ADVANCING AUTOMATION IN EARLY-STAGE NAVY SHIP SYSTEM DESIGN

Julie S. Chalfant[1]

ABSTRACT

The increased complexity and impact of ship system performance has made ship system design an important part of the early stages of ship design. This paper provides an overview of templating, describes a methodology to include ship system design in the Navy's early-stage design process, and explores the required algorithms underlying the templating process.

KEY WORDS

Design; Ship systems

INTRODUCTION

Ship systems, such as the electrical distribution and thermal management systems, are larger, more complex, and more integrated than ever before due to the radical increase in electrical power used by new sensor and weapons systems, the resulting large thermal load placed on cooling systems, and the advances in integration of ship, mission and machinery control systems. Thus, there is a significant need for greater detail in ship system design to be provided earlier in the ship design process. Advances in computing capability over recent years allow an increase in detail of early-stage ship designs along with a simultaneous increase in the number of ship designs explored, facilitating design processes such as set-based design.

The Navy currently employs a suite of early-stage design tools consisting of an overarching design space exploration tool, termed the Rapid Ship Design Environment (RSDE), which runs a number of modular programs that perform design and analysis functions for different aspects of the ship design. All of these programs persist design data in the Leading Edge Architecture for Prototyping systems (LEAPS) data repository using a product meta-model that defines the storage of surface ship data, titled the Formal Object Classification for Understanding Ships (FOCUS). The current design tools produce a hullform using the Hull Form Transformation Utility to support synthesized ship design using the Advanced Ship and Submarine Evaluation Tool (ASSET), then evaluate resistance and seakeeping performance using Ship Hullform Characteristics Program - LEAPS and Integrated Hydro Design Environment. See Doerry [1] for more details. Until recently, the Navy's tools did not include a system modeling and analysis tool that details the components and provides synthesis capability.

In recognition of the need for including systems earlier in the ship design cycle, the Office of Naval Research (ONR) worked with the Electric Ship Research and Development Consortium (ESRDC) and the Naval Surface Warfare Center Carderock Division (NSWCCD) to develop a software suite for the design and analysis of ship systems. The resultant software environment, titled Smart Ship systems Design (S3D), is LEAPS- and FOCUS-compatible and is currently being adopted into the RSDE design suite by the Navy ship design community. S3D significantly enhances the state of the art for simulation capability in the early stages of ship design; within S3D, a fully-connected shipboard distribution system can be created by selecting components from an equipment library, connecting them logically in a one-line-diagram view, placing them in three-dimensional space in a ship model, and running load-flow-level simulations on electrical, thermal and mechanical views of the systems, for both individual alignments and more complex mission scenarios. This capability may be used to create new systems from scratch or to flesh out rudimentary distribution systems created in other design programs such as ASSET. However, at the current time, this process demands a person-in-the-loop; the creation of such systems is a manual process.

In order to incorporate the system design and analysis capability into the design-space exploration paradigm of the Navy's early-stage design process, a methodology for automation of system design is needed. This paper describes a body of work that provides a methodology for semi-automated design of ship systems, allowing the programmatic

[1] Sea Grant Design Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. email: chalfant@mit.edu
This material is based upon research supported by, or in part by, the U.S. Office of Naval Research (ONR) under award number ONR N00014-16-1-2945 Incorporating Distributed Systems in Early-Stage Set-Based Design of Navy Ships; ONR N00014-16-1-2956 Electric Ship Research and Development Consortium; and by the National Oceanic and Atmospheric Administration (NOAA) under Grant Number NA14OAR4170077 - MIT Sea Grant College Program. Approved for public release under DCN# 43-9349-22.

creation and analysis of ship systems under the guidance of the user, assembled from pre-designed templates and tailored to the ship design. We refer to this overall methodology as *Templating*. The ultimate goal is a software tool which takes as input a set of pre-designed system segments, termed templates, and integrates them into a fully functioning system model in a ship design, with all components appropriately sized and located. The resultant system model provides metrics such as size, weight and complexity. Further, the model is available for system simulation under various operational conditions to provide metrics such as efficiency and survivability while also allowing exploration of reconfigurability, reliability, maintainability, and a host of other "ilities."

The process for creating a fully functional ship system from templates requires several steps:

Assembly of the templates into a logically connected system by copying relevant templates into the ship design and connecting them appropriately to one another. This yields a logically appropriate one-line diagram with components placed in an approximate geographic position within the ship.

Determination of the capacity of each component. Since the templating capability facilitates the creation of ship systems from an assembly of parts or system sub-sections, it is not possible to determine the required capacity of each element of a system until the system is fully assembled with all loads and sources connected and placed in three-dimensional space. An algorithm has been developed to determine the maximum amount of energy handled by each component given any possible alignment of the system.

Dimensioning of each component based on the capacity required. Physics-based sizing algorithms for a variety of component types are under exploration.

Final placement of the components in three-dimensional space. A methodology for automatically arranging components in a ship design in a manner that eliminates overlaps, provides spacing between components, and minimizes connection length has been developed.

This paper provides an overview of the templating process and the algorithms underlying each step and discusses use cases for application of templating to advance automated system design in the early-stage ship design process.

SMART SHIP SYSTEMS DESIGN (S3D)

The Smart Ship Systems Design (S3D) software environment facilitates the construction of ship system designs in four domains: electrical, mechanical, piping, and ducting/air. Each domain is represented by a two-dimensional one-line diagram view. Further, CAD renderings of components can be used for three-dimensional arrangement and analysis.

