J Intell Manuf (2013) 24:575–596
DOI 10.1007/s10845-011-0585-6

# SysML-based design chain information modeling for variety management in production reconfiguration

**Dazhong Wu · Linda L. Zhang · Roger J. Jiao · Roberto F. Lu**

Received: 2 September 2010 / Accepted: 12 August 2011 / Published online: 1 September 2011


**Abstract** Satisfying diverse customer needs leads to proliferation of product variants. It is imperative to model the coherence of functional, product and process varieties throughout the design chain. Based on a model-based systems engineering approach, this paper applies the Systems Modeling Language (SysML) to model design chain information. To support variety management decisions, the SysML-based information models are further implemented as a variety coding information system. A case study of switchgear enclosure production reconfiguration system demonstrates that SysML-based information modeling excels in conducting requirements, structural, behavioral and constraints analysis and in performing trade-off study. In addition, it maintains semantic coherence along the design chain, keeps traceability across different levels of abstraction, thus improving interoperability among heterogeneous tools.

**Keywords** Design chain management · Information modeling · SysML · Model-based systems engineering · Variety management · Production reconfiguration

D. Wu (✉) · R. J. Jiao
G.W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: dwu42@gatech.edu

L. L. Zhang
Department of Management, IESEG School of Management, Lille, France

R. F. Lu
Boeing Research & Technology, Seattle, WA, USA

## Introduction

A design chain underpins a series of activities that are coordinated based on heterogeneous information throughout the product realization process, including product planning, concept development, embodiment design, detail design, testing and refinement, and production planning. Consequently, as an integral part of product lifecycle management, Design Chain Management (DCM) has emerged as a systems engineering methodology, emphasizing the coherence of managing engineering and business functions across internal and external organizational collaborations (Ming et al. 2005; Zeng 2004). It aims at coordinating distributed participants in a design chain such that they can contribute to various capabilities necessary for product development in a manner that facilitates full-scale manufacture to commence (Twigg 2005). The available studies on DCM are mainly concerned with theories, methodologies, information platforms and managerial implications in the field of collaborative product development, such as the design chain operations reference model (Nyere 2006) and design chain partner selection (Akira 2001; Wang and Lin 2006).

As manufacturing enterprises become increasingly concerned with meeting the dynamic requirements of a global marketplace, it is paramount important to effectively and efficiently handle frequent design changes and process variations. This necessitates a systematic means of managing product variety that is conducive to customer satisfactions while leveraging legacy investment in design and production (Panchal et al. 2009). Due to the finite manufacturing resources existing on shop floors, manufacturers are confronted with difficulties in dealing with variety dilemma (Jiao et al. 2000; Ramdas 2003). The direct consequence of product customization on production is evidenced by exponentially increased process variety, resulting in production

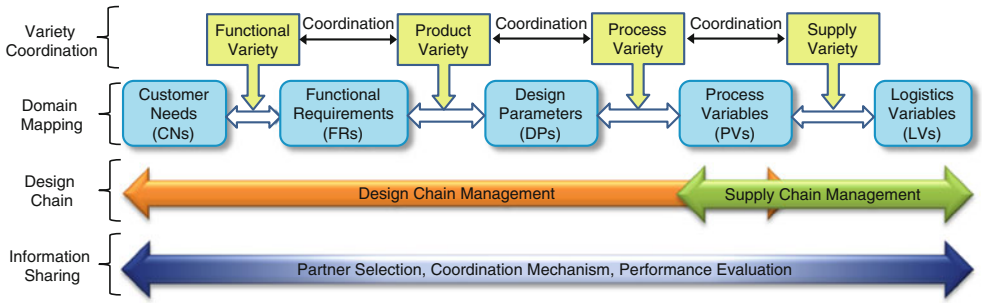


**Fig. 1** A holistic view of DCM

variations and consumption of extra resources on the shop floor. In this regard, product variety imposes higher production costs, including the incremental fixed investments on, e.g., machines, tools, fixtures, setups, cycle times, and labor, on a design chain (Jiao et al. 2007). To systematically justify a variety management strategy, it is important to explore how design changes propagate from design to production and to facilitate variety decision-making at different organizational levels (Ulrich et al. 1998; Ramdas 2003).

Successful variety decision-making requires both the ability to predict the impact of product design on production planning and the specific guidelines addressing the fundamental issue of increasing variety while reducing costs. Among many design for X methodologies, design for production (DFP) lends itself to be effective in integrating design with production, and in particular coordinating product and process varieties (PPV) (Chincholkar et al. 2003). In this paper, we focus on one important guideline for DFP: production reconfiguration (i.e., the configuration of processes from existing elements, such as operations, machines, and tools) (Salvador et al. 2009; Zhang et al. 2010). In production reconfiguration, the useful information, domain specific, and proven experiential knowledge of the past design and production is reused in new design problems, resulting in efficient and effective PPV management.

To implement production reconfiguration, decision support tools are required to compile useful information embedded in data and knowledge bases, to perform trade-off analysis with respect to product variety, production costs and performance, and ultimately to make better variety management decisions. In this regard, the key challenge of decision making support lies in the formal modeling and simulation techniques, which incorporate a production perspective to DCM (Huang et al. 2007; Johnson et al. 2008). By capitalizing on modeling and simulation, decision support to DFP is expected to synthesize design alternatives, evaluate production performance, select alternatives, and finally configure optimal design and production.

## A holistic view of DCM

Figure 1 presents a holistic view of DCM along the entire spectrum of product realization. It encompasses a series of domain mappings from customer needs (CNs) to functional requirements (FRs), to design parameters (DPs), to process variables (PVs), and to logistics variables (LVs) (Suh 2001). Different from supply chain management, which focuses on the PV and LV domains, DCM deals with issues associated with the first four domains ranging from CNs to PVs. While the CN-FR mapping results in specifications of functional variety, the mapping from FRs to DPs entails product variety to satisfy functional variety. The fulfillment of DPs is embodied as PVs, involving PPV coordination. Similarly, PV-LV mapping is associated with coordination of process and supply varieties. Moreover, information sharing in a design chain is important to the supporting activities, including partner selection, coordination mechanism and performance evaluation.

Design chain information modeling is very complex because of a series of multi-disciplinary activities and a collaborative distributed environment, as shown in Fig. 1. Consequently, it is imperative to analyze design chain systems by multi-view information modeling, such as requirements, structure, behaviors, and constraints (Shah et al. 2009). On the other hand, due to the lack of systematic approaches in dealing with frequent design changes and process variations, effective variety management becomes extremely important. In view of the above issues, in this study, we adopt a model-based systems engineering approach to develop an information model with the Systems Modeling Language (SysML), in attempting to manage PPV in production reconfiguration. First, we elaborate below technical challenges and the corresponding solutions.

## Technical challenges

1. *Management of system complexity*: Design chain information system is a large-scale, complex system, consisting of heterogeneous components such as hardware,

software, data, people, and facilities. It becomes increasingly complex when these components are handled by multi-disciplinary development teams. Competitive pressures demand that an information system must leverage technological advances to satisfy various customer needs, resulting in product variety at reduced costs (Friedenthal et al. 2008). In particular, this requires a systematic and multi-disciplinary approach that can manage system complexity.

2. *Semantic coherence*: In practice, product design and production planning information is managed in the forms of engineering documents, CAD drawings, bill of materials (BOMs), process routings, etc. Due to these different formats and the large volumes of data and information, a fundamental issue in information modeling is to ensure semantic coherence. In the context of DCM, this requires semantic mapping along the CN, FR, DP and PV domains. Such mapping involves not only semantics within each domain but also information content consistency across domains. However, the increasing product complexity makes it difficult to maintain information traceability across different domains, for example, keeping traceability across decision hierarchy when configuring BOMs and routings, and establishing the correspondence relationships between them. Moreover, different modeling languages demand interoperability among heterogeneous tools (e.g., CAD, CAE, discrete event simulation). Consequently, it underscores the importance in developing an information model using a formal language to ensure semantic coherence.
3. *Coordinated product and process varieties*: One of the key challenges in effective variety management arises from the traditional approaches to handling variety, where product variants are treated separately by specifying individual BOMs. Such approaches work well with a small number of variants, but not with a large number of variants (Jiao et al. 2000). The major reason is that the diverse product features and their various combinations result in not only large product variety but complex BOM structures. Moreover, due to recurrent process variations, how to configure process routings in an effective and efficient way suggests itself as an important issue to be addressed. Accordingly, variety management turns out to be an effective way to achieve production reconfiguration. However, in order to reconfigure production resources, it is essential to find a way that can coordinate PPV from both design and production perspectives.
4. *Design chain decision support*: Among various decisions, variety implementation decisions are very challenging. This is because these decisions involve both product and process varieties, their integration, management and creation. In this regard, decision support is necessary for identifying the impact of these decisions on production costs and performance in the early stage of a design chain. This requires the information system to conduct performance evaluation by establishing performance indicators (e.g., production rate, resource utilization), handling constraints and satisfying multiple objectives, such as production costs and lead time minimization (Grabot et al. 1996).

Strategy for solution

1. *Model-based systems engineering*: Increased system complexity demands rigorous and formalized systems engineering practice. Model-based systems engineering helps manage complexity and improve communications among multi-disciplinary development teams. As an enabler for model-based systems engineering, SysML is capable of representing requirements, structure, functions, and behaviors of complex systems. A SysML model can be used to conduct simulation studies by combining SysML modeling constructs with other execution environments, such as Modelica and Simscape (Johnson et al. 2007; Cao et al. 2010). To support applications that beyond software engineering, SysML reuses and extends the below subset of UML 2.1 constructs (Friedenthal et al. 2008): (1) *extending UML classes into blocks*; (2) *enabling requirements modeling*; (3) *supporting parametric modeling*; and (4) *extending UML standard ports to flow ports*. Through these extensions, a SysML model can represent many essential aspects of a system using the requirement, structure, behavior, and parametric diagrams. Figure 2 depicts the SysML diagram taxonomy.
2. *SysML for information modeling*: The semantics ensured by SysML enables a modeler to develop an integrated model where model elements in one diagram can be related to model elements in another diagrams. The use of XMI format enables SysML to be interchangeable between different software tools. Therefore, a SysML-based information model bridges the semantic gap between heterogeneous systems, and multiple disciplines as well. Moreover, the SysML-based information model allows changes to any artifacts to be traced between the requirements and specifications. It can improve the interoperability among various tools for requirements, structural, behavioral, and constraint analysis. A requirements model captures the functional, design, and process requirements and relationships among them. This is accomplished through *satisfy, verify, refine, derive, trace*, and *containment* relationships. A structural model is used to capture key elements of production reconfiguration with focus on generic vari-

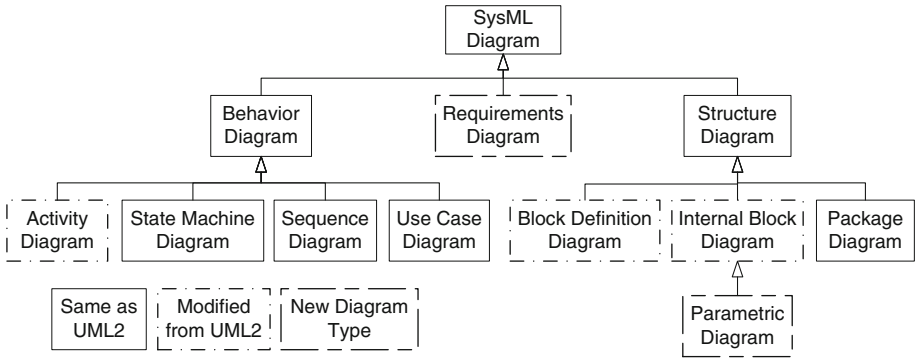


**Fig. 2** The SysML diagram taxonomy (OMG 2009)

ety planning (GVP) along with design and manufacturing feature mapping. A behavior model captures information flows along the CN, FR, DP, and PV domains. A constraint model can represent constraints on system properties. Each parametric diagram models a particular aspect of the system (e.g., cycle time, design and manufacturing costs, resource utilization). Multiple parametric diagrams can then be created to support trade-off analysis by combining with other execution environments.

