JPL's Foundry Furnace: web-based concurrent engineering for formulation

Jonathan Murphy[(1)], Jon Blossom[(1)], Garrett Johnson[(1)], Mike Kolar[(1)], Jim Chase[(1)], Kelley Case[(1)]

[(1)]Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109, USA
Email:jonathan.murphy@jpl.nasa.gov

INTRODUCTION

The Jet Propulsion Laboratory's Innovation Foundry is an enterprise tasked with shepherding space mission concepts through the formulation lifecycle [1]. It oversees a number of "virtual teams" for the various stages of formulation. Among these is Team X, which has had considerable success over its more than 20-year history [2]. In a Team X study, domain experts (including engineers devoted to the various spacecraft subsystems) work concurrently and collaboratively over several days to arrive at a feasible point design with a reasonable cost estimate. They use a set of linked Excel workbooks, each developed and approved by a responsible "line organization" within JPL. This toolset has served Team X well over the years, and has evolved since its inception. At the same time, the Innovation Foundry's portfolio of formulation teams has expanded, and so has the scope of the design challenges they face. The A-Team runs workshop-like architecture studies to focus science investigations, generate mission concepts, assess feasibility, and explore trade spaces [3]. Team Xc performs rapid point design in the style of Team X, but for CubeSats and small spacecraft, using a different toolset [4]. Proposal teams further mature concepts to the point where they can be proposed.

Recognizing the importance of trade space modeling [5] combined with new IT services for providing and integrating data, JPL is developing the Foundry Furnace web-based software infrastructure. It will support A-Team, Team Xc, and Team X, providing study management, a catalog of hardware components, a library of re-usable analyses, and a design environment. It is a modernization of JPL's concurrent engineering infrastructure, embracing the core concepts of Model-Based Systems Engineering, and built with modern software design philosophies.

MOTIVATION

At JPL, the time is ripe for a new concurrent engineering software infrastructure. Established in 1995, the infrastructure has consisted of legacy Excel workbooks interconnected using an evolving code base. In 2008, the last significant upgrade introduced a SQL database for storing parameters, ColdFusion to support study management, and VBA plug-ins to connect the workbooks. The content of the workbooks, on the other hand, has asymmetrically evolved via the ownership and maintenance of the individual line organizations at JPL. The workbooks include hardware databases, analysis tools, and institutional cost models, each independently developed over time within the Excel environment. While this amalgamation of software and data continues to provide exceptional design support, it is becoming increasingly difficult to maintain and impractical to upgrade, particularly with the advent newer software technologies and methods.

There is also motivation beyond a code overhaul. The current system cannot easily accommodate new analyses or tools; it is limited to Excel-based design tools, and the set of parameters to be exchanged is not easily modified. In an ideal system, if a unique study calls for a unique analysis or requires a higher-fidelity analysis to address a driving concern, it could be quickly integrated to support the real-time study environment.

Moreover, since the inception of Team X, JPL has expanded its formulation team roster. Team Xc operates on a similar paradigm to Team X, but is focused on small spacecraft. It has its own set of Excel-based tools, using a different exchange service based on Phoenix ModelCenter [4]. It has successfully integrated some MATLAB-based analyses, but still generally suffers from the same inflexibility as Team X.

Earlier in the formulation process, the A-Team performs architecture exploration studies [3]. The format of an A-Team study is necessarily more free-form than a Team X study due to the more open set of questions to be answered, and studies may include science definition exercises, concept brainstorming, feasibility assessments, or wide exploration of the trade space. Because the roles and analyses needed for A-Team studies are much more diverse and unpredictable, the A-Team has not developed its own concurrent engineering infrastructure, but could significantly benefit from a sufficiently flexible design environment.

Thus, in developing the next generation infrastructure to provide more flexibility, there is a three-fold opportunity. First, the infrastructure can expand beyond Team X to provide services for A-Team and Team Xc as well. Second, a common infrastructure provides continuity between the phases of the formulation lifecycle, such that a concept moves smoothly from inception and trade space exploration in A-Team to an end-to-end point design in Team X or Team Xc. Finally, if the environment is flexible enough to support new analyses on short timescales, then analyses developed as part of one study can be reused for later concepts with relative ease.

