NISTIR 7922

Engineering Change Management Concepts for Systems Modeling

Conrad Bock
Allison Barnard Feeney

http://dx.doi.org/10.6028/NIST.IR.7922

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

NISTIR 7922

Engineering Change Management Concepts for Systems Modeling

Conrad Bock
Allison Barnard Feeney
Systems Integration Division
Engineering Laboratory

http://dx.doi.org/10.6028/NIST.IR.7922

April 2013

U.S. Department of Commerce
Rebecca Blank, Acting Secretary

National Institute of Standards and Technology
Patrick D. Gallagher, Under Secretary of Commerce for Standards and Technology and Director

Engineering Change Management Concepts for Systems Modeling

Conrad Bock
Allison Barnard Feeney
April 17, 2013

A significant cost of manufactured systems arises in changing system specifications after they have been built and delivered to customers. This might be due to errors in the original specifications, feedback from customers, competition from other producers, or new ideas for product improvement. It involves revisiting decisions and tasks that were considered completed, including modifying engineering specifications produced by these activities. This paper reviews engineering change management to understand the demands it places on system models and modeling languages, in particular the Systems Modeling Language.

1 Introduction

Specifications for manufactured systems inevitably change after systems are built and delivered to customers (*engineering change*) [1]. This might be due to errors in the original specifications leading to product failure or unintended negative consequences, feedback from customers on problems or new features, competition from other producers, new ideas for product improvement, or new products adapted from existing ones. The need for post-production changes is reduced with more adaptable systems, but increasingly complex products, shorter product lifecycles, and faster technological innovation make it impossible to completely predict how systems will need to change [2].

Engineering changes account for a significant portion of the cost of manufactured systems [3], and delays in responding to them reduce customer satisfaction and competitiveness [4]. Part of the cost and delay is in modifying engineering specifications, and part is in moving approved changes into production [5]. These problems are due in part to difficulty in determining the effect of changes on other parts of the product and organization, which sometimes are cascading and non-converging, and on poor classification of changes leading to inefficient approval processes [2].

Techniques for facilitating modification of system specifications during engineering change processes include:

- System design principles that increase adaptability to change, including platform-based techniques ("design for change") [6] [7] [8].

- Model content and modeling language capabilities that facilitate engineering change processes, including relationships between varying specifications of the same system, and between system specifications and the organization producing it.

This paper focuses on the second technique above, reviewing engineering change management generally to identify the demands it places on models and languages for specifying systems, specifically the Systems Modeling Language, an extension of the Unified Modeling Language for systems engineering (SysML/UML) [9] [10]. The paper does not address evaluation of product or specification quality, such as system validation and model maturity. The paper covers processes for managing engineering change in Section 2, and how changes to system specifications interact with production in Section 3. Section 4 describes the implications of Sections 2 and 3 for system modeling languages, SysML in particular, and Section 5 summarizes the paper.

2 Engineering change processes

Engineering change processes begin with reports about system problems or suggestions for improvements (*change requests*) and ultimately produce modifications to specifications, including when they will be produced, see Section 3.1 (*change orders*). Change processes are the portion of system development that prioritize change requests and evaluate alternative proposed change orders based on the effect on the productivity and profitability of the engineering organization, rather than the degree to which orders satisfy the change request. A proposed change order might completely address a change request, but have a negative effect on the productivity and profitability of the organization, whereas another order might partially address the request, and improve the productivity and profitability of the organization.

Contrary to the terminology, change requests describe the situations that led to the requests, rather than propose specific changes to the system [5] [11]. For example, when systems behave in undesirable ways, change requests describe the undesirable behaviors and the circumstances in which they occurred, rather than propose system modifications that requestors believe will eliminate the undesired behaviors. Change requests enable engineers to determine the benefit and priority of addressing change requests, and encourage consideration of more and possibly better change orders for the same requests. Change requests that only propose system modifications are effectively change orders. Engineers are forced to guess the original situation, and when guessing incorrectly might reject legitimate requests or propose solutions that do not address the actual problem.

Engineering change processes consider the need for requested changes and the effect of potential change orders on the rest of the system and the engineering organization producing it. Changes usually must be prioritized due to limited system and production development resources, and be analyzed for the possibility of causing cascading changes. Section 2.1 covers the prioritization of change requests, while Section 2.2 describes impact analysis for change orders. Section 2.3 identifies key characteristics of change processes. This paper does not cover other effets of potential change orders on organizations, such as role-based access and overlapping change orders.