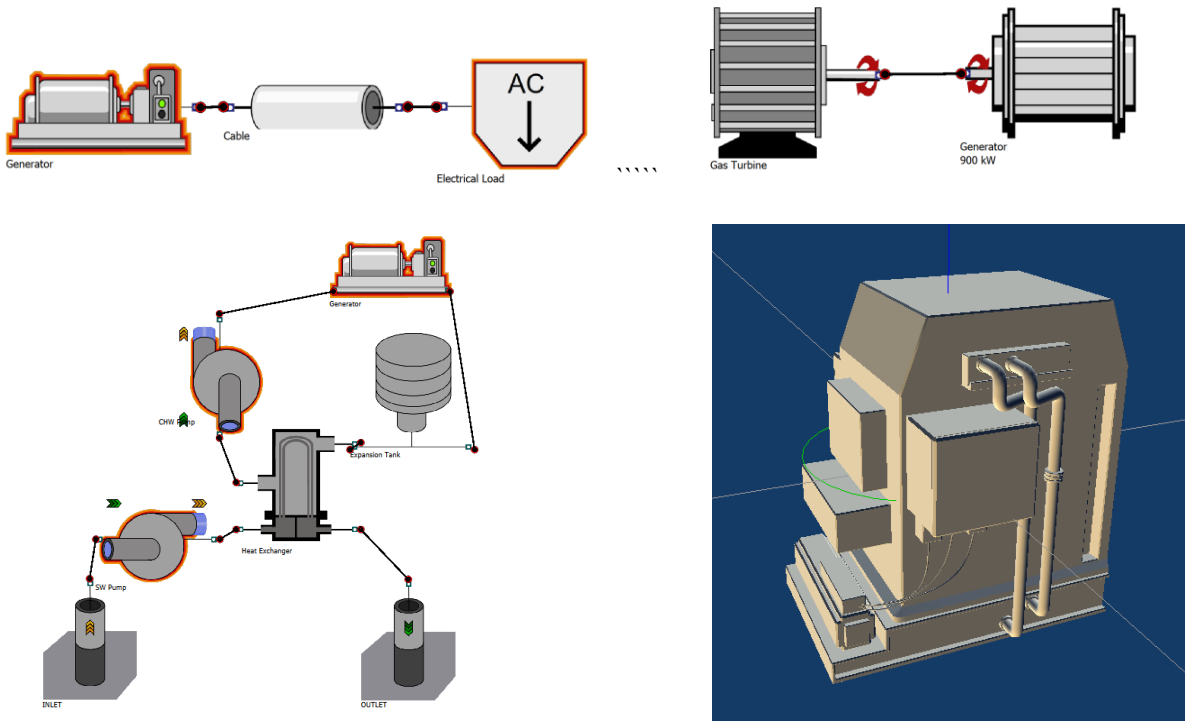

Figure 1. A generator with representations in the electrical domain (top left), mechanical domain (top right), and piping domain (bottom left), along with a CAD representation (bottom right).

The current version of S3D addresses the electrical, mechanical and piping domains. Preliminary work has been accomplished in the ducting/air arena, but this work is not currently integrated. RSDE contains some CAD functionality, which will likely replace the early work in S3D for three-dimensional representation.

A single component has separate mathematical representations in each of the applicable domains, supporting a power-flow level analysis and co-simulation of the system. For example, a generator may have an electrical model representing the generation of electrical power, a mechanical model representing the generator connection to a prime mover such as a gas turbine engine, a piping model representing the cooling water flow, and a CAD model showing the space requirements; see Figure 1. Components are connected to one another via ports, shown as red dots in Figure 1. Properties are associated with the components and the ports, and are used in the simulation of the system. Component properties address the component as a whole; for example, a generator may have properties for rated electrical power, rated mechanical power, electrical efficiency, piping pressure loss, dimensions and weight. Port properties address the flow of a commodity through that port; for example, an electrical port may have properties for voltage and frequency, whereas a piping port may have properties for diameter and flow rate.

Some properties can be modified by the user, while others are dictated by the solution of the system. For example, a component may have an efficiency curve that provides the efficiency of the component as a function of load; this property is modifiable by the user. The component may also have an efficiency property, which denotes the efficiency at a moment in time for the load on the component. The component load is determined from the solution of the system of equations governing the behavior of the set of connected components that make up a ship system, and the efficiency property is populated by the software selecting the efficiency from the efficiency curve based on the calculated load.

The electrical and mechanical system solvers perform a steady-state load-flow study of the interconnected systems. The electrical solver determines real and reactive power and the magnitude and phase angle of the current flowing through each node. The mechanical solver determines power and rotation rate flowing through each node. The piping solver determines flow rates, pressures, and temperatures of the fluid flowing through each node. Details of the solver performance can be found in [2].

A large library of components is available for selection and addition to a project. At present, the electrical components include a variety of generators, loads, cables, converters, energy storage devices, and electrical protection; mechanical components include motors, shafts, propulsors, gears and engines; and piping components include chillers, heaters, tanks, pumps, valves, pipes, and tees. Additional components are under development.

TEMPLATE DEFINITION AND CREATION

A template is a reusable system or portion of a system composed of components connected to one another. A template may exist within a single domain such as electrical or mechanical, or may have representations in several domains. A single template can represent a full ship system, a sub-system, or a zonal or spatial portion of a larger system; these examples are further described in the Templating Use Case Examples section later in the paper. Templates are connected together to create a fully defined ship system with all components connected and located within the ship hullform. As an example, Figure 2 shows a template for a power generation module, consisting of a dual-wound generator, two rectifiers, and cables for connection to the electrical distribution system. This template could be combined with templates for ship-wide electrical distribution, in-zone electrical distribution, and specific ship loads to create a full representation of a full shipboard electrical power system.