The main driving motivators for the new infrastructure, then, are *adaptability* to solve new problems, *re-usability* of those solutions, and *continuity* through the formulation lifecycle.

CHANGES IN THE CONCURRENT ENGINEERING PARADIGM

The Foundry Furnace is a re-imagining of JPL's concurrent engineering framework. It addresses the above driving motivators though several key features:

1. Web-based architecture
2. Support for multiple analysis applications
3. An expressive central data model, supporting a consistent design ontology, drawing from the principles of Model-Based Systems Engineering
4. Query-based linking
5. Re-usable libraries for hardware, analysis, designs, and other concerns

Web-based Architecture

Currently, Team X design sessions occur in specialized facilities with preconfigured workstations and several projectors to display the broader design context, such as the system mass roll-up, system cost, a diagram of the concept of operations, etc. This configuration severely limits the ability for an outside collaborator to seamlessly embed themselves within the session from both a software and knowledge perspective. Additionally, if a design engineer is unable to attend the design session in person, their participation is also severely limited.

A web-based platform enables collaborators to remotely or locally collaborate on a design while keeping the context of the study intact using only their web browser. With an always-accessible system, a remote expert can review, edit, or add a new model or hardware into the application to serve the needs of the system from any location. Additionally, one of the promises of the Furnace application is the ability to integrate and execute analyses made with a variety of applications, including python, Excel, MATLAB, and others. This means that the execution environment must be capable of running all model types – not a reasonable expectation for the personal computer of any user wishing to participate. Therefore, the Furnace system is centered on a heavy-lifting execution engine that integrates and evaluates the system models, meaning the client application can be very light. This makes a web-based application a very real option and one that affords the end-users high ease-of-access while still leveraging the power of diverse modeling languages.

Support for Multiple Analysis Applications

Excel has proven to be a powerful spacecraft design environment for decades, but it is not always the best application for a given analysis. Team X engineers already use a variety of external tools to complete their work, ranging from MATLAB to PowerPoint. There has been some success in connecting tools through Excel, but for the most part integration of external tools is achieved by manually copying data to and from the Excel workbooks. This places an extra burden on the engineers; but more egregiously, these analyses are not updated automatically as the design changes, resulting in a greater chance of a non-converged design. Furthermore, technical traceability is lost in this process, resulting in great difficulty to track down the sources of problems.

The Foundry Furnace aims to support the most frequently-used applications directly in the web-based design environment. The design application (called the "Integrated Modeling Environment", or IME) will natively support Excel-like models and python models, which will allow simple and rapid editing and testing. This ensures that when an engineer needs to make a last-minute change, or create a new model from scratch during a short fast-paced session, this can be done quickly without leaving the application.

For tools that are implemented in other languages, the Foundry Furnace will support uploading and integration in a "black-box" format, meaning the model must be edited on the user's local machine rather than in the web environment, but that it will be stored and version-controlled on the server, and executed on the server as required. This ensures repeatability and traceability, and means that the analysis will stay in synchronization with the design.

Not every model can be run on a server, though. Many of the models run by engineers both in and out of the Foundry involve custom and specialized user interfaces to accommodate the function of the model. These require user input to drive and update the results, meaning they can't be run and integrated into a systems model on a remote server, short of re-implementing the UI in a new framework. Models like this have been built to and will continue to have to run on the user's machine, so IME will enable plugins that allow for data to be pushed to and received from analyses such as these. While user interaction will inhibit the completion of these types of analyses, the data will still propagate through and be stored within the system model, along with providing notice to users on the propagation of design analyses.

Model-Based Systems Engineering (MBSE)

JPL's existing concurrent engineering environment relies on pre-configured analyses, with predefined parameter links between domains. Within a domain (for example, a spacecraft subsystem), there may be any number of analyses and tools, all implemented as a single monolithic Excel workbook. This makes it impractical to integrate new analyses on a short time-scale, since any new analysis will need to be manually linked to the other analyses. To allow the system to be *adaptable* to a new analysis and to support new concerns and designs, modeling tools need to move away from a rigidly "hardwired" web of analysis. This will require *automatic linking*, meaning the design will need to be comprehensible by automated algorithms and thus the design description will need to be more rigorously structured. Specifically, an *ontology* will be needed as a consistent means of describing the design.

