DIGITAL INDUSTRIES SOFTWARE

Leveraging a unified digital engineering ecosystem

Integrate the digital thread and digital twin to improve performance, cut costs and boost customer satisfaction

Executive summary

The digital twin concept is valuable, but its full power emerges when it becomes accessible beyond the team that created it and can be used by customers in new and productive ways. Openness and interoperability are essential, requiring standardized data formats, common communication protocols, consistent identification schemes and a modular architecture with well-documented open interfaces. The Siemens Xcelerator business platform of software, hardware and services provides a well-formed, interconnected digital engineering ecosystem, with Teamcenter® software, which is used to orchestrate change, configuration, collaboration and storage, at its core. Using Siemens Xcelerator to manage the relationships between models with a comprehensive, up-to-date and integrated digital twin enables improved performance, reduced maintenance costs and enhanced customer satisfaction.

Contents

Introduction

This white paper covers a set of related questions about data interoperability and standards, model basis for the digital twin, distributed application programming interface (API) access to various facts of the digital thread, and the introduction of SysMLv2.

The concept of a digital twin is valuable within the single lifecycle of a product, and useful among the team of individuals that created the digital twin. The team members typically have intimate knowledge of their part of the work or part of the product, making the digital twin one of the go-to data sets. The value of the digital twin can be increased dramatically when it becomes useful outside of the life of a single product and outside of the team that created it. Its utility grows exponentially when it is accessible to the customers that can use it to interact with the product in new and innovative ways.

There is a corollary for digital threads; the team members that created it to satisfy immediate needs are likely to have built them for a unique and timely purpose. Digital threads become dramatically more valuable when they can be used across multiple products and teams. Thankfully, new technologies and standards have been developed to assist in the formation, access, organization and distribution of digital threads and twins.

Data interoperability and standards

Siemens Digital Industries Software considers openness critical to the digital enterprise as all phases of the product and production lifecycle are connected by a comprehensive digital data model regardless of where the data originated. The first requirement of interoperability is openness: without access to data (regardless of format), there is no ability to interact.

Customers need the support of software vendors and service providers to optimize the reuse of digital data, facilitating collaboration across ideation, realization and use processes. Incompatible data formats and versioning problems that inhibit data reuse and collaboration are counterproductive and need to be eliminated. Likewise, data formats that are not machine readable, in a proprietary binary format, poorly documented, or inflexible, must also be eliminated.

The value we place on openness is reflected in our open software architectures, published data formats, user communities and steering initiatives. We have learned that successful and responsible vendors compete on product value while adopting open-business models that recognize the large investment that customers have made in existing information technology (IT) infrastructures.

Siemens is a key participant in the international Codex of PLM Openness (CPO), an initiative for agreeing on a mutual understanding of the importance of openness among software customers, vendors and service providers. The CPO aligns us with the evolving needs of customers and service providers, providing an effective framework and reference that supports our ongoing commitment to openness.

Siemens is working with standards development organizations (SDO) like the Society of Automotive Engineers (SAE) to digitalize the industry standards, making standard element parts of the digital twin. Siemens also works with other SDOs, including the Object Management Group (OMG), the International Organization for Standardization (ISO) and the American Society for Testing and Materials (ASTM), with the idea that product lifecycle management (PLM) will not just comply with standards, it will embody them.

With openness in mind, we turn our attention to what is exchanged across open interfaces. To achieve interoperability, the data that is exchanged must be formatted using open standards to the maximum extent possible. Siemens has a broad network of partners and collaborators, but this does not account for unknown or future collaborators. This is where we rely on our work with standards bodies to identify the data formats to exchange through open interfaces.

The free exchange of easily accessible information in nonproprietary formats offers its own set of challenges. A robust security framework must be in place to ensure the data remains confidential and only accessible to authorized parties. Additional safeguards must be in place to ensure the integrity of the data when it is used, and if possible, when it is exchanged out of the system of record. Confidentiality and integrity must be balanced with accessibility, protecting confidentiality and integrity. This should not come at the expense of complicating accessibility (for example, with proprietary protocols or interfaces).