3. *Generic variety planning* (*GVP*): Inherent in a family of customized products and the corresponding processes, there exist a common product structure and a common process structure (Jiao et al. 2007). Product and process differentiation is embodied in diverse variants of such common structures. Capitalizing on these common structures, we propose a concept of GVP to unify the functional, design, and production structures, in attempting to support PPV coordination. Viewed from an architecture perspective, GVP involves a generic variety structure (GVS) and GVS instantiation (i.e., generic planning). It models variety propagation from the DP domain to the PV domain. In line with the fact that variety features link DPs to PVs, a feature-based representation scheme is integrated into GVP to ensure product semantics.
4. *Variety coding information system*: Based on the SysML-based information model, a variety coding information system is developed to implement production reconfiguration. Underpinning the coding information system, a variety coding methodology accommodates PPV coordination based on variety mapping between the DP and PV domains. On the other hand, variety implementation decision support is driven by a data, knowledge and model repository. This repository is built on a coding scheme and a coding database. A three-layer mapping mechanism across the planning, feature, and coding layers further helps keep traceability across different levels of abstraction.

In the next section, we present the related work on DCM, information models, modeling languages, and variety management. Section "Application case" introduces an application case of switchgear enclosure production reconfiguration. The GVP is discussed in "Generic variety planning" to coordinate functional, product, and process varieties of switchgear enclosures. Section "A SysML model for production reconfiguration" presents the SysML-based information model for the switchgear enclosure production reconfiguration. In "Variety coding information system", a variety coding information system is developed based on the SysML-based information model to demonstrate variety management. We conclude the paper in "Concluding remarks" with an outline of potential avenues for future research.

## Related work

### Design chain management

A design chain is referred to as the relationships between a product assembler and his part suppliers (Clark, 1988). It involves concept design, detail engineering, process engineering, and prototype manufacturing (Twigg 1998). Poirier and Reiter (1996) extend this chain concept to a product development chain, which involves product assemblers, suppliers and customers. Liu and Zeng (2010) formalize DCM using the environment-based design theory. Wu et al. (2007) establish a universally applicable collaborative design reference model and apply it to the motorcycle industry. Nagarajan et al. (2004) study design chain partner selection problem for co-development. IBM (2007) proposes a product development integration and service oriented architecture, where DCM enables the business-centric view of an entire enterprise that orchestrates various product and process functions. As one of the critical issues associated with DCM, capturing the full range of engineering information commonly shared in product development is very important.

This calls for a multi-view information model that allows for the exchange of information generated in various product development activities within a single company, or even across companies in a heterogeneous environment (Sudarsan et al. 2005).

### Information models

Many formation models associated with product design and manufacturing are proposed to address various issues based on different modeling tools. Candadai et al. (1996) discuss a STEP-based product information model for exchange of production information between designers and manufacturing personnel. Recognizing the inefficiency of the existing approaches towards computer aided conceptual design, Brunetti and Golob (2000) develop a feature-based product model to capture product semantics handled in the conceptual design phase. Zha and Sriram (2006) propose a knowledge-intensive support framework to assist platform-based product design with focus on customer requirements modeling, product architecture modeling, product platform establishment and product variant assessment. Sudarsan et al. (2005) propose a unified view of the NIST core product model and open assembly model in UML to support tolerance representation and propagation, representation of kinematics, and engineering analysis at the system level.

Since the 1990's, various ontology-based methods have been developed to represent knowledge and to generate inter-ontology mapping. Cho et al. (2006) propose meta-concepts for ontology developers to consistently identify domain concepts of parts libraries and to systematically structure them. Lin et al. (2004) propose a manufacturing system engineering ontology to enhance the semantic interoperability and reuse of knowledge resources. Li and Ramani (2007) propose to use shallow natural language processing and domain-specific design ontology for design information retrieval. Fiorentini et al. (2008) analyze the requirements for the development of structured knowledge representation models for manufacturing products using ontology. Lim et al. (2010) propose a methodology for building a semantically annotated multi-faceted ontology for product family modeling.

As a general-purpose modeling language, SysML has recently received increased attention. Researchers have approached SysML from different perspectives to address different issues. For instance, Johnson et al. (2007) put forward a formal approach to model dynamic system behaviors in SysML by means of a language mapping between SysML and Modelica. Peak et al. (2007) introduce SysML parametric concepts to demonstrate how SysML captures engineering knowledge in a reusable form. Huang et al. (2007) explore the use of SysML to model a system and to support the automatic generation of simulation models.

### Modeling languages

Information models can be developed based on a number of modeling languages, such as IDEF, EXPRESS, UML, and SysML. IDEF is a family of modeling methods, each of which addresses a unique aspect of the system under consideration. When individual IDEF methods are combined together, they incorporate more product life cycle concerns early in the design process. The IDEF family is designed to promote the integration in such environments, where optimal results depend on the effective use of enterprise information and knowledge assets (Mayer et al. 1995). EXPRESS is a modeling language for specifying information requirements of product data. It is formalized in the ISO standard for the exchange of product model. The main feature of EXPRESS is the possibility to formally validate a population of data types. Moreover, it can be used to explicitly describe complex constraints, data structures and relationships for engineering activities (Loffredo 1998). UML is a standardized modeling language for creating visual models of software systems and for automatically generating code. With their unique features, IDEF, EXPRESS, and UML all can be used to develop conceptual information models (Lee 1999).

SysML is a UML profile which allows modeling systems from a domain neutral perspective (OMG 2009). The advantages of SysML over UML is that SysML allows modelers to use requirement diagrams to efficiently capture functional and performance requirements, along with their relationships, whereas with UML, modelers are subject to the limitations of use case diagrams to define high-level functional requirements (Friedenthal et al. 2008). With SysML, parametric diagrams can be used to precisely define performance and quantitative constraints, whereas UML provides no straightforward means to capture such information. In this paper, we focus on SysML-based information modeling to address variety management in production reconfiguration.

### Variety management

Simply increasing variety does not guarantee an increase in profits. On the contrary, it jeopardizes a manufacturing firm's competitiveness. Thus, variety management suggests itself to be an important dimension of successful business practice (Ramdas 2003). Olavson and Fry (2006) describe variety management principles and process to help companies better understand variety and make optimal decisions. Zipkin (2001) analyzes the opportunities to implement high variety strategy ensuring that customers are not confused with the complexity inherent in mass customization. Ramdas et al. (2010) propose a methodology to help managers navigate the complex decisions of product differentiation.

To address variety management from the engineering perspective, van Veen and Wortmann (1992) proposes a concept of generic BOM. Essentially, a generic BOM organizes all design data pertaining to a product family as a single structure using generic representation. In dealing with the interconnections between products and processes, most of the traditional approaches treat product and process variants separately by developing individual BOMs and routings. Ulrich (1995) defines product architecture and discusses its impact to cope with product variety. Robertson and Ulrich (1998) further define the product platform concept for reducing product variety complexity. Incorporating production data in the generic BOM, Jiao et al. (2000) put forward generic bills-of-materials-and-operations to accommodate management of large numbers of product and process variants. Martin and Ishii (2002) describe a design for variety methodology to aid in developing a product platform architecture that incorporates standardization and modularization. Jiao et al. (2007) present a holistic view of variety management geared towards the derivation of product and process variants with given customer requirements, including generic variety representation, generic product and process structures, and generic planning.

## Application case

In this paper, switchgear enclosures are adopted as a running example to demonstrate the success of a SysML-based information model in variety management for production reconfiguration. To survive the intensive global business competition, the case company struggles to produce diverse switchgear enclosure variants while maintaining profitable costs. This requires an information system that can provide decision support to variety management in switchgear enclosure production reconfiguration.

As a typical sheet metal product, a switchgear enclosure consists of a number of components, including assemblies, subassemblies and basic components at different levels of the product hierarchy. Figure 3 shows some key components of a switchgear enclosure (e.g., basis frame, end plates, floor plates, roof plates, rear assembly). Built upon the common product and process structures inherent in the family of switchgear enclosures, GVP unifies the functional, design, and production structures of switchgear enclosures. Accordingly, it is applied to coordinate the large variety of switchgear enclosures and corresponding manufacturing

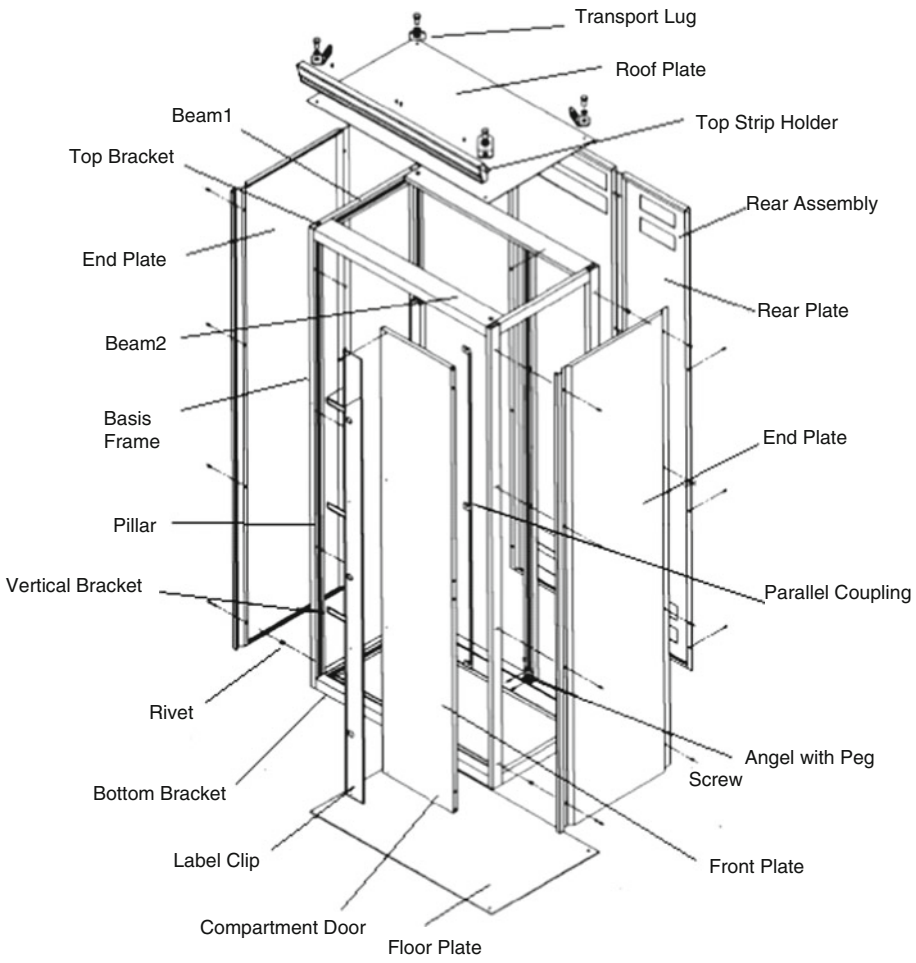


**Fig. 3** A switchgear enclosure and its key components

processes by linking DPs to PVs with common variety feature information. The application of GVP to switchgear enclosure production reconfiguration is presented in "Generic variety planning".

In managing system complexity, a SysML-based information model is developed to perform requirements, structural, behavioral, and constraint analysis. It captures the structure of the switchgear production reconfiguration system, specifies requirements from different stakeholders (e.g., customers, designers, process engineers), analyzes the constraints on physical and performance properties (e.g., production rate, resource utilization, manufacturing costs), evaluates alternatives, and ultimately provides decision support to meet costs and performance objectives. The SysML-based information model for switchgear enclosure production reconfiguration is discussed in " A SysML model for production reconfiguration".

With the SysML-based information model, a prototypical information system is further developed to demonstrate its effectiveness in switchgear enclosure production reconfiguration. Underpinning the variety coding information system, the variety coding methodology includes a coding scheme and a code database. In addition, a frame-based knowledge base is developed to coordinate product design and manufacturing process knowledge. The prototypical information system for switchgear enclosure production reconfiguration is introduced in "Variety coding information system".