To this end, the Foundry Furnace is drawing from the formalisms of Model Based Systems Engineering (MBSE) and will use a single centralized, organized data model that is built using an internally consistent ontology. Most current MBSE efforts use the Systems Modeling Language (SysML), but early pilots indicated that SysML would be too "heavyweight" for concurrent engineering in early mission formulation. Instead, Foundry Furnace uses a data model that draws some vocabulary and concepts from SysML, but is substantially simpler, and has some additional concepts to address the concerns of early mission formulation and concurrent use. A description of the software service that provides access to the data model can be found in the "Software Architecture" section of this paper. Some of the basic MBSE concepts in use are described here, along with their closest SysML equivalents:

> **Structured Composition:** Foundry Furnace uses a hierarchical composition structure. Data "Blocks" live in a single hierarchical tree, where each Block can have any number of child Blocks, and every Block has exactly one parent. This is substantially simpler than the many kinds of composition which are possible in SysML. A Furnace Block is essentially equivalent to a SysML Block, and the closest SysML equivalent to the Furnace composition structure is an instanced Part Property (slot). Each Block in Foundry Furnace can additionally have any number of Parameters, which are equivalent to SysML's Value Properties.
>
> **Type Inheritance**: Foundry Furnace uses an extremely simplified form of inheritance for Blocks, using a single library of re-usable Block "Interfaces" that define a Block's Parameters.
>
> **Analysis:** An "Analysis" in Foundry Furnace is represented as a specialized Block. It lives in the same composition tree as the Blocks that describe the design, but can be separated from the design. Thus, as a concept progresses through the design process, the analyses that were used in one study can be replaced or updated as needed without re-defining the design itself. The Foundry Furnace "Analysis" is a close conceptual equivalent to the SysML "constraint".

In addition, there are a couple of concepts in Foundry Furnace that have less obvious SysML equivalents, though the same concerns can (and have) been addressed by others using SysML:

> **Options**: Foundry Furnace allows multiple options to be defined as "deltas" from a common baseline. In addition, options may be defined at lower levels of composition. For example, the propulsion chair may wish to define a few options for the propulsion subsystem that can then be re-used in several different system-level options.
>
> **State and Time:** Foundry Furnace allows users to specify that certain Parameters are "State Variables" which can vary with time. Users can further define re-usable "States" that describe the operating modes for some subset of the design. This aspect is functionally equivalent to a SysML State Machine. Lastly, the behavior of the design can be modeled over one or more mission "Timelines".

With the move to an MBSE-based approach come the challenges of defining a consistent and useful ontology. In the previous paradigm, where every link was made manually, engineers could be relied upon to understand the intent of another domain's output parameter and act upon it appropriately. If links are to be automated, those output parameters must conform to an agreed-upon ontology. Since there is no "one true" way to model anything, and different people and different fields may use different ways of saying the same or similar things, the common ontology will need to be negotiated as a satisfactory "treaty" among the domains.

Query-based Linking

The monolithic Excel workbooks that have evolved over two decades contain analyses, catalog data, design data, diagrams, and documentation that taken as a whole define an implicit subsystem model capable of describing a wide variety of architectures. Outputs of each subsystem model are defined and published to a well-understood centralized exchange system, where they can be accessed by the other subsystem models. The combined system model can define and analyze a wide variety of spacecraft and mission designs – including every mission study ever performed and then some – and no further model creation or integration work is generally required during study sessions.

The level of flexibility and coverage of the Team X system model is a significant accomplishment representing years of experience, but we have already recognized and reached its limitations. The development of the Furnace is tasked with expanding the capabilities of the Foundry teams to accommodate new system and subsystem architectures, new technologies, and new mission types. The tools must facilitate the re-use and re-organization of existing architectural elements, and the addition of new ones. It must allow engineers to integrate their models in a time-constrained concurrent design session such that they can focus study time less on the modeling process and more on the design and analysis of the system.

