SIEMENS
Ingenuity for life

Model-based systems engineering

Integrated MBSE in automotive

Executive summary

Siemens Digital Industries Software believes that today's complex automotive products require a systems-driven approach to product development that combines systems engineering with an integrated product definition and the ability to unify your product development framework with your manufacturing and shop floor operations. To facilitate model-based systems engineering, Siemens Digital Industries Software provides functional networking, a consistent process-enabled framework, advanced modeling and simulation, an intuitive user experience and an open product lifecycle management (PLM) environment.

siemens.com/plm

Contents

Overview

Twenty-first century engineering enterprises face new challenges as they strive to create the best products in increasingly short timeframes. For many industries, each new generation of products is inevitably more sophisticated and complex as new product versions require the participation of more engineering disciplines than ever before. Along these lines, the automotive industry has spent the last 100 years evolving automotive vehicles from purely mechanical devices to combinations of mechanical parts and electronics that are now controlled through onboard computer systems.

The continued introduction of electrical, electronics and software components into the product development process has created the need for more efficient and effective integration of all participating engineering disciplines. Previously, much of this cross-domain knowledge was held in the heads of individual engineers. But that is no longer possible.

Today's automotive vehicles are too complex. And because of that complexity, it takes a long time to be absolutely sure the automotive supply chain has covered everything and understands the impact that individual decisions are having on other aspects of the product design or its manufacturing processes.

Unfortunately, taking extra time to get it right can cause your resource costs to increase significantly and your time-to-market to extend its cycle times. On the other hand, if you rush your process too much, you end up paying the warranty penalty. Trying to solve this dilemma by reducing content complexity doesn't work either, since eliminating new features means removing the innovations that your consumers want.

Simply put, traditional product development no longer works. You can't do it all manually anymore, especially when the people you need to work with (your supply chain) are now globally dispersed.

All of these factors combine to create a new imperative for the automotive industry. The need to manage sophisticated products requires a model-based process. This is why systems engineering is a key element in the Siemens Digital Industries Software's vision for automotive companies and their suppliers. Our systems-driven approach to automotive product development marries systems engineering with an integrated product definition and the ability to integrate product development with what actually happens in your manufacturing and assembly plants.

Siemens Digital Industries Software's approach to systems driven product development is facilitated by the following key capabilities.

Functional networks, a series of networks that facilitate the functional breakdown of your entire product, including requirements allocated to the product's various functions, logical relationships defined between these functions and the physical implementation of these functions.

Consistent process framework, a framework that crosses all of your product development and manufacturing domains and integrates your functional network through change and issue management, configuration management and schedule management – thereby unifying your mechanical, electrical, software, and electronic development domains with your process planning and plant operations.

Advanced modeling and simulation, the ability to derive engineering requirements from user needs and validate that your engineering specifications fulfill these needs early in the product development process.

Open PLM environment, a PLM-driven infrastructure where your home-grown and commercial software tools can be easily integrated and efficiently leveraged.

Intuitive user experience, the ability to enable all of your product development and manufacturing/production stakeholders to access all of the product, process and manufacturing information that they need to intuitively understand your product as a complete system – as well as to help these users, depending on their task and role, proactively leverage this information in the context of their job tasks.

This white paper describes on how these capabilities can be leveraged to provide automakers with a robust model-based systems engineering environment.

Functional networks

Functions drive the product's lifecycle

Today's vehicles can be seen an integrated set of functions. A car has to accelerate. A car has to stop. A car has to provide a safe and comfortable passenger environment. These are just a few examples of the high level functions that a vehicle has to provide.

These high level functions can be broken down – decomposed – into lower levels of more specific, detailed functions that eventually are delivered or implemented by physical components. Most of the functions of today's cars are the same from one vehicle to another – and have been for years (of course that is now changing with the advent of electric vehicles).