Openness and interoperability are essential for the digital twin to interact with another digital twin and systems. A list of requirements include:

- Standardized, nonproprietary data formats must be used to ensure that data can be easily shared
- Standards based, common communication protocols must be used to enable communication and interaction with the digital twin/digital thread
- Consistent identification schemes must be provided to ensure that data is easily identifiable and can be matched across systems
- A modular architecture with well documented open interfaces must be used to enable integration with another digital twin
- The command-and-control aspects of the digital thread must be well documented to enable composing from multiple sources
- The digital twin must be managed in a system designed with security and privacy in mind to ensure that data is protected when shared with other systems and the digital twin
- The digital thread management system must be designed with security in mind to limit the potential for misuse
- The digital twin management system benefits from the ability to manage variants, allowing the twin to be reused to the maximum extent possible
- The digital twin management system benefits from the ability to manage unknown file and data formats, allowing growth beyond the designed boundaries of the system

Model basis for the digital twin

At its simplest level, a digital twin is a model with enough fidelity to support a decision. A digital twin created early in the lifecycle of a product is likely to be high level but matures over time to support decisions from a variety of perspectives. It is important to have a common configuration baseline for those decisions from different perspectives. Those different perspectives may require various kinds of models that leverage common data elements and parameters related to them, so there must be interoperability between the models. This is where most organizations are in their digital transformation journey – a conglomeration of specific, purpose-built models that are roughly related (with links, by proximity in file folders or by human glue). Models are specified to support specific business decisions.

To evolve from this position and gain additional value, companies benefit from process commonality and documented governance. When the business decisions can be clearly identified, processes can be developed to carry them out. The basis of model selection is well defined processes that are supported by a culture with an expectation to meet governance demands. From there, tools and automation can be built to improve the utility of the models.

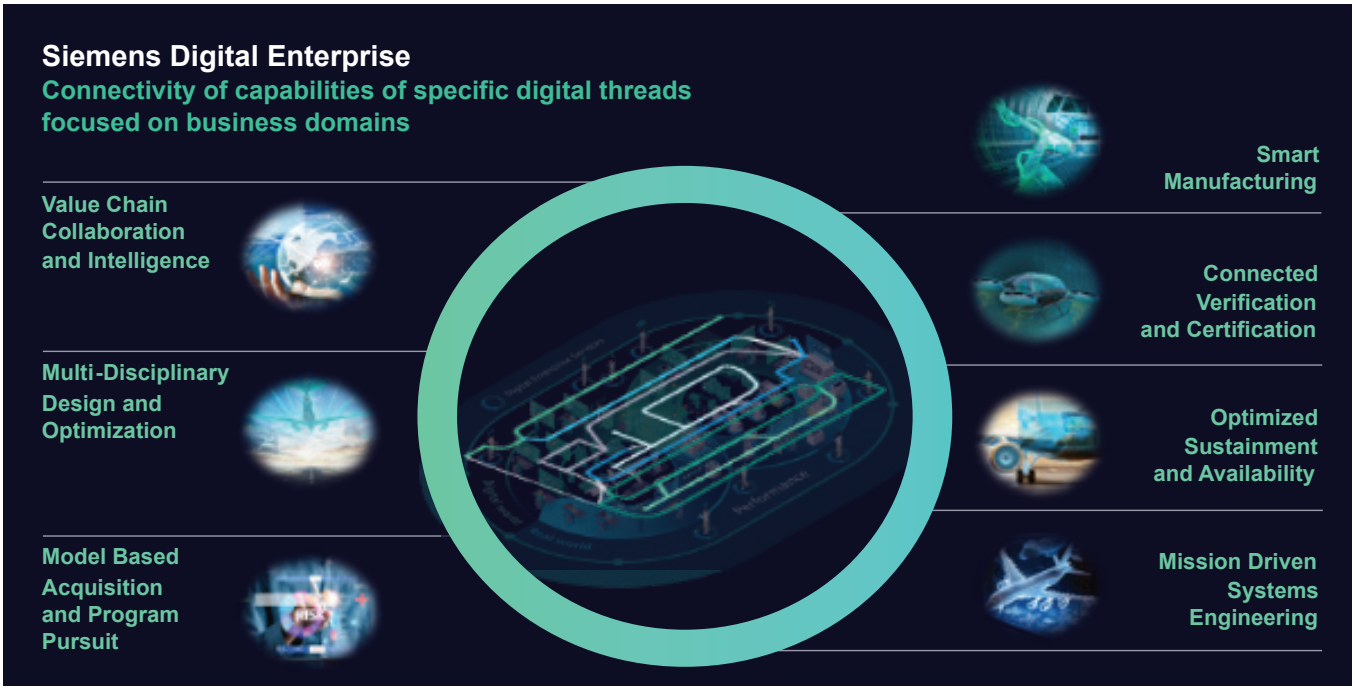
To specify the models or collection of simulation/analytical models that a digital twin represents, it is important to consider the purpose of the digital twin and the types of data and insights that are needed:

- Define the business goals the modeling activity is intended to satisfy. Establish how satisfying the business goals will be measured
- Define the purpose of the digital twin, including what problems it is intended to solve and what insights it is intended to provide. Bound the scope to a single or small number of problems, with planned interfaces to interact with other twins
- Show the components of the system that need to be modeled to achieve the purpose of the digital twin
- Determine the level of fidelity required for the models, including whether high-level or detailed models are needed
- Choose the appropriate modeling techniques for the components being modeled
- Perform incremental validation of the models using testing and analysis to ensure they accurately represent the system and are converging toward satisfaction of the business goal(s)
- Integrate the models into the digital twin to develop the representation of the system
- Validate the integrated models still represent the system and are converging toward satisfaction of the business goal(s)

Some types of business goals that are relevant are:

- Establish the performance of a system
- Determine the per-unit cost of manufacturing the system for prototypes, low- and full-rate production
- Establish the first pass yield of production
- Estimate the mean time between failures in the system and the criticality of each type of failure
- Identify the level of repair of the system components and the locations where repair and maintenance can be performed
- Identify supply chain risk in sourcing parts, materials for manufacturing and risk in distributing the product to the end customer
- Ensure correct behavior of the system in its expected environments and under extreme conditions
- Respond to changes in data with accurate results within two weeks of publication of the data. Document the response to change (incorporated, rejected, no impact, etc.)
- Identify the source of defects in manufacturing within three business days of identifying of a trend
- Determine the source of reduced field performance of the product and estimate the time to failure

Siemens describes our products by digital threads and business personas. These are visualized as subway lines with various stops. Each stop includes process diagrams describing how our products connect to bring value to our customers. The process diagrams are further decomposed into scenarios (like a user story) that enable decisions in specific business domains. An example is depicted below:



Determining the level of connectivity between models to support a digital twin

The level of connectivity expected should be established based on the business goal(s) the models address. With modern interfaces and data formats, a high level of connectivity between models is expected. The digital twin should be able to access and integrate data from a variety of sources, including design models, simulation models and operational data to provide a comprehensive view of the system. The level of connectivity should be balanced with the effort required to establish connectivity and the value the connected model set will provide. Although it is possible to connect a large number of models and see the effect of simulation of the large model set, the result may not provide a practical or useful outcome. Consider a large, connected model with two simulation runs with a single noncritical parameter that varies by 0.5 percent, which may not be useful to perform both simulations. Therefore, the level of connectivity should be flexible and allow variation.

