

Ontology matchmaking of product ramp-up knowledge in manufacturing industries

How to transfer a cake-baking recipe between bakeries

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of
Doktor der technischen Wissenschaften

by

Dipl.-Ing. Roland Willmann

Registration Number
8625991

to the Faculty of Informatics
at the Vienna University of Technology

Advisor:

Ao. Univ. Prof. Dr. Wolfgang Kastner

Ao. Univ. Prof. Dipl.-Ing. Dr. techn. Mag.rer.soc.oec Stefan Biffl

The dissertation has been reviewed by:

Prof. Dr. Wolfgang Kastner

Prof. Dr. Stefan Biffl

Prof. Dr. Arndt Lüder

Treffen, 3.10.2016

Dipl.-Ing. Roland Willmann

Erklärung zur Verfassung der Arbeit

Dipl.-Ing. Roland Willmann
Hügelweg 5, 9521 Treffen

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Treffen, 3. 10. 2016

Dipl.-Ing. Roland Willmann

Acknowledgements

This work was written with support of the “Institute of Computer Aided Automation” and the “Christian Doppler Laboratory” for “Software Engineering Integration for Flexible Automation” in addition to my professional activities.

At this point, I would like to thank Prof. Dr. Wolfgang Kastner and Prof. Dr. Stefan Biffl. I am sincerely grateful about the reviews and advise of over all the years. Through our common process of direct critique, brain storming and active paying attention to my needs I could gain a lot of experience with respect to the writing of this work.

To my assisting advisors, Dr. Estefania Serral Asensio and Dr. Marta Sabou, I am particularly grateful for their professional support over all the years. Thank you for the fruitful discussions, suggestions and co-authorship in our publications.

I am grateful to my daughters Susy and Chrissy that they accepted my low presence for many years and showed patience and understanding. So great how you did develop personally in this time, it was easier for me to stay focused on this work.

Sincere thanks and appreciation appertains to my wife Claudia for years of renounced common leisure time and still dynamic motivational support. Without her contribution, the work would not have been possible. This work is dedicated to her.

Abstract

The manufacturing industry has to face shortening of product life cycles and decreasing lot sizes (lot size 1) which is particularly challenging with respect to the ability to perform an efficient and predictable product ramp-up phase. This development is underlined through the increasing individualization of products based on the principles of mass production support as it is envisioned by *Industrie 4.0* or the *Industrial Internet of Things (IIoT)*. For many manufacturing industries, the deterministic ramp-up of new products in existing production systems is therefore mission-critical.

The ramp-up of a new product starts with the transition of knowledge about the new product from the development production system to the mass-production system, and it ends when instances of the new product are produced and shipped at the latter with planned volume and with repeatable low number of defects. A similar situation has to be faced in case of capacity extension when knowledge of a pilot production system has to be transferred to further mass-production systems. However, manufacturing companies still struggle with the predictability of costs and duration as well as the ultimately recoverable yield for the ramp-up of new products. Deviations between plan and reality of ramp-up projects affect a company's profit negatively in two ways – unplanned loss of revenue due to delayed product roll-out and unplanned additional costs due to higher resource consumption.

Commonly heterogeneous equipment across the production systems which are involved in such scenarios of knowledge transfer prohibits exact copying of the whole knowledge about a product and its production process. Therefore, a multi-disciplinary ramp-up team from the fields of product engineering, process engineering and equipment engineering must specify all handling instructions for operating personnel, equipment recipes, data collection models and process control models at each individual production system. It is always the target of this ramp-up process to produce the new product within the context of a capable production process with the required quality and volume.

Unfortunately, today's ramp-up teams' production systems lack a well-structured approach and a common multidisciplinary knowledge base in order to take systematically advantage of knowledge about the already performed processes, their capabilities and knowledge about already produced forerunner products. This knowledge can be used in order to specify the production processes of the new product at the respective production system.

Enterprises in manufacturing industries partially address this challenge by introducing a standardized design of subparts, by the design of product platforms or by closer integration of product design and the capabilities of their production systems (a.k.a. design for manufacturing).

Such organizational measures result in improvements within the sphere of influence of a company. However, it is still difficult to take advantage of such measures along the supply chain. And also within companies the underlying management of information is not necessarily designed to be processed automatically in the context of a product ramp-up project.

In this thesis, a knowledge-based product ramp-up process (K-RAMP) and the underlying multidisciplinary information model are proposed in order to interconnect information about a new products of an original production system (e.g., low volume pilot production or development line) with information about the production of forerunner products of a target production system (e.g., high volume production). Based on the existing knowledge assets of forerunners', subcomponents or already qualified process segments of the target production system, the introduced concept systematically helps to determine and to recommend opportunities for reuse in order to produce the new product. As novelty, the developed derivation logic does not only consider identities between product specifications or between specifications of capabilities of the production system but it also similarities due to generalization of information fragments.

Axiomatic design is applied to design K-RAMP. The independence axiom of *axiomatic design* in conjunction with decomposition and zig-zagging is used across functional requirements, the information model (a.k.a. design parameters) and procedures (a.k.a. process variables) in order to ensure completeness but also to avoid over-engineering of the concept. In addition, *axiomatic design* does not predetermine the final technical implementation, which was an important asset during the selection process of the research method with respect to the design phase of K-RAMP.

An implementation of the design is performed which is based on the *Semantic Web*. The *Semantic Web* was chosen because it comprises emerging technologies for a variety of fields in the *Internet* including the *IIoT* for purpose of the digitalization of manufacturing industries. Recalling the vision of *Industrie 4.0* with respect to the digitalization of the whole supply chain, these technologies are assumed as most promising for an implementation of K-RAMP. In the context of the thesis, it is also researched for what purposes features of the *Semantic Web* are suitable for implementation of the knowledge base. Domains, where alternative solutions need to be applied are also addressed. Finally, this leads to a hybrid concept of K-RAMP which comprises predominantly *Semantic Web* technologies but also some imperatively programmed components.

The hybrid implementation of the design is evaluated by a systematic sequence of the verification of partial design elements. For this purpose, an easy to follow case study is implemented as proof of concept – the transfer of a cake-baking process from one bakery to another. This cake-baking case study is implemented by the use of all means which are commonly applied in manufacturing industries for representation of product designs or the specification of production processes. This is ensured by consideration of industry standards and guidelines (ISA95 [1], SEMATECH CIM Framework [2], VDI/VDE 3682 [3]) with a broad base of application as well as personal experience as leading process analyst and architect in software projects for the management and analysis of production data and process data, process control and equipment integration in semiconductor production, the manufacturing of photovoltaic panels, cells or modules, special steel milling or automotive assembly within a period of twenty-five years. The case study and the generalization of the results demonstrate the applicability of K-RAMP for various domains of industrial manufacturing.

The product design of a Viennese chocolate cake is specified by means of the developed ontology models for one (original) bakery. The same way, also two other cakes, including their specific handling instructions, equipment recipes and specifications of the production system at a target bakery, are described. By intention, the product specification of one of these cakes has some similarities with the Viennese chocolate cake while the other cake has almost none. During the

evaluation, a step-by-step verification of the matchmaking process is performed. Intended similarities of both cakes and their process-related information are considered in order to generate recommendations for knowledge reuse. This verification of the matchmaking process comprises process segments and their setups needed, in order to bake the Viennese chocolate cake at the target bakery by reusing the existing process information. Finally, a reflection of the evaluation results underlines premises which need to be satisfied for applying K-RAMP with maximal success. The concrete application of K-RAMP in a real production system is not part of this thesis but subject of future research. With respect to the premises of an effective application of K-RAMP, useful standardization initiatives are recommended as part of the summary statement.

It is not the intention of K-RAMP to make production knowledge reusable. This challenge is subject to the respective enterprises within the scope of their management strategies concerning the modularity, scalability and compatibility of products and process segments. But facing the digitalization of the supply chain, such challenges are already mastered by some enterprises and have to be mastered by the remaining majority of enterprises in any case in the upcoming decade. K-RAMP contributes to this development with an adequate information model which is derived from existing industry standards and guidelines as well as with a structured, automatable process. Assuming the previously outlined premises as given, K-RAMP can automatically derive recommendations about production knowledge reuse in the scope of a product ramp-up scenario. Such recommendations may help to perform product ramp-up projects faster and with more deterministic results.

Therefore, this work is of particular interest for ICT-experts in the manufacturing industry who are facing challenges from the perspective of information science in order to improve the situation of product ramp-up projects. For the same target group, the work is also of interest with respect to the application of *Semantic Web* technologies in production environments in general. Due to the application of these technologies, the results of this work are also applicable as starting point for new research which is fostered by *Industrie 4.0*. This is specifically true for the unification and standardization of public product specifications or production service specifications along the supply chain and research on search engines for the same purpose. This work is therefore of particular interest for research on *the application of the Semantic Web for mastering of digitalized supply chains in manufacturing industries* as well as on *knowledge representation in manufacturing industries for mastering decreasing product lifecycles*.

Kurzfassung

Die produzierende Industrie muss sich verkürzten Produktlebenszyklen und abnehmenden Losgrößen (Losgröße 1) stellen, was wiederum herausfordernd bezüglich der Fähigkeit ist, den Anlauf neuer Produkte effizient und mit vorhersagbarem Ausgang durchzuführen. Aufgrund der, durch *Industrie 4.0* oder dem *Industrial Internet of Things (IIoT)*, vergegenwärtigten Entwicklung zunehmend individualisierter Produkte, auf der Grundlage von Prinzipien der Massenproduktion, wird dies unterstrichen. Für die produzierende Industrie ist der deterministische Anlauf neuer Produkte (product ramp-up), in bestehenden Fertigungslinien, daher zunehmend erfolgskritisch.

Der Produktanlauf beginnt mit der Übertragung von Wissen über das neue Produkt, aus der Produktentwicklunglinie in das Produktionssystem für die Massenfertigung. Diese Phase endet, nachdem Artikel des neuen Produktes, in der geplanten Menge sowie mit wiederholbar wenigen fehlerhaften Teilen, hergestellt und ausgeliefert werden können. Eine ähnliche Situation ist bei Kapazitätserweiterungen der Produktion gegeben, wenn Wissen aus einer Pilotfertigung an weitere Produktionssysteme zur Massenfertigung transferiert werden muss. Jedoch selbst heute noch quälen sich produzierende Unternehmen mit der Vorhersagbarkeit von Kosten und Dauer sowie der letztendlich erzielbaren Ausbeute, beim Anlauf neuer Produkte. Abweichungen zwischen Plan und Wirklichkeit des Produktanlaufes beeinflussen den Gewinn eines Unternehmens in zweifacher Hinsicht negativ: einerseits durch ungeplante Umsatzverluste aufgrund eines verspäteten Markteintrittes und andererseits durch ungeplante Zusatzkosten aufgrund zu hoher Ressourcennutzung.

Eine heterogene Maschinen- und Anlagenausstattung, über die involvierten Produktionssysteme hinweg, verhindert in solchen Szenarien des Wissenstransfers das exakte Kopieren des gesamten erforderlichen Wissens über das Produkt und dessen Herstellungsprozess. Ein multidisziplinäres Ramp-up-Team, aus den Fachgebieten der Produkt-, Prozess- und Anlagentechnik, muss für das Produktionssystem alle Handlungsanweisungen für das Fertigungspersonal sowie Anlagenrezepte und Strategien zur Datenerfassung und Prozessregelung beschreiben, um das neue Produkt, im Rahmen eines fähigen Produktionsprozesses, mit geforderter Qualität und Menge effizient herzustellen.

Jedoch fehlen Ramp-up-Teams heute ein wohlstrukturiertes Vorgehen und eine gemeinsame multidisziplinäre Wissensbasis, um systematisch Vorteile aus bereits bestehendem Wissen über bereits laufende Produktionsprozesse, deren Fähigkeiten und Wissen über bereits hergestellte

Vorläuferprodukte zu nutzen. Dieses Wissen kann benutzt werden, um den Herstellungsprozess neuer Produkte im jeweiligen Produktionssystem zu beschreiben.

Die Unternehmen der produzierenden Industrie stellen sich dieser Herausforderung teilweise durch die Einführung eines standardisierten Bauteildesigns, durch das Design von Produktplattformen oder durch eine engere Integration von Produktdesign und den Fähigkeiten ihrer Produktionssysteme (design for manufacturing). Solche organisatorische Maßnahmen führen zu Verbesserungen im Einflussbereich eines Unternehmens. Allerdings ist es immer noch schwierig, die Vorteile solcher Maßnahmen entlang der gesamten Wertschöpfungskette zu adaptieren. Und auch innerhalb der Unternehmen ist die zugrundeliegende Verwaltung von Informationen nicht notwendigerweise auf eine automatische Verarbeitung, im Zuge des Produktanlaufes, ausgelegt.

In dieser Arbeit werden ein wissensbasierter Prozess und das zugrundeliegende Informationsmodell zum Anlauf neuer Produkte (“knowledge-based production ramp-up process”, kurz: K-RAMP) aus einer ursprünglichen Fertigungslinie (z. B. eine Pilotproduktion mit geringem Ausstoß) in eine Zielfertigungslinie (z.B. Massenfertigung) vorgestellt. Auf der Grundlage der vorhandenen Wissensressourcen, abgeleitet von bereits hergestelltem Halbzeug oder bereits qualifizierten Abschnitten von Produktionsprozessen des Zielproduktionssystems, findet das vorgestellte Konzept Möglichkeiten zu deren Nachnutzung, für die Herstellung des neuen Produktes. Als Neuigkeit, berücksichtigt die entwickelte Ableitungslogik nicht nur die Identität von Produktbeschreibungen oder von Fähigkeiten von Produktionssystemen, sondern auch pure Ähnlichkeiten zwischen diesen Entitäten, aufgrund der Verallgemeinerung von Informationselementen.

Axiomatic Design wird für den Entwurf von K-RAMP genutzt. Das Independence Axiom von *Axiomatic Design* wird in Verbindung mit Dekomposition und zig-zagging über die funktionalen Anforderungen (functional requirements), das Informationsmodell (design parameter) und Prozeduren (process variables) hinweg eingesetzt, um die Vollständigkeit des Konzeptes, bei gleichzeitiger Vermeidung einer Überdimensionierung, sicherzustellen. Zusätzlich präjudiziert *Axiomatic Design* nicht die finale technologische Umsetzung, was ein wesentlicher Vorteil im Zuge des Auswahlprozesses der Forschungsmethode für die Entwurfsphase von K-RAMP war.

Die Implementierung des Entwurfes erfolgt auf der Grundlage des *Semantic Web*. Das *Semantic Web* wurde gewählt, da es neu aufkommende Technologien für viele Bereiche des *Internet* umfasst, einschließlich des *IIoT* zum Zweck der Digitalisierung der Fertigungsindustrien. Ruft man sich die Vision von *Industrie 4.0* bezüglich der Digitalisierung der gesamten Wertschöpfungskette in Erinnerung, so werden diese Technologien als am meisten versprechend für die Implementierung von K-RAMP angesehen. Im Rahmen dieser Arbeit wird auch erforscht, für welche Zwecke das *Semantic Web* bei der Implementierung der Wissensbasis geeignet ist. Bereiche, wo alternative Lösungen angewandt werden müssen, werden ebenso adressiert. Schließlich führt dies zu einem hybriden Konzept für K-RAMP, welches überwiegend Technologien des *Semantic Web* umfasst, jedoch auch einige imperativ programmierte Komponenten benötigt.

Die hybride Implementierung des Entwurfs wird durch eine systematische Abfolge der Verifikation von Teilkomponenten bewertet. Zu diesem Zweck wird eine einfach nachvollziehbare Fallstudie implementiert – die Übertragung eines Backprozesses von einer Bäckerei zu einer anderen. Dieser Anwendungsfall wird so implementiert, dass er alle üblicherweise verwendeten Merkmale des Produktdesigns sowie der Spezifikation eines Produktionsprozesses aufweist. Dazu wird auf Industriestandards und Richtlinien (ISA95 [1], SEMATECH CIM Framework [2], VDI/VDE 3682 [3]) mit breiter Anwendungsbasis und auf persönlichem Hintergrundwissen, als leitender Prozessanalyst und Architekt in Software-Projekten, zur Verwaltung und Analyse von

Produktions- und Prozessdaten, der Prozesssteuerung und Maschinenintegration, bei der Herstellung von Halbleitern, Photovoltaikpanelen, -zellen oder -modulen, dem Walzen von Spezialstahl oder in der Automobilmontage, in einem Zeitraum von fünfundzwanzig Jahren, aufgebaut. Durch diese Fallstudie und die Verallgemeinerung der Ergebnisse wird die breite Anwendbarkeit von K-RAMP für verschiedene Bereiche der industriellen Fertigung demonstriert.

