Smart Factory Product Lines: A Configuration Perspective on Smart Production Ecosystems

Deepak Dhungana Andreas Falkner Alois Haselböck Herwig Schreiner
Siemens AG Österreich, Corporate Technology
1210 Vienna, Austria
{deepak.dhungana, andreas.a.falkner, alois.haselboeck, herwig.schreiner}@siemens.com

ABSTRACT

Smart production aims to increase the flexibility of the production processes and be more efficient in the use of resources. Two important pillars of this initiative are "smart products" and "smart factories". From the perspective of product line engineering, these can be seen as two product lines (product line of factories and product line of goods) that need to be integrated for a common systems engineering approach. In this paper, we look at this problem from the perspective of configuration technologies, outline the research challenges in this area and illustrate our vision using an industrial example. The factory product line goes hand-in-hand with the product line of the products to be manufactured. Future research in product line engineering needs to consider an ecosystem of a multitude of stakeholders - e.g., factory component vendors, product designers, factory owners/operators and end-consumers.

Categories and Subject Descriptors

D.2 [**Interoperability**]: Data mapping; J.7 [**Computer Applications**]: Computers in other systems

Keywords

Smart Factory, Product Line of Factories, Smart Product, Smart Production, Product and Production Configuration

1. INTRODUCTION

Systematic reuse approaches such as product line engineering (PLE) [39] have already proven to reduce complexity and cost - at the same time increasing quality of the product [46]. Product configuration technologies (e.g., CSP [5], ASP [36], GCSP [27], SAT [35]) have largely contributed to efficient management of highly specialized, customized products, both in the mass market (consumer goods) [49] and industrial context (engineer-to-order products) by supporting product line engineering for product portfolio management (variability) and product derivation activities [40].

SPLC 2015, July 20 - 24, 2015, Nashville, TN, USA
© 2015 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ISBN 978-1-4503-3613-0/15/07...$15.00
DOI: http://dx.doi.org/10.1145/2791060.2791066

With the increasing demand for individualized products, the need for flexible production processes, modular factories and intelligent production infrastructures is also increasing. Many research initiatives in this area aim to revolutionize the future of industrial production, e.g., Smart Manufacturing (SMLC)[1], the industrial Internet[2], Industry 4.0[3] etc.

"Smart Products" and "Smart Factories" are seen as two main pillars in making the vision of "*anytime anywhere production*" a reality [29] – which means, factories of the future are flexible and reconfigurable to support a large array of production processes and the products are smart enough to control their own production. A smart product is not only aware of its features and dependencies but also knows how it can be manufactured, thus it is aware of the requirements a factory must fulfill. Variability in the components of a smart factory and the flexibility in their production processes is a key to automated re-configuration and therefore vital for the success of the smart production vision. Equally important is the high degree of variability and customizability of the products, so that each customer can define her own variant of the product suited best to her needs. The vision is to introduce concepts of product line engineering in the production facilities – to create a market for personalized factories – to produce personalized goods in the future. From the perspective of product line engineering (PLE), two product lines are involved in tackling this problem. In contrast to traditional PLE, where one company or organization has control over the product line, the stakeholders in smart production form a complex ecosystem (cf. Figure 1) that encompasses the whole production industry.

Product Line of Goods (PLG): PLG is about the management of product portfolios through systematic reuse of domain artifacts. Traditionally, product line engineering has focused more on PLG and the production processes for manufacturing the derived products has not been considered in sufficient detail.

Product Line of Factories (PLF): PLF is about intelligent management of factory capabilities to provide self-organizing production facilities. Factories of the future must be managed as product lines – reusable domain assets being the factory components (HW and SW). Application engineering (product derivation in traditional PLE) is guided by the products to be manufactured and results in a factory configuration suited for a particular PLG or a variant derived thereof.

[1]https://smartmanufacturingcoalition.org/
[2]http://industrialinternetconsortium.org/
[3]http://plattform-i40.de/

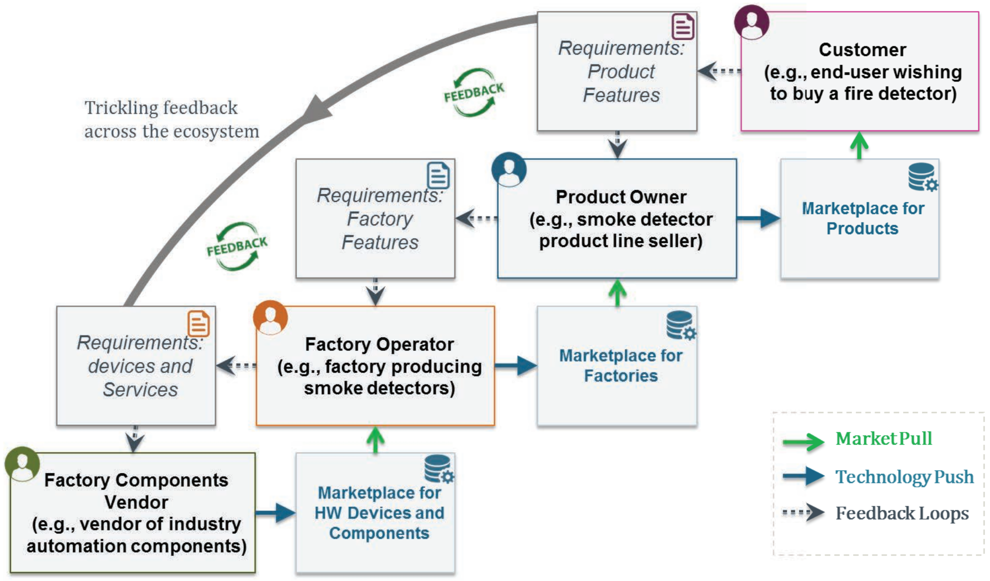
Trickling feedback across the ecosystem
FEEDBACK
FEEDBACK
Requirements: Product Features
Customer (e.g., end-user wishing to buy a fire detector)
Requirements: Factory Features
Product Owner (e.g., smoke detector product line seller)
Marketplace for Products
Requirements: devices and Services
Factory Operator (e.g., factory producing smoke detectors)
Marketplace for Factories
Factory Components Vendor (e.g., vendor of industry automation components)
Marketplace for HW Devices and Components
Market Pull
Technology Push
Feedback Loops

Figure 1: Simplified and idealistic view of stakeholder interaction and building blocks of the smart production ecosystem.

2. SMART PRODUCTION ECOSYSTEM

The smart production ecosystem is a community of stakeholders interacting with each other and with their environment (market) such that information (and more importantly *value*) is exchanged for the mutual benefit of the participants. It is a collaborative network for product design, supply chain management, production planning and logistics processes (cf. Figure 1).