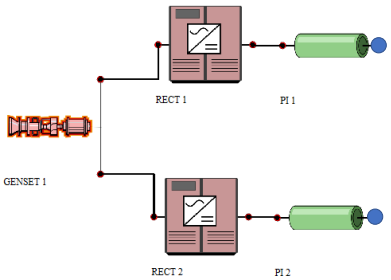

Figure 2. Sample template.

Templates are created within S3D by assembling individual components into a full system or a portion of a system. Currently, templates are stored as designs within a LEAPS database. However, since it is not possible for more than one

process to access a single LEAPS database simultaneously, another storage mechanism such as an .xml file may be required in the future.

Template Properties

To fully implement the templating procedures described herein, some unique characteristics of a template are required in addition to a normal S3D design. See Figure 3 for an example template concept with properties denoted as shown in the LEAPS editor.

The *Template Identifier (templateID)* establishes the provenance of items added by the templating process. This identifier along with an instantiation number is associated with each component when it is copied into a ship design or into another template. The instantiation number is required for cases in which a single template is applied multiple times into the same design.

Template dimensions may be associated with a template. These are used to scale the position of components when placing them into a ship design. The specific template dimensions are template overall length, template overall breadth and template overall depth. These define a box within which the components are located in the template. When a template is copied into a ship concept, the component locations can be scaled to the size of the space into which they are copied; the template dimensions are used to accomplish this scaling.

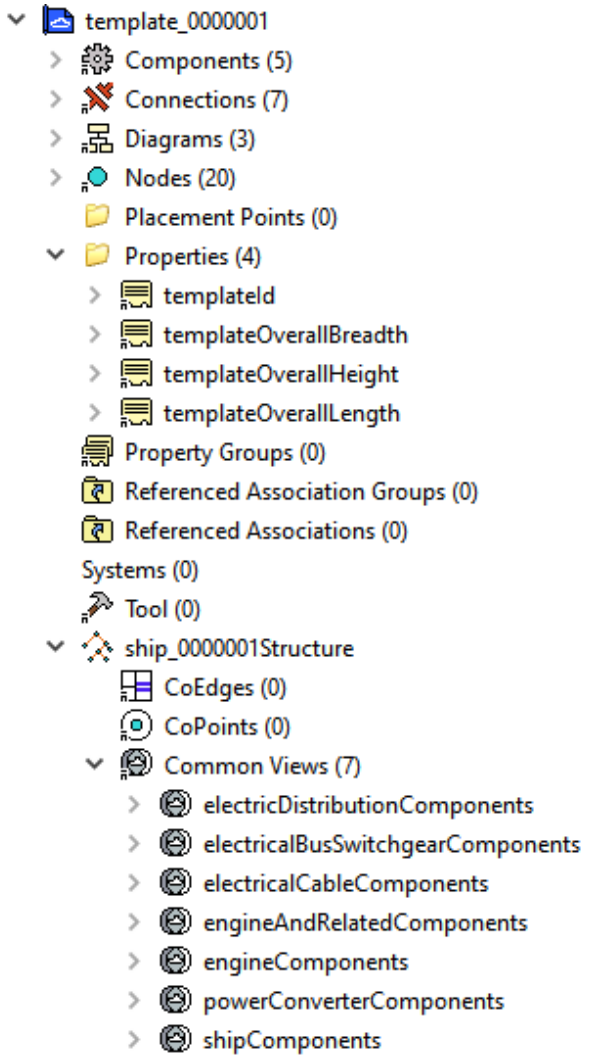

Figure 3. A template shown in LEAPS editor. Note the template-specific properties of templateId, templateOverallBreadth, templateOverallDepth and templateOverallLength.

Template Node Properties

Templates are connected to other templates or to components at *Template Nodes*, shown as blue circles in Figure 2. To fully characterize the connection and ensure that the template connection occurs at the proper location, two new properties are associated with the template node.

A template node *Descriptive Name (descriptiveName)* is used to ensure that template nodes are connected properly. All nodes in S3D have a node type identifier (portTypeId) that identifies it as, for example, an electrical node or a piping node; only nodes of like portTypeId's may be connected to one another. Further identification is required in the templating process because two nodes of the same type may have different purposes. As an example, consider the power generation module represented in the example template shown in Figure 2. This template contains two electrical template nodes, one of which is intended to be connected to the port bus and one to the starboard bus. These two nodes will have the same portTypeId, designating them as electrical ports, but different descriptiveNames, denoting their different connection locations.

Template node descriptive names are designated by the user who creates the template, thus allowing templates to be designed for purposes not previously considered. For example, a current paradigm may suggest port bus and starboard bus as reasonable descriptive names for a standard dual-bus architecture, but a small autonomous underwater vessel may have a single central bus, and a multi-hull vessel may have port and starboard buses in each of three hulls. Leaving the naming convention open allows future development currently unforeseen.

A *template node Plug Direction (plugDirection)* is required to differentiate between several plugs with the same descriptiveName. The plug direction does not denote the direction of flow at the plug; rather, it is a device to facilitate proper connections. Plug directions are denoted in pairs, e.g. "in" paired with "out". A plug is only allowed to connect with a plug of the opposite direction. The plug pairs used in creating several example systems include forward-aft, up-down, inboard-outboard, and in-out. Additional plug direction pairs may be added as defined by the user. Plug connections are allowed between plugs with the same descriptive name and opposite directions. Connections are prohibited between plugs within the same template instance, thus preventing loops.