## Generic variety planning

As the underlying mechanism to coordinate the functional, product, and process varieties, GVP involves a static GVS and its dynamic instantiation in generic planning. The GVS includes a generic function structure (GFS), a generic design structure (GDS), and a generic production structure (GPS). Underpinned by the GVS, generic planning entails the derivation of variants by instantiating parameters organized in the GVS. GVP assists designers to better understand the impact of variety features on PPV and variety propagation from design to production.

### Generic function structure

A product is designed to achieve a certain system overall function. The delivery of such a system overall function is normally accomplished through the realization of a number of child functions (Alblas et al. 2010). In turn, these child functions have their own child functions along with their relationships. The decomposition of the overall function into child functions at different levels forms a GFS pertaining to a product family, as shown in Fig. 4a. Each node corresponds to a generic function, representing a number of function variants of the same type. The large number of valid combination of function variants at different levels of the hierarchy gives rise to diverse product variants.

### Generic design structure

As with the GFS underlying a product family from a functional perspective, the GDS embodies a product family from a design perspective. The GDS in relation to a product family is characterized by (1) *product structure*, (2) *variety feature*, and (3) *configuration constraint*.

1. *Product structure*: Product structure refers to the hierarchy of the product-assembly-subassemblies-parts tree (BOM) and the associated individual parts, subassemblies, and assemblies that constitute the hierarchy, as shown in Fig. 4b. All variants of a product family share such a common tree structure which reveals the topology for end-product configuration.
2. *Variety feature*: A variety feature is an attribute associated with an artifact that has a specific function assigned to it. Different instances of a particular variety feature represent the diversity of variety feature variants, thus creating product or process variety. Five fundamental variety design features are: (a) *color*, (b) *material*, (c) *thickness*, (d) *number of components* and (e) *structure configuration*. Since each component of a switchgear enclosure can be offered in similar geometry, geometry is not considered as a variety design feature in this study.
3. *Configuration constraint*: A constraint is a specific shared property of a set of entities that must hold in all cases. An example of a constraint is as follows: Certain values of one variety feature are only compatible with some values of another variety feature. Such restrictions form the bases of configuration constraints, guiding the valid combination of variety feature values.

For the family of switchgear enclosures in the application case, the GDS has been constructed, as shown in Fig. 5. Each node in the GDS represents a generic item; the number above each node denotes the quantity per of the represented generic item in one generic parent item (e.g., 12 rivets are necessary in a rear assembly). Take a basis frame as an example. Table 1 gives the detailed product data, such as variety features and feature value set. As an immediate child component of a switchgear enclosure, the basis frame has its own child components, such as top bracket, bottom bracket, and vertical bracket (Note, as standard parts, screws are not given in Table 1). Each child component is characterized by a number of parameters, each of which assumes a number of value instances given in Table 1. Also given in Table 1 are some examples of constraints among parameters.

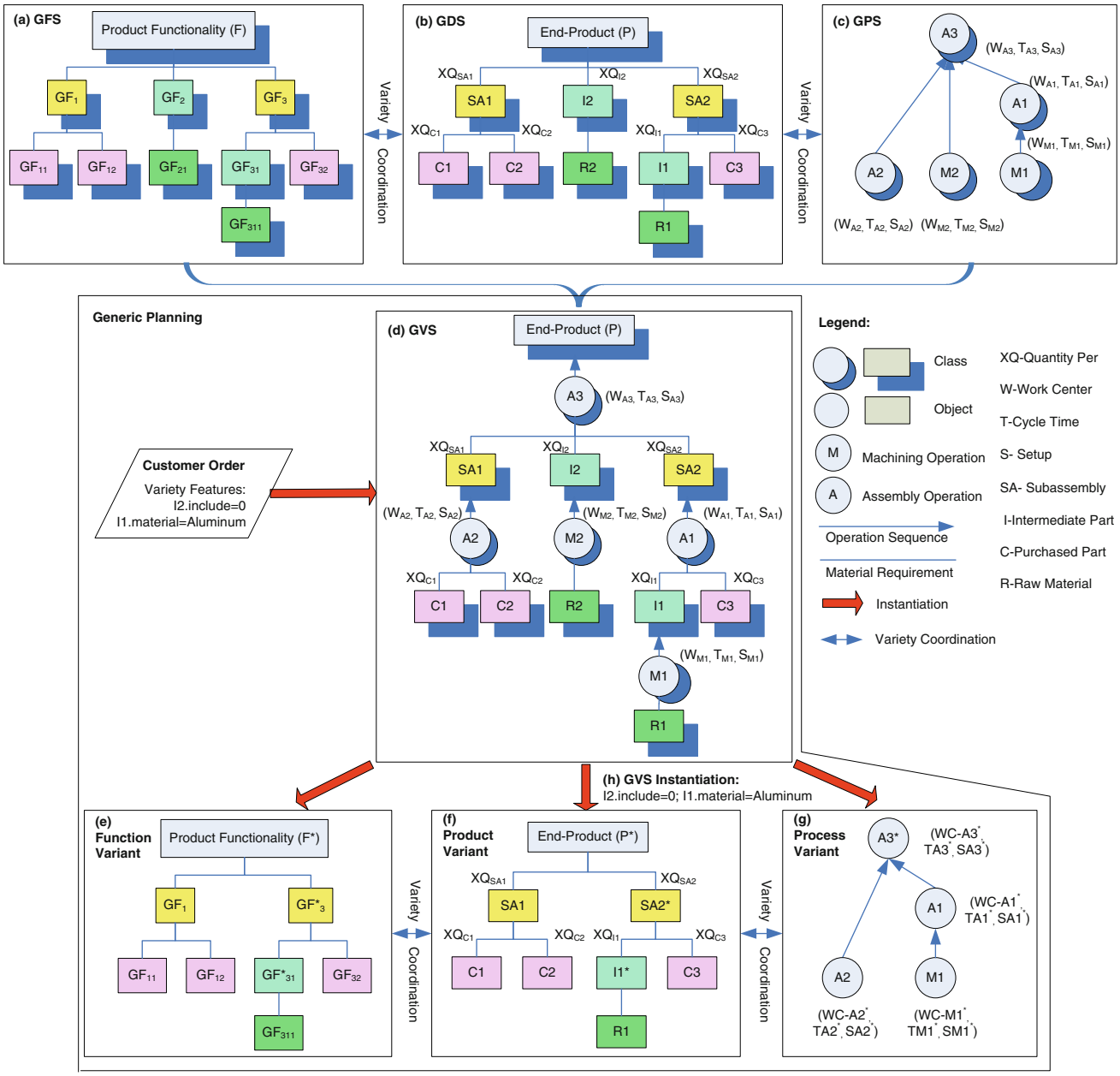


**Fig. 4** Genetic variety planning

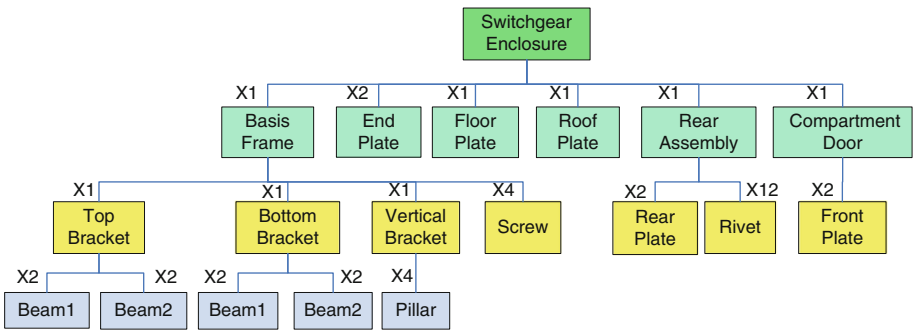


**Fig. 5** The GDS of the switchgear enclosure family

**Table 1** An example of generic items and variety feature for switchgear enclosures

| Generic item | Variety feature | Feature value set |
|---|---|---|
| Basis frame | Configuration | 1/2/3 compartment(s) |
| | Color | White; Grey; Blue; Black |
| | Material | Aluminum; Steel; Copper |
| | Thickness | 1/2/4 mm |
| Top bracket | Color | White; Grey; Blue; Black |
| | Material | Aluminum; Steel; Copper |
| | Thickness | 1/2/4 mm |
| Bottom bracket | Color | White; Grey; Blue; Black |
| | Material | Aluminum; Steel; Copper |
| | Thickness | 1/2/4 mm |
| Vertical bracket | Color | White; Grey; Blue; Black |
| | Material | Aluminum; Steel; Copper |
| | Thickness | 1/2/4 mm |

| Constraint # | Constraints fields | Constraint type |
|---|---|---|
| 1 | TopBracket.Material | Material compatible |
| | BottomBracket.Material | |
| | VerticalBracket.Material | |
| 2 | TopBracket.Color | Color compatible |
| | BottomBracket.Color | |
| | VerticalBracket.Color | |

### Generic production structure

The design similarity inherent in a product family leads to process similarity among the routings to produce the corresponding product family members. Such process similarity is embodied by the similar process elements, such as operations, operation precedence and manufacturing resources. Resulting from the process similarity, the routings to produce a product family can be organized as the GPS, as shown in Fig. 4c. In the GPS, Each node denotes a generic operation, be it a manufacturing or assembly type. It is described by a generic work center, setup and cycle time. The various combinations of specific work centers, cycle times and setups lead to operation variants necessary to produce a given product item.

Similarly, the GPS for the process family of switchgear enclosures has been constructed, as shown in Fig. 6. Each node in the GPS represents a generic operation. For example, to produce Top Bracket in Fig. 5, an assembly operation, $A_{TB}$, and two manufacturing operations, $M_{B1}$ and $M_{B2}$, are necessary. Table 2 gives the details of the generic operations involved in producing Basis Frame.

### Generic planning

Generic planning is introduced to determine specific function, product, and process variants through instantiation of the GVS, as shown in Fig. 4d–g. It involves two steps. In the first step, the GFS, GDS, and GPS are unified into a single generic structure, the GVS in Fig. 4d. A single-level GVS is first derived by specifying the sequence of operations required for producing an item in conjunction with materials and resources. Subsequently, the multi-level GVS is obtained by linking the single-level GVS of lower-level intermediate components through the operations that require them. The second step is to instantiate the GVS with respect to the given values of particular variety features specified in a customer order, as shown in Fig. 4h. The instantiation results in the required product variant, the corresponding process variant (i.e., routing) and the hierarchy of the specific functions involved in the product variant, as shown in Fig. 4e–g.