Lets consider a scenario, where a customer wishes to buy a set of fire detectors[4] for a new building. Today, fire detectors can be purchased at home improvement, hardware, and large department stores and the customers' choices are limited to the different variants already available in the store. Depending on the types of sensor (smoke, flame, gas, etc.), power supply, alarm indicators, etc., different variants of the power detector are more or less suitable for different customers' requirements. Due to the explosion in the number of variants, not all combinations of all features can be pre-manufactured and kept in stock. In order to allow any physically possible combination of the fire detector features, the factory must be able to produce all of these variants efficiently, requiring a low-cost automated reconfiguration process.

The vision is that the customer will be able to customize a new variant of the fire detector and the factory of the future will be able to produce exactly the required variant. Reconfiguration of the factory is guided by the production requirements of the selected product variant. For this, the factory is seen as a product line of production capabilities – a concrete instance of which can be automatically engineered to suit the needs of the goods to be produced.

2.1 Configuration in Smart Production

As depicted in Figure 2 and described in more detail in the following, three configuration scenarios are relevant in this context.

[4]Fire detectors has been taken as an example for the sake of simplicity. This could be replaced by other products requiring more sophisticated production facilities.

Product Configuration by Customer: This configuration step is also referred to as the product derivation step in product line engineering. Based on a product line model (feature model, decision model, OVM model, CVL model or any other notation), the customer is presented with a set of decisions, on the basis of which a valid combination of the product features is generated, which represents the end-product wished by the customer.

Factory Configuration to support a Product Line: Manufacturing a new product typically means changing the physical setup of the factory (adding new devices, re-organizing the layout, defining new workflows, etc.). As the efforts of changing the physical setup is considerably high, it is not realistic that a factory can produce any kind of product at any time. Instead, a factory is configured such that it is capable of producing a family of related products. The factory components (robots, conveyers, welders, cutters, sprayers, etc.) can be seen as the domain assets, out of which one variant of the factory is generated (e.g., a factory for producing a product line of fire detectors).

Production Configuration to support a Product Instance: After the end-consumer has selected the features required for her product (e.g., flame sensor and smoke sensor in a fire detector), the factory that was previously prepared for the corresponding product line can then be configured (modified) to produce exactly the required variant. This step is typically a software configuration problem – e.g., adapting the material flow, changing automation scripts, configuring speeds, etc. Apps and services are therefore seen as the domain artifacts, which can be reused in this configuration step.

2.2 Stakeholders of the Ecosystem

In a simplified ecosystem around smart production, we pick four primary actors relevant for configuration activities and consider a production scenario of the future and the interaction between these stakeholders (cf. Figure 1).

Factory components vendor: Companies sell the components for factory automation in a market place for hard-

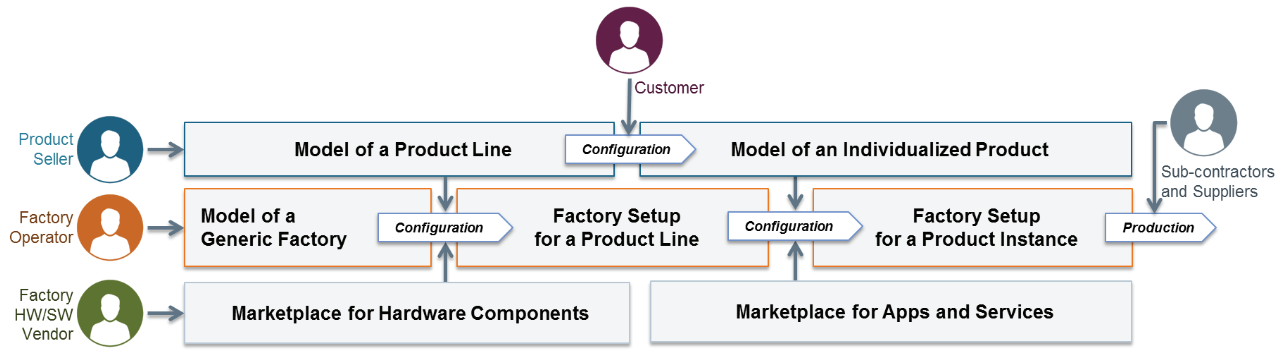
Customer
Product Seller
Factory Operator
Factory HW/SW Vendor
Sub-contractors and Suppliers
Model of a Product Line
Configuration
Model of an Individualized Product
Model of a Generic Factory
Configuration
Factory Setup for a Product Line
Configuration
Factory Setup for a Product Instance
Production
Marketplace for Hardware Components
Marketplace for Apps and Services

Figure 2: Configuration steps involved in creating a market for customizable production facilities and customizable products.

ware devices and components. Such vendors are interested in selling hardware and software components required for automating factories of the future. At the same time, the vendor is also interested in the feedback from factory operators about the new requirements arising during factory setup, so that new components, devices and applications can be developed – therefore rely on the feedback of factory operators.

Smart Factory Operator: Factory operators (e.g., an electronic devices factory in Nashville) buy automation components to set up a smart factory and make their factories available in the marketplace for factories. Such operators are interested in setting up a flexible production plant in order to be able to manufacture a wide range of products without much re-configuration overhead. At the same time, the operator is also interested in finding out about the new features required by new kinds of products – therefore reply on the feedback of product owners. Driven by theses new requirements and by operational data of the factory, the factory operator will give feedback to the components vendor. Suppliers and subcontractors (e.g., factory producing smoke sensors only) can also be seen as smart factory operators.

Product Owner: Product designers and sellers (e.g., an owner of smoke detectors product line) are interested in finding the most efficient production sites, so that the quality of the products is maintained at the same time cutting costs and increasing resource efficiency. At the same time, product owners are interested in new kinds of features to be added to their products – therefore rely on the feedback of end-users.

Customer: The end-user (e.g., somebody looking for a smoke detector) is interested in quick delivery of highly customized products. A configuration tool guides the customer through the product configuration and individualization process, guaranteeing that the final product meets her requirements, is technically feasible and producible. End-users can give feedback to product owner about the suitability of the offered features. Feedback could also be automatically derived by data analytics techniques, e.g. by mining social media product reviews or by comparing the new order with similar orders from the past.

2.3 Marketplaces

Smart production is about strengthening the economy – it is through the use of markets that the relevant stakeholders are brought together. In this paper, we use the term “marketplace” to denote a functional component of the ecosystem where demand and supply are matched. Marketplaces create economic value for buyers, sellers, market intermediaries and for society at large. Figure 1 shows the different marketplaces, as described in the following.

