Systems Engineering Model and Training Application for Desktop Environment

Jeffrey T. May[1]
United Space Alliance, LLC, Kennedy Space Center, Fl

Provide a graphical user interface based simulator for desktop training, operations and procedure development and system reference. This simulator allows for engineers to train and further understand the dynamics of their system from their local desktops. It allows the users to train and evaluate their system at a pace and skill level based on the user's competency and from a perspective based on the user's need. The simulator will not require any special resources to execute and should generally be available for use. The interface is based on a concept of presenting the model of the system in ways that best suits the user's application or training needs. The three levels of views are Component View, the System View (overall system), and the Console View (monitor). These views are portals into a single model, so changing the model from one view or from a model manager Graphical User Interface will be reflected on all other views.

I. Introduction

With the development of a new launch vehicle and ground support system soon to be a reality here at KSC and at other NASA and NASA Contractor facilities across the country, it is now more important than ever that system engineers receive training using state of the art capabilities. Whether working manned or unmanned launch vehicles, it is always in the mind of a system engineer that a mistake in a written procedure, an errant console command, or inability to properly safe a system in a timely manner could result in loss of mission, vehicle, or even loss of life. Utilizing enhanced methods of training to increase engineering competency and system knowledge makes for a safer work environment and results in a more effective engineering workforce.

NASA and its Contractors will experience a period of rapid change in the workplace during the development of the new launch vehicles. The existing engineering work force will have to be re-trained on operations of new systems and on new vehicle hardware. Change will also come from turn-over of the workforce as many experienced engineers leave for new assignments or retirement and new engineers are needed to work the launch vehicles. And finally change will be required to make operations on the new vehicles more efficient. Creating a scaled-down, diversely trained engineering workforce allows for cost-effective and cost-competitive operations. This work-force will be expected to quickly and efficiently learn to process and launch a brand new vehicle at the same level of quality that has been provided for the Space Shuttle program. This can only be achieved through the implementation of a more effective engineering training program.

The system engineer is responsible for developing, updating, and improving upon processes and procedures within his system. These processes and procedures include maintenance, testing, repair, operation and launch of the system. Typically the engineering end-to-end process includes troubleshooting of failures, disposition of the appropriate repair procedure, supporting actual hands-on vehicle work, and final performance of retest. This normally requires a coordinated effort among several engineers across multiple systems, working in concert at remote locations with the technicians and quality control personnel on the work floor. The system engineer is ultimately responsible for ensuring each team member's task is properly executed. Further responsibilities for the system engineer include risk analysis and understanding the consequences of system hardware failures, presenting technical papers to engineering management and to Program Boards on hardware anomalies or process changes, training of less experienced engineers, etc., This is not a comprehensive list of all the tasks of the system engineer, but rather a list of the tasks for which an engineer would benefit from the results of training and proficiency demonstrations afforded by the use of a desktop simulator tool.

This paper presents a proposed concept for a desktop simulator tool. The desktop simulator is intended and designed to solve some of the system engineer's largest needs. It is a tool that allows a system engineer to view and

[1] System Engineer, Main Propulsion System, 1102 John Glenn Blvd Titusville, Fl 32780 USK-420

manipulate a model of his system from a standard desktop computer via different views and perspectives and using custom tools to allow him to make the best use of it. It is based on three different categories of views; console, component and system. It is developed to support formal training, operations and procedure development, and as a system reference.

II. Engineering Training Needs

A. Training

One of the most important needs of a system engineer is sufficient training. In today's environment there is a formal process for verifying that a system engineer is trained, but the training itself is not very formalized. Many of the senior Shuttle engineers were trained long ago, before simulation tools existed and have since held the same positions. Training of new engineers typically consists of self initiated studying of randomly gathered brain books, cheat sheets, schematics and references and by working alongside the senior engineers in an OJT manner until the experienced engineer feels that the trainee is adequately trained to work on their own. This method of training is very time consuming to the senior engineer and does not allow the inexperienced engineer to learn at his/her own pace and in repetition. The capability of OJT training is also very limited in comparison to training that can be done using a simulator.