Factors that drive the requirements of a digital twin

Several factors can drive the requirements of a digital twin, with the most important being the types of analysis that a digital twin must be able to satisfy. Here are some examples:

- **Accuracy and reliability:** A digital twin may not require a high degree of accuracy to satisfy many analyses, or only a portion of the twin may have high accuracy while other portions are low accuracy. For example, in model calibration, the digital twin must be physically accurate and precise as well as accurately represent the behavior of the system being modeled. For first article inspection (FAI), only a portion of the model may be required to be accurate, and the model may not have a behavioral component
- **Connectivity of the system:** The degree to which the internal workings of a system are connected can drive the requirements for the digital twin. If the intricate details of the system are needed, for example to optimize the system, the digital twin may also have requirements to exhibit this connectivity
- **Performance and efficiency:** The performance and efficiency of the model set can drive the requirements for the digital twin. This is especially true when the digital twin is used in virtual simulations or when comparing multiple systems. Requirements for performance and efficiency also work with requirements for accuracy and reliability; it may be necessary to show performance requirements across a range of environmental conditions or multiple twins may be established with purpose-built performance that is highly accurate only in corner cases
- **Cost-effectiveness**: The cost-effectiveness of the digital twin can be a driving factor. The cost of implementing a digital twin must be justified for the risk reduction and cost savings. For example, in manufacturing FAI, the digital twin should be more effective than simply creating a prototype part, tooling and inspection tooling/instrumentation. The most cost-effective means should be implemented to provide value to the organization
- **Interoperability**: The level of interoperability required by the digital twin influences the requirements. For example, in model calibration, the digital twin must be able to communicate with other models and data sources to provide a comprehensive view of the system, while in FAI, the twin may be required to communicate with measurement equipment and measured data sources
- **Security and privacy:** The security and privacy of the digital twin are always a concern. If the digital twin is detailed enough to expose intellectual property (IP) such as manufacturing techniques, it must be properly protected from unauthorized use

The requirements that a digital twin must meet vary. The requirements may also change over time, so a flexible system is needed to manage both the requirements and their impact on each digital twin.

Siemens Xcelerator products serve as the foundation for a well-formed, interconnected digital engineering ecosystem. At the core of the portfolio is Teamcenter, the foundational PLM system that enables teams to orchestrate change, configuration, collaboration, workflow management and storage of digital artifacts. This foundation is built to meet the needs of the systems engineering organization by ingesting, decomposing, disseminating and relating the system definition to the rationale for the definition and implementation of the solution.

The system definition is typically driven by documentation created during a contracting phase or concept definition phase, where the problem space is modeled and simulated to identify areas of concern to be addressed. Using Siemens Xcelerator supports the ability to manage simulation artifacts and simulation activity, allowing a coordinated and highly visible modeling and simulation approach to problem space exploration. Siemens has partnered with simulation companies on the industrial metaverse to enable behavioral simulation, event-driven simulation and physics-based game theory simulation of real-world environments. The simulation artifacts, processes, evolution and outcomes are stored in Teamcenter. This allows the parametric information identified in simulation to be codified as objects that continue to evolve with requirements elaboration, contract creation and validation. Connecting simulation and early concept development to the contracting activity enables a more robust technical data package to be delivered to suppliers, jumpstarting the supplier in developing a compelling offering. The simulation and early concept data is then used throughout the lifecycle of solution development, cross-checking the alignment of the solution under development to meet the mission needs. The simulation and early concept data is also used as the starting point for the development of the validation and system test plan, defining system tests that evolve with the system definition, enabling incremental and virtual validation.

Deciding on sensor placement and data collection

This is highly dependent on how the system is modeled. Ideally, the model's structure shows the interfaces when the behavior will be shown, and the system is segmented so that portions are independently verifiable. Each verification case can be modeled as part of the system model, allowing collaboration between the design team and verification team. The verification case can detail the input stimulus and expected response at each interface, as well as define the order of operations to perform verification. Details on the interfaces that are used in the verification case should include the data requirements and quality of the data at the interface. The impact of the changes required to support physical sensors on the product should also be considered; if the sensors are temporary or not integrated into the product, the real-word accuracy of the results is reduced.