What makes each car unique is the set of requirements that are applied to these functions. How fast does the car have to move? What is its required stopping distance and from what speed? How safe does the passenger compartment have to be? And, how comfortable? The answer to all of these questions can be found in the vehicle's requirements.

Requirements are allocated (related) to specific functions so that product teams understand what criteria to use when designing specific functions. And of course, requirements and functions have dependencies between each other that need to be understood so that proper trade-off decisions can be made. If the car is required to have a top speed of 175 mph, then that speed has an impact on the function of stopping the car – and has to be understood since that particular car is going to need really big brakes!

Basically then, all of these dependencies have to be modeled and understood so that as design decisions are made, the car's relationships and dependencies can be evaluated – and the best possible design will be developed.

The way functions will be implemented is defined in a logical architecture. Logical models can, and often do, exist at each level of the functional decomposition – and they define how the different functions interact with each other. In electrical terms, this would be expressed in a diagram that indicates what inputs and outputs on one device are connected to specific inputs and outputs on another device.

Siemens Digital Industries Software's model-based systems engineering solution provides a complete set of capabilities to put all of this together – the functional model, allocated requirements and the logical model – in functional networks where all relationships and dependencies are defined and understood throughout the entire vehicle.

Optimize re-use and manage evolving technologies

Sixty to 80 percent of a vehicle is functionally the same from one vehicle to the next. Logically, the way the functions interact is fairly similar as well (again, note the electric vehicle exception since electric vehicles will cause new models to be constructed from both a functional and logical perspective). Requirements can come from many places, including customers, regulatory bodies, government, internal development standards and manufacturing best practices. Product teams collect these requirements and manage them as individual objects. This object approach allows requirements to be related to one or multiple functions that they affect, even when these functions cross different vehicle programs. Requirements also determine variation in the functional and logical models as they might require new features or different technologies.

Systems engineering plays a crucial role in facilitating an effective requirements process. Since requirements originate from many different sources, they need to be evaluated, refined and ultimately focused on a particular vehicle platform or program. As the product team moves from a very coarse definition of the vehicle at the concept level to a fine-grained or detailed representation at the component level, the vehicle's requirements, functions and logical models are also refined, gaining more detail and granularity as they move down the development side of the system's V.

Validate early in the development process

The resulting functional networks and the interactions between product functions describe the interfaces and constraints that eventually will be implemented by various organizational domains (mechanical, electrical, software, manufacturing process, risk, reliability, cost, service, and disposal). This network includes descriptions of the interfaces. In conjunction with exchanged

inputs and outputs, these interfaces – as well as the functional definitions, and their related performance data – are required to validate the behavior of each function. For example, consider a sun roof that automatically closes when it starts to rain. This capability is a high-level function that is allocated to the components necessary to carry out that function – the motor, sensors, ECU, mechanical parts, wire harness, and other equipment. The functional description, in conjunction with logical definition, is the basis for creating behavioral models that simulate the function as well as its interdependencies to other functions.

One model across all variants – the 150 percent view
The result of this process is a complete functional model of an entire system (also known as the product platform). This "150 percent model" covers all possible product configurations – including combinations that may not be actually produced. Since this model describes the logical interaction of the vehicle at different granularity levels (such as the system and subsystem level), it can play a crucial role in helping the product team to find conflicts that potentially have escaped notice and might cause extra expense, late re-design or worse, field recalls.

Many different models are required to refine functions and gain important insights, including: signal analysis, energy transfer diagram, state diagrams, FMEAs, EMI/EMC, network diagrams, fault trees, behavior and manufacturing processes to name just a few. The end result is a specification for the product's physical elements.