2.1 Urgency of change requests

Evaluating the need to address engineering change requests starts with examining the situations in which change requests arise, to determine how quickly the change should be addressed (*urgency*). For example, change requests might arise from difficulties in parts fabrication or assembly, or from behaviors of the system during operation that are undesirable or insufficient for the intended tasks [12].

Urgency is measured by when the product is built to the changed and original specifications:

- Very urgent changes require production to halt immediately for the current specification. These are usually changes involving safety in the operation or production of the system, rejection by operators of the system, or inability to build the system properly due problems in the system specification.

- Moderately urgent changes allow continued production for the current specification, but not for the full length of time originally planned, see Section 3.1. For example, a moderately urgent change can occur in the middle of regular product cycles, such as the middle of a model year for automobiles. These changes improve the perceived quality of the system or efficiency of production enough to justify the cost of modifying production processes earlier than planned.

- Least urgent changes do not affect production for the current specification, which continues for the full length of time originally planned. Production of the changed specification might occur after or during production for the original specification, see Section 3.2. If production for the original specification is reduced to accommodate parallel production for the changed one, then the change urgency is between moderate and least.

The level of urgency is independent of other classifications of change requests that are less concerned with situations in which the change requests arise, such as whether the changes are fixes or improvements, involve many or few changes to the system specification, or are requested from customers, production, or suppliers. For example, major safety problems might be addressed by simple fixes, or require only corrections to seemingly minor errors in the specifications, or arise when systems or components are being stored or transported. Likewise, least urgent changes might be major improvements, or involve many changes to system specifications, or significantly improve supplier efficiency.

2.2 Impact of change orders

Engineering change orders can affect any part of the production process, from raw materials to transportation of finished goods. Some or all of these areas might be significant sources of cost increases or decreases for any particular change order being considered. For example, change orders that affect production tooling are usually expensive, but in some cases, new tooling might be significantly less expensive and more efficient due to new technology available. Engineers must consider each potential change order's effect on every area of production individually to find unexpected cost increases or decreases.

Change orders can involve more systems components than originally intended (*change propagation*). Orders that appear simple at first can result in additional unanticipated changes within the same order or in subsequent orders. Ideally, these begin to decrease over time and stop, but they can also increase again, and in extreme cases continue indefinitely [13]. The production cost calculation described above must be repeated for all affected components, for the original intended change and all cascading changes. In addition, the risks of unexpected effects on production or operation of a system increases with every component changed. For these reasons, predicting and reducing propagation of changes is an essential task in determining the impact of potential change orders.

Change orders that increase the chance of propagation include:

- Changes to shape, material, or interactions with other components, rather than changes to component internals only [14].

- Applying new technologies, or using new subfunctions that are not already in use within the same product family, and innovation in general [15].

- Propagation to components in different subsystems rather than in the same subsystem [3].

- Adding new components that change the assembly of existing components rather than just introducing separate additional assembly steps [5].

- Changes to components that are used in other subsystems or products [13].

System components can be analyzed to assess their susceptibility to propagating change to other components in the system. Some system component or subsystems can be changed quite a bit without affecting other components, some affect other components to the same degree they are changed, and some cannot be changed much without affecting other components [13]. These assessments depend on the degree of change, for example, components that usually can be changed without affecting others might have a significant effect if their operating margins are exceeded. Repeated changes can reduce the capacity of components to absorb changes. This means susceptibility to propagating change can only be determined statistically [16] [17], and in the end each change requires its own propagation analysis, which depends critically on knowledge of the operating margins of each component [13].

2.3 Characteristics of change processes

Engineering change processes vary widely between and within engineering organizations depending on the kinds of things being produced, the kind of engineering changes involved, and market demands on the products. Some products have a high cost of failure and require much more controlled and elaborate change processes, such as aircraft and medical equipment, while other products only cause inconvenience when they fail, as with many consumer products. Some engineering changes are very minor and only require simple change processes, such as a changes that add parts without changing assembly, while others involve potentially complicated analysis and evaluation, as when change propagation occurs across subsystems, see Section 2.2. Finally, some engineering changes must be done quickly and unexpectedly to address safety issues or respond to competition, while others have longer and regular deadlines, such as model year upgrades to automobiles.