While the current methodology of training has proven to be adequate for the Shuttle program, there are several issues with transitioning to a new program that would make fast efficient training more critical than ever. First of all, much of the talent and expertise has either left, or will be leaving the space program via retirement or other opportunities or may be changing positions as a new vehicle comes about. Another issue is that there will be changes to the vehicle that even the most experienced of Shuttle engineers will have to learn. Another concern is because, as we downscale the workforce and try to be more productive through cross-training, there will be less time for supporting OJT and less of the currently used 'buddy system' which will mean each engineer will need to be able to work more independently.

The Shuttle currently does have model and simulation training, but it does not meet the needs addressed by the desktop simulator. First of all, the model and simulations require the use of a firing room and are geared primarily as integrated rehearsals for vehicle tanking and launch. They assume that all participating members are already fully trained on their system, and much of the time spent is focused on communication and problem solving protocols at the management level. Even then, due to the physical and human resources required to perform a simulation the training is quite limited. It would not be uncommon for a certified console engineer to sit console for as little as 25 hours of simulated training in an average year, with as few as 1 or 2 hours per year directed specifically to that engineer. The reason is because these simulations are not meant to train the engineer on his system or his procedures as this would not be a practical way to do it.

As we desire to increase and improve training it will be critical to formalize the training process. This includes keeping record of the low level training that an engineer receives. It should provide quantifiable or reviewable results that allow the trainee and management to focus on, and strengthen areas in which he they may be weak in. For proper training it is important that an engineer can repetitively train on a vast array of scenarios until they are as familiar to the engineer as the back of their hand, so to speak. To do this would require that the simulator is readily available, flexible to different types of training and changes to the training and can support different methods of interaction.

B. Procedures and Operations

Another reason why it is so important for a system engineer to be able to visualize their system is to allow them to better develop procedures and perform operations. Often there are times when a procedure, troubleshooting plan or concept of operations is developed at an engineer's desk only to find out that there was either an error in the procedure, confusion in the configuration or there was a difference between how the engineer thought something would operate and how it really does. Having some form of simulation would allow an engineer to work through a procedure locally. The engineer could also simulate different configurations, determining the best configuration for a test or potentially helping to pinpoint the most likely cause for a failure.

C. System Reference

A final simulation need for a system engineer is for a good system reference. As it has been said a picture is worth a thousand words, how much would a dynamic animation be worth? Even the most trained and senior

engineers come across questions or lack of complete understanding that a good graphic simulation could clear up. Often there are times where an engineer may understand two separate concepts from different perspectives only to find out that they really related in ways that they had not thought of prior to seeing it graphically. There is also benefit to being able to dynamically analyze a system. One example of this would be while performing failure analysis of electrical circuits where different commands and failures could be introduced and the effects visually displayed. A final use of a simulator in a reference capacity would be to explain concepts to engineers of systems which yours interacts as well as with management.

III. Desktop Simulator Concept

The concept of a desktop simulator is based on presenting the engineer with a user-friendly, readily accessible podium that provides detailed system or component data relative to his/her specific system. Engineers working on complex fluid or mechanical systems need to understand and interact with their system in various ways in order to perform the many tasks required of the system engineer. This means he/she needs to be trained on and understand the interaction of systems and components in numerous aspects. Full system competency of a component such as a valve would include understanding of system hardware installation requirements and installation drawings, electrical functionality and drawings, fluid schematics, test requirements documentation, etc., The desktop simulator is uniquely capable of making all of these resources available to the engineer for the purposes of training and as a valuable desktop resource tool.

For example, take the case of a valve cycle error occurring (real or simulated) due to an electrical circuit anomaly, such as a diode failure. A system engineer in training mode may chose to simply explore and understand the general electrical schematic of the component using the simulator. A team of senior engineers might want to systematically work through the same simulator schematic to troubleshoot the anomaly or to better visually understand the affects and criticality of this failure. Another engineer may want to proactively see the effect it would have upon the system if the valve failed to open during operation and yet a third engineer may be training to be a console operator and simply need to be able to identify that the valve did not open when commanded and to determine what immediate safing procedure may be required.