Managing complexity and tracking against high-level goals – integration architecture
Vehicle programs are evaluated in terms of their ability to achieve goals that, ideally, are formulated as measurable targets – such as fuel economy, vehicle mass or safety. In system-driven product development, these vehicle-level targets are decomposed ("engineered") into targets for subsystems or specific vehicle properties – such as engine parameters, transmission parameters, mass and aero drag. This translation is refined from high level parameters for subsystems to increasingly detailed parameters of the subsystem components based on heuristic or quantitative models that are continuously refined to describe these dependencies as accurately as possible.
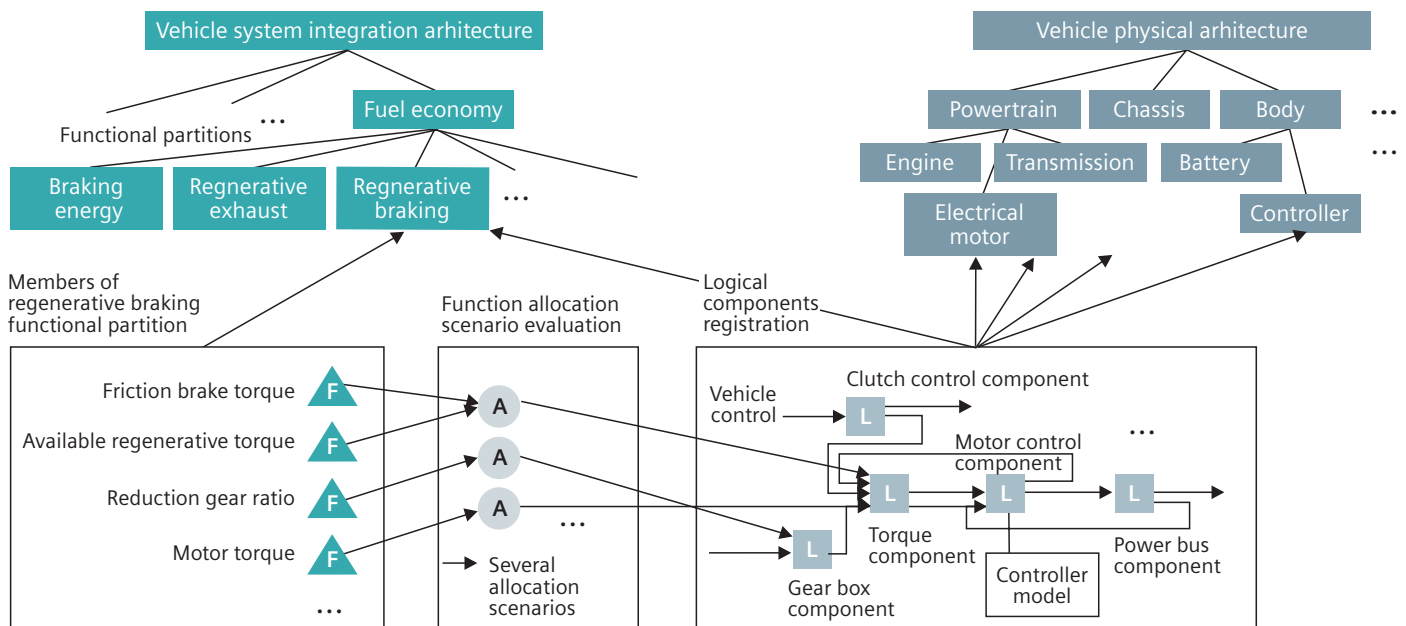
This refinement of high level targets to lower level targets creates a structure that ultimately enables individual engineers to understand how changes that they make to a component or subsystem impact the goals of the vehicle program. This structure also allows the program manager to track if the program is "on target" for its high level goals. If it is not, the structure enables the program manager to determine what causes the deviation and how it can be compensated. This structure is called the "integration architecture."

Even if a vehicle has thousands of requirements and functions, only a small number of these are associated to each of its elements. The integration architecture enables engineers to manage this level of complexity while understanding how it contributes to the performance of the whole system.