Market for Factory Components: Vendors of hardware and software components can offer their products and services through a market targeted specifically to operators of production facilities. This means, factory operators select their purchases from the available product offerings in a common marketplace. In the context of model-driven software product line engineering, this could be realized by publishing formal models of the component capabilities in a standardized language. For example, the vendor of a conveyer belt describes its features, its interfaces to other systems, including constraints/rules for deployment, installation and operation.

Market for Factories: In the smart production ecosystem, production facilities are also resources that can be offered in the market and sold as a service. This means, factories of the future present their capabilities to potential customers and offer them in a marketplace. Product owners (designers/sellers) can match their production requirements with the capabilities of the factories and the factory owners can eventually reconfigure existing factories to meet the requirements of products to be manufactured.

Market for Products (e.g., consumer goods): As supported by traditional product line engineering, any product owner wishing to sell her products can model these and present them to the customer in an appropriate way, where the customer can select the features of the product required to meet her requirements. For the end-consumer the market of the future has the potential to be highly flexible and at the same time more transparent, e.g., explaining why production of certain feature combinations is more costly and thus leads to a higher price of the product.

Market for Data: Just like there is a market for physical devices and services, there is also a market for data, such as relevant demographics, consumer profiles, or comparison with the known preferences of similar consumers. Such information can be collected in multiple levels of the ecosystem and be used to discover or estimate market potentials, support new pricing strategies, etc. The marketplace for data is a cross-sectional subject (and as such not depicted in Figure 1), because also operational data (e.g., performance, adherence to delivery dates, or quality indicators of factories) and factory component data (e.g., prices, short delivery times) could be subject of trading.

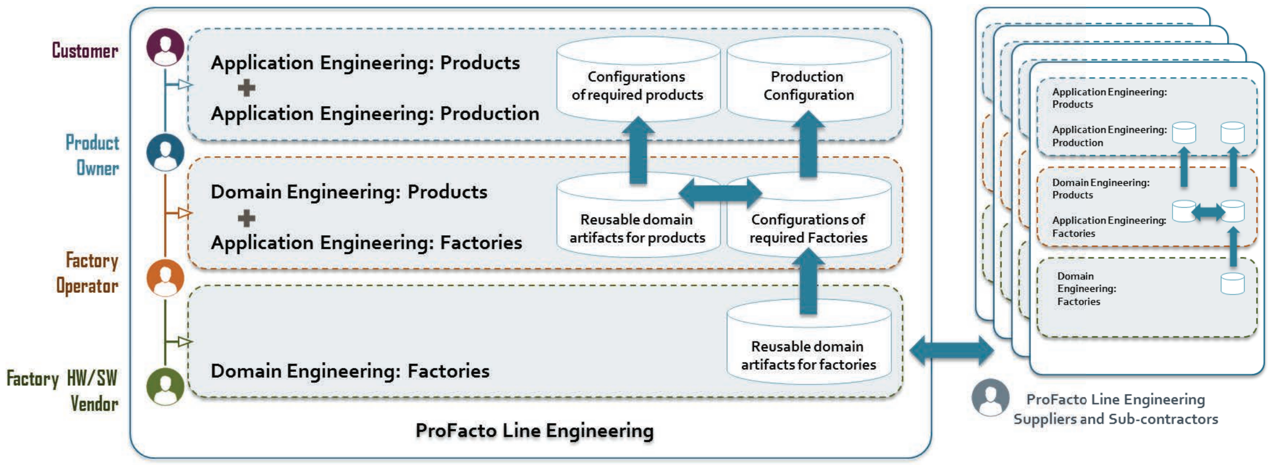
Customer
Product Owner
Factory Operator
Factory HW/SW Vendor
Application Engineering: Products
Application Engineering: Production
Configurations of required products
Production Configuration
Domain Engineering: Products
Application Engineering: Factories
Reusable domain artifacts for products
Configurations of required Factories
Domain Engineering: Factories
Reusable domain artifacts for factories
ProFacto Line Engineering
Application Engineering: Products
Application Engineering: Production
Domain Engineering: Products
Application Engineering: Factories
Domain Engineering: Factories
ProFacto Line Engineering Suppliers and Sub-contractors

Figure 3: Domain engineering and application engineering activities in the Profacto Line Engineering approach

3. VISION: PROFACTO ENGINEERING

Smart products are called smart, because they are aware of their own production process. This means, in the future, the designer of products must also consider associated manufacturing steps – an integrated engineering is necessary. We call the integrated approach of applying product line engineering principles to both **pro**duct and **facto**ry = **Profacto** Line Engineering approach (cf. Figure 3). In this section we present some initial ideas for the Profacto Line Engineering approach from the perspective of configuration activities necessary in the ecosystem.

3.1 Engineering Processes

In product line engineering, there is a clear separation of the domain engineering phase, where reusable domain artifacts are defined and developed, and the application engineering phase, where customer-specific solutions are created by systematic reuse of the domain artifacts [39]. Typically, different teams within an organization are responsible for the two activities. In the context of Profacto Line Engineering, a separation of these phases and activities still makes sense and is required for a successful reuse approach. However, these roles are distributed among the ecosystem stakeholders – beyond the control of one organization (cf. Figure 3).

Domain Engineering for Product Line of Factories: In SPLE, domain engineering is responsible for establishing the reusable platform and thus for defining the commonality and the variability of the product line [39]. In the context of smart factories product lines, the domain assets are developed by factory component vendors and made available to the factory operators through a marketplace for hardware and software components. Domain engineering activities consist of product management, requirements engineering, design, implementation and testing of the hardware and software components required for a factory setup. Modeling the components can include geometrical models of machines, variability models of devices and services, compliance to standard interfaces, and the contribution of the devices to non-functional properties of the factory, e.g, the extent of boosting productivity, reducing lead times, optimizing inventories, increasing the availability of resources or promoting flexibility. Functional description of the components such as the features of robots, assembly stations, transfer systems, test stations, or conveyer systems must also be modeled, so that these can be discovered by factory operators.

Domain Engineering for Products + Application Engineering for Factories: Smart products determine the necessary configuration of the factory. In our envisioned Profacto Line Engineering approach, product lines of goods to be produced guide application engineering of the factory. This step is referred to as Application Engineering Factories (cf. Fig 3) in order to distinguish it from a succeeding step where the factory has to be optimized for one particular instance of the product. Domain engineering for products (goods to be produced) is similar to the traditional approach followed in product line engineering - except that in smart production, product variability modeling must also consider the requirements for the factory capabilities. This means, based on the smart variability models of the products, it is possible to determine the initial layout of the factory and plan the production processes (which allow for some variations). Application Engineering Factories is thus the process in which the factories are built/reconfigured by reusing domain artifacts (factory hardware and software components). The requirements for Application Engineering for factories can be derived from product models.

