Module and interface: Towards a cross-disciplinary understanding based on quantified product design

Kurt Sandkuhl[1,2], Ulf Seigerroth[1], Dan Lennartsson[1] and Dag Raudberget[1]

[1] *School of Engineering, Jönköping University, Jönköping, Sweden*
[2] *Institute of Computer Science, Rostock University, Rostock, Germany*

Abstract

In the interdisciplinary work of computer scientists and mechanical engineers on integrating IT components into physical products and creating products-services-systems, we observed that design and development processes cross-cutting the traditional boundaries of disciplines reveal unexpected differences in seemingly easy-to-define and understood terminology. More concretely, at first glance, the terms module and interface have the same meaning in both disciplines. Still, substantial differences became apparent when implementing design and development processes for products that extend a traditional physical product by IT-controlled functionality and customer services. Motivated by an example of quantified product design, this paper analyses commonalities and differences between the meaning of module and interface in the involved disciplines. We propose an integrative definition with the term "interface" in its focus.

Keywords

Module, Modularization, Interface, Quantified Product, Digital Transformation

1. Introduction

The digital transformation of enterprises and their business models has been subject to much research during the last decade. Digital transformation is "concerned with the changes digital technologies can bring about in a company's business model, which result in changed products or organisational structures or in the automation of processes" [1]. In this context, research areas attracting much interest are, to take only some examples, new kinds of products and services, such as smart connected products [2] and quantified products [3]. Research in this area often involves various scientific disciplines, for example, computer science, business administration, mechanical engineering, industrial organisation or social sciences. In our own research on integrating IT components into physical products and creating product-service systems, we observed in interdisciplinary work with mechanical engineers that design and development processes cross-cutting the traditional boundaries of disciplines reveal unexpected differences in seemingly easy-to-define and understood terminology. More concretely, the terms module and interface at first glance have the same meaning in both disciplines, but when implementing design and development processes for products that extend a traditional physical product by IT-controlled functionality and customer services building upon this functionality, substantial differences become clear. This motivates an investigation into the concepts of module and interface, including the definition of a joint understanding.

Motivated by an example of quantified product design, this paper analyses commonalities and differences between the meaning of module and interface in the involved disciplines. We propose an integrative definition with the term "interface" in its focus and identify required changes in the engineering process. In the field of information science, the challenge addressed is the transition from an overloaded semantic concept “module” to a concept definition accommodating and operationalising the required differences in semantics. The contributions of our work are (a) an

Companion Proceedings of the 16th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling and the 13th Enterprise Design and Engineering Working Conference, Vienna, Austria, November 28 - December 1, 2023
kurt.sandkuhl@uni-rostock.de (K. Sandkuhl); ulf.seigerroth@ju.se (U. Seigerroth); dan.lennartsson@ju.se (D. Lennartsson); dag.raudberget@ju.se (D. Raudberget)
0000-0002-7431-8412 (K. Sandkuhl); 0000-0002-5881-0669 (U. Seigerroth)

analysis of the semantics of "module" in different disciplines, (b) a conceptual model integrating the view from mechanical engineering, information systems engineering and service engineering, and (c) a proposal how to apply this conceptualisation in an engineering process.

The paper is structured as follows: Section 2 introduces the research method applied. Section 3 summarises the theoretical foundation. Section 4 contains an industrial case study from QP engineering that motivates the work. Section 5 analyses the concepts of "module" and "interface" from the viewpoint of different engineering disciplines. Section 6 proposes a conceptual model combining the different viewpoints and showing its integration into an engineering methodology. Section 7 applies the conceptual model in the industrial use case. Section 8 summarises the findings and discusses future work.

2. Research approach

This paper is part of a research process aiming at technological and methodical support for engineering quantified products (QP). It follows the paradigm of design science research [4] and concerns a design artefact developed in this project, a conceptual model expressing an integrative understanding of the concepts "module" and "interface" for mechanical engineering, information systems engineering and service engineering. The research question for this paper is: *In the context of QP engineering, how can the meaning of "module and "interface" be defined and integrated into engineering processes?* Sub-questions are (a) *What are differences between mechanical engineering, information systems engineering and service engineering in the use of "module" and "interface"? (b) What would be a common conceptual understanding of module and interface? (c) How to integrate this joint understanding into QP development?*

This paper combines two different streams of previous work: research on modularisation [5] in mechanical engineering and work on quantified products [6] in information systems engineering. The research approach used for developing the artefact is a combination of literature study, exploratory case study and prototyping. Based on the RQ, we identified research areas with relevant work and analysed the literature (sec. 3 and 5). The purpose of the analysis was to find relevant related work and examples for the cross-disciplinary use of the terms module and interface. As the literature study showed a lack of work in this field, we conducted a qualitative case study (sec. 4). According to [7] the case study in this paper is exploratory, as it is used to explore cross-disciplinary engineering work in its real-world environment. Based on the case study material and the literature, we derive a conceptual model expressing the meaning of module and interface (sec. 6). and propose its integration into the engineering process (sec. 7).

3. Background and related work

3.1. Engineering product-service systems

Product-service system is commonly used as a term describing the combination of (digital or physical) products with services accompanying them or enabled by the product. Various techniques and technologies have been identified, which can be applied to implement product-service systems. For the combination of physical products with IT components and services, development methods for cyber-physical systems (CPS) are most relevant. An overview of methodologies in this area is provided in [8]. The conclusion of this overview paper basically is that the conceptual design phase and system design phase have to be distinguished. In both phases, physical process, computation, and integration of physical process and computation need to be modeled, which essentially requires modelling approaches from the participating disciplines. From an information system (IS) perspective, the IT part can be considered as a specific category of information systems. Thus, methods for IS development can be considered as contributing to product-service system development.