Here are some strategies for using system models to inform these decisions:

- Place sensors at areas of high concentrations of stress, or that will be subject to multiple types of stress simultaneously
- Place enough sensors to detect if a sensor is giving improper or implausible readings
- Place sensors in areas to support analysis and decision-making using the models that are part of the digital twin
- Identify the critical components of the system being modeled, their failure modes and effects and the inputs and outputs of these components. This aids in figuring out the interdependence between components and subsystems and assessing the criticality to safety. This determination helps show which sensors or virtual sensors are needed to monitor these components
- Define the data requirements for the digital twin, including what types of data are needed and how frequently it should be collected
- Consider the quality of the data that will be collected by the sensors or virtual sensors, including accuracy, precision and reliability. This will help inform decisions on data massaging and processing
- Evaluate placement options for the sensors or virtual sensors, considering factors such as accessibility, ease of installation and potential impact on the system being monitored
- Integrate the sensors or virtual sensors into the digital twin, ensuring they are properly calibrated and configured to collect the necessary data. In some cases, it may be necessary to embed additional physical or virtual sensors in the system itself. A digital twin can help identify when a sensor may need to be considered as part of the system or added as instrumentation external to the system
- Monitor the data collected by the sensors or virtual sensors and adjust sensor placement and data massaging as needed to ensure the digital twin provides accurate and useful insights into the behavior of the system

Distributed API access to facets of the digital thread

Consider a digital twin with a mostly abstract representation of a system at a low level of fidelity. It may be simpler to retrieve the model set and run it on the desktop. API access would only need to be provided to retrieve the model. Alternatively, consider a digital twin that is a high-fidelity representation of the system that may span multiple sources of truth. API access may need to provide the ability to ingest data, execute the model and produce reports on the results. The breadth of API accesses is as rich as the breadth of twins. Here are some useful examples of API access:

- **Create/read/update/delete operations (CRUD):** Basic CRUD operations form the foundation for most APIs. When these operations are applied to a digital twin, they focus on the data needed to execute the twin and the results the twin produces
- **Start/stop/restart:** The environment that holds or represents the digital twin may be executable. If this is the case, API access is needed to control the execution environment

- **Access control and authentication:** A digital twin may have various levels of fidelity or different functionality provided based on the access to the model repository. For example, if a digital twin is being created for internal employees, it may contain highly accurate models and measured data. When that same twin is accessed by a supplier, the model repository may respond with an alternate representation of the model that is less accurate
- **Configuration control APIs:** The models in the repository may be evolving as the digital twin is being created. It is necessary for the digital twin to switch from the latest published version, the latest version or some historic version. This will allow the twin to accurately reflect the state of the system it represents at various lifecycle states
- **Metadata APIs:** When composing a digital twin, it may be necessary for the model repository to expose the library of models it stores and information about each model. Metadata such as the fidelity of the model, the latest release date, the pedigree and provenance of the model, and how many other twins it is used in may be important in selecting the right model to include.

It is worth noting that the API accesses in the model repositories should adhere as much as possible to published industry standards. The digital twin ecosystem and models used to represent twins will evolve, and open standards can enable a more rapid evolution of the twins.

Supporting relationships between models and digital twins

PLM systems must support managing relationships between models and a digital twin for them to be effective. PLM systems are designed to manage ALL the data and information related to the lifecycle of a product, including design, manufacturing and operational data. The more comprehensive the data set and the accuracy of the relationships of those artifacts pertaining to the product that is contained within the digital twin, the more valuable it becomes. PLM systems can be used to:

- Manage the models used in the digital twin, including their version history, revision control and relationships with other models
- Manage the digital twin, including its configuration, data sources and relationships to other systems and applications
- Integrate data from various sources, including computer-aided design (CAD) models, simulation models and operational data to support the creation and maintenance of the digital twin. Ideally, traditionally siloed data sources (like manufacturing operations, inventory and service) become integrated into the PLM system and the digital twin
- Facilitate collaboration between stakeholders to enable better decision-making throughout the lifecycle. PLM systems can facilitate the best-informed collaboration between different stakeholders involved in the creation and maintenance of the digital twin, including designers, engineers, builders and operators to drive the fastest and most frequent decisions impacting the quality and performance of the product
- Provide analytics and insights into the quality and performance of the digital twin and the system being modeled, supporting continuous improvement and optimization from concept development through the end of life of the system

The PLM system in Siemens Xcelerator offers an object level access control scheme. This access control scheme allows objects (such as requirements, CAD, system elements, parameters, etc.) to be controlled at the user, group, organization, classification, project and system level. The access controls are enforced across objects, preventing issues such as data leaks due to mishandling data with workflows. This is particularly important in the concept and system definition phases of a program, where multiple subcontractors may be involved that may not have a need to know some or all the program data.