The design concept of the desktop modeler is that there are essentially three views of the system as well as other tools and GUIS that allow for model manipulation, training sessions, playback and recording etc. The three views are the Component View, the System View, and the Console View. These views would be controlled by client application on each user's desktop. There would be a model interface on a server or remote computer. While there would be crude modeling capability for test and debug, the model that ultimately runs the simulator and maintains "state " would most likely be a third party tool, but the interface, scripts for initialization and fault injection, and any logic that extends the capability of the model would be part of the desktop modeler application.

The Component view would be an animated/ dynamic view of the pneumatic and electrical hardware of an individual component. An example of a component view is shown in Fig. 1. This view allows the user to see electrical schematics down to the wire and electrical component level, while simultaneously being able to see the functional interaction as the valve is pneumatically actuated in a cross sectional functional schematic view.

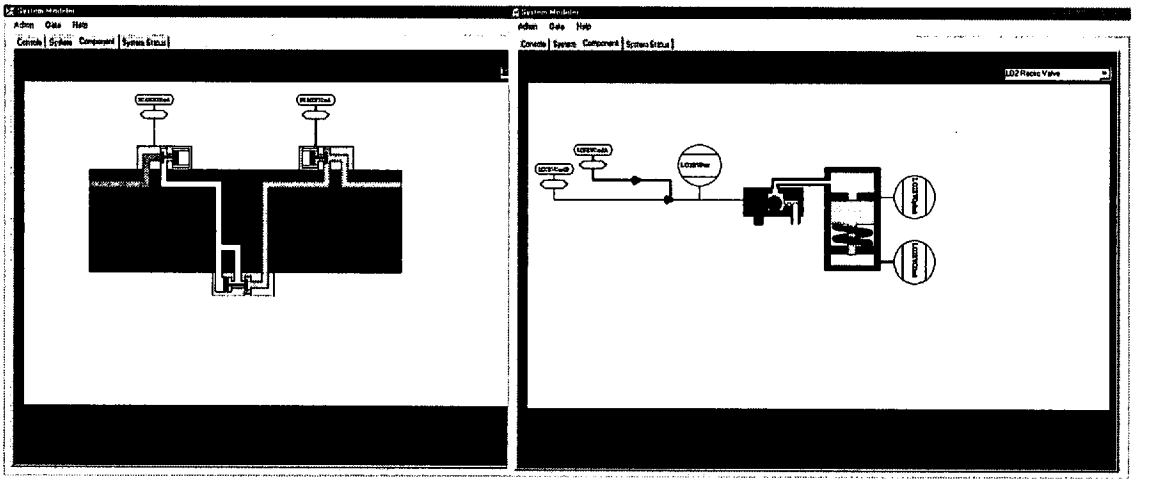

Figure 1. Examples of Representative Component Views

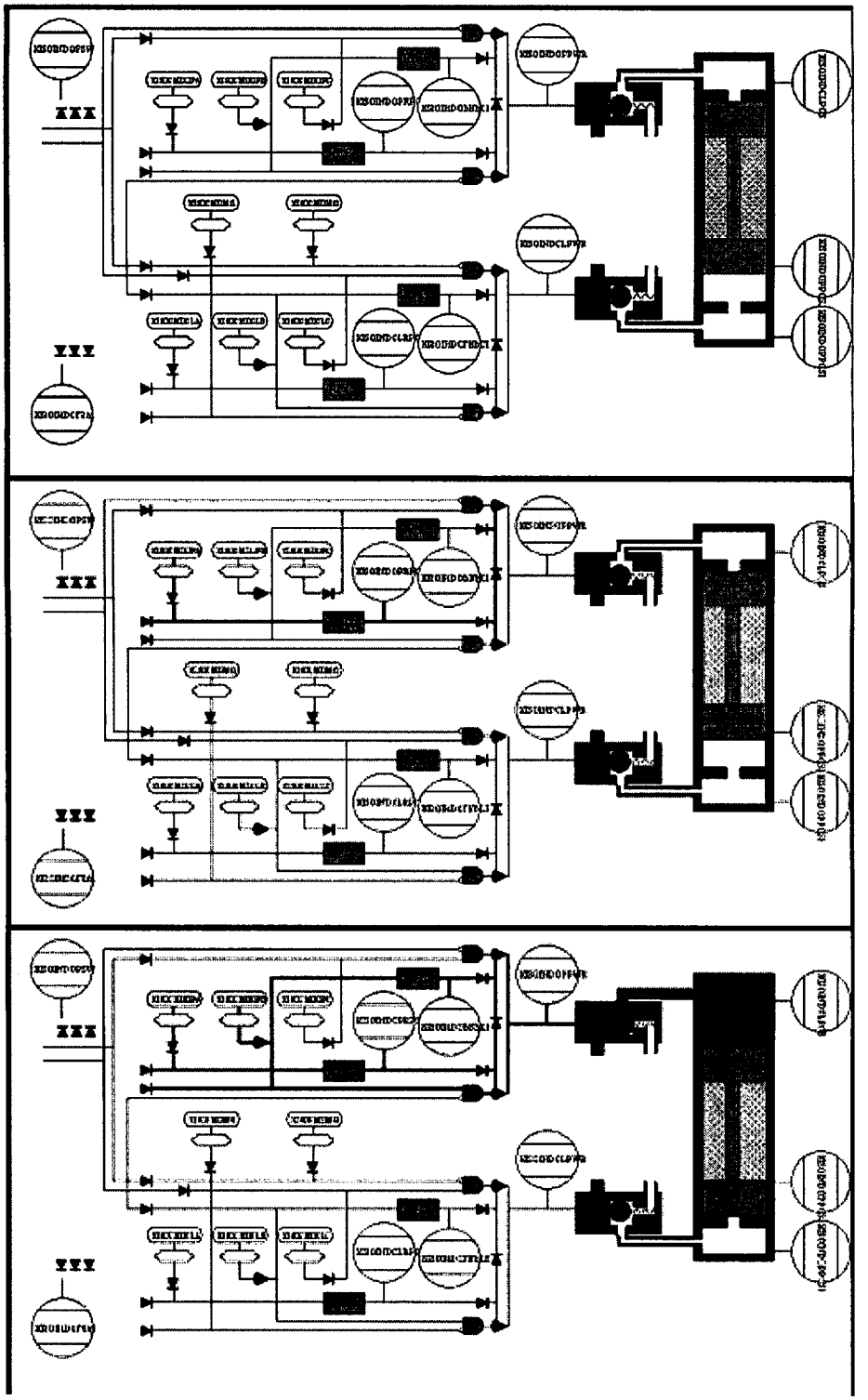

Figure 2 Demonstration of Component View

Figure 2 represents an example of how a valve cycle occurs today in one of the Space Shuttle fluid systems to better understand some of the ways that the tool can be used. The first image indicates a closed valve with no power or pneumatic pressure applied. This particular valve requires 2 of 3 commands to be set on to cycle the valve. The second image shows the valves state with a single command set on. In the third image, a second command is set on revealing the wiring logic required to energize the solenoid and allow high pressure helium to the open actuator the valve driving the piston to the Open position. The mechanics of the piston to the actual valve blade are not shown here, but could be shown on another schematic if necessary. It is easy to see from these graphics that a simulator could quickly explain how the electrical and pneumatic circuits work together to cycle a valve and what the results would be for various combinations of commands and or failures such as a broken wire or failed diode.