Das Produktdesign einer bekannten Wiener Schokoladentorte wird mittels der entwickelten Ontologien für eine (ursprüngliche) Bäckerei beschrieben. In derselben Weise werden auch zwei weitere Mehlspeisen, einschließlich ihrer detaillierten Handlungsanweisungen, Maschinenrezepte und Spezifikationen des Produktionssystems einer Zielbäckerei beschrieben. Es ist dabei beabsichtigt, dass eine der beiden Mehlspeisen Ähnlichkeiten mit der Schokoladentorte aufweist, während dies für die zweite Mehlspeise fast gar nicht zutrifft. Im Zuge der Evaluierung wird Schritt für Schritt verifiziert, ob beabsichtigte Ähnlichkeiten beider Mehlspeisen bei der Vernetzung der Informationsmodelle berücksichtigt werden, um entsprechende Empfehlungen abzuleiten. Diese Verifikation des Vernetzungsprozesses umfasst die Prozesssegmente und deren spezifische Einstellungen, welche zum Herstellen der Schokoladentorte in der Zielbäckerei benötigt werden und dabei vorhandene Prozessinformationen wiederverwendet. Die Reflexion der Evaluierungsergebnisse streicht schließlich die Prämissen hervor, welche für einen maximalen Erfolg der Nutzung von K-RAMP berücksichtigt werden müssen. Die konkrete Anwendung von K-RAMP in einem realen Produktionssystem ist nicht Teil der Arbeit, sondern das Ziel nachfolgender Forschung. Bezüglich der Voraussetzungen für eine effektive Anwendung von K-RAMP werden nützliche Initiativen zur Standardisierung im Rahmen der Zusammenfassung empfohlen.

Es ist nicht die Intention von K-RAMP Produktionswissen wiederverwendbar zu machen. Dieser Herausforderung müssen sich die entsprechenden Unternehmen, im Rahmen ihrer Managementstrategien, bezüglich der Modularisierung, Skalierbarkeit und Kompatibilität von Produkten und Prozesssegmenten, stellen. Die Digitalisierung der Wertschöpfungskette vor Augen, wurden diese Herausforderungen von manchen Unternehmen bereits gemeistert. Von der Mehrheit der verbleibenden Unternehmen müssen diese Herausforderungen jedenfalls innerhalb des nächsten Jahrzehntes gemeistert werden. K-RAMP trägt zu dieser Entwicklung mit einem geeigneten Informationsmodell bei, welches von existierenden Industriestandards und Richtlinien abgeleitet ist sowie mit einem strukturierten, automatisierbaren Prozess. Nimmt man die zuvor genannten Prämissen als gegeben an, so kann K-RAMP zur automatischen Herleitung von Empfehlungen, zur Nachnutzung von Produktionswissen, im Rahmen eines Produkthanlaufes, genutzt werden. Solche Empfehlungen könnten dazu beitragen, Projekte zum Produkthanlauf rascher und mit vorhersagbareren Ergebnissen umzusetzen.

Aus diesem Grund ist diese Arbeit für IT-Experten, angesichts von Herausforderungen aus der Sicht der Informatik, zur Verbesserung der Situation von Projekten zum Produkthanlauf, von besonderem Interesse. Für dieselbe Zielgruppe ist diese Arbeit bezüglich der Anwendung des *Semantic Web* in Produktionsumgebungen interessant. Aufgrund der Nutzung von *Semantic Web* Technologien ist das Ergebnis dieser Arbeit auch als Startpunkt für neue Forschungsarbeiten anwendbar, die durch *Industrie 4.0* getrieben werden. Dies trifft insbesondere auf Initiativen zur Vereinheitlichung und Standardisierung von veröffentlichten Produktbeschreibungen oder Beschreibungen von Produktionsdienstleistungen, entlang der Wertschöpfungskette, zu. Dies gilt auch für die Erforschung von Suchmaschinen für denselben Zweck. Diese Arbeit ist daher von besonderem Interesse bei der Erforschung *der Anwendung des Semantic Web zur Beherrschung digitalisierter Wertschöpfungsketten in der produzierenden Industrie sowie der Wissensrepräsentation in der produzierenden Industrie, zur Beherrschung von sich verkürzenden Produktlebenszyklen.*

Content

1	INTRODUCTION	1
1.1	An everyday learning problem	1
1.2	The relevance for the manufacturing industry	2
1.3	Problem statement	5
1.4	Research questions and objectives	9
1.5	Methodological approach	11
1.5.1	Research tasks	11
1.5.2	Application of axiomatic design	11
1.5.3	Evaluation	14
1.6	Structure of work	14
2	STATE OF THE ART	17
2.1	Considered existing knowledge domains	17
2.2	Industry standards and industry terminology	18
2.3	How factories master the ramp-up complexity	19
2.4	Quality assurance in a production process	20
2.5	Production control and process control	23
2.6	Semantic Web and Ontologies in the industry	26
2.7	Summary	29
3	THE K-RAMP DESIGN	31
3.1	Applied methods	31
3.1.1	Overview	31
3.1.2	Usage of Unified Modeling Language	31
3.1.3	Consideration of design patterns	32
3.2	General design	32
3.2.1	Initial mapping of design domains	32
3.2.2	First decomposition step	36

3.2.3	Second decomposition step	45
3.2.4	Product design	77
3.2.5	Process design	77
3.2.6	Verification of Question 1	80
3.3	Ontology models by using the Semantic Web	81
3.4	Design of hybrid components	86
3.5	Architecture overview	96
3.6	Summary	97
4	TRANSFER OF A CAKE RECIPE	99
4.1	Overview	99
4.2	Evaluation plan	99
4.3	The evaluation environment	102
4.4	Creation of ontology models	104
4.4.1	Common taxonomy assumptions	104
4.4.2	Ontology of original bakery	119
4.4.3	Ontology of target bakery	131
4.5	Evaluation of matchmaking and recommendations	137
4.5.1	Combination of the original bakery and the target bakery	137
4.5.2	Relations between Product Category Sets	138
4.5.3	Assertion of Recommendations	143
4.6	Summary	144
5	REFLECTION	147
5.1	Overview	147
5.2	Evaluation versus objectives	147
5.3	Applicability of K-RAMP in real production	148
5.3.1	Generalization of the evaluation of K-RAMP and limitations	148
5.3.2	Presupposed architecture of production ICT	155
5.3.3	Quality of provided product and process basic data	157
5.3.4	Organizational premises	157

5.4	Summary	157
6	SUMMARY AND FUTURE WORK	159
6.1	General conclusions	159
6.2	Need for unified taxonomy and ontology models	160
6.3	Integration with the equipment engineering domain	161
7	APPENDIX	162
7.1	Cake recipes	162
7.1.1	Viennese Sachertorte	162
7.1.2	Meraner Torte	163
7.1.3	Eclairs	163
7.2	Dictionary	164
7.3	Bibliography	166

1 Introduction

1.1 An everyday learning problem

Jeff is fascinated about the chocolate cake of his mother. She makes it according to an old Viennese recipe. In particular, he fell in love with the chocolate gloss of this cake. Jeff likes baking as well, so he asks his mother how she makes this wonderful chocolate gloss because he wants to surprise his friends while giving a party in one week from now. However, he has to bake five cakes because he has invited many friends. So he has also some time pressure learning to bake these cakes properly.

“Well”, Jeff’s mother says, “I am switching the hotplate to level 5, placing a pot with water on the hotplate, wait for 7 minutes, taking 150g of brand-X chocolate, put it in a second pot, place this second pot in the first pot, ...”. Jeff is confused. He does not know his mother’s oven, because he has his own one. What does it mean to switch the plate to level 5? He has a wood-fired oven which does not provide any levels to switch the hotplate. Jeff likes baking with it because he has inherited this oven from his grandma. And what kind of chocolate is brand-X chocolate?

“The brand-X chocolate is a dark chocolate or bitter chocolate with 60% cacao at least”, explains his mother, “and you have to liquefy this chocolate in a hot water bath”. Now everything is getting clearer to Jeff. He knows that there are several brands of dark chocolate and for sure he will find some with 60% cacao in his supermarket of choice. He also knows how to make a hot water bath using his wood-fired oven. Therefore, all the previous very detailed information of his mother about arrangements of pots and switching of hot plates becomes obsolete.

What happened during the conversation between Jeff and his mother? In the second trial, Jeff’s mother generalized her specific knowledge to common concepts which are more useful for Jeff. She left out specific details which are of no value for Jeff, because he does not use the same cooking equipment, and she used generic concepts like “hot water bath” or “dark chocolate with 60% cacao at least” instead.

On this general conceptual level, Jeff was able to combine the transferred knowledge of his mother with personal experiences about his cooking equipment. In his mind he is therefore able to compile a sequence of actions which fits to his own cooking equipment but results in a chocolate cake with a quality close to the one of his mother. Moreover, by reusing his baking experience he is able to impress his friends at the party. He will be able to master the challenge within one week of time only.

Factories have to deal with similar situations in case of ramping up the production process of a new product. The production process was developed and tested in a pilot line (original production system) and has to be transferred to volume production (target production system). Alternatively, an existing volume production process for a dedicated product has to be dislocated from one production facility (original production system) to another (target production system), for instance, in case of capacity ramp-up. The question to be solved is about how to perform the ramp-up phase for production of a new product more deterministic with respect to achievable quality, ramp-up costs and ramp-up time.

However, today's ramp-up teams lack a well-structured approach and a common multidisciplinary knowledge base in order to systematically take advantage of existing knowledge from existing manufacturing processes, in order to specify new production processes [4]. The proposed "knowledge-based production ramp-up process" (K-RAMP) determines and recommends opportunities for the reuse of such existing production knowledge. It may thus contribute to a more deterministic ramp-up of a new product in a target production system.

Based on the existing production knowledge assets of currently produced subproducts or already mastered subsequences of the production process of the target production system and the design knowledge of the new product, the introduced ontology matchmaking process systematically determines opportunities for production knowledge reuse and derives recommendations in order to produce the new product based on existing production knowledge.

Axiomatic design is used as methodological approach in order to design the knowledge base and procedures. It is argued, for what purposes Semantic Web is suitable for implementing the knowledge base and for what reasons alternative solutions have to be applied. In terms of evaluation of the matchmaking process, an easy to follow case study is applied – the transfer of a cake-baking recipe from a bakery to another.

Therefore, this work is of particular interest for ICT-experts in the manufacturing industry who are facing challenges from the perspective of information science in order to improve the situation of product ramp-up projects. For the same target group, the work is also of interest with respect to the application of *Semantic Web* technologies in production environments in general. Due to the application of these technologies, the results of this work are also applicable as starting point for new research which is fostered by *Industrie 4.0*. This is specifically true for the unification and standardization of public product specifications or production service specifications along the supply chain and research on search engines for the same purpose. This work is therefore of particular interest for research on *the application of the Semantic Web for mastering of digitalized supply chains in manufacturing industries* as well as on *knowledge representation in manufacturing industries for mastering decreasing product lifecycles*.

1.2 The relevance for the manufacturing industry

Delayed completion of product ramp-up may have significant impact to the success of the product on the market. Customers may migrate to competitive products, thus reducing the planned sales quantity. As a consequence, there is not only a reduction of the planned turnover and profit. The planned production capacity is too large for the now lower sales. Aside of these major reasons for costs and losses due to non-deterministic product ramp-up, the costs for ramping up new products are typically operational expenses of the manufacturing companies and have therefore an immediate impact on their profitability and liquidity [5, pp. 96-98]. Non-deterministic product ramp-up also causes non-deterministic need for additional budget which is difficult to gain.

Abele et al. [5] are discussing product ramp-up in conjunction with the construction of a new facility. However, the principle statements are also correct for the concrete situation of this work, where the production of a new product shall be transferred from one existing production system to another one. Utilization of production resources (staff and equipment) without adding value, additional setup times of actually productive equipment (increased idle times and thus reduced added value), consumption of energy or scraped material (reduced added value, useless emissions to the environment), enhanced risk of negative impact on the quality of simultaneously ongoing volume production of forerunner products (customer satisfaction) are additional cost drivers. Personnel costs of process experts for know how transfer at far distant locations are another one.

Ramping up of a new product in a production system is an interdisciplinary task which involves experts from several domains (ramp-up team). Knowledge about currently produced products and the capabilities of the underlying production processes of a production line have to be combined with the specific requirements for the new product. During the ramp-up phase, a collection of product-specific instructions has to be created which makes up the overall production flow (process plan), the details of each single step of the production flow (process operation), as well as of control loops which need to be considered amongst the process operations. The completeness of each instruction must be ensured in order to meet the required characteristics of the product design and therefore the final product's quality.

Depending on the complexity of the product and the production process, the ramp-up team has to master a challenging but also error prone task, which is sometimes also based on trial and error. The complexity of the product is driven by the number of components (subproducts) which are used to compose the new product or by high quality demands, while the complexity of the production process depends on the number of single process steps which have to be performed with continuously high precision.

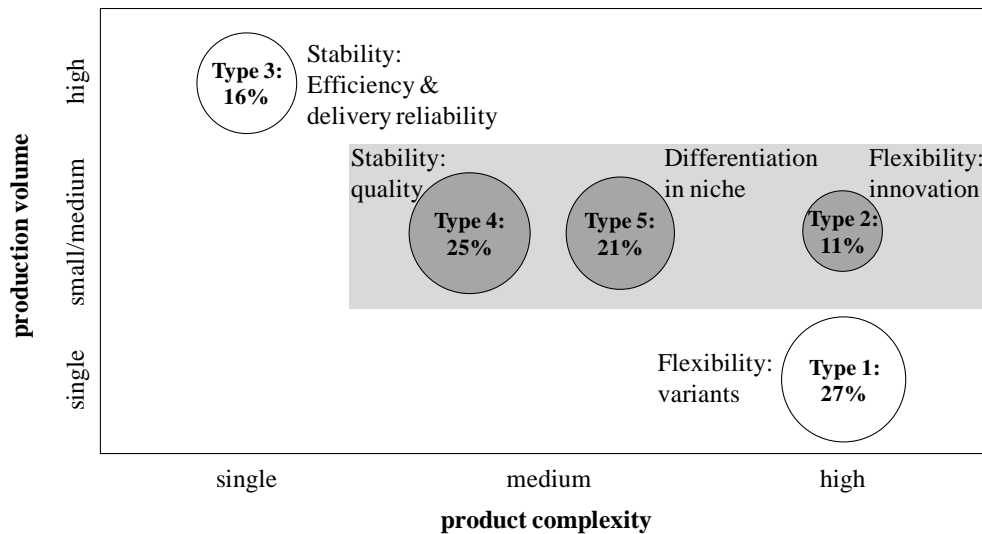


Figure 1.1: Types of German enterprises according to production volume and product complexity – source [6, p. 3].

Kinkel and Maloca published the results of a survey [6] amongst German enterprises in the manufacturing industry. In Figure 1.1 one result of this report is visualized and the enterprises with the highest relevance for K-RAMP are grayed. According to this assumption, almost 60% of the manufacturing enterprises in Germany are challenged by small to medium production volume with respect to each produced product (10,000 workpieces per product or product variant per year [7, p.

95]) and by medium to high product complexity. K-RAMP is of particular relevance for these enterprises as there is a significant overhead

- due to a large number of new product ramp-ups (as a consequence of the low production volume per product and the product complexity on a certain level) and
- because of the given potential to categorize production knowledge for reuse.

Typical examples of such enterprises are in the domain of car manufacturing or semiconductor manufacturing (Type 2 or Type 4 depending on strategic priority), or confectioneries (Type 5).

K-RAMP may be also of some relevance for enterprises of Type 1 (e.g., plant engineering) or Type 3 (e.g., food and beverage). However, in case of Type 1 it depends strongly on the specific nature of produced products, with respect to the potential for categorization and reuse of production knowledge. For Type 3, the number of ramp-ups of new products may be low and the resulting benefits may have minor economical relevance.

