
473     pop20v = pop20.Data("Expression")

474     po4offnopreint.po4com7.value = pop20v
475     po4offnopreint.po4com7.AddItem "TRIA ( 27, 30, 42 )", 0
476     po4offnopreint.po4com7.AddItem "TRIA ( Min, Mode, Max )", 1
477     po4offnopreint.po4com7.AddItem "NORM ( Mean, StdDev )", 2
478     po4offnopreint.po4com7.AddItem "EXPO ( Mean )", 3
479     po4offnopreint.po4com7.AddItem "UNIF ( Min, Max )", 4

480     Dim pop21 As Module
481     Dim pop21i As Long
482     Dim pop21v As String
483     pop21i = m.Modules.Find(smFindTag, "pop21")
484     Set pop21 = m.Modules(pop21i)
485     pop21v = pop21.Data("Expression")

486     po4offnopreint.po4com9.value = pop21v
487     po4offnopreint.po4com9.AddItem "TRIA ( 36, 40, 56 )", 0
488     po4offnopreint.po4com9.AddItem "TRIA ( Min, Mode, Max )", 1
489     po4offnopreint.po4com9.AddItem "NORM ( Mean, StdDev )", 2
490     po4offnopreint.po4com9.AddItem "EXPO ( Mean )", 3
491     po4offnopreint.po4com9.AddItem "UNIF ( Min, Max )", 4

492     Dim pop22 As Module
493     Dim pop22i As Long
494     Dim pop22v As String
495     pop22i = m.Modules.Find(smFindTag, "pop22")
496     Set pop22 = m.Modules(pop22i)
497     pop22v = pop22.Data("Expression")

498     po4offnopreint.po4com11.value = pop22v
499     po4offnopreint.po4com11.AddItem "TRIA ( 36, 40, 56 )", 0
500     po4offnopreint.po4com11.AddItem "TRIA ( Min, Mode, Max )", 1
501     po4offnopreint.po4com11.AddItem "NORM ( Mean, StdDev )", 2
502     po4offnopreint.po4com11.AddItem "EXPO ( Mean )", 3
503     po4offnopreint.po4com11.AddItem "UNIF ( Min, Max )", 4

504     Dim pop23 As Module
505     Dim pop23i As Long
506     Dim pop23v As String
507     pop23i = m.Modules.Find(smFindTag, "pop23")
```

```
    Set pop23 = m.Modules(pop23i)
    pop23v = pop23.Data("Expression")

    po4offnopreint.po4com13.value = pop23v
    po4offnopreint.po4com13.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com13.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com13.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com13.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com13.AddItem "UNIF ( Min, Max )", 4

    'Code below populates large combo boxes for ON-17 thru ON-21
    Dim pop24 As Module
    Dim pop24i As Long
    Dim pop24v As String
    pop24i = m.Modules.Find(smFindTag, "pop24")
    Set pop24 = m.Modules(pop24i)
    pop24v = pop24.Data("Expression")

    po4offnopreint.po4com25.value = pop24v
    po4offnopreint.po4com25.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com25.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com25.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com25.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com25.AddItem "UNIF ( Min, Max )", 4

    Dim pop25 As Module
    Dim pop25i As Long
    Dim pop25v As String
    pop25i = m.Modules.Find(smFindTag, "pop25")
    Set pop25 = m.Modules(pop25i)
    pop25v = pop25.Data("Expression")

    po4offnopreint.po4com27.value = pop25v
    po4offnopreint.po4com27.AddItem "TRIA ( 54, 60, 84 )", 0
    po4offnopreint.po4com27.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com27.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com27.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com27.AddItem "UNIF ( Min, Max )", 4

    Dim pop26 As Module
    Dim pop26i As Long
    Dim pop26v As String
    pop26i = m.Modules.Find(smFindTag, "pop26")
    Set pop26 = m.Modules(pop26i)
    pop26v = pop26.Data("Expression")

    po4offnopreint.po4com29.value = pop26v
    po4offnopreint.po4com29.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com29.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com29.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com29.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com29.AddItem "UNIF ( Min, Max )", 4

    Dim pop27 As Module
    Dim pop27i As Long
    Dim pop27v As String
    pop27i = m.Modules.Find(smFindTag, "pop27")
    Set pop27 = m.Modules(pop27i)
    pop27v = pop27.Data("Expression")

    po4offnopreint.po4com31.value = pop27v
    po4offnopreint.po4com31.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com31.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com31.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com31.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com31.AddItem "UNIF ( Min, Max )", 4

    Dim pop28 As Module
    Dim pop28i As Long
```

```
    Dim pop28v As String
    pop28i = m.Modules.Find(smFindTag, "pop28")
    Set pop28 = m.Modules(pop28i)
    pop28v = pop28.Data("Expression")

    po4offnopreint.po4com33.value = pop28v
    po4offnopreint.po4com33.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com33.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com33.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com33.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com33.AddItem "UNIF ( Min, Max )", 4

    'Code below populates small combo boxes for ON-02 thru ON-08
    Dim pop18u As Module
    Dim pop18ui As Long
    Dim pop18uv As String
    pop18ui = m.Modules.Find(smFindTag, "pop18")
    Set pop18u = m.Modules(pop18ui)
    pop18uv = pop18u.Data("Units")

    po4offnopreint.po4com2.value = pop18uv
    po4offnopreint.po4com2.AddItem "Seconds", 0
    po4offnopreint.po4com2.AddItem "Minutes", 1
    po4offnopreint.po4com2.AddItem "Hours", 2
    po4offnopreint.po4com2.AddItem "Days", 3

    Dim pop100u As Module
    Dim pop100ui As Long
    Dim pop100uv As String
    pop100ui = m.Modules.Find(smFindTag, "pop100")
    Set pop100u = m.Modules(pop100ui)
    pop100uv = pop100u.Data("Units")

    po4offnopreint.po4com4.value = pop100uv
    po4offnopreint.po4com4.AddItem "Seconds", 0
    po4offnopreint.po4com4.AddItem "Minutes", 1
    po4offnopreint.po4com4.AddItem "Hours", 2
    po4offnopreint.po4com4.AddItem "Days", 3

    Dim pop19u As Module
    Dim pop19ui As Long
    Dim pop19uv As String
    pop19ui = m.Modules.Find(smFindTag, "pop19")
    Set pop19u = m.Modules(pop19ui)
    pop19uv = pop19u.Data("Units")

    po4offnopreint.po4com6.value = pop19uv
    po4offnopreint.po4com6.AddItem "Seconds", 0
    po4offnopreint.po4com6.AddItem "Minutes", 1
    po4offnopreint.po4com6.AddItem "Hours", 2
    po4offnopreint.po4com6.AddItem "Days", 3

    Dim pop20u As Module
    Dim pop20ui As Long
    Dim pop20uv As String
    pop20ui = m.Modules.Find(smFindTag, "pop20")
    Set pop20u = m.Modules(pop20ui)
    pop20uv = pop20u.Data("Units")

    po4offnopreint.po4com8.value = pop20uv
    po4offnopreint.po4com8.AddItem "Seconds", 0
    po4offnopreint.po4com8.AddItem "Minutes", 1
    po4offnopreint.po4com8.AddItem "Hours", 2
    po4offnopreint.po4com8.AddItem "Days", 3

    Dim pop21u As Module
    Dim pop21ui As Long
    Dim pop21uv As String
    pop21ui = m.Modules.Find(smFindTag, "pop21")
```

```
626     Set pop21u = m.Modules(pop21ui)
627     pop21uv = pop21u.Data("Units")

628     po4offnopreint.po4com10.value = pop21uv
629     po4offnopreint.po4com10.AddItem "Seconds", 0
630     po4offnopreint.po4com10.AddItem "Minutes", 1
631     po4offnopreint.po4com10.AddItem "Hours", 2
632     po4offnopreint.po4com10.AddItem "Days", 3

633     Dim pop22u As Module
634     Dim pop22ui As Long
635     Dim pop22uv As String
636     pop22ui = m.Modules.Find(smFindTag, "pop22")
637     Set pop22u = m.Modules(pop22ui)
638     pop22uv = pop22u.Data("Units")

639     po4offnopreint.po4com12.value = pop22uv
640     po4offnopreint.po4com12.AddItem "Seconds", 0
641     po4offnopreint.po4com12.AddItem "Minutes", 1
642     po4offnopreint.po4com12.AddItem "Hours", 2
643     po4offnopreint.po4com12.AddItem "Days", 3

644     Dim pop23u As Module
645     Dim pop23ui As Long
646     Dim pop23uv As String
647     pop23ui = m.Modules.Find(smFindTag, "pop23")
648     Set pop23u = m.Modules(pop23ui)
649     pop23uv = pop23u.Data("Units")

650     po4offnopreint.po4com14.value = pop23uv
651     po4offnopreint.po4com14.AddItem "Seconds", 0
652     po4offnopreint.po4com14.AddItem "Minutes", 1
653     po4offnopreint.po4com14.AddItem "Hours", 2
654     po4offnopreint.po4com14.AddItem "Days", 3

655     'Code below populates small combo boxes for ON-17 thru ON-21
656     Dim pop24u As Module
657     Dim pop24ui As Long
658     Dim pop24uv As String
659     pop24ui = m.Modules.Find(smFindTag, "pop24")
660     Set pop24u = m.Modules(pop24ui)
661     pop24uv = pop24u.Data("Units")

662     po4offnopreint.po4com26.value = pop24uv
663     po4offnopreint.po4com26.AddItem "Seconds", 0
664     po4offnopreint.po4com26.AddItem "Minutes", 1
665     po4offnopreint.po4com26.AddItem "Hours", 2
666     po4offnopreint.po4com26.AddItem "Days", 3

667     Dim pop25u As Module
668     Dim pop25ui As Long
669     Dim pop25uv As String
670     pop25ui = m.Modules.Find(smFindTag, "pop25")
671     Set pop25u = m.Modules(pop25ui)
672     pop25uv = pop25u.Data("Units")

673     po4offnopreint.po4com28.value = pop25uv
674     po4offnopreint.po4com28.AddItem "Seconds", 0
675     po4offnopreint.po4com28.AddItem "Minutes", 1
676     po4offnopreint.po4com28.AddItem "Hours", 2
677     po4offnopreint.po4com28.AddItem "Days", 3

678     Dim pop26u As Module
679     Dim pop26ui As Long
680     Dim pop26uv As String
681     pop26ui = m.Modules.Find(smFindTag, "pop26")
682     Set pop26u = m.Modules(pop26ui)
683     pop26uv = pop26u.Data("Units")
```

```
    po4offnopreint.po4com30.value = pop26uv
    po4offnopreint.po4com30.AddItem "Seconds", 0
    po4offnopreint.po4com30.AddItem "Minutes", 1
    po4offnopreint.po4com30.AddItem "Hours", 2
    po4offnopreint.po4com30.AddItem "Days", 3

    Dim pop27u As Module
    Dim pop27ui As Long
    Dim pop27uv As String
    pop27ui = m.Modules.Find(smFindTag, "pop27")
    Set pop27u = m.Modules(pop27ui)
    pop27uv = pop27u.Data("Units")

    po4offnopreint.po4com32.value = pop27uv
    po4offnopreint.po4com32.AddItem "Seconds", 0
    po4offnopreint.po4com32.AddItem "Minutes", 1
    po4offnopreint.po4com32.AddItem "Hours", 2
    po4offnopreint.po4com32.AddItem "Days", 3

    Dim pop28u As Module
    Dim pop28ui As Long
    Dim pop28uv As String
    pop28ui = m.Modules.Find(smFindTag, "pop28")
    Set pop28u = m.Modules(pop28ui)
    pop28uv = pop28u.Data("Units")

    po4offnopreint.po4com34.value = pop28uv
    po4offnopreint.po4com34.AddItem "Seconds", 0
    po4offnopreint.po4com34.AddItem "Minutes", 1
    po4offnopreint.po4com34.AddItem "Hours", 2
    po4offnopreint.po4com34.AddItem "Days", 3

    'Code below populates large combo boxes for ON-12 thru ON-16 and ON-25 thru ON-
29
    Dim pop29 As Module
    Dim pop29i As Long
    Dim pop29v As String
    pop29i = m.Modules.Find(smFindTag, "pop29")
    Set pop29 = m.Modules(pop29i)
    pop29v = pop29.Data("Expression")

    po4offnopreint.po4com15.value = pop29v
    po4offnopreint.po4com15.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com15.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com15.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com15.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com15.AddItem "UNIF ( Min, Max )", 4

    po4offnopreint.po4com35.value = pop29v
    po4offnopreint.po4com35.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com35.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com35.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com35.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com35.AddItem "UNIF ( Min, Max )", 4