The System View would be an animated/ dynamic view of the overall system. Like the component view there can be several different system views as defined by the needs of the engineers. The system view shows the state of the overall system as defined by the model, and can allow a user to see how the components interact with each other. One of several use cases for a system such as Main Propulsion would be to watch a playback of a cryogenic tanking from throughout the various stages of loading. The user would be able to monitor his system behavior during these operations and develop a better understanding of over-all system operations. There would likely be a dialog box that explains some of the subtle points that cannot be expressed by the graphics alone.

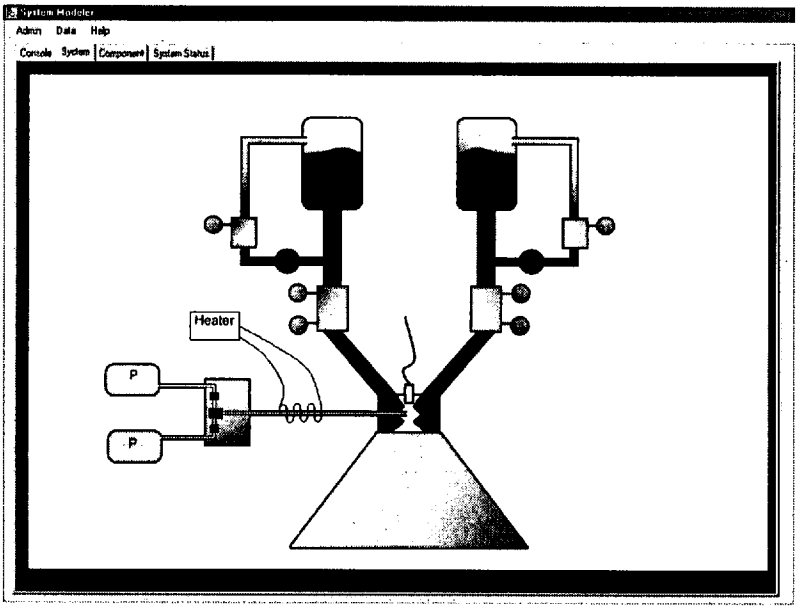

Figure 3. Example of a Representative System View

The Console View should look identical to the view that the console operator has on station in the firing room. This view is where the user will train to be a console operator. It is possible that the console view may be similar to the System View depending on how the consoles are developed but they should still be kept separate for two reasons. First, they serve very different functions. One is to understand the systems integrated operations, the other is to train as a console operator. This means that even though the schematics are similar the data displayed and the interactivity provided will be very different. The second reason is because there may be systems such as on Shuttle where the console displays cannot change and are fixed to lines, squares and circles but the System View can provide a better picture of what is going on. An example of a Console use case would be to train for an operation you may be performing from console. You would be able to load a training scenario (initializes the system, injects a failure, may simulate some other minor external stimulus) and then run the procedure. Even without a single failure, the ability to run a procedure would be invaluable. In this simulator however, failures can be injected and the trainee can document that he found them, perform troubleshooting based on his console views or perform required safing as necessary. All of this would be logged and recorded to be saved in a training package.

