Web-based solution to automate capability matchmaking for rapid system design and reconfiguration

Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 81 (2019) 288–293

www.elsevier.com/locate/procedia

52nd CIRP Conference on Manufacturing Systems

Web-based solution to automate capability matchmaking for rapid system design and reconfiguration

Anant Mital[a], Niko Siltala[b,*], Eeva Järvenpää[b], Minna Lanz[b]

[a]*Faculty of Information Technology and Communication Sciences, Tampere University, Korkeakoulunkatu 1, 33720 Tampere, Finland*
[b]*Faculty of Engineering and Natural Sciences, Tampere University, Korkeakoulunkatu 6, 33720 Tampere, Finland*

* Corresponding author. Tel.: +358-40-536-6017. *E-mail address:* niko.siltala@tuni.fi

Abstract

Modern manufacturing companies desire rapid responsiveness from their production systems, in order to operate efficiently in highly dynamic environments. There is a need for design tools, which can support production system design and reconfiguration by providing automatic matchmaking between product requirements and resource capabilities. Such matchmaking is currently time consuming and heavily dependent upon designer's experience. This paper introduces a prototype of a web-based software service, which carries out this matchmaking task automatically, and how it is designed to meet the requirements of manufacturing industry. Software engineering process has been followed for the development. We also describe a case example, which illustrates how the production system designer can interact with matchmaking activity through its web service interface. We expect that web service-based approach to matchmaking will reduce the technical barrier for adoption by the manufacturing industry as existing planning and reconfiguration systems can utilize the service with small efforts.

Keywords: information model; capability matchmaking; REST web service; capability-based matchmaking; reconfiguration

1. Introduction

The requirements for production systems and networks are continuously shifting towards higher flexibility, making rapid responsiveness a new strategic goal for manufacturing enterprises along with quality and cost-effectiveness [1]. There is a need for production systems that can rapidly adapt to the required changes in processing functions, production capacity, and dispatching of orders. Currently, the system design and reconfiguration planning are manual processes, which rely heavily on the designers' expertise and tacit knowledge to find feasible solutions by comparing the characteristics of the product to the technical properties of the available resources. This slow process sets limitations to the amount of potential configuration alternatives that can be considered. Meeting the requirements of fast adaptation calls for new solutions, such as formal information models representing resources and products, as well as computer-aided intelligent planning methods and tools, that would drastically reduce the time and effort put into system design, both in brownfield [2] and greenfield scenarios.

Within the past decade, there have been multiple different projects trying to provide computerized support for the reconfiguration planning process. The recently finished European Union funded project ReCaM [3] aimed to develop a set of integrated tools for rapid and autonomous reconfiguration of production systems. The approach relies on a formal unified functional description of resources, providing a foundation for rapid creation of new system configurations through capability-based matchmaking of product requirements and resource offerings.

This paper, building upon our past publications relating to formal information models and ontologies [4, 5], introduces a web-based software solution, which utilizes these information models, capability matching algorithm and other software tools to automate the capability matchmaking process. We expose

10.1016/j.procir.2019.03.050

the functionality of this tool through a RESTful web service [6], which can be easily coupled with any external production system design and reconfiguration software to execute capability matchmaking based on provided inputs. In the following sections, we will detail on requirements, design and architecture decisions, technology, and development tools used for developing the web component of capability matchmaking system. We will also briefly describe our past research related to formal information models and ontologies, which have gone into development of the system.

The paper is organized as follows. Section two introduces shortly the capability matchmaking approach and its associated concepts. Section three focuses on the development methodology of the web service. In section four we demonstrate an industrial use case of capability matchmaking and in final section we present the conclusions.

2. Capability Matchmaking

The matchmaking system intends to ease up the production system design and reconfiguration procedure by automatically suggesting alternative resource combinations for specific product requirements. The matchmaking utilizes formal representation of product requirements as well as resources and their capabilities and interfaces as input, and tries to make a match between these by using rule-based reasoning. We shortly explain these aspects in the following sub-sections.