    Dim pop30 As Module
    Dim pop30i As Long
    Dim pop30v As String
    pop30i = m.Modules.Find(smFindTag, "pop30")
    Set pop30 = m.Modules(pop30i)
    pop30v = pop30.Data("Expression")

    po4offnopreint.po4com17.value = pop30v
    po4offnopreint.po4com17.AddItem "TRIA ( 81, 90, 126 )", 0
    po4offnopreint.po4com17.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com17.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com17.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com17.AddItem "UNIF ( Min, Max )", 4
```

```
    po4offnopreint.po4com37.value = pop30v
    po4offnopreint.po4com37.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com37.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com37.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com37.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com37.AddItem "UNIF ( Min, Max )", 4

    Dim pop31 As Module
    Dim pop31i As Long
    Dim pop31v As String
    pop31i = m.Modules.Find(smFindTag, "pop31")
    Set pop31 = m.Modules(pop31i)
    pop31v = pop31.Data("Expression")

    po4offnopreint.po4com19.value = pop31v
    po4offnopreint.po4com19.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com19.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com19.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com19.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com19.AddItem "UNIF ( Min, Max )", 4

    po4offnopreint.po4com39.value = pop31v
    po4offnopreint.po4com39.AddItem "TRIA ( 18, 20, 28 )", 0
    po4offnopreint.po4com39.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com39.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com39.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com39.AddItem "UNIF ( Min, Max )", 4

    Dim pop32 As Module
    Dim pop32i As Long
    Dim pop32v As String
    pop32i = m.Modules.Find(smFindTag, "pop32")
    Set pop32 = m.Modules(pop32i)
    pop32v = pop32.Data("Expression")

    po4offnopreint.po4com21.value = pop32v
    po4offnopreint.po4com21.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com21.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com21.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com21.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com21.AddItem "UNIF ( Min, Max )", 4

    po4offnopreint.po4com41.value = pop32v
    po4offnopreint.po4com41.AddItem "TRIA ( 18, 20, 28 )", 0
    po4offnopreint.po4com41.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com41.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com41.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com41.AddItem "UNIF ( Min, Max )", 4

    Dim pop33 As Module
    Dim pop33i As Long
    Dim pop33v As String
    pop33i = m.Modules.Find(smFindTag, "pop33")
    Set pop33 = m.Modules(pop33i)
    pop33v = pop33.Data("Expression")

    po4offnopreint.po4com23.value = pop33v
    po4offnopreint.po4com23.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com23.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com23.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com23.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com23.AddItem "UNIF ( Min, Max )", 4

    po4offnopreint.po4com43.value = pop33v
    po4offnopreint.po4com43.AddItem "TRIA ( 27, 30, 42 )", 0
    po4offnopreint.po4com43.AddItem "TRIA ( Min, Mode, Max )", 1
    po4offnopreint.po4com43.AddItem "NORM ( Mean, StdDev )", 2
    po4offnopreint.po4com43.AddItem "EXPO ( Mean )", 3
    po4offnopreint.po4com43.AddItem "UNIF ( Min, Max )", 4
```

```
802     'Code below populates small combo boxes for ON-12 thru ON-16 and ON-25 thru ON-
29
803     Dim pop29u As Module
804     Dim pop29ui As Long
805     Dim pop29uv As String
806     pop29ui = m.Modules.Find(smFindTag, "pop29")
807     Set pop29u = m.Modules(pop29ui)
808     pop29uv = pop29u.Data("Units")

809     po4offnopreint.po4com16.value = pop29uv
810     po4offnopreint.po4com16.AddItem "Seconds", 0
811     po4offnopreint.po4com16.AddItem "Minutes", 1
812     po4offnopreint.po4com16.AddItem "Hours", 2
813     po4offnopreint.po4com16.AddItem "Days", 3

814     po4offnopreint.po4com36.value = pop29uv
815     po4offnopreint.po4com36.AddItem "Seconds", 0
816     po4offnopreint.po4com36.AddItem "Minutes", 1
817     po4offnopreint.po4com36.AddItem "Hours", 2
818     po4offnopreint.po4com36.AddItem "Days", 3

819     Dim pop30u As Module
820     Dim pop30ui As Long
821     Dim pop30uv As String
822     pop30ui = m.Modules.Find(smFindTag, "pop30")
823     Set pop30u = m.Modules(pop30ui)
824     pop30uv = pop30u.Data("Units")

825     po4offnopreint.po4com18.value = pop30uv
826     po4offnopreint.po4com18.AddItem "Seconds", 0
827     po4offnopreint.po4com18.AddItem "Minutes", 1
828     po4offnopreint.po4com18.AddItem "Hours", 2
829     po4offnopreint.po4com18.AddItem "Days", 3

830     po4offnopreint.po4com38.value = pop30uv
831     po4offnopreint.po4com38.AddItem "Seconds", 0
832     po4offnopreint.po4com38.AddItem "Minutes", 1
833     po4offnopreint.po4com38.AddItem "Hours", 2
834     po4offnopreint.po4com38.AddItem "Days", 3

835     Dim pop31u As Module
836     Dim pop31ui As Long
837     Dim pop31uv As String
838     pop31ui = m.Modules.Find(smFindTag, "pop31")
839     Set pop31u = m.Modules(pop31ui)
840     pop31uv = pop31u.Data("Units")

841     po4offnopreint.po4com20.value = pop31uv
842     po4offnopreint.po4com20.AddItem "Seconds", 0
843     po4offnopreint.po4com20.AddItem "Minutes", 1
844     po4offnopreint.po4com20.AddItem "Hours", 2
845     po4offnopreint.po4com20.AddItem "Days", 3

846     po4offnopreint.po4com40.value = pop31uv
847     po4offnopreint.po4com40.AddItem "Seconds", 0
848     po4offnopreint.po4com40.AddItem "Minutes", 1
849     po4offnopreint.po4com40.AddItem "Hours", 2
850     po4offnopreint.po4com40.AddItem "Days", 3

851     Dim pop32u As Module
852     Dim pop32ui As Long
853     Dim pop32uv As String
854     pop32ui = m.Modules.Find(smFindTag, "pop32")
855     Set pop32u = m.Modules(pop32ui)
856     pop32uv = pop32u.Data("Units")

857     po4offnopreint.po4com22.value = pop32uv
858     po4offnopreint.po4com22.AddItem "Seconds", 0
```

```
  859     po4offnopreint.po4com22.AddItem "Minutes", 1
  860     po4offnopreint.po4com22.AddItem "Hours", 2
  861     po4offnopreint.po4com22.AddItem "Days", 3

  862     po4offnopreint.po4com42.value = pop32uv
  863     po4offnopreint.po4com42.AddItem "Seconds", 0
  864     po4offnopreint.po4com42.AddItem "Minutes", 1
  865     po4offnopreint.po4com42.AddItem "Hours", 2
  866     po4offnopreint.po4com42.AddItem "Days", 3

  867     Dim pop33u As Module
  868     Dim pop33ui As Long
  869     Dim pop33uv As String
  870     pop33ui = m.Modules.Find(smFindTag, "pop33")
  871     Set pop33u = m.Modules(pop33ui)
  872     pop33uv = pop33u.Data("Units")

  873     po4offnopreint.po4com24.value = pop33uv
  874     po4offnopreint.po4com24.AddItem "Seconds", 0
  875     po4offnopreint.po4com24.AddItem "Minutes", 1
  876     po4offnopreint.po4com24.AddItem "Hours", 2
  877     po4offnopreint.po4com24.AddItem "Days", 3

  878     po4offnopreint.po4com44.value = pop33uv
  879     po4offnopreint.po4com44.AddItem "Seconds", 0
  880     po4offnopreint.po4com44.AddItem "Minutes", 1
  881     po4offnopreint.po4com44.AddItem "Hours", 2
  882     po4offnopreint.po4com44.AddItem "Days", 3

  883   End Sub
```

Project/po5offhyper

```
    1   Private Sub ComboBox1_Change()

    2   End Sub

    3   Private Sub ComboBox11_Change()

    4   End Sub

    5   Private Sub CommandButton6_Click()
    6     Me.Hide
    7     If po1prelim.po1opt1.value = True Then
    8       po3offpreint.Show
    9     Else
   10       po4offnopreint.Show
   11     End If

   12   End Sub

   13   Private Sub CommandButton7_Click()
   14     Hierarchy.done07.Visible = True

   15     'Code below checks if any option button sets are not clicked, and if so, forces
the user to make a decision
   16     Dim msgResult As Integer
   17     If (po5opt1.value = False And po5opt2.value = False) Then
   18       msgResult = MsgBox("You must make a hypergolic fuels decision.  Are
hypergolic fuels required?", vbYesNo)
   19       If msgResult = vbYes Then
   20         po5opt1.value = True
   21       Else
   22         po5opt2.value = True
   23       End If
   24     End If
   25     If (po5opt1.value = True And po5opt3.value = False And po5opt4.value = False)
Then
   26       msgResult = MsgBox("You must make a hypergolic fuels loading decision.  Click
```

```
Yes if hypergolics are loaded now, in the integration facility.  Click No if hypergolics
are loaded later, on the launch pad.", vbYesNo)
   27       If msgResult = vbYes Then
   28         po5opt3.value = True
   29       Else
   30         po5opt4.value = True
   31       End If
   32     End If
   33     If (po5opt5.value = False And po5opt6.value = False) Then
   34       msgResult = MsgBox("You must make an ordnance decision.  Is ordnance
required?", vbYesNo)
   35       If msgResult = vbYes Then
   36         po5opt5.value = True
   37       Else
   38         po5opt6.value = True
   39       End If
   40     End If
   41     If (po5opt5.value = True And po5opt7.value = False And po5opt8.value = False)
Then
   42       msgResult = MsgBox("You must make an ordnance installation location decision.
Click Yes if ordnance is loaded now, in the integration facility.  Click No if ordnance
is loaded later, on the launch pad.", vbYesNo)
   43       If msgResult = vbYes Then
   44         po5opt7.value = True
   45       Else
   46         po5opt8.value = True
   47       End If
   48     End If

   49     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
   50     Dim m As Model
   51     Set m = ThisDocument.Model

   52     Dim pop34 As Module
   53     Dim pop34i As Long
   54     pop34i = m.Modules.Find(smFindTag, "pop34")
   55     Set pop34 = m.Modules(pop34i)
   56     pop34.Data("Expression") = po5com1.Text
   57     pop34.Data("Units") = po5com2.Text

   58     Dim pop71 As Module
   59     Dim pop71i As Long
   60     pop71i = m.Modules.Find(smFindTag, "pop71")
   61     Set pop71 = m.Modules(pop71i)
   62     pop71.Data("Expression") = po5com1.Text
   63     pop71.Data("Units") = po5com2.Text

   64     Dim pop35 As Module
   65     Dim pop35i As Long
   66     pop35i = m.Modules.Find(smFindTag, "pop35")
   67     Set pop35 = m.Modules(pop35i)
   68     pop35.Data("Expression") = po5com3.Text
   69     pop35.Data("Units") = po5com4.Text

   70     Dim pop77 As Module
   71     Dim pop77i As Long
   72     pop77i = m.Modules.Find(smFindTag, "pop77")
   73     Set pop77 = m.Modules(pop77i)
   74     pop77.Data("Expression") = po5com3.Text
   75     pop77.Data("Units") = po5com4.Text

   76     Dim pop36 As Module
   77     Dim pop36i As Long
   78     pop36i = m.Modules.Find(smFindTag, "pop36")
   79     Set pop36 = m.Modules(pop36i)
   80     pop36.Data("Expression") = po5com5.Text
   81     pop36.Data("Units") = po5com6.Text
```

```
    Dim pop37 As Module
    Dim pop37i As Long
    pop37i = m.Modules.Find(smFindTag, "pop37")
    Set pop37 = m.Modules(pop37i)
    pop37.Data("Expression") = po5com7.Text
    pop37.Data("Units") = po5com8.Text

    Dim pop38 As Module
    Dim pop38i As Long
    pop38i = m.Modules.Find(smFindTag, "pop38")
    Set pop38 = m.Modules(pop38i)
    pop38.Data("Expression") = po5com9.Text
    pop38.Data("Units") = po5com10.Text

    'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
    Dim pov6 As Module
    Dim pov6i As Long
    pov6i = m.Modules.Find(smFindTag, "pov6")
    Set pov6 = m.Modules(pov6i)
    If po5opt2.value = True Then
      pov6.Data("Initial Value") = "0"
    ElseIf po5opt3.value = True Then
      pov6.Data("Initial Value") = "1"
    Else
      pov6.Data("Initial Value") = "2"
    End If

    Dim pov7 As Module
    Dim pov7i As Long
    pov7i = m.Modules.Find(smFindTag, "pov7")
    Set pov7 = m.Modules(pov7i)
    If po5opt6.value = True Then
      pov7.Data("Initial Value") = "0"
    ElseIf po5opt7.value = True Then
      pov7.Data("Initial Value") = "1"
    Else
      pov7.Data("Initial Value") = "2"
    End If

    'code below hides the current form and shows the next form in the sequence
    Me.Hide
    po6erect.Show

  End Sub

  Private Sub CommandButton9_Click()
    Hierarchy.done07.Visible = True

    'Code below checks if any option button sets are not clicked, and if so, forces
the user to make a decision
    Dim msgResult As Integer
    If (po5opt1.value = False And po5opt2.value = False) Then
      msgResult = MsgBox("You must make a hypergolic fuels decision.  Are
hypergolic fuels required?", vbYesNo)
      If msgResult = vbYes Then
        po5opt1.value = True
      Else
        po5opt2.value = True
      End If
    End If
    If (po5opt1.value = True And po5opt3.value = False And po5opt4.value = False)
Then
      msgResult = MsgBox("You must make a hypergolic fuels loading decision.  Click
Yes if hypergolics are loaded now, in the integration facility.  Click No if hypergolics
are loaded later, on the launch pad.", vbYesNo)
      If msgResult = vbYes Then
        po5opt3.value = True
      Else
```

```
  138         po5opt4.value = True
  139       End If
  140     End If
  141     If (po5opt5.value = False And po5opt6.value = False) Then
  142       msgResult = MsgBox("You must make an ordnance decision.  Is ordnance
required?", vbYesNo)
  143       If msgResult = vbYes Then
  144         po5opt5.value = True
  145       Else
  146         po5opt6.value = True
  147       End If
  148     End If
  149     If (po5opt5.value = True And po5opt7.value = False And po5opt8.value = False)
Then
  150       msgResult = MsgBox("You must make an ordnance installation location decision.
Click Yes if ordnance is loaded now, in the integration facility.  Click No if ordnance
is loaded later, on the launch pad.", vbYesNo)
  151       If msgResult = vbYes Then
  152         po5opt7.value = True
  153       Else
  154         po5opt8.value = True
  155       End If
  156     End If

  157     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
  158     Dim m As Model
  159     Set m = ThisDocument.Model

  160     Dim pop34 As Module
  161     Dim pop34i As Long
  162     pop34i = m.Modules.Find(smFindTag, "pop34")
  163     Set pop34 = m.Modules(pop34i)
  164     pop34.Data("Expression") = po5com1.Text
  165     pop34.Data("Units") = po5com2.Text

  166     Dim pop71 As Module
  167     Dim pop71i As Long
  168     pop71i = m.Modules.Find(smFindTag, "pop71")
  169     Set pop71 = m.Modules(pop71i)
  170     pop71.Data("Expression") = po5com1.Text
  171     pop71.Data("Units") = po5com2.Text

  172     Dim pop35 As Module
  173     Dim pop35i As Long
  174     pop35i = m.Modules.Find(smFindTag, "pop35")
  175     Set pop35 = m.Modules(pop35i)
  176     pop35.Data("Expression") = po5com3.Text
  177     pop35.Data("Units") = po5com4.Text

  178     Dim pop77 As Module
  179     Dim pop77i As Long
  180     pop77i = m.Modules.Find(smFindTag, "pop77")
  181     Set pop77 = m.Modules(pop77i)
  182     pop77.Data("Expression") = po5com3.Text
  183     pop77.Data("Units") = po5com4.Text

  184     Dim pop36 As Module
  185     Dim pop36i As Long
  186     pop36i = m.Modules.Find(smFindTag, "pop36")
  187     Set pop36 = m.Modules(pop36i)
  188     pop36.Data("Expression") = po5com5.Text
  189     pop36.Data("Units") = po5com6.Text

  190     Dim pop37 As Module
  191     Dim pop37i As Long
  192     pop37i = m.Modules.Find(smFindTag, "pop37")
  193     Set pop37 = m.Modules(pop37i)
  194     pop37.Data("Expression") = po5com7.Text
```

```
195     pop37.Data("Units") = po5com8.Text

196     Dim pop38 As Module
197     Dim pop38i As Long
198     pop38i = m.Modules.Find(smFindTag, "pop38")
199     Set pop38 = m.Modules(pop38i)
200     pop38.Data("Expression") = po5com9.Text
201     pop38.Data("Units") = po5com10.Text

202     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
203     Dim pov6 As Module
204     Dim pov6i As Long
205     pov6i = m.Modules.Find(smFindTag, "pov6")
206     Set pov6 = m.Modules(pov6i)
207     If po5opt2.value = True Then
208       pov6.Data("Initial Value") = "0"
209     ElseIf po5opt3.value = True Then
210       pov6.Data("Initial Value") = "1"
211     Else
212       pov6.Data("Initial Value") = "2"
213     End If

214     Dim pov7 As Module
215     Dim pov7i As Long
216     pov7i = m.Modules.Find(smFindTag, "pov7")
217     Set pov7 = m.Modules(pov7i)
218     If po5opt6.value = True Then
219       pov7.Data("Initial Value") = "0"
220     ElseIf po5opt7.value = True Then
221       pov7.Data("Initial Value") = "1"
222     Else
223       pov7.Data("Initial Value") = "2"
224     End If

225     Me.Hide
226     Hierarchy.Show

227   End Sub

228   Private Sub Label11_Click()

229   End Sub

230   Private Sub Label12_Click()

231   End Sub

232   Private Sub OptionButton1_Click()

233   End Sub

234   Private Sub OptionButton2_Click()

235   End Sub

236   Private Sub OptionButton4_Click()

237   End Sub

238   Private Sub OptionButton6_Click()

239   End Sub

240   Private Sub po5opt1_Click()
241     po5frm1.Visible = True

242   End Sub
```

```
243   Private Sub po5opt2_Click()
244     po5frm1.Visible = False

245   End Sub

246   Private Sub po5opt5_Click()
247     po5frm2.Visible = True

248   End Sub

249   Private Sub po5opt6_Click()
250     po5frm2.Visible = False

251   End Sub

252   Private Sub ToggleButton1_Click()

253   End Sub

254   Private Sub ToggleButton4_Click()

255   End Sub

256   Private Sub UserForm_Click()

257   End Sub

258   Private Sub UserForm_Initialize()
259     Dim m As Model
260     Set m = ThisDocument.Model

261     'Code below populates large combo boxes for OT-03, OT-06 and OT-07 thru OT-09
262     Dim pop34 As Module
263     Dim pop34i As Long
264     Dim pop34v As String
265     pop34i = m.Modules.Find(smFindTag, "pop34")
266     Set pop34 = m.Modules(pop34i)
267     pop34v = pop34.Data("Expression")

268     po5offhyper.po5com1.value = pop34v
269     po5offhyper.po5com1.AddItem "TRIA ( 756, 840, 1176 )", 0
270     po5offhyper.po5com1.AddItem "TRIA ( Min, Mode, Max )", 1
271     po5offhyper.po5com1.AddItem "NORM ( Mean, StdDev )", 2
272     po5offhyper.po5com1.AddItem "EXPO ( Mean )", 3
273     po5offhyper.po5com1.AddItem "UNIF ( Min, Max )", 4

274     Dim pop35 As Module
275     Dim pop35i As Long
276     Dim pop35v As String
277     pop35i = m.Modules.Find(smFindTag, "pop35")
278     Set pop35 = m.Modules(pop35i)
279     pop35v = pop35.Data("Expression")

280     po5offhyper.po5com3.value = pop35v
281     po5offhyper.po5com3.AddItem "TRIA ( 324, 360, 504 )", 0
282     po5offhyper.po5com3.AddItem "TRIA ( Min, Mode, Max )", 1
283     po5offhyper.po5com3.AddItem "NORM ( Mean, StdDev )", 2
284     po5offhyper.po5com3.AddItem "EXPO ( Mean )", 3
285     po5offhyper.po5com3.AddItem "UNIF ( Min, Max )", 4

286     Dim pop36 As Module
287     Dim pop36i As Long
288     Dim pop36v As String
289     pop36i = m.Modules.Find(smFindTag, "pop36")
290     Set pop36 = m.Modules(pop36i)
291     pop36v = pop36.Data("Expression")

292     po5offhyper.po5com5.value = pop36v
293     po5offhyper.po5com5.AddItem "TRIA ( 108, 120, 168 )", 0
```

```
    po5offhyper.po5com5.AddItem "TRIA ( Min, Mode, Max )", 1
    po5offhyper.po5com5.AddItem "NORM ( Mean, StdDev )", 2
    po5offhyper.po5com5.AddItem "EXPO ( Mean )", 3
    po5offhyper.po5com5.AddItem "UNIF ( Min, Max )", 4

    Dim pop37 As Module
    Dim pop37i As Long
    Dim pop37v As String
    pop37i = m.Modules.Find(smFindTag, "pop37")
    Set pop37 = m.Modules(pop37i)
    pop37v = pop37.Data("Expression")

    po5offhyper.po5com7.value = pop37v
    po5offhyper.po5com7.AddItem "TRIA ( 9, 10, 14 )", 0
    po5offhyper.po5com7.AddItem "TRIA ( Min, Mode, Max )", 1
    po5offhyper.po5com7.AddItem "NORM ( Mean, StdDev )", 2
    po5offhyper.po5com7.AddItem "EXPO ( Mean )", 3
    po5offhyper.po5com7.AddItem "UNIF ( Min, Max )", 4

    Dim pop38 As Module
    Dim pop38i As Long
    Dim pop38v As String
    pop38i = m.Modules.Find(smFindTag, "pop38")
    Set pop38 = m.Modules(pop38i)
    pop38v = pop38.Data("Expression")

    po5offhyper.po5com9.value = pop38v
    po5offhyper.po5com9.AddItem "TRIA ( 27, 30, 42 )", 0
    po5offhyper.po5com9.AddItem "TRIA ( Min, Mode, Max )", 1
    po5offhyper.po5com9.AddItem "NORM ( Mean, StdDev )", 2
    po5offhyper.po5com9.AddItem "EXPO ( Mean )", 3
    po5offhyper.po5com9.AddItem "UNIF ( Min, Max )", 4

    'Code below populates small combo boxes for OT-03, OT-06 and OT-07 thru OT-09
    Dim pop34u As Module
    Dim pop34ui As Long
    Dim pop34uv As String
    pop34ui = m.Modules.Find(smFindTag, "pop34")
    Set pop34u = m.Modules(pop34ui)
    pop34uv = pop34u.Data("Units")

    po5offhyper.po5com2.value = pop34uv
    po5offhyper.po5com2.AddItem "Seconds", 0
    po5offhyper.po5com2.AddItem "Minutes", 1
    po5offhyper.po5com2.AddItem "Hours", 2
    po5offhyper.po5com2.AddItem "Days", 3

    Dim pop35u As Module
    Dim pop35ui As Long
    Dim pop35uv As String
    pop35ui = m.Modules.Find(smFindTag, "pop35")
    Set pop35u = m.Modules(pop35ui)
    pop35uv = pop35u.Data("Units")

    po5offhyper.po5com4.value = pop35uv
    po5offhyper.po5com4.AddItem "Seconds", 0
    po5offhyper.po5com4.AddItem "Minutes", 1
    po5offhyper.po5com4.AddItem "Hours", 2
    po5offhyper.po5com4.AddItem "Days", 3

    Dim pop36u As Module
    Dim pop36ui As Long
    Dim pop36uv As String
    pop36ui = m.Modules.Find(smFindTag, "pop36")
    Set pop36u = m.Modules(pop36ui)
    pop36uv = pop36u.Data("Units")

    po5offhyper.po5com6.value = pop36uv
    po5offhyper.po5com6.AddItem "Seconds", 0
```