In conventional block modeling, the modeler generally constructs the data blocks and analyses using commonly available modeling tools, and then adds the links that define the flow of data from one entity to another dependent entity. This linking and integration process is generally manual and time-consuming, and requires engineers to understand the needs of the target block as well as all of the data sources within the existing system. Engineers must also keep those needs in mind as components are added or removed, as those links may require updates when the model changes – a task that is too tedious and slow for modifications made during a typical three hour Team X design session.

The Furnace follows a different approach. In the Integrated Modeling Environment, users compose a complete system largely by connecting pre-defined pieces, which may represent complete subsystem designs, sub-assemblies, or individual hardware components from the hardware and template libraries. This mix-and-match approach maximizes the engineer's ability to respond and adapt to evolving and unanticipated requirements, but it also shifts the burden of integrating the pieces back into the study session time.

To simplify the integration process, the Furnace incorporates a system for encoding the data needs of each analysis with the analysis itself, giving the system the information it needs to make intelligent linking decisions on behalf of the user. When analysis code is saved into the re-usable model library, the creator can attach linking rules to the analysis inputs. These rules are search queries for information in the rest of the system model. When executed in the context of the system of interest, they result in a set of data sources that should be linked to the instantiated analysis. This approach is called Query-Based Linking. It allows the system to suggest links when components are added or removed, relieving the majority of the model integration burden. Put simply, the analysis can be written such that it knows what information it needs in order function.

As a simple example, imagine an analysis that calculates the power needs of the spacecraft. Its input parameters are the power needs of each individual component in the system. In a traditional modeling application, the addition of new hardware requires someone to be aware of the existence of this power rollup, be aware of its input needs, and create a link from a newly-added component to the power rollup analysis. If the analysis is added later, the person adding the link must comb through the model to find any elements that should be connected to the analysis. If any link is neglected, the results of the analysis will not represent the actual power needs of the system as described.

Using Query-Based Linking, a Furnace user can instead attach a rule to this notional power rollup analysis input saying "find all components that require power." When a new component is added to the system, the rule is re-evaluated; if the new component requires power, a link from the new component to the power rollup will be added automatically. Similarly, if the rollup analysis is added to a system model already containing components, links will automatically be created from those components to the newly-added analysis. Linking becomes part of the analysis authoring process rather than the model integration process and therefore falls outside of the concurrent design session except when a human is required to validate or disambiguate situations where the link rules may be incomplete. And in the cases where a link rule is non-existent or incorrect, this approach can cleanly coexist with more conventional links.

Query-Based Linking requires new infrastructure included in the Foundry Furnace. The Model Registry must include a facility for defining and attaching link rules along with model parameters. The Integrated Modeling Environment must provide a system for interpreting those rules in context and generating the links, as well as a user interface for indicating, inspecting, and modifying generated links. Most importantly, Query-Based Linking relies on a standardized modeling ontology, and is thus enabled by the use of a Model-Based Systems Engineering paradigm.

Re-usable Libraries

The current Team X Excel workbooks contain everything for them to independently function, including databases of hardware, hardcoded unit conversions, and the analysis models that ultimately drive design decisions. The modeling expertise of the Foundry, however, has continued to expand with A-Team and Team Xc, teams, developing models to explore earlier mission concepts and small-satellites respectively. However, the self-contained nature of the content in the Team X workbooks has made it difficult to share work across the teams. To solve this, Furnace relies on a series of re-usable libraries that can be shared across the various teams making up the Foundry, including storage for unit conversion rules, hardware descriptions, analysis models, design templates, and a systems engineering ontology.

The libraries are more than just independent stores of data – they are designed to work in concert and reference each other to facilitate a higher level of integration, traceability, and data structure. The hardware database leverages the unit database and ontology, for example, to annotate the hardware and parameters with roles and behaviors. The result is that data between these applications can be exchanged and integrated freely. With this kind of connectivity, a model in the library may reference several different types of hardware while being able to rely on "mass" meaning the same thing among all the components and rely on automatic unit conversion between the parameters. By using the same or similar ontology and unit set between teams in the Foundry, hardware and models can be shared over the life of a concept.