2.1. Involved information models

We have defined several ontologies as connected information models [4]. Central model of them is Manufacturing Resource Capability Ontology (MaRCO) [4], which is a Web Ontology Language (OWL)-based information model that can be used to describe capabilities, i.e. functionalities, of resources and resource combinations. MaRCO imports another ontology called Process Taxonomy Model, which categorizes different manufacturing and assembly processes in a hierarchical structure. MaRCO model defines relations between simple (atomic) and combined capabilities. For instance, robot has a simple capability “Moving” and gripper has a simple capability “Grasping”. Together they have a combined capability “Transporting”. Based on these relations, the potential device combinations that have a certain combined capability can be identified programmatically by utilizing information provided by SPARQL (SPARQL Protocol and Resource Description Framework Query Language) queries. Detailed information about MaRCO can be found from our earlier publications [4,7].

While MaRCO is used to represent the resource capabilities, another model is needed to describe the product requirements. For that, we have developed a Product Model ontology, which was presented in detail in [5]. The Product Model describes the parts and their basic characteristics, sub-assemblies and their contained parts, processes related to the parts and sub-assemblies, capability requirements related to the processes, and sequence of the processes. Also, the Product Model imports the same Process Taxonomy as the Capability Model. This allows to build a link between the requirements and provided capabilities.

2.2. Matchmaking viewpoints and rules

The overall matchmaking process [8] has three stages, which all require their specific algorithms and rules: 1) Defining the combined capabilities and calculating their parameters when new resource combinations are formed; 2) Checking the interface compatibility of the resources when new resource combinations are formed; and 3) Matching the product requirements against the capabilities of the combined resources. We have discussed the combined capabilities and their parameter calculation in [9], the interface matchmaking in [10], and the capability matchmaking in [11]. All these three aspects are included into the operation chain of the capability matchmaking software.

For rule implementation we use SPIN (SPARQL Inferencing Notation). SPIN is a World Wide Web Consortium’s (W3C) Member Submission that has become the de-facto industry standard to represent SPARQL rules and constraints on Semantic Web models [8]. SPIN can be used to link class definitions with SPARQL queries to capture constraints and rules that formalize the expected behavior of those classes. A suitable reasoner tool such as SPIN Application Programming Interface (API) can then infer the extra information created by the rules and use it for example in SPARQL query execution [9]. SPIN is used both in the combined capability inference [9] and capability matchmaking [11].

2.3. Matchmaking inputs and procedure

The matchmaking requires as an input the Product Requirement Description (PRD) and the Resource Descriptions of the resources (Resource Pool) that should be included into the matchmaking. In case of reconfiguration scenario, the existing system description (System Layout) should also be provided as an input. These inputs form the search space for the matchmaking. The search space is read into the Matchmaking Ontology. Matchmaking Ontology imports both the MaRCO and the Product Model ontologies and contains the SPIN rules that are used to compare the product requirements against the provided capabilities, and to make match between those.

These inputs are provided to the matchmaking software by external design and planning systems, which control the matchmaking process. The resource information is collected from a Resource Catalogue(s), where resource providers have provided descriptions of their offerings in the Resource Description format [12].

The capability-matching algorithm takes the capability requirements and match them with the existing capabilities or create new resource combinations that match with the requirements. The found matches to each process step are then provided back to the external design tools, which will then make the decision about resource selection and system configuration based e.g. the availability and other valued criteria.

3. Development of Capability Matchmaking Web Service

Capability matchmaking web service is the prototype implementation of automatic matchmaking for production system design and reconfiguration described in the previous section. It provides a web interface to the capability matchmaking algorithm, which in conjunction with formal information models produces matchmaking results – suitable resources or resource combinations for a specific production step. The capability matchmaking web service by its design can couple easily with the external design and planning systems, which then can utilize the capability matchmaking algorithm to produce matchmaking results remotely. This eliminates the need for client systems to take a major software development on their part and thus helps in quick adoption of the service. The capability matchmaking web service, the capability matchmaking algorithm and the formal information models are the three major components, which operationalizes the capability matchmaking process. They together constitute a system, which makes the process of matchmaking smooth to operate. In this paper, our focus is on describing web service component of this system.

We have used software process [13] to build the web service. Software process breaks down the task of software production into a set of related activities, which should be completed for development of the software. We have divided our development process into stages of requirements definition, analysis, architecture design, implementation and validation. In the sub-sections below, we will describe these stages.