The development of digital services builds on a long tradition of information systems engineering [9], business process engineering [10], capability management [11], and enterprise modeling [12]. Each of these specializations of, what might be broadly called, enterprise and information systems

engineering have developed their own strategies to deal with the need to tackle integration of requirements from different stakeholders, like those posed by the connection to physical products. Systems and service innovation is an inherently collaborative process involving different partners and stakeholders.

3.2. Quantified products

Work on QP is related to various research streams of the past years with a focus on turning data into value. Service sectors start to "collect data on everything" [13] to achieve automation and increased efficiency. Smart connected products (SCP) are not only a category of CPS products but also represent a business model category characterized by IT- and data-based services [2]. The term quantified vehicle adapts this idea for the automotive industry as a complete ecosystem for using data for data-driven services [14].

Products are often equipped with sensors, e.g., for capturing geographic positions, energy consumption, usage profiles or other information pertinent to operation and maintenance of the product. Such products also collect additional data, which could be of interest for third parties. For instance, suppliers want to know how their components are used in the field [15].

For the definition of the term quantified product, we differentiate between product data traditionally captured during product life-cycle management (PLM) and data collected by manufactured individual copies of a product, which we refer to as product instances: "A Quantified Product is a product whose instances collect data about themselves that can be measured, or, by design, leave traces of data, including operational, physical, behavioral, or environmental information for the purpose of data analysis and (optional) data sharing and services" [3].

Often enterprises quantify the products by (a) collecting data not only from a single device but the entire fleet of products operating in the field, (b) using this data for monitoring and real-time control in a management system for the fleet, and (c) offering aggregated data on marketplaces. Implementation of QP is accompanied by substantial changes in the companies' PLM of QP, as different kinds of products and their life-cycles have to be coordinated: the actual physical product; services built upon data from connected physical products and fleets of products, data-driven services using the fleet data for ecosystem services. Coordination of these different life-cycles requires both organizational, infrastructural, and technical solutions.

4. Industrial case of QP engineering

This section summarizes the design and content of a case study on QP engineering performed in the manufacturing industry. The purpose of the case study was twofold: on the one hand side we intended to study if different engineering disciplines address the task of modularizing products and services in different ways and show a different understanding of the terms module and interface. On the other hand, we wanted to analyze the actual engineering processes and the artefacts used to specify modules.

4.1. Case study design

In order to investigate potential differences between engineering disciplines in understanding what a “module” is, we had the possibility to study a developer of quantified products from the manufacturing sector. Within this enterprise, we had the possibility to collect information in several ways:

- One researcher worked during several months as part-time member of the development team for power garden products at the enterprise. The researcher maintained a work diary and collected information about the work processes, technologies used and practices by taking notes.
- The enterprise offered access to the documentation accompanying the development process (including specifications and models).

We defined two propositions P1 and P2. P1: The case shows differences between the involved engineering disciplines in the understanding of modules and modularization. P2: Artefacts documenting modularization and module specifications can be observed in the case study. To set clear boundaries for the case we defined that only data collected during the work in the development team in the organization units responsible for power garden product shall be taken into account. Not subject of the case study are other organization units of the enterprise, product and services provided to other market segments, the clients of the enterprise and sub-suppliers used for QP engineering.

4.2. Summary of case study data

The case study used in this paper is related to a research project conducted with a producer of garden and outdoor products for both, the consumer and business market (robotic and conventional lawn mowers, various sizes and types of chainsaws, garden tractors, watering systems, trimmers, and other devices). The business line "outdoor products" was the main partner in this research project that focused on the shift from products without connectivity to the Internet to smart and networked ones (QPs). The core objective of the business line was to develop an ecosystem of products and services that extends the conventional (physical) product lines and offers a common application architecture and service infrastructure. The ecosystem is meant to include different profit centers of the business line with their services and products, as well as product and services from external companies. The products in most cases have built-in electronic components implementing networking and communication capabilities where the product components connect to a common services architecture and infrastructure. The embedded systems and infrastructure services primarily are controlling the mechatronic components in the product, but also used for collecting data when the product is in operation. The data includes, e.g., status of mechatronic components, performance data and service information, or information from the product's operational environment.

The QP in focus of our investigation is the robotic lawn mower (RLM). RLMs are connected via a wireless connection to a base station installed at the point of operation. The base station in turn is connected to the Internet and the service infrastructure of the case company for providing software updates and services to the RLM. When installed at end-consumers, the RLM also provides operations data to the base station. When installed at business customers, like, e.g., garden service providers, RLMs commonly are operated as a fleet of robotic mowers. The case company provides fleet management services and the required data collection through a cloud-based platform.

In the development of the RLM, several disciplines cooperated, like mechanical engineers for the physical components (e.g., chassis, cutting unit, clutch, or wheels), electrical engineers for embedded systems and electrical parts (e.g., motor, control unit, battery, or sensors), software engineers for integration with back-office software and cloud, and business engineers for ecosystems service design and business process integration. The management set the objective of a modular design allowing for reuse of components and platform concepts for product lines. Work on the product architecture revealed differences in the understanding of the involved disciplines what a module is.