The design template library builds on every other library in the system and represents a key component of how Team X operates. The current Excel workbooks function on a fixed model structure, in which analyses are either turned on or off to enable quick development and design of a functioning system – design templates serve the same purpose in that they allow for a fast integration of more diverse partial or full system designs of models, parameters, and hardware.

The interconnectivity of the libraries keeps the data structured, searchable, and reusable in a variety of different design scenarios, and ultimately can enable more rapid connections between system components during design by structuring the data up front.

SOFTWARE ARCHITECTURE DESCRIPTION

The Furnace is based on Systems-of-Systems and Service-Oriented Architecture styles and concepts and is composed of stand-alone and integrated functional area services that satisfy Foundry requirements. This open architectural approach simplifies capabilities development to satisfy new business needs, enables cost reduction through extensive use of design patterns and IT solutions, and provides opportunities to replace custom implementations with COTS and Open Source solutions in the future. In addition, the Furnace data-interchange standards provide data objects and transformation that allow integration with MBSE efforts at JPL as well as exporting system descriptions for import by funded flight projects.

The Foundry Furnace system consists of the following major functional areas:

Study Management

The Study Management system coordinates the business behind conducting a study in the JPL Innovation Foundry. It includes features to help study administrators maintain a roster of available team members and their areas of expertise, track clients and client needs, schedule meetings and study sessions, send invitations and updates to interested parties, report business metrics, manage access and security to study reports and artifacts, and streamline similar activities involved in management of the study process itself.

Catalogs and Registries

The Furnace includes a number of specific Catalogs and Registries where data is curated and maintained. Each stand-alone database application supports separate applications under a common security layer but is also accessible for general use. These databases include: (i) the Hardware Catalog that holds descriptions of various pieces of hardware approved for use in spacecraft designs, (ii) the Model Registry that stores analysis models – including both code and meta-information – for use in evaluating spacecraft designs, (iii) the Constants Dictionary that provides a central source for reusable constant data such as planetary masses, and (iv) other domain-specific databases will be constructed as the need arises. Additionally, the Common Resources database provides the foundations for data exchange across the system – including standard units, unit conversions, and definitions for system-wide data ontologies.

Integrated Modeling Environment

During a concurrent design session, Furnace users will spend the vast majority of their time in the Integrated Modeling Environment (IME) application. Using IME, engineers collaborate on a shared system model, using analyses, and data accessed through the Catalogs and Registries services. Participants in the study can refine and contribute their

subsystem designs, analyse and preview their effects on the integrated system model, create design options to explore within the shared system, visualize data from their analyses, and create reports to share with customers.

Execution Service

The Execution Service supports the IME model analysis process. Its primary roles are to generate an appropriate workflow for the integrated system model under study, to propagate linked data between analysis blocks during the analysis process, and to coordinate the execution of individual analyses running in a variety of external tools and languages on a variety of platforms.

Centralized Data Model Service

Foundry Furnace uses a centralized data model, as described in the "Model-Based Systems Engineering" section. The software element that provides that model as a service is called "Crucible". In a design study, Crucible will instantiate a new model tree, which can be configured and populated using re-usable design templates. That model tree will include a full description of the design, the analyses, timeline descriptions, and any options under consideration. During an active session, the Integrated Modeling Environment will edit the Crucible model, and Crucible will call on the Execution Service to evaluate analyses.

Since Crucible is a standalone service, separate from the Integrated Modeling Environment, it can be accessed by other applications as well. If an analysis is unsuitable for integration in IME (for example, because it contains a Graphical User Interface as part of the workflow), it can participate in the design process through direct use of the Crucible API.

Security

Studies conducted by the Foundry often have specific security considerations, as does the data under management, and the Security layer ensures the safety of information within the Foundry Furnace. All of the Furnace applications exist under, and abide by the rules of, a common Security layer. This includes a single sign-on service for access to all Furnace applications as well as role-based control of access to data and study information.

Cost Analytics

The Cost Analytics system is a repository of historical cost information for Foundry studies as well as JPL and NASA mission system costs. This repository of integrated data enables the analysis of a design cost based on previous designs with similar technical features, as well as supporting time-series analysis of systems lifecycle cost estimates.

