Applicable workshop objective: **T-3**

Developing an Open Platform for Democratised MBSE

Andrey Vasilyev[1], Gianmaria Bullegas[1], Omar Nachawati[1],
Maged Elaasar[2], Steven Jenkins[2]
[1]Perpetual Labs Ltd., UK, {gian,omar,andrey}@perpetuallabs.io
[2]NASA Jet Propulsion Laboratory, USA, maged.e.elaasar@jpl.nasa.gov

Introduction

Integration of advanced system simulation and analysis with Model Based Systems Engineering (MBSE) models is difficult to achieve, particularly during the early stages of system lifecycle, because models from Domain-Specific Tolls (DSTs) often lack common notation [1]. Collecting, aggregating and exchanging information at the system level is complex and often error-prone, which hampers its use in a multi-disciplinary concurrent design setting [2]. This limits the ability to analyse system-level requirements (such as performance and dependability) in the early design phases, causing the postponement of design decisions to later phases. This, in turn, reduces possible opportunities to study alternative solutions and validation of fitness for purpose. It also increases the costs (in terms of time and skill) of design refinements if irreversible consecutive design decisions are made in the early stages of the development [3]. The limitations of current MBSE frameworks and their poor integration with system simulation environments, contribute to an increased barrier-to-entry for the widespread adoption of Model-Based Design (MBD).

In response to these challenges, Perpetual Labs in collaboration with the OpenCAESAR[1] project and the Open Source Modelica Consortium (OSMC)[2] are developing the GitWorks platform which aims to establish a rigorous MBSE framework while democratising access to MBD for industry, academia and independent developers. One of the core features of the platform is tight integration of systems simulation capabilities based on the Modelica language with MBSE activities. This is achieved by coupling a web-based collaborative Modelica editor and simulator with a Web application for integrated data and knowledge management. Additionally, GitWorks Commons provides a searchable repository of models, tools and services with a try-before-you-buy business model. GitWorks has been designed as a turn-key solution for model management and integration provided with the convenience of a Software as a Service product.

GitWorks Platform

The GitWorks platform aims to streamline many systems engineering practices by integrating three major concepts and their inter-related functionalities: DEMOps, Semantic Twin and Digital Prototype.

DevOps for Digital Engineering and Manufacturing (DEMOps)

DevOps is a set of software engineering practices designed to establish a collaborative, multidisciplinary and continuous delivery of software products while ensuring its quality and reliability. We borrow this concept and adapt it for Concurrent Engineering Design processes by establishing a Git-based configuration management (CM) workflow. The following are the key principles of such a DEMOps approach:

- **Traceability** of information allows the users to observe a full history of changes made across all inter-related models and documents. As a result, all design decisions and constraints can be traced back to their respective authorities and rationales.
- **Repeatability** ensures that the analysis of the system design including its dependencies can be encapsulated such that it can be repeatable. This property establishes the confidence in the design process over time.

[1] http://www.opencaesar.io
[2] https://openmodelica.org/home/consortium

- **Durability** is the ability to version-control the design and analysis information of the system, thereby enabling rigorous audits.
- **Efficiency** is the ability to automate processes that are otherwise manual by employing continuous integration and continuous delivery (CI/CD) practices. This property aims to minimise any costs and errors associated with manual operations.

Semantic Twin

In order to capture all facets of system design and corresponding systems engineering activities a precise language rooted in mathematics and formal logic is required. Through such a language, system descriptions are expressed using a set of common vocabularies consisting of concepts, their relationships and corresponding properties. As a result, the systems engineering knowledge stored within various information artefacts produced during different stages of product life cycle can be unified and stored in a machine-interpretable knowledge graph called Semantic Twin. This technology is capable of significantly improving typically human-led activities (such as reporting, verification and validation of system information and others) through powerful automations such as logical reasoning, machine learning, data mining, etc.