3.1. Requirements

Requirements are specifications according to which a software (SW) system should function. The requirements we identified for the prototype are such that they require the SW system, to be designed in a manner that it is able to operate with existing production design and reconfiguration systems and help production system designer in the resource selection process. The software engineering literature classifies the requirements into User and System requirements based on the level of description [13]. The scope of our research project does not require elicitation of user requirements, so we will emphasize on system requirements. The system requirements enumerated here are not exhaustive, the ones presented here relate to the web component of the system, and to the production system design.

- R1. The capability matchmaking system should provide a formal mechanism for information exchange. The information representations should follow a standardized interface to be easily created and interpreted by various stakeholders. This requirement is applicable to both inputs received and outputs generated by the system.
- R2. The product requirement description, resource description and production system-layout description documents act as input to the system. They can be in the form of catalogues/database located either locally or remotely. The system should have the ability to obtain these electronically from the specified locations.
- R3. The system should be able to index the resource-pools and system-layouts provided by clients and return them a reference identifier (ID).
- R4. The system should be able to respond to requests irrespective of their origin, if the request and associated inputs conform to information exchange mechanism specified by the system.
- R5. The system should acknowledge the requests for matchmaking with a matchmaking request ID.
- R6. The system should respond with matchmaking result response when a matchmaking result request is sent with a matchmaking request ID.
- R7. The system should return error responses in case it is not able to accept requests due to a technical failure.
- R8. The system should cache matchmaking results for some definite period.

The system requirements are usually classified into functional and non-functional requirements [13]. The requirements we have enumerated above are functional in nature as they specify what functions system should perform. Non-functional requirements apply to the whole system and can be in the form of product, organizational or external requirements. They might also relate to emergent properties of the system like security and reliability [13]. We have deliberately avoided writing on non-functional requirements as this work is in nature of a research prototype and in our opinion, incorporation of the non-functional requirements can be considered during development of a more mature version of software.

3.2. Analysis

The requirements listed in the previous section identify the functions of the capability matchmaking system's web component. To analyze these requirements, we use the sequence diagrams to depict the use cases and demonstrate the requirements as part of an interaction sequence. This help us to understand the technical solution, which will fulfil the requirements.

Fig. 1 shows the interactions between an external system (client) with the capability matchmaking system to trigger a matchmaking request. The first two scenarios are for creation of the matchmaking search space by the external production design system by providing resource-pools and system-layouts. The matchmaking system stores and registers these inputs and returns the identifiers (respool_ID and syslayout_ID) to the client system (R2, R3, R4). In the matchmaking request scenario, the client system issues a request message for matchmaking by specifying the input identifiers (respool_IDs or syslayout_IDs). These identifiers help the system to identify and procure the inputs needed for executing the matchmaking process. The Matchmaking request is responded with a matchmaking request ID by the system (R5). In the matchmaking response scenario, the system returns the matchmaking result as response to the result request message. The request message contains the request ID obtained by client during the previous interaction with the system, and it is used

to pick the right matchmaking result. (R6, R8). If a matchmaking request is still under preparation, a result not ready response is sent back to client systems (R4, R7). Error message is sent to the client system in case of a technical error in the system (R7).

The interactions described above must be carried out with messages following a specified schema as they are created by external systems and capability matchmaking system requires parsing them accurately for carrying out matchmaking tasks (R1, R2). As part of this research, we have developed Extensible Markup Language (XML) messages, which are defined by XML schemas (XSD). These messages can also be seamlessly transmitted and intercepted as JavaScript Object Notation (JSON) formatted messages.

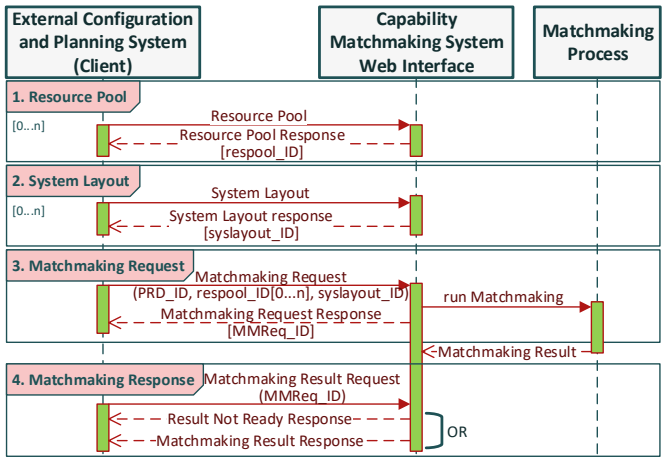

Fig. 1. Matchmaking interaction scenario.

