RAPID GENERATION OF SYSTEM MODELS FOR SIMULATION BASED DESIGN

Grant Conklin

90478466

Submitted for consideration in

Region 3, IEEE

Student Paper Competition

Endorsement of Branch Counselor

The author(s) of this paper are student members of this IEEE student branch and will still be undergraduate students at the time of SoutheastCon. This paper is the only entry of our student branch.

Signature: ________________________________

Name: Dr. Feng Zhao

Telephone Number of Counselor: (803) 777 - 6303
RAPID GENERATION OF SYSTEM MODELS FOR SIMULATION BASED DESIGN

ABSTRACT

When engineers are designing a very complex system, it would be greatly advantageous for them to be able to construct simulation models of every reasonable configuration for that system. This would allow them to fully explore the system's design space and choose the optimal configuration for implementation. However, this is not possible within a reasonable period of time using current simulation technology, which relies heavily on large amounts of low-level user input to construct simulation models. Consequently, it requires an enormous amount of time and effort to create large numbers of complex simulation models. A faster way of creating simulation models is needed, one that does not require such large amounts of user input. To solve an instance of this problem, we developed a simulation model generation application (SMGA) software tool at the University of South Carolina for generating simulation models for the electrical systems of ships using the Virtual Test Bed (VTB) simulation software suite. VTB was selected because it exposes its internal simulation model construction methods as a .NET library, and as a result can be easily leveraged to create a SMGA. The SMGA we created rapidly generates simulation models by allowing a user to specify small numbers of high-level parameters, which are then used by the SMGA to create, configure, and connect the appropriate templates. By automatically keeping track of certain statistics, our SMGA aids an engineer in adding a greater level of detail to a simulation model than they would otherwise be
capable of achieving within a reasonable timeframe. Our SMGA tool also reduces the possibility of accidental human error, and as a result greatly increases simulation model reliability.

**INTRODUCTION**

When a very complex system is being designed, the method most often used by contemporary engineers is to observe how comparable systems were designed in the past, and then choose a configuration for their new system which is reasonably similar. A simulation model of the system is then created and evaluated to provide a level of assurance that the system will actually function as designed. The reason for this approach is that the complexity of large systems means that exploring the entire design space of such a system in order to find the optimal implementation would take far too much time and effort even when using contemporary simulation software packages. The enormous amount of time required to create large numbers of complex simulation models using current simulation software technology means that only a limited number of system configurations can be tested. As a result, the system design actually selected for implementation is often not the best design; rather, it is the one that is similar to systems already known to function while still being "close enough" to the system design specifications.

However, if detailed simulation models could rapidly be created for multiple possible system configurations, it would become possible to test large numbers of possible system designs in order to determine which configuration is optimal. This would require a software tool capable of rapidly generating simulation models for system configurations which are potential solutions to a particular system design problem. This would greatly facilitate the exploration of the system's design space, and thus the rapid selection of the optimal solution to
the design problem. This is not possible given current simulation technology, however; contemporary simulation software packages rely on a user building each simulation by means of a scripting language or by interacting with a graphical user interface (GUI). A faster way of creating simulation models is needed, one that can generate many complex simulation models of the particular system being designed without requiring a large amount of user input. At the University of South Carolina, we have developed such a software tool, targeted at designing electric ship systems using the Virtual Test Bed (VTB) Simulation Software Suite. This paper details the theory behind this tool, how we created it, and how an engineer can use it to arrive at an optimal configuration for the particular system they are designing.

**THEORY**

We began the process of creating a simulation model generation application (SMGA) by defining an entirely different user input architecture in which a user specifies small numbers of high-level parameters which are then used to configure and connect templates together in order to generate simulation models. The SMGA is responsible for intelligently filling in the details between the user’s specifications as necessary in order to generate the simulation model, thus saving the user an enormous quantity of time. We continued the process of creating the tool by identifying recurring patterns in the target design space which were then leveraged to greatly decrease the amount of time required to explore that design space. This was accomplished by assuming that the optimal system design process would follow a path through the system design space determined by these recurring design patterns (Figure 1). These design patterns were then implemented as templates within our SMGA.
Figure 1: Identifying patterns in a system's design space makes it possible for us to create a software application to rapidly explore it and choose an optimal system configuration.