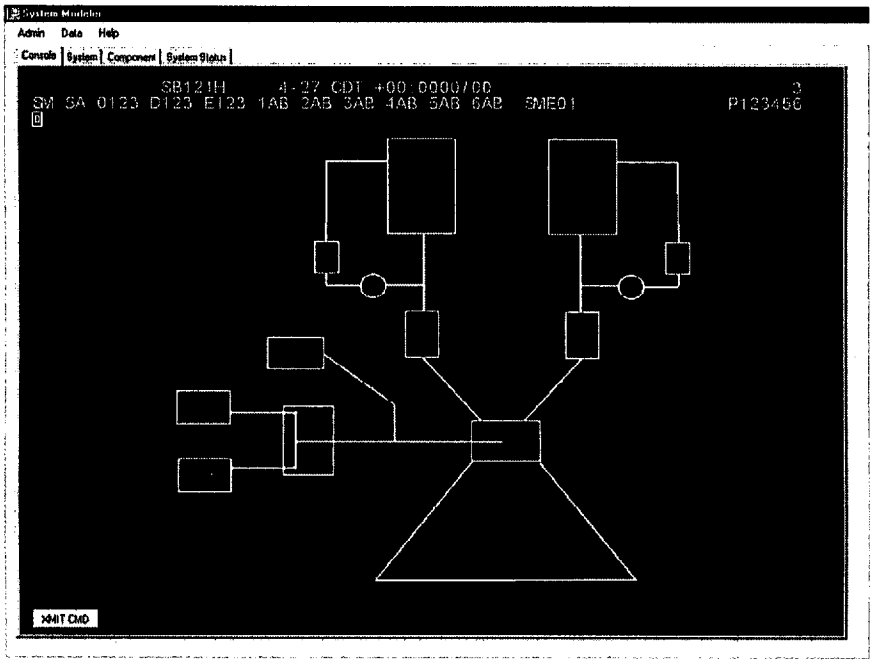

Figure 4. Example of a Representative Console View

IV. How the Desktop Simulator Can Help the Engineer

The desktop simulator concept was developed to meet many of the needs expressed previously in this paper. It is intended to support training effort, be used as an aid in developing procedures and conducting vehicle operations and to serve as a reference to system engineers.

A. Training

There are many training benefits of the desktop simulator. The first is that the different dynamic views will allow the user to see his system in a dynamic and animated model. The engineer will not have to look at a series of cross-sections or reference drawings and have to mentally picture how his system operates. This is especially important for systems which will invariably be composed of complex electrical circuits and hardware. He will also be able to see the state changes of his entire system at once and how a cycling vent valve can cause temperature fluctuations in different part of his system. Another benefit is that it provides a resource for the engineer to watch how a system behaves during a specific operation or procedure allowing the engineer to understand the procedures and processes of his system as well as how the components themselves operate. It also provides console training. This may be the most important training that a system engineer can have since many of the decisions they make must be quick and the results of their decisions could mean the difference in being trained to successfully safe his system in an emergency. Console training could come in several different forms. It could be training to allow the trainee to become familiar with the operation of the console and the resulting displays. It could be to allow the user to become familiar with the operation of a standard procedure and how they would go about problem solving simple failures. It could be to allow the trainee to be exposed to critical failures and to learn what safing would be required and how they would go about expressing it to the rest of the launch team.

One thing that is very important in training is repetitive learning, which involves doing an operation over and over until it becomes second nature. Having a simulator available at everyone's desktop would provide the necessary availability for any engineer to get this repetitive testing as opposed to only being able to do integrated simulation based training for a few hours per year.

Training with the simulator also has the advantage of having events or milestones recorded. The training results could be in the form of a written essay, multiple choice exam, audio recordings, or simply a log of activity. Training may also consist of simply "watching an operation playback" with informational pop-ups to explain what is going on. It could just be some hands on demo type stuff that tells the engineer what commands to click and then asks

questions to get the trainee to observe certain details. The training would be in pre-defined modules using scripts and scenarios that would allow the trainee to be guided to which training he needs to be exposed to and in which order. This would ensure that the trainee was exposed to all required training and in a timely manner.

B. Procedures and Operations

There are several ways that an engineer can use the desktop simulator to help write procedures and perform operations. The first way is by actually performing the procedure using the simulator via the System View or the Console View. Performing the procedure will allow the engineer to visually see what configurations the system will be in and whether any of those configurations could be harmful (releasing a hazardous commodity or over pressurizing a line) or invalidate a test (pressure at test site inadvertently vented off). It may also reveal a simpler and potentially safer test configuration. An engineer could also find that he gets unexpected results resulting from a failure to completely understand his system. Finding these results via the simulator will avoid additional work of re-writing and approving paper and re-performing the operation as well as maintaining the company's integrity and avoiding schedule impacts.