The broadest distinction in change processes is how long they typically take to complete [13]. Slower processes allow more time for accurately predicting and evaluating change propagation, for choosing an overall approach before a particular solution is considered, and for separate evaluation of the proposed solution to decide whether to proceed to production or continue the change process. Faster processes go directly to immediately available solutions and spend less time on predicting change propagation, discovering propagated changes after the initial change is made. Fast processes are useful for meeting unexpected short deadlines, making routine changes such as for cable harnessing, and addressing unexpected change propagation resulting from slower processes.

The time taken for slow and fast processes is only typically longer and shorter, rather than guaranteed. For example, slow processes might find very little possibility of change propagation and complete quickly, while fast processes might generate significant unpredicted change propagation and take a long time to reach complete solutions. The typical time taken for change processes is not necessarily related to how rigorous or formal they are. Slow processes might involve very little tracking and review, though they usually do, and fast processes might be carefully managed with automated workflow, though they usually try to reduce process overhead in favor of delivery time.

Engineering organizations usually have a fixed set of change processes and assign changes to them based on urgency of the change request and the results of change propagation analysis on proposed change orders, see Sections 2.1 and 2.2. For example, an organization might have emergency, minor, and full change processes, and assign change requests and potential change orders to the processes based on whether the request urgency is immediate, medium term, or long term, and whether the order is to expected to affect a single component, multiple components, or only assembly [5]. All immediately urgent requests would go through the emergency process, otherwise change orders affecting multiple components go through the substantial change process, while those affecting single components are assigned to the minor process, and those affecting only assembly go through either the minor or substantial process depending on whether the urgency is immediate or long term.

Engineering processes, including change processes, can be designed to encourage or inhibit change by reducing or increasing restrictions on engineering choices for particular products or product families [15]. Systems might be limited to specific technologies, to specific product families, to the same subfunctions and solution principles with a single product family, or to using the same product modules, or to a fixed set of configurations. These restrictions from looser to tighter correspond to a decreasing degree of change allowed and decreasing complexity of change processes. Change processes become more complicated when more change is allowed, even if the latitude is not used, because all changes must go through the same review and evaluation process. This overhead can be reduced by determining when change orders can go through a simplified process, for example, when there are no changes to the shape, material, or interactions with other components [12].

Change processes have many commonalities, despite their wide variety and lack of standardization. They all at least:

- Evaluate the urgency of requests to some degree.
- Develop one or more potential change orders.
- Determine the impact of potential orders to some level of confidence.
- Perform production planning, including determining effectivity, either as part of change order development or afterwards, see Section 3.

Industry organizations have many recommendations related to engineering change processes, including:

- Step-by-step procedures or guidelines [18] [11].
- Information services used in carrying out change processes [19].
- Information structures transmitted between participants involved change processes [20].

- Exchange of engineering change information across companies [21].

Academics also study engineering change processes, including:

- A recent survey proposing a general engineering process based on available literature [1].
- Case studies developing out change process strategies [5] [15] [12] [22].
- Papers identifying measures of change process quality, such as the number of active change requests, time to produce change orders, and cost [3] [22] [23].

3 Production of engineering changes

Engineering changes result in different systems built and delivered to customers, based on different system specifications (engineering changes might be called "produced changes"). Changes during the development of systems (before they are ready to be produced) are not engineering changes even though they involve changes to system specifications. Changes during system development do not affect the rest of the engineering organization as produced changes do, even though they are also important to manage (for the efficiency of the engineering process and because unproduced specifications might be the basis for multiple produced ones, see Section 3.2). This section covers two topics in the production of engineering changes, with Section 3.1 about when production should occur, and Section 3.2 about the production and management of multiple specifications of the same system.

3.1 Effectivity

The *effectivity* of system specifications is the time during which systems should be built according to the specifications. Choosing specification effectivity involves many factors, including product seasonality and competition, time required for setting up or modifying production, distribution, and service processes, availability of components or parts from suppliers, and regulatory requirements [14]. Effectivity can be chosen before, during, or after specification development. For example, it might be determined before development to ensure delivery deadlines are met, or during development to account for unanticipated engineering problems, or after the specification is finished due to complexities in production planning. See Section 2.1 for more about choosing effectivity.

Specification effectivity includes the end of production as well as the start. Sometimes end effectivity is chosen along with start effectivity, if it is known early enough when the product will no longer be needed. More often end effectivity is chosen much later than start effectivity. For example, end effectivity might be chosen to respond to competition, finish production processes that have already started, use up remaining component or part inventory or tooling life that won't be needed in the future, inability to procure replacement parts or tooling, meet new regulatory requirements that prevent production beyond an earlier date than expected, or halt production immediately due to safety problems that require engineering changes [24] [5].