SUPPORT FOR LEGACY TOOLS

Team X and Team Xc are active teams with frequent studies and include Excel-based tools containing large numbers of design tools, domain analyses, and data sources. It would be unreasonable to attempt an overnight re-implementation in the new infrastructure. Instead, the new infrastructure will remain backwards-compatible with the existing Excel workbooks. Over the course of multiple years, the Foundry will work with Team X and Team Xc chairs to transition tooling, beginning with a few "early adopters" who are interested in using the new capabilities.

During the transition, the existing Excel workbooks will still be able to push to and pull from the current parameter-based system. However, in the background, a parallel design will be represented in the new Crucible data model service with a mapping maintained between the new and old infrastructures. The task of creating this mapping, from the old list of several thousand parameters to the new expressive data model, is currently underway, and serves a dual purpose of verifying that the new data model can in fact support the legacy design capabilities.

One of the objectives of the new system is to be more *flexible* than the old. By maintaining backward-compatibility with the old data structure, it may not be possible to fully realize this flexibility. Operating in "legacy mode" will impose constraints on what can be represented; but as domains are transitioned, it should be possible to gradually remove these constraints, until finally the full potential can be realized.

CONCLUSION

Foundry Furnace is a re-imagining of JPL's concurrent engineering infrastructure. It aims to be more *adaptable*, to support new design challenges with new architectures and new analyses. It aims to make designs and analyses more *re-usable*, to more easily address existing challenges in new contexts. And it aims to provide better *continuity* of design infrastructure across the formulation lifecycle.

It employs a modern, service-oriented, web-based software architecture with services that provide study management, re-usable libraries of analyses and hardware, and a concurrent design environment. To enable re-configurability of design and analysis, Furnace turns to the Model-Based Systems Engineering paradigm, and distils it to a simplified subset of the standard MBSE vocabulary that is adequate for mission formulation. This more expressive data model, combined with a consistent ontology, will enable adaptability by moving away from the current web of rigidly linked analyses towards a re-configurable set of analysis components that encapsulate rules for finding their inputs.

Parts of the Foundry Furnace are already operational with significant functionality being delivered in 2017. But the transition to the new infrastructure will still require extensive efforts beyond the software development. The Foundry will need to build a shared ontology that is acceptable to the various line organizations who will be the ultimate users of the system. And because of the extensive tooling already implemented in the current infrastructure, the transition of domain analysis and design capabilities will occur over an extended time frame with backward-compatibility maintained for the duration.

As the Foundry Furnace is adopted throughout the formulation lifecycle, the process of taking a mission concept from inception to an end-to-end mission concept should become easier and more streamlined, with less effort spent re-describing what is already understood, and more effort spent solving engineering problems.

ACKNOWLEDGEMENTS

The authors would like to thank the JPL Innovation Foundry, including Tony Freeman, Greg Garner, and Johnny Kwok, for their vision and support. This endeavor would not be possible without the significant contributions from the larger Foundry Modernization team and with particular recognition to several early concept and software contributors, including Jeremy Arca, Carlos Balacuit, Bjorn Cole, Greg Dubos, and John Ziemer. Finally, this work is has been conducted in collaboration with the Project Systems and Formulation Section (312) and the Integrated Model-Centric Engineering Office (IMCE).

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

REFERENCES

[1] B. Sherwood, D. McCleese, “JPL Innovation Foundry,” *Acta Astronautica*, vol. 89, pp. 236-247, 2013.
[2] R. Wheeler, M. Adler, and B. Sherwood, “The New Team X,” *58th International Astronautical Congress*, 2007.
[3] J. Ziemer, J. Ervin, and J. Lang, "Exploring Mission Concepts with the JPL Innovation Foundry A-Team”, *AIAA Space Conferences Proceedings*, San Diego, California, September 10-12, 2013
[4] P. Zarifian, et al., "Team Xc: JPL’s Collaborative Design Team for Exploring CubeSat, NanoSat, and SmallSat-based Mission Concepts," *IEEE Aerospace Conference*, 2015
[5] M. Jones, J. Chase, “Conceptual Design Methods and the Application of a Tradespace Modeling Tool for Deep Space Missions,” *IEEE Aerospace Conference*, 2008