5. Different perspectives on modularization

5.1. Modularization in mechanical engineering

From a mechanical engineering perspective, modularization can make companies more competitive by enabling scale of economy, reduce costs, and improve product quality [16], and also enable companies to better adapt to changing market conditions and customer needs [17]. All modular approaches are built upon a product architecture. The importance and role of product architecture for physical products is described in [18]. Here, the core building block are the physical modules, defined as a "group of components that can be combined and integrated with other modules to create a product or a system". The goal is to organize components into standardized units that perform a specific function and that are prepared to be integrated into a larger system and can be swapped out or replaced without having to modify the entire system. Scholars have presented several modularization approaches supposed to meet industrial needs, including design with modules,

identification of modules, design of modules, modular architectures and economic perspectives on modularity.

A practical challenge has been the lack of a coherent module definition. In spite of decades of research effort, the term module is ambiguous, as [19] discuss. [20] identify the common denominator of module definitions in literature, into the definition of modular product design as the "activity of designing a product that is made up of modules". The authors further discuss the how the definition can be applied to the data-driven services in the ecosystem, where modules can be considered as groups of service components leading to a partial function of the overall service or product service system (PSS). In the Modular Function Deployment (MFD) framework [21], modules are defined as "a collection of technical solutions that perform a function, with standardized interfaces selected for company-specific strategic reasons".

The specification of interfaces is critical in modularization and include more aspects than the physical connection between components [22]. Smart connected products or quantified products are characterized by a high integration and interaction between physical elements and software and services, which emphasize the importance of interfaces, which has been add addressed by [23]. [24] further argue that the importance of interfaces grows as part of an increasing product complexity, where interfaces are enablers for both smart products and for the circular economy. Consequently, there is a need for a structured approach to both manage and design interfaces through the product lifecycle and reuse them between product architectures and generations. Operationalizing interface design is not straightforward, requiring both the action of interface design and stringent documentation of interfaces in separate documents, to highlight the importance [23].

5.2. Modularization in information system engineering

In computer science, the concept of modularization and the use of modules is one of the essential concepts in software engineering education. The seminal paper by Parnas [25] was very influential for research in this area. In software engineering, a module is a self-contained unit of functionality that can be used independently or as part of a larger software system [26]. A module is typically designed to perform a specific task or set of tasks and has well-defined inputs and outputs. The purpose of a module is to simplify software design and development by breaking down complex systems into smaller, more manageable components. Modules can be used to enhance code reusability, improve code maintainability, and facilitate teamwork by allowing developers to work on specific parts of a system independently [27]. The view of IS engineering is very close to software engineering. A module is designed to perform specific tasks or functions within a larger system. Modules can be thought of as building blocks that can be combined together to create an IS. Each module has a well-defined purpose, inputs, and outputs, and can be developed and tested independently of other modules.

An interface, on the other hand, is a shared boundary between two or more modules, components, or systems. It defines the protocols, methods, and data formats that are used to integrate and apply modules [28]. Interfaces are important because they enable different parts of a system to interact with each other, regardless of the underlying implementation. This makes it easier to replace or upgrade one component without affecting others. Interfaces are also used to enforce software architecture and design patterns, as they provide a clear separation between the implementation and the functionality.

Both modules and interfaces are essential concepts in information systems engineering because they promote modularity, encapsulation, and abstraction. Modularity refers to the practice of breaking down a complex system into smaller, more manageable components. In software engineering, encapsulation refers to the practice of hiding the implementation details of a module or component from the outside world. Abstraction refers to the practice of presenting a simplified interface or model of a complex system. These practices make it easier to design, develop, test, and maintain software systems, and are key factors in ensuring software quality and reliability.

5.3. Modularization in service engineering

In the service sector, the work of Simon has been very influential on service modularity concepts and approaches. Simon addresses the possibility of reducing complexity or organizational tasks by breaking them down into units [29]. The main idea of modularity is that a complex system can be built from smaller parts (modules), which can be designed, developed and improved independently, but still function together as a whole [16]. The concept of modularity is generally considered to be a suitable solution for controlling complexity without limiting applicability of the product as it promises a wide market coverage with limited additional costs [30].

Modules of services are defined by [31] in the context of service modularity as "a system of components that offers a well-defined functionality via a precisely described interface and with which a modular service is composed, tailored, customized, and personalized". Closely connected to service modules and components is the topic of interfaces. Interfaces between service modules control the communication and information flow between the modules [32] by defining a set of rules and guidelines. Interfaces can be directed towards service content or service package [32]. Service modules can contain the 'technical' service contents from which to build up the complete service offering. Interfaces are required to connect individual modules and manage possible interactions between their contents. Service packages contains all modules desired by a particular customer. Interactions and linkages within the service package are required to be able to connect the various providers involved in service delivery and allow them to exchange information about customers.

"Interfaces on the level of components are expected to manage content interactions and therefore are concerned with the flow of service customers. Interfaces on the level of the service package are expected to manage service provider interactions and therefore are concerned with the flow of information. [32]". The work by [33, 34] contains an overview about services modularity frameworks and proposes a classification of existing work along the runtime and build-time dimension and the dimension of modularization objective (efficiency-oriented or market-oriented). In the context of our work, this classification is in particular interesting as it in part is based on work from mechanical engineering emphasizes the relevance of relating concepts in both disciplines.