Initial choices for effectivity must be refined to account for potentially complicated and multi-layered production processes, especially when changing specifications that are already being produced. For example, an initial start effectivity might be the time at which products are available to customers, but this implies a time when tooling must be in place or modified to build the products, when new or modified supplied parts must be pulled from inventory, or when new or modified raw materials must be available at production facilities [5]. For changed specifications, refined effectivities must ensure new systems are built according to the changed specifications, rather than accidental combinations of the original and changed specifications. For example, if multiple parts are changed, they must be fed into production processes using all the new parts, rather than just some of them. Sometimes effectivity is given in terms of serial numbers rather than dates, to ensure each individual item leaving a factory is built to exactly one specification [24].

It is critical that effectivity times identify the point in production they are about [5]. The example above refines effectivity backwards in the production process from customer delivery, but the effectivity might be refined forward in production, for example, the time at which a changed part should be produced. Refinements in this case would proceed forward in the process described above, rather than using, for example, only a start time without knowing if it is for the first customer delivery or first part production. Effectivity might also specify facility locations, for example, when products are produced in multiple countries, but are more popular in some regions than others.

3.2 Versions and Variants

Systems are usually subject to multiple engineering changes, each resulting in separate specifications of the same system, some, all, or none of which might currently be in production. This can happen in at least two ways, based on when the specifications are produced:

- *Versions*: Each engineering change builds on another in series, usually with earlier ones intended to be phased out of production at some point, as illustrated at the top of Figure 1.[1] These changes are typically to fix unforeseen problems, reduce cost of production, installation, or repair, improve existing functionality, add new functionality, or respond to changed requirements, which might involve changing or removing existing functionality. Specifications are kept even after it is decided to phase them out of production, for maintenance and repair of products already delivered to customers, or to answer regulatory or litigation questions arising from products still in operation, or possibly for continued production to use up components already purchased.

- *Variants*: Multiple engineering changes are made to the same original specification, resulting in multiple kinds of systems built and delivered to customers for an indefinite period, as illustrated at the bottom of Figure 1. These changes are usually to give customers options about additional functionality they purchase over a "base" product built to the original specification. Specifications for multiple base products can result from changes to the same unproduced specification, which serves as a "platform" for multiple sets of variants, usually called "product lines" or "product families" [6] [25].[2]

[1] The term "version management" for specifications sometimes refers to information systems supporting check in/out and locking of specifications for editing, but this is a different meaning of "version" than used in this paper, which does not address finer granularities of modification, such as objects and files.

[2] Systems might share central components, such as different automobiles built on the same frame, but the term "platform" applied to these is different than defined in this paper. Shared components as platforms might be for products that are too different to be considered in the same family, for example, automobile frames used for both compact and crossover sports utility vehicles.