Analysis of the system or component level could help an engineer pinpoint a failure or at least understand what results they should expect to see when mirroring a troubleshooting procedure on the actual vehicle. Finally the engineer can use the console view to better understand the procedure from an operational perspective. There are times when a procedure may be written a specific way because it simplifies the writing of the procedure itself, but ends up taking hours to execute and results in an increase of manual steps (which increases chance of error) or which may prove to be mentally or ergonomically challenging. Being able to access the simulator from the desktop and using the 3 view concept is critical to making this work.

C. System References

By being able to access views of the system, senior engineers can analyze and look at all sorts of relationships within their system. They can energize different commands and observe the effect that it has on the individual components or the system as a whole. They can set commands in the console view and see the results on the electrical schematic or they can fail a diode on in an electrical schematic and then flip to the console display to see how (or if) it would be detectable by the engineer on console. The simulator can be used in risk analysis and process improvement reviews to pro-actively look for failures. A desktop tool would allow an engineer to quickly show a peer engineer or manager how a component of the system works or perhaps the peer could determine the functionality on their own.

V. Design and Features of Desktop Simulator

It is important to note that this tool is just a prototyped concept at this point in its development. A prototype of the application is being built for this demonstration using a crude model to highlight some of the GUI concepts, but an actual architecture cannot be developed until more details are known about the future vehicle's design and operations. While many of the concepts here can be applied to various types engineering systems, this prototype is geared with a focus towards operations of space vehicles. In addition, the focus of this tool is on the GUI and user interaction tools and not on the model itself, while a model could be custom built for this application, the intent would be that this tool would be an interface to an existing vehicle model. At this point the tool would most likely be a local desktop application that consists of the GUIs and which communicates with a server that houses the model and training database. Alternatives could be to use VM Servers or web apps and embedded tools provide a richer more interactive functionality. Each user could interact with his own instance of the model or several users and / or a trainer could share a model. In either case, all displays for a given user will use the same model. The model and the displays should be de-coupled such that they are not directly dependent on each other. The model shall not be dependent on the displays at all and the GUIs shall only be dependent on a model-display interface such that the model can be changed and only the interface affected, assuming that the model represents the same hardware. The displays should be easy to create, but will likely need a software developer to create due to the complexity and uniqueness of each system. There are three general types of views, but the specific GUIs within each view are determined by the needs of the engineering system. The views are really defined by their use or application rather than what is on them. Often in space systems, there are a lot of similar components within and across different systems. The architecture and GUI development process should be such that similar components, logic, graphics and displays are re-used.

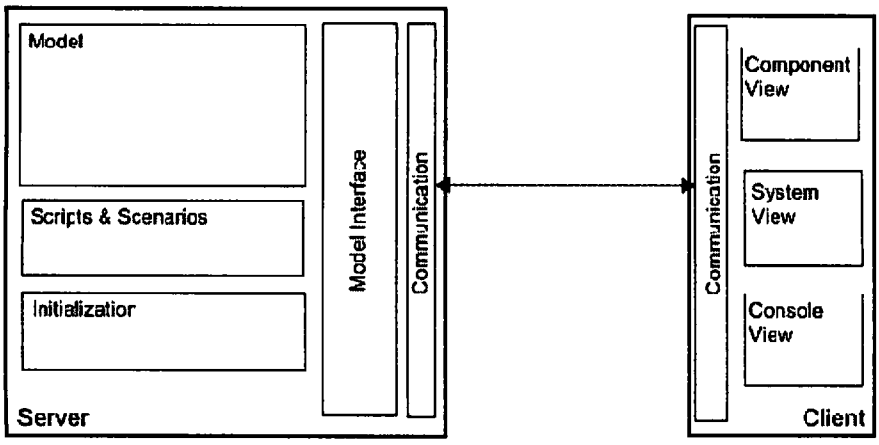

Figure 5. High Level Architecture

VI. Conclusion

With the increased capability of computers and networking and the need to do things better, faster and cheaper, training via modeling and simulation has really taken off. For several reasons, the Shuttle Program has not needed to make heavy use of modeling and simulation. One reason is because most of the engineers who are still supporting Shuttle were trained before robust modeling and computer generated simulation capabilities existed. Their competency was achieved through time and experience as the Shuttle evolved. Another reason is because tasks often involve several engineers working together, increasing the overall training level of the engineers in an on-the-job training environment. A final reason is that modeling and simulation capability began growing at about the same time as the Shuttle program began winding down, and utilizing these new resources was not deemed necessary for an already highly trained and experienced work force.