As an example, consider several sub-systems, all of which need to be plugged into the port bus. If all the appropriate template nodes have the same descriptive name, "portBus", including the nodes on the port bus switchboard to which these loads are to be connected, then it is possible that the nodes of the loads would be connected to one another rather than connected to the switchboard. This problem is precluded if the switchboard nodes have the plug direction "inboard" and the load nodes have the plug direction "outboard."

Template Availability within RSDE

When fully integrated within the RSDE toolset, it is envisioned that a set of pre-defined templates will be available for selection from a library of templates. Prior to beginning a ship trade study, additional templates can be created and stored in this library by the user and become available to the program during the trade space exploration process. In addition, templates can be refined using S3D to upgrade or replace components manually and to define properties on already instantiated components. Alternatively, templates may be refined by selecting a component in the template and replacing it with another template, or selecting a sub-template and replacing it with another.

TEMPLATING STEP 1: ASSEMBLING TEMPLATES INTO SYSTEMS

At the start of the templating process, a set of templates are available in a "template database", and a "ship database" exists into which these templates will be placed. The ship database is likely to be populated with a hullform, decks, bulkheads and superstructure and may have some components placed prior to beginning the templating process; however, the amount of data pre-populated into the ship database is flexible. The templating process works equally well with a completely empty ship database or with a heavily populated ship database.

The first step in the templating process is to copy templates into a ship design concept in the ship database, transferring all data to the new design and integrating the new information with any pre-existing information. For example, if a template adds definition to an existing electrical distribution system, the new components and connections are added to the ship concept's existing electrical distribution system diagram and the components are stored in the existing pertinent common views and systems, instead of creating new diagrams, common views or systems. Diagrams, common views and systems are LEAPS methodologies for categorizing data pertinent to a design; further definition can be found in [3].

Templates are copied into the ship database in a component-centric manner. After ensuring that all common views, systems and diagrams in the template exist or have been created in the ship concept, each component in the template is deep-copied into the ship database. A deep copy indicates that all of the properties, property groups, tool data, and nodes associated with the component are replicated in their entirety in the ship concept, recursively including all of their respective properties, property groups, and tool data through all possible sub-levels. The ship component is assigned to any ship concept common views and systems with which the template component had been associated in the template. Connections between components in the template are re-created in the ship instantiation and added to the corresponding ship diagram.

A component in a LEAPS concept includes a location and orientation relative to the coordinate axes of the concept; see Figure 4. If desired, a template may be assigned to any of the structural divisions of a ship such as a single compartment, a zone, or an entire hull, here termed a shipboard space. If no location is selected, the template is applied to the ship concept with no adjustment to component locations; in this case, components are placed using the location and orientation stored in the template. If a template is placed in a shipboard space, the component locations are translated so that the centroid of the template coincides with the centroid of the selected shipboard space. Further, the user can opt to scale locations relative to the selected space, in which case the centroid of the template is matched to the centroid of the selected shipboard space, the overall dimensions of the template are matched to the overall dimensions of the shipboard space, and all component locations are scaled using the resultant values.

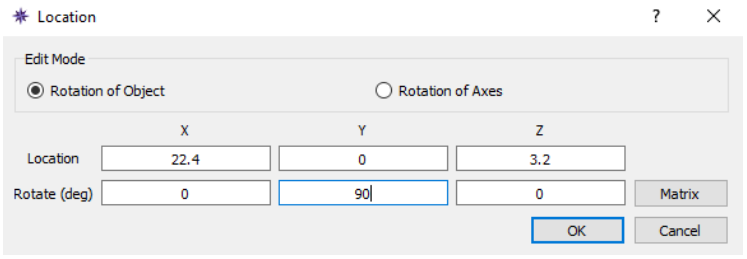

Figure 4. Component location and orientation.

Once the template is fully copied into the ship concept, template nodes are connected to one another as necessary. Note that since this is accomplished in an automated manner, the template node connection needs to be set up such that it is not possible to connect the node incorrectly. The framework to accomplish this is established using the node descriptiveName and plugDirection properties described above; however, the user who designs or selects the templates must be cognizant of these conventions in planning the future use of the templates.

Some additional administrative actions must be taken as well. For example, each component and connection has a global unique identifier (GUID), which must be updated when a new item is instantiated. All new components and connections must be placed into the S3D schematic view. Any links or references to ship data must be updated. For example, a propeller component may have a link to the ship's speed-power curve; this link must be updated to reflect the new ship data.

TEMPLATING STEP 2: COMPONENT SIZING

Since the templating capability facilitates the creation of ship systems from an assembly of parts or system sub-sections, it is not possible to determine the required capacity of each element of a system until the system is fully assembled and placed in three-dimensional space. Variations in the sources and loads that make up the system and variations in the topology into which they are assembled will affect the capacity of each component. Further, the location of components will affect the length of distribution components such as cables, pipes or shafts, which will extend or contract in length as components are placed farther apart or closer together. This in turn will affect the losses in the distribution components, as they are usually calculated on a per-unit-length basis.

To determine the capacity of each component, it is necessary to determine the maximum amount of power that can flow through each component *given any possible alignment of the system*. An algorithm to determine this power requirement using graph theory is described in Chalfant et al. [4] and described further below. This algorithm is applied to a fully connected system, and the rated power of each component is set to the value determined by the algorithm. If desired, a margin can be applied at this point.