Describing the mechatronics system – physical architecture

For mechatronics systems like today's automotive vehicles, the physical architecture contains all mechanical components and assemblies, the electrical architecture (including ECUs, sensors, wire harnesses and corresponding schematics), signals and messages, as well as the complete software bill of materials (including boot loaders, application software, configuration files, calibration data). Logical and functional components are associated to the physical architecture. For example, a behavioral model for the function of the electric sun roof is associated with the corresponding elements in the physical architecture so that engineers can understand how a changed part or software component might require the underlying behavioral model to be updated and re-validated.

The integration architecture and the physical architecture organize the requirements, functional, logical, and physical view of the vehicle so that complexity at the lowest level is manageable even if mechanics, electronics and software are all involved. These factors are also highly re-usable from one vehicle program to the next and across multiple vehicle platforms.

Configurable plant and process management – systems-driven process throughout the product lifecycle

The systems engineering process extends into manufacturing, including designing, validating and testing production processes and manufacturing equipment needed for a particular product. The translation of manufacturing requirements into the equipment and operations definition and driving validation through virtual commissioning is analogous to the systems engineering process for product design.

The process starts by identifying manufacturing requirements based on the product definition (in the example below requirements for the sheet metal used for the engine compartment). From the product definition (which is mostly a combination of 3D CAD and PMI data), product teams can derive specific requirements for the production of the detailed product design. Once these requirements are understood, the product team might identify an existing plant where the vehicle will be produced or select the most applicable plant if additional equipment will be needed.

Once the equipment has been defined, the process continues with the definition of work orders, the equipment BOM and layout, which then serve as a basis for developing and validating the required automation for a given production line or cell. At each step of this process, the product team will perform some form of validation based on associated product and production requirements. It is important to validate both the product and production requirements (for example, finishing tolerance of the parts is a product requirement that can only be validated during production definition).
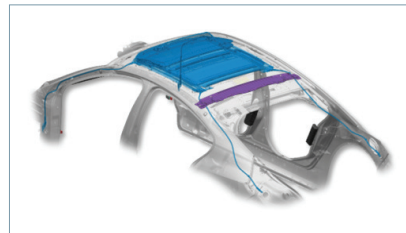
Siemens Digital Industries Software's approach to model-based systems engineering takes the physical aspect of all dimensions of the design into account – including parts, physical design, processes, plants and the actual physical vehicle represented by a VIN number. This comprehensive approach is known as the RFLP5 method.