Application Engineering for Products + Application Engineering for Production: Application engineering is the process of software product line engineering in which the applications of the product line are built by reusing domain artifacts and exploiting the product line variability [39]. End-users (customers) specify their requirements (either by feature selection or in some other more sophisticated forms) and the product sellers have the task of building the requested variant. For a seamless production planning, the customized product instance must be able to guide the final application engineering process for the factory. In Fig. 3, this task is referred to as Application Engineering for Production.

3.2 Modeling and Configuration View

Figure 4 depicts the different modeling and configuration activities in a Profacto Line Engineering environment.

Component Models and Factory Models: Component models are representations of production facilities like roboters, conveyers, 3D printers, sprayers, etc. Each component is a product itself, often with a complex and highly configurable hardware and software structure. Traditional, PLE modeling techniques, such as feature models or other variability modeling languages, can be used for such specifications.

A factory consists of various hardware and software components from different vendors. The specific models of the components of a factory are a main part of the factory model. Additionally, the factory model contains knowledge about the properties of the different components, their interplay, their locations, their maintenance states, etc. Thus, the factory model represents the current production capability of the factory.

In our simplified fire detector example, a factory must have available components for assembling the different kinds of sensors, for 3D-printing of the housing, for painting the housing with a user-specified color, and so forth. It shall be noted that often a factory is able to manufacture products of different product lines and kinds, so only a subset of the factory components in a very particular configuration is usually needed for a specific product line. Depending on the size of the factory, even parallel production of completely different product lines is possible.

Product Models and Product Line Specific Factory Configuration: In a traditional PLE environment, the product model represents all possible product variants, but is agnostic about the production facilities necessary for manufacturing a product instance. E.g., in our fire detector example, the product model contains specifications of all the different, offered sensor technologies, constraints about the housing shape (e.g., the 3D format of the shape specification file), or a set of possible colors for the housing.

In our Profacto Line, this model is enriched by those parts of the factory models which either influence factory configuration or which potentially reduce the variability of the product line, because the factory is not able to build certain variants of the product (e.g., 3D printers have limited size, which restricts the possible dimensions of printed objects).

The smart product line model can now be used for configuring the factory, which must be equipped and arranged in such a way that all or as many as possible different product variants can be manufactured. Such configurations will also contain process specifications which determine which parts are produced/assembled in which order and under which additional, non-functional requirements (e.g., quality gates, random sample tests, etc.).

Product Configuration and Product Specific Production Configuration: Based on the product line model, a configurator is provided and used by the consumer to customize her variant of the product. It shall be noted that, although the product model contains production knowledge, the end-user usually won't be directly confronted with production-specific decisions. Production issues will only be manifested in restrictions on the offered product variety. If there is no factory setting which supports a certain product variant, this variant must be removed from the product model or, at least, it should not be offered to the customer. So we see dependencies in both directions, from production to product and from product to production.

The product configuration will have influences on the configuration of the production site. From the underlying smart product model, a product instance knows its requirements imposed on the production. This knowledge is used, ideally in an automated way, to reconfigure the factory's production line for that product. For instance, the 3D dimensions of the specified fire detector housing may demand the usage of a special 3D printer; the selection of a specific housing color will induce the change of the paint cartridge of the paint-spray line. It is also imaginable, that special product variants and their production requirements may lead to the selection of an alternative factory.

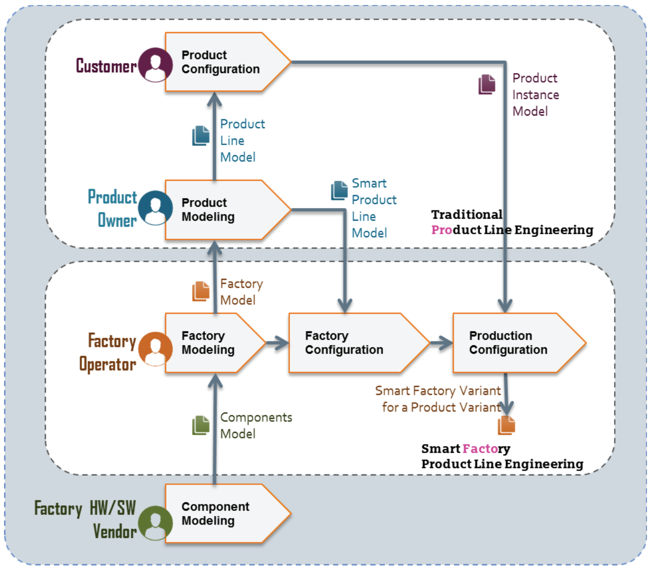

Figure 4: Stakeholder interaction through modeling and configuration in Profacto Line Engineering.

4. RESEARCH CHALLENGES

Besides the practical problems of actually reconfiguring the physical setup of factories (physical setup is out of scope of this paper), there are also many conceptual hurdles to overcome. From a configuration perspective, we summarize a set of challenges and describe how software product line engineering techniques can provide a remedy.

4.1 Modeling Smart Factory Components

In order to be part of the smart production ecosystem, a vendor publishes the capabilities of its devices and components in such as way, that these can be used by factory operators to reconfigure, update and upgrade existing production facilities. A big challenge lies in finding a good abstraction for describing the capabilities of the components, so that these are meaningful and useful for automated access to hardware. Some of the challenges and related research issues can be summarized as follows:

Distributed Knowledge Bases: The knowledge about the factory components, product features and factory operation heuristics are distributed among several participants of the ecosystem. An integrated view on the overall knowledge remains a big challenges, although some initial work in this direction have already been published (e.g., feature model merging [44]).

Inter-model (Inter-vendor) dependencies: Smart factory components are likely to come from different vendors. From a modeling perspective, the challenge lies in finding adequate notations for modeling dependencies (constraints) between products from different vendors. Some initial work in this direction has been published in [21], however, more research is needed to establish a standard way of doing this.

4.2 Modeling a Smart Factory

Ideally, a smart factory is one specific configuration of existing smart factory components – selected for one production use case. This means, modeling a smart factory is conceptually equivalent to instantiating a product from the product line of factories. However, this is only theoretically correct. Practically, the challenge is to describe existing components of a factory, so that the factory can be "offered" in the marketplace for factories – where product owners can find it, evaluate its capabilities and choose its components for the production of a specific product.

Integrated HW and SW configuration: Factories of the future are cyber-physical systems, modeling these means creating a consolidated view on both HW and SW components that go into factory setup and operation. Integrated HW and SW configuration requires consistent transfer of information along the complete engineering chain of involved technical systems - which is a challenge because of the System-of-systems [16] nature of the factories.