The GitWorks platform uses the Ontology Modelling Language (OML) as the core ontological language in order to create such a Semantic Twin. OML was developed as a part of the CAESAR project at Jet Propulsion Laboratory (JPL) [4]. OML provides an expandable set of vocabularies with well-defined semantics that can be used as a foundation to model different types of engineering artefacts in a semantically consistent and interoperable fashion [5]. OML extends OWL 2 DL (Web Ontology Language 2 - Description Logic) in such a way that retains the benefits of OWL 2 DL while addressing some of its limitations [4]. The resulting approach is similar to what OSMoSE initiative [6] is striving to achieve, but relies on a different technology stack utilising many existing semantic web technologies and tools.

Perpetual Labs is actively working on multiple adapters for automatically enriching the Semantic Twin from the source information artefacts. Such adapters are based on a number of foundational vocabularies that represent a semantic mapping between a given model formalism and OML. Employed in conjunction with a CI/CD pipeline, such adapters are capable of automatically reading the artefact files from a Git repository, extracting their representation in OML and running various automated scripts (e.g. to detect any abnormalities within the original artefacts). Currently, the adapters for SysML v1/v2, Modelica models, Word and Excel documents are in development with plans to provide further integrations with other tools such as DOORS NG, Cameo, etc.

The Semantic Twin forms a federated knowledge graph that is stored as Resource Description Framework (RDF) triples in a custom versioned triple store developed by Perpetual Labs based on Jena TDB2 [7].

Digital Prototype

In the context of DEMOps, ’Digital Prototype’ refers to the usage of digital environments to facilitate the co-simulation of engineering models, connection with HardWare-In-the-Loop (HWIL) frameworks and real-time system data to support Virtual system Integration, Validation and Verification from the early stages of the product life cycle. In GitWorks, these functionalities are fulfilled by the Modelica Studio environment. In its current version, Modelica Studio (see Fig. 1) only supports editing and simulation of Modelica-based models.

Modelica is a physics-based acausal and object-oriented modelling language with a growing community that produces and manages a constantly-increasing code-base of models [8]. Modelica models’ code is written in plain text, and dependencies among them are tracked via ‘uses()’ annotation. Such model representation lends itself to be easily translated into an OML representation and a corresponding Semantic Twin, thereby enabling advanced model-management capabilities such as semantic search, analysis, cross-referencing, checking and automated model configuration.