6. Towards an integrated concept of module and interface

The discussion in section 5 made clear that there is a lot of common ground between the disciplines, but also some differences. Some of the important commonalities are: aim of modularization in all disciplines is to reduce complexity by dividing the whole in smaller "units" and allow for reuse of these units; in all disciplines, modules have interfaces; motivations for modularization include economic (better fit for customer, higher efficiency by configurability), technical (standardized way of documentation/configuration) and process aspects (simplify testing, variant creation). Examples of differences are that interfaces in physical products can have to be implemented as components whereas software engineering and service engineering clearly consider interfaces as specification only to be implemented in modules. Furthermore, the aspects to be defined in a module interface are quite different.

In the industrial case of QP development it became clear that a joint understanding between the involved disciplines of what a module is and how to specify an interface would be valuable as QP include modules of different disciplines (e.g., a physical module and a service) and interfaces to modules of QP can include different kinds of interfaces. Design with modules and design of modular products from economic and customer perspective would be eased with this joint understanding.

In section 6.1, we present a conceptual model that establishes a joint understanding of module, interface and related terms for the disciplines. Section 6.2 shows how to integrate the conceptual model into engineering processes and section 6.3 into the modelling tools supporting the engineering processes.

6.1. Conceptual model

Fig.1 shows the initial version of our conceptual model, modelled in UML, that consists of concepts (classes) and specialization, composition and aggregation relations. At the core of the conceptual model are the concepts of module and interface. The definition for the term module of [16] seems to be accepted in all three disciplines: "a unit whose internal structural elements are powerfully connected and is relatively weakly connected to elements in other units". The module is modelled as a concept that consist of the module definition that can be specialized into module types. With this mechanism, specific module types for physical products, services or information systems can be defined that share a common conceptual ground but accommodate the specifics of the discipline. Conceptually, a product consists of modules. However, as the same product can have different configurations (for example, for customer groups), we also need the concept of module variants, i.e. module variants are required to allow for the configuration of modules in a product. The second essential term in the conceptual model is interface. We define interface based on "interface" and "interface specification" in [28] as "description of essential characteristics, requirements and constraints at a common boundary between two or more modules". The interface concept in Fig. 1 is an abstract concept that needs specialization. The specialization reflects the different interface types identified in 5.1, 5.2 and 5.3, i.e., API, call, and protocol for IS modules; command & control, field/environment, spatial, transfer and use interfaces for physical product modules; and service provider and service communication interfaces for service modules. As interfaces can be solitary or composed of several interfaces, we used the concept of interface specification as composite element. Interface is related to module types via the concept of module definition.

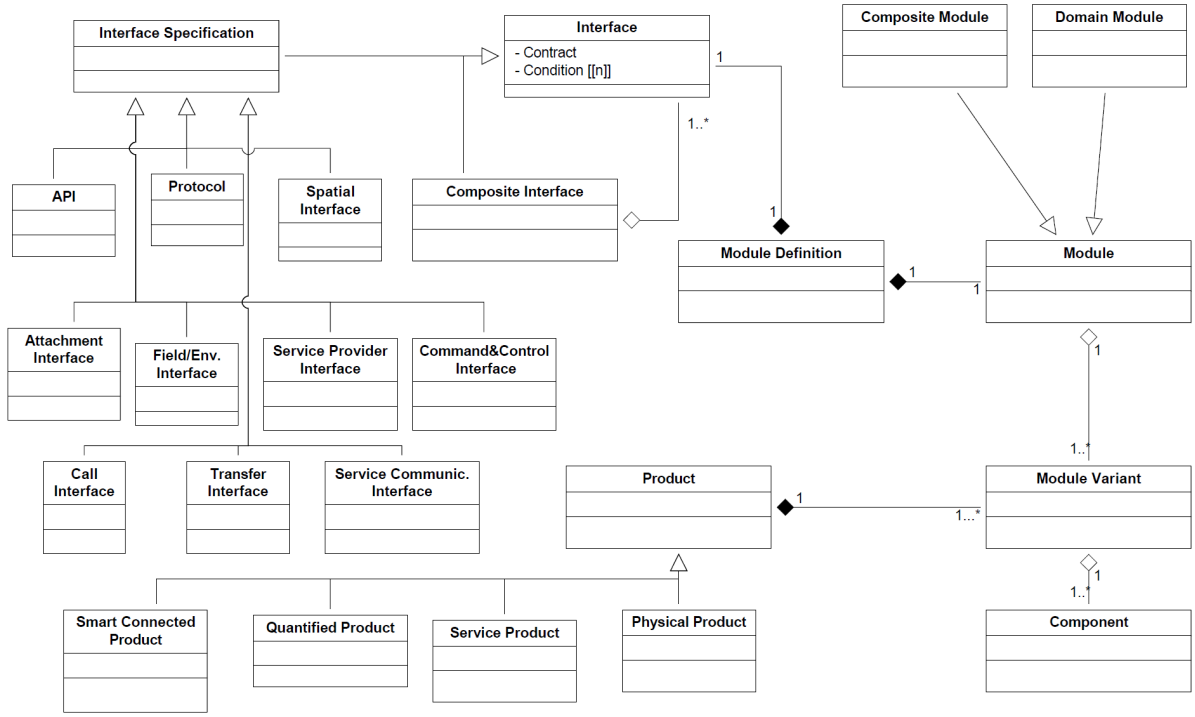

Figure 1: Common conceptual model for module and interface for different disciplines

6.2. Integration into engineering process

Engineering processes of products and services involving different disciplines and module types often consist of specific processes for the individual engineering disciplines and an overarching process managing the integration of discipline-specific tasks. The common understanding of “module” allows for the definition of the overall product and service modularization on the level of the overarching process. How this might affect the engineering process is discussed using the example of QP engineering.