Modeling and configuring redundancy: A factory typically has a backup plan to survive failures due to different forms of mechanical complexity – the solution is to build redundant infrastructure. The idea is that if one fails, the others should still work. It remains a big challenge to find suitable abstractions to model this kind of "over-design" in product line engineering, which is certainly required in factories of the future.

4.3 Modeling a Smart Product

A smart product is smart because it knows its production requirements and can control its own production process in a smart factory. This means, modeling a smart product is about managing product requirements (features) and factory requirements in an integrated manner. The challenge lies in the alignment of product requirements and the factory requirements. Some of the related challenges in managing requirements across product lines is also described by Bühne et. al [10]. Some other challenges, can be summarized as follows:

Remain vendor agnostic in production: Integrated modeling of products and production means that the product modeler has to be aware of the production metaphors and capabilities of the factories. This may lead to product models, that are tightly coupled to a certain factory or devices of one particular vendor. It is therefore a challenging task for the product modeler to remain vendor agnostic during modeling. Similar topics have been addressed in [31], where an approach for supplier independent feature modeling is presented.

Intellectual property and trade secrets: Smart production requires a certain level of openness from all market players, that is to say standardization, in particular when it comes to exchanging data. However, there is a big risk in opening up all product and production data, as these are important intellectual properties of the market players. Publishing smart product models could potentially lead to competitive disadvantage. The challenge is to find new ways of modeling and sharing the product models such that digital rights management is possible. Some initial work in this area has been published by Clements et. al. [15].

4.4 Configuration of Factories

Typically production facilities are inherently flexible, i.e., reconfigurable and reprogrammable. From the perspective of smart production, it is important to leverage the flexibility of factories – the challenge lies in creating frameworks, tools and techniques for aiding the design and configuration of engineering, procurement, manufacturing and distribution activities within the smart production ecosystem.

Deriving a Product Line specific Factory: Factories of the future are generic production facilities, that can be easily adapted to the needs of the product to be manufactured. This means, before the factory can manufacture products of a product line, it has to be physically reconfigured for the specific production setting. This includes reconfiguration of the cyber-physical components of the factory [37]. The challenge we are dealing with in this paper is on the model level – and it is about generating a specific instance of the factory using the generic factory model and the product line model of the smart product.

Configuration of a Product-specific Production: Higher freedom of choices for the end-consumer means higher effort in adaptation of the factory to produce a customized product. In smart production, a challenge is to find a balance between adequate number of choices for the customer and manageable effort in reconfiguring the factory for manufacturing each customized product. From the perspective of configuration technologies, the challenge lies in automatically creating a product specific instance of the factory from a product line specific setting of the same factory.

4.5 Management of Suppliers and Contractors

Integration of heterogeneous supply chains [30] has been previously discussed, but the integration and reconciliation of the processes among the different players remains a big challenge. This may mean planning and optimization of delivery times (scheduling), selection of optimal suppliers (multi-objective optimization) or even synchronization of product release cycles.

Change Management: One of the more difficult aspects of supply chain management is trying to understand the full capabilities of the suppliers. By understanding what a supplier can do during critical times can greatly increase your response time to changes in demand. This also comes with challenges in validating the changes - because of fuzzy boundaries between the "customer" and the "vendor" within the ecosystem.

Modeling and managing service level agreements: Variability of service causes unpredictability of inventory levels and product availability in the supply chain. This means, one of the challenges for Profacto Line Engineering is to reduce variability in supply chain processes and integrating key information from the suppliers to enable predictive insights into potential issues.

4.6 Propagation of Change Requests

As depicted in Figure 1, different stakeholders in the smart production ecosystem are interested in feedback from different user groups. The challenge is to integrate these feedback cycles such that information is passed on to all relevant stakeholders in adequate manner. For example, feedback from end-users may be relevant to factory operators or even factory component vendors – trickling this information through the ecosystem for continuous monitoring and evolution of the overall system remains a big challenge.

Rapid feedback mechanisms: In distributed, parallel societal environments and markets, it is often not easy to get direct feedback – neither from the customer to the vendor nor the other way round. This makes it difficult for vendors to evaluate their services and products and at the same time, customers do not have a window to communicate with the vendors. It remains a big challenge for Profacto engineering to implement rapid feedback mechanisms in this context and incorporate the feedback for continuous improvement.

Integration of feedback cycles: Evolution and constant improvement of the ecosystem processes can only occur through intelligent integration of the different feedback cycles between different stakeholders and smart data analytics to gain insights into information passed in multi-modal communication channels.

5. RESEARCH AGENDA FOR PLE

So far, we have presented our vision of an integrated approach and the research challenges from the perspective of product configuration. In this section, we outline a research agenda for the PLE community – by identifying research areas that need close attention to make the vision of Profacto Line Engineering successful.

5.1 Exploiting Synergies with other Fields

Many problems, where the PLE community is still struggling, have been elegantly solved in other research communities and application areas. It is therefore a good idea to understand the core solution ideas in other fields and exploit the synergies for problem solving. For example, one could study the properties of biological ecosystems [11, 18] and different biological role models [17] and apply them to Profacto Line Engineering.

Learning from Software Product Line Engineering and Software Ecosystems: Traditional product line engineering provides a lot of insights into how variability of the products must be managed, which processes must be followed in a proper product line engineering organization, etc. Apart from that, many tools for modeling products [19, 3], and deriving customer specific variants [40] are already available in the market.

In the context of factory variability, some existing work in the areas of variability in services [1] and functional safety variability [42] can be taken as references. Apart from that, problems related to distributed modeling environments [16, 20] have also been dealt with by the community. Automated Merging of Feature Models Using Graph Transformations [44] is another way of dealing with models in an ecosystem context. Dhungana et al. [21] have tackled the problem of integrating multiple product lines which together constitute a multi-product line. Hierarchical product lines [7] can be applied when there is a broad family with a number of focused product categories. Similarly, Bühne et al. [10] extend an existing meta-model for variability modeling to structure product lines into a hierarchy. Bosch [8] discusses the transition from product lines to software ecosystems. A lot of formalization work in SPLE [35, 5, 14, 32] can be reused.

Learning from Mass Customization: Research in PLE has so far been performed isolated from research on mass customization [38]. While this isolation is both practically and historically rooted, we feel it will be fruitful for the PLE community to consider potential overlaps and synergies with mass customization, e.g., new customer integration techniques, modular design techniques, flexible manufacturing systems (FMSs), and supply chain management methods. Customization is already a huge trend in retail, where mass customization has solutions to many problems of the PLE community. With smart production, PLE must embrace and integrate mass customization techniques.