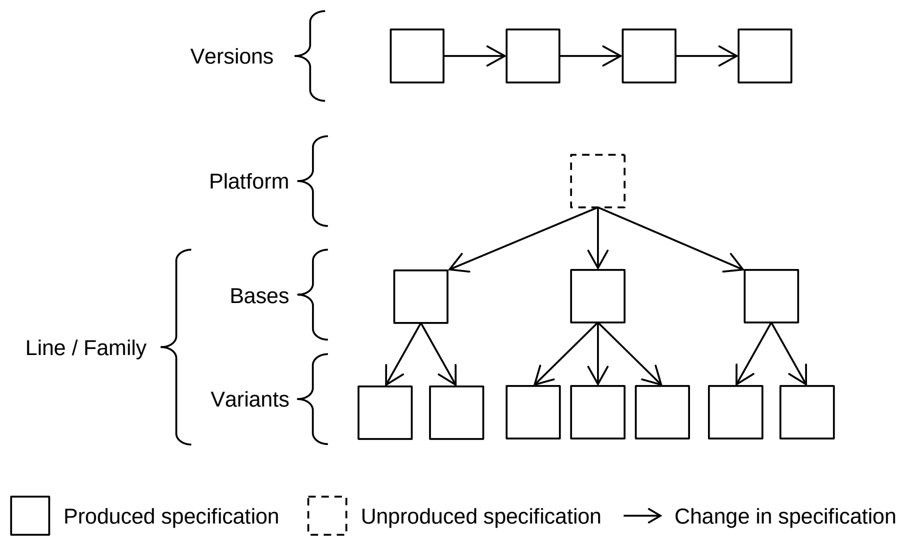

Figure 1: Versions and Variants

Engineers must be able to manage multiple specifications of the same system, as in the specifications for versions and variants above. These specifications can share common components, even when they are for different versions or variants. For example, specifications for versions of the same automobile might all share the same specification for the engine. This means specifications for versions or variants go to production along with the particular component specifications they need, which is sometimes called "configuration management" [14]. See Section 4 about modeling support for managing multiple specifications of the same system.

4 Modeling support for engineering change

Model-based techniques for specifying systems and products are a kind of digital method that integrates engineering concepts into computer languages, enabling engineering-friendly computational assistance in developing system specifications. They are known to provide major efficiency improvements in system specification over document-based approaches [7]. Engineers build system models using modeling languages that enable them to express engineering knowledge and decisions in computer-manageable ways. Section 4.1 describes the implications of Sections 2 and 3 for system modeling languages generally, while Section 4.2 focuses on SysML/UML in particular and assumes understanding of SysML terminology and concepts.

4.1 Modeling capabilities for engineering change

Based on Sections 2 and 3, models used in engineering change cover at least three kinds of subject matter:

- Systems themselves, including system requirements [9] [10].
- Management of changes within engineering, including requests and orders, their urgency and impact respectively, and change processes.
- Production planning, including order effectivity.

A critical capability of systems models for engineering change is to facilitate change propagation, see Section 2.2. Engineering organizations can incur very large costs when change orders are deployed that result in further change orders to address unexpected consequences. Change propagation analysis determines how many specification changes will be needed to address change requests, which in turn determines when orders are complete enough to deploy. Systems modeling languages can facilitate change propagation by including at least two kinds of product information:

- Operating margins, to determine if propagated effects require more changes in system elements [13]. For example, if components are designed to operate within certain temperature limits, and proposed change orders cause temperatures to move only within those limits, then the orders cause no change propagation, at least not due to temperature effects. Otherwise, more changes are needed to address the problem of component temperatures being outside the limits. Operating margins can be for characteristics involved in the intended behavior of the system or not. Change propagation can occur through non-functional characteristics, such as the color of a car affecting the interior temperature.

- Component interactions, distinguishing functional and non-functional interactions, to find all propagated effects [13] [26]. For example, mechanical energy might be intended to flow between components, but perhaps heat does also. Proposed change orders that do not account for non-functional heat transfers might cause components to go out of their operating temperature limits. Modeling component interactions also facilitates communication between different groups of engineers working on large systems. It is important for models to include non-functional as well as functional interactions, since each group has an incomplete understanding of the system and will not be able to completely predict their effect on it.

The management of changes within engineering requires additional modeling beyond the systems themselves, including links between specifications of the same system, see Section 3.2. These links correspond to changes that transform one specification into another, as illustrated in Figure 2. Changes should be modeled separately from specifications, to enable the same changes to be applied to multiple specifications, for example, to the common aspects of multiple variants. Change models also enable producers and other users of specifications to easily find out how new specifications changed and determine the effect on their tasks.[3] Links and their specification changes are the basis for change orders, which should also be linked together to enable change orders to contain more than just changes to specifications, such as links to production plans and other information, see below.

[3] Compare this to just ordering the specifications in time as in typical configuration or version management systems. These systems might be able to generate specification changes by comparing them, but they still must be stored for linking to change requests and other information. Interchangeable formats for change models are needed to share them.