Fig. 1 shows capability matchmaking system's web interface as the top layer, which accepts the requests and sends back the responses. From perspective of the capability matchmaking system, this layer should provide a robust framework for request and response mechanism without having any dependence on the technological capabilities of the external receiver systems (R4).

Web services as client and server applications that communicate over the web using Hypertext Transfer Protocol (HTTP) protocol, fulfill the requirements identified for communication interface of the system as they provide a standard means of interoperating between software applications running on a variety of platforms and frameworks. The service consumer and provider use messages to exchange invocation request and response information in the form of self-containing documents that make very few assumptions about the technological capabilities of the receiver fits also well to the requirements [14].

The two most common type of web services are Simple Object Access Protocol (SOAP) and RESTful. Both these approaches have their strengths and limitations for implementation [15]. We have developed the web interface of the capability matchmaking software as a RESTful web service [6]. Our decision to use Representational State Transfer (REST) was based on its offering of robust framework of request and response mechanism without imposing any significant technical limitations and contracts on client systems to produce and consume messages. Furthermore, RESTful Web services are easy to develop and maintain as REST leverages existing W3C/Internet Engineering Task Force (IETF) standards and the necessary infrastructure has become ubiquitous.

3.3. Architecture Design

The capability matchmaking system follows a layered architecture. Fig. 2 outlines the various layers and the interactions between them, which together demonstrate the Architecture constructing the SW system. The top most layer in Fig. 2 represents the external client systems, which interacts with the web service component of the system. They exchange information through XML/JSON messages either to trigger matchmaking or to obtain matchmaking results. The Web Service layer performs multiple tasks. It receives the various request messages (see Fig. 1) from the client systems and validates the inputs in the received messages. After performing validation, the web service layer gathers the input resources from different catalogue(s)/database(s) and invokes the matchmaking process in Business layer. The web service layer is also responsible for the response messages (see Fig. 1).

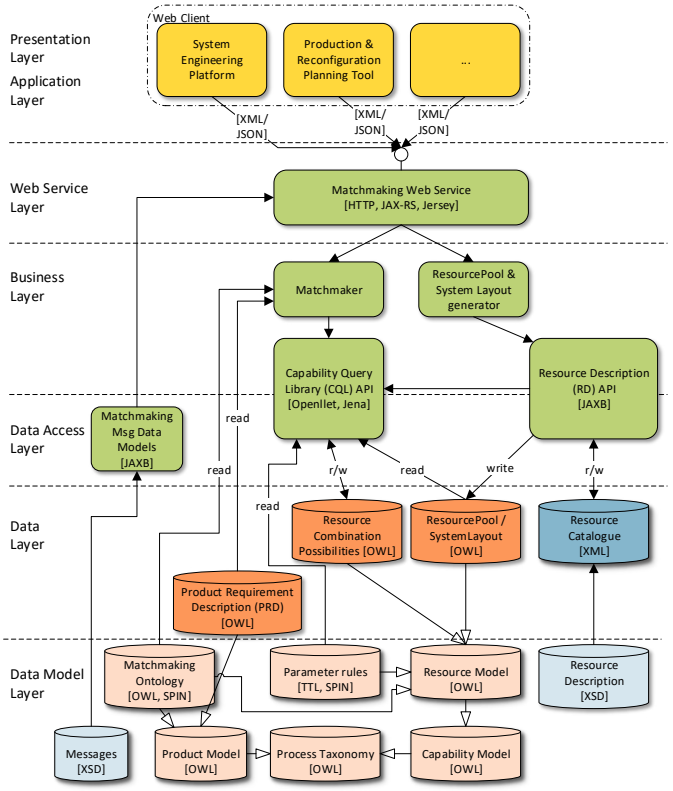

Fig. 2. System Architecture.