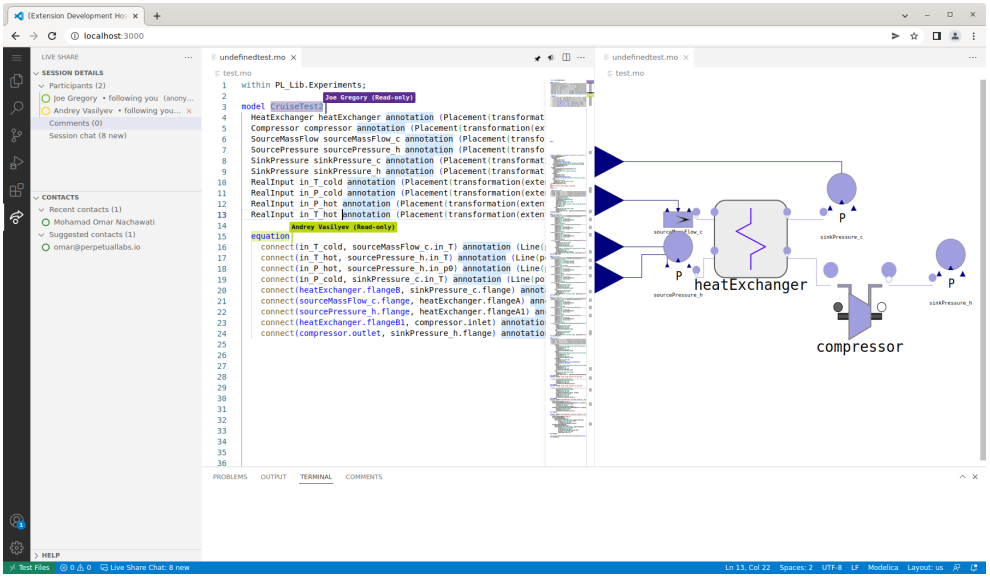

Figure 1: Screenshot of Modelica Studio collaborative editor

Digital Prototype and Semantic Twin are, therefore, closely coupled and together they enable several novel workflows from the earliest stages of MBSE activities. For example, the development of system models can be greatly aided by generating partial Modelica models from the architectural information embedded in SysML diagrams. Defining the model's physics-based behaviour can be augmented by searching the GitWorks Commons for available component models. Furthermore, the model simulation results can be used to populate the corresponding nodes within the Semantic Twin with requirements verification information.

Additionally, the Digital Prototype must be supported by a scalable (cloud based) computational infrastructure to enable the more computationally-demanding workflows, and must also be integrated with the CI/CD pipeline to promote automation. In GitWorks, this is achieved via the integration of a standard CI/CD pipeline with a scalable HPC environment.

Conceptual Platform Architecture

From the perspective of its users, GitWorks provides multiple first-party Web application environments for interacting with the platform: Commons, Projects, and Community Environments. In addition to these environments, through Git and REST-based APIs, GitWorks is designed to integrate with third-party authoring and reporting applications as well.

GitWorks Commons

The GitWorks Commons (see Fig. 2) enables the seamless publishing and reuse of design and engineering artefacts across tool, domain and organisation boundaries. Within the GitWorks platform, an artefact is considered as the basic unit of reuse. The coarsity of this unit depends on the tool and domain vocabulary.

To support federated collaboration for enhanced data security and intellectual property protection, we are also actively working on the integration of co-simulation engines, specifically Maestro2 by the INTO-CPS project [8].

GitWorks Projects

The Workbench environment is a Semantic Twin-powered Web application for integrated data and knowledge management, and exploration and visualisation of the digital thread across multiple disciplines, organisations and product lifecycle stages. It enables the exploration, querying, and modification of OML-based knowledge base using a Web-based GUI and provides customizable OML vocabularies for different cyber-physical system lifecycle activities, such as requirements analysis, system modelling, verification and maintenance.

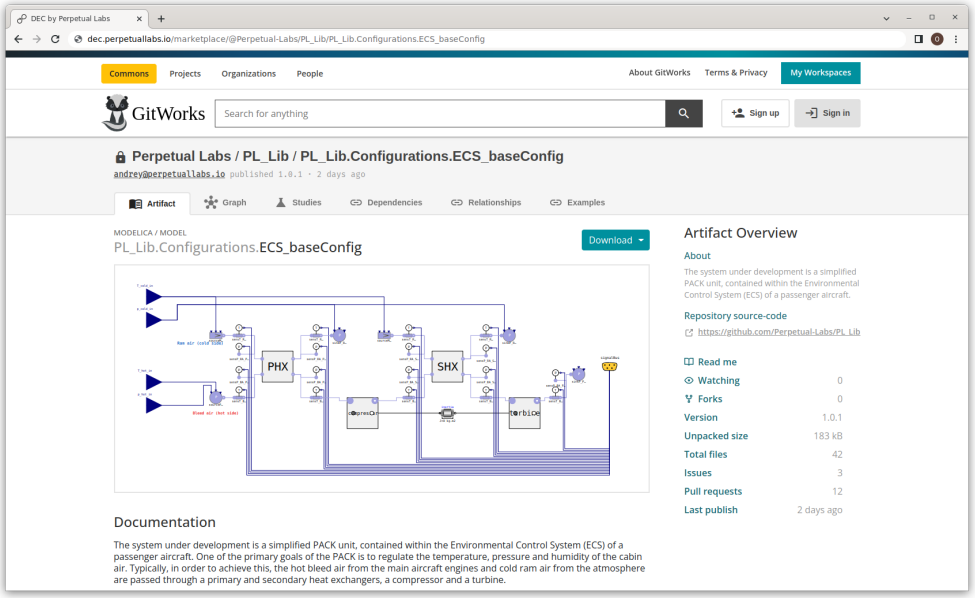

Figure 2: Screenshot of the GitWorks Commons UI for Modelica and other digital engineering artefacts