When a government reference architecture and solution architectures are managed in Siemens Xcelerator, many previously tedious tasks can be automated or eliminated. The first and most valuable task affected is the ability to publish and distribute the architecture, information about it and the relationships between it and supporting objects. System engineers use their native modeling tools to interact with Siemens Xcelerator, reducing the time spent switching from one tool to another. This allows the system engineer to focus on his/her core tasks while collaborating with those who are not system engineers. Siemens Xcelerator is also web enabled and the systems engineering tools directly embed this web interface, allowing the system engineer to perform collaboration, discussion, change management, analysis and other activities directly from the modeling tool. For the person who is not a system engineer, when the architecture is synchronized with Siemens Xcelerator, it is broken down into its component functional and logical elements. The architecture requirements and parameters are also synchronized. These elements are then available for use in other engineering domains. For example, parameters such as the weight and cost of the system can be used in CAD or cost estimation applications that are connected to Siemens Xcelerator. The interfaces defined in the logical elements can be related to interface control documents, parameters and protocol specifications unique to that interface. Functional elements can be related to the software or firmware objects that implement them, allowing change orchestration between a hardware supplier and a subcontractor that may only be providing software. In addition to the decomposition of a single architecture, using Siemens Xcelerator offers the ability to relate and act on system architectures that may be in different design tools, allowing a reference architecture to be published in one modeling tool, and one or more solution architectures to be published in another modeling tool.

Siemens Xcelerator offers multiple facilities for exchanging data primarily through the implementation of open standards. Data in Siemens Xcelerator can be accessed using protocols such as REST and OData. This enables Siemens Xcelerator to be used to either form the foundation of the digital engineering environment or merely participate in several digital threads in the environment. The object types that are accessible by these interfaces are not limited, allowing the system definition to span multiple authoritative systems of truth.

By using a PLM system to manage relationships between models and the digital twin, organizations can ensure the digital twin is properly configured, up to date and integrated with other systems and applications, supporting improved performance, reduced maintenance costs and enhanced customer satisfaction.

SysMLv2

For about the last 17 years, systems engineers have been using the SysML modeling language to perform model-based systems engineering (MBSE). When it came out, SysML started a revolution in the way we think about and describe systems. However, the heavy dependence on SysML and the limitations of the foundational unified modeling language (UML) language have driven the industry to call for an upgrade. Version 2 of SysML is in the works and is expected to be released by the Object Management Group (OMG) in 2025. SysMLv2 brings the promise of a purpose-built language, increasing compatibility, interoperability and offering better methods of defining lifecycle properties. This combination of features enables improved interaction with digital threads and the possibility of documenting the definition of a digital twin (some of the SysMLv2 language constructs that support these activities are referenced using the Courier New font in the examples below). However, SysMLv2 does not describe an engineering method or process, opening the possibility of defining a standard way to describe twins and threads.

V2 and the digital thread

The digital thread can be viewed as a way to connect disparate engineering information for providing computer assistance in defining and executing the design, manufacture and support of products. Generically, defining the thread can be thought of as using the input-process-output (IPO) method, where each step or activity in the thread is a process that takes input and produces output. We add feedback and configuration control to the complete set of processes to describe the thread, then inspect the processes and movement of data for automation. SysMLv2 has a set of elements as part of the base language to describe this process and data combination. In addition, since elements follow a definition/usage pattern, repeatable processes can be defined (as actions that are performed), then used in multiple threads or in different places in the same thread. Most elements can also have attributes, allowing the exchange of parametric or structured information and metadata in addition to the element itself.