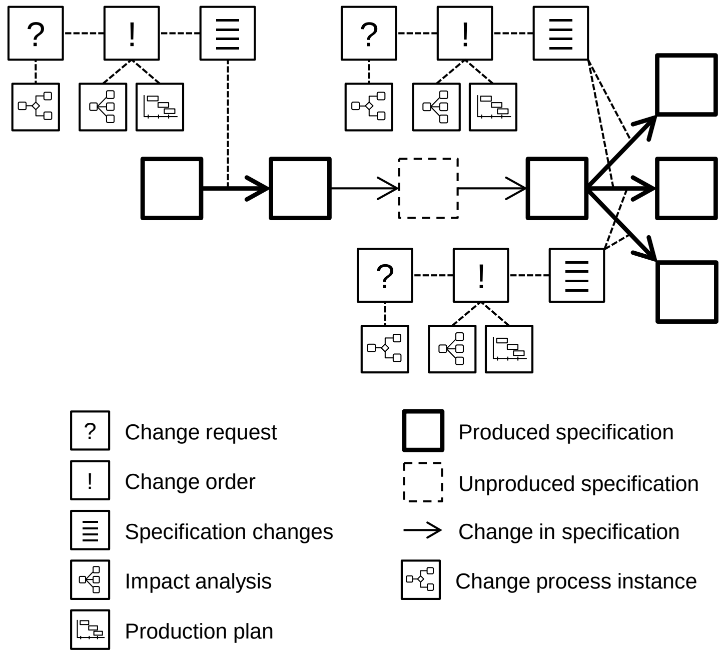

Figure 2: Modeling engineering change

Change orders should be linked to their corresponding change requests, to clarify why changes are being made [5], as illustrated in Figure 2. Change request models should include their urgency, see Section 2.1. If system requirements are kept separate from system specifications, then requirements must be managed in the same way as the specifications, with links between requirements for the same system giving the changes to requirements and linking to change requests. This enables the relationships between requirements and specifications satisfying them to be preserved as systems change.

Management of changes within engineering also requires models for:

- Results of change impact analysis, including change propagation and susceptibility of components to propagating changes, see Section 2.2. These facilitate communication between engineers and engineering teams evaluating the impact of proposed change orders.

- Change processes, including assignments of urgency/impact combinations to process, see Section 2. These can drive workflow automation, to ensure engineers know what tasks are pending, and to enable managers to track progress.

Change impact analysis should be linked to change orders,[4] while change processes should be linked to change requests, as illustrated in Figure 2. Linked change processes are instances of change processes performed starting with change requests, to enable tracking who is participating in what role during that particular process instance, and the status of managing that particular request [27]. Process instances follow models of change processes adopted by the engineering

[4] Results of propagation susceptibility analysis apply to the whole system, including the components that are not being changed, but can still depend on the particular changes being considered, see Section 2.2. Multiple change links from the same specification can share component propagation susceptibility analysis to the degree they can.

organization [28], which might change while handling a single request, due to new information about urgency or impact.

Finally, models are needed for planning production according to changes in system specifications, as illustrated in Figure 2, including when systems should be produced, see Section 3.1. Production planning is outside the scope of engineering change management, but interacts with it to ensure change orders are producible, preferably before orders are issued. Evaluation of proposed change orders should include some estimate of the effects on production, which can be combined with results of impact analysis to determine the overall cost of proposed orders [29], as input to the final decision about addressing the original change request.

4.2 SysML/UML for engineering change

SysML/UML is the most widely used modeling language for systems engineers. It is an open standard published by the Object Management Group (OMG), resulting from work initiated by the International Council on Systems Engineering (INCOSE). INCOSE recognized the need for integration of software with other kinds of engineering and chose to extend OMG'S UML, already the most widely used modeling language for software. UML has many commonalities with systems engineering modeling, because software is also a system, but UML has software-specific capabilities that are excluded from SysML, and SysML adds capabilities specific to systems engineering. SysML was first published in 2007, with the most recent update in 2012. It is adopted by practically all commercial and open source systems engineering modeling tools.

The first kind of engineering change capabilities to consider for SysML/UML are those about systems themselves, in particular, those supporting change propagation analysis, see Section 4.1:

- Operating margins are usually restrictions on numeric values for properties of systems or components, such as temperature, motion, flow rates, and stress. In SysML/UML currently these can only be captured as constraints, typically in UML's Object Constraint Language [30]. Constraint languages are more verbose than engineers probably want to use for such common information as operating margins. Since margins are almost always numeric intervals, an extension of UML Property or specialization of SysML ValueType could introduce metaproperties for numeric intervals, syntactically similar to Multiplicities, but with semantics applying to property values themselves rather than the number of values.

- Component interactions are captured in SysML as flow properties and item flows, which are extensions of UML properties and information flows. These are expressive enough for component interactions, but cannot currently distinguish functional from non-functional flows. Distinguishing functional and non-functional system elements should be a general capability for SysML/UML constructs, including all properties, including flow properties and those with operating margins, as well as the various kinds of flows, including item flows and object flows, and including behavioral elements such as actions and states. Since system engineers are usually concerned with functional elements, the view capabilities in SysML/UML might need to be upgraded to distinguish functional and non-functional elements.