In the Business layer, the matchmaking procedure is carried out. The matchmaking process uses Capability Query Library (CQL) to interact with the ontology models and to run various rules. After matchmaking is completed, the results are passed on to Web Service Layer to be cached. Later they can be retrieved by client systems by invoking a matchmaking result request. The Data layer contains the manifestations of information specified by information models defined in Data

Model layer. The information models are briefly described in the section two and in our previous publications.

3.4. Implementation

The capability matchmaking software is built with Java Technology [16]. Our choice of Java as the primary programming language does not preclude any attempts to build the application with a different programming language. The development of RESTful Web services is carried out with Java API for RESTful Web Services (JAX-RS) API's [14] open source reference implementation Jersey [17], in order to simplify and standardize the RESTful implementation. We have used Apache Tomcat [18], which is a popular lightweight and open source web server, for hosting our capability matchmaking web service and associated software modules.

Matchmaking process is implemented as well in Java and it uses CQL to interact with different ontologies. CQL utilizes the Apache Jena [19] framework, which is used in building semantic web and linked data applications, and Openllet [20] as an OWL reasoner. A reasoner can find or infer knowledge that is not explicitly stated in the ontology. The CQL mainly gets information from the ontology with SPARQL. These queries are executed through Openllet to add semantic reasoning. The information models for messages used with the application are defined by XML schemas (XSD), and they are further converted into Java objects by using JAXB (Java Architecture for XML Binding) [21].

3.5. Validation

Validation phase of the software development process is concerned with checking that the developed product meets the specifications, which are identified during the requirements phase. This task is carried out by testing the system against a set of inputs and then comparing the output of system with the expected outputs. This task is continuous and concurrent with the development of the system. Usually the validation testing is an iterative process carried out by testing team after a major chunk of development is completed. In the present prototype development, we have carried out the validation testing within our research team and with the industrial partners associated with the project.

4. Case Example

A production system designer has a new switch valve, which needs to be manufactured. He/she needs to design a production system for the valve and decides to utilize help of automatic matchmaking system in search of feasible assembly solutions. Fig. 3 illustrates such a search of suitable production resource(s) as part of the design task for a production system. It follows the matchmaking scenario described in Fig. 1. First, the system designer needs to have (or make) a Product Requirement Description (PRD) representing the assembly process requirements for the switch valve. The PRD is graphically illustrated in the top left corner of Fig. 3. Second, the resource search space is to be defined. It could be an existing system layout, but in this case the designer is working with greenfield case, and such production system does not exist. Thus, he/she selects some production resource catalogue(s) and specific resources (if not all), and creates a resource pool out of them (Fig. 1 / Sequence 1).

After this, actual matchmaking is initiated (Fig. 1 / Sequence 3) by the system designer, in this case with help of web browser. The left side of Fig. 3 demonstrates the matchmaking request and right the matchmaking result. The first input of the request contains the reference to the PRD. For illustration, only one process step "screwing" is focused with the matchmaking request (green arrows). The second input is the reference to the resource pool created earlier (blue arrow).