This has resulted in modeling and simulation effectively being limited to software testing and tanking and countdown simulations. These integrated training sessions require a tremendous amount of resources to setup and run, yet provide limited training specific or targeted to a single user. It is also more geared toward simulation of the firing room protocol and not to developing expertise on any vehicle system itself. The current model for the shuttle can only be accessed via a single lab with limited time, and can only be viewed via a minimal interface and at a high resolution (cannot see circuits at diode level). Often the integrated model cannot precisely model individual system behaviors such pressure-temperature relationships at the component or subcomponent level.

As we move into future Programs which may be government or commercially run, it will be critical to get engineers trained quickly and at a reasonable cost. Manpower is typically the most expensive resource of any space program, and if we hope to travel to the Moon or to Mars we will have to do it by relying on fewer, but more highly trained engineers. In order to increase productivity from the work-force, cross training and dual role engineering is also becoming more prevalent. Rapid but effective training is required to make this work. It is also possible that if space vehicles do become commercial, there will be an assortment of vehicle configurations that will require continual training and changes to procedures to accommodate the different vehicle configurations.

Effective training requires several things. It requires that a user repeat a task over and over until it becomes second nature. Training requires that the user performs the task regularly over the years so as not to lose proficiency. It must be engaging and realistic to the degree that it stimulates the mind to want to continue learning while also being user-friendly. This means it needs to provide the correct information in a way that fits the needs of the user. Often the user needs to see the same information in different ways to fully understand the hardware/electrical/physics interactions within their systems.

A desktop simulator addresses most of these needs by putting the simulation capability at the desk of the user. It allows the tool to be available to the user whenever he wants to use it, without the requirement of tying up physical resources and working to the schedule of others. It separates the model from the graphical interface so that the displays can be built based on the needs of the system engineer and independent of the model. It will provide functionality to make training engaging and realistic because it is separated from the model and is developed for the

specific purpose of training. The simulator also allows the user to manipulate an item in one view and see it in another, allowing the same data/information to be viewed from different perspectives. Finally the simulator allows the user to log on and record training activity and performance. This is beneficial because it can be used to document training or work and demonstrate abilities or progress to the individual or to management. It also helps to identify to the student what is needed to complete training and ensures that each engineer receives well rounded and complete training.

Additionally the simulator can be used for more than just training. It can provide a way to assist with developing procedures by allowing visualization of different configurations or results from different console operations. It can also allow analysis of a system to determine expected behaviors and find single point failures or determine consequences of certain failures. Finally it provides a method to show others how a system operates using an easy to understand dynamic graphical interface.

While a custom model could be written to support the simulator, the model is not considered to be part of the desktop simulator. The desktop simulator is merely the graphical displays consisting of 3 basic view types, and tools that presents the model data to the user in a variety of ways and allow for the configuration, manipulation and interaction with the model. These interactions can be manual or automated, such as interactions triggered by a sequencer or some other software. The tool will have the capability for playback of data which may have been derived from the model or from real-hardware. There will also be tools to facilitate the training. This includes logging activity, but also includes failure scripts, scenario scripts, and additional training materials.

It is important to point out that a simulator cannot replace on the job training and experiences. What it can do is put the trainee in a position where he is more effective and equipped during that on the job training, and where he can observe situations that he may not get the opportunity to experience during on the job training. It also provides an additional level of system knowledge that cannot be obtained via on the job training but would be needed in the event of a problem or out of family condition.

Acknowledgments

Jeff May would like to thank Rick Zeitler USA MPS Subsystem Area Manager for encouraging me to pursue the use of software technology to improve training, and for reviewing and redlining this paper.