```
353     po5offhyper.po5com6.AddItem "Minutes", 1
354     po5offhyper.po5com6.AddItem "Hours", 2
355     po5offhyper.po5com6.AddItem "Days", 3

356     Dim pop37u As Module
357     Dim pop37ui As Long
358     Dim pop37uv As String
359     pop37ui = m.Modules.Find(smFindTag, "pop37")
360     Set pop37u = m.Modules(pop37ui)
361     pop37uv = pop37u.Data("Units")

362     po5offhyper.po5com8.value = pop37uv
363     po5offhyper.po5com8.AddItem "Seconds", 0
364     po5offhyper.po5com8.AddItem "Minutes", 1
365     po5offhyper.po5com8.AddItem "Hours", 2
366     po5offhyper.po5com8.AddItem "Days", 3

367     Dim pop38u As Module
368     Dim pop38ui As Long
369     Dim pop38uv As String
370     pop38ui = m.Modules.Find(smFindTag, "pop38")
371     Set pop38u = m.Modules(pop38ui)
372     pop38uv = pop38u.Data("Units")

373     po5offhyper.po5com10.value = pop38uv
374     po5offhyper.po5com10.AddItem "Seconds", 0
375     po5offhyper.po5com10.AddItem "Minutes", 1
376     po5offhyper.po5com10.AddItem "Hours", 2
377     po5offhyper.po5com10.AddItem "Days", 3

378   End Sub
```

Project/po6erect

```
1   Private Sub CommandButton6_Click()
2     Me.Hide
3     po5offhyper.Show

4   End Sub

5   Private Sub CommandButton7_Click()
6     Hierarchy.done08.Visible = True

7     'code below checks if any option button sets were not clicked, and if so,
forces the user to make a decision
8     Dim msgResult As Integer
9     If (po6frm2.Visible = True And po6opt1.value = False And po6opt2.value = False)
Then
10      msgResult = MsgBox("You must make an erecting mechanism choice.  Click Yes if
the erecting mechanism is part of the vehicle transporter.  Click no if the erecting
mechanism must be attached at the pad.", vbYesNo)
11      If msgResult = vbYes Then
12        po6opt1.value = True
13      Else
14        po6opt2.value = True
15      End If
16    End If

17    'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
18    Dim m As Model
19    Set m = ThisDocument.Model

20    Dim pop39 As Module
21    Dim pop39i As Long
22    pop39i = m.Modules.Find(smFindTag, "pop39")
23    Set pop39 = m.Modules(pop39i)
24    pop39.Data("Expression") = po6com1.Text
25    pop39.Data("Units") = po6com2.Text
```

```
    Dim pop40 As Module
    Dim pop40i As Long
    pop40i = m.Modules.Find(smFindTag, "pop40")
    Set pop40 = m.Modules(pop40i)
    pop40.Data("Expression") = po6com3.Text
    pop40.Data("Units") = po6com4.Text

    Dim pop41 As Module
    Dim pop41i As Long
    pop41i = m.Modules.Find(smFindTag, "pop41")
    Set pop41 = m.Modules(pop41i)
    pop41.Data("Expression") = po6com5.Text
    pop41.Data("Units") = po6com6.Text

    Dim pop42 As Module
    Dim pop42i As Long
    pop42i = m.Modules.Find(smFindTag, "pop42")
    Set pop42 = m.Modules(pop42i)
    pop42.Data("Expression") = po6com7.Text
    pop42.Data("Units") = po6com8.Text

    Dim pop43 As Module
    Dim pop43i As Long
    pop43i = m.Modules.Find(smFindTag, "pop43")
    Set pop43 = m.Modules(pop43i)
    pop43.Data("Expression") = po6com9.Text
    pop43.Data("Units") = po6com10.Text

    Dim pop44 As Module
    Dim pop44i As Long
    pop44i = m.Modules.Find(smFindTag, "pop44")
    Set pop44 = m.Modules(pop44i)
    pop44.Data("Expression") = po6com11.Text
    pop44.Data("Units") = po6com12.Text

    Dim pop45 As Module
    Dim pop45i As Long
    pop45i = m.Modules.Find(smFindTag, "pop45")
    Set pop45 = m.Modules(pop45i)
    pop45.Data("Expression") = po6com13.Text
    pop45.Data("Units") = po6com14.Text

    Dim pop46 As Module
    Dim pop46i As Long
    pop46i = m.Modules.Find(smFindTag, "pop46")
    Set pop46 = m.Modules(pop46i)
    pop46.Data("Expression") = po6com15.Text
    pop46.Data("Units") = po6com16.Text

    'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
    Dim pov8 As Module
    Dim pov8i As Long
    pov8i = m.Modules.Find(smFindTag, "pov8")
    Set pov8 = m.Modules(pov8i)
    If po6opt1.value = True Then
      pov8.Data("Initial Value") = "0"
    Else
      pov8.Data("Initial Value") = "1"
    End If

    'code below hides current form and shows the next form in the sequence
    Me.Hide
    po7umbilical.Show

  End Sub

  Private Sub CommandButton9_Click()
```

```
   83     Hierarchy.done08.Visible = True

   84     'code below checks if any option button sets were not clicked, and if so,
forces the user to make a decision
   85     Dim msgResult As Integer
   86     If (po6frm2.Visible = True And po6opt1.value = False And po6opt2.value = False)
Then
   87       msgResult = MsgBox("You must make an erecting mechanism choice.  Click Yes if
the erecting mechanism is part of the vehicle transporter.  Click no if the erecting
mechanism must be attached at the pad.", vbYesNo)
   88       If msgResult = vbYes Then
   89         po6opt1.value = True
   90       Else
   91         po6opt2.value = True
   92       End If
   93     End If

   94     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
   95     Dim m As Model
   96     Set m = ThisDocument.Model

   97     Dim pop39 As Module
   98     Dim pop39i As Long
   99     pop39i = m.Modules.Find(smFindTag, "pop39")
  100     Set pop39 = m.Modules(pop39i)
  101     pop39.Data("Expression") = po6com1.Text
  102     pop39.Data("Units") = po6com2.Text

  103     Dim pop40 As Module
  104     Dim pop40i As Long
  105     pop40i = m.Modules.Find(smFindTag, "pop40")
  106     Set pop40 = m.Modules(pop40i)
  107     pop40.Data("Expression") = po6com3.Text
  108     pop40.Data("Units") = po6com4.Text

  109     Dim pop41 As Module
  110     Dim pop41i As Long
  111     pop41i = m.Modules.Find(smFindTag, "pop41")
  112     Set pop41 = m.Modules(pop41i)
  113     pop41.Data("Expression") = po6com5.Text
  114     pop41.Data("Units") = po6com6.Text

  115     Dim pop42 As Module
  116     Dim pop42i As Long
  117     pop42i = m.Modules.Find(smFindTag, "pop42")
  118     Set pop42 = m.Modules(pop42i)
  119     pop42.Data("Expression") = po6com7.Text
  120     pop42.Data("Units") = po6com8.Text

  121     Dim pop43 As Module
  122     Dim pop43i As Long
  123     pop43i = m.Modules.Find(smFindTag, "pop43")
  124     Set pop43 = m.Modules(pop43i)
  125     pop43.Data("Expression") = po6com9.Text
  126     pop43.Data("Units") = po6com10.Text

  127     Dim pop44 As Module
  128     Dim pop44i As Long
  129     pop44i = m.Modules.Find(smFindTag, "pop44")
  130     Set pop44 = m.Modules(pop44i)
  131     pop44.Data("Expression") = po6com11.Text
  132     pop44.Data("Units") = po6com12.Text

  133     Dim pop45 As Module
  134     Dim pop45i As Long
  135     pop45i = m.Modules.Find(smFindTag, "pop45")
  136     Set pop45 = m.Modules(pop45i)
  137     pop45.Data("Expression") = po6com13.Text
```

```
138     pop45.Data("Units") = po6com14.Text

139     Dim pop46 As Module
140     Dim pop46i As Long
141     pop46i = m.Modules.Find(smFindTag, "pop46")
142     Set pop46 = m.Modules(pop46i)
143     pop46.Data("Expression") = po6com15.Text
144     pop46.Data("Units") = po6com16.Text

145     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
146     Dim pov8 As Module
147     Dim pov8i As Long
148     pov8i = m.Modules.Find(smFindTag, "pov8")
149     Set pov8 = m.Modules(pov8i)
150     If po6opt1.value = True Then
151       pov8.Data("Initial Value") = "0"
152     Else
153       pov8.Data("Initial Value") = "1"
154     End If

155     Me.Hide
156     Hierarchy.Show

157   End Sub

158   Private Sub Label1_Click()

159   End Sub

160   Private Sub Label11_Click()

161   End Sub

162   Private Sub Label12_Click()

163   End Sub

164   Private Sub OptionButton1_Click()

165   End Sub

166   Private Sub OptionButton10_Click()

167   End Sub

168   Private Sub OptionButton2_Click()

169   End Sub

170   Private Sub OptionButton4_Click()

171   End Sub

172   Private Sub OptionButton6_Click()

173   End Sub

174   Private Sub po6opt1_Click()
175     Label5.Visible = False
176     po6com3.Visible = False
177     po6com4.Visible = False

178   End Sub

179   Private Sub po6opt2_Click()
180     Label5.Visible = True
181     po6com3.Visible = True
182     po6com4.Visible = True
```

```
183   End Sub
184   Private Sub ToggleButton1_Click()
185   End Sub
186   Private Sub UserForm_Click()
187   End Sub
188   Private Sub UserForm_Initialize()
189     Dim m As Model
190     Set m = ThisDocument.Model
191     'Code below populates large combo boxes for IL-01 thru IL-11
192     Dim pop39 As Module
193     Dim pop39i As Long
194     Dim pop39v As String
195     pop39i = m.Modules.Find(smFindTag, "pop39")
196     Set pop39 = m.Modules(pop39i)
197     pop39v = pop39.Data("Expression")
198     po6erect.po6com1.value = pop39v
199     po6erect.po6com1.AddItem "TRIA ( 54, 60, 84 )", 0
200     po6erect.po6com1.AddItem "TRIA ( Min, Mode, Max )", 1
201     po6erect.po6com1.AddItem "NORM ( Mean, StdDev )", 2
202     po6erect.po6com1.AddItem "EXPO ( Mean )", 3
203     po6erect.po6com1.AddItem "UNIF ( Min, Max )", 4
204     Dim pop40 As Module
205     Dim pop40i As Long
206     Dim pop40v As String
207     pop40i = m.Modules.Find(smFindTag, "pop40")
208     Set pop40 = m.Modules(pop40i)
209     pop40v = pop40.Data("Expression")
210     po6erect.po6com3.value = pop40v
211     po6erect.po6com3.AddItem "TRIA ( 27, 30, 42 )", 0
212     po6erect.po6com3.AddItem "TRIA ( Min, Mode, Max )", 1
213     po6erect.po6com3.AddItem "NORM ( Mean, StdDev )", 2
214     po6erect.po6com3.AddItem "EXPO ( Mean )", 3
215     po6erect.po6com3.AddItem "UNIF ( Min, Max )", 4
216     Dim pop41 As Module
217     Dim pop41i As Long
218     Dim pop41v As String
219     pop41i = m.Modules.Find(smFindTag, "pop41")
220     Set pop41 = m.Modules(pop41i)
221     pop41v = pop41.Data("Expression")
222     po6erect.po6com5.value = pop41v
223     po6erect.po6com5.AddItem "TRIA ( 27, 30, 42 )", 0
224     po6erect.po6com5.AddItem "TRIA ( Min, Mode, Max )", 1
225     po6erect.po6com5.AddItem "NORM ( Mean, StdDev )", 2
226     po6erect.po6com5.AddItem "EXPO ( Mean )", 3
227     po6erect.po6com5.AddItem "UNIF ( Min, Max )", 4
228     Dim pop42 As Module
229     Dim pop42i As Long
230     Dim pop42v As String
231     pop42i = m.Modules.Find(smFindTag, "pop42")
232     Set pop42 = m.Modules(pop42i)
233     pop42v = pop42.Data("Expression")
234     po6erect.po6com7.value = pop42v
235     po6erect.po6com7.AddItem "TRIA ( 27, 30, 42 )", 0
236     po6erect.po6com7.AddItem "TRIA ( Min, Mode, Max )", 1
237     po6erect.po6com7.AddItem "NORM ( Mean, StdDev )", 2
```