The dimensions and weights of components are updated at this point using sizing algorithms based on such properties as rated power, voltage, temperature, or flow rate.

Note that the algorithm determines maximum possible power using the full connected load, which most likely over-sizes the system. Further research is required into the development and application of algorithms that address the actual usage profiles of connected equipment.

Maximum Flow Algorithm

It is possible to rapidly approximate the amount of power flowing through each component in a system for a given operational alignment by linearizing the component impacts and solving the resultant system of linear equations; however, the flow must be determined under any possible alignment of the system. Automatically determining the maximum flow by solving all possible permutations of switch positions produces a set of $2n$ solutions, where n is the number of switches, which rapidly becomes unmanageable.

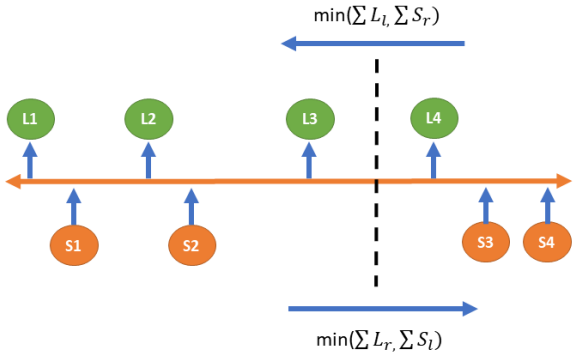

Figure 5. Push-pull algorithm example

Doerry [5] introduced an algorithm to determine maximum flow in a single bus, which is summarized as follows.

At any point along the orange line shown in Figure 5, the maximum flow from left to right is the minimum of the sum of the source to the left or the sum of the load to the right, because the source must have a load to draw the power (load limited), and the load must have a source to supply the power (generation limited). Similarly, the maximum flow from right to left is the minimum of the sum of the source to the right or the sum of the load to the left. The overall maximum flow through any point is then calculated as the maximum of the two flows (left to right or right to left).

$$MaxFlow = max \begin{pmatrix} min(\sum S_R, \sum L_L) \\ min(\sum S_L, \sum L_R) \end{pmatrix} \quad [1]$$

Using equation (1) on the system shown in Figure 5 and assuming that L1 = 1, L2 = 2, S1 = 1 and so on, the maximum flow through the dashed line is calculated as shown in equation (2)

$$\begin{aligned} MaxFlow &= max \begin{pmatrix} min(S3+S4, L1+L2+L3) \\ min(S1+S2, L4) \end{pmatrix} \\ &= max \begin{pmatrix} min(7,6) \\ min(3,4) \end{pmatrix} = 6 \end{aligned} \quad [2]$$

This can easily be expanded to an acyclical network. Using the same algorithm, the maximum power flowing through any edge of a graph can be determined by summing all upstream and downstream loads and sources.

There is a complication in a real system in that there are losses along the path in each component that is traversed, so following a path through the network adds power demand or reduces power supply due to losses in the components. Therefore, the maximum power at a point in the network can be determined by modifying equation (1) so that the sum of the sources is calculated by tracing the shortest (lowest loss) path from each source to that point, and the sum of the loads is calculated by tracing the longest (highest loss) path from that point to each load; this is represented in equation (3), in which l_{Xi} is the sum of the losses along a path from the point of interest to the i^{th} source. Note that the highest weight path is not necessarily the longest path; it is the lossiest path.

$$MaxFlow = max \begin{pmatrix} min \begin{pmatrix} \sum_i (S_i - min(l_{Si}))_R \\ \sum_i (L_i + max(l_{Li}))_L \end{pmatrix} \\ min \begin{pmatrix} \sum_i (S_i - min(l_{Si}))_L \\ \sum_i (L_i + max(l_{Li}))_R \end{pmatrix} \end{pmatrix} \quad [3]$$

Shipboard distribution systems are not necessarily acyclical networks; in fact, resiliency and redundancy constraints almost guarantee cycles within the networks. To determine the maximum power at any point in a cyclical network, one

must find the longest (highest weight) path from that point to each load and source, without cycling to repeat any portion of the path.

Unfortunately, determination of longest path in a cyclical network is an NP-hard problem, which cannot be solved in polynomial time. There are a few possible solutions to this difficulty. One is to solve each likely configuration of the system; this requires a manual input of configurations to be solved. A second possibility is to run the algorithm on the system as-is despite the NP-hard challenge; since ship systems are on the order of hundreds of nodes rather than millions of nodes, a sufficiently powerful computer will be able to solve the system in tens of minutes. A third possibility is to run the algorithm using the shortest path rather than the longest path to each load, which potentially underestimates the maximum load possible. The potential underestimation can be overcome by judicious application of a design margin to the results. The solution implemented herein is as follows.

In the early stages of design, when thousands or millions of designs may be automatically created and compared, a speedy solution is required. Therefore, in early-stage design, the algorithm is applied using shortest path rather than longest path to each load. The error induced is the difference in losses over the two paths, which is typically very small. Most likely, this error is merely a difference in length of the distribution components. In the extreme, another lossy component such as a converter may be encountered. Since the design margin is typically 20% and the additional losses are typically less than 5% and often well below 1%, this is a reasonable assumption.

In the later stages of design, when refinement of the system is required, then one of the other options would be recommended. In most cases, the automated determination of the longest path in the entire system is solvable in a reasonable amount of time. If the solution is exceeding the desired time, then manual input of possible configurations is possible.