The GitWorks platform leverages GitLab to provide DevOps capabilities for artefact repositories, to include Git-based version control, issue management, and CI/CD. To enable Semantic Twin-powered authoring and reporting, every artefact repository on GitWorks is backed by both a Git repository and a corresponding RDF triple store. The RDF triple store serves as a cache to accelerate semantic queries against the repository and can be constructed on demand directly from the files in the Git repository.

GitWorks Community

Finally, the Community environment enables users and organisations on the platform to connect with one another in a kind of social network for Digital Engineering. Each user is provided with a profile page that contains a public bio with an activity stream and links to associated artefact repositories, project workspaces, and organisations.

Case-study

PL is working on a case study in collaboration with the CAESAR project JPL to demonstrate the capabilities of the platform. The case study is focused on the design of a fictitious mission for deep space exploration named Kepler-16. A demonstration of the platform and the case-study is planned for the workshop.

Conclusions & Future Work

This paper presented our vision for GitWorks: the first DevOPS platform for Digital Engineering and Manufacturing. The core concepts of the platform (DEMOps, Semantic Twin and Digital Prototype) are integrated together to enable and promote rigorous MBSE practices while lowering the barrier of entry for anyone.

Plans for future work include further development of OML vocabularies and OML adaptors to increase the number of different modelling paradigms and commercial tools supported by the GitWorks platform. Alongside that, we continue to develop the user interface of the web applications to ensure a streamlined experience. Additionally, we plan to demonstrate the application to various use cases including satellite systems and composite structures design and fabrication.

Acknowledgements

The work presented here is supported by the Engineering and Physical Sciences Research Council (UK) under the InnovateUK project 97404.

References

[1] A.M. Madni, M. Sievers, Model-based systems engineering: Motivation, current status, and research opportunities, Syst. Eng. Electr. 21 (2018) 172–190.

[2] McDermott, Hutchison, Salado, Henderson, Benchmarking the benefits and current maturity of model-based systems engineering across the enterprise: Results of the MBSE maturity survey, Research Center (SERC) …. (2020).

[3] B.W. Stirgwolt, T.A. Mazzuchi, S. Sarkani, A model-based systems engineering approach for developing modular system architectures, J. Eng. Des. 33 (2022) 95–119.

[4] D. Wagner, S.Y. Kim-Castet, A. Jimenez, M. Elaasar, N. Rouquette, S. Jenkins, CAESAR Model-Based Approach to Harness Design, in: 2020 IEEE Aerospace Conference, ieeexplore.ieee.org, 2020: pp. 1–13.

[5] T. Bayer, J. Day, E. Dodd, L. Jones-Wilson, A. Rivera, N. Shougarian, S. Susca, D. Wagner, Europa Clipper: MBSE Proving Ground, in: 2021 IEEE Aerospace Conference (50100), ieeexplore.ieee.org, 2021: pp. 1–19.

[6] Terraillon, Digital transformation in the European Space Industry, ERTS2022. (2022). https://hal.archives-ouvertes.fr/hal-03694417/.

[7] J.J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, K. Wilkinson, Jena: implementing the semantic web recommendations, in: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters, Association for Computing Machinery, New York, NY, USA, 2004: pp. 74–83.

[8] P. Fritzson, V. Engelson, Modelica — A unified object-oriented language for system modeling and simulation, ECOOP'98 — Object-Oriented Programming. (1998) 67–90. https://doi.org/10.1007/bfb0054087.