```
    po6erect.po6com7.AddItem "EXPO ( Mean )", 3
    po6erect.po6com7.AddItem "UNIF ( Min, Max )", 4

    Dim pop43 As Module
    Dim pop43i As Long
    Dim pop43v As String
    pop43i = m.Modules.Find(smFindTag, "pop43")
    Set pop43 = m.Modules(pop43i)
    pop43v = pop43.Data("Expression")

    po6erect.po6com9.value = pop43v
    po6erect.po6com9.AddItem "TRIA ( 81, 90, 126 )", 0
    po6erect.po6com9.AddItem "TRIA ( Min, Mode, Max )", 1
    po6erect.po6com9.AddItem "NORM ( Mean, StdDev )", 2
    po6erect.po6com9.AddItem "EXPO ( Mean )", 3
    po6erect.po6com9.AddItem "UNIF ( Min, Max )", 4

    Dim pop44 As Module
    Dim pop44i As Long
    Dim pop44v As String
    pop44i = m.Modules.Find(smFindTag, "pop44")
    Set pop44 = m.Modules(pop44i)
    pop44v = pop44.Data("Expression")

    po6erect.po6com11.value = pop44v
    po6erect.po6com11.AddItem "TRIA ( 27, 30, 42 )", 0
    po6erect.po6com11.AddItem "TRIA ( Min, Mode, Max )", 1
    po6erect.po6com11.AddItem "NORM ( Mean, StdDev )", 2
    po6erect.po6com11.AddItem "EXPO ( Mean )", 3
    po6erect.po6com11.AddItem "UNIF ( Min, Max )", 4

    Dim pop45 As Module
    Dim pop45i As Long
    Dim pop45v As String
    pop45i = m.Modules.Find(smFindTag, "pop45")
    Set pop45 = m.Modules(pop45i)
    pop45v = pop45.Data("Expression")

    po6erect.po6com13.value = pop45v
    po6erect.po6com13.AddItem "TRIA ( 27, 30, 42 )", 0
    po6erect.po6com13.AddItem "TRIA ( Min, Mode, Max )", 1
    po6erect.po6com13.AddItem "NORM ( Mean, StdDev )", 2
    po6erect.po6com13.AddItem "EXPO ( Mean )", 3
    po6erect.po6com13.AddItem "UNIF ( Min, Max )", 4

    Dim pop46 As Module
    Dim pop46i As Long
    Dim pop46v As String
    pop46i = m.Modules.Find(smFindTag, "pop46")
    Set pop46 = m.Modules(pop46i)
    pop46v = pop46.Data("Expression")

    po6erect.po6com15.value = pop46v
    po6erect.po6com15.AddItem "TRIA ( 27, 30, 42 )", 0
    po6erect.po6com15.AddItem "TRIA ( Min, Mode, Max )", 1
    po6erect.po6com15.AddItem "NORM ( Mean, StdDev )", 2
    po6erect.po6com15.AddItem "EXPO ( Mean )", 3
    po6erect.po6com15.AddItem "UNIF ( Min, Max )", 4

    'Code below populates small combo boxes for IL-01 thru IL-11
    Dim pop39u As Module
    Dim pop39ui As Long
    Dim pop39uv As String
    pop39ui = m.Modules.Find(smFindTag, "pop39")
    Set pop39u = m.Modules(pop39ui)
    pop39uv = pop39u.Data("Units")

    po6erect.po6com2.value = pop39uv
    po6erect.po6com2.AddItem "Seconds", 0
```

```
297     po6erect.po6com2.AddItem "Minutes", 1
298     po6erect.po6com2.AddItem "Hours", 2
299     po6erect.po6com2.AddItem "Days", 3

300     Dim pop40u As Module
301     Dim pop40ui As Long
302     Dim pop40uv As String
303     pop40ui = m.Modules.Find(smFindTag, "pop40")
304     Set pop40u = m.Modules(pop40ui)
305     pop40uv = pop40u.Data("Units")

306     po6erect.po6com4.value = pop40uv
307     po6erect.po6com4.AddItem "Seconds", 0
308     po6erect.po6com4.AddItem "Minutes", 1
309     po6erect.po6com4.AddItem "Hours", 2
310     po6erect.po6com4.AddItem "Days", 3

311     Dim pop41u As Module
312     Dim pop41ui As Long
313     Dim pop41uv As String
314     pop41ui = m.Modules.Find(smFindTag, "pop41")
315     Set pop41u = m.Modules(pop41ui)
316     pop41uv = pop41u.Data("Units")

317     po6erect.po6com6.value = pop41uv
318     po6erect.po6com6.AddItem "Seconds", 0
319     po6erect.po6com6.AddItem "Minutes", 1
320     po6erect.po6com6.AddItem "Hours", 2
321     po6erect.po6com6.AddItem "Days", 3

322     Dim pop42u As Module
323     Dim pop42ui As Long
324     Dim pop42uv As String
325     pop42ui = m.Modules.Find(smFindTag, "pop42")
326     Set pop42u = m.Modules(pop42ui)
327     pop42uv = pop42u.Data("Units")

328     po6erect.po6com8.value = pop42uv
329     po6erect.po6com8.AddItem "Seconds", 0
330     po6erect.po6com8.AddItem "Minutes", 1
331     po6erect.po6com8.AddItem "Hours", 2
332     po6erect.po6com8.AddItem "Days", 3

333     Dim pop43u As Module
334     Dim pop43ui As Long
335     Dim pop43uv As String
336     pop43ui = m.Modules.Find(smFindTag, "pop43")
337     Set pop43u = m.Modules(pop43ui)
338     pop43uv = pop43u.Data("Units")

339     po6erect.po6com10.value = pop43uv
340     po6erect.po6com10.AddItem "Seconds", 0
341     po6erect.po6com10.AddItem "Minutes", 1
342     po6erect.po6com10.AddItem "Hours", 2
343     po6erect.po6com10.AddItem "Days", 3

344     Dim pop44u As Module
345     Dim pop44ui As Long
346     Dim pop44uv As String
347     pop44ui = m.Modules.Find(smFindTag, "pop44")
348     Set pop44u = m.Modules(pop44ui)
349     pop44uv = pop44u.Data("Units")

350     po6erect.po6com12.value = pop44uv
351     po6erect.po6com12.AddItem "Seconds", 0
352     po6erect.po6com12.AddItem "Minutes", 1
353     po6erect.po6com12.AddItem "Hours", 2
354     po6erect.po6com12.AddItem "Days", 3
```

```
355     Dim pop45u As Module
356     Dim pop45ui As Long
357     Dim pop45uv As String
358     pop45ui = m.Modules.Find(smFindTag, "pop45")
359     Set pop45u = m.Modules(pop45ui)
360     pop45uv = pop45u.Data("Units")

361     po6erect.po6com14.value = pop45uv
362     po6erect.po6com14.AddItem "Seconds", 0
363     po6erect.po6com14.AddItem "Minutes", 1
364     po6erect.po6com14.AddItem "Hours", 2
365     po6erect.po6com14.AddItem "Days", 3

366     Dim pop46u As Module
367     Dim pop46ui As Long
368     Dim pop46uv As String
369     pop46ui = m.Modules.Find(smFindTag, "pop46")
370     Set pop46u = m.Modules(pop46ui)
371     pop46uv = pop46u.Data("Units")

372     po6erect.po6com16.value = pop46uv
373     po6erect.po6com16.AddItem "Seconds", 0
374     po6erect.po6com16.AddItem "Minutes", 1
375     po6erect.po6com16.AddItem "Hours", 2
376     po6erect.po6com16.AddItem "Days", 3

377   End Sub
```

Project/po7umbilical

```
1   Private Sub CommandButton6_Click()
2     Me.Hide
3     If po1prelim.po1opt3.value = True Then
4       po2on.Show
5     Else
6       po6erect.Show
7     End If

8   End Sub

9   Private Sub CommandButton7_Click()
10    Hierarchy.done09.Visible = True

11    'code below checks for option button sets that were not clicked, and if so,
forces the user to make a decision
12    Dim msgResult As Integer
13    If (po7opt1.value = False And po7opt2.value = False And po7opt3.value = False)
Then
14      msgResult = MsgBox("You must make an umbilical connection choice.  Click Yes
if both propellant and electrical connections need to be made.  Click No if only
propellant connections need to be made.  Click Cancel if umbilical connections are
already made.", vbYesNoCancel)
15      If msgResult = vbYes Then
16        po7opt3.value = True
17      ElseIf msgResult = vbNo Then
18        po7opt2.value = True
19      Else
20        po7opt1.value = True
21      End If
22    End If
23    If (po7opt4.value = False And po7opt5.value = False) Then
24      msgResult = MsgBox("You must make a RP decision.  Will the vehicle use RP?",
vbYesNo)
25      If msgResult = vbYes Then
26        po7opt4.value = True
27      Else
28        po7opt5.value = True
29      End If
30    End If
```

```
31      If (po7opt4.value = True And po7opt6.value = False And po7opt7.value = False)
Then
32        msgResult = MsgBox("You must make a decision concerning which stages use RP.
Click Yes if RP is used in stage 1 only.  Click No if RP is used in stage 1 and stage
2.", vbYesNo)
33        If msgResult = vbYes Then
34          po7opt6.value = True
35        Else
36          po7opt7.value = True
37        End If
38      End If
39      If (po7opt4.value = True And po7opt7.value = True And po7opt8.value = False And
po7opt9.value = False) Then
40        msgResult = MsgBox("You must make a decision concerning parallel RP loading
operations.  Can stage 1 and stage 2 be loaded with RP in parallel?", vbYesNo)
41        If msgResult = vbYes Then
42          po7opt8.value = True
43        Else
44          po7opt9.value = True
45        End If
46      End If

47      'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
48      Dim m As Model
49      Set m = ThisDocument.Model

50      Dim pop65 As Module
51      Dim pop65i As Long
52      pop65i = m.Modules.Find(smFindTag, "pop65")
53      Set pop65 = m.Modules(pop65i)
54      pop65.Data("Expression") = po7com1.Text
55      pop65.Data("Units") = po7com2.Text

56      Dim pop66 As Module
57      Dim pop66i As Long
58      pop66i = m.Modules.Find(smFindTag, "pop66")
59      Set pop66 = m.Modules(pop66i)
60      pop66.Data("Expression") = po7com3.Text
61      pop66.Data("Units") = po7com4.Text

62      Dim pop67 As Module
63      Dim pop67i As Long
64      pop67i = m.Modules.Find(smFindTag, "pop67")
65      Set pop67 = m.Modules(pop67i)
66      pop67.Data("Expression") = po7com5.Text
67      pop67.Data("Units") = po7com6.Text

68      Dim pop68 As Module
69      Dim pop68i As Long
70      pop68i = m.Modules.Find(smFindTag, "pop68")
71      Set pop68 = m.Modules(pop68i)
72      pop68.Data("Expression") = po7com7.Text
73      pop68.Data("Units") = po7com8.Text

74      Dim pop72 As Module
75      Dim pop72i As Long
76      pop72i = m.Modules.Find(smFindTag, "pop72")
77      Set pop72 = m.Modules(pop72i)
78      pop72.Data("Expression") = po7com9.Text
79      pop72.Data("Units") = po7com10.Text

80      Dim pop73 As Module
81      Dim pop73i As Long
82      pop73i = m.Modules.Find(smFindTag, "pop73")
83      Set pop73 = m.Modules(pop73i)
84      pop73.Data("Expression") = po7com9.Text
85      pop73.Data("Units") = po7com10.Text
```

```
    Dim pop75 As Module
    Dim pop75i As Long
    pop75i = m.Modules.Find(smFindTag, "pop75")
    Set pop75 = m.Modules(pop75i)
    pop75.Data("Expression") = po7com9.Text
    pop75.Data("Units") = po7com10.Text

    Dim pop74 As Module
    Dim pop74i As Long
    pop74i = m.Modules.Find(smFindTag, "pop74")
    Set pop74 = m.Modules(pop74i)
    pop74.Data("Expression") = po7com11.Text
    pop74.Data("Units") = po7com12.Text

    Dim pop76 As Module
    Dim pop76i As Long
    pop76i = m.Modules.Find(smFindTag, "pop76")
    Set pop76 = m.Modules(pop76i)
    pop76.Data("Expression") = po7com11.Text
    pop76.Data("Units") = po7com12.Text