For the engineering of QP, the integration of three lifecycles has been observed [6]: the lifecycle of the physical product, of the data-driven services and the ecosystem consisting of the business services. The three lifecycles are managed by a common QP management process. The models and information developed in these lifecycles are available in a joint repository. The conceptual model presented in Fig. 1 can be applied in and improve all three lifecycle phases, as the common understanding of modules allows for an overall modularization of the QP that includes the modularizations of the physical product, IS systems (data-driven services or components deployed on the physical product), and services offered. Furthermore, the modelling of requirements and specifications, i.e., the early phases of engineering can use the joint conceptualization for producing aligned models. It should be noted that the definition of modules in all disciplines has to include interface specifications which lead to composite interfaces in case of modules integrating modules from different disciplines. The composite interface specifies the dependencies of the included modules' individual interfaces.

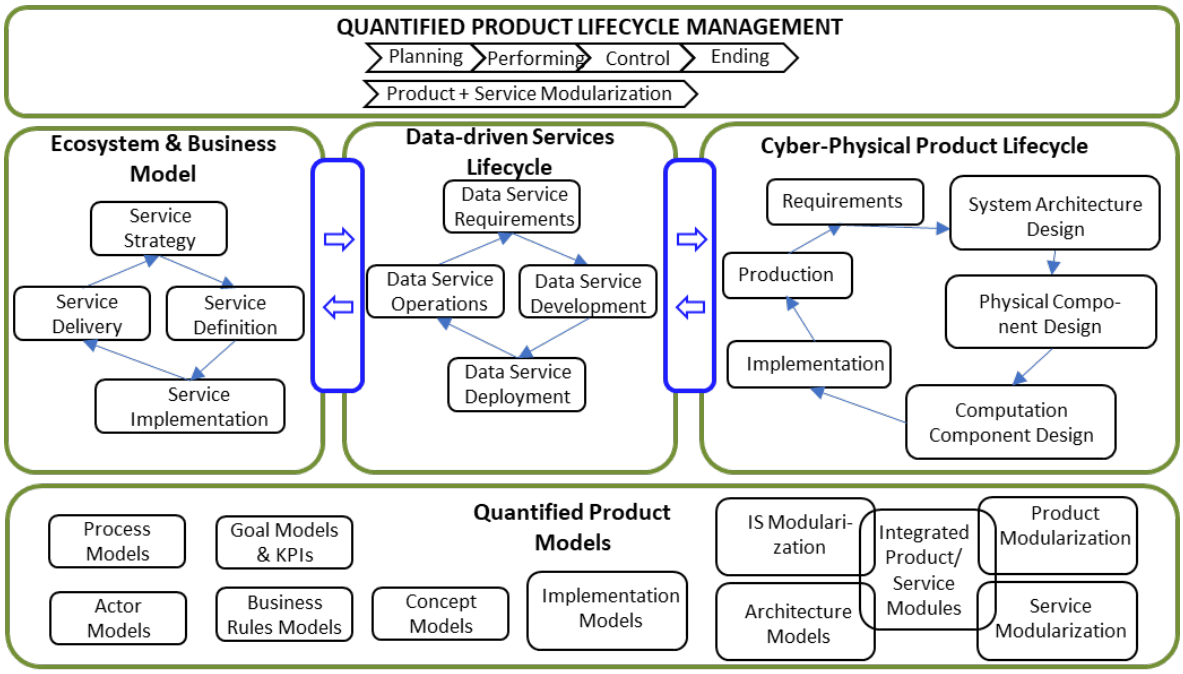

Figure 2: QP Lifecycle Processes

Figure 2 shows the three lifecycle models and their integration. On the level of the (overarching) QP lifecycle management, the joint product and service modularization process for all engineering disciplines involved has been added. The integrated product and service modularization model is a new artefact in the QP models repository that interlinks the specific modularizations for physical product, IS and services.

In addition to the integration into lifecycle processes described above, integration into modelling methods supporting the lifecycle process is important to ensure applicability of the conceptual model in the practice of engineering QP. Our proposal is to integrate the conceptualization into the enterprise modelling (EM) method 4EM [12] that can be used to model the ecosystem, the services provided and the requirements and conceptual models for the data-driven services and cyber-physical part of the QP. A key aspect in EM is the integrated view on the various aspects of the enterprise. An enterprise model therefore consists of a set of interlinked sub-models, each of them focusing on a specific aspect like processes, goals, concepts, business rules. 4EM shares many underlying principles of the, so called, multi-perspective, approaches that recommend to analyze organizational problems from several perspectives, such as vision (goals, objectives), data (concepts, attributes), business processes (processes, tasks, activities), organizational structure (actors, roles, organizational units).

In 4EM, the conceptualization of modules and interfaces could be operationalized as an extension of the product/service sub-model (PSM). Our approach is to integrate the "module" concept into the

PSM meta-model by (a) changing the “component => part_of => product” path into “component => part_of => module variant => part_of => product”. The modified path also exists in the same way in the conceptual model; product and component also have the same meaning in PSM as in the conceptual model, (b) integrating all other concepts from the conceptual model into the PSM model using the relations to module given in the conceptual model and adding them, (c) extending the visual modelling language (notation) of 4EM accordingly.