## A SysML model for production reconfiguration

The development of the information model for the production reconfiguration system poses many challenges, such as

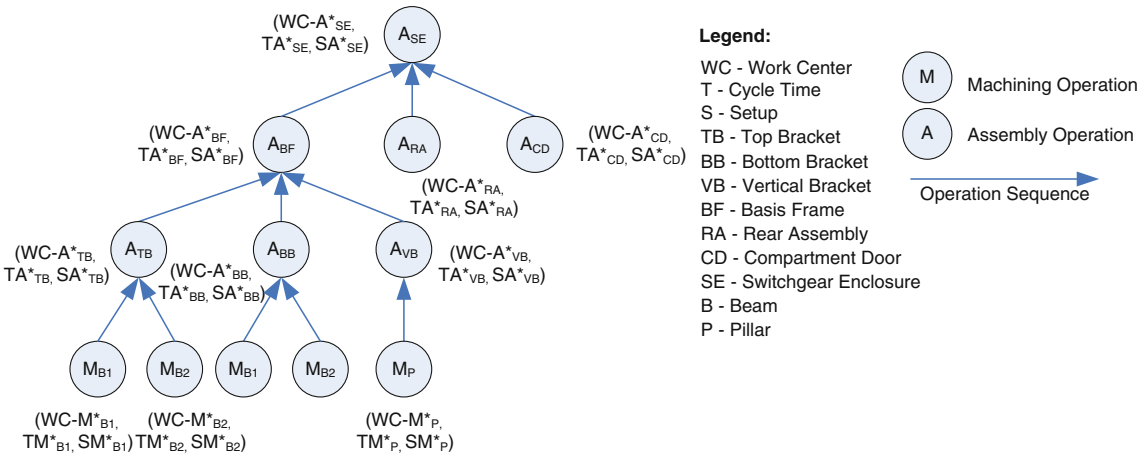


**Fig. 6** The GPS of the switchgear enclosure family

**Table 2** Generic operations for "Basis Frame" assembly

| Generic operation | Generic item | Variety feature | Part | Work center | Cycle time | Fixture/ setup |
|---|---|---|---|---|---|---|
| Top bracket fabrication 1 | Top bracket | Color | White/Grey/Blue/Black | Painting machine | 10 | Fixture 1 |
| | | Material | Aluminum/Steel/Copper | | | |
| | | Thickness | 1/2/4 mm | Rolling machine | 20 | |
| Bottom bracket fabrication 2 | Bottom bracket | Color | White/Grey/Blue/Black | Painting machine | 12 | Fixture 2 |
| | | Material | Aluminum/Steel/Copper | | | |
| | | Thickness | 1/2/4 mm | Rolling machine | 22 | |
| Vertical bracket fabrication 3 | Vertical bracket | Color | White/Grey/Blue/Black | Painting machine | 20 | Fixture 3 |
| | | Material | Aluminum/Steel/Copper | | | |
| | | Thickness | 1/2/4 mm | Rolling machine | 30 | |

The unit of measure for cycle time is minute

encoding system complexity, ensuring semantics, capturing variety handling framework, and representing decision support process, as discussed in "A holistic view of DCM". Due to the system complexity, the information model involves many elements. For clarity, definitions for some key elements that constitute the information model are listed in the nomenclature.

As pointed out by Friedenthal et al. (2008), modeling requirements, structure, behaviors, and constraints can provide a robust system description. Therefore, the SysML-based information model for production reconfiguration consists of (1) *requirements analysis*, (2) *structural analysis*, (3) *behavioral analysis*, and (4) *constraint analysis*.

Requirements analysis depicts the hierarchy of requirements and interrelationships among requirements. The feature-based GVP structural model captures the key elements of production reconfiguration system and variety features pertaining to design and production. The behavior model captures the flow of information along multiple domains. Constraint analysis facilitates the simulation of production performance and costs and performs verification and validation against system requirements. The behavior model of decision support process is developed to integrate requirements, structural, and constraint analyses.

## Requirements analysis

A requirement specifies a condition that must be satisfied, a function that a system must perform, or a performance level that a system must achieve. Requirements come from many sources, such as customers, designers, process engineers, or organizations. As an example, Fig. 7 shows a requirement diagram of production reconfiguration system. This example highlights a number of different requirements relationships, including *satisfy, verify, refine, derive, trace*, and *containment*.

For example, the *satisfy* relationship is used to assert that *variety decisions satisfy* the requirements of *minimum production costs* and *minimum lead time*. The *verify* relationship is used to *verify* that the requirements of *minimum production costs* and *minimum lead time* are satisfied by a test case called *simulation for production*. The test case, *simulation for production*, represents a method for performing the verification. The *refine* relationship reduces ambiguity in the requirement of *variety decision* by relating it to two elements: *product variety* and *process variety*, which clarify the requirement. Similarly, it breaks down the requirement of *minimum lead time* to the requirements of *minimum setup time* and *minimum cycle time*, and *minimum production costs* to *minimum fixed costs* and *minimum variable costs*. The *derive* relationship between a source requirement, *customer needs*, and the derived requirements, *minimum production costs* and *minimum lead time*, is established based on marketing analysis. Similarly, the requirement of *production reconfiguration* is derived from the requirements of *minimum production costs* and *minimum lead time*, based on production reconfiguration principle. The *trace* relationship provides a general-purpose relationship between the requirements of *BOM* and *process routing*. The *trace* relationship is relatively weak compared with other relationships. However, it is useful for relating lower-level requirements to higher-level requirements in the *BOM* and *process routing* hierarchy, respectively. For example, the requirements specification can be traced from the end product, to the assembly, and to the part requirements. Moreover, it is useful for establishing a relationship between *BOM* specifications and *process routing* specifications based on the mapping between design and manufacturing features. The *containment* relationship represents how complex production reconfiguration requirements can be partitioned into a set of lower-level requirements. For instance, the requirements of *production reconfiguration* contain two requirements associated with *process routing* and *BOM*.

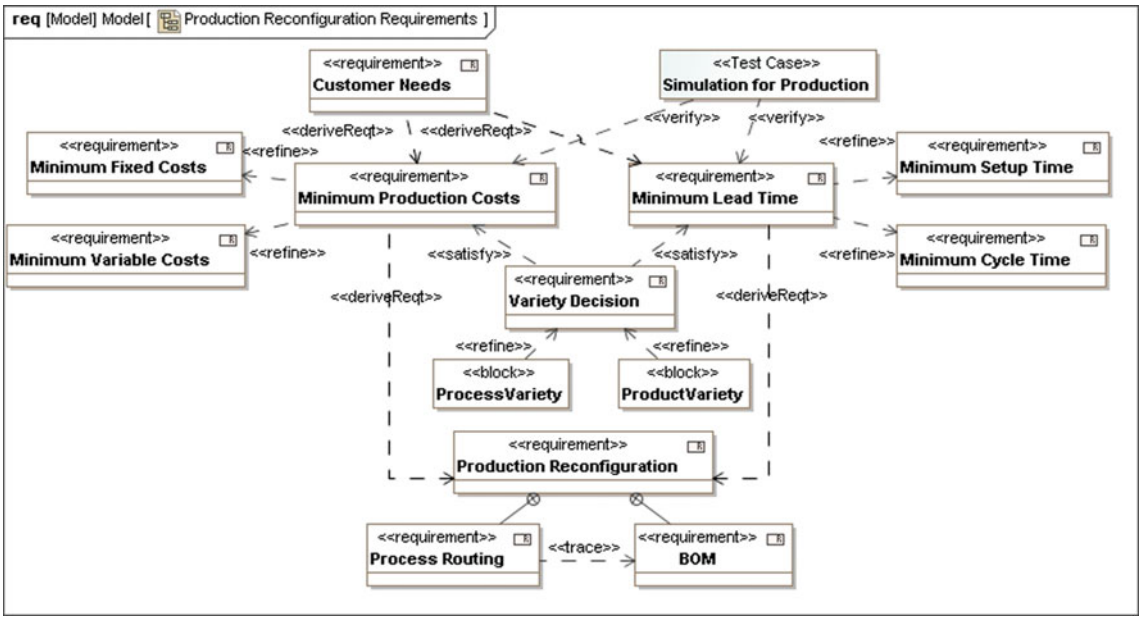


**Fig. 7** SysML requirement diagram for production reconfiguration

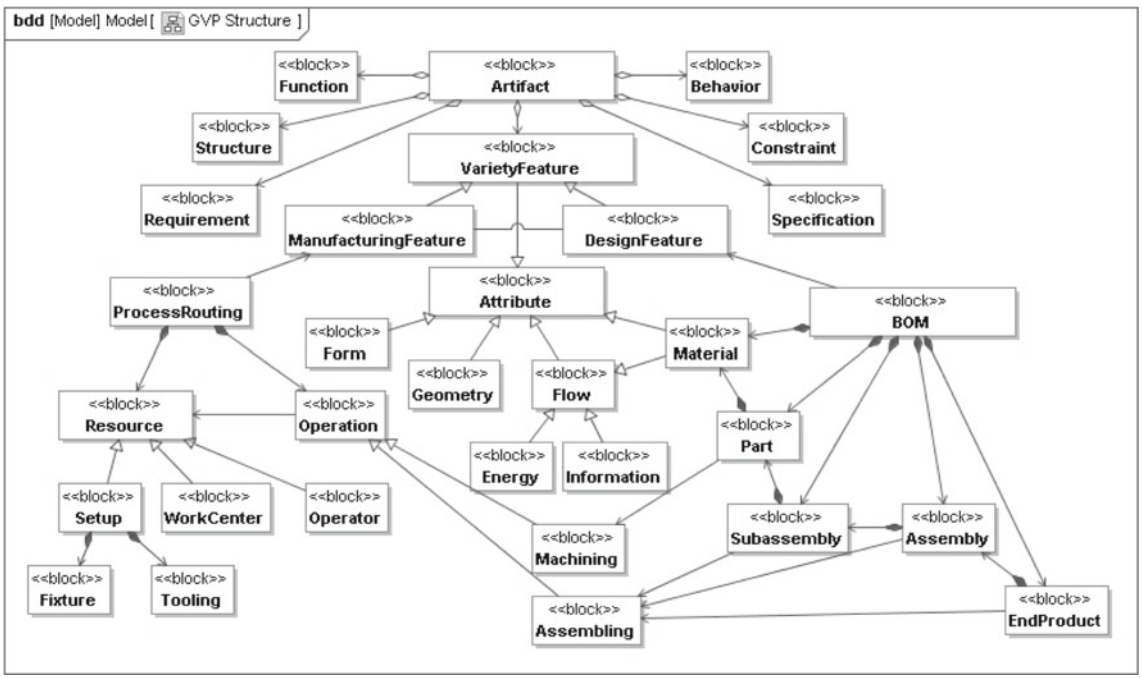


**Fig. 8** SysML block definition diagram for GVP structural model

Generic variety structural modeling

To achieve semantics and interoperability, our approach for modeling GVP extends the NIST UML-based core product model to capture the complexity of variety management for production reconfiguration. Since production reconfiguration involves both design and process planning, the information model for representing GVP is very complex. This model incorporates the basic information elements of the core product model, and extends it by combining product model with its process model. The process model contains information about manufacturing process.

Figure 8 shows the GVP structure using a block definition diagram. In the context of production system, a *block* in SysML can represent any entity (e.g., *part, function, operation*, or *process routing*). By including a number of blocks,

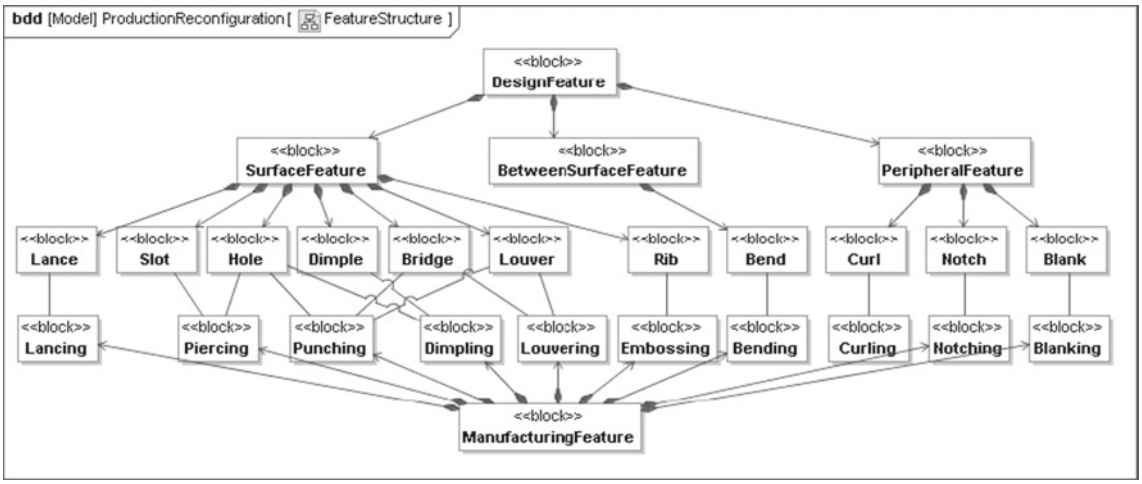


**Fig. 9** SysML block definition diagram for sheet metal feature mapping

a block definition diagram can define the hierarchy of the system of interest in terms of its structural feature and the relationships between the blocks. For example, *artifact* represents a distinct entity in an *end product*. Such an entity can be a *requirement, structure, function, behavior, constraint, specification* or *variety feature*. The more general classifier of the *variety feature* is the *attribute* associated with an *artifact*. The *attribute* represents a generalization of *form, geometry, flow*, and *material*. The *flow* indicates the flow of *energy, information*, or *material*. The *design feature* and *manufacturing feature* are more specific elements of the *variety feature*. The *BOM* and *process routing* are associated with the respective *design feature* and *manufacturing feature* by sharing the same *variety feature*. In particular, the *BOM* for a product variant is an instance of a particular set of *design features*. Similarly, a *process routing* for a process variant is an instance of a particular set of *manufacturing features*. The *BOM* consists of *material, part, assembly, subassembly*, and *end product*. The *process routing* is comprised of *resource* and *operation*. The *operation* can be specialized into *machining operation* or *assembling operation*, and is characterized by *resource*. The *resource* is a generalization of *setup, work center*, and *operator*. The *setup* is comprised of *fixture* and *tooling*. With *machining operations, material* can be transformed into *part*. With *assembling operation, part* can be assembled into *subassembly* or *assembly*; *end product* can be produced from *assembly*.