There are a clearly planned budget and a predefined duration which must not be exceeded by the ramp-up team while executing a ramp-up project. After completion of the ramp-up phase, the resulting instructions must enable the available production resources as well as suppliers and the control software to produce instances of the new product (workpieces) with repeatable quality on a certain level. This quality level to be achieved (initial yield) is predefined as well and limits the count of workpieces which are allowed to be scrapped or reworked because of missed quality criteria after the ramp-up project is finished.

Slamanig and Winkler [8, p. 488] report that the majority of companies still struggle to perform the ramp-up of new products within the planned costs or budget or to achieve the planned yield after ramp-up. “In the past, almost two-thirds of the companies were unable to meet their time-related targets, nearly 60% of the companies failed to achieve their cost-related goals, and 47% of the companies stated that they could not attain their objectives in process quality”. Altogether, the results revealed that the companies within the industries being investigated by Slamanig and Winkler’s study lack considerable knowledge and expertise in managing their product change projects in their supply chain networks.

A significant proportion of the problem is caused by poor planning and information exchange. With increasing complexity of the product and the production process, it is assumed that the problem is also valid within a production system and not only across the supply chain. Such complex products are, for instance, integrated circuits (ICs) and their production processes, which are said to be the most complex ones one can imagine in today’s manufacturing industry.

For this reason, the ramp-up of a new product is still an individual project (ramp-up project) instead of a routine process, although some companies have developed technical concepts and business models in order to lower the risk of product ramp-up. Such measures include aspired quality gates during the ramp-up project and a modular product design in order to maximize the reuse of existing production knowledge.

The trend towards individualized products and shorter product life cycles, which is expected to be accelerated by *Industrie 4.0*, enforces companies to improve the performance of product ramp-up projects. The manufacturing industries have to deal with shortening of product life cycles and an increasing number of new products in their production lines [9] [10]. As a consequence, *product ramp-up projects have to be better predictable with respect to their costs and durations*

but also concerning the achieved initial yield. Moreover, costs and duration of ramp-up projects have to be minimized, and the initially achieved yield has to be maximized.

The introduced *K-RAMP ontology matchmaking process* may contribute to these challenges by reducing the ramp-up time at the target production system due to an *automation of gathering knowledge*. Moreover, it may increase the potential of reuse of existing knowledge due to the introduced ontology models and the matchmaking process itself. The introduced ontology models lead to a *structure of production basic data* of production systems which is *beneficial for a more efficient product ramp-up* within enterprises but also along the supply chain which is envisioned to be highly digitalized in the near future.

1.3 Problem statement

The time span within which a new product is ramped up after the end of product development to stable volume production is called the ramp-up phase (Figure 1.2). In accordance to Terwiesch et al. [11, p. 435], time-to-volume attracted reasonable more attention than time-to-market already to the beginning of the century. “The fundamental difference between time-to-market and time-to-volume is that the former ends with the beginning of commercial production whereas the latter explicitly includes the period of production ramp-up. Production ramp-up is the period during which a manufacturing process makes the transition from zero to full-scale production at target level of cost and quality.”

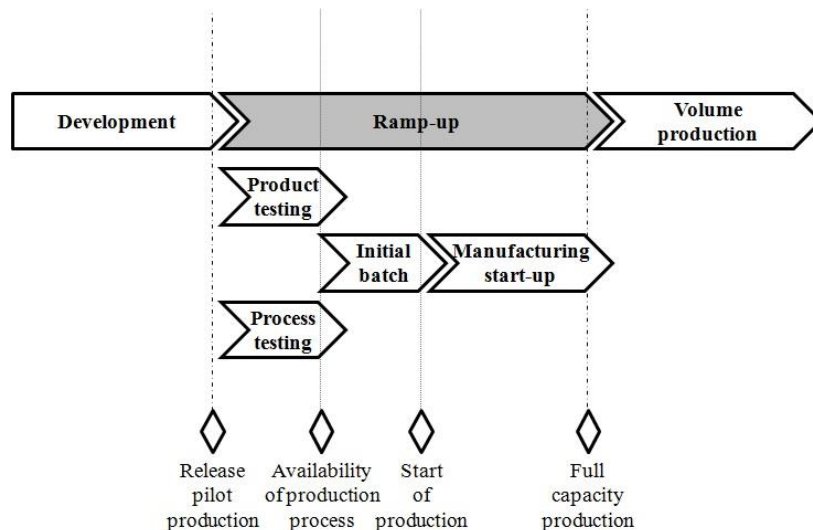


Figure 1.2: Location of the ramp-up phase of a new product based on [8, p. 484].

The ramp-up phase involves the adjustment of all elements of the production system, so that workpieces can be produced repetitively with the required quality, in the planned quantities and within the proposed production costs per workpiece. This is a complex and still error prone multidisciplinary task which involves engineers of several domains of expertise. This work solely focuses on activities, in order to setup the production system for producing workpieces in accordance with the new product’s specification. Accompanying measures, such as marketing, sales or finance, are not included in this thesis. As it is shown in the sequel, the major issue is related to knowledge management within the production system and along the supply chain of a production process.

The ramp-up phase usually begins when the development phase of the new product ends (Figure 1.2). From the development phase there is obtained a so called “bill of material” – abbreviated as BOM – of the new product. This is the hierarchical structure of all subparts or subproducts which compose the final product, including the detailed characteristics of each ingredient of the subproducts (e.g., size, weight, uniformity of surface).

Let us take a chocolate cake as an example. The chocolate cake consists of the chocolate gloss and the layered structure of dough and jam. The chocolate gloss consists of liquid chocolate and the liquid gloss. The layered structure of dough and jam is assembled by an upper layer of dough, a lower layer of dough and a layer of jam in-between. Liquid dark chocolate is a mixture of hot water and solid chocolate. This decomposition of the chocolate cake is continued until further decomposition is out of scope of the bakery, respectively the production system. For instance, the decomposition of the solid dark chocolate is out of scope of a bakery because it is acquired from a supplier. The solid chocolate is therefore called consumable material. Consumable material and subproducts have to follow specific characteristics because they influence the function of the product – like the portions of cacao fat, sugar and milk of the chocolate influences the expected taste of the chocolate gloss.

After the new variety of a chocolate cake is created successfully, the bakery has to ensure that other occurrences of this chocolate cake (a.k.a. workpieces) are produced with repeatable quality. There should be as little as possible variation with respect to quality from one occurrence to the other. In case of a chocolate cake, the quality can be determined by matching features like the taste, the uniformity of the surface of the gloss or the softness of the dough. In order to ensure this little variation, an accurate process plan is required. This process plan is known as the recipe of the cake. In more general terms, in the context of this work, it shall be named “the device-independent process plan”.

Why is it a device-independent process plan? Usually the recipe of the cake does not consider specific handling instructions of the used kitchen tools, like the mixer, the oven or the hot plate. This set of handling instructions is too specific as it focuses on a particular tool set. After developing the recipe of the cake, it must be possible that the cake can be baked with little variation of quality, with large volume in different dislocated bakeries and by using different tool sets.

In order to bake the chocolate cake at a particular bakery, device-specific handling instructions must be specified for the respective bakery’s tool set, using

- the knowledge about the BOM,
- the device-independent process plan, and
- the experience about the capabilities and usage of the local tool set.

These device-specific handling instructions comprise

- device-specific process instructions on the operation of locally available devices, such as oven-specific programs and time spans of heating phases which need to be considered during the baking process as well as
- device-specific control instructions in order to decide about the completion of the need for re-adjustment of process operations, for instance, the decision when the baking process has been completed based on the color of the dough in the oven.

Device-specific process instructions include the *setup of local devices* in order to treat workpieces properly, while device-specific control instructions enable *device-specific decision-making during one particular execution* of a process operation (process job) while treating a workpiece.

Recalling the chocolate cake example, the strength of the bakery might be more than creating a new variety of one cake product occasionally and to produce thousands of instances of it (compare Type 3 of Figure 1.1). Assuming that the strength of the bakery is the creation of customizable cake products, instead of hundreds of occurrences only a few copies are created upon customer requests by specifying the taste, the structure, the color or the size of the cake (more likely Type 2 of Figure 1.2). In such a business model, the bakery must be able to develop appropriate handling instructions as fast as possible and as accurate as possible. Speed is essential because the customer wants the cake as fast as possible, the tools of the bakery need to be utilized to be able to pay for their investment, and the personnel must be utilized as well. Accuracy is essential to avoid failures during baking and therefore losses due to time, wasted material or energy.

As previously highlighted, this business model is exactly the one which is faced by most manufacturing enterprises today. Also trends like decreasing product life cycles or the lot size 1 are challenging these enterprises. Emerge of *Industrie 4.0* will further accelerate this trend. The faster and the more accurate the setup of the devices of a production system can be adapted in order to produce workpieces according to new product specifications the lower are the costs to be faced during the ramp-up phase.

The development phase provides a device-independent process plan because this process plan does not comprise specific details in order to setup individual devices. Such devices, using the general terminology of the manufacturing industry, are production machines or measurement systems. Both in common are named equipment of a particular production system. The need for device-independent process plans is caused by the differences between the equipment of the volume production line (the target production system) where workpieces of the new product shall be produced and the equipment of the pilot line (original production system) where only the first workpieces were created, for development and evaluation purpose. The differences between both production systems are caused by equipment which was acquired from different suppliers and at different times. Equipment variations are therefore caused by the variation of equipment structure across vendors as well as by the age and thus the different stages of technical progress of equipment.

In order to face this challenge, some companies are following the strategy of “copy exactly”, where every production system is an exact copy of a production system template which is following enterprise-wide design rules. Using this approach, the complexity of a product ramp-up project is reduced significantly because the information of the original production system simply needs to be transferred to the target production system without modifications. No additional assumptions need to be performed because equipment and control software in both production systems use exactly the same configuration.

However, Terwish et al. [12] highlight that although the copy exactly approach sounds attractive it is coming with a price. Copy exactly requires identical production equipment for every production system thus reducing complexity of change in case of transfer of products between production systems. As a consequence, for complex production processes, either significant investment is needed for leading edge production equipment in all production systems simultaneously or increasingly out-dated equipment has consequently to be used. The latter must

be then used for production of new leading-edge products as well. Probably not all companies want or can deal with such restriction. Therefore, it is also admitted by Terwiesch et al. [12, p. 4] that, for instance, most semiconductor manufacturers still favor a much more aggressive process change during product ramp-up and do not follow the copy exactly approach. Therefore, the discussed approach is beneficial for a broad range of ecosystems of industrial manufacturing.

Throughout the statements in Chapter 1.2 it becomes obvious that the manufacturing industry is still struggling to perform product ramp-up project within the planned time and budget as well as to achieve the planned initial quality at the start of volume production. In other words, usually discrepancies between plan and reality are observable. The less a new product deviates from its forerunner products the more reliable a ramp-up project and its outcome will be planned and the lower the deviation from the reality will be. This sequence of implications is true because a maximum of production knowledge can be reused implicitly if there is a minimum of variation between the new product and its forerunner products.

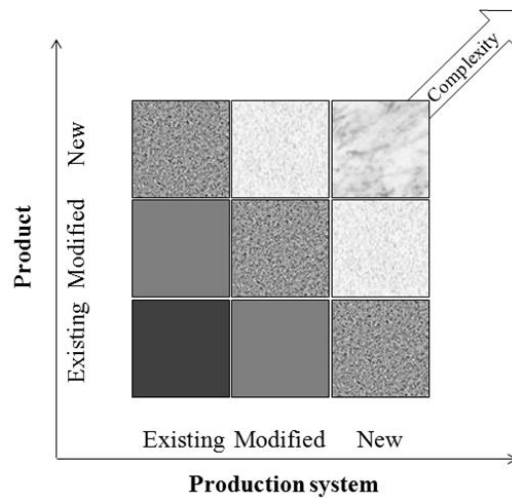


Figure 1.3: Increasing complexity of ramp-up based on variation of product and production system - based on [13].

The ability to the maximum reuse of existing knowledge of forerunner products, including handling instructions of process operations, of the target production system is a critical success factor for a product ramp-up project (Figure 1.3). The more the new product deviates from its forerunners or the more the target production system deviates from the original production system the more complex the task of gaining and recombining of knowledge (learning) will be.

The diagram in Figure 1.3 maps the *degree of novelty* of a new product compared to its forerunner products (existing knowledge) to the *deviations* between the original production system and the target production system. The “Product” is also the placeholder for every subproduct which composes the new product. In the same sense, the “Production system” in this chart is equivalent to each process segment, a subsequence of the overall production process, which makes up a part of the overall production process. The diagram has to be read as a mapping between each subproduct and the respective process segment which produces this subproduct. A new product may be composed of existing subproducts (e.g., standard parts, reusable forerunner components), and also the respective process segments of the original production system and the target production system may have (almost) no differences. This is particularly the case if the same production equipment and metrology systems can be used in both production systems for producing the respective subproduct. The complexity of change is very low in this case. Recalling

the protagonists of the introductory story, Jeff, he may be familiar with folding a spongh dough and he uses even the same handmixer as his mother. Therefore, neither the subproduct nor the procedure of its creation (process segment) is new for him.

The opposite corner represents new subproducts (e.g., due to an innovative feature of the product) or at least modified subproducts as well as increasingly significant differences between the structure of process segments of the original production system and the target production system. In this case, the complexity of knowledge transfer is very high. In the context of the introductory story, Jeff has no experience with the subproduct chocolate gloss, which is accordingly new for him. He is also using a different type of equipment and has therefore to use a different procedure than his mother.

Gaining knowledge during the ramp-up of a new product is a reasonable factor [14, pp. 256-259]. For instance, looking to the semiconductor manufacturing industry Oh shows, “that increasing yield of the 64M DRAM from 20% to 80% lasted approximately one year, with significant dependency on the learning curve of labors and stabilization of the new production process” [15, pp. 32-34].

Companies are implementing organizational measures and technical measures in order to reduce this complexity. One strategy is to focus on standardized parts in products thus reducing the number of new subproducts [16]. The other strategy is increasingly uniform equipment across all involved production systems [12]. Certainly, the combination of both strategies is also applicable. However, as already highlighted, there are limitations with respect to both strategies still resulting in challenges of non-deterministic ramp-up of new products.

Summarizing the previous sections, the outcome of a product ramp-up project is still difficult to predict. The essential task during a ramp-up project is the specification and evaluation of device-specific handling instructions which can be used to produce workpieces in accordance with the new product’s specification by utilizing production resources of the target production system. The complexity of a ramp-up project increases with the complexity of the product and the quality demands for its features. This increasing complexity has an immediate impact on the complexity and required precision of the underlying production process.

Cost, duration and the achieved initial yield are the metrics for measuring the success of a ramp-up project. Most manufacturers have to deal with heterogeneous equipment across the original production system and the target production system. They are therefore not able to copy information about the new product exactly between both production systems. A crucial factor of success is the ability to reuse existing production knowledge for planning and during execution of a product ramp-up project. From this assumption, the research questions, objectives and hypotheses of this work are derived.

1.4 Research questions and objectives

The systematic of research questions and objectives of this work are visualized in Figure 1.4. In detail, the following questions need to be answered:

- (Question 1) What is the design of a multidisciplinary knowledge base which is able to provide an automated matchmaking process of (1a) product-related design knowledge and (1b) device-independent process-related knowledge from an original production system and (2a) product-related design knowledge and (2b) comprehensive process-related knowledge from the production of forerunners at a differently

structured target production system, in order to generate recommendations for information reuse at the target production system?

- (Question 2) How is a multidisciplinary knowledge base for answering (Question 1) implemented with features of the Semantic Web?
- (Question 3) How is the ontology model of a knowledge base as requested by (Question 2) structured and designed solely with specifications of the Semantic Web?
- (Question 4) How are the ontology model and additional hybrid components of a knowledge base structured and designed by maximizing the use of the Semantic Web?

From the perspective of the research on *knowledge representation in manufacturing industries for mastering decreasing product lifecycles*, an answer to (Question 1) is the foundation for transferring undetermined product ramp-up “projects” into deterministic product ramp-up “processes” by the help of automated reasoning on multidisciplinary interconnected knowledge assets from the domain of product engineering and process engineering across multiple production systems. Answering (Question 1) contributes therefore particularly to improved knowledge models within the scope of an enterprise.