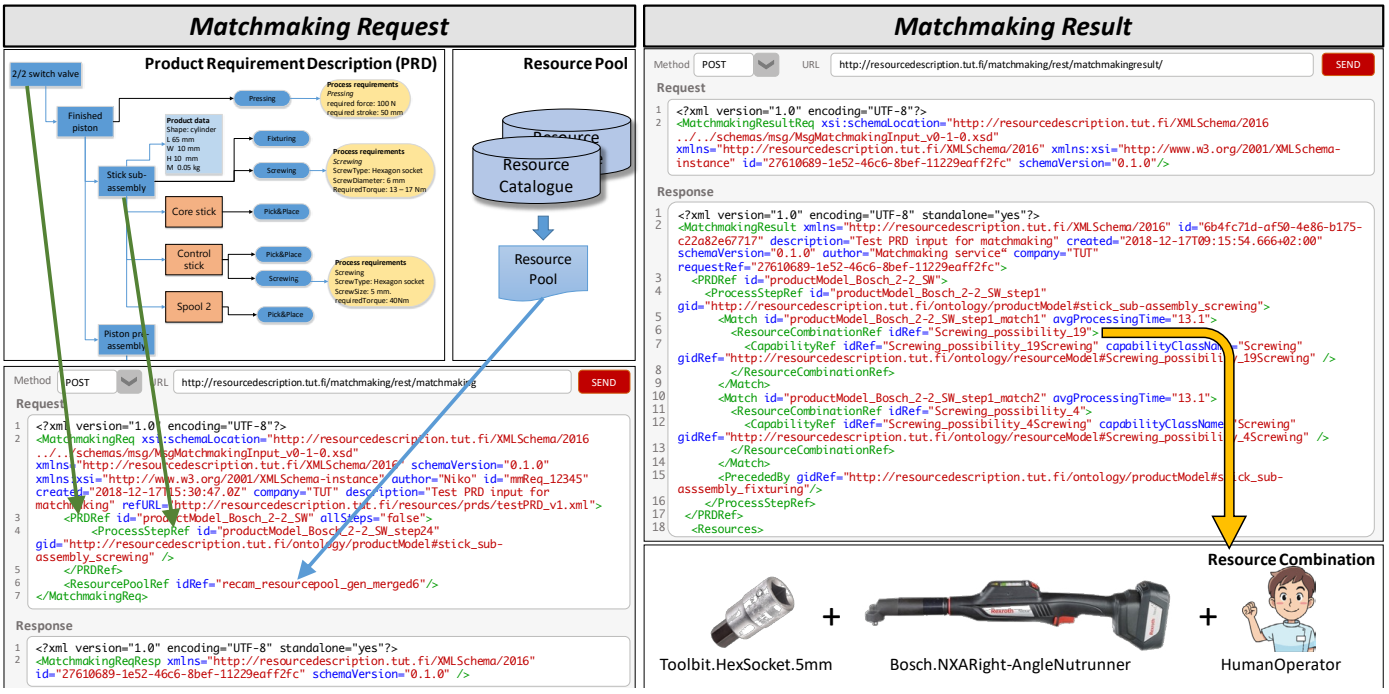

Fig. 3. Illustration of matchmaking scenario.

After sending this request, the matchmaking process is started, and generated id of this request is returned as a response (bottom left in Fig. 3).

The right side of Fig. 3 illustrates retrieval of matchmaking result (Fig. 1/Sequence 4). A request is sent with id provided in the previous step. Matchmaking result message is received as response. It contains resource combinations that match with the requirements of the process steps, and details of the included resources. The matchmaking result, visible in Fig. 3, contains two matching resource combinations. The system designer has selected one of the found matches for his/her production system. The selected resource combination is illustrated as physical resources in bottom right corner of Fig. 3.

5. Conclusions

We have developed a prototype to automate the matchmaking activity for combining production resources, which is traditionally done by a human designer, based on his/her experience. Thus, as potential impact of our tool, it is expected that the process will be much faster and more alternative system design solutions can be considered (e.g. wider search spaces, offerings from multiple vendors), which means that better, new, even unexpected alternatives may come into the picture. The speed increase will come due to computation power, and formalized capability and resource descriptions.

In this paper, we presented the conception and development process of capability matchmaking web service, which provides an interface to the capability matchmaking system that is intended to support designers in production system design and reconfiguration processes. The capability matchmaking system helps the designer by automatically providing information about resources and resource combinations matching for each required process step of the manufactured product. On the technical side, the service follows the RESTful architecture, which by design is such that it can be used and invoked easily, by any existing design and planning system, after providing necessary inputs. Therefore, it does not require any specific technical ability on the client side to consume the XML or JSON messages. Even a simple web browser can be used for message exchange.

We provided details about the development process and explained the rationale behind software design and architecture decisions. The development process gives us the opportunity to look back at the system and try to find components where improvements can be made. One major issue we have observed with the prototype implementation is performance efficiency in terms of both memory and time. In addition, security and authentication process can be improved in future implementations. However, one of the main outcome of the capability matchmaking prototype software implementation is that it demonstrates that the matchmaking process is working and that matchmaking can be invoked easily outside the application through its web service interface.

Acknowledgements

This research has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 680759 (www.recam-project.eu).