We rapidly developed our SMGA by building on top of an existing simulation software package (Figure 2). This allowed our SMGA to use the simulation software’s libraries of components, file formats, and analysis tools. Most importantly, however, it allowed our SMGA to leverage the existing internal software methods already used by the simulation software package to create simulation models from user input. This allowed our SMGA to leverage existing code to create simulation models while bypassing the simulation software's original
GUI through an entirely different user input architecture. In this way, our SMGA was created without us needing to resort to fabricating an entirely new simulation software package.

Figure 2: Building a SMGA on top of an existing simulation software package allows for full access to the simulation software's resources while allowing us to define a new user input architecture.

The reason we created our SMGA in the first place was to rapidly find the optimal system configuration for a given set of design specifications. As a result, the design process for use with such a tool must also revolve around the same goal (Figure 3). This design process emphasizes the evaluation of as many design options as possible in the smallest feasible period of time.
Figure 3: The design process for use with a SMGA. The goal is to rapidly find the optimal system configuration for a given set of design specifications.

SYSTEM DESIGN APPLICATION

At the University of South Carolina (USC), we developed an application utilizing the concepts presented in this paper to generate simulation models for the electric systems of ships. The process of how we created this application and the means by which it can be leveraged to design electric systems for ships will serve as an example for how the previously discussed theoretical concepts can be practically implemented.

Choosing a Simulation Software Package

We chose the Virtual Test Bed (VTB) simulation software suite for the reason that it has the unique feature of having its internal simulation construction software methods exposed as
a .NET library in the form of a DLL. This made it easy for us to access and leverage V TB’s resources to build our SMGA.

**Identifying Design Patterns within the Design Space**

In order for a software application to be capable of generating simulation models, it has to have templates with which to construct the simulation models. These templates represent recurring patterns within the given design space, which must be identified before the templates can be created. Within the design space of ship power distribution systems, the identification of design patterns is achieved by indentifying the basic building blocks common to all relevant system configurations. The building blocks common to all ship power distribution systems are generators, load centers, the individual loads connected to each load center, a main power distribution bus, and a propulsion system (Figure 4). Further patterns can be identified in the form of the power distribution architectures used within each building block. We chose these power distribution architectures to be: three phase AC, ungrounded bipolar DC, and monopolar grounded DC. Once we identified all of these design space patterns, we implemented them in the form of templates, with each template representing one of the system building blocks. We created each template in three versions corresponding to the three different power distribution architectures chosen. We configured our SMGA to be capable of generating interfaces between different templates using any of these three power distribution architectures, so simulation models could be generated using whatever combination of power distribution architectures the user desires.
Figure 4: The relationships between the recurring design space patterns of an electric ship, which we implemented in the form of templates.

Creating the Simulation Model Generation Application

A SMGA must be capable of generating and/or connecting together templates in order to form a system based on a user’s specifications. The application we developed at USC uses a GUI to take high-level user parameters (Figure 7), and then creates the required templates for a ship's electric system using three different methods: The load center and generator templates are user-defined, but must already exist in the application's template library (a user can also create new versions of these templates if they so desire). The main power distribution bus and propulsion system templates are generated by the application from parameters which the user
specifies in the application's GUI; the load templates are generated from parameters which the user specifies in an Excel worksheet (Figure 6). The reason we chose an Excel worksheet as the means by which a user specifies load template parameters was that it makes it convenient for a user to quickly specify parameters for large numbers of loads. After all of these templates have been loaded/created and configured according to the user's specifications, the SMGA connects them all together using the appropriate power distribution architectures, installing the correct power converters if there is a mismatch in the power distribution architectures at any junction between templates (see the appendix for examples of the templates used by the application to generate simulation models).

Figure 5: The process which our SMGA uses to generate a simulation model for a ship's electrical system.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Subsystem Connector Name</td>
<td>Bus Type</td>
<td>Bus Voltage (V RMS)</td>
<td>Component Type</td>
<td>S(W,V,A)</td>
<td>Power Factor</td>
<td>R (Ω)</td>
<td>L (mH)</td>
<td>C (μF)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>AC2_STARBOARD_FAIL_PORT_BUS</td>
<td>Three Phase AC</td>
<td>13800</td>
<td>R</td>
<td>250</td>
<td>0.5</td>
<td>50</td>
<td>10</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>AC2_PORT_FAIL_STARBOARD_BUS</td>
<td>Three Phase AC</td>
<td>10000</td>
<td>RC</td>
<td>600</td>
<td>0.5</td>
<td>50</td>
<td>10</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>AC2_STARBOARD_BUS</td>
<td>Three Phase AC</td>
<td>5160</td>
<td>R</td>
<td>500</td>
<td>0.55</td>
<td>50</td>
<td>10</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>AC2_PORT_BUS</td>
<td>Three Phase AC</td>
<td>520</td>
<td>RLC</td>
<td>50</td>
<td>0.5</td>
<td>50</td>
<td>10</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6: Template parameters for loads are specified in an Excel worksheet.
Figure 7: Our SMGA's GUI. This is where a user specifies template parameters.