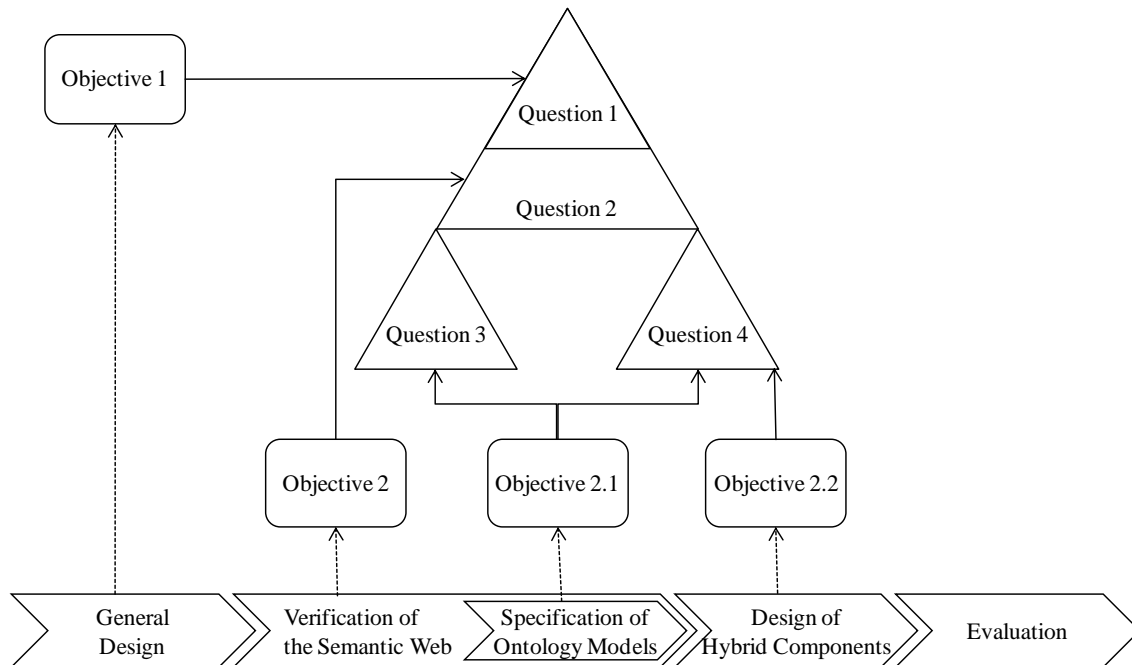


Figure 1.4: Structure of questions, objectives and research activities.

However, the envisioned digitalization of the supply chain, emerging business models of production services or individualized, customizable products transfer the challenges of a deterministic and efficient product ramp-up from the scope of an enterprise to the scope of the whole supply chain. Therefore, (Question 2), (Question 3) and (Question 4) are asking for an implementation of (Question 1) using the Semantic Web and thus motivating the research on *the application of the Semantic Web for mastering an important aspect of digitalized supply chains in manufacturing industries*.

These questions lead to a series of research objectives which are introduced in the sequel.

- (Objective 1) A general design has to be created in order to verify (Question 1)

(Objective 2) Mapping of the general design to Semantic Web specifications in order to answer (Question 2).

(Objective 2.1) Specify one or more ontology models which answer (Question 3).

(Objective 2.2) Design a hybrid architecture including one or more ontology models which answer (Question 4).

The methodological approach to achieve those objectives is described in the sequel.

1.5 Methodological approach

1.5.1 Research tasks

In order to achieve (Objective 1), a general design of an interdisciplinary knowledge base, for matchmaking of design knowledge of a new product and production knowledge for the production of forerunners, is created (*General Design*). (Objective 1) is achieved, if an appropriate design for automated matchmaking can be created thus answering (Question 1).

This *General Design* results in decomposition structures of

- functional requirements,
- of information models which provide exactly the information needed to satisfy the functional requirements, and
- of procedures in order to collect, calculate and imply information in accordance to the information models.

Axiomatic design is used throughout the *General Design* and is introduced in more detail in Chapter 1.5.2.

By using the functional requirements, the information models and the procedures from the *General Design*, it is determined which features of the Semantic Web are useful for which purpose and whether there are technical needs resulting from the *General Design* which cannot be satisfied by Semantic Web at all (*Verification of the Semantic Web*). One essential activity of this task is the *Specification of Ontology Models* which are based on the information models of the *General Design*. It is an important principle of the *Specification of Ontology Models* phase to maximize the utilization of specifications of the Semantic Web (e.g., for reasoning, querying) which are widely supported by software products. In the event of need for a hybrid solution, respective additional components are derived from the technical needs which cannot be solved through those standardized specifications of the Semantic Web (*Design of Hybrid Components*).

The theoretical transfer of a chocolate cake recipe from an original bakery to a target bakery and the matchmaking between the design knowledge of the chocolate cake and design knowledge and production knowledge of exemplary forerunners is implemented as case study and used for the purpose of *Evaluation*. The details about this *Evaluation* are described in Chapter 1.5.3.

1.5.2 Application of axiomatic design

The *General Design* of K-RAMP's information model is systematically derived from customer needs by utilizing the principles of axiomatic design [17]. The customer needs are extracted from (Question 1). According to Suh [18], axiomatic design is a structured design

methodology which leads from the customer needs of a problem to the optimal solution. Suh proves this fact based on several examples [18, pp. 317-372].

The design for solving a problem is performed by mapping between particular design domains which are related to each other as shown in Figure 1.5. The customer domain comprises the expectations from the customer’s perspective by the use of customer attributes (CA). CAs are answering the question: “What does the customer need?”. (Question 1) is the root for the definition of customer attributes in the context of the K-RAMP design.

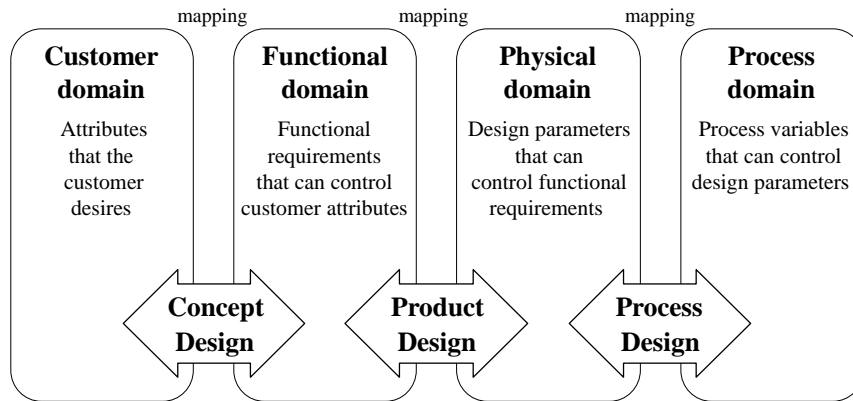


Figure 1.5: The conceptual structure of axiomatic design – source [18].

Within the functional domain, the minimal set of independent functional requirements (FR) is specified which must be provided in order to fulfill the customer needs completely. The physical domain describes the minimal design needed in order to implement the FRs. The question, “how must the solution be designed in terms of key physical variables in order to satisfy the specified functional requirements?”, is answered through this specification of design parameters (DPs). Finally, the process domain specifies the key process variables (PV) in order to generate the specified DPs. The answered question of the process domain is: “How does the solution act in order to generate the design specification?”

Suh recommends slight variations of interpretations, particularly for FRs, DPs and PVs, depending on the specific field of expertise where axiomatic design is applied [18]. Applying axiomatic design for software systems, associates the functional domain with the requirements specification of a software development process, the physical domain with components, classes, attributes and associations (in object-oriented terminology) and the process domain with sub-routines or methods.

Through the *General Design* these suggestions are kept in mind with slight enhancements in order to consider a system design based on predicate logic. The information model of DPs comprises components, classes, attributes and associations (relations), but also implications rules. PVs comprise generic procedures, like data management, reasoning or asserting information items.

The *General Design* is starting with the essential CAs. FRs are specified and mapped with those CAs which are affected by the respective FRs (concept design). Consequently, the same procedure is performed by mapping DPs on FRs (product design) and PVs on DPs (process design).

Next, a first decomposition of the root CAs is performed by so called zigzagging between the functional domain and the customer domain [18, pp. 21-22]. The initial CAs are decomposed to support the definition of the initial FRs. Consequently, the initial FRs have to be decomposed to

answer the question, how the new CAs on the first decomposition level can be achieved by appropriate FRs. The same decomposition is cascaded to the physical domain and further to the process domain. Over several iterations, this process results in tree-like decomposition structures for each domain.

The mapping between elements of dependent domains on each decomposition level leads to design matrices. The concept design represents the mapping between the functional domain and the customer domain. The product design specifies the mapping between the physical domain and the functional domain. And the process design is the mapping between the process domain and the physical domain. While the decomposition of CAs is stopped after the first decomposition and the concept design is omitted, the product design and the process design is performed to the useful level of detail.

There are two axioms which have to be satisfied in conjunction with each design matrix [18, pp. 68-69].

- Satisfying the *Independence Axiom* ensures that, for instance, FRs are maintained independently of each other through appropriate DPs. In the ideal case, each DP allows to control exactly one associated FR (a.k.a. uncoupled product design). Equivalent statements are also valid for the concept design or the process design.
- Satisfying the *Information Axiom* ensures that, out of several choices, the design is selected which introduces a minimum of information content (lowest entropy). The chosen DP which introduces the lowest information content has the highest probability to satisfy a FR without the need for additional external information. Again, the equivalent statement also works for the process design.

An empirical examination of the results of this work was not possible due to lack of appropriate production environments. For this reason, axiomatic design was chosen as the procedure model in order to derive a comprehensible design from the formulated (Question 1) of the work.

K-RAMP is not a pure ontology engineering task. As it can be concluded from the research questions, it is not completely sure whether the research task can be solved solely by ontology engineering. For this reason, methodologies like METHONTOLOGY which are focusing on the engineering of ontologies [19] do not apply here.

However, comparing, for instance, METHONTOLOGY with the process model of axiomatic design for software systems significant similarities can be perceived. Like in the *Specification* phase of METHONTOLOGY, the applied design process starts with the definition of the scope of K-RAMP, the expected customer needs and the derived functional requirements. The specification of the DPs represent the *Conceptualization* phase. These specifications lead over to increasingly formalized (*Formalization*) descriptions of each information model as part of zig-zagging on DPs. The Unified Modelling Language (UML) is used for the conceptualization while predicate logic is used for the description of formal dependencies between the concepts. In this phase, also conclusions are made with respect to the reuse of existing information models (*Integration*). The *Implementation* and *Evaluation* are performed in separated process steps in this work. With respect to the *Implementation*, a maximum of OWL2 functional syntax is applied as formal language. Therefore, an almost direct mapping between the formalized results of the axiomatic design process and the *Implementation* phase is expected. As hybrid components will be possibly expected, their code fragments shall be specified in pseudo-code. This step is followed by the *Evaluation*.

1.5.3 Evaluation

The full execution of the evaluation procedure is documented in Chapter 4. Recalling (Question 1), a multidisciplinary knowledge base shall be able to provide automated matchmaking of knowledge from two sources of knowledge, the original production system and the target production system. The knowledge which is used for evaluation purposes is thus split into two knowledge domains, the knowledge domain of the original production system and the knowledge domain of the target production system. The original production system contributes with knowledge about a new product's design specification and device-independent process-related knowledge from the original production system. The target production system contributes with knowledge about design specifications of existing products (forerunners) as well as all process-related knowledge about the production of these existing products in the target production system.

Exemplary design knowledge and device-independent process-related knowledge of one new product (original production system) as well as design knowledge and all process-related knowledge of one or more existing products (target production system) are therefore used for the purpose of evaluation. It is assumed, that equipment of the target production system deviates from the original production system. This constraint is simply considered by not transferring any device-specific production knowledge to the target production system.

To implement an easy understandable verification, a case study is defined, which maps the production of a special variety of a Viennese chocolate cake. The mentioned chocolate cake shall be transferred from the original bakery in Vienna (original production system) to an existing target bakery somewhere else (target production system). The target bakery is already capable to bake lots of special sweet dishes and cakes. However, as a distribution partner, also the previously mentioned chocolate cake shall be baked locally in the future. Of course, the target bakery is equipped completely different than the original bakery in Vienna. For this reason it is not possible to transfer the entire information of the baking process as an exact copy (see problem of a product ramp-up as discussed in Chapter 1.3).

1.6 Structure of work

In Chapter 2 the state of the art, particularly the challenges and performed measures of companies are discussed in more depth. The challenge of product ramp-up in an IC-production is taken as best practice with respect to necessary ramp-up activities. The IC-production was chosen because of the high complexity of IC-products and the production process. However, IC-production has been also chosen because of the high quality of available product basic data and process basic data as well as organizational measures which have been taken already in those companies. This chapter is of particular interest because the architecture of management software and control software (the production-ICT) of IC-manufacturers represents the most advanced status quo with respect to production control and process control. Moreover, there are already concepts in place which maximize opportunities for reuse of elements of currently produced products (forerunner products).

Chapter 3 introduces the design of K-RAMP, the verification of the Semantic Web and the design of hybrid components in order to transform the research questions to revisable TBOX ontology models and enhancing procedures. The result of this chapter is K-RAMP from its ontology model perspective and from the perspective of procedures which overcome gaps between Semantic Web's standardized specifications and the designed information models. It is explained

how the approach determines reusable elements of forerunner products or of existing process segments for the creation of forerunner products.

In Chapter 4, the evaluation of the K-RAMP ontology models in combination with its hybrid components is described by the use of a generally understandable cake-baking case study. The basic idea of this evaluation logic is already introduced in the previous Chapter 1.5.3. Implicitly, this chapter also introduces a recommended structure of product basic data and of process basic data for upload to a K-RAMP knowledge store in real production systems. Chapter 5 reflects the implementation to the situation in production systems today and highlights the premises to be considered in order to support the concept of K-RAMP in an optimal way. In Chapter 6 the further contribution of K-RAMP with respect to *Industrie 4.0* is discussed. In this context, useful standardization initiatives and next research steps are presented.

2 State of the art

2.1 Considered existing knowledge domains

Throughout this chapter, the state of the art with respect to the ramp-up of new products is lightened from several perspectives. Each perspective is assigned to one specific subchapter. There is a particular reason why this chapter is organized that way. Just as K-RAMP shall be branch-independent, there are already branch-independent standards in place which also introduce a branch-independent terminology. In order to align the terminology of K-RAMP for the reader, Chapter 2.2 does not only summarize such standards and established data models. This chapter also serves as an introduction to the most relevant terminologies which is also used in the K-RAMP ontology models.

K-RAMP combines several puzzle pieces. Some of these pieces are methods and measures which are already applied in the industry in conjunction with product ramp-up. These measures and methods are, for instance, organizational strategies or technical strategies which have been introduced by companies for this purpose. Such strategies are one essential puzzle piece for effective and efficient reuse of existing production knowledge during the ramp-up of a new product and are discussed in Chapter 2.3.

An essential part of a ramp-up team's work is the qualification of segments of the overall production process of a new product. For this purpose, the general meaning and methodology of process qualification, as it is used in nowadays' manufacturing industries, is discussed in Chapter 2.4.

The qualification of process segments also requires supervision and control. These methods link the characteristics of products with the setup of production process operations and underlying production equipment. For this reason, such methods are already of interest during the ramp-up phase. This linkage is not the aim of K-RAMP in this work. However, it provides further insight to the complexity of a product's ramp-up in a specific production system. Therefore, Chapter 2.5 addresses the state of the art of statistical process control (SPC) and advanced process control (APC) in order to draw a comprehensive picture of tasks with respect to the ramp-up of a new product. The ability to adjust the setup of a process segment by APC in order to manipulate the process results is common practice in advanced manufacturing industries. K-RAMP does not yet consider these techniques.

The state of the art of the Semantic Web and related research works in domains which are relevant for K-RAMP is introduced in Chapter 2.6. This is motivated by the fact that Semantic

Web contributes with essential concepts for semantic modeling, rule-based reasoning, exchange of information, publications of useful ontology models or taxonomy models as well as algorithms for matchmaking of ontology models.