6.3. Integration into engineering tool environment

The use case described in section 3 indicated that tool support applicable by all engineering disciplines would ease the introduction of the common conceptualization of module and interface. From a functional perspective, tool support would have to cover

- the actual development of the modularization for a product with modules for different disciplines including the definition of interfaces,
- the storage or linking of all documents related to a module (e.g., interface specification, functional description, implementation documentation), and
- retrieval/search possibilities in modules and interfaces to support reuse and new product designs.

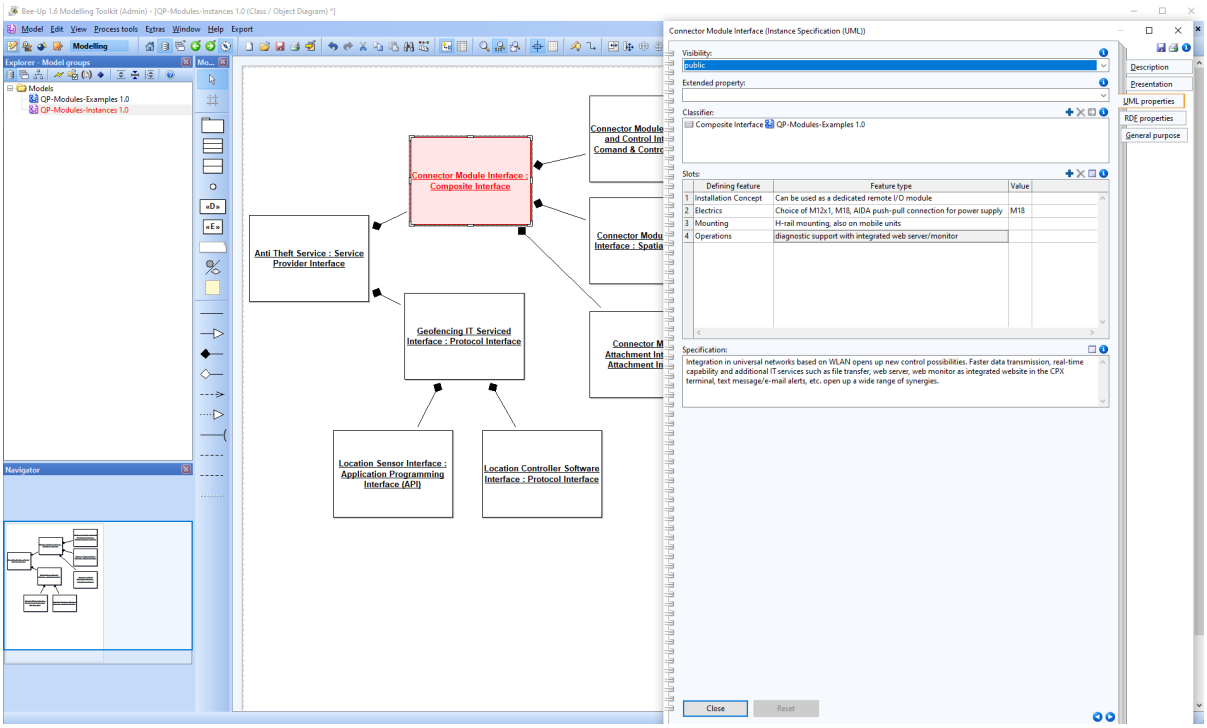

Figure 3: Object diagram incl. notebook for connector component (screenshot BEE-Up tool)

For the prototype implementation of such a tool we decided to use the BEE-Up[2] tool in combination with a knowledge graph based on GraphDB[3]. The BEE-Up tool supports different modelling languages, such as UML, ERD or BPMN. We captured the meta-model presented in Figure 1 as UML class diagram. Based on this class diagram, modelling of an actual module structure and the corresponding interfaces is possible using UML object diagrams associated to the class diagrams. Class diagram and object diagram are interlinked and provide the required functionality for development of the modularization. BEE-Up also offers the possibility in the “notebook” of each

[2] https://bee-up.omilab.org/activities/bee-up/
[3] https://www.ontotext.com/products/graphdb/

model element to add attributes, create links to related resources, and define associations to other models. This functionality can be used to implement the required linking of all documents related to a module. Figure 3 shows an excerpt of the object diagram for the use case including the notebook entry for the composite interface of the connector component. BEE-Up models with all information included in the notebook can be exported to RDF. RDF in turn can be imported into a triplestore, such as GraphDB, with the possibility to navigate the modules, interfaces and linked documents (see sec. 7 for an example). This is a simple way to implement a prototype with navigation possibilities.

7. Concept evaluation in the case study

The integrated concept of module and interface was evaluated in the case study from section 4. We used the case study for an initial validation of both, the applicability of a conceptual model and the process and tool integration. The RLM-related product and service integration used for this purpose was the anti-theft service. If an RLM leaves a geographical area this is detected and interpreted as theft or theft attempt. For business customers this service is offered as actual tracking service, where a security service provider traces the devices in order to recover them. For end consumers, this service is designed to send an alarm notification to the owner. From a modularization perspective, the alarm service for the end consumer requires a location sensor module in the device, the connector module connecting the device via base station to the cloud, the location controller software model, and the geofencing IT service. In the version for businesses, additionally the IT-Services modules “notify security-company” and “provide-supervision” are required. An excerpt of the modularization is presented in Figure 3.

In the case study, we used IDL specifications for software module interfaces, WSDL for defining IT service interfaces, a company-internal service description template, and a combination of CAD, BOM and fact sheet for specification of the physical interface. The actual interface specifications are stored in the company Intranet and work areas usually used of the engineering discipline responsible for the module. Integration of the different module interfaces and related documents is done via the information stored in the attributes of the individual modules and their interface (i.e., the BEE-Up notebook). A knowledge graph providing access and navigation possibilities can serve as access point to this information in the enterprise.