As a key element of the GVP structural model, *variety features* are the information carriers, linking *design features* with *manufacturing features*, thus facilitating design and production coordination. The typical sheet metal design and manufacturing features along with the feature mapping between them are captured using a block definition diagram, as shown in Fig. 9. *Design features* consist of *surface feature, between surface feature*, and *peripheral feature*. The *surface feature* is comprised of *lance, slot, hole, dimple, bridge, louver*, and *rib*. *Bend* is a *between surface feature*. *Peripheral feature* contains *curl, notch*, and *blank*. Associated with each *design feature*, there are one or more *manufacturing features* to produce it. For example, *Piercing* can produce *slot* and *hole*. *Punching* can fabricate *hole, bridge*, and *louver*. Likewise, *hole* and *dimple* can be made by *dimpling*, etc. Consequently, *design features* and corresponding *manufacturing features* are coordinated by sharing the same *variety feature* information (i.e., shape and parametric information).

## Behavior modeling

While a structural model captures the system hierarchy and relationships among its internal elements, it cannot model the flows of inputs and outputs through a sequence of actions, which entails behavior modeling. An activity diagram in SysML can capture the flows of information, energy, and material, and represent a set of actions describing activity execution and input transformation. Figure 10 shows an activity diagram of the GVP behavior model, capturing the information flow along *CN, FR, DP*, and *PV* domains. It also shows explicit allocation of behavior partitions, representing which parts are responsible for executing which activities. The *CNs* are elicited from *customers*, and subsequently transformed to *FRs* by *market analysts*. The *FRs* are transformed to *DPs* in the form of *BOMs* by *design engineers* when developing detailed design. The *DPs* and *BOMs* are then transformed to *PVs* and *routings* by *process engineers*. Ultimately, *PVs* and *routings* are transformed to *production plans* by *production engineers* through reconfiguring production. As the GVP behavior model captures the common principal actions within product realization process, it can be reusable for other particular design scenarios.

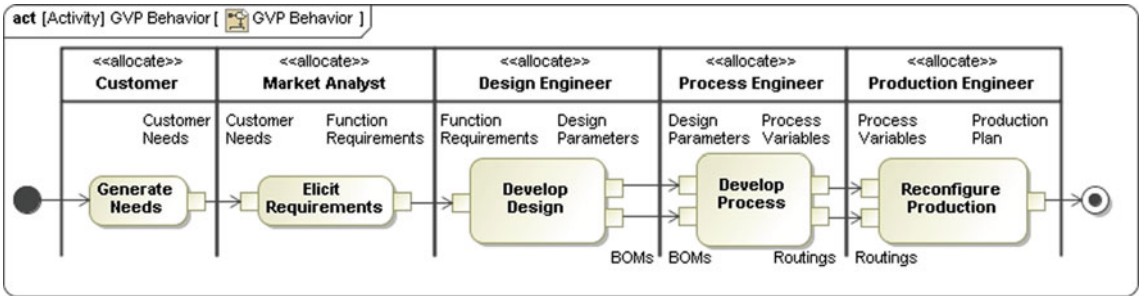


**Fig. 10** SysML activity diagram for GVP behavior model

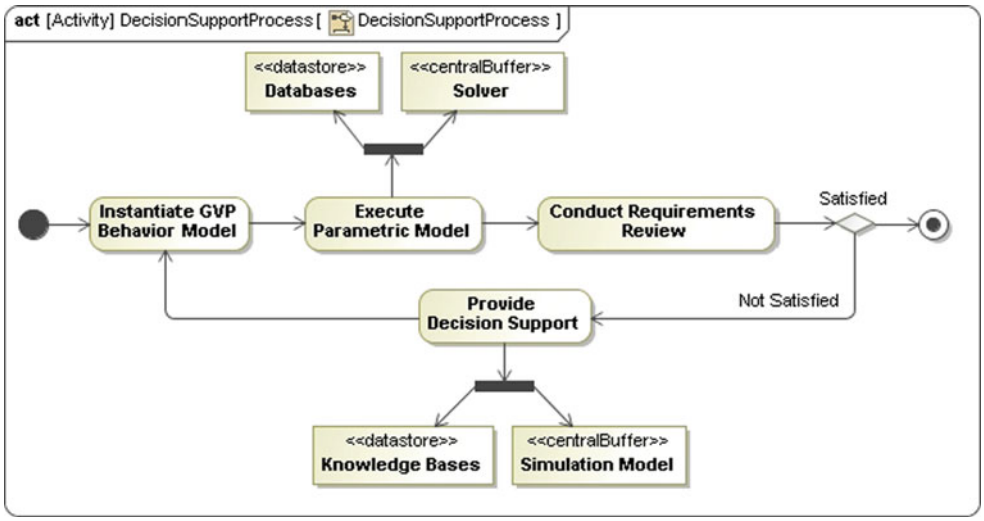


**Fig. 11** SysML activity diagram for decision support process

Apart from the GVP behavior model, a decision support behavior model is developed to capture how decisions are made. As shown in Fig. 11, the starting point is to *instantiate the GVP behavior model*. A *parametric model* (discussed in "Constraint analysis") is executed with *databases* and a *solver*. The *solver* supports the *parametric model* execution. Subsequently, a *requirements review* is conducted to evaluate if the requirements are satisfied. If not, the guidance on variety implementation in production reconfiguration is then provided based on *knowledge bases* and a *simulation model*. After several iterations, when the variety creation and cost objectives are both achieved through trade-off analysis, the decision support process ceases.

## Constraint analysis

As shown in Fig. 11, constraint analysis is necessary to provide decision support. With constraint blocks, a parametric diagram can be developed to model connections between constraint parameters and value properties of the production reconfiguration system. The parametric diagram binds the relevant properties of the block and parameters of the analysis model. Modeled as mathematical equations, constraints in the parametric diagram can be solved by solvers (e.g., Mathematica, Matlab) for analysis. In addition, quantitative requirements can also be formulated as constraint statements (e.g., If dimple_weight < maximum_weight, Then result = true, Else result = false) and incorporated in parametric diagrams. When parametric simulation is executed, the constraint statement verifies whether or not the requirements are met.

As shown in Fig. 12, a block definition diagram of *production performance* model with constraint blocks represents the *production performance* model structure. It involves *production rate, production capacity, resource utilization, lead time, manufacturing costs* and *constraints*.

Figure 13 represents the interrelationships of the *constraints* in the block definition diagram by binding the relevant parameters. For example, the *cycle time* consists of *actual machining operation time, workpart handling time*, and *tool handling time* as follows:

$$T_c = T_o + T_h + T_{th}, \tag{1}$$

where $T_c$, $T_o$, $T_h$ and $T_{th}$ denote the *cycle time, actual machining operation time, workpart handling time*, and *tool handling time*, respectively.

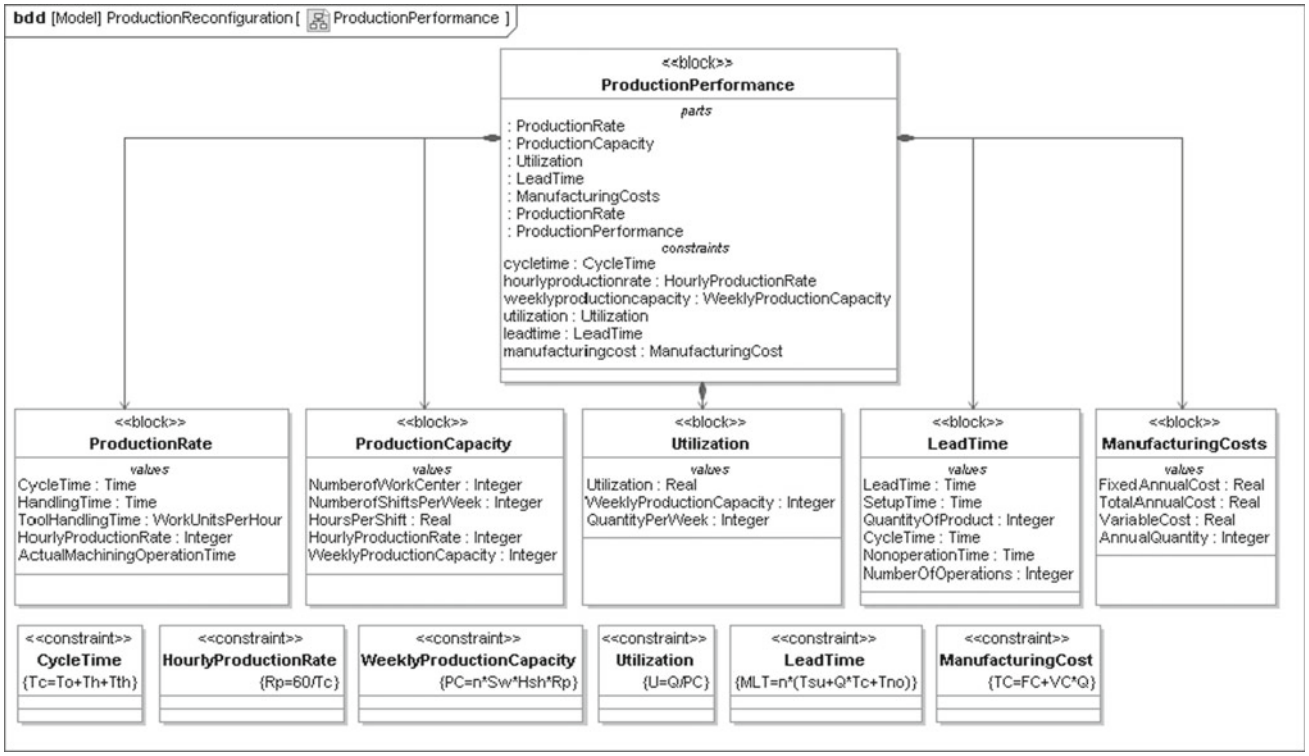


**Fig. 12** SysML block definition diagram for production performance model

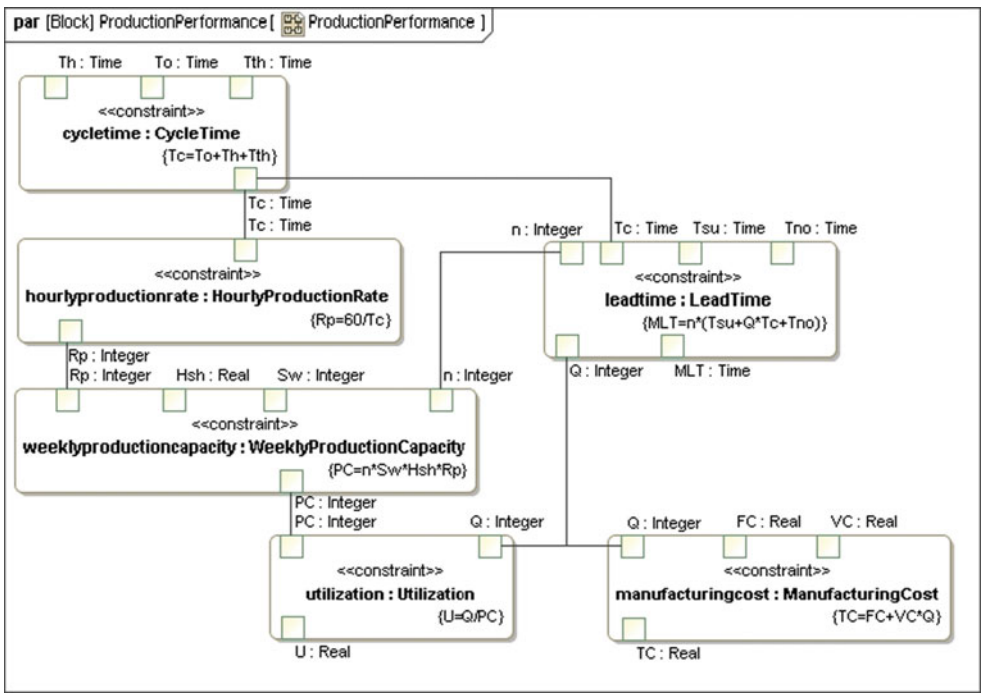


**Fig. 13** SysML parametric diagram for production performance model

With the *cycle time, hourly production rate* is approximated as:

$$R_p = 60/T_c, \tag{2}$$

where $R_p$ denotes *hourly production rate* (the constant 60 converts minutes to hours).