Algorithm Application

The application of the maximum flow algorithm to a typical S3D electrical system diagram is accomplished by representing the system as a graph in which each component (including distribution components) is a vertex of the graph and each electrical connection is an edge. Sources and loads are set to their rated (maximum) power. Losses in each component are linearized and set to the value they would see at maximum rated power. For example, load-dependent losses in a power converter are set to the percentage loss the converter sees at maximum rated electrical power. Path to each load and to each source is traced using a breadth-first search. The rated power of each component excluding sources and loads is then set to equal the value determined from the maximum flow algorithm as shown in equation (3), increased by a margin set by the user.

Once the maximum power required of each component is determined, sizing algorithms are used to determine the dimensions and weights of the components. At present, only a few components have sizing algorithms available; this is an area of future work.

TEMPLATING STEP 3: COMPONENT POSITIONING

At this point in the process, components have been added to a ship concept and are roughly located in three-dimensional space; however, components may overlap one another or intersect with the ship structure. The goal of this step is to resolve all overlaps while retaining components within the designated space. There are several use cases for this functionality, for example:

- Components have all been placed at the centroid of a space. They should be separated so they do not overlap, and placed in a manner that considers dimensions, interactions and connections with other components; the other components may be located within the space or in other spaces.
- Some components have fixed locations; other components need to be arranged around the fixed components in a manner that considers dimensions and interactions between components.
- Components are placed in approximately the right location, but small overlaps must be resolved.

Distribution components such as cables and piping, whose lengths are determined by the components they connect, are not placed in this step; rather, they are sized after non-distribution components are placed. One of the goals in this step is to minimize the length of distribution components while satisfying other constraints such as survivability.

The algorithm used for this purpose is a force-directed graph-drawing algorithm. In simple terms, the algorithm assigns repelling forces to all components and attracting forces to only connected components, with a goal of equal separation between all components. Further, components are restricted to fall within the confines of the designated space, so repelling forces cannot force components out of the space entirely. Individual components may be designated as fixed, in which case their location does not change.

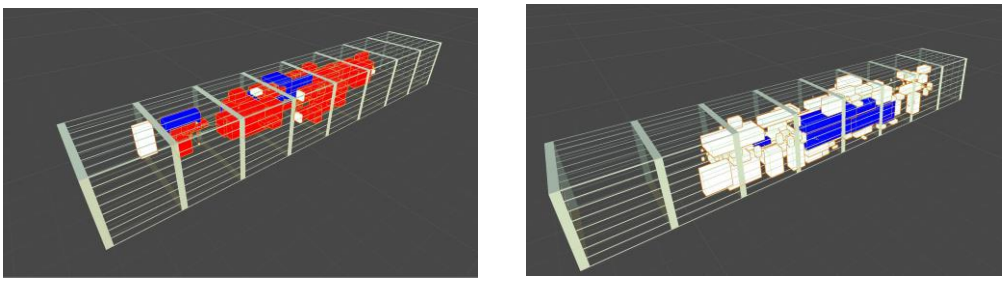

Figure 6. Component placement algorithm example with 248 components: beginning position (top) and final position (bottom). Blue components are fixed, red components overlap, and white components are neither fixed nor overlapping. Note that all collisions are resolved and components remain within the boundaries.

Figure 6 shows a set of components before and after the component placement algorithm has been run. Fixed components are shaded blue, overlapping components are shaded red, and components that are neither fixed nor overlapping are shaded white. It can be seen in the image that all overlaps have been resolved. Details of the algorithm along with examples can be found in [6].

TEMPLATING USE CASE EXAMPLES

Several possible use cases are described below.

Full System Copy

A complete ship system with all energy domains, all components (combat systems, sensors, mechanical, electrical, etc.) and all connections, is pre-designed and copied as a whole into a new ship hull. The topology and connectivity of the systems and components and the number of zones remain unchanged.

An affine transformation of the equipment locations to match the dimensions of the new ship hull must be accomplished and the lengths of distribution components must be adjusted as well.

Some components depend greatly on the size of the ship hull; for example, the required power to propel the ship at various speeds will change, so the size and power requirements of the drive train must be sized accordingly. These changes may then require changes to the components in support systems such as electrical power or cooling. The sizing algorithms described above will adjust components accordingly.

The affine transformation of equipment locations and any resizing of components due to changes in requirements may result in collisions or overlaps; therefore, the component positioning algorithm must be run as well.

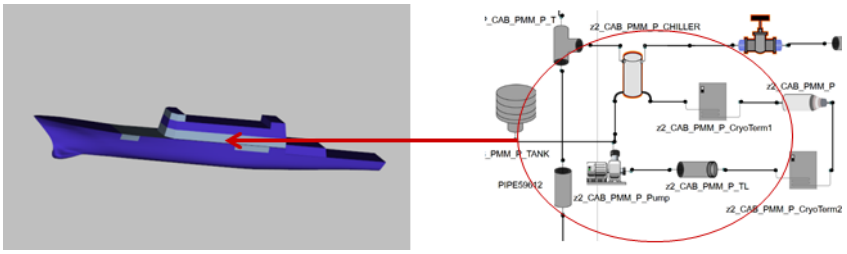

Figure 7. Full system copy

Assembly of Zones or Compartments

A geographic section of a ship system design, such as a single zone or compartment, consisting of all components and connections in all energy domains is copied into a ship and connected to existing systems or to templates of other zones. As an example, several possible zones could exist in a pre-designed state and are combined into a full ship system architecture. The zones could be arranged in a different order or multiple copies of a zone could be instantiated in different locations.