    Dim pop78 As Module
    Dim pop78i As Long
    pop78i = m.Modules.Find(smFindTag, "pop78")
    Set pop78 = m.Modules(pop78i)
    pop78.Data("Expression") = po7com13.Text
    pop78.Data("Units") = po7com14.Text

    'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
    Dim pov9 As Module
    Dim pov9i As Long
    pov9i = m.Modules.Find(smFindTag, "pov9")
    Set pov9 = m.Modules(pov9i)
    If po7opt1.value = True Then
      pov9.Data("Initial Value") = "0"
    ElseIf po7opt2.value = True Then
      pov9.Data("Initial Value") = "1"
    Else
      pov9.Data("Initial Value") = "2"
    End If

    Dim pov10 As Module
    Dim pov10i As Long
    pov10i = m.Modules.Find(smFindTag, "pov10")
    Set pov10 = m.Modules(pov10i)
    If po7opt5.value = True Then
      pov10.Data("Initial Value") = "0"
    ElseIf po7opt6.value = True Then
      pov10.Data("Initial Value") = "1"
    Else
      pov10.Data("Initial Value") = "2"
    End If

    Dim pov11 As Module
    Dim pov11i As Long
    pov11i = m.Modules.Find(smFindTag, "pov11")
    Set pov11 = m.Modules(pov11i)
    If po7opt8.value = True Then
      pov11.Data("Initial Value") = "1"
    Else
      pov11.Data("Initial Value") = "0"
    End If

    'code below hides the current form and shows the next form in the sequence
    Me.Hide
    po8propellant.Show

  End Sub
```

```
  Private Sub CommandButton9_Click()
    Hierarchy.done09.Visible = True

    'code below checks for option button sets that were not clicked, and if so,
forces the user to make a decision
    Dim msgResult As Integer
    If (po7opt1.value = False And po7opt2.value = False And po7opt3.value = False)
Then
      msgResult = MsgBox("You must make an umbilical connection choice.  Click Yes
if both propellant and electrical connections need to be made.  Click No if only
propellant connections need to be made.  Click Cancel if umbilical connections are
already made.", vbYesNoCancel)
      If msgResult = vbYes Then
        po7opt3.value = True
      ElseIf msgResult = vbNo Then
        po7opt2.value = True
      Else
        po7opt1.value = True
      End If
    End If
    If (po7opt4.value = False And po7opt5.value = False) Then
      msgResult = MsgBox("You must make a RP decision.  Will the vehicle use RP?",
vbYesNo)
      If msgResult = vbYes Then
        po7opt4.value = True
      Else
        po7opt5.value = True
      End If
    End If
    If (po7opt4.value = True And po7opt6.value = False And po7opt7.value = False)
Then
      msgResult = MsgBox("You must make a decision concerning which stages use RP.
Click Yes if RP is used in stage 1 only.  Click No if RP is used in stage 1 and stage
2.", vbYesNo)
      If msgResult = vbYes Then
        po7opt6.value = True
      Else
        po7opt7.value = True
      End If
    End If
    If (po7opt4.value = True And po7opt7.value = True And po7opt8.value = False And
po7opt9.value = False) Then
      msgResult = MsgBox("You must make a decision concerning parallel RP loading
operations.  Can stage 1 and stage 2 be loaded with RP in parallel?", vbYesNo)
      If msgResult = vbYes Then
        po7opt8.value = True
      Else
        po7opt9.value = True
      End If
    End If

    'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
    Dim m As Model
    Set m = ThisDocument.Model

    Dim pop65 As Module
    Dim pop65i As Long
    pop65i = m.Modules.Find(smFindTag, "pop65")
    Set pop65 = m.Modules(pop65i)
    pop65.Data("Expression") = po7com1.Text
    pop65.Data("Units") = po7com2.Text

    Dim pop66 As Module
    Dim pop66i As Long
    pop66i = m.Modules.Find(smFindTag, "pop66")
    Set pop66 = m.Modules(pop66i)
    pop66.Data("Expression") = po7com3.Text
```

```
198     pop66.Data("Units") = po7com4.Text

199     Dim pop67 As Module
200     Dim pop67i As Long
201     pop67i = m.Modules.Find(smFindTag, "pop67")
202     Set pop67 = m.Modules(pop67i)
203     pop67.Data("Expression") = po7com5.Text
204     pop67.Data("Units") = po7com6.Text

205     Dim pop68 As Module
206     Dim pop68i As Long
207     pop68i = m.Modules.Find(smFindTag, "pop68")
208     Set pop68 = m.Modules(pop68i)
209     pop68.Data("Expression") = po7com7.Text
210     pop68.Data("Units") = po7com8.Text

211     Dim pop72 As Module
212     Dim pop72i As Long
213     pop72i = m.Modules.Find(smFindTag, "pop72")
214     Set pop72 = m.Modules(pop72i)
215     pop72.Data("Expression") = po7com9.Text
216     pop72.Data("Units") = po7com10.Text

217     Dim pop73 As Module
218     Dim pop73i As Long
219     pop73i = m.Modules.Find(smFindTag, "pop73")
220     Set pop73 = m.Modules(pop73i)
221     pop73.Data("Expression") = po7com9.Text
222     pop73.Data("Units") = po7com10.Text

223     Dim pop75 As Module
224     Dim pop75i As Long
225     pop75i = m.Modules.Find(smFindTag, "pop75")
226     Set pop75 = m.Modules(pop75i)
227     pop75.Data("Expression") = po7com9.Text
228     pop75.Data("Units") = po7com10.Text

229     Dim pop74 As Module
230     Dim pop74i As Long
231     pop74i = m.Modules.Find(smFindTag, "pop74")
232     Set pop74 = m.Modules(pop74i)
233     pop74.Data("Expression") = po7com11.Text
234     pop74.Data("Units") = po7com12.Text

235     Dim pop76 As Module
236     Dim pop76i As Long
237     pop76i = m.Modules.Find(smFindTag, "pop76")
238     Set pop76 = m.Modules(pop76i)
239     pop76.Data("Expression") = po7com11.Text
240     pop76.Data("Units") = po7com12.Text

241     Dim pop78 As Module
242     Dim pop78i As Long
243     pop78i = m.Modules.Find(smFindTag, "pop78")
244     Set pop78 = m.Modules(pop78i)
245     pop78.Data("Expression") = po7com13.Text
246     pop78.Data("Units") = po7com14.Text

247     'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
248     Dim pov9 As Module
249     Dim pov9i As Long
250     pov9i = m.Modules.Find(smFindTag, "pov9")
251     Set pov9 = m.Modules(pov9i)
252     If po7opt1.value = True Then
253       pov9.Data("Initial Value") = "0"
254     ElseIf po7opt2.value = True Then
255       pov9.Data("Initial Value") = "1"
256     Else
```

```
257       pov9.Data("Initial Value") = "2"
258     End If

259     Dim pov10 As Module
260     Dim pov10i As Long
261     pov10i = m.Modules.Find(smFindTag, "pov10")
262     Set pov10 = m.Modules(pov10i)
263     If po7opt5.value = True Then
264       pov10.Data("Initial Value") = "0"
265     ElseIf po7opt6.value = True Then
266       pov10.Data("Initial Value") = "1"
267     Else
268       pov10.Data("Initial Value") = "2"
269     End If

270     Dim pov11 As Module
271     Dim pov11i As Long
272     pov11i = m.Modules.Find(smFindTag, "pov11")
273     Set pov11 = m.Modules(pov11i)
274     If po7opt8.value = True Then
275       pov11.Data("Initial Value") = "1"
276     Else
277       pov11.Data("Initial Value") = "0"
278     End If

279     Me.Hide
280     Hierarchy.Show

281   End Sub

282   Private Sub Label11_Click()

283   End Sub

284   Private Sub Label12_Click()

285   End Sub

286   Private Sub Label16_Click()

287   End Sub

288   Private Sub OptionButton1_Click()

289   End Sub

290   Private Sub OptionButton2_Click()

291   End Sub

292   Private Sub OptionButton4_Click()

293   End Sub

294   Private Sub OptionButton6_Click()

295   End Sub

296   Private Sub po7opt1_Click()
297     po7frm1.Visible = False
298     po7frm2.Visible = False

299   End Sub

300   Private Sub po7opt2_Click()
301     po7frm1.Visible = True
302     po7frm2.Visible = False

303   End Sub
```

```
304   Private Sub po7opt3_Click()
305     po7frm1.Visible = True
306     po7frm2.Visible = True

307   End Sub

308   Private Sub po7opt4_Click()
309     po7frm3.Visible = True
310     po7frm4.Visible = True
311     po7frm5.Visible = True
312     po7frm6.Visible = True

313   End Sub

314   Private Sub po7opt5_Click()
315     po7frm3.Visible = False
316     po7frm4.Visible = False
317     po7frm5.Visible = False
318     po7frm6.Visible = False

319   End Sub

320   Private Sub po7opt6_Click()
321     po7frm4.Visible = False
322     po7frm5.Visible = True
323     po7frm6.Visible = False

324     If po7opt4.value = True Then
325       po8propellant.po8frm1.Visible = False
326       po8propellant.po8frm2.Visible = True
327     End If

328   End Sub

329   Private Sub po7opt7_Click()
330     po7frm4.Visible = True
331     po7frm5.Visible = True
332     po7frm6.Visible = True
333     If po7opt4.value = True Then
334       po8propellant.po8frm1.Visible = False
335       po8propellant.po8frm2.Visible = False
336     End If

337   End Sub

338   Private Sub ToggleButton1_Click()

339   End Sub

340   Private Sub UserForm_Click()

341   End Sub

342   Private Sub UserForm_Initialize()
343     Dim m As Model
344     Set m = ThisDocument.Model

345     'Code below populates large combo boxes for UM-02 thru UM-05
346     Dim pop65 As Module
347     Dim pop65i As Long
348     Dim pop65v As String
349     pop65i = m.Modules.Find(smFindTag, "pop65")
350     Set pop65 = m.Modules(pop65i)
351     pop65v = pop65.Data("Expression")

352     po7umbilical.po7com1.value = pop65v
353     po7umbilical.po7com1.AddItem "TRIA ( 27, 30, 42 )", 0
354     po7umbilical.po7com1.AddItem "TRIA ( Min, Mode, Max )", 1
```

```
    po7umbilical.po7com1.AddItem "NORM ( Mean, StdDev )", 2
    po7umbilical.po7com1.AddItem "EXPO ( Mean )", 3
    po7umbilical.po7com1.AddItem "UNIF ( Min, Max )", 4

    Dim pop66 As Module
    Dim pop66i As Long
    Dim pop66v As String
    pop66i = m.Modules.Find(smFindTag, "pop66")
    Set pop66 = m.Modules(pop66i)
    pop66v = pop66.Data("Expression")

    po7umbilical.po7com3.value = pop66v
    po7umbilical.po7com3.AddItem "TRIA ( 4.5, 5, 7 )", 0
    po7umbilical.po7com3.AddItem "TRIA ( Min, Mode, Max )", 1
    po7umbilical.po7com3.AddItem "NORM ( Mean, StdDev )", 2
    po7umbilical.po7com3.AddItem "EXPO ( Mean )", 3
    po7umbilical.po7com3.AddItem "UNIF ( Min, Max )", 4

    Dim pop67 As Module
    Dim pop67i As Long
    Dim pop67v As String
    pop67i = m.Modules.Find(smFindTag, "pop67")
    Set pop67 = m.Modules(pop67i)
    pop67v = pop67.Data("Expression")

    po7umbilical.po7com5.value = pop67v
    po7umbilical.po7com5.AddItem "TRIA ( 27, 30, 42 )", 0
    po7umbilical.po7com5.AddItem "TRIA ( Min, Mode, Max )", 1
    po7umbilical.po7com5.AddItem "NORM ( Mean, StdDev )", 2
    po7umbilical.po7com5.AddItem "EXPO ( Mean )", 3
    po7umbilical.po7com5.AddItem "UNIF ( Min, Max )", 4

    Dim pop68 As Module
    Dim pop68i As Long
    Dim pop68v As String
    pop68i = m.Modules.Find(smFindTag, "pop68")
    Set pop68 = m.Modules(pop68i)
    pop68v = pop68.Data("Expression")

    po7umbilical.po7com7.value = pop68v
    po7umbilical.po7com7.AddItem "TRIA ( 27, 30, 42 )", 0
    po7umbilical.po7com7.AddItem "TRIA ( Min, Mode, Max )", 1
    po7umbilical.po7com7.AddItem "NORM ( Mean, StdDev )", 2
    po7umbilical.po7com7.AddItem "EXPO ( Mean )", 3
    po7umbilical.po7com7.AddItem "UNIF ( Min, Max )", 4