2.2 Industry standards and industry terminology

Since the late 1990s, computer integrated manufacturing (CIM) became state of the art in many industrial areas. System integration and system communication standards since then are targeting on easy plugging of domain-specific software applications. Beside others, some standards are specifying an aligned architecture for factory automation and manufacturing information exchange [1] [20] or broadly accepted terminology, relationships and actions with respect to manufacturing operations management [21]. The CIM Framework guideline of International Sematech [2] introduces an architecture guideline, which is today implemented by most manufacturers of integrated circuits (ICs). Although the introduced data models and interfaces are focusing on the production of ICs, there are also generic abstraction layers which are useful for discrete manufacturing in general. The VDI/VDE 3682 standard [3] provides guideline for the specification of production processes.

The terminology of K-RAMP shall be familiar for the engineers and software experts of the manufacturing industry who are using it. Moreover, outside of the scope of K-RAMP, it shall be possible to implement automated procedures in order to transform existing CIM data to ABOX ontology models upon K-RAMP's TBOX models and vice versa. Therefore, the K-RAMP's TBOX shall comprise branch-independent conceptual models which are based on the familiar terminology and semantics of these standards.

In the ISA-95 standards [1, p. 40], the different main areas of exchanged information are highlighted, which are

- information required to produce a product,
- information about the capability to produce a product and
- information about the actual production of a product.

In this standard, the most generic and industry-independent data-models are specified. With [20], the industry-independent attributes are introduced. Where necessary within K-RAMP, the same terminology is applied.

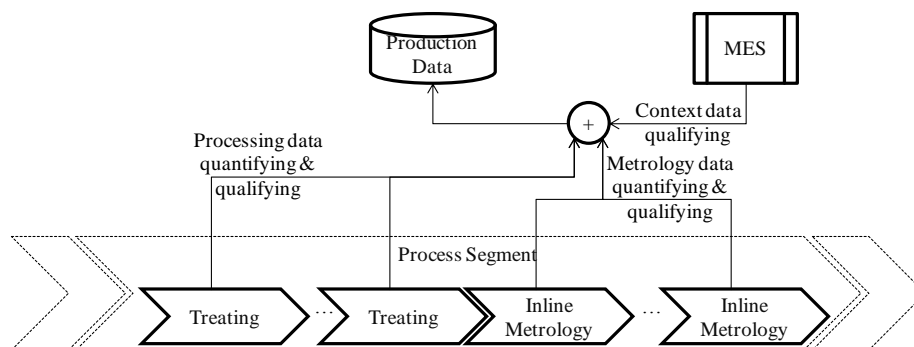


Figure 2.1: Structure of process segments as the source of quality data.

Throughout the concept of K-RAMP, process operations [2, p. 222] represent the atomic part of a production process which is directly associated with a set of particular machines as well as

appropriate machine recipes and handling instructions for human operators. A process operation represents either a material processing action (value-adding performance) or a material measurement action (support performance). An arbitrary sequence of process operations followed by one or more metrology steps represents the already introduced term “process segment” [20, p. 48] (Figure 2.1).

In the production system, subproducts are created or treated by manufacturing resources by executing the sequence of process operations within a process segment one after the other. Therefore, the sequence of process operations within a process segment depends on the utilized manufacturing resources. Process segments and process operations are thus concepts of the device-dependent knowledge of a production system which cannot be transferred easily between heterogeneously equipped production systems.

The creation of subproducts can be also outsourced to suppliers. In this case, a process segment of the production system, which is named “goods receipt”, measures the incoming subproduct (a.k.a. consumable material) of the suppliers. Such process segments comprise only of one or more process operations for material measurement (metrology operations) to ensure that the received parts are delivered to the production with the required quality. Therefore, the existence of a metrology operation is in common for all process segments, independently whether the process segment is treating material as part of the production process or whether the process segment does only measure received parts of suppliers during goods receipt.

In combination with a particular setup, each process segment is qualified for a specific process capability. A process capability can be treated as an agreement about the result of a process segment if a particular setup is applied [20, p. 94] [2, p. 226]. A subproduct requires specific process capabilities for its creation or its composition from other subproducts. Through the introduction of process capabilities, this requirement is completely independent from any production resource. Therefore, process capabilities are an essential design concept of production systems in order to separate specifications of subproducts or products from the device-specific setup of production resources.

A process plan is a sequence of process segments (see process flow context in [2, p. 221]) which needs to be executed in order to create the new product. The hierarchical composition of a product from subproducts and the specific process capabilities, which are needed for each composition step, result in a directed path of process capabilities. Consequently, this direct path of process capabilities also causes a directed path of process segments and thus a directed path of process operations. Therefore, a process plan comprises all information which is needed for the creation of a product by utilizing available production resources.

K-RAMP’s matchmaking determines reusable information about existing subproducts and process capabilities of the target production system based on the information along the composition structure of an introduced product. This contribution helps the ramp-up team to build a process plan of a new product faster.

2.3 How factories master the ramp-up complexity

In Chapter 1.3 some strategies of manufacturing companies in order to master the complexity of new product ramp-up are already discussed. A possibility is the “copy exactly” approach in which all production systems are equipped identically and therefore also the production knowledge can be copied exactly between two production systems. As a consequence, there is no risk in conjunction with any product ramp-up. However, it is also stated that due to several reasons most

manufacturing companies have to deal with heterogeneous equipment across and even within production systems.

In order to reduce complexity during a product ramp-up, IC-manufacturers, for instance, have introduced the concept of process technologies and silicon intellectual property (silicon IP). Silicon IP comprises off-the-shelf functions like A/D converters, memory and processors [22, p. 19] which can be randomly combined during the design of a new IC-product. Silicon IP consists of a particular layout of electronic elements which is mandatory in order to implement some function and helps to prepare the physical setup of production equipment (e.g., reticles) accordingly. Moreover, silicon IP comprises an appropriate stack of material layers which is needed to achieve the expected electrical behavior of such a function. If a function is needed, the appropriate silicon IP is reused to realize this function within an IC-product. However, there is more than one function within an IC-product. This fact and the stack of material layers lead to process technologies.

Also process technologies (e.g., CMOS) are commonly used by IC-manufacturers. A process technology is defined by a specific stack of material layers which is build by a particular sequence of process segments on a circular ultrathin disk of monocrystalline silicon (a.k.a. silicon wafer). Also the size range of features of each layer is specific for process technologies. Each process technology is therefore linked to a dedicated set of equivalent process plans which can be performed to build the requested stack of layers. Therefore, process technologies represent a link between the process plan and the design of specific IC-products.

To some extent, process technologies can be seen as reusable templates which can be used to build individual IC-products. Variations of individual products are achieved by modification of the layout of single layers, the thicknesses of layers, controlled impurities of the material on each layer or other means of parameterization.

Because of this branch-specific method of process technologies and modularization of products by the use of silicon IP, the semiconductor industry provides a good pattern for customization of products using templates and unified underlying production processes. Some other industries, like the production of printed circuit boards (PCBs) apply this method as well [23].

Also the automotive manufacturing industry uses the concept of product templates. In this industry, product templates are called platforms. An outstanding example of a platform approach is Toyota's policy concerning its car models. "Toyota is currently launching new generations of its successful car models that utilize more than 70% of their forerunners' components, and the platforms of these cars have remained largely constant through successive car generations" [24].

The concept of process technologies or product platforms can be considered similar. For instance, Ong et al. are summarizing possible design principles for the design of product platforms independently of a specific branch of industrial manufacturing [16, pp. 81-112]. The approach of product platforms has therefore general validity in the manufacturing industry for reduction of the complexity of management of product variants and therefore the complexity of product ramp-up projects. Facing the emerge of *Industrie 4.0*, this method will be rather likely adopted by other manufacturing industries as well.

2.4 Quality assurance in a production process

Each product has to fulfill a set of functional requirements which satisfy certain customer needs. The satisfaction of functional requirements is measured in every production system during

the process segment named “quality assurance”. Commonly, this is one of the last segments in the production process.

During execution of the quality assurance, samples are taken and, from these samples, predefined characteristics are measured which are specified in the functional domain. It is verified, whether the measured characteristics are located within the product-specific specification limits. If this is the case, the group of workpieces, from which the sample was taken, has passed the test. Otherwise, the entire group is either discarded or, possibly, cost-intensive reworking must be performed. Similarly, in advanced production processes samples of the semifinished workpieces are removed and tested almost after every single processing action. During these inline metrology steps it is not possible to test the function of the final product. However, it is possible to test characteristics of the design specifications of the subproduct.

The ratio of measured parts depends on the sampling rate, which can be calculated with statistical means as they are, for instance, described by [25] or [26]. It is therefore assumed as premises of K-RAMP, that all process segments deliver metrology data based on a statistically meaningful sampling rate.

The first pass yield (FPY) is calculated as the portion of defect-free workpieces (2.1) of a specific subproduct which are passing a particular process segment [27, pp. 179-180] at the first pass (without rework). The FPY is calculated from the results of each inline metrology operation and the quality assurance process segment.

$$FPY = \left(1 - \frac{\text{Count of defective parts}}{\text{Count of all parts}}\right) \cdot 100\% \quad (2.1)$$

The so called rolled throughput yield (RTY) represents the portion of all produced workpieces (2.2) which are passing the overall production process at the first pass, which means that there is no potential rework required in order to correct defects [27, pp. 179-180]. The RTY is calculated for each product which is produced in the production system. The FPY_i are considered to be independent from each other accordingly.

$$RTY = FPY_1 \cdot \dots \cdot FPY_n \quad (2.2)$$

High quality, being close to the optimum of production costs, requires a high RTY and thus low costs with respect to expensive rework loops (aside of other cost drivers). As RTY strongly depends on the performance of each FPY, the same requirement is also valid for each process segment. Keeping the initially achieved yield as performance metric of a ramp-up project in mind, RTY and the FPY_i are becoming obviously critical performance indicators of any ramp-up project. At the beginning of a product ramp-up project, it is therefore important to determine the potential of reusable and thus already mastered subproducts and process capabilities with respect to their FPY.

If no reuse of subproducts of forerunners or reuse of existing process segments is possible, new or modified subproducts or process segments have to be introduced. Under certain conditions it might be possible to derive new subproducts or process segments from existing ones by appropriate adjustment of characteristics. However, in all cases it has to be validated whether the new or modified subproduct respectively the new or modified process segment meets expected FPY_i and RTY.

This validation is called the “qualification of a subproduct” if the subproduct is provided by a supplier along the supply chain (a.k.a. consumable material or consumable), and it is called

“process qualification” in case of a process segment which is performed within the domain of the production system. Consumable material is qualified if it satisfies repeatedly the expected specification. Process segments are qualified for a particular process capability if the specification of this process capability is repeatedly satisfied. In case of consumable material, this specification is usually equivalent to the functional specification or the design specification in accordance to agreements with the suppliers. In case of process segments, as underlined before, this specification is represented by the process capability. A repeatedly achieved specification requires that each gauged characteristic of a workpiece which is treated by a process segment is within the specified ranges of a process capability the process segment is qualified for.

The specification range is delimited by specification limits (Figure 2.2) – commonly an upper specification limit (USL) and a lower specification limit (LSL). In exceptional cases also onesided specification limits are possible if the opposite side is delimited due to physical restrictions for instance. Usually, each consumable material or each process capability is specified by more than one characteristic, each with its particular specification range. While the qualification of consumable material is ensured within the production process of the supplier, the qualification of process segments has to be ensured within the domain of the own production system. Every workpiece which is produced with at least one characteristic outside of the specification range is counted as defective part and must be either scrapped or reworked until all characteristics are within their specification ranges.

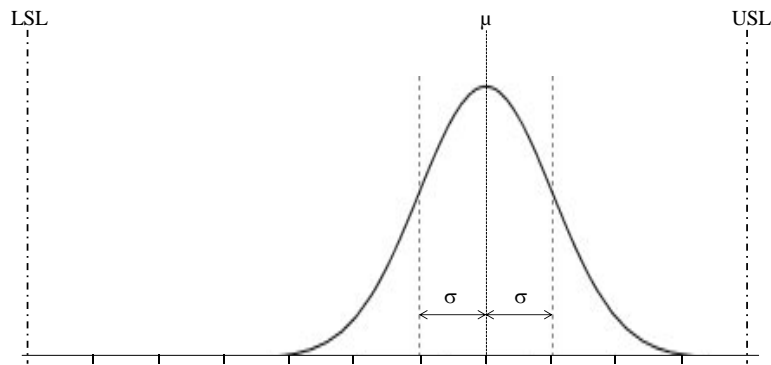


Figure 2.2: Exemplary normal distribution of measurements between specification limits (LSL – lower specification limit, USL – upper specification limit).

The critical process capability index (c_{pk}) is commonly used in the industry in order to quantify the capability of a process segment to avoid defective parts. Assuming normal distribution of the gauged values of a characteristic, the c_{pk} represents a multiple of three standard deviations (σ) from the shortest distance between the process average (μ) and the specification limits (LSL or USL)(2.3).

$$c_{pk} = \frac{\min(\mu - LSL; USL - \mu)}{3\sigma} \quad (2.3)$$

Based on the c_{pk} , it is immediately possible to derive the probability of measurements outside the specification range and thus the number of defective parts due to this particular characteristic. In high-quality production, the c_{pk} of each characteristic of a process segment (and consequently of each acquired consumable material) is 2.0. A c_{pk} of 2.0 is equivalent to 0.002 defective parts per million (ppm). Consequently, the FPY of a process segment is in close relationship with the achieved c_{pk} . With respect to the ramp-up of a new product, the c_{pk} is therefore the most critical performance indicator which is associated with the achieved initial yield of the volume production.

Being able to determine reusable subproducts or process capabilities thus helps to save qualification efforts for the production of introduced subproducts.

There is not always a normal distribution of measurements. For this reason, the ISO 21747 standard defines time dependent distribution models and standardizes the calculation of the c_{pk} for each type of distribution model [28].

2.5 Production control and process control

Chapter 2.4 introduces the need for process qualification in today’s production systems. During this qualification phase instructions are developed that accurately describe the settings of the controls and switches, as well as the handling of production equipment. These instructions have to be followed by human operators or by production machines during future volume production.

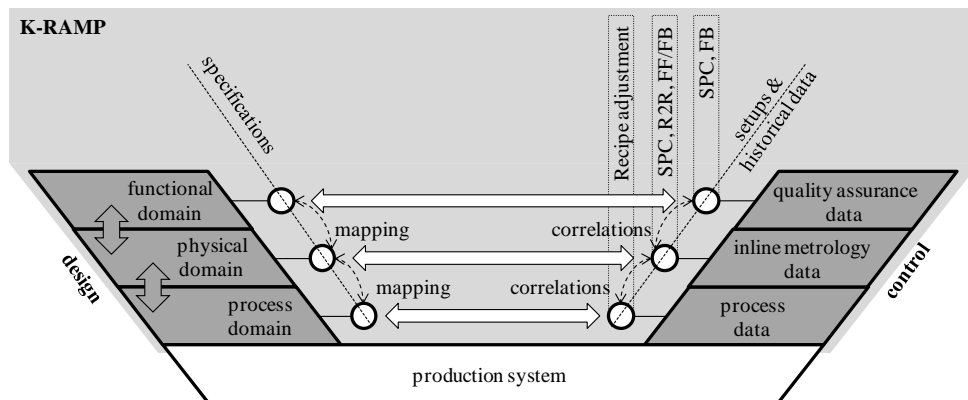


Figure 2.3: Overview of relationships between design and control of the production process of a product.

Through the discussion about the state of the art in this chapter, Figure 2.3 is referred to more often. The left leg of the shown V-model refers to the design phase of the product and the underlying production process but not to the production system hardware, as this is out of scope for K-RAMP. Important work about the systematic design of a product and the underlying production process from customer needs and from derived functional requirements were introduced with axiomatic design by Suh [17] [18]. A brief introduction to axiomatic design is provided in Chapter 1.5 because this approach is used for the system design of K-RAMP as well. However, implementing an approach which enables a more predictable product ramp-up project also requires a systematic concept for specifying the mapping between functional requirements, product design and the layout of the underlying production process. This idea is applied as part of best practices in advanced manufacturing industries and therefore considered as terminology in the left leg of the V-model of Figure 2.3.

The structure of the left leg can be also mapped to the common terminology of production systems. The functional domain covers the specification of product functions based on characteristics which are clearly specified and gaugeable. Each subproduct contributes with subfunctions which are specified accordingly. In order to ensure the implementation of these subfunctions and functions, each subproduct and the product itself have to follow a certain design, which is again characterized in a gaugeable way. In order to create subproducts or the product in accordance to their design specifications, the applied process segments must provide particular

capabilities. Therefore, the interface of the process domain is represented by afore introduced process capabilities. Design specifications of subproducts respectively the product must be aligned with the specification of process capabilities.