The role of configuration in SPLE has been well understood and is adopted by the SPLE community. Similarly, configuration plays a central role in mass customization [25], too. Successful mass customisers first find out what limits their customers are happy to live within, and then organize their operations accordingly. New reconfigurable manufacturing systems can be used to convert individually customized designs into individually customized products. The research agenda also asks for new techniques for designing products with the goal of achieving low production costs [45] and considering the logistics efforts for production. Recently, 3D printing has been a celebrated as a leap forward in mass customization. The role of 3D printing in product line engineering has been studied by Acher et. al. [2].

Technical Product Configuration: Product configuration is the derivation of an individualized product from a component inventory so that various restrictions and dependencies are satisfied and the user requirements are fulfilled. A declarative model of the domain is essential, and many modeling languages and formalisms exist and have been intensely studied over the last decades, e.g., rule-based, constraint-based, logic-based. An overview can be found in [26]. In the proposed Profacto framework, a product model (PLG) must also contain knowledge about its production, which must interplay with the factory model (PLF). Either the two models are combined during design time as described in [16], where injection mechanisms are used to generate a conjoint domain model without redundancies. Or the models are explicitly synchronized, e.g. via graph grammars [43], or graph transformations augmented by semantic technologies [6].

Configuration of cyber-physical systems has to deal with new kinds of variability, such as the variability of topological abstractions [23] and variability of runtime capabilities (aka dynamic software product lines). Apart from that, new research must also consider variability in data flow and information flows, variability of workflow and material flows through the factories of the future.

Factories of the Future: Factories of the Future[5] is a funding scheme of the EU focusing on manufacturing technologies which are environmentally and socially sustainable and highly performing. The main objectives, which are especially related to our vision of integration of product lines of products and product lines of factories, are adaptive and smart manufacturing equipment (cyber-physical systems), networked factories, and customer-focused manufacturing, linking products and production processes. Factories of the Future fit well in the vision of Industry 4.0 that smart products need smart production [24]. Three of the main design principles of Industry 4.0 [9] are crucial for a demand-driven, highly-customizable production process: modularity (adaptation of smart factories to changing requirements), interoperability in the communication of cyber-physical systems, humans and the factory, and service orientation (Internet of Services). Especially modularity is a key principle in the

[5]http://ec.europa.eu/research/industrial_technologies/factories-of-the-future_en.html

SmartFactory$^{\mathrm{KL}}$ project [34], defining a modular architecture of production facilities to make it possible that different product instances can be produced on the same production line. For that, fast reconfiguration of the production line is necessary. Also, e.g., [48, 47] emphasize that flexibility and changeability in manufacturing is a key issue for future factories. In the context of Industry 4.0 and Factories of the Future, our work envisions an integration of product models and production models from a configuration perspective. The configuration of a customized product will result in the reconfiguration of the factory equipment by synchronizing the corresponding product and production models.

Model-driven Engineering: A growing amount of companies is using model-engineering techniques (MDE) for developing and maintaining their product lines and software models. The possibilities of describing complex systems at different levels of abstraction and viewpoints seem to be especially suited for product and production models. This separation of concerns and thus, heterogeneity among the different partly overlapping models, requires an increased effort in keeping those models consistent. In complex environments, software artifacts are modeled in different languages and with different underlying meta-models. When models from different vendors of factory HW and SW components are used, the situation gets even more challenging. For the combination of or communication between those models, MDE develops model virtualization and transformation methods to support operations in a common meta-model language.

One of the most important transformation operators here is model synchronization [28]: Since one or more developers are working on the same product configurator model but may model it from different viewpoints (e.g., product viewpoint and factory viewpoint), changes or updates to models can happen concurrently and must be propagated to all other related models to solve potential inconsistencies those changes might have caused. Therefore, dedicated approaches for bi-directional model synchronization are required.

A number of model synchronization approaches have been developed a in the last years (e.g., based on Triple Graph Grammars [33] or answer set programming [13]), but there are still open research questions concerning standardization in the meta-model languages, and robustness, performance and applicability of synchronization methods in industrial environments.

Business Analytics: There are different data sources which could potentially be exploited for smart factory product lines: (1) The database of all configured products (containing also those which are finally not ordered) could be analyzed for product design to optimize product variability and complexity management. (2) Product sensor data may enable conclusions about the technical feasibility and performance of the different product variants. (3) Factory sensor data may enable conclusions about the performance of the factory configuration and the used HW and SW components. (4) Social media data (e.g., product review sites) may allow for predicting user needs on product variability. Many highly scalable machine learning and data mining methods are already available [4], but still a lot of research work has to be done to study predictive and prescriptive techniques to exploit these analytics data in the feedback loop from the product in use to the product design, the configurator performance, the test and simulation environment and the production process.

5.2 Potential Research Streams

A research stream is a direction for future research, targeted to solve a certain aspect of the overall research goal – here we identify some potential streams for driving Profacto Line Engineering.

Data Anaytics for Profacto Lines: The role of data analytics for product lines has not been studied in detail so far. This is a lost opportunity. The importance of making use of market data, user behaviour data and product usage data for better performance of recommendation engines, better user experience for users of the configurators and for increasing developer productivity throughout all stages of the application development lifecycle by offering good tool support and agile development should be studied in more detail.

With novel approaches in this direction, it should be possible for factory vendors, factory operators, product owners/sellers to organize and manage market data for better pricing and analytics and in the long run for better exploitation of data generated by different ecosystem participants. For example, (i) Better market segmentation and product portfolio management targeted to the market segments, (ii) Transparency in stakeholder interactions and ecosystem processes, (iii) Contextual personalization of marketing strategy and product placement.

Open Innovation for Profacto Lines: Product lines must evolve to meet the needs of the customers. In the smart production ecosystem, this process is best driven by the customers. By allowing customers to create real and virtual products, companies can use customers as marketers and co-creators– this is often refereed to as open innovation [12].

This means, novel approaches are required to combine (company and factory) internal and external ideas to advance the development of new technologies. Profacto Line Engineering relies on the mass for sustainable evolution. Some of the obvious benefits of open innovation for Profacto Line Engineering are: (i) A part of the product design task can be delegated to the customers. However, the implications for Profacto Line Engineering must be studied in more detail, (ii) Ideas banks and other collaborative networks can be used to gather insights into new directions for the evolution of products and factories.

Privacy Preserving Ecosystem Interactions: Product design specifications, factory capabilities and details of factory components typically represent crucial intellectual properties of their owners (product sellers, factory operators, factory vendors). For a smooth operation of the smart production ecosystem, all these entities must be formally modeled and shared (published in marketplaces) between the stakeholders. However, due to the crucial nature of the information, voluntary sharing of such models and information is very unlikely.

Future research in this area therefore need to consider privacy-preserving ways of modeling the information – so that the intellectual property rights are preserved but enough information is shared. Special audit methods can be adopted to ensure the production of intellectual property developed specially for one customer [15]. Some other examples of items in the research agenda are: (i) Encrypted sharing of models and other entities in the ecosystem and analysis operations on encrypted models (without having to decrypt),

(ii) Tools and techniques for detecting violation of digital rights management based on the analysis of published models of products, factories and factory components.