When using the common modularization terminology in the use case company, we observed that the tool support described in section 6.3 should be complemented by an extension of the enterprise modeling tool. Before working on actual product, service and IS modules, the business model integration of products and services has to be finalized which is done by business-oriented stakeholders within the business model lifecycle depicted on the left side of Figure 2. The business-oriented stakeholders do not work with modules and interfaces but with products and services, and they use other modelling tools. Thus, we extended our modeling language 4EM.

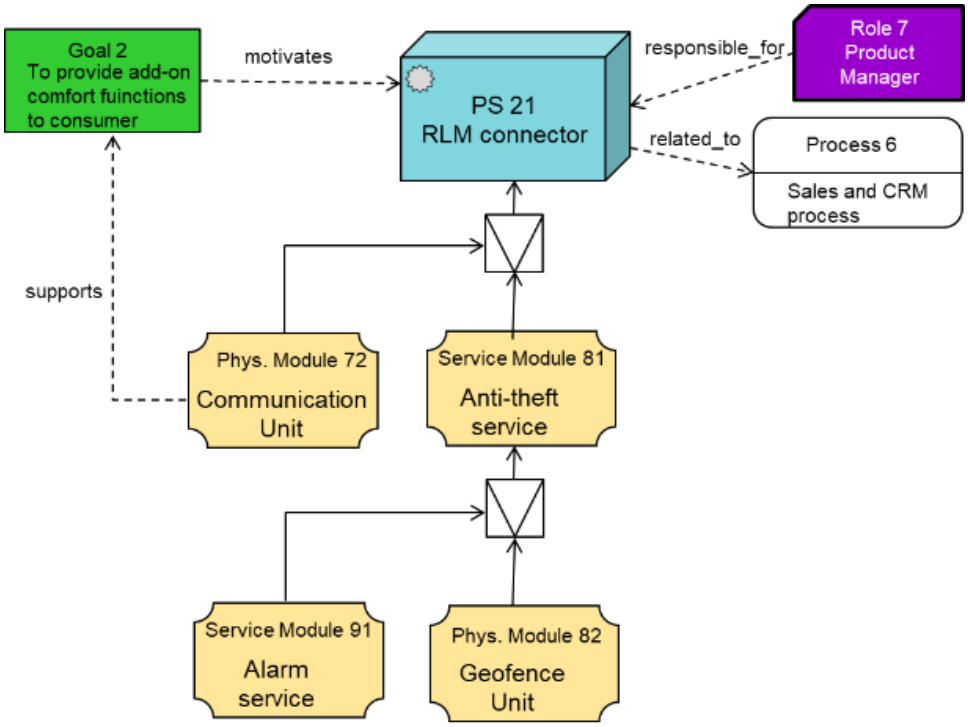

Figure 4: Example 4EM model with new concept "module" included

Based on the anti-theft service example, Figure 4 demonstrates the use of the module concept after its integration into the PSM. For the robot lawn mower (RLM), there is an add-on product, called "connector", that is part of the anti-theft service and connects the RLM to the Internet, which requires the module "communication unit". With this connector, an anti-theft service can be implemented. The anti-theft service module requires a combination of service (alarm service) and physical module (geofence unit). From a 4EM perspective, the visual language of PSM in this example has a new symbol representing all types of modules. The model excerpt in Figure 4 also demonstrates the integration with other 4EM sub-models, like the actor and resource model (role "product manager" as responsible actor for the product RLM connector), the process model (process "sales and CRM") and the goal model (goal "to provide add-on comfort-functions" that motivates the RLM connector product.

It should be noted that the model excerpt originates from an early phase of QP modelling. The 4EM approach tolerates relaxing the constraints on concept use in early modelling phases, which allows for applying module instead of module variant. Module variants would have to be decided on in later modelling phases when production constraints are integrated into the model. For the support of enterprise modelling with 4EM when planning adaptations on business model level, we plan to incorporate the changes in 4EM's product service model in the 4EM tool[4].

8. Summary and future work

Motivated from experiences in a case of quantified product design, the paper aimed at investigating differences between different disciplines in modularization and the use of module and interface. For QP and other smart connected product, the cooperation between mechanical engineers, software or information systems engineers and experienced service developers is mandatory. This cooperation can be eased if modularization concepts and model reuse works across the disciplines. A common understanding expressed in a conceptual model is a foundation for this.

For physical products, [5] conclude that a clear alignment with the business goals is crucial for achieving a product-strategic modularization that lets the resulting modular system reach its full potential. Based on our experience, we claim that this view is applicable also to software engineering

[4] https://austria.omilab.org/psm/content/4em/download

and service engineering domains. Methods and tools developed for designing the business case, setting goal values for product/system features and interface alignment should therefore be used in all three domains.

Our current work has a number of limitations. The data presented in section 4. were only collected in two projects of one domain, i.e., they are not representative. The conceptualization was tested only in one case and has to refined and revised. Both will be part of future work. Furthermore, tool support for module management in repositories with retrieval and version management mechanisms need to be investigated.

References

[1] T. Hess, C. Matt, A. Benlian, and F. Wiesböck, "Options for formulating a digital transformation strategy," *MIS Quarterly Executive*, vol. 15, no. 2, 2016.