In case of instructions for human operators, the term “handling instruction” is commonly used. Handling instructions are made available to operators in printed or electronic form directly at the respective production equipment. An instruction of production equipment is called “machine recipe”. A machine recipe is an electronic representation of a machine setup. Fully automated production equipment is loading the appropriate machine recipe automatically and according to the workpiece to be treated. For semiautomated production machines, handling instructions may refer to dedicated machine recipes which have to be loaded by the operator manually. Moreover, the setup of a machine may describe activities in order to manipulate the physical machine (e.g., replacing of a drill).

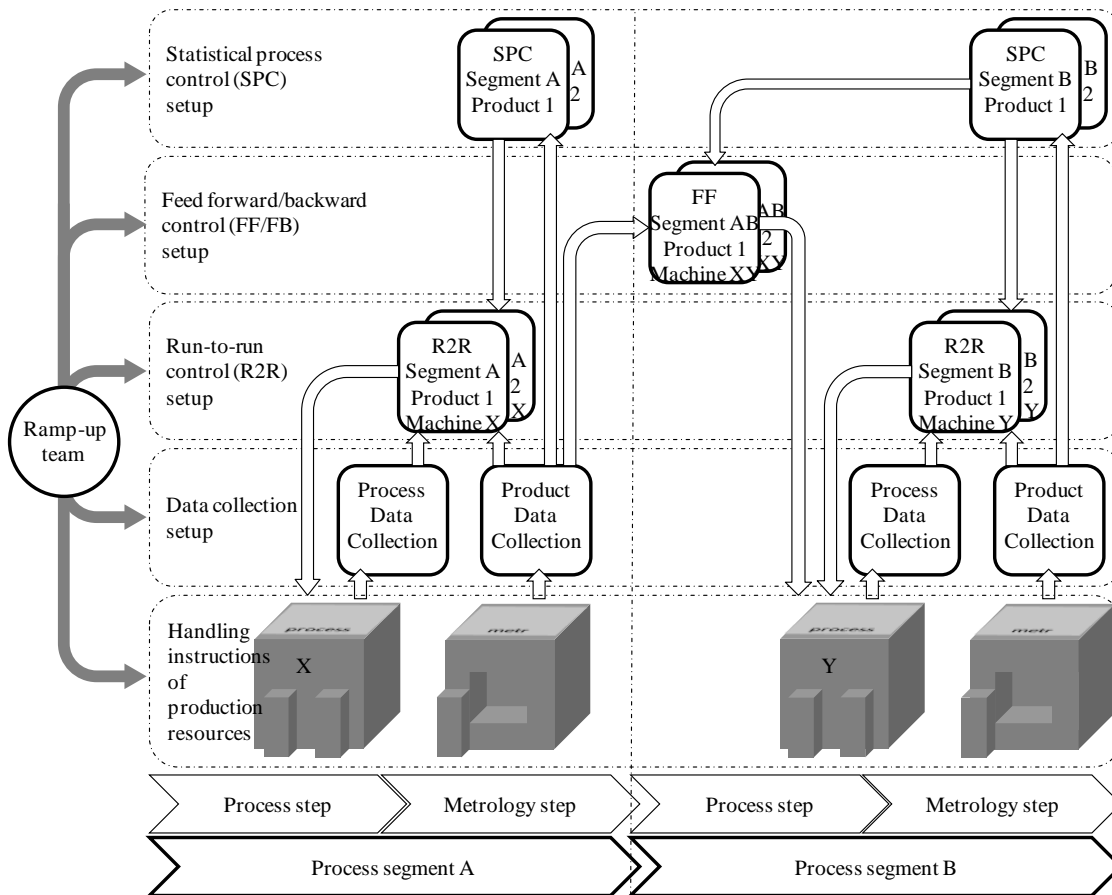


Figure 2.4: Assignment of data collection and control components to process segments and production resources (e.g., equipment).

Knowledge management in conjunction with the engineering of production systems is, for instance, discussed by Moser et al. [29], [30] or Konrad et al. [31]. However, K-RAMP does not focus on the engineering of production systems but on the association between product specifications and process specifications on the one side and collected production data on the other. The right leg of the V-model refers to the elements of the production process, their contribution to collected data and the ability to adjust these elements for improved process capability.

During a ramp-up project, it has to be ensured that new handling instructions or machine recipes become available for the production system. In advanced manufacturing environments, in particular in production systems with simultaneous production of many products, production control is performed by a manufacturing execution system (MES). It is one of the tasks of the MES to indicate the operator the correct handling instruction for processing the current workpiece appropriately. In case of fully automated production equipment, the MES indicates the production equipment directly. One outcome of the ramp-up project is therefore the process plan. For each of its steps the respective handling instructions and machine recipes have to be specified. Again, it is essential to maximize the reuse of existing process segments and therefore the reuse of existing handling instructions and machine recipes.

During volume production the stability of the overall production process and each of its process segments has to be supervised by means of statistical process control (SPC). Therefore, during the ramp-up project SPC-charts for specific product characteristics have to be developed accordingly. In particular, the specific control limits of SPC-charts have to be determined [26, pp. 309-312]. Similarities of subproducts compared to forerunners are considered for reuse of existing of SPC-charts definitions.

For instance, in modern factories for integrated circuits, almost every single process operation is followed by a metrology operation which measures samples. A process control system adjusts the process setup in real-time and triggers corrective actions immediately and automatically in the event of excessive process variation or error-related process changes. Over the years, process control systems were enhanced by a variety of powerful control components, which run generally fully automated. These control components usually require a product-specific control setup. The most common control components which are currently in use are shown in Figure 2.4 and provide the following functionality:

- Statistical process control (SPC), aligns data of measured samples with historical trends and control limits which are based on observed statistical data. In connection with its classical meaning, notifications to process engineers or immediate corrective actions are triggered automatically if an instable behavior of the controlled process is recognized [26].
- Run to run control (R2R) can be understood as logical complement to SPC. While SPC monitors metrics and triggers corrective actions in case of deviations, R2R acts through readjustment of processes before a deviation occurs. R2R uses pre- and post-process measurements of samples which are taken out of the controlled process, as well as the expected quality target (e.g., the thickness of a coated layer on the surface of a product) and knowledge about the relation between this quality target and a few setup-parameters of the controlled process (e.g., the duration of the coating process and the temperature in the process chamber during deposition of the coating material).
- Feed forward / feed backward control (FF/FB) considers dependencies along the process sequence. For example, the layer thickness of the photo resistor after the lithography process can affect the subsequent reactive ion etch-process (RIE) [32]. A FF model holds the information, how the RIE recipe has to be adjusted for compensation of photo resistant thickness. FB can be understood similarly to R2R with the only difference that not the immediately last process step but the setup of another one in the process history is adjusted.

- Data Collection – the previously introduced control components require production data to be recorded with high quality. Therefore, also data collection models need probably to be individualized for each product.

Depending on the complexity of the product and the production process, the structure of process segments and associated control components may result in hundreds of control setups which have to be reviewed and adapted during a new product's ramp-up. The important purpose is to ensure the proper monitoring and control of the quality of workpieces of the new product during the ongoing production process.

2.6 Semantic Web and Ontologies in the industry

Semantic Web technology introduces methods for creation of distributed information models [33, p. 20]. Assuming the interaction between a source production system and a target production system, this capability has to be considered as essential. Moreover, the resource description framework (RDF) which is actually the foundation of all Semantic Web specifications which are relevant for K-RAMP provides means for expressing information and exchange of information without loss of meaning [34].

RDF allows to represent information items as statements in the form of three resources called `rdf:subject`, `rdf:predicate` and `rdf:object`. The leading `rdf:` is the namespace which is used for all concepts being introduced by RDF. Using such statements it is possible to assert individuals and to classify them – like the statement `q:n1` “Couverture is of dark chocolate” is specified as

```
q:n1 rdf:subject :Couverture;
     rdf:predicate :isOf;
     rdf:object :DarkChocolate.
```

This statement can be used by another statement (a.k.a. reification) in order to express that a “Viennese chocolate cake has a couverture which is or dark chocolate” by specifying

```
q:n2 rdf:subject :VienneseChocolateCake;
     rdf:predicate :comprises;
     rdf:object q:n1.
```

Moreover, RDF is based on the extendible markup language (XML). Each item within an RDF statement is thus represented by an uniform resource identifier (URI) and therefore implicitly unique. Instead of the Turtle syntax which is previously used, each statement can be therefore also expressed in XML.

```
<rdf:Description rdf:about="q:n1">
  <rdf:subject rdf:resource=„:Couverture“>
  <rdf:predicate rdf:resource=„:isOf“ />
  <rdf:object rdf:resource=„:DarkChocolate“ />
  <rdf:type rdf:resource=
    „http://www.w3c.org/1999/02/22-rdf-syntax-ns#Statement“>
</rdf:Description>
<rdf:Description rdf:about="q:n2">
  <rdf:subject rdf:resource=„:VienneseChocolateCake“>
  <rdf:predicate rdf:resource=„:comprises“ />
  <rdf:object rdf:resource=„q:n1“ />
  <rdf:type rdf:resource=
    „http://www.w3c.org/1999/02/22-rdf-syntax-ns#Statement“>
</rdf:Description>
```

Due to RDF, Semantic Web introduces flexible means for storing and exchanging information items with globally unique identities. Off-the-shelf database products are provided by multiple vendors which are able to store information based on such statements.

RDF Schema (RDFS) [35] introduces a semantic extension to RDF and provides a data-modeling vocabulary for RDF data which comprises properties, domain and range restrictions of those properties as well as class hierarchies. Using RDFS, it is possible to define consistent vocabulary and to share this vocabulary between multiple databases. Another Semantic Web standard which is specified on top of RDF or RDFS is the Web Ontology Language (OWL) [36]. OWL introduces logical expressions (and, or, not), (in)equality of individuals, enumerations, cardinality constraints of properties and specific types of properties in order to express relations, like transitivity, reflexivity, symmetry or chained relations. It is also possible to specify disjoint relations between two classes of individuals. The most recent recommendation of the OWL specification is OWL2.

With OWL, the expression of information models is based on individual resources (individuals), classes of individuals (classes or concepts) and properties of resources. As OWL is build on the specification of RDFS and RDF, afore introduced features are also valid for OWL. There are three “species” of OWL which are separated to OWL Full, OWL DL (description logics) and OWL Lite. OWL Lite comprises a subset of features for creating classification hierarchies and simple constraints. OWL DL envelopes the feature set of OWL Lite enhanced by as much as possible features in order to maximize the expressiveness of the language while retaining computational completeness – all conclusions are guaranteed to be computable – and decidability – all computations finish in finite time. OWL Full envelops the feature set of OWL DL enhanced by all possible features without computational guarantees. Most commonly, Semantic Web databases provide OWL DL and OWL Lite provides too less features with respect to the K-RAMP information model. For this reason all further research about the utilization of Semantic Web specifications in conjunction with K-RAMP are also limited to OWL DL.

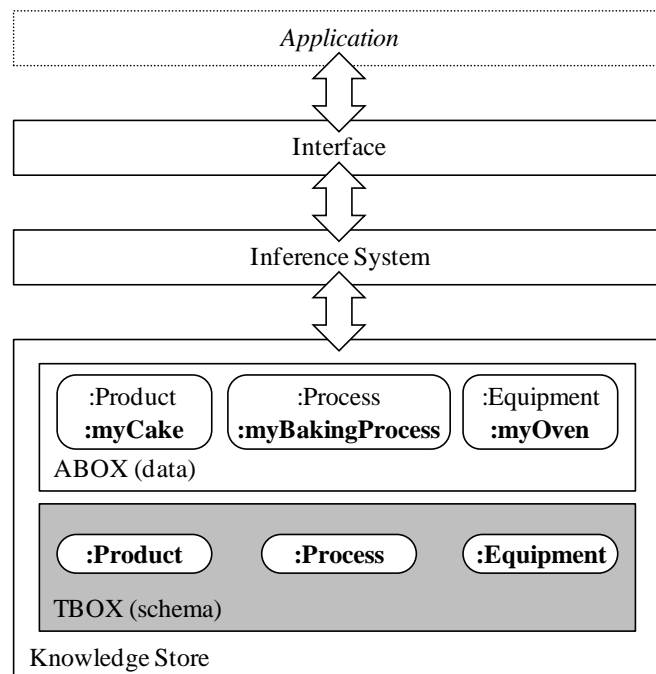


Figure 2.5: Schematic overview of an OWL DL architecture.

The architecture of an OWL DL knowledge base (Figure 2.5) comprises a knowledge store, an inference engine and an interface which is finally used by respective applications. The knowledge representation is strictly separated in two layers within the knowledge store. The terminology box (TBOX) specifies the schema of one or more knowledge domains. The assertion box (ABOX) utilizes the schema being introduced by the TBOX in order to create (assert) individual resources in a specific context. The K-RAMP information model is provided in the form of TBOXes. However, the cake-baking models being established for the case study are provided as ABOXes.

Each information model (ontology) of a TBOX or an ABOX is specified by means of OWL DL. Therefore, it is possible to share TBOX models and ABOX models between knowledge stores. Ontology editors, like Protégé, are used to specify such models. In order words, mature off-the-shelf products are used model definition, the management of information models and concrete data as well as for sharing of models and data. The focus is solely on the description of the information model.

The inference engine of an OWL DL knowledge store implements the interpretation of all forms of restrictions which are introduced by OWL DL. The most generic interface for data access is SPARQL [37], the query language of the Semantic Web.

Why are Semantic Web technologies applied for information storage and exchange in conjunction with K-RAMP? Due to its original intention, Semantic Web is closely integrated with other Internet specifications (e.g. HTML). Recalling the need of the problem to be addressed, information must be exchanged between production systems, either within the domain of an enterprise or across enterprises within the domain of a supply chain. This is the motivation, why the utilization of Semantic Web specifications is part of the research of this work. From the perspective of broad applicability in the future, Germany's strategic initiative *Industrie 4.0* [38, p. 40] indicates the need for a common approach concerning how to see things in production engineering, mechanical engineering, process engineering, automation engineering, as well as ICT and the internet. The concept to be discussed in the subsequent sections addresses such a common approach for the domain of product engineering and process engineering across dislocated production systems. Therefore, a common technology for information exchange and information storage has to be considered on the level of the production ICT in the manufacturing industry as well.

Ontology models are applied also in the arena of industrial applications. A significant portion of publications are in the domain of production engineering support. For instance, OntoCAPE introduces TBOX ontology models for the engineering of production equipment [39] [40]. Konrad et al. are particularly focusing on the ramp-up of assembly lines [31]. Winkler and Biffel introduce a multi-disciplinary engineering environment considering process automation and quality management aspects [41]. Willmann et al. published principle concepts about a deterministic process for the ramp-up of a new product in a production system by the use of ontology models and features of OWL [42] [43].

However, the distribution of ontology models and Semantic Web applications on the Internet is currently predominated by linkage of social data, governmental and geographical data, life science or multimedia. However, these domains demonstrate how it is possible to establish a growing network in particular knowledge domains. With respect to the future development of K-RAMP and the alignment of knowledge domains being introduced in Chapter 2.2, those existing applications of the Semantic Web and the alignment of ontology models may be a best practices pattern.

2.7 Summary

There are already several standards and guidelines in place which do not only introduce common terminologies for manufacturing industries but also information models in order to master production basic data and product basic data. The motivation for the development of these standards was, for instance, driven by the need for more efficient system integration projects or due to the need for a clear communication of software requirements between factories and vendors in case of software acquisition. By performing such system integration projects or by acquiring software along those standards, some industries have implicitly adopted the underlying terminology of these standards. Where useful, the same terminology is also adopted in the K-RAMP information model.