While it is possible to write requirements in SysMLv2, it is more beneficial to use purpose-built tools to write and manage requirements, then provide a copy or representation using v2. This allows other nonrequirements tools to make queries of the v2 model using standard notations (REST, OSLC, Python, Java, .NET, etc., known as the Platform Specific Model (PSM)) without having to parse text or rely on translators. In the IPO process and along the digital thread, providing a SysMLv2 interface to the process on both the input and output sides increases the interoperability of the process. We can make queries of the requirements tool using v2 and get the list of requirements back, then apply those requirements using v2 semantics to show where or how the thread satisfies the requirements. This is useful when we can read files, but much more interesting when we mechanize these tools with a digital thread.

Many options are available to mechanize the digital thread using SysMLv2 repositories. The v2 standard defines basic patterns for commit, branch, and tag as the basic interface to a repository. One interesting solution uses multiple vendor independent v2 repositories. In this scenario, we need a third capability to manage the multiple data sets across repositories; in other words, we need a configuration management system of the digital thread. The digital thread configuration management system needs to know each repository, the data stored, the pattern to use to combine the data and it must interact with the branching and tagging system to properly combine the data. In this way, multiple parallel authors may work in vendor or domain specific tools and if a v2 representation is stored and available, it can be combined with other data sets. We can continue to explore the configuration of the digital thread in this multiple repository managed fashion by adding in authorization controls, location specific caches and other infrastructure considerations partially independently of the engineering activity. Another advantage of the multirepository approach is if a software vendor does not provide the complete set of features required, the individual company can build their own capabilities to operate in parallel with commercial software.

V2 and the digital twin

Users today are trying to expand the use cases applied to SysML v1 into areas where it doesn't fit. We've seen users trying to replace PLM systems with a v1 model, storing snapshots of files with the model or attempting to manage change processes with the model. These use cases require significant customization of the modeling tools, often overburdening the modelers with hastily designed manual tasks and static (usually out of date) snapshots of data. These use cases violate the general principle of separation of architecture and implementation, effectively handcuffing the implementation teams and eliminating any hope of reuse.

SysMLv2 does not prevent users from tightly coupling the architecture with one implementation, but it does provide data constructs that can help separate these concerns. When a design methodology is applied on v2, the choices for what and how to model can be both documented and codified into the language. Codifying the design methodology into v2 will enable verification tools to interrogate the models for correctness in the application of the model, or automation to extract only the definition or only the documentation or even separate the definition of the system from a single instance of an implementation.

Consider a modeling methodology that emphasizes our ability to enable a robust digital twin. In this methodology, we require separate packages for the most popular architecture usages (the external boundary between the system and the world), requirements usages (the specific requirements that apply to the top-most architecture), views (the filtered model elements that address a stakeholder concern) and cases (use case, analysis case, verification case, trade study case). Definitions may be either included in the same packages or kept in separate packages. By organizing models this way, we can specify that every use that appears in these packages is considered included as part of the architecture of the system of interest.

We then specify that a separate set of packages be automatically defined by our configuration-managed digital thread. As specific implementation choices are made, the part and attribute usages are created in an implementation package, specializing in or subsetting the usages in the architecture. The next step is to document the result of manufacturing each lot – this time automatically generated from manufacturing execution system (MES) or manufacturing operations management (MOM) systems – as a snapshot of the implementation package. The snapshot construct in v2 enables defining the system at a point in time or for a period of time (consider either the as-is state today, or the maintenance schedule that should be performed after 5,000 hours of service). Finally, a model is made using the individual construct to represent each serial number of each system that rolls off the assembly line. The snapshot can include updated values for the attributes, showing how an individual has evolved over time. It should be noted the automatic creation of the implementation, snapshot and individual models may be dynamic; meaning these models can be made from existing PLM, MES, MOM and enterprise resource planning (ERP) data and not stored; or the existing systems of record can respond in a v2 format for the data requested.