We also endowed The SMGA with several features designed specifically to resolve several issues which we found engineers were having problems with. First, having circuit breakers in a large electrical system is a standard safety requirement, but from our own personal experience, we noticed that most people left them out of simulation models simply because it was too difficult to accurately calculate the breaker trip currents. This was due to the difficulty of keeping track of the power consumption in all branches of the circuit. Also, if an engineer decided to modify the loads being used in the circuit, all the breaker trip currents leading to that branch of the circuit would have to be re-calculated, and this took too much
time to be practical. As a result, we added a feature which allowed the SMGA to keep track of the total power production of all generators as well as the total power consumption of all loads in a simulation model (Figure 8). Knowing the power statistics for each branch of the ship's electrical circuit allowed the SMGA to accurately calculate all of the required breaker trip currents, which allowed circuit breakers to be added to the simulation and consequently increased the level of realism possible in the simulation models. This also allowed a user to easily determine if their system model was underpowered or overpowered. The other feature we added was to allow a user to save/load files containing all of the settings used to generate a certain simulation model. This was useful if a user wanted to quickly change an existing simulation model. For instance, if a user wanted to add some more loads, or change a power distribution architecture in a section of an existing simulation model, they could load the settings used to generate that simulation model, make the required changes, and then simply re-generate the simulation model. All templates would be re-evaluated, all connections reviewed, and all template parameters automatically re-calculated to account for any changes in the system. This allowed engineers using the software to collaborate with each other to rapidly change, modify, and evaluate different system configurations.

Because it requires so much less user input, our SMGA also empowers a user to create a simulation model with a much greater level of confidence in its accuracy, since the possibility of human error is greatly decreased. As a result, a simulation model generated by our SMGA tool will be far more reliable and the results from such a model far more trustworthy than those from a simulation model built by a human.
Figure 8: Our SMGA also keeps track of the power production/consumption statistics for a simulation model.

FUTURE POTENTIAL

The future possibilities of SMGA tools as it regards design applications are essentially limitless, since patterns can be identified in any design space and leveraged to save time in a similar fashion to the example we presented in this paper.

Software engineers tasked with creating such tools in the future will have three major choices to make. First, what detail level(s) will the tool use when generating simulation models? Such tools are potentially able to construct system models of any detail level desired, from overall approximations to extremely detailed system designs. Secondly, what types of subsystems within the overall system will the tool be capable of including in the simulation models it generates? We created the SMGA presented in this paper to model the electrical systems of ships, but it would be entirely possible to create a multidisciplinary tool that could, for example, generate simulation models which included not only electrical systems, but also control systems, thermal management systems, etc. for a specific design application. Thirdly, since many different types of systems share similarities within specific subsystems, it would be possible to create a tool that could generate simulation models for a specific subsystem present
in many different types of systems. For instance, power grids, electric vehicles, and aircraft all possess electric power distribution systems. A tool could be created which was designed to generate simulation models of electric power distribution systems in general, and as a result could be used in the design of any system which possessed an electric power distribution subsystem. If this idea were carried one step further, several of these specific-subsystem SMGAs could be linked together to create a larger SMGA capable of generating multidisciplinary simulation models for a wide variety of design applications.

CONCLUSIONS

Using contemporary simulation technology, an engineering team has a limited ability to explore a system's design space due to time constraints. The simulation model generation concepts presented in this paper will make it possible for an engineering team of the future to confidently design and implement optimal system configurations with the knowledge that the exploration of large design spaces can be accomplished within a reasonable timeframe. We demonstrated this by showing how we implemented these concepts in the form of a software tool for generating electrical system simulation models for ships. Similar software tools could be created for many possible system design applications, and configured to generate simulation models using multiple engineering disciplines and varying levels of detail.
APPENDIX

Figure I: A generated propulsion system template with a three phase AC power distribution architecture.

Figure II: A generator template with an ungrounded bipolar DC power distribution architecture.
Figure III: A generated main power distribution bus template with an ungrounded bipolar DC power distribution architecture.