Traceability

Requirements

Functional

Logical

Physical design

Part

Process

Plant

Physical

Knowledge re-use
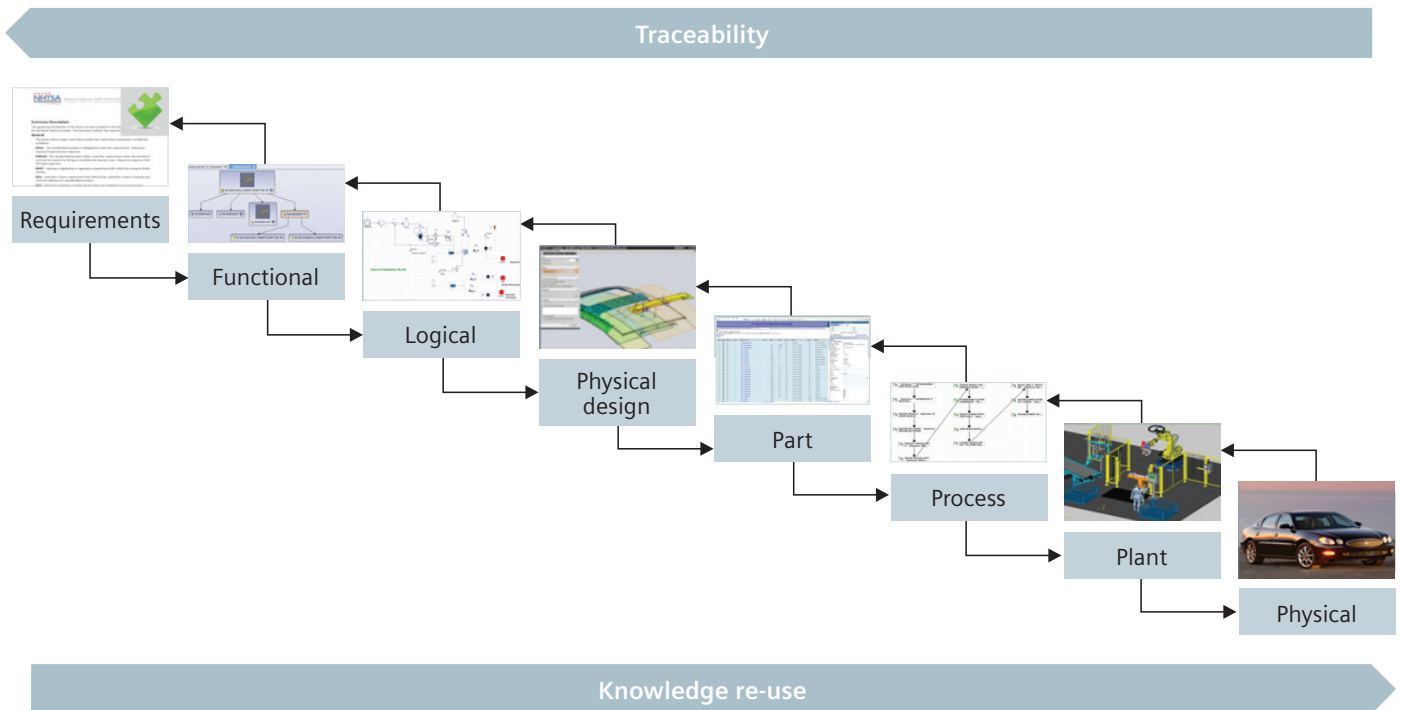
Traceability and consistency – integration from development to manufacturing and service

Applying the functional network across the entire product lifecycle, including manufacturing and service, helps remove the "work and reconcile" tasks that occur in most vehicle development programs. Today, gates or milestone reviews provide an opportunity to bring everyone together and reconcile the integration of all the pieces that must work together to deliver the function. Our functional networks enable you to build-in an integration framework from the beginning. This enables you to align all of your functional organizations at any point in the development process, not just at certain milestones.

In addition, the physical design can also act as a template, bringing automation and knowledge re-use into the model. Template values can be derived from requirements or other relationships and dependencies. Rules can be applied through templates that ensure boundaries established by related requirements are automatically checked and flagged if an invalid condition is encountered.

Consistent process framework

In order to make the systems-driven approach work consistently for every engineer, Siemens Digital Industries Software's model-based systems engineering solution provides five key capabilities, including:

Ability to connect all engineering data in a common data model, which facilitates a deep understanding of dependencies created in the functional and logical architecture.

Open PLM platform for interacting with specific authoring tools, which enables product teams to interface with a variety of authoring tools across multiple domains.

Ability to minimize variability across all domains, by establishing one common, consistent product configuration framework for all engineers.

Common program and platform management, which drives the PLM process in a single program context from program initiation to service

Change and issue management across all domains, which ensures complete traceability throughout the PLM process.

Connect all engineering data in a common object data model

Driving the development process through functions creates consistency between your mechanical, electronic, electrical and software processes. Each domain is able to use a different set of methods and tools, while sharing the product functions and leveraging a common definition that engineers can work with on a cross-domain basis.

However, the dependencies between engineering domains can go very deep into the actual design data. For example, an electrical designer will develop a schematic for a wire harness based on the function definition. This schematic contains all of the information about the components that interconnect. Nevertheless, the engineer will have to make an assumption on the length of the connectors. Later, another engineer will design the actual routing of the cables, determining the actual length. This information is important for the electrical engineer. With this in mind, the system-driven product development solution needs to provide a data model that represents the schematic and is able to link parameters in the schematic to parameters of the actual "mechanical" design of a wire harness.