There are already structured methods and best practices at enterprises in place in order to reduce complexity from product ramp-up scenarios by fostering reuse of existing production knowledge. However, in case of product innovations there is still the need to generalize from existing knowledge to the new requirements. This interdisciplinary task still strongly depends on the performance of individuals. And even the appropriate and comprehensive reuse of existing production knowledge is still a communication-related challenge within ramp-up teams. The K-RAMP process contributes in this situation with automated reasoning of reusable production knowledge and derives recommendations for the ramp-up team members. Due to this contribution, the members of the ramp-up team are able to focus their work on the real necessary tasks as they are described in Chapter 2.4 and Chapter 2.5. This is particularly the case, if K-RAMP does not determine any opportunity for reuse or only an opportunity which requires adaptations of the existing processes. Completely new process segments need to be developed or existing process segments need to be used as base line for adopted process segments by the ramp-up team. In all cases the ramp-up team members need to qualify the new process segments for the new capabilities.

The Semantic Web and the application of ontology models are already arrived in some manufacturing industries. However, while the Semantic Web is rapidly emerging in other fields of the Internet it is still more on an academic level in the field of industrial applications. With the advent of Industrie 4.0 it is assumed that the underlying technologies will be also adopted in this field. K-RAMP builds only on specifications of the Semantic Web which are already broadly supported by a variety of software products. These specifications are RDF, RDFS, OWL2, SPARQL and SWRL. Due to this constraint, an adoption of K-RAMP shall be made easy.

3

The K-RAMP Design

3.1 Applied methods

3.1.1 Overview

This chapter comprises the results of the executed research plan, as it is visualized in Figure 1.4. The structure of the following subchapters is in accordance to the steps of the research plan. Consequently, Chapter 3.2 describes the results and the causal linkage between (Question 1) and the general design. Axiomatic design (see Chapter 1.5) is chosen as methodology to achieve a complemented system design with respect to the specification of (Question 1). The general design comprises the relevant functional requirements, the comprehensive information models as well as the procedures in order to deal with the information models. However, it does not provide a particular technical decision.

Standardized features of the Semantic Web are verified as an appropriate technology for the implementation of the general design in Chapter 3.2, thus addressing (Objective 2) and consequently research question (Question 2). The result of Chapter 3.2 is a prerequisite of Chapter 3.3 accordingly. Chapter 3.3 comprises also the design of ontology models which are based on the information models of the general design. Moreover, the ontology models and the applied standardized features of the Semantic Web also intend to cover a maximum of the procedures of the general design and are therefore immediately addressing (Objective 2.1) or (Objective 2.2) and the research questions (Question 3) or (Question 4). Finally, in this chapter also the gaps are highlighted which cannot be covered by those standardized features of the Semantic Web. Chapter 3.4 addresses these gaps and provides proposals for the implementation of complemented hybrid components which results finally results in (Objective 2.2) and verifies research question (Question 4). The following discussion is thus anticipated, that it is not possible to implement the designed information model solely by features of OWL DL.

3.1.2 Usage of Unified Modeling Language

The unified modeling language (UML) is applied for the specification of static models [44]. UML is applied in a way which considers already the later implementation of ontology models with OWL. For this reason, attributes are not specified according to the UML-notation but as association to classes of the respective data type.

Only directed associations are used between classes. This decision allows an easy translation of association to object properties in ontology models of OWL DL.

3.1.3 Consideration of design patterns

The General Responsibility Assignment Software Patterns (GRASP) [45] are partially used for decision-making. For instance, the low-coupling pattern or the high-cohesion pattern is occasionally used to decide the assignment of a class to an information model.

3.2 General design

3.2.1 Initial mapping of design domains

The general design is the first research task which needs to be performed in accordance to the research plan in Figure 1.4. The design phase starts with the analysis of (Question 1) and its transformation to the root CA (Table 3.1). The specification of CA refers exactly to research question (Question 1). The summary of needs comprises an automated matchmaking process. This matchmaking process is applied on two different sources of knowledge – the original production system and the target production system. Product-related design knowledge is provided by both sources. Process-related knowledge is provided comprehensively (device-independent and device-dependent knowledge) by the target production system only. The original production system only provides the device-independent domain of process-related knowledge. As an important constraint, the “copy exactly” approach must be excluded as premise. Consequently, if it is possible to design a solution which satisfies these needs, it is possible to answer (Question 1).

Summary of the root customer attribute – root customer need

CA = A knowledge base which is able to provide

1. an automated matchmaking process of
 - (1) product-related knowledge and device-independent process-related knowledge of a new product from an original production system and
 - (2) product-related knowledge and all process-related knowledge, from the production of existing forerunners.

Table 3.1: Summary of the root customer attribute.

In order to meet this CA, the root FR which shall be satisfied by K-RAMP is specified as summarized in Table 3.2. An automated matchmaking process is required as well as access to product-related knowledge and device-independent process-related knowledge from two different production systems. Device-specific process-related knowledge is required only from the target production system.

There are the following constraints to be considered during the design process (Table 3.3). Constraint C_1 states that the expected solution must be independent of any industrial sector (industrial branch). Particularly, the support of any discrete production processes shall be supported. Constraint C_2 covers the fact that the device-independent process-related knowledge must reflect two differently structured production systems. This constraint excluded immediately the “copy exactly” approach. Constraint C_3 ensures that each information element is unique across both involved production systems. Constraint C_4 limits the time needed in order to achieve a stable and unique result.

An appropriate root DP must be specified next. The root DP to be specified must comprise a summary of the technical measures which must be taken in order to satisfy the previously discussed root FR. DP is summarized in Table 3.4 and explained in the sequel.

In this work, design parameters are used to comprise the static part of the design (e.g., classes and associations between classes as well as implication rules). The root DP and its decomposition structure, which is discussed later, specify the most generic part of the information model accordingly.

Summary of the root functional requirement

FR = *a knowledge base to*

1. *manage and exchange product-related knowledge and device-independent process-related knowledge (FR-1).*
2. *manage device-specific process-related knowledge (FR-2).*
3. *assert recommendations automatically by matchmaking of product-related knowledge, device-independent process-related knowledge and device-dependent process-related knowledge (FR-3).*

Table 3.2: Summary of the root functional requirement.

Summary of constraints

C_1 = *The independence from any industrial sector has to be ensured.*

C_2 = *Device-independent process-related knowledge of the original production system and the target production system are structured differently.*

C_3 = *None-ambiguity of each element (e.g., product structure, process plan, handling instruction, production resource, process segment) has to be ensured across production systems.*

C_4 = *The required information must be provided in finite time.*

Table 3.3: Summary of constraints to be considered during the design.

Summary of the root design parameter

DP = *an information model which comprises*

1. *product-related information and device-independent process-related information (DP-1).*
2. *device-dependent process-related information (DP-2).*
3. *recommended activities to link (1) and (2) (DP-3).*

Table 3.4: Summary of the root design parameter

Figure 3.1 introduces the idea behind the K-RAMP solution and the role of information models, as they are specified in DP and later in its decomposition hierarchy. Using product-related knowledge from the original production system (*BOM of new product*) covers primarily the requirement of FR to use the product-related knowledge of the original production system. There is also product-related knowledge in the domain of the target production system which covers product knowledge about forerunners (*BOM of forerunners*). As a final situation in Figure 3.1, both product-related information models – the one of the original production system, which is transferred to the target production system, and the initial one of the target production system – result in one comprehensive new product-related information model at the target production system.

However, there is also a device-independent process-related information model (*device-independent process-related information for new product*), which is transferred from the original production system to the target production system.

Which aspects of the process-related information are assumed as device-independent? It is possible to categorize process segments according to basic production techniques. Industrial branches may use the basic production techniques, namely master forming, forming, separating, merging, coating, altering [46, p. 15] [47] or more specialized derivations of them, like browning, cooking, baking, boiling or heating in restaurant kitchens, or mounting windshield, undercoat painting or thixocasting in automotive manufacturing, or chemical vapor deposition, dry etch or lithography in semiconductor manufacturing. These production techniques can be formalized in taxonomy models without consideration of specific equipment.

In order to bridge the gap between the device-specific process-related information of the original production system and the device-specific process-related information of the target production system, such taxonomy models of production techniques are designed through DP.

Another essential portion of the device-independent process-related information covers process capabilities. Process capabilities are used as a contract between products and process segments. In order to create or treat some product or subproduct, an appropriate process capability is required. On the other hand, process segments of a production system are qualified in order to satisfy the specifications of a process capability (see Chapter 2.4). Consequently, process capabilities do not refer anyhow to specific equipment and are therefore device-independent. But they are referenced from both sides – from the product-related information model and from the device-specific process-related information model.

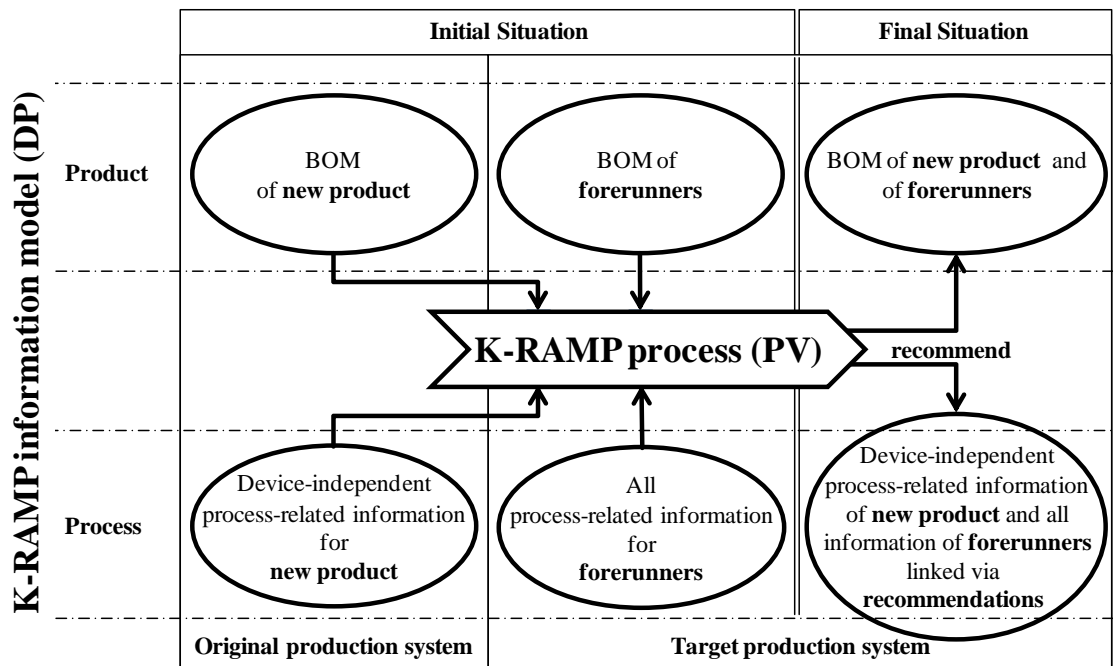


Figure 3.1: Conceptual overview of the K-RAMP approach.

Production techniques and process capabilities which are linked with the new product's decomposition structure (BOM) are also transferred to the target production system. By the use of production techniques and introduced existing process capabilities in conjunction with the new

product's composition structure, *it is possible to derive recommendations for reuse of existing process capabilities*. As a consequence, the existing setup of process segments and process segments themselves which are implementing the process capabilities at the target production system are recommended for reuse.

Therefore, the result of K-RAMP with respect to the process-related information is *device-independent process-related information of new product and all information of forerunners linked via recommendations* (Figure 3.1).

Summarizing the previous sections, the information model, as specified within the root DP satisfies exactly the requirements of the corresponding root FR. On the root level of DP, the specification of this information model is still comprehensive and brief, but it clarifies the major associations between information domains. During the ongoing decomposition process along the axiomatic design approach, this root information model is specified more concrete in iterative steps.

Finally, the root process variable PV is specified to ensure, that the K-RAMP process is performed as a set of sequentially and simultaneously executed procedures. The K-RAMP process accesses the information model of DP bidirectionally (see Figure 3.1). As the overall aim, the K-RAMP process asserts afore mentioned recommendations through reasoning. In order to achieve this aim, a set of subprocedures needs to be specified, which have to be extracted through iterative decomposition in accordance with the axiomatic design approach. A summary of the root level specification of PV is provided in Table 3.5.

Summary of the root process variable
<p>PV = A set of procedures for</p> <ol style="list-style-type: none"> 1. <i>management and exchange of product-related information and device-independent process-related information (PV-1).</i> 2. <i>management of device-specific process-related information (PV-2).</i> 3. <i>management of recommendations for mapping of product-related information and device-independent process-related information with device-dependent process-related information (PV-3).</i>

Table 3.5: Summary of the root process variable.

In order to bring the specified information model of DP to life, PV has to provide two categories of procedures – procedures to manage and procedures to exchange information. The category of procedures named *Management of Information* is based on given information model and comprises the ability to

1. create information items,
2. store information items with a universally unique identity,
3. modify information items,
4. delete information items and
5. query information items.

The category of procedures named *Exchange of Information* is needed in order to bring together the information of two production systems at one place. This category is based on a given information model and comprises the ability to

1. extract information from an information store into any transferable format,
2. optionally to transfer the extracted information to a different location and
3. to import the transferable formatted information to an information store.

Product-related and device-independent process-related information is exchanged between the location of the original production system and the location of the target production system. Without limiting the validity of this specification, a technical realization of PV can also provide means to exchange every domain of information between different locations. There is no need to limit this feature.

3.2.2 First decomposition step

3.2.2.1 Decomposition of the root functional requirement

Throughout the decomposition process the focus will be on the functional domain, the physical domain and the process domain. A systematic decomposition of CAs has been omitted because it has turned out that FRs and CAs usually result in equivalent statements. For example, the customer of the K-RAMP design (ramp-up team) has the need to reproduce links between product-related information and device-independent process-related information. However, this customer's need leads consequently to a functional requirement for linkage between product-related information and device-independent process-related information. Also Suh does not apply decomposition of CAs for the design of software systems. Alternatively, during decomposition of the functional requirements, it is shown that the higher decomposition level is the superset of functional requirements of the next lower decomposition level. In other words, the specification of a functional requirement on the high level must comprise the specifications of all functional requirements on lower levels.

Summary of the first decomposition of FR

FR₁ = Manage and exchange product-related knowledge and device-independent process-related knowledge about the specification of products and its subproducts (product structures) and process capabilities, as well as production techniques. Use this knowledge with biunique identification of knowledge items at several locations.

FR₂ = Manage production resource-related knowledge.

FR₃ = Manage device-specific process-related knowledge which consists of process segments and the setup of process segments, process operations, all forms of required instructions (equipment recipe, operator handling instructions and control software setup).

FR₄ = Manage and automatically assert recommendations with quantified penalties which may be used to link (sub-)products with existing process segments which are already used to produce similar forerunner (sub-)products.

Table 3.6: Summary of the first decomposition of FR.

For the decomposition of FR, it is necessary to ask for the subsets of functional requirements which are needed in order to satisfy the specification of DP properly. This step is called zig-zagging in terms of axiomatic design. The results of the first decomposition of FR are summarized in Table 3.6.

K-RAMP shall be able to manage and to exchange product-related knowledge and device-independent process-related knowledge, which shall be available biunique (constraint C₃) at both

involved production systems (FR_1). By extracting FR_1 from FR , the definition of FR_1 is specified in more detail. Exchange of product-related knowledge and device-independent process-related knowledge is necessary because it is collected at the original production system and needs to be applied at the target production system.

The separation of FR_2 and FR_3 is actually not necessary. However, the reason for this separation is more detailed research on the topic of FR_2 in the future and because detailed research on FR_2 is not relevant for K-RAMP. Due to the binding between design parameters and functional requirements also on the levels of decomposition, this splitting also results in a respective structure of design parameters. Future enhancement of FR_2 and therefore of DP_2 is then possible without interfering with other parts of the K-RAMP design. FR_2 covers a subset of the statement FR_2 and supports a portion of DP_2 .

The function requirement FR_3 specifies the complementary part of FR_2 with respect to the statements FR_2 in more detail. Moreover, by satisfying FR_3 the statement DP_2 of design parameter DP is fully supported. FR_4 requires the assertion of recommended actions to link introduced product categories or products with existing process capabilities and consequently process segments. These recommendations shall also provide a quantified penalty. In Table 3.6 the detailed specification of FR_3 and FR_4 are also summarized.