Quality Management in Ecosystems: An uncontrollable interplay of different stakeholders, tools, modeling notations, languages, processes, formats, etc makes quality management a nightmare in Profacto Line Engineering. Therefore, future research must make substantial progress in this area before Profacto Line Engineering can be the state of practice. Riedl et al. [41] have investigated the issue of quality management in service ecosystems, which can be a good starting point for research in this area.

6. SUMMARY AND CONCLUSIONS

In this paper, we presented a new engineering paradigm called Profacto Line Engineering (**Pro**duct and **facto**ry = **Profacto**) which aims to support the configuration activities in the smart production ecosystem. The goal is to apply principles of software product line engineering for an integrated approach, dealing with smart products and smart factories. Successful adoption of this vision needs to solve several challenges (cf. Section 4) along the way. Future research in this direction can tackle this problem from multiple perspectives, which are presented as opportunities for future research streams (see Section 5).

Ecosystems are not controllable, just like markets, these highly dynamic environments are likely to be influenced by a multitude of factors, not controllable by individual organizations. Future research in the area of smart production ecosystems can nevertheless identify the factors responsible for a healthy ecosystem and foster these. The distributed nature of knowledge in the ecosystem and the managerial independence of the involved parties makes Profacto Line Engineering more than the sum of two product lines to manage (product line of goods and product line of factories). In many cases, there are technical problems to be solved (e.g., distributed modeling, redundancy modeling, SLA modeling, etc). In other cases, there are still many organizational and process issues to be addressed (e.g., quality management, change management, vendor interoperability etc). Some of these problems have already been tackled in other related fields. We conclude that research in Profacto Line Engineering can largely benefit from research in related fields such as mass customization, technical product configuration, factories of the future and software ecosystems.

7. REFERENCES

[1] M. Abu-Matar and H. Gomaa. Variability modeling for service oriented product line architectures. In *Software Product Line Conference (SPLC), 2011 15th International*, pages 110–119, Aug 2011.

[2] M. Acher, B. Baudry, O. Barais, and J.-M. Jézéquel. Customization and 3d printing: A challenging playground for software product lines. In *Proceedings of the 18th International Software Product Line Conference - Volume 1*, SPLC '14, pages 142–146, New York, NY, USA, 2014. ACM.

[3] T. Asikainen, T. Männistö, and T. Soininen. A unified conceptual foundation for feature modelling. In *SPLC*, pages 31–40, 2006.

[4] R. Bartlett. *A PRACTITIONER'S GUIDE TO BUSINESS ANALYTICS: Using Data Analysis Tools to Improve Your OrganizationŠs Decision Making and Strategy: Using Data Analysis Tools to Improve Your OrganizationŠs Decision Making and Strategy.* McGraw-Hill Education, 2013.

[5] D. Benavides, S. Segura, P. T. Martín-Arroyo, and A. R. Cortés. Using java CSP solvers in the automated analyses of feature models. In *Generative and Transformational Techniques in Software Engineering, International Summer School, GTTSE 2005, Braga, Portugal, July 4-8, 2005. Revised Papers*, pages 399–408, 2005.

[6] R. Bill, S. Steyskal, M. Wimmer, and G. Kappel. On synergies between model transformations and semantic web technologies. In *Proceedings of the 8th Workshop on Multi-Paradigm Modelling (MPM) @ MODELS 2014*, pages 1–10. CEUR, 2014. Vortrag: 8th Workshop on Multi-Paradigm Modelling (MPM) @ MODELS 2014, Valencia, Spain; 2014-09-30.

[7] J. Bosch. The challenges of broadening the scope of software product families. *Commun. ACM*, 49(12):41–44, 2006.

[8] J. Bosch. From software product lines to software ecosystems. In *SPLC 2009*, pages 111–119, San Francisco, CA, USA, 2009. Software Engineering Institute, CarnegieMellon.

[9] M. Brettel, N. Friederichsen, M. Keller, and M. Rosenberg. How virtualization, decentralization and network building change the manufacturing landscape: An industry 4.0 perspective. *International Journal of Mechanical, Industrial Science and Engineering*, 8(1):37–44, 2014.

[10] S. Bühne and K. Lauenroth. Modelling requirements variability across product lines. In *13th International Conference on Requirements Engineering*, pages 41– 50. IEEE CS, 2005.

[11] Z. Chen, S. M. Dahlgaard-Park, and L. Yu. Service quality management and ecosystem theory. *Total Quality Management & Business Excellence*, 25(9-10):1190–1205, 2014.

[12] H. W. Chesbrough. *Open Innovation: The New Imperative for Creating and Profiting from Technology.* Harvard Business School Publishing, 2003.

[13] A. Cicchetti, D. Di Ruscio, R. Eramo, and A. Pierantonio. Jtl: A bidirectional and change propagating transformation language. In B. Malloy, S. Staab, and M. van den Brand, editors, *Software Language Engineering*, volume 6563 of *Lecture Notes in Computer Science*, pages 183–202. Springer Berlin Heidelberg, 2011.

[14] A. Classen, M. Cordy, P. Heymans, A. Legay, and P. Schobbens. Formal semantics, modular specification, and symbolic verification of product-line behaviour. *Sci. Comput. Program.*, 80:416–439, 2014.

[15] P. C. Clements, C. W. Krueger, J. Shepherd, and A. Winkler. A ple-based auditing method for protecting restricted content in derived products. In *17th International Software Product Line Conference, SPLC 2013, Tokyo, Japan - August 26 - 30, 2013*, pages 218–226, 2013.

[16] D. Dhungana, A. A. Falkner, and A. Haselböck. Generation of conjoint domain models for system-of-systems. In *Generative Programming: Concepts and Experiences, GPCE'13, Indianapolis, IN, USA - October 27 - 28, 2013*, pages 159–168, 2013.

[17] D. Dhungana and I. Groher. Genetics as a role model for software variability management. In *31st International Conference on Software Engineering, ICSE 2009, May 16-24, 2009, Vancouver, Canada, Companion Volume*, pages 239–242, 2009.

[18] D. Dhungana, I. Groher, E. Schludermann, and S. Biffl. Software ecosystems vs. natural ecosystems: learning from the ingenious mind of nature. In *Software Architecture, 4th European Conference, ECSA 2010, Copenhagen, Denmark, August 23-26, 2010. Companion Volume*, pages 96–102, 2010.

[19] D. Dhungana, P. Grünbacher, and R. Rabiser. The DOPLER meta-tool for decision-oriented variability modeling: A multiple case study. *Automated Software*

Engineering, 18(1):77–114, 2011.