Siemens Digital Industries Software addresses these needs by providing an open, standards based PLM solution with a single source of product and process knowledge and a common data object model that can be leveraged across multiple engineering domains. Openness means two things:

• Providing a design data model based on open standards wherever possible

• Supporting and driving integration with domain specific design tools

Working with our PLM-enabled framework, individual development teams can retain their focus by continuing to use their mechanical, electrical, electronic or software design methods and tools of choice, while working together in-context to meet overall development goals. This approach enables them to achieve higher product-related quality goals and safety requirements, which result in lower warranty cost, better customer satisfaction and shorter time-to-market.

Leverage an open platform to facilitate use of the "right" authoring tools

No supply chain, user community or technology vendor starts fresh. Established ecosystems of software tools already exist. It is not realistic to require your engineering and manufacturing domains to adopt new tools as you implement your system-driven product development solution. As a result, you will need an open PLM environment that enables you to consolidate a data model based on open, published standards.

Siemens Digital Industries Software provides the following integrations to today's most widely used authoring, design and diagnostics tools to ensure the consistency of your PLM-driven processes.

Mechanical lifecycle, including support for major MCAD tools such as NX™ software, Catia, Pro-E, SolidWorks, SolidEdge® software and Inventor.

Electronics lifecycle, including integrations with ECAD PCB design tools from Mentor, Cadence, Altium, Intercept, and a gateway for other EDA tools (Zuken and Protel) Teamcenter® Simulation Process Management software is used to integrate and manage multi-CAD part libraries, maintain links to vendor data, facilitate compliance management, support check-in/check-out and manage native design archives, fabrication and assembly data, derived files and extracted BOMs. Teamcenter also is used to facilitate native ECAD data visualization and design for assembly (DFA) processes that automatically validate a design against assembly rules to improve manufacturability.

Software lifecycle, including supporting source code, software design elements, parameters, binaries and hardware as part of the same product structure and linking this product structure to requirements and their dependencies across multiple engineering domains.

Within each engineering domain, the cycle of design, validation and test is executed based on the common PLM backbone. Domain-specific requirements are transformed into a detailed design and then validated against the requirements outlined for that domain.

Integrate nominal and failure-mode behavior

Most engineers automatically focus on the "working" (nominal) behavior of the product, and less on the "not working" behavior (failure modes). One of the reasons the latter is harder to do is that understanding, simulating and testing against failure modes is not possible without a systems perspective. Model-based systems engineering directly supports failure mode and effects analysis (FMEA).

FMEA can provide an analytical approach when dealing with potential failure modes and their associated causes. When considering possible failures in a design – like safety, cost, performance, quality and reliability – engineers can get a lot of information about how to alter the development/manufacturing process in order to avoid these failures. FMEA is an easy way to determine which risk raises the greatest concern, thereby identifying a potential problem before it arises.

Manage variability across all domains

Variability occurs everywhere in the development process: at the user requirements and functional network levels, within the logical architecture and across all of your engineering domains.

Robust cross-domain traceability has to be able to manage the variability in the interrelationships between requirements, functions and logical components, e.g., which versions of which requirements apply to which versions of which functions performed by which versions of which components, assembled in this factory for this customer. A variety of different variables can affect configuration including customer features, environment variables and engineering and order release status. Configuration and multi-domain navigation must be able to combine different kinds of variables in the right context to find relevant results quickly.

For example, analysis of a simulation may show that a sun roof does not close properly at 100 mph. To understand the results, engineers may want to view all the components in the product that specifically configure together with a sun roof. But they may also need to look further, since some components which may affect the sun roof operation (such as a roof support bar assembly) are always in the product when configuring engineering design options and may not show up with a specific configuration expression as direct as "sunroof = yes."