In this example, each template is copied into the ship concept, the centroid of the template components is shifted to match the centroid of the zone, affine transformation of component locations is accomplished to match zonal dimensions, and distribution component lengths are adjusted.

Since multiple templates are assembled into a single system architecture, the component sizing algorithms and the component position algorithm must be run on the entire system architecture.

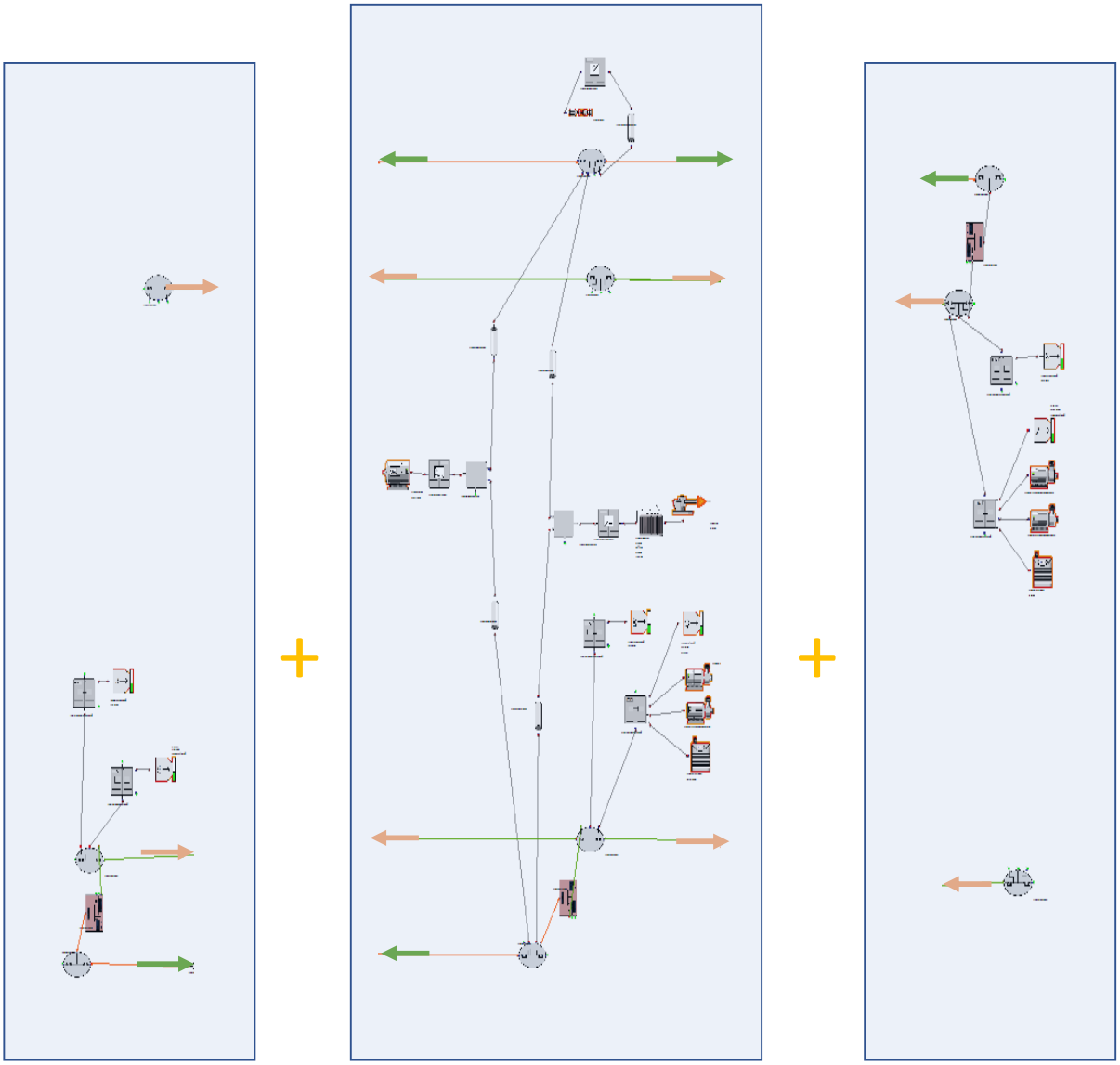

Figure 8. Assembly of multiple templates

Template Replacement

A section of a system may be removed and replaced, if that section was originally inserted using a template. For example, a study in which different combat systems are compared using the same underlying support system design can be accomplished by placing each combat system in its own template. The combat systems are changed out by undoing the placement of one template and inserting a different template. After template insertion, the component sizing and component placement algorithms must be run.

A template that has been placed in a design can be removed from that design by selecting any component that was a member of that original template and requesting removal of the entire template. If the same template was instantiated several times into a design, the process distinguishes between repeat copies of a template and removes only the designated copy. This is accomplished through the inclusion of a templateID along with a sequential instantiation number. This instantiated templateID becomes a property of each component instantiated through the application of a template, and a list of templateIDs is attached to the ship design concept, so that the instantiation number can be properly iterated when a single template is applied to a ship design repeatedly.

Subsystem Addition

Templates representing different energy domains, or portions of energy domains, can be overlaid into a ship system architecture across multiple zones. For example, a full system architecture could be constructed from one template containing an electrical generation and distribution system, one template containing a cooling water system, one template containing a mechanical drive system, and one template containing sensor and combat systems. Template nodes would be placed at the connection points between the support systems and the components requiring support.