[20] D. Dhungana, P. Grünbacher, R. Rabiser, and T. Neumayer. Structuring the modeling space and supporting evolution in software product line engineering. *Journal of Systems and Software*, 83(7):1108–1122, 2010.

[21] D. Dhungana, D. Seichter, G. Botterweck, R. Rabiser, P. Grünbacher, D. Benavides, and J. A. Galindo. Configuration of multi product lines by bridging heterogeneous variability modeling approaches. In *SPLC*, pages 120–129, 2011.

[22] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[23] A. Fantechi. Topologically configurable systems as product families. In *Proceedings of the 17th International Software Product Line Conference*, SPLC '13, pages 151–156, New York, NY, USA, 2013. ACM.

[24] T. Feld, M. Hoffmann, and R. Schmidt. Industry 4.0 - from smart products to smart production. *Information Management & Consulting*, 27(3):38–42, 2012. German.

[25] A. Felfernig. Standardized configuration knowledge representations as technological foundation for mass customization. *IEEE Transactions on Engineering Management*, 54(1):41–56, 2007.

[26] A. Felfernig, L. Hotz, C. Bagley, and J. Tiihonen. *Knowledge-based Configuration: From Research to Business Cases*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1 edition, 2014.

[27] G. Fleischanderl, G. Friedrich, A. Haselböck, H. Schreiner, and M. Stumptner. Configuring large systems using generative constraint satisfaction. *IEEE Intelligent Systems*, 13(4):59–68, 1998.

[28] H. Giese and R. Wagner. From model transformation to incremental bidirectional model synchronization. *Software and System Modeling*, 8(1):21–43, 2009.

[29] I. . W. Group. Recommendations for implementing the strategic initiative industrie 4.0. *The Industrie 4.0 Platform*, 4, 2013.

[30] H. Hartmann, M. Keren, A. A. J. Matsinger, J. Rubin, T. Trew, and T. Yatzkar-Haham. Using MDA for integration of heterogeneous components in software supply chains. *Sci. Comput. Program.*, 78(12):2313–2330, 2013.

[31] H. Hartmann, T. Trew, and A. A. J. Matsinger. Supplier independent feature modelling. In *Software Product Lines, 13th International Conference, SPLC 2009, San Francisco, California, USA, August 24-28, 2009, Proceedings*, pages 191–200, 2009.

[32] P. Heymans, P. Schobbens, J. Trigaux, Y. Bontemps, R. Matulevicius, and A. Classen. Evaluating formal properties of feature diagram languages. *IET Software*, 2(3):281–302, 2008.

[33] S. Hildebrandt, L. Lambers, H. Giese, J. Rieke, J. Greenyer, W. Schäfer, M. Lauder, A. Anjorin, and A. Schürr. A survey of triple graph grammar tools. *ECEASST*, 57, 2013.

[34] S. Hodek and F. Floerchinger. An approach for modular production from mechanics to decentralized control, realized in the smartfactory$^{\mathrm{kl}}$. In *Proceedings of 12th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2009, September 22-25, 2008, Palma de Mallorca, Spain*, pages 1–7, 2009.

[35] M. Mendonça, A. Wasowski, and K. Czarnecki. Sat-based analysis of feature models is easy. In *Software Product Lines, 13th International Conference, SPLC 2009, San Francisco, California, USA, August 24-28, 2009, Proceedings*, pages 231–240, 2009.

[36] V. Myllärniemi, J. Tiihonen, M. Raatikainen, and A. Felfernig. Using answer set programming for feature model representation and configuration. In *Proceedings of the 16th International Configuration Workshop, Novi Sad, Serbia, September 25-26, 2014.*, pages 1–8, 2014.

[37] K. Nie, T. Yue, S. Ali, L. Zhang, and Z. Fan. Constraints: The core of supporting automated product configuration of cyber-physical systems. In A. Moreira, B. Schätz, J. Gray, A. Vallecillo, and P. Clarke, editors, *Model-Driven Engineering Languages and Systems*, volume 8107 of *Lecture Notes in Computer Science*, pages 370–387. Springer Berlin Heidelberg, 2013.

[38] J. Pine. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Harvard Business Press, USA, 1999.

[39] K. Pohl, G. Böckle, and F. J. v. d. Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[40] R. Rabiser, P. Grünbacher, and D. Dhungana. Requirements for product derivation support: Results from a systematic literature review and an expert survey. *Information & Software Technology*, 52(3):324–346, 2010.

[41] C. Riedl, T. Böhmann, M. Rosemann, and H. Krcmar. Quality management in service ecosystems. *Information Systems and e-Business Management*, 7(2):199–221, 2009.

[42] M. Schulze, J. Mauersberger, and D. Beuche. Functional safety and variability: Can it be brought together? In *Proceedings of the 17th International Software Product Line Conference*, SPLC '13, pages 236–243, New York, NY, USA, 2013. ACM.

[43] A. Schürr. Specification of graph translators with triple graph grammars. In *Graph-Theoretic Concepts in Computer Science, 20th International Workshop, WG '94, Herrsching, Germany, June 16-18, 1994, Proceedings*, pages 151–163, 1994.

[44] S. Segura, D. Benavides, A. R. Cortés, and P. Trinidad. Automated merging of feature models using graph transformations. In *Generative and Transformational Techniques in Software Engineering II, International Summer School, GTTSE 2007, Braga, Portugal, July 2-7, 2007. Revised Papers*, pages 489–505, 2007.

[45] H.-E. Tseng, C.-C. Chang, and S.-H. Chang. Applying case-based reasoning for product configuration in mass customization environments. *Expert Syst. Appl.*, 29(4):913–925, Nov. 2005.

[46] F. van der Linden, K. Schmid, and E. Rommes. *Software product lines in action - the best industrial practice in product line engineering*. Springer, 2007.

[47] B. Vogel-Heuser, C. Diedrich, D. Pantförder, and P. Göhner. Coupling heterogeneous production systems by a multi-agent based cyber-physical production system. In *12th IEEE International Conference on Industrial Informatics, INDIN 2014, Porto Alegre, RS, Brazil, July 27-30, 2014*, pages 713–719, 2014.

[48] H.-P. Wiendahl, H. ElMaraghy, P. Nyhuis, M. Zäh, H.-H. Wiendahl, N. Duffie, and M. Brieke. Changeable manufacturing - classification, design and operation. {*CIRP*} *Annals - Manufacturing Technology*, 56(2):783 – 809, 2007.

[49] H. Xie, P. Henderson, and M. Kernahan. A constraint-based product configurator for mass customisation. *IJCAT*, 26(1/2):91–98, 2006.