Relevance can be determined by many different conditions involving a combined configuration of requirements, functions, logical components, and designs, such as:

- Sharing a common variable expression such as "sunroof = yes"

- Configuration rules which imply a common variable expression such as "if 'model= sport_xyz' then 'sunroof = yes' "

Overlap in related functions or requirements can be relevant to the same logical component configuration. For example, functions for protecting the occupant during a crash might include a specific reference to a particular roof support component implemented by a specific design assembly re-used from a previous program without a sun roof. Multi-domain configuration combines configuration requests within domains (i.e., to find all requirements relevant to a particular program) and configuration for related data across domains.

To illustrate the concept of variant management, consider the following example; as part of the sun roof, two options are available – manual control and control based on a rain sensor. As illustrated below, these options can be mapped to an option package (e.g. trim level) called Basic and Fully Loaded. As users work their way through the model-based systems engineering process, they will be able to consistently refer to the option values that are defined as part of the product.

Leverage program and platform management

Program management supports enterprise project planning and execution. Team performance is enhanced through real-time and goal-directed collaboration. This form of organized collaboration should be available to all layers of the organization including your value network of suppliers and customers.

The integrated program management functionality in Siemens Digital Industries Software's model-based systems engineering solution includes capabilities for managing programs and their individual projects, allowing organizations to establish best practices and support continuous improvement. Real-time interaction among groups and members from distributed locations on specific business objectives and critical deadlines is enabled.

Program management is more than just establishing schedules and managing tasks. It also facilitates product platform configurability. In many companies, only a very few team members understand and control the variant data in their business systems.

CAD designers and part engineers are not expected to author the correct variability without guidance, although they must be aware of variability to support their design. Siemens Digital Industries Software's solution enables configuration experts to define-in variability using nomenclature that designers already understand. This use of the product architecture enables individual companies to use unique taxonomy to define their generic parts or modules. These generic parts are placeholders for design and can include parts that may not yet exist.

Establish change and issue management across all domains

System-driven product development process must be able to manage the complexities of product variation by accurately representing the "whole product" bill of material and common architecture breakdown for the product line – including software and electronics – for all product configurations. To address these needs, Siemens Digital Industries Software's solution provides:

Context management, including unique capabilities to simplify design collaboration by defining, managing and sharing working contexts for product development.

Change management, including a common infrastructure to consistently manage change across multiple domains and different development stages.

Lifecycle representations, including the ability to associatively define and relate multiple BOM representations of the product structure for different lifecycle stages or processes.

Option and variant capabilities, including the ability to organize products into modules and marketing options; thereby facilitating faster response to market opportunities and increased product/part re-usability.

Modeling and simulation

Early, virtual validation enables product teams to leverage models as much as possible instead of tying them to physical artifacts. Models can be used to represent component, subsystem, system, and product-level behaviors. Models serve three important purposes within model-based systems engineering by letting product teams:

- Covert marketplace and customer needs into engineering requirements

- Drive early validation

- Accelerate trade-off decisions to the early stages in product development

Behavioral models convert customer needs into engineering requirements

Models play a critical role in cascading customer needs down into engineering requirements. For example, a customer-driven need might be to reduce vibration during engine idling. The customer usually perceives vibration in the steering wheel and in the seat. In addition, perception to vibration can vary with vibration frequency.

How then can the customer need for a smooth idling experience be translated into engineering requirements for the suspension and the suspension bushings?

Systems-level models can play a very useful role in this process. Models can be built to capture the frequencies of the main subsystems. The model is then excited with frequencies corresponding to engine idle. Through a process of adjusting the model parameters, engineers can arrive at targets for the subsystems and subsequently for the components that comprise the system.

Models drive early validation

A second important role that models play is in validating the design to ensure that the requirements have indeed been met. Often these models are more detailed and provide higher fidelity information.

Models support trade-off decisions

The third important role for models is to enable the right trade-offs to be made between conflicting requirements. Continuing with our earlier example, the suspension bushings also need to satisfy requirements arising from vehicle handling and rough road impact considerations. The most effective way to consider multiple requirements and their trade-offs is through simulation.