    'Code below populates small combo boxes for UM-02 thru UM-05
    Dim pop65u As Module
    Dim pop65ui As Long
    Dim pop65uv As String
    pop65ui = m.Modules.Find(smFindTag, "pop65")
    Set pop65u = m.Modules(pop65ui)
    pop65uv = pop65u.Data("Units")

    po7umbilical.po7com2.value = pop65uv
    po7umbilical.po7com2.AddItem "Seconds", 0
    po7umbilical.po7com2.AddItem "Minutes", 1
    po7umbilical.po7com2.AddItem "Hours", 2
    po7umbilical.po7com2.AddItem "Days", 3

    Dim pop66u As Module
    Dim pop66ui As Long
    Dim pop66uv As String
    pop66ui = m.Modules.Find(smFindTag, "pop66")
    Set pop66u = m.Modules(pop66ui)
    pop66uv = pop66u.Data("Units")

    po7umbilical.po7com4.value = pop66uv
    po7umbilical.po7com4.AddItem "Seconds", 0
```

```
414     po7umbilical.po7com4.AddItem "Minutes", 1
415     po7umbilical.po7com4.AddItem "Hours", 2
416     po7umbilical.po7com4.AddItem "Days", 3

417     Dim pop67u As Module
418     Dim pop67ui As Long
419     Dim pop67uv As String
420     pop67ui = m.Modules.Find(smFindTag, "pop67")
421     Set pop67u = m.Modules(pop67ui)
422     pop67uv = pop67u.Data("Units")

423     po7umbilical.po7com6.value = pop67uv
424     po7umbilical.po7com6.AddItem "Seconds", 0
425     po7umbilical.po7com6.AddItem "Minutes", 1
426     po7umbilical.po7com6.AddItem "Hours", 2
427     po7umbilical.po7com6.AddItem "Days", 3

428     Dim pop68u As Module
429     Dim pop68ui As Long
430     Dim pop68uv As String
431     pop68ui = m.Modules.Find(smFindTag, "pop68")
432     Set pop68u = m.Modules(pop68ui)
433     pop68uv = pop68u.Data("Units")

434     po7umbilical.po7com8.value = pop68uv
435     po7umbilical.po7com8.AddItem "Seconds", 0
436     po7umbilical.po7com8.AddItem "Minutes", 1
437     po7umbilical.po7com8.AddItem "Hours", 2
438     po7umbilical.po7com8.AddItem "Days", 3

439     'Code below populates large combo boxes for UM-09 thru UM-11
440     Dim pop72 As Module
441     Dim pop72i As Long
442     Dim pop72v As String
443     pop72i = m.Modules.Find(smFindTag, "pop72")
444     Set pop72 = m.Modules(pop72i)
445     pop72v = pop72.Data("Expression")

446     po7umbilical.po7com9.value = pop72v
447     po7umbilical.po7com9.AddItem "TRIA ( 108, 120, 168 )", 0
448     po7umbilical.po7com9.AddItem "TRIA ( Min, Mode, Max )", 1
449     po7umbilical.po7com9.AddItem "NORM ( Mean, StdDev )", 2
450     po7umbilical.po7com9.AddItem "EXPO ( Mean )", 3
451     po7umbilical.po7com9.AddItem "UNIF ( Min, Max )", 4

452     Dim pop74 As Module
453     Dim pop74i As Long
454     Dim pop74v As String
455     pop74i = m.Modules.Find(smFindTag, "pop74")
456     Set pop74 = m.Modules(pop74i)
457     pop74v = pop74.Data("Expression")

458     po7umbilical.po7com11.value = pop74v
459     po7umbilical.po7com11.AddItem "TRIA ( 54, 60, 84 )", 0
460     po7umbilical.po7com11.AddItem "TRIA ( Min, Mode, Max )", 1
461     po7umbilical.po7com11.AddItem "NORM ( Mean, StdDev )", 2
462     po7umbilical.po7com11.AddItem "EXPO ( Mean )", 3
463     po7umbilical.po7com11.AddItem "UNIF ( Min, Max )", 4

464     Dim pop78 As Module
465     Dim pop78i As Long
466     Dim pop78v As String
467     pop78i = m.Modules.Find(smFindTag, "pop78")
468     Set pop78 = m.Modules(pop78i)
469     pop78v = pop78.Data("Expression")

470     po7umbilical.po7com13.value = pop78v
471     po7umbilical.po7com13.AddItem "0", 0
472     po7umbilical.po7com13.AddItem "TRIA ( Min, Mode, Max )", 1
```

```
473     po7umbilical.po7com13.AddItem "NORM ( Mean, StdDev )", 2
474     po7umbilical.po7com13.AddItem "EXPO ( Mean )", 3
475     po7umbilical.po7com13.AddItem "UNIF ( Min, Max )", 4

476     'Code below populates small combo boxes for UM-09 thru UM-11
477     Dim pop72u As Module
478     Dim pop72ui As Long
479     Dim pop72uv As String
480     pop72ui = m.Modules.Find(smFindTag, "pop72")
481     Set pop72u = m.Modules(pop72ui)
482     pop72uv = pop72u.Data("Units")

483     po7umbilical.po7com10.value = pop72uv
484     po7umbilical.po7com10.AddItem "Seconds", 0
485     po7umbilical.po7com10.AddItem "Minutes", 1
486     po7umbilical.po7com10.AddItem "Hours", 2
487     po7umbilical.po7com10.AddItem "Days", 3

488     Dim pop74u As Module
489     Dim pop74ui As Long
490     Dim pop74uv As String
491     pop74ui = m.Modules.Find(smFindTag, "pop74")
492     Set pop74u = m.Modules(pop74ui)
493     pop74uv = pop74u.Data("Units")

494     po7umbilical.po7com12.value = pop74uv
495     po7umbilical.po7com12.AddItem "Seconds", 0
496     po7umbilical.po7com12.AddItem "Minutes", 1
497     po7umbilical.po7com12.AddItem "Hours", 2
498     po7umbilical.po7com12.AddItem "Days", 3

499     Dim pop78u As Module
500     Dim pop78ui As Long
501     Dim pop78uv As String
502     pop78ui = m.Modules.Find(smFindTag, "pop78")
503     Set pop78u = m.Modules(pop78ui)
504     pop78uv = pop78u.Data("Units")

505     po7umbilical.po7com14.value = pop78uv
506     po7umbilical.po7com14.AddItem "Seconds", 0
507     po7umbilical.po7com14.AddItem "Minutes", 1
508     po7umbilical.po7com14.AddItem "Hours", 2
509     po7umbilical.po7com14.AddItem "Days", 3

510   End Sub
```

Project/po8propellant

```
1   Private Sub CommandButton6_Click()
2     Me.Hide
3     po7umbilical.Show

4   End Sub

5   Private Sub CommandButton7_Click()

6     Hierarchy.done10.Visible = True
7   End Sub

8   Private Sub CommandButton9_Click()
9     Hierarchy.done10.Visible = True

10    'code below checks for option button sets that were not clicked, and if so,
forces the user to make a decision
11    Dim msgResult As Integer
12    If (po8opt1.value = False And po8opt2.value = False And po8opt3.value = False)
Then
13      msgResult = MsgBox("You must make a decision concerning cryogenic loading
operations.  Click Yes if both stages and fuel/oxidizer can be loaded in parallel.  Click
```

```
No if fuel and oxidizer cannot be loaded in parallel but stages can be loaded in
parallel.  Click Cancel if neither stages nor fuel/oxidizer can be loaded in parallel.",
vbYesNoCancel)
   14       If msgResult = vbYes Then
   15         po8opt3.value = True
   16       ElseIf msgResult = vbNo Then
   17         po8opt2.value = True
   18       Else
   19         po8opt1.value = True
   20       End If
   21     End If

   22     'code below populates appropriate arena modules with distributions and units
the user put into the combo boxes
   23     Dim m As Model
   24     Set m = ThisDocument.Model

   25     Dim pop79 As Module
   26     Dim pop79i As Long
   27     pop79i = m.Modules.Find(smFindTag, "pop79")
   28     Set pop79 = m.Modules(pop79i)
   29     pop79.Data("Expression") = po8com1.Text
   30     pop79.Data("Units") = po8com2.Text

   31     Dim pop83 As Module
   32     Dim pop83i As Long
   33     pop83i = m.Modules.Find(smFindTag, "pop83")
   34     Set pop83 = m.Modules(pop83i)
   35     pop83.Data("Expression") = po8com1.Text
   36     pop83.Data("Units") = po8com2.Text

   37     Dim pop87 As Module
   38     Dim pop87i As Long
   39     pop87i = m.Modules.Find(smFindTag, "pop87")
   40     Set pop87 = m.Modules(pop87i)
   41     pop87.Data("Expression") = po8com1.Text
   42     pop87.Data("Units") = po8com2.Text

   43     Dim pop81 As Module
   44     Dim pop81i As Long
   45     pop81i = m.Modules.Find(smFindTag, "pop81")
   46     Set pop81 = m.Modules(pop81i)
   47     pop81.Data("Expression") = po8com3.Text
   48     pop81.Data("Units") = po8com4.Text

   49     Dim pop85 As Module
   50     Dim pop85i As Long
   51     pop85i = m.Modules.Find(smFindTag, "pop85")
   52     Set pop85 = m.Modules(pop85i)
   53     pop85.Data("Expression") = po8com3.Text
   54     pop85.Data("Units") = po8com4.Text

   55     Dim pop88 As Module
   56     Dim pop88i As Long
   57     pop88i = m.Modules.Find(smFindTag, "pop88")
   58     Set pop88 = m.Modules(pop88i)
   59     pop88.Data("Expression") = po8com3.Text
   60     pop88.Data("Units") = po8com4.Text

   61     Dim pop80 As Module
   62     Dim pop80i As Long
   63     pop80i = m.Modules.Find(smFindTag, "pop80")
   64     Set pop80 = m.Modules(pop80i)
   65     pop80.Data("Expression") = po8com5.Text
   66     pop80.Data("Units") = po8com6.Text

   67     Dim pop84 As Module
   68     Dim pop84i As Long
   69     pop84i = m.Modules.Find(smFindTag, "pop84")
```

```
    Set pop84 = m.Modules(pop84i)
    pop84.Data("Expression") = po8com5.Text
    pop84.Data("Units") = po8com6.Text

    Dim pop89 As Module
    Dim pop89i As Long
    pop89i = m.Modules.Find(smFindTag, "pop89")
    Set pop89 = m.Modules(pop89i)
    pop89.Data("Expression") = po8com5.Text
    pop89.Data("Units") = po8com6.Text

    Dim pop82 As Module
    Dim pop82i As Long
    pop82i = m.Modules.Find(smFindTag, "pop82")
    Set pop82 = m.Modules(pop82i)
    pop82.Data("Expression") = po8com7.Text
    pop82.Data("Units") = po8com8.Text

    Dim pop86 As Module
    Dim pop86i As Long
    pop86i = m.Modules.Find(smFindTag, "pop86")
    Set pop86 = m.Modules(pop86i)
    pop86.Data("Expression") = po8com7.Text
    pop86.Data("Units") = po8com8.Text

    Dim pop90 As Module
    Dim pop90i As Long
    pop90i = m.Modules.Find(smFindTag, "pop90")
    Set pop90 = m.Modules(pop90i)
    pop90.Data("Expression") = po8com7.Text
    pop90.Data("Units") = po8com8.Text

    Dim pop91 As Module
    Dim pop91i As Long
    pop91i = m.Modules.Find(smFindTag, "pop91")
    Set pop91 = m.Modules(pop91i)
    pop91.Data("Expression") = po8com9.Text
    pop91.Data("Units") = po8com10.Text

    'Code below takes user's option button decisions and translates them into
initial values for the variables that control the corresponding decision modules
    Dim pov12 As Module
    Dim pov12i As Long
    pov12i = m.Modules.Find(smFindTag, "pov12")
    Set pov12 = m.Modules(pov12i)
    If po8opt1.value = True Then
      pov12.Data("Initial Value") = "0"
    ElseIf po8opt2.value = True Then
      pov12.Data("Initial Value") = "1"
    Else
      pov12.Data("Initial Value") = "2"
    End If