The second area of engineering change capability to consider for SysML/UML involves managing system models within a change process, see Figure 2 in Section 4.1. Managing models would usually be considered to be a topic for extensions of the Meta Object Facility at

OMG (MOF) [31], but since MOF is a subset of SysML/UML, and engineering organizations are systems also, this might be considered an extension of SysML/UML. The extension would involve creating packages that import models (which are also packages) and capture relationships between them, most likely as dependencies.[5] Dependencies would need to be extended with links to change models. Change modeling could be approached in multiple ways, including:

- Transforms written in a model transformation language, most likely OMG's Query, View, Transform [33].

- SysML/UML behaviors with actions that make the changes, preferably in the executable subset of UML (fUML) [34].

- An extension of Element in MOF/SysML/UML (they are all the same in this case) that indicates whether the element is added, removed, or replaced with another element, and a similar extension of Property for property values.

Change models will link to change orders, and orders to change requests, impact analysis results, and production plans, see Figure 2. These might only support engineering processes, whereupon they would be new metamodels outside of SysML/UML, or they might support production and operation of systems, whereupon they could be extensions of SysML/UML. The latter approach provides more precise semantics, which enables more computer-assistance, and more reliable implementation and testing of standards [35]. It can be combined as needed with the former approach for aspects that do not have implications for the production and operation of systems, such as the filers of change requests and approvers of change orders.

Change requests link to process instances, which in turn link to process models. SysML/UML has process modeling capability, and includes fUML, which also models process instances. These have the advantage of being familiar systems engineers, but another popular language for organizational processes is the Business Process Model and Notation (BPMN) [28], which drives widely-used workflow automation systems. BPMN has the disadvantage of having a very inexpressive model for process instances. Extensions of SysML/UML in this area would need to consider other engineering change process standards, see Section 2.3.

5 Summary

This paper reviews engineering change management generally to identify the demands it places on models and languages for specifying systems and describing their changes. It explains central concepts in engineering change processes, including the difference between change requests and change orders, and the determination of the urgency and impact of requests and orders, respectively, as well as general characteristics of change processes, such as encouraging or inhibiting change, and assigning requests and orders to processes, see Section 2. Two important concepts in the relationship between production and engineering changes are outlined: when specifications are produced (effectivity) and the difference between versions and variants, see Section 3. Finally, critical capabilities for models and languages supporting specification changes are identified, SysML in particular, including operating margins and component interactions for change propagation analysis, and the network structure of changed models, see Section 4.

[5] These relationships must not modify the models, and dependencies in the finalized UML 2.5 will not [32].

Acknowledgements

The authors thank Philip Spiby, David Price, John Watson, Mahesh Mani, and Eswaran Subrahmanian for their input.

References

[1] Jarratt, T., Eckert, C., Caldwell, N., Clarkson, P. "Engineering change: an overview and perspective on the literature," Research in Engineering Design, 22:2, pp. 103-124, 2011.
[2] Tavcar, J., Duhovnikb, J., "Engineering change management in individual and mass production," Robotics and Computer-Integrated Manufacturing 21:3, pp. 205-215, 2005.
[3] Terwiesch, C., Loch C., "Managing the process of engineering change orders: the case of the climate control system in automobile development," Journal of Product Innovation Management, 16:2, pp. 160-172, 1999.
[4] Loch, C., Terwiesch, C., "Accelerating the Process of Engineering Change Orders: Capacity and Congestion Effects," Journal of Product Innovation Management, 16:2, pp. 145-159, 1999.
[5] Balcerak, K., Dale, B., "Engineering change administration: the key issues," 5:2, pp. 125-132, 1992.
[6] Wortmann, H., Alblas, Alex., "Product platform life cycles: a multiple case study," International Journal of Technology Management, 48:2, pp. 188-201, 2009.
[7] Boehm, B., et. al., "Systems 2020 Strategic Initiative," Stevens Institute of Technology, Systems Engineering Research Center, Final Report SERC-2010-TR-009, 2010.
[8] Rowe, D., Leaney, J., Lowe, D., "Defining Systems Architecture Evolvability - A Taxonomy of Change," Proceedings of the IEEE International Conference and Workshop on Engineering of Computer Based Systems, Israel, 1998.
[9] Object Management Group, "OMG Systems Modeling Language, version 1.3," http://www.omg.org/spec/SysML/1.3, 2012.
[10] Object Management Group, "Unified Modeling Language, Superstructure," http://www.omg.org/spec/UML/2.4.1, 2011.
[11] Guess, V., *CMII for Business Process Infrastructure*, Holly Publishing Company, 2002.
[12] Pikosz, P., Malmqvist, J., "A Comparative Study of Engineering Change Management in Three Swedish Engineering Companies," Proceedings of the ASME Design Engineering Technical Conference, pp. 78-85, Atlanta, 1998.
[13] Eckert, C., Clarkson, J., Zanker, W., "Change and customisation in complex engineering domains," Research in Engineering Design, 15:1, pp. 1-21, 2004.
[14] Crow, K., "Configuration Management and Engineering Change Control," DRM Associates, http://www.npd-solutions.com/configmgt.html, 2002.
[15] Veldman, J., Alblas, A., "Managing design variety, process variety and engineering change: a case study of two capital good firms," Research in Engineering Design, 23:4, pp. 269-290, 2012.
[16] Clarkson, P., Simons, C., Eckert,C., "Predicting Change Propagation in Complex Design," Transactions of the ASME, 126:5, pp. 788-797, 2004.