*Weekly production capacity* is defined as the maximum output rate of a production facility per week as follows:

$$PC = nS_w H_{sh} R_p, \tag{3}$$

where $PC$, $n$, $S_w$, and $H_{sh}$ denote the *production capacity, number of machines, number of shifts per week*, and *number of hours per shift*, respectively.

*Utilization* refers to the ratio between the output of a production facility and its capacity as follows:

$$U = Q/PC, \tag{4}$$

where $U$ and $Q$ denote the utilization of the facility and the quantity produced by the facility.

*Lead time* is defined as the total time required to process a given part or product through the plant. In this paper, it is assumed that *setup times, operation cycle times*, and *nonoperation times* are all equal. Further, the batch quantities of all parts or products processed through the plant are assumed to be equal; and they are all processed through the same number of machines. With the above assumptions, the *lead time* is approximated as follows:

$$MLT = n(T_{su} + QT_c + T_{no}), \tag{5}$$

where $MLT$, $T_{su}$, and $T_{no}$ denote the *manufacturing lead time, setup time*, and *nonoperation time*.

The total *manufacturing costs* can be calculated based on the equation below:

$$TC = FC + VC(Q), \tag{6}$$

where $TC$, $FC$, and $VC$ denote the total, fixed, and variable costs.

## Variety coding information system

Built upon the SysML-based multi-view information model, the variety coding information system transforms requirements to design alternatives and production solutions through the structural and behavior models while addressing constraints in parametrics. It has a user interface separating the system from its environment. The behavior of the information system provides variety implementation decision support when it receives new customer orders. Underpinning the variety coding information system, the variety coding methodology includes a coding scheme and a code database. A frame-based knowledge base organizes design and manufacturing feature knowledge.

### Coding methodology

The rationale of variety coding methodology lies in the variety feature mapping between PPV, coordinated by GVP. It attempts to link design data to production data using consistent code. As shown in Fig. 14, the mapping between PPV is captured in a three-layer model, including the planning, feature, and coding layers, which represents different levels of abstraction. At each layer, all the items in design and process feature domains are connected with each other by correspondence relationships. Within each domain of the same layer, the individual items are interacted with each other by composition relationships. In addition, the items within each

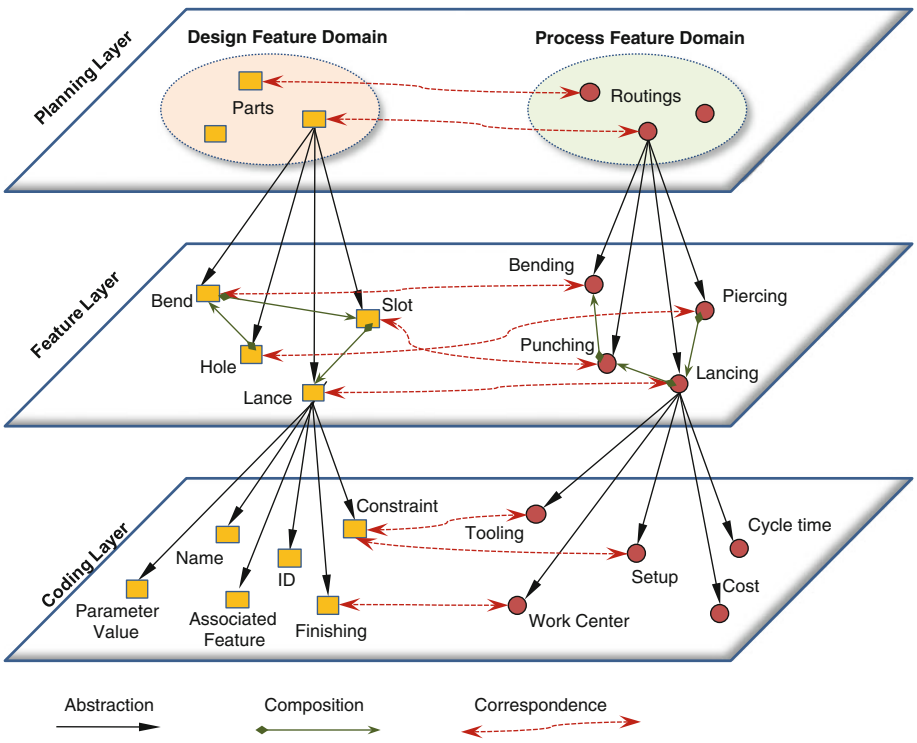


**Fig. 14** Variety mapping across different levels of abstraction

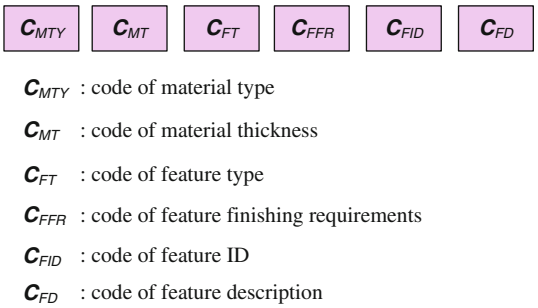


**Fig. 15** The coding scheme

domain across different layers are related by abstraction relationships.

At the planning layer, a part in the design feature domain is associated with a routing in the process feature domain by the correspondence relationship. Across the planning and feature layers, a part can be decomposed to a set of design features. Similarly, a routing can be decomposed to several manufacturing features by the abstraction relationship. At the feature layer, various design features (e.g., "Hole", "Bend", "Slot", "Lance") are connected with each other through the composition relationship; manufacturing features are composed of "Bending", "Piercing", "Punching", "Lancing", and so on. In addition, each design feature has one or more corresponding manufacturing features in the process feature domain, and vice versa. Across the feature and coding layers, a design feature consists of a set of design code; a set of manufacturing code constitutes a manufacturing feature. At the coding layer, there is a series of code corresponding to both design and manufacturing features. Take the design feature "Lance" as an example. It is specified by code representing "Name", "ID", "Parameter Value", "Associated Feature", "Constraint", and "Finishing". In addition, the code representing "Constraint" is connected with the code representing "Tooling" and "Setup" by the correspondence relationship.

### Coding scheme

The coding scheme is the kernel of the variety coding information system. As shown in Fig. 15, the first code in the coding scheme is the code for different material types, such as aluminum, stainless steel, titanium, and copper. According to the ascending order of non-negative integer, each material is assigned to a unique digit. The second code represents the thickness of the material. For most sheet metal parts, the thickness ranges from 0.5 mm to 4 mm. Material along with its thickness results in different heat treatment specifications (e.g., whether heat treatment should be carried out on sheet metal or on finished parts). The third code represents the feature type. Along with material and material thickness, the feature type determines the selection of operations and the corresponding work centers. The fourth code indicates finishing requirements. The fifth code, feature ID code, denotes the relationship of one feature with other features. It also influences the selection of operations. The last code is defined for feature description, containing variety feature values and feature location values. The above code collectively determines the cycle times, setups, and tools/fixtures for different operations, which, in turn, contribute to production performance and cost estimation.

### Code database

A code database is further developed based on the coding scheme to store useful information. The code database contains a number of fields, each of which includes a number of data attributes. Some of the most important fields are listed in Table 3. Take the *part, material, design feature, manufacturing feature, operation, work center*, and *setup* fields as examples. The *part* field records the part code, part name, part ID, and order quantity, etc. The *material* field contains material code, material name, material ID, and material specification, etc. The *design feature* field shares some common data attributes with *manufacturing feature*, including feature code, feature name, operation ID, and work center ID, etc. The *operation* field consists of operation code, operation time, average cost, and cycle time, etc. The *work center* field stores work center code, tolerance, and operation ID, etc. The *setup* field shows setup code, setup time, and operation ID, etc. As a result, this code base links design features with manufacturing features through the common information elements embedded in different fields, thus facilitating PPV coordination.

### Frame-based knowledge base

Along with the variety coding methodology, a knowledge base drives decision support by storing and coordinating experiential knowledge of design and manufacturing. Such knowledge guides engineers to determine design parameters and process variables. In view of the fact that frames are effective in representing classes and objects, a frame-based knowledge base is developed to provide a means for collecting, organizing and retrieving knowledge.

A knowledge frame is a formatted representation of constraints and rules associated with design, manufacturing, and their relationships. The connection between a knowledge frame and its frame member entails a class-object relationship. For illustrative purpose, a design knowledge frame example for a typical design feature, "Dimple", is given in Fig. 16. As an object of "Dimple" class, "Dimple1" contains several design rules characterized by "SurfaceFeatures", and "DesignFeatures" super frames. Take "Rule 1001" as an example. It describes a design constraint: The

**Table 3** Fields and data attributes in the coding database

| Fields | Data attributes |
|---|---|
| Part | Part code, part name, part ID, part description, process plan, drawing No., order quantity, due date, customer ID |
| Material | Material code, material name, material ID, material specification, nick name |
| Material thickness | Thickness code, material ID, thickness value, operation ID, work center ID |
| Design feature | Feature code, feature name, operation ID, work center ID, parametric, shape ID |
| Mfg. feature | Feature code, feature name, operation ID, work center ID, parametric, shape ID |
| Feature description | Description code, description ID, feature name, feature ID, part ID, feature finishing requirement, variety feature specification, feature location etc. |
| Heat treatment (HT) | HT code, HT name, HT ID, part ID, material ID, time duration, temperature, oven material thickness ID |
| Operation | Operation code, operation time, operation ID, operation description, work center ID, feature name, average cost, cycle time |
| Work center (WC) | WC code, WC name, WC ID, machine name, tolerance, minimum economic quantity, hard/soft material thickness range, feature name, operation ID |
| Bending die/punch | Feature ID, die/punch set code, die/punch set model, die/punch set location, die/punch set specification, WC ID, operation ID |
| Punching die/punch | Feature ID, die/punch set code, die/punch set model, die/punch set location, die/punch set specification, WC ID, operation ID |
| Setup | Setup code, setup time, setup ID, setup description, WC ID, operation ID, feature name, average time |
| Cost | Cost code, cost value, WC ID, operation ID, feature name, setup ID |
| Cycle time | Cycle time code, time value, WC ID, operation ID, feature name, setup ID |

FrameMember: **Dimple1**
Frame: **Dimple**
Super Frame: SurfaceFeatures, DesignFeatures
Sub Frames:
**Rule 1001**
[1: Height>6]=>[Height=6 ELSE Height=Height]
**Rule 1002**
[1: SheetThick>3]=>[SheetThick=3 ELSE SheetThick=SheetThick]
**Rule 1003**
[1: DistToBend2<(5*SheetThick+DiaTol/2)]=>[DistToBend2=(5*SheetThick+DiaTol/2) ELSE DistToBend2=DistToBend3]
**Rule 1004**
[1: DistToBend3<(5*SheetThick+DiaTol/2)]=>[DistToBend3=(5*SheetThick+DiaTol/2) ELSE DistToBend3=DistToBend3]
**Rule 1005**
[1: DistToBend5<(5*SheetThick+DiaTol/2)]=>[DistToBend5=(5*SheetThick+DiaTol/2) ELSE DistToBend5=DistToBend5]
**Rule 1006**
[1: DistToBlankEdge1<(DiaTol/2+2*SheetThick)]=>[DistToBlankEdge1=(DiaTol/2+2*SheetThick) ELSE DistToBlankEdge1=DistToBlankEdge1]

**Fig. 16** Design feature knowledge frame for "Dimple"

maximum height for "Dimple1" cannot exceed 6mm. This design knowledge helps engineers make better decisions by coordinating product design with the existing manufacturing resources.