Several hurdles must be overcome to make the process efficient. Siemens Digital Industries Software's approach to simulation is specifically designed to overcome these challenges and make engineers more productive by providing integrations, simpler access to geometry data, and a consistent data model:

Integration Integration with other simulation tools and with product development applications is important to both the efficiency and effectiveness of a model-based systems engineering solution. Integration is a core philosophy that permeates Siemens Digital Industries Software's simulation solutions.

Simplified geometry format Analysts often need to edit models but lack of expertise in knowing how to work with complex CAD geometry. Siemens Digital Industries Software addresses this dilemma by providing a data model that separates design models from analysis models while providing tools that enable analysts to modify geometry without having to master the intricacies of CAD technology. When the design is changed, analysis models and results can be updated automatically. This dramatically reduces the amount of time it takes to extract information, use the simulation, process results, and feed this information back into the design process.

Consistent data object model In model-based systems engineering, the true value of simulation is not limited to islands of automation but ripples downstream in the product lifecycle through re-use of the simulation results, as well as other CAE-derived knowledge. To facilitate this, Siemens Digital Industries Software provides the right integrated tools for your PLM environment – tools that enable you to capture, update, and share relevant engineering data and product knowledge across you entire global enterprise and supply chain.

Immersive user experience

The rich information that model-based systems engineering provides must be at the fingertips of every person involved in your product lifecycle. Information needs to be provided intuitively, in accordance with the context and situation of the engineer, focused on the business activity at hand and leveraging the geometric model:

With this in mind, Siemens Digital Industries Software's system-driven product development solution is:

Intuitive Users should feel comfortable interacting with the PLM-enabled environment. Our solution understands each user's role and presents information to the user commensurate with tasks associated with that role. This information is intuitive since it is consistent with the training and expectations of the user's role.

Context and situational sensitive When a user initiates any activity in a model-based systems engineering environment, our solution will identify a body of information as being critical to support that activity. In addition, as the activity progresses, further information can be identified as critical to the successful completion of the activity. The user's productivity is optimized as our software automatically manages the context that determines the scope of this information.

Activity focused When a user performs an activity, our solution will invoke business processes that provide context to its execution. In certain instances, this might be a higher level process that defines specific actions that will take place. In other cases, the process may actually execute the activity. Our solution anticipates the user's need for information and application usage.

Intelligent in terms of its geometry As a design matures to the point where actual product geometry is created, users must be able to use the geometry model to interact with and validate the system model in meaningful ways. Our solution provides geometric product models with a fully functional window into the interrelated functions and requirements that helped define that geometry. These models can also serve as a critical input to more advanced validations of the system's performance.

Conclusion

Siemens Digital Industries Software provides the foundation to
support system-driven product development throughout your enterprise including capabilities that enable you to:

• Capture the voice of the customer

• Support your design, manufacturing and service operations

• Facilitate enterprise program management, change and issue management and configuration management

• Ensure consistent data and business processes

Our open architecture is a necessary prerequisite to facilitating consistency in the real world – enabling you to integrate your processes across all engineering and manufacturing disciplines, leverage specialized design and simulation tools of choice and quickly adopt methods and rising innovations that nobody anticipates today but may become a future de facto standard.

Siemens Digital Industries Software

Headquarters
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

Americas
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

Europe
Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

Asia-Pacific
Unit 901-902, 9/F
Tower B, Manulife Financial Centre
223-231 Wai Yip Street, Kwun Tong
Kowloon, Hong Kong
+852 2230 3333

About Siemens Digital Industries Software
Siemens Digital Industries Software, a business unit of Siemens Digital Industries, is a leading global provider of software solutions to drive the digital transformation of industry, creating new opportunities for manufacturers to realize innovation. With headquarters in Plano, Texas, and over 140,000 customers worldwide, we work with companies of all sizes to transform the way ideas come to life, the way products are realized, and the way products and assets in operation are used and understood. For more information on our products and services, visit siemens.com/plm.

siemens.com/plm