[17] Romlia, F., Hou, C., Junxianc, C., Rafie, A., "Subsystems Change Ranking Methodology (SCRaM) for Complex Product Redesign Process," Advanced Materials Research, 308-310, pp. 167-173, 2011.
[18] Verband der Automobilindustrie, "Engineering Change Management Reference Process," http://www.vda.de/en/publikationen/publikationen_downloads/detail.php?id=709, 2010.
[19] Object Management Group, "Product Lifecycle Management Services", http://www.omg.org/spec/PLM/2.1, 2011.
[20] Open Applications Group, "Open Applications Group Integration Specification," http://www.oagi.org/oagis/9.0, 2005.
[21] OASIS, "Product Life Cycle Support DEXs," http://docs.oasis-open.org/plcs/dexlib/oasis_cover.htm, 2011.
[22] Huang, G., Yee, W., Mak, K., "Current practice of engineering change management in Hong Kong manufacturing industries," Journal of Materials Processing Technology 139:1-3, pp. 481-487, 2003.
[23] Saeed, B., Bowen, D., Sohoni, V., "Avoiding Engineering Changes Through Focused Manufacturing Knowledge," IEEE Transactions on Engineering Management, 40:1, pp. 54-59, 1993.
[24] Shiau, J., Li, X., "Product Configuration for Engineering Change Decision," Proceedings of the IEEE International Conference on Networking, Sensing and Control, pp. 691-696, Okayama, 2009.
[25] Alblas, A., Wortmann, H., "Impact of product platforms on lean production systems: evidence from industrial machinery manufacturing," International Journal of Technology Management, 571-3, pp. 110-131 2012.
[26] Jarratt, T., Eckert, C., Clarkson, P., "Development of a Product Model to Support Engineering Change Management," Proceedings Fifth International Symposium on Tools and Methods; Tools and Methods of Competitive Engineering, Switzerland, Horváth and Xirouchakis, eds., pp. 331-342, 2004.
[27] Rouibah, K., Caskey, K., "A workflow system for the management of inter-company collaborative engineering processes," Journal of Engineering Design 14:3, pp. 273-293, 2003.
[28] Object Management Group, "Business Process Model and Notation," http://www.omg.org/spec/BPMN/2.0, 2011.
[29] Sule, D., *Production Planning and Industrial Scheduling: Examples, Case Studies and Applications*, Second Edition, CRC Press, 2007.
[30] Object Management Group, "Object Constraint Language," http://www.omg.org/spec/OCL/2.3.1, 2012.
[31] Object Management Group, "Meta Object Facility Core Specification," http://www.omg.org/spec/MOF/2.4.1, 2011.
[32] Object Management Group, "Unified Modeling Language," http://www.omg.org/spec/UML/2.5/Beta1, 2012.
[33] Object Management Group, "Meta Object Facility Query/View/Transformation," http://www.omg.org/spec/QVT/1.1, 2011.
[34] Object Management Group, "Semantics of a Foundational Subset for Executable UML Models," http://www.omg.org/spec/FUML/1.1, 2012.
[35] C. Bock, Zha, X., Suh, H., Lee, J., "Ontological product modeling for collaborative design," Advanced Engineering Informatics, 24:4, pp. 510-524, 2010.