## Prototype implementation

In view of system complexity, in this study, a radial-layered DCM system architecture is developed for better understanding the whole system, as shown in Fig. 17. It consists of five layers: (1) *an information model*, (2) *functional modules*, (3) *a user interface*, (4) *users*, and (5) *a domain information repository*. The information model, functional modules, user interface and users serve as the frontend platform, whilst the information repository performs as the backend server for storing data, information and models specific to different applications.

The SysML-based information model is the core of the system architecture, representing key aspects of the system, including requirements, structure, behaviors, and constraints, as discussed in "A SysML model for production reconfiguration". The information system is handled by a variety of users, including customers, market analysts, engineering managers along with design, process and production engineers. Customers purchase switchgear enclosures through orders. Market analysts gather and analyze information about customer needs. Design engineers are concerned with developing the preliminary design and most critical parts. Process engineers take care of the manufacturing processes. The task of production engineers is to minimize production time and costs at the shop floor level. Engineering mangers provide leadership for engineers and support coordination between design, production, accounting, finance, and human resource departments.

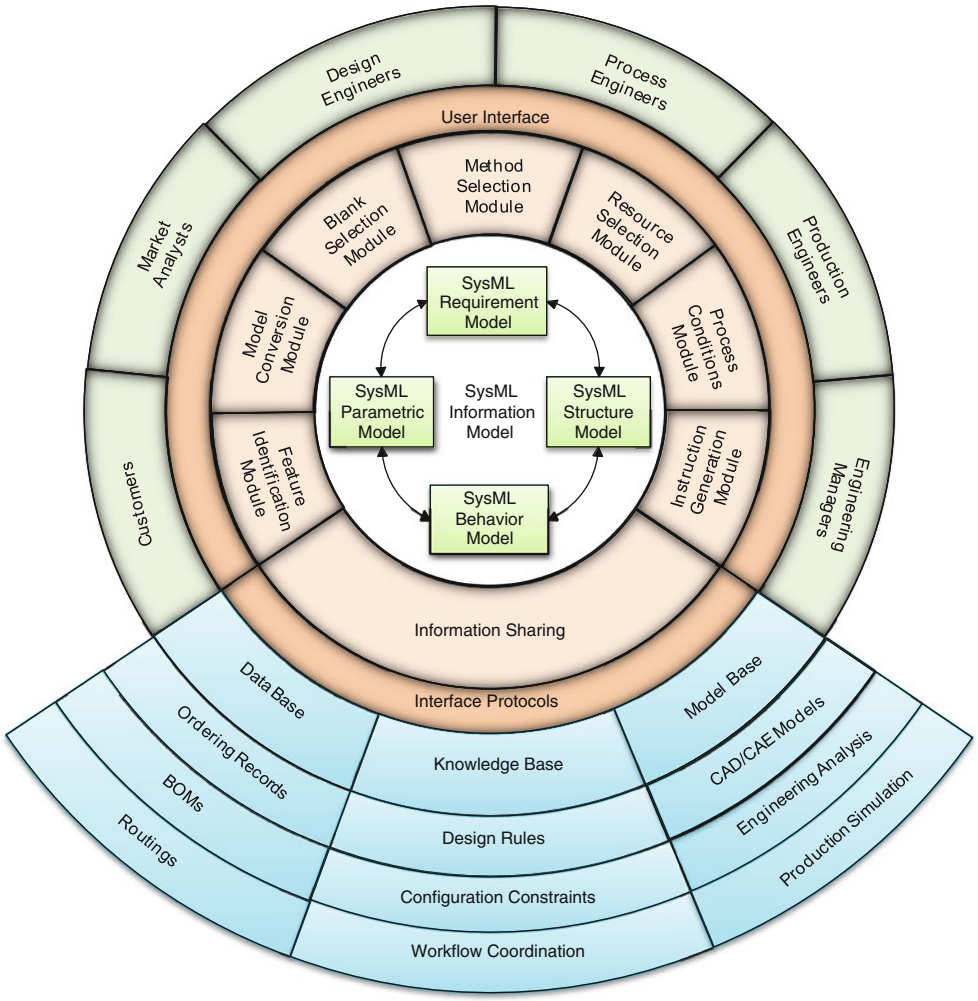


**Fig. 17** System architecture of sheet metal production reconfiguration

The functional modules of switchgear enclosure production reconfiguration system encompass (1) feature identification, (2) model conversion, (3) blank selection, (4) method selection, (5) resource selection, (6) process conditions, and (7) instruction generation. The user interface enhances the communication between users and the system, through which users interact with these functional modules. System protocols provide formal description of message formats and rules for exchanging messages. It serves as a common channel to support information sharing between different modules.

The domain information repository consists of databases, knowledge bases, and model bases. The database cluster encompasses ordering records, BOMs and routing data. The knowledge bases are categorized by production rules, configuration constraints, and workflow coordination. Production rules are implemented in the form of if-then rules. Configuration constraints include functional, safety, quality, manufacturing, timing, economic, ergonomic, and ecological life-cycle constraints. The workflow coordination cluster oversees various tasks and communications among different functional departments along the design chain. The domain model bases comprise CAD/CAE, engineering analysis, and production simulation models. The CAD/CAE models contain engineering drawings and finite element analysis. Engineering analysis involves performance and cost models. The production simulation model is more concerned with discrete event simulation, which has been commonly used to study the production system behaviors by simulating a large amount of production scenarios. Ultimately, it provides suggestions to improve the production performance in terms of productivity, resource utilization, lead time, and the like, along with production costs.

A prototype of variety coding information system is implemented for switchgear enclosure production reconfiguration. For illustrative simplicity, an example of a basis frame is used to demonstrate the information system's effectiveness in achieving production reconfiguration through variety management. Figure 18 shows the engineering BOM and associated process routings generated by the prototype system, which allows for PPV coordination. As shown in Fig. 19,

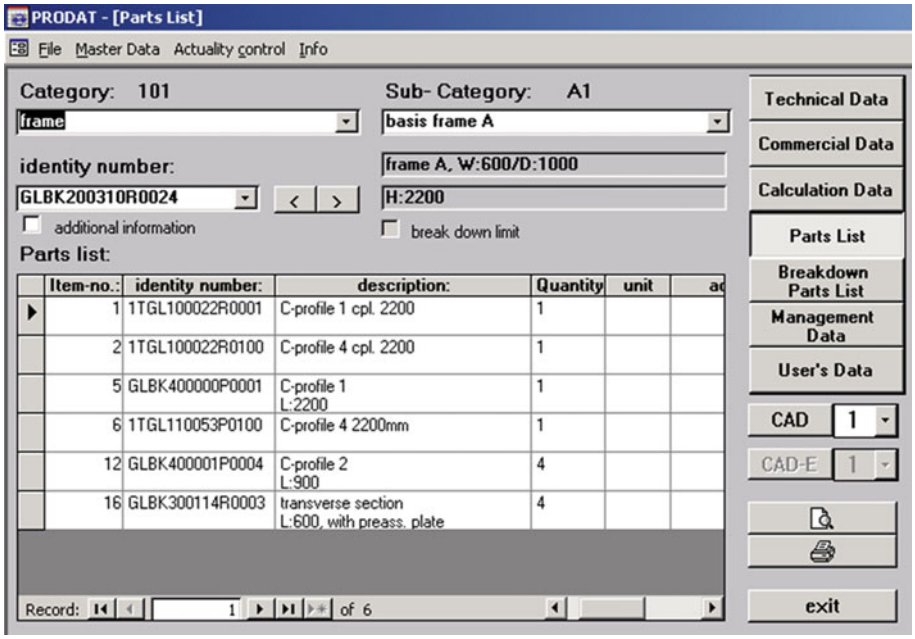


**Fig. 18** Engineering BOM and process plans for a basis frame

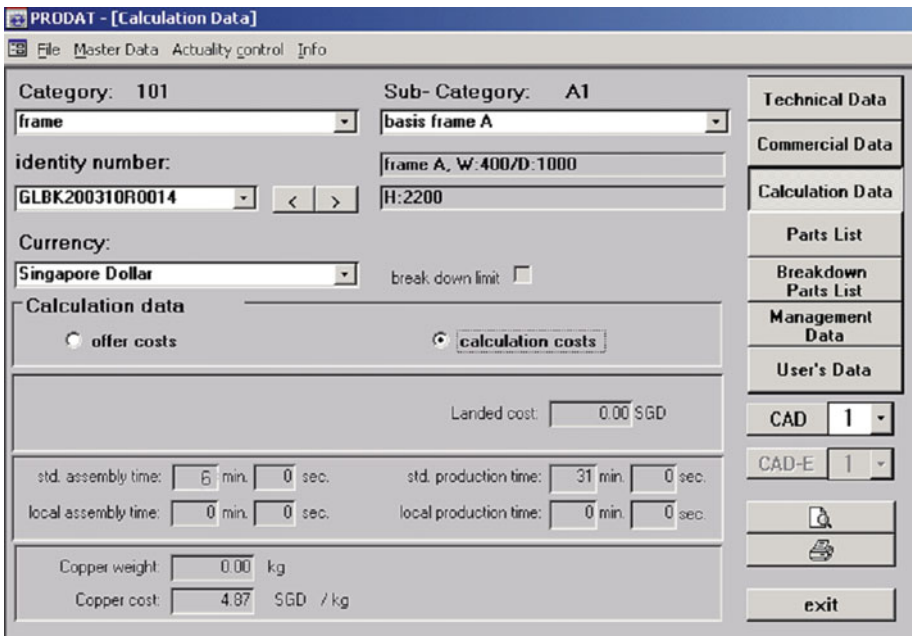


**Fig. 19** Variety-time-cost trade-off analyses

some preliminary variety-time-cost trade-off analyses are performed to provide decision support to variety management in production reconfiguration. Due to the goal of this paper is not to investigate the trade-off analysis but to demonstrate the SysML-based information model can formally capture information and knowledge, the validation of optimality of the alternatives is out of the scope of this paper.

## Concluding remarks

In this paper, we presented a holistic view of DCM along the entire spectrum of product realization, aiming to achieve production reconfiguration through effective variety management that aligns the customer, product, and process domains. We then approach DCM from model-based systems engineering perspective and attempt to manage system complexity with focus on managing PPV. Essentially, effective variety management relies on GVP and the associated information model.

GVP unifies the GFS, GDS, and GPS associated with a product family through variety features, thus facilitating variety propagation modeling from design to production. It enables the derivation of product and process variants from the GVS for specific customer orders. A SysML-based

information model is developed to formally represent the requirements, structure, behaviors, and constraints of the switchgear enclosure production reconfiguration. It enables the system to share and exchange information among multiple domains in a multi-disciplinary team by ensuring semantic coherence along the entire design chain, keeping the traceability across levels of abstraction, and improving the interoperability among tools. Moreover, it supports trade-off analysis to evaluate production performance and costs by combining with other execution environments.

Finally, a variety coding information system is developed based on the variety coding methodology and the SysML-based information model. The prototype is implemented to demonstrate its success in achieving production reconfiguration through variety management.

Nonetheless, the prototype system is based on one company only, which may not be sufficient for generalizing results for multiple industries. In this regard, to quantify the resulting improved performance with respect to production time and costs, the coding information system might need to be empirically tested in a number of companies of different industries. In light of the focus of this study, we do not address integration of discrete event simulation tools, and evaluation of the value to customers of different variety dimensions, which deserve separate studies.