References

[1] Koren Y, Shpitalni M. Design of reconfigurable manufacturing systems. J Manuf Syst 2010;29(4):130-41. doi:10.1016/j.jmsy.2011.01.001.
[2] Pakkanen J, Juuti T, Lehtonen T. Brownfield Process: A method for modular product family development aiming for product configuration. Des Stud 2016;45:210–41. doi:10.1016/j.destud.2016.04.004.
[3] ReCaM consortium, ReCaM project web page, http://www.recam-project.eu. [Accessed 28.02.2019].
[4] Järvenpää E, Siltala N, Hylli O, Lanz M. The development of an ontology for describing the capabilities of manufacturing resources. J Intell Manuf 2018; p.1-20. doi:10.1007/s10845-018-1427-6
[5] Järvenpää E, Siltala N, Hylli O, Lanz, M. Product Model Ontology and Its Use in Capability-Based Matchmaking. Procedia CIRP 2018;72:1094-9. doi:10.1016/j.procir.2018.03.211
[6] Fielding R. Architectural Styles and The Design of Network-based Software Architectures. PhD thesis. Univ of California, Irvine; 2000.
[7] Järvenpää E, Lanz M, Siltala N. Formal Resource and Capability Models supporting Re-use of Manufacturing Resources. Procedia Manuf 2018;19:87-94. doi:10.1016/j.promfg.2018.01.013
[8] Järvenpää E, Siltala N, Hylli O, Lanz M. Capability matchmaking procedure to support rapid configuration and re-configuration of production systems. Procedia Manuf 2017;11:1053-60. doi:10.1016/j.promfg.2017.07.216.
[9] Järvenpää E, Hylli O, Siltala N, Lanz M. Utilizing SPIN Rules to Infer the Parameters for Combined Capabilities of Aggregated Manufacturing Resources. IFAC-PapersOnLine 2018;51:84-9. doi:10.1016/j.ifacol.2018.08.239
[10] Siltala N, Järvenpää E, Lanz M. Creating Resource Combinations Based on Formally Described Hardware Interfaces. In: Ratchev S, editor. Precis. Assem. Technol. Syst. - 8th IFIP WG 5.5 Int. Precis. Assem. Semin. IPAS 2018, Chamonix, France: Springer Nature Switzerland AG 2019; 2019, p. 29–39. doi:10.1007/978-3-030-05931-6_3.
[11] Järvenpää E, Siltala N, Lanz M. Formal resource and capability descriptions supporting rapid reconfiguration of assembly systems. 2016 IEEE Int. Symp. Assem. Manuf., IEEE; 2016, p. 120-5. doi:10.1109/ISAM.2016.7750724
[12] Siltala N, Järvenpää E, Lanz M. Formal Information Model for Representing Production Resources. Adv Prod Manag Syst Initiat Sustain World APMS 2016 IFIP Adv Inf Commun Technol 2016;488:53–60. doi:10.1007/978-3-319-51133-7_7
[13] Sommerville I. Software Engineering. 9th ed. ISBN:0137035152 9780137035151. Addison-Wesley Publishing Company, USA; 2010
[14] Java EE 6 Tutorial. https://docs.oracle.com/javaee/6/tutorial/doc/; 2013. [Accessed 28.02.2019]
[15] Pautasso C, Zimmermann O, Leymann F. RESTful Web Services vs. "Big" Web Services:Making the Right Architectural Decision. Proceeding 17th Int. Conf. World Wide Web - WWW '08, New York, New York, USA: ACM Press; 2008, p. 805–14. doi:10.1145/1367497.1367606
[16] Java Technology. https://www.oracle.com/java/. [Accessed 28.02.2019].
[17] Jersey – reference implementation of JAX-RS. https://jersey.github.io/. [Accessed 28.02.2019].
[18] Apache Tomcat. http://tomcat.apache.org/. [Accessed 28.02.2019]
[19] Apache Jena. https://jena.apache.org/. [Accessed 28.02.2019]
[20] Openllet API. https://github.com/Galigator/openllet. [Accessed 28.02.2019]
[21] Ort E, Mehta B. Java Architecture for XML Binding (JAXB). https://www.oracle.com/technetwork/articles/javase/index-140168.html; 2003. [Accessed 28.02.2019]