[2] M. E. Porter and J. E. Heppelmann, "How smart, connected products are transforming competition," *Harvard business review*, vol. 92, no. 11, pp. 64–88, 2014.

[3] K. Sandkuhl, "Features of Quantified Products and Their Design Implications," in *International Baltic Conference on Digital Business and Intelligent Systems*, 2022, pp. 152–163.

[4] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS quarterly*, pp. 75–105, 2004.

[5] D. Lennartsson, D. Raudberget, K. Sandkuhl, and U. Seigerroth, "Modularisation Metrics-Contrasting Industrial Practice and State-of-Research," *Proceedings of the Design Society*, vol. 2, pp. 2483–2492, 2022.

[6] K. Sandkuhl, N. Shilov, U. Seigerroth, and A. Smirnov, "Towards the quantified product-product lifecycle support by multi-aspect ontologies," *IFAC-PapersOnLine*, vol. 55, no. 2, pp. 187–192, 2022.

[7] R. K. Yin, "The abridged version of case study research," *Handbook of applied social research methods*, vol. 2, pp. 229–259, 1998.

[8] P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, and S. Achiche, "Design, modelling, simulation and integration of cyber physical systems: Methods and applications," *Computers in Industry*, vol. 82, pp. 273–289, 2016.

[9] M. Snoeck, "Enterprise information systems engineering," *The MERODE Approach*, 2014.

[10] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, "Business process management: A survey," *Business process management*, vol. 2678, no. 1019, pp. 1–12, 2003.

[11] S. Bērziša *et al.,* "Capability driven development: an approach to designing digital enterprises," *Business & Information Systems Engineering*, vol. 57, pp. 15–25, 2015.

[12] K. Sandkuhl, J. Stirna, A. Persson, and M. Wißotzki, *Enterprise modeling*: Springer, 2014.

[13] V. Mayer-Schönberger and K. Cukier, *Big data: A revolution that will transform how we live, work, and think*: Houghton Mifflin Harcourt, 2013.

[14] C. Kaiser, "QUANTIFIED VEHICLES: DATA, SERVICES, ECOSYSTEMS," 2022.

[15] P. Farahani, C. Meier, and J. Wilke, "Digital supply chain management agenda for the automotive supplier industry," in *Shaping the digital enterprise*: Springer, 2017, pp. 157–172.

[16] C. Y. Baldwin and K. B. Clark, *Design rules: The power of modularity*: MIT press, 2000.

[17] R. N. Langlois, "Modularity in technology and organization," *Journal of economic behavior & organization*, vol. 49, no. 1, pp. 19–37, 2002.

[18] K. Ulrich, "The role of product architecture in the manufacturing firm," *Research policy*, vol. 24, no. 3, pp. 419–440, 1995.

[19] J. K. Gershenson, G. J. Prasad, and Y. Zhang, "Product modularity: definitions and benefits," *Journal of Engineering design*, vol. 14, no. 3, pp. 295–313, 2003.

[20] F. Borjesson, R. Fancher, and U. Sellgren, "On a methodology for component selection in modular branding: An industrial pilot study," *Concurrent Engineering*, vol. 22, no. 2, pp. 93–105, 2014.

[21] G. Erixon, *Modular function deployment*, 1998.

[22] D. V. Steward, "The design structure system: A method for managing the design of complex systems," *IEEE transactions on Engineering Management*, no. 3, pp. 71–74, 1981.
[23] B. Bettig and J. K. Gershenson, "The representation of module interfaces," *International Journal of Product Development*, vol. 10, no. 4, pp. 291–317, 2010.
[24] D. Lennartsson, D. Raudberget, U. Seigerroth, and K. Sandkuhl, "An Approach Towards Operationalization of Modularization Interfaces for Industrial Product Development," 2022.
[25] D. L. Parnas, "On the criteria to be used in decomposing systems into modules," *Communications of the ACM*, vol. 15, no. 12, pp. 1053–1058, 1972.
[26] A. Abran, J. W. Moore, P. Bourque, R. Dupuis, and L. Tripp, "Software engineering body of knowledge," *IEEE Computer Society, Angela Burgess*, vol. 25, 2004.
[27] I. Sommerville, *Engineering software products*: Pearson London, 2020.
[28] I. S. ISO, "IEC/IEEE 24765: 2010 systems and software engineering-vocabulary," *IEEE Standards Association*, vol. 24765, p. 418, 2010.
[29] G. J. Klir and H. A. Simon, *The architecture of complexity*: Springer, 1991.
[30] S. Pekkarinen and P. Ulkuniemi, "Modularity in developing business services by platform approach," *The International Journal of Logistics Management*, 2008.
[31] T. Tuunanen, A. Bask, and H. Merisalo-Rantanen, "Typology for modular service design: review of literature," *International Journal of Service Science, Management, Engineering, and Technology (IJSSMET)*, vol. 3, no. 3, pp. 99–112, 2012.
[32] C. de Blok, B. Meijboom, K. Luijkx, J. Schols, and R. Schroeder, "Interfaces in service modularity: a typology developed in modular health care provision," *Journal of operations management*, vol. 32, no. 4, pp. 175–189, 2014.
[33] A. Lubarski, "Understanding service modularity-antecedents, processes, and operationalization," Universität Bremen, 2019.
[34] J. Pöppelbuß and A. Lubarski, "A classification framework for service modularization methods," *Enterprise Modelling and Information Systems Architectures (EMISAJ)*, vol. 13, 14-1, 2018.