    Me.Hide
    Hierarchy.Show

  End Sub

  Private Sub Label11_Click()

  End Sub

  Private Sub Label12_Click()

  End Sub

  Private Sub OptionButton1_Click()

  End Sub
```

```
124   Private Sub OptionButton2_Click()

125   End Sub

126   Private Sub OptionButton4_Click()

127   End Sub

128   Private Sub OptionButton6_Click()

129   End Sub

130   Private Sub ToggleButton1_Click()

131   End Sub

132   Private Sub UserForm_Click()

133   End Sub

134   Private Sub UserForm_Initialize()
135     Dim m As Model
136     Set m = ThisDocument.Model

137     'Code below populates large combo boxes for PL-02 thru Pl-06
138     Dim pop79 As Module
139     Dim pop79i As Long
140     Dim pop79v As String
141     pop79i = m.Modules.Find(smFindTag, "pop79")
142     Set pop79 = m.Modules(pop79i)
143     pop79v = pop79.Data("Expression")

144     po8propellant.po8com1.value = pop79v
145     po8propellant.po8com1.AddItem "TRIA ( 54, 60, 84 )", 0
146     po8propellant.po8com1.AddItem "TRIA ( Min, Mode, Max )", 1
147     po8propellant.po8com1.AddItem "NORM ( Mean, StdDev )", 2
148     po8propellant.po8com1.AddItem "EXPO ( Mean )", 3
149     po8propellant.po8com1.AddItem "UNIF ( Min, Max )", 4

150     Dim pop81 As Module
151     Dim pop81i As Long
152     Dim pop81v As String
153     pop81i = m.Modules.Find(smFindTag, "pop81")
154     Set pop81 = m.Modules(pop81i)
155     pop81v = pop81.Data("Expression")

156     po8propellant.po8com3.value = pop81v
157     po8propellant.po8com3.AddItem "TRIA ( 27, 30, 42 )", 0
158     po8propellant.po8com3.AddItem "TRIA ( Min, Mode, Max )", 1
159     po8propellant.po8com3.AddItem "NORM ( Mean, StdDev )", 2
160     po8propellant.po8com3.AddItem "EXPO ( Mean )", 3
161     po8propellant.po8com3.AddItem "UNIF ( Min, Max )", 4

162     Dim pop80 As Module
163     Dim pop80i As Long
164     Dim pop80v As String
165     pop80i = m.Modules.Find(smFindTag, "pop80")
166     Set pop80 = m.Modules(pop80i)
167     pop80v = pop80.Data("Expression")

168     po8propellant.po8com5.value = pop80v
169     po8propellant.po8com5.AddItem "TRIA ( 54, 60, 84 )", 0
170     po8propellant.po8com5.AddItem "TRIA ( Min, Mode, Max )", 1
171     po8propellant.po8com5.AddItem "NORM ( Mean, StdDev )", 2
172     po8propellant.po8com5.AddItem "EXPO ( Mean )", 3
173     po8propellant.po8com5.AddItem "UNIF ( Min, Max )", 4

174     Dim pop82 As Module
```

```
    Dim pop82i As Long
    Dim pop82v As String
    pop82i = m.Modules.Find(smFindTag, "pop82")
    Set pop82 = m.Modules(pop82i)
    pop82v = pop82.Data("Expression")

    po8propellant.po8com7.value = pop82v
    po8propellant.po8com7.AddItem "TRIA ( 27, 30, 42 )", 0
    po8propellant.po8com7.AddItem "TRIA ( Min, Mode, Max )", 1
    po8propellant.po8com7.AddItem "NORM ( Mean, StdDev )", 2
    po8propellant.po8com7.AddItem "EXPO ( Mean )", 3
    po8propellant.po8com7.AddItem "UNIF ( Min, Max )", 4

    Dim pop91 As Module
    Dim pop91i As Long
    Dim pop91v As String
    pop91i = m.Modules.Find(smFindTag, "pop91")
    Set pop91 = m.Modules(pop91i)
    pop91v = pop91.Data("Expression")

    po8propellant.po8com9.value = pop91v
    po8propellant.po8com9.AddItem "10", 0
    po8propellant.po8com9.AddItem "TRIA ( Min, Mode, Max )", 1
    po8propellant.po8com9.AddItem "NORM ( Mean, StdDev )", 2
    po8propellant.po8com9.AddItem "EXPO ( Mean )", 3
    po8propellant.po8com9.AddItem "UNIF ( Min, Max )", 4

    'Code below populates small combo boxes for PL-02 thru Pl-06
    Dim pop79u As Module
    Dim pop79ui As Long
    Dim pop79uv As String
    pop79ui = m.Modules.Find(smFindTag, "pop79")
    Set pop79u = m.Modules(pop79ui)
    pop79uv = pop79u.Data("Units")

    po8propellant.po8com2.value = pop79uv
    po8propellant.po8com2.AddItem "Seconds", 0
    po8propellant.po8com2.AddItem "Minutes", 1
    po8propellant.po8com2.AddItem "Hours", 2
    po8propellant.po8com2.AddItem "Days", 3

    Dim pop81u As Module
    Dim pop81ui As Long
    Dim pop81uv As String
    pop81ui = m.Modules.Find(smFindTag, "pop81")
    Set pop81u = m.Modules(pop81ui)
    pop81uv = pop81u.Data("Units")

    po8propellant.po8com4.value = pop81uv
    po8propellant.po8com4.AddItem "Seconds", 0
    po8propellant.po8com4.AddItem "Minutes", 1
    po8propellant.po8com4.AddItem "Hours", 2
    po8propellant.po8com4.AddItem "Days", 3

    Dim pop80u As Module
    Dim pop80ui As Long
    Dim pop80uv As String
    pop80ui = m.Modules.Find(smFindTag, "pop80")
    Set pop80u = m.Modules(pop80ui)
    pop80uv = pop80u.Data("Units")

    po8propellant.po8com6.value = pop80uv
    po8propellant.po8com6.AddItem "Seconds", 0
    po8propellant.po8com6.AddItem "Minutes", 1
    po8propellant.po8com6.AddItem "Hours", 2
    po8propellant.po8com6.AddItem "Days", 3

    Dim pop82u As Module
    Dim pop82ui As Long
```

```
234     Dim pop82uv As String
235     pop82ui = m.Modules.Find(smFindTag, "pop82")
236     Set pop82u = m.Modules(pop82ui)
237     pop82uv = pop82u.Data("Units")

238     po8propellant.po8com8.value = pop82uv
239     po8propellant.po8com8.AddItem "Seconds", 0
240     po8propellant.po8com8.AddItem "Minutes", 1
241     po8propellant.po8com8.AddItem "Hours", 2
242     po8propellant.po8com8.AddItem "Days", 3

243     Dim pop91u As Module
244     Dim pop91ui As Long
245     Dim pop91uv As String
246     pop91ui = m.Modules.Find(smFindTag, "pop91")
247     Set pop91u = m.Modules(pop91ui)
248     pop91uv = pop91u.Data("Units")

249     po8propellant.po8com10.value = pop91uv
250     po8propellant.po8com10.AddItem "Seconds", 0
251     po8propellant.po8com10.AddItem "Minutes", 1
252     po8propellant.po8com10.AddItem "Hours", 2
253     po8propellant.po8com10.AddItem "Days", 3

254   End Sub
```

Bibliography

Banks, J., J. S. C. II, et al. (2005). Discrete-event System Simulation, Prentice Hall International Series in Industrial and Systems Engineering.

Boeing (2000). Sea Launch Users Guide. Seattle, Washington, Boeing Commercial Space Company.

Brown, K. K. (2003). Technology Challenges for Operationally Responsive Spacelift, College of Aerospace Doctrine, Research and Education Air University.

Cates, G. R., M. Mollaghasemi, et al. (2002). Modeling the Space Shuttle. 2002 Winter Simulation Conference.

H. Murat Gunaydin, P. D. (1998). "The Delphi Method." Retrieved October 2005, from http://www.iyte.edu.tr/~muratgunaydin/delphi.htm.

John B. Shroeder, I. s. e. (2006).

Kelton, W. D., R. P. Sadowski, et al. (2004). Simulation With Arena, Third Edition, McGraw-Hill.

Lee, A. E. M. and K. T. Schmierer (1994). A Simulation of the B-2 two-level Avionics Maintenacne System Concept. Wright Patterson AFB, OH, AFIT.

McCleskey, C. M. (2005). Space Shuttle Operations and Infrastructure: A Systems Analysis of Design Root Causes and Effects, National Aeronautics and Space Administration.

O'Malley, M. T. (2006). Aircraft Maintainance Technician, United States Air Force. Personal interview. 16 September 2005.

Rooney, B. D. and A. Hartong (2004). A discrete-event simulation of turnaround time and manpower of military RLVs. A Collection of Technical Papers - AIAA Space 2004 Conference and Exposition, Sep 28-30 2004, San Diego, CA, United States, American Institute of Aeronautics and Astronautics Inc.

Schlagheck, R. A. and J. K. Byers (1971). Simulating the Operations of the Reusable Shuttle Space Vehicle. 1971 Summer Computer Simulation Conference.

Steele, M. J., M. Mollaghasemi, et al. (2002). Generic Simulation Models of Reusable Launch Vehicles. 2002 Winter Simulation Conference.

Stiegelmeier, A. T. (2006). A Discrete-event Simulation Model for Evaluating Air Force Reusable Military Launch Vehicle Prelaunch Operations. ENS. Wright Patterson AFB, Graduate School of Engineering and Management, Air Force Institute of Technology (AU). MS Logistics Management.

Zapata, E. and A. J. Ruiz-Torres (2000). Sumulation Based Operational Analysis of Future Space Transportation Systems. 2000 Winter Sumulation Conference.

Vita

Capt Pope hails from Locust Grove, Georgia where he graduated from Henry County High School. He entered the Air Force in 1992 as an aircraft armament systems specialist. His first 3 years were spent at Kadena AB in Okinawa, Japan working on F-15c and F15d aircraft. He received an incentive flight for extinguishing a fire in an F-15c prior to a functional check flight. At that point, he decided to finish college. Clayton State, University of Maryland, Louisiana Tech, Louisiana State University, and finally Southern Illinois University make up his diverse college career which culminated in a B.S. in Industrial Technology from Southern Illinois University in 1997 while stationed at Barksdale AFB in Shreveport, Louisiana. After a year long tour at Osan AB in Songtan, Republic of Korea, he was accepted to Officer Training School. Commissioned in August of 2001, his first duty station was Malmstrom AFB, Great Falls, Montana as a Missile and Munitions Maintenance officer in the 341st Missile Maintenance Squadron. In 2004, Capt Pope was selected to attend the Graduate Logistics Management program, Graduate School of Engineering and Management, Air Force Institute of Technology.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
23-03-2006	Master's Thesis	March 2005 – March 2006

4. TITLE AND SUBTITLE

DISCRETE EVENT SIMULATION MODEL OF THE GROUND MAINTENANCE OPERATIONS OF A REUSABLE MILITARY LAUNCH VEHICLE

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Pope, John T. III, Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-8865

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GLM/ENS/06-14

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

AFRL/VAOT
Attn: Mr. Bruce Thieman
2130 8th Street
WPAFB OH 45433
DSN: 785-5162
e-mail: bruce.thieman@wpafb.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The Air Force uses a family of expendable launch vehicles to meet its spacelift needs. Unfortunately, this method is not responsive: months of preparation are typically required and launch costs are high. Consequently, the Air Force seeks a reusable military launch vehicle that can be launched inexpensively and quickly regenerated between flights. Air Force Research Laboratory personnel desire a tool to help evaluate candidate designs and perform tradeoff studies necessary to acquire a launch vehicle that will achieve Air Force goals. The objective of this research was first to develop a conceptual model of maintenance operations needed to regenerate a launch vehicle between flights, and then to translate this conceptual model into a discrete event simulation tool. This research was accomplished concurrently with Stiegelmeier, who focused on vehicle prelaunch operations.

15. SUBJECT TERMS

Reusable Military Launch Vehicle, Reusable Launch Vehicle, Launch Vehicle, Ground Processing, Maintenance Operations, Computer Simulation, Discrete Event Simulation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Alan W. Johnson, Ph. D.
U	U	U	UU	249	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3636, ext 4703; e-mail: Alan.Johnson@afit.edu

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18