## Nomenclature

| | |
|---|---|
| *Artifact* | A distinct entity in a product (e.g., an assembly, a subassembly, or a part). |
| *Function* | One aspect of what the artifact is supposed to do, which specifies the relationship between inputs and outputs in terms of energy, material and information. |
| *Behavior* | Information represented as a sequence of states and transitions between them, supporting the simulation of the product under some given conditions. |
| *Form* | Proposed design solution for the design problem specified by the function, represented in terms of its geometry and material. |
| *Geometry* | The spatial description of the artifact (e.g., shape, dimension). |
| *Material* | The description of the internal composition of the artifact. |
| *Assembly* | A composition of its subassemblies and parts. |
| *Part* | The lowest level component. |
| *Flow* | The medium (e.g., information, energy, material) that serves as the output of one or more function(s) and one or more corresponding input(s). |
| *Requirement* | A statement that specifies an attribute, capability, characteristic, or quality of an artifact that governs some aspect of its function, form, geometry or material, which is a element of the specification. |
| *Specification* | The collection of information relevant to the design and manufacturing of an artifact deriving from customer needs and/or engineering requirements. |
| *Processing operation* | Transform a material from one state to a More advanced state that is closer to the final desired artifact (e.g., bending, punching, heat treatment, etc.). It adds value by changing the geometry and/or properties, of the material. |
| *Assembly operation* | Joins two or more artifacts to create a new Artifact, which is an assembly or subassembly. |
| *Process routing* | The sequence of individual processing and Assembly operations needed to produce an artifact. |
| *Product variety* | The number of distinctive end items or Product variants offered by a producer. Product variety stems from differences in product physical form and its function. |
| *Process variety* | The number of distinctive process variants Applied on the shop floor to produce corresponding product variants. Process variety is the direct consequence of product variety. |
| *Variety creation* | The approaches to create product variety That can meet customer needs. It involves creating distinctive product physical form and function. |
| *Variety implementation* | The activities to implement variety creation. It involves how product and process varieties are coordinated to create product variety. |
| *Variety feature* | An attribute associated with an artifact that Has a specific function assigned to it. Different instances of a particular variety feature represent the diversity of variety feature variants, and thus creating product or process variety. |
| *Design feature* | Information associated with a specific Design attribute of an artifact. The most common design features are associated with form, geometry, and material, which contain both shape information and parametric information. It is a kind of variety feature that create product |

| | |
|---|---|
| | variety. A product variant is an instance of a particular set of design features. |
| *Manufacturing feature* | Information associated with a specific Manufacturing attributes of an artifact. The most common manufacturing features are associated with manufacturing process to produce corresponding design features. It is a kind of variety feature that create process variety. A process variant is an instance of a particular set of manufacturing features. |

**Acknowledgments** The authors would like to thank Chris Paredis and Leon McGinnis for providing access and support for MagicDraw. Huangming Wu is also acknowledged for assistance in conducting case studies and implementing the prototype system.

## References

Akira, T. (2001). Bridging inter- and intra-Firm Boundaries: Management of supplier involvement in automobile product development. *Strategic Management Journal, 22*(5), 403–433.

Albla s, A., Zhang, L., & Wortmann, H. (2010). Representing function-technology platform based on the unified modeling language. *International Journal of Production Research*, Under Review.

Brunetti, G., & Golob, B. (2000). A feature-based approach towards an integrated Product Model including conceptual design information. *Computer-Aided Design, 32*(14), 877–887.

Candadai, A., Herrmann, J. W., & Minis, I. (1996). Application of group technology in distributed manufacturing. *Journal of Intelligent Manufacturing, 7*(4), 271–291.

Cao, Y., Liu Y., & Paredis, C. J. (2010). Integration of system-level design and analysis models of mechatronic system behavior based on SysML and simscape. In *Proceedings of the ASME design engineering technical conferences*, Montreal, Canada.

Chincholkar, M. M., Herrmann, J. W., & Wei, Y. F. (2003). Applying design for production methods for improved product development. In *Proceedings of the ASME design engineering technical conferences*, DETC2003/DFM-48133, Illinois, USA.

Cho, J., Han, S., & Kim, H. (2006). Meta-ontology for automated information integration of Parts libraries. *Computer-Aided Design, 38*(7), 713–725.

Clark, P. A., & Starkey, K., 1988, *Organization Transitions and Innovation - Design*, London.

Fiorentini, X., Sudarsan, R., Suh, H., Lee, J., & Sriram, R. D. (2008). An evaluation of description logic for the development Of product models. *Journal of Computing and Information Science in Engineering*, Accepted for publication.

Friedenthal, S., Moore, A., & Steiner, R. (2008). *A Practical Guide to SysML: The Systems Modeling Language*.

Grabot, B., Blanc, J. C., & Binda, C. (1996). A decision support system for production activity control. *Decision Support Systems, 16*(2), 87–101.

Herrmann, J. W., & Chincholkar, M. M. (2002). Reducing throughput time during product design. *Journal of Manufacturing Systems, 20*(6), 416–428.

Huang, E., Ramamurthy, R., & McGinnis, L. (2007). System and simulation modeling using SysML. in *Proceedings of the 2007 winter simulation conference* (pp. 796–803).

IBM. (2007). *Software group-extended product lifecycle management*. Solution Deep-Dive Design Chain Management.

Jiao, J., Tseng, M. M., Ma, Q., & Zou, Y. (2000). Generic bill of materials and operations for high-variety production management. *Concurrent Engineering: Research and Application, 8*(4), 297–322.

Jiao, J., Zhang, L., & Pokharel, S. (2007). Process platform planning for variety coordination from design to production in mass customization manufacturing. *IEEE Transactions on Engineering Management, 54*(1), 112–129.

Johnson, T. A., Paredis, C. J., & Burkhart, R. M. (2008). Integrating models and simulations of continuous dynamics into SysML. *6th international Modelica conference* (pp. 135–145). Linköping, Sweden: Modelica Association.

Johnson, T. A., Paredis, C. J., Burkhart, R., & Jobe, J. M. (2007). Modeling continuous system dynamics in SysML. *ASME international mechanical engineering congress and exposition*, Washington, USA.

Lee, Y. T. (1999). Information modeling: From design to implementation. In *Proceedings of the second world manufacturing congress*, Canada.

Li, Z., & Ramani, K. (2007). Ontology-based design information extraction and retrieval. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 21*(2), 137–154.

Lim, S. C. J., Liu, Y., & Lee, W. B. (2010). A methodology for building a semantically annotated multi-faceted ontology for product family modeling. *Advanced Engineering Informatics*, Online.

Lin, H. K., Harding, J. A., & Shahbaz, M. (2004). Manufacturing system engineering ontology for semantic interoperability across extended project teams. *International Journal of Production Research, 42*(24), 5099–5118.

Liu, W., & Zeng, Y. (2010). Formalization of design chain management using environment-based design theory. *Journal of Intelligent Manufacturing*, (in press).

Loffredo, D. (1998). *Efficient database implementation of EXPRESS information models*. Ph.D. Thesis, Rensselaer Polytechnic Institute, New York.

Martin, M. V., & Ishii, K. (2002). Design for variety: Developing standardized and modularized product platform architectures. *Research in Engineering Design, 13*(4), 213–235.

Mayer, R. J., Crump, J. W., Fernandes, R., Keen, A., & Painter, M. K. (1995). *Information integration for concurrent engineering (IICE) compendium of methods report*. Wright-Patterson Air Force Base, Ohio 45433-7604, http://www.idef.com/pdf/compendium.pdf.

Ming, X. G., Yan, J. Q., Lu, W. F., & Ma, D. Z. (2005). Technology solutions for collaborative product lifecycle management—status review and future trend. *Concurrent Engineering: Research and Applications, 13*(4), 311–319.

Nagarajan, R. P., Passey, S. J., Wong, P. L., Pritchard, M. C., & Nagappan, G. (2004). Performance measures and metrics for collaborative design chain management. In *the 10th international conference on concurrent enterprising*, Seville, Spain.

Nyere, J. (2006). *The design chain operations reference model, supply chain council (SCC)*.

OMG (Object Management Group). (2009). *OMG systems modeling language specification*. http://www.omg.org/spec/SysML/1.2/.

Olavson, T., & Fry, C. (2006). Understanding the dynamics of value-driven variety management, *MIT Sloan Management Review*, (Fall), 63–69.

Panchal, J. H., Fernandez, M. G., Allen, J. K., Paredis, C. J., & Mistree, F. (2009). A modular decision-centric approach for reusable design processes. *Concurrent Engineering: Research and Applications, 17*(1), 5–19.

Peak, R. S., Burkhart, R. M., Friedenthal, S. A., Wilson, M. W., Bajaj, M., & Kim, I. (2007). Simulation-based design using SysML part 1: A parametrics primer. *INCOSE international symposium*.

Poirier, C. C., & Reiter, S. E. (1996). *Supply chain optimization: Building the strongest total business network*. California, USA: Berrett-Koehler Publishers.

Ramdas, K. (2003). Managing product variety: An integrative review and research directions. *Production and Operations Management, 12*(1), 79–101.

Ramdas, K., Zhylevskyy, O., & Moore, W. (2010). A methodology to support product differentiation decisions. *IEEE Transactions on Engineering Management, 57*(4), 649–660.

Robertson, D., & Ulrich, K. T. (1998). Planning product platforms. *MIT Sloan Management Review*, (Summer), 19–31.

Salvador, F., De Holan, P., & Piller, F. (2009). Cracking the code of mass customization. *MIT Sloan Management Review*, (Spring), 71–78.

Shah, A. A., Kerzhner, A. A., Schaefer, D., & Paredis, C. J. (2009). *Multi-view modeling to support embedded systems engineering in SysML*. Lecture Notes in Computer Science, Springer.

Sudarsan, R., Baysal, M. M., Roy, U., Foufou, S., Bock, C., Fenves, S. J., Subrahmanian, E., Lyons, K., & Sriram, R. D. (2005). Information models for product representation: Core and assembly models. *International Journal of Product Development, 2*(3), 207–235.

Suh, N. P. (2001). *Axiomatic design: Advances and applications*. New York: Oxford University Press.

Twigg, D. (1998). Managing product development within a design chain. *International Journal of Operations and Production Management, 18*(5), 508–524.

Twigg, D. (2005). *Design chain, the Blackwell encyclopedia of management: Operations management* (2nd ed.). Oxford: Blackwell.

Ulrich, K. T. (1995). The role of product architecture in the manufacturing firm. *Research Policy, 24*, 419–440.

Ulrich, K. T., Randall, T., Fisher, M., & Reibstein, D. (1998). *Managing product variety: A study of the bicycle industry, Managing Product Variety*. Netherlands: Kluwer.

Ulrich, K. T. (2007). Variety. *Design: Creation of Artifacts in Society*. Retrieved from the World Wide Web: http://opim.wharton.upenn.edu/~ulrich/documents/ulrich-variety.pdf.

van Veen, E. A., & Wortmann, J. C. (1992). Generic bill of material processing systems. *Production Planning & Control, 3*(3), 314–326.

Wang, J., & Lin, H. Y. (2006). A fuzzy hybrid decision-aid model for selecting partners in the design chain. *International Journal of Production Research, 44*(10), 2047–2069.

Wu, W. H., Yeh, S. C., & Fang, L. C. (2007). The development of a collaborative design chain reference model for the motorcycle industry. *The International Journal of Advanced Manufacturing Technology, 35*(3–4), 211–225.

Zeng, Y. (2004). Environment-based formulization of design problem. *Journal of Integrated Design and Process Science, 8*(4), 45–63.

Zha, X. F., & Sriram, R. D. (2006). Platform-based product design and development: A knowledge-intensive support approach. *Knowledge-Based Systems, 19*(7), 524–543.

Zhang, L., Xu, Q., & Jiao, J. (2010). Domain-based production reconfiguration with constraint satisfaction. *Computers in Industry*, Under Review.

Zipkin, P. (2001). The limits of mass customization. *MIT Sloan Management Review, 42*(3), 81–87.