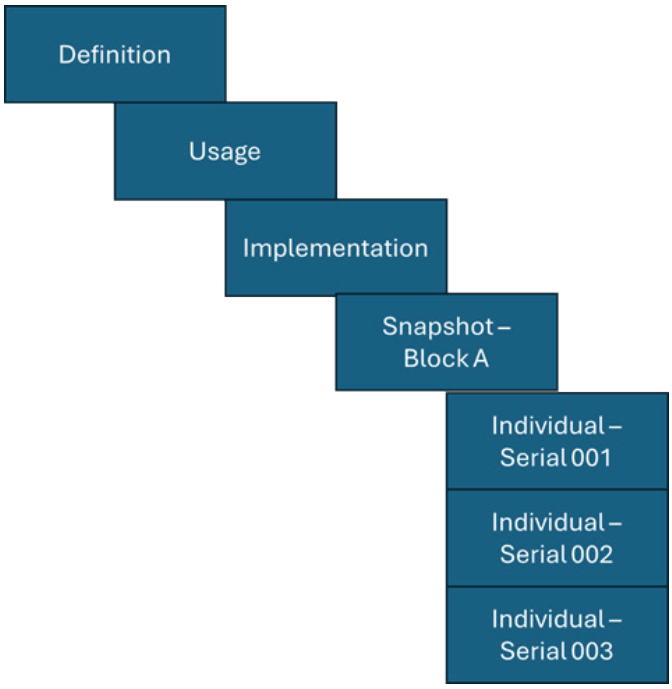


Figure 1. Organization of a SysMLv2 methodology.

Responding to queries about a digital twin of the system using SysMLv2 enables the possibility to use a simpler, vendor-independent interface without necessarily needing a direct connection to a system of record. When combined with the digital thread, the process of publishing the digital twin can be automated, reducing the need for people to perform multiple data entries.

Conclusion

The increased expectations of a digital twin have driven the need for technology standards to evolve in support of them. Companies can no longer tolerate heavily proprietary model formats and inaccessible data sets. Highly integrated products have also led to the improved need for collaboration between models, contributing to the demand for interoperability. Thankfully the standards organizations have responded to the demand and software vendors such as Siemens have faithfully implemented the new standards. Technologies such as SysMLv2, Jupiter Tessellation (JT) and PLM XML facilitate the open exchange of information in standard formats. Digital thread needs have also caused a growth in available standards, with technologies such as JSON, REST and OSLC supporting multiple access methods to interact with the thread. Although these technologies have taken great leaps forward, there is still room for improvement. The industry would benefit from standards that define the execution semantics for a thread, assist in visualizing the digital twin and digital threads, establishing the basis for digital twin model metadata (STEP AP243, MoSSEC is a good start), federating and coordinating change across repositories (OSLC tracked resource sets is a good start), and defining quality metrics for threads and twins are needed.

Note

This series of papers was written by Siemens in support of a request for PLM Vendor Perspectives by the American Institute of Aeronautics and Astronautics (A)IAA Digital Engineering Integration Committee (DEIC). The final AIAA Paper, "Digital thread and twin Integration: Defining the problem space – OEM and vendor perspective on gaps" delves into the challenges and gaps identified by the industry while integrating digital threads with the digital twin. The AIAA paper builds on the past digital thread and twin position and realization papers published during 2021 and 2022 and will be presented at AIAA SciTech 2025.

Siemens Digital Industries Software

Americas: 1 800 498 5351

EMEA: 00 800 70002222

Asia-Pacific: 001 800 03061910

For additional numbers, click here.

Siemens Digital Industries Software helps organizations of all sizes digitally transform using software, hardware and services from the Siemens Xcelerator business platform. Siemens' software and the comprehensive digital twin enable companies to optimize their design, engineering and manufacturing processes to turn today's ideas into the sustainable products of the future. From chips to entire systems, from product to process, across all industries, Siemens Digital Industries Software – Accelerating transformation.

siemens.com/software

86239-D2 9/25 K