Similarly, if selected conventional power cables were to be replaced with high-temperature superconducting (HTS) cables, then the cryogenic plant to support the HTS system could be overlaid into the conventional system using a template containing the cryogenic components.

Another example could be the addition of a system that was not previously modeled in detail. For example, if at later stages of design an auxiliary system such as the anchor and anchor windlass were to be added, the appropriate components could be added by template.

After copying the templates into the ship concept and connecting all applicable template nodes, component sizing and component positioning algorithms are required for final adjustment.

Aggregate Component Replacement

An aggregate component can be replaced with a breakdown systematic representation. The breakdown system representation is stored as a template. To replace the aggregate component, delete the component in question and insert the template, connecting the template nodes to the nodes that the aggregate components was connected to. In theory, the remainder of the system architecture should not change, since the aggregate component should have represented the full tax of the sub-components on the system, in terms of space, weight, power, cooling, etc. However, it may be wise to run the component sizing and component positioning algorithms on the resultant system as a check; perhaps further refinement of the design will cause some of the attendant properties to change.

Figure 9 depicts an example in which a single chiller component is replaced with a system template that includes separate evaporator, condenser, heat exchanger, expansion valve, and piping. Note that the connections external to the template remain the same as the connections required by the component: saltwater inlet and outlet, and chilled water inlet and outlet.

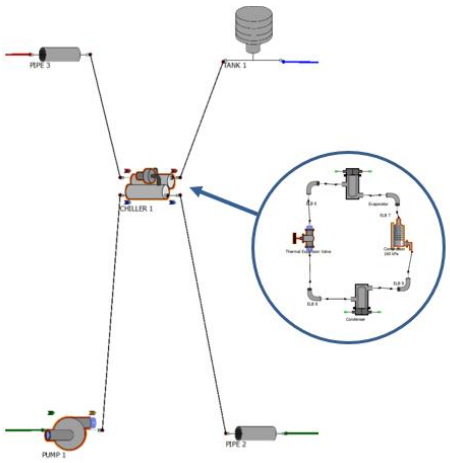

Figure 9. Replacement of an aggregate component with system template.

Testing Fully Designed Systems

In later stages of design, it may be desirable to test fully designed systems without modification of the properties of the components to ensure that the systems will operate properly as designed, or to analyze and compare several possible combinations of systems and loads operating together under various mission scenarios.

To achieve this use case, the system templates are stored either as full system designs or as a set of sub-systems and loads. They are instantiated in the ship designs using the templating process, following the use case processes described above, with one major difference: the component sizing algorithm is not used. Thus, the systems are tested as designed without modification to the size of any individual component, with the exception of the lengths of distribution components which change in response to the size of the hull. Further, the user can decide whether to employ the component placement algorithm to adjust the placement of components.

Upgrade and Modernization

Alterations, upgrades or modifications to existing ships such as a new combat system load replacing an old combat system load, with no modification to the electrical and thermal support systems other than the connections to the new load, can be tested to show that the existing systems are able to support the new load. In this case, the old load is removed, the new combat system, stored as a template, is installed using the template insertion process but excluding the component sizing algorithm. If it is desirable to modify the locations of the newly installed equipment, all previously existing equipment can be designated as fixed and the component placement algorithm can be run to adjust positions of only the newly installed components.

CONCLUSIONS AND FUTURE WORK

A process to automate the design of ship systems in early-stage ship design has been developed along with supporting algorithms and methodologies to implement the process. Each step has been described and example code has been developed to test the concepts. This functionality enables the inclusion of system design into broad, automated trade-space exploration. The integration of templating into the Navy's early-stage design process is further explored in [7].

Several areas remain for future exploration. The maximum flow algorithm has been developed for electrical and mechanical systems only, and must be expanded to open- and closed-loop piping systems. Further, a methodology for

incorporating analysis of electrical load usage needs to be overlaid on the maximum flow algorithm to preclude over-design. As mentioned previously, sizing algorithms for many components remain to be developed.

Finally, all the algorithms need to be integrated into a single, cohesive code. Presently, algorithms exist in stand-alone programs in various languages. The pertinent information can be exported from and imported to a LEAPS database, but the programs are not seamlessly integrated with one another.

REFERENCES

[1] N. Doerry, "A Vision for Ship Design and Analysis Tools", *SNAME (mt) Marine Technology*, July (2012): 8-9.

[2] R. Leonard, "Solver Theory", *Smart Ship System Design,* University of South Carolina, October 14 (2021) https://s3d.cec.sc.edu/dokuwiki/doku.php?id=documentation:solvers:power_flow_solver_theory

[3] *LEAPS Editor User's Manual, LEAPS Version 5.0*, Carderock Division – NSWC, March (2015).

[4] J. S. Chalfant, Z. Wang, M. Triantafyllou, "Expanding the Design Space Explored by S3D", *IEEE Electric Ship Technologies Symposium, IEEE ESTS 2019*, Arlington, VA, August 13-16, (2019).

[5] N. Doerry and J. Amy, "Design considerations for a reference MVDC power system," *SNAME Maritime Conv*, pp. 1–5, 2016.

[6] DaMarcus Patterson, Julie Chalfant, Michael Triantafyllou, "Automation of Component Placement in Early-Stage Ship Design", *ASNE Technology, Ships and Systems Symposium, 2020, Virtual*, January 26-28, (2021).

[7] N. Patterson and J. Chalfant, "Ship System Design Space Exploration using Templating", *submitted.*