3.2.2.2 Decomposition of root design parameter

The decomposition of DP has to consider three directions of dependencies. First, in an ideal design each functional requirement on the respective decomposition level should have exactly one assigned design parameter. Consequently, in case of K-RAMP DP should be decomposed to DP_1 to DP_4 , which are associated with FR_1 to FR_4 . Therefore, each information model specified in DP_i has to support the requirements of FR_i . Secondly, the summary of all information models of DP_i shall cover the specification of DP . Thirdly, the summary of all information models of DP_i shall provide the comprehensive information which is needed to realize the specification of PV . For each information model of a DP_i , the following questions need to be answered during its specification:

1. Does the information model of DP_i cover the requirements of FR_i ? If not, what is missing in the information model of DP_i ? Is FR_i specified correctly and completely?
2. Does the union of all information models of DP_i s on one composition level provide the complete specification of the information model on the next higher composition level? If not, is DP_i specified correctly and completely? What is missing in the information model of DP_i ? What is missing in other information models on the same decomposition level?
3. Does the union of all information models on one composition level support the complete specification of the process variable on the next higher composition level? If not, is DP_i specified correctly and completely? What is missing in the information model of DP_i ? What is missing in other information models on the same decomposition level?

A negative answering of 1 may cause an iterative review of the associated functional requirement with cascading impact on specifications on the upper levels in case of changes. A negative answering of 2 or 3 may cause changes in information models on the current composition level and consequently a cascading impact on associated functional requirements and so on. For this reason, each decomposition step is performed thoroughly before the next step is performed. However, answering those questions at every decomposition step ensures that the final design meets to original customer needs respectively the research question (Question 1) in case of this work.

In order to support FR₁, the information model of DP₁ comprises an information model of product-related information and device-independent process-related information. Which concepts are comprised in particular by product-related information respectively by device-independent process-related information? Figure 3.2 provides an overview of the information domains and the most relevant concepts. Product-related information comprises the concept *Product*. Throughout the following discussion, also the concept subproduct is applied, which states that this is a part of the composition structure of a completed *Product* from the production system’s perspective. But it is still a *Product* from the perspective of its own composition level.

However, a *Product* is too specific information for the purpose of K-RAMP. It represents a specific offer of a specific product vendor, like the bar of solid dark chocolate of brand A of a vendor B. However, there are also other brands of solid dark chocolate. And other production systems are probably more experienced with those brands. Even the same brand of solid dark chocolate may be proposed in different sizes and with different weight. Or the same *Product Category* of chocolate cake may be baked as different *Products* with different sizes and shapes. All *Products* have a common set of specifications, and some specific specification makes up the individuality of the *Product*. For this reason, it is more useful to compose the product-related information model on the concept of the *Product Category*. For instance, there is a *Product Category* named “Solid Dark Chocolate”. Similar to classification, a *Product* “Solid Dark Chocolate of Brand A with Weight 250g” is a member of this particular *Product Category*. Based on this categorization it is possible to compose a bill of materials based on *Product Categories*, instead of subproducts. And it is possible to share *Product Categories* across production systems, each with its individual membership of *Products*.

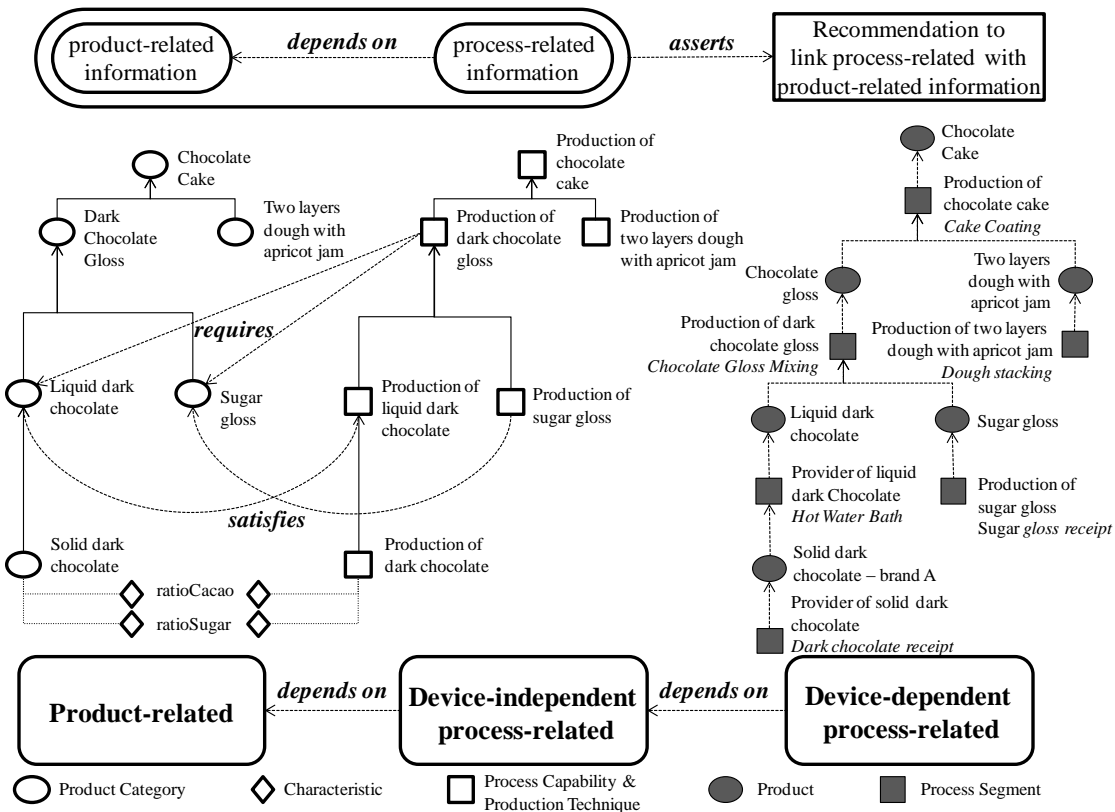


Figure 3.2: Overview of information domains and most relevant concepts.

FR₁ underline that specifications of products, subproducts or process capabilities are required. In the information model of DP₁, *Product Categories* and *Products* are specified by *Characteristics* accordingly. The set of *Characteristics* in fact describes the nature of a *Product Category* and enables distinction between *Product Categories*. The same is also the case for a *Product*. A *Product* is then and only then a member of a *Product Category* if the *Product Category* is specified by a subset of the *Characteristics* of a *Product* and the value range of each *Characteristic* of the *Product* is enclosed by the value range of the *Product Category's* *Characteristic*.

Process Capability is a concept of the information model of DP₁ which is related to device-independent process-related information. *Process Capabilities* are used to link *Process Segments* with *Products* or *Product Categories*. Each *Process Capability* of a process plan is linked to the composition structure of the *Product Category* which should be produced for two purposes. First, in order to produce a *Product Category* a particular *Process Capability* guarantees to satisfy the *Product Category's* specified *Characteristics*. Secondly, this *Process Capability* needs the usage of *Product Categories* on the next lower decomposition level of the product-tree in order to redeem its guarantee. Consequently, a device-independent structure of *Process Capabilities* is formed which is in-line with the respective composition structure of *Product Categories* (Figure 3.2).

Production Technique is a concept of the device-independent process-related information which is used for the categorization of process segments. Recalling Chapter 2.2, process segments are assigned to the device-specific domain of production knowledge. Nevertheless, *Production Techniques* can be maintained completely independent of their later association with process segments. Figure 3.2 is showing the general dependency of device-independent process-related information on product-related information respectively the dependency of device-specific process-related information of device-independent process-related information for this reason. Through the categorization of process segments by *Production Techniques*, it is possible to sketch the potential set of process segments which may enable a given *Process Capability* without using process segments directly. *Production Techniques* are used to determine process segments which are related to each other because they are treating products with equivalent means.

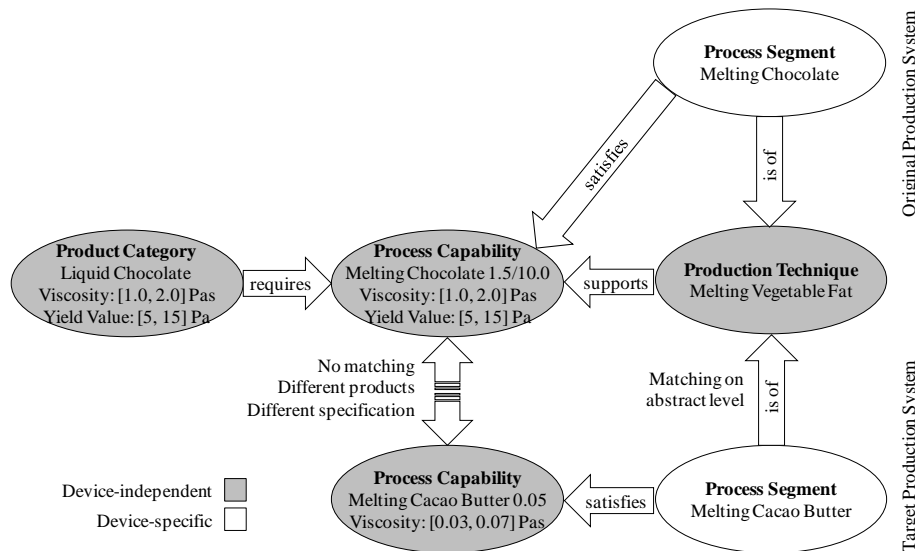


Figure 3.3: Motivation for information domain "Production Techniques".

Figure 3.3 introduces an example in order to explain the generalization aspect which is supported through *Production Techniques*. A *Product Category* “Liquid Chocolate” with particular *Specification* requires an appropriate *Process Capability* “Melting Chocolate 1.5/10.0” for melting chocolate with process average of 1.5 Pas for viscosity and 10.0 Pa for the yield value. There is probably a device-specific process segment “Melting Chocolate” at the original production system which satisfies this *Process Capability*. Therefore, the *Product Category* can be produced at the original production system according to the *Characteristics* of its *Specification*. However, at the target production system, there is a *Process Capability* “Melting Cacao Butter 0.05” which does not match immediately to “Melting Chocolate 1.5/10.0”. At the first glance, there is not opportunity to reuse existing information. However, at the second glance the membership of the process segments “Melting Cacao Butter” and “Melting Chocolate” in the common *Production Technique* “Melting Vegetable Fat” is determined. Therefore, the *Process Capability* “Melting Cacao Butter 0.05” can be indirectly assumed as similar to the *Process Capability* “Melting Chocolate 1.5/10.0” and together with the process segment “Melting Cacao Butter” it can be at least proposed as a candidate for reuse. At this stage of decomposition this statement is still vague. It shall provide one possible opportunity for matchmaking which is discussed in more detailed in the sequel.

For better management of the information model in a specific application (e.g. with or across different branches of manufacturing industries), the information model of DP_1 is split in several models. Figure 3.4 shows the components which comprise the product-related and device-independent process-related domains of information accordingly. The models for *Product Categories*, *Products* and *Process Capabilities* are in the center of interest. The model *Product Categories* comprises the previously described structure of *Product Categories* and the model *Products* the structure of *Products* accordingly. The model of *Process Capabilities* introduces the concept with the same name and associates it with *Products* or *Product Categories*. Both models use the model *Specifications* and the more basic models *Characteristics* and *Engineering Units* in order to establish means for structured description and comparison of their concepts.

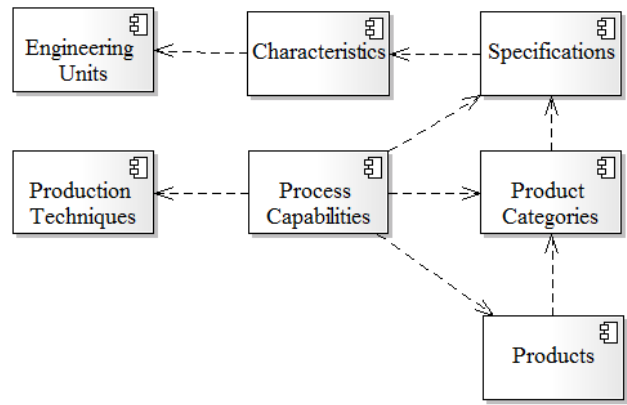


Figure 3.4: Device-independent process-related models of DP_1 and their dependencies.

One example *Process Capability* is “Melting chocolate to viscosity between 1.0 and 2.0 Pas and yield value between 5 and 15 Pa”. Both, *Process Capabilities* and *Product Categories* are using the information domain *Specifications* with different semantics, but finally with the target to map *Product Categories* to *Process Capabilities* or as a final consequence, *Products* to *Process Segments*.

Specifications are used to specify dimensions, shapes, weights and other *Characteristics* within certain ranges of tolerance (specification range). From the perspective of *Product*

Categories, Specifications are used to describe how a *Product Category* is designed or how a particular function shall behave. From the perspective of *Process Capabilities, Specifications* are used to specify what a certain process segment is capable to be used for with sufficient yield. Multiple *Specifications* share the same *Characteristics*, because the same *Characteristic* (e.g., weight) may be used with different tolerances for different *Product Categories* or *Products* or *Process Capabilities*. Finally, elements of the information domain *Engineering Units* are used to specify the common scaling and offset of *Characteristics*.

The model *Production Techniques* is used to model some taxonomy of *Production Techniques* which are used across all involved production systems.

Figure 3.4 and other figures in the sequel only show a minimum set of dependencies between the models of DP₁. As the dependency relationship is transitive, other dependencies can be derived accordingly. For instance, the model *Process Capabilities* depends on *Specifications* and *Specifications* depends on *Engineering Units*, which implies implicitly that *Process Capabilities* also depends on *Engineering Units*.

To summarize the models as specified by DP₁, *Products, Product Categories, Process Capabilities* and *Production Techniques* are exchanged between production systems. Moreover, the bill of materials is made up from *Product Categories* or *Products*. And the *Process Capabilities* are used to compose a set of required capabilities which are expected from process segments of the underlying production system. Each production system may manage its own categorization of *Products* through *Product Categories* and of *Process Segments* through *Production Techniques*. However, it is useful to share common catalogues of *Product Categories* or *Production Techniques* between all involved production systems. Common catalogues of *Product Categories* or *Production Techniques* are nowadays' reality of the majority of manufacturing industries due to standardization of materials and spare-parts.

The model of DP₁ represents the complete specification of the statement DP-1 of the specification of DP (Table 3.4). Separating the whole information model to models, satisfies constraint C₁ as it allows deriving of branch-specific information models with different levels of individuality. For instance, *Engineering Units* may be very generic while *Production Techniques* may be more branch-specific or *Process Capabilities* are even enterprise-specific. In addition, the information model of DP₁ as a whole contributes to PV-1 of PV (Table 3.5) because DP₁ comprises the complete information model which is used by PV due to statement PV-1.

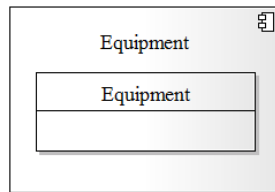


Figure 3.5: Information model of DP₂.

One intention of PV is to reuse “device-specific” *Process Segments*. This intention implicitly introduces the need to link *Process Segments* with “devices” respectively production resources (*Equipment*). DP₂ provides the information model for modeling *Equipment* thus mapping to FR₂ (Figure 3.5). *Equipment*-related information comprises at least some human readable unique identification of the respective *Equipment*. As a device-dependent process-related partition of the overall information model, the specification of DP₂ represents a detailed subset in accordance with statement DP-2 of DP (Table 3.4). Finally, the information model of DP₂ is applied as necessary subset of the information model which is applied by PV due to statement PV-2.

On this first decomposition level, the management of *Equipment* is assigned to a separate design parameter although there is no specific design need for this decision. Information models are split to models within the context of one design parameter wherever it is useful to maintain them separately and independently in real life. With respect to DP₂, a separate design parameter is chosen due to the intention of future enhancement of the respective information model for equipment engineering (see Chapter 6).

According to the simple structure of DP₂, there is only the concept *Equipment* to be considered in conjunction with K-RAMP. Also the structure of this information model with respect to associations is trivial as there are no associations foreseen. For this reason there is no further decomposition required for DP₂ (Figure 3.5) during the next decomposition step.

The device-specific process-related information model of DP₃ is divided into models where a separate maintenance may be useful as well (Figure 3.6). The model *Process Segments* covers comprehensive building blocks which can be combined to process plans. A *Process Segments* is used to comprise a sequence of *Process Operations* in order to satisfy certain *Process Capabilities*. As previously introduced through the example in Figure 3.3, *Process Segments* are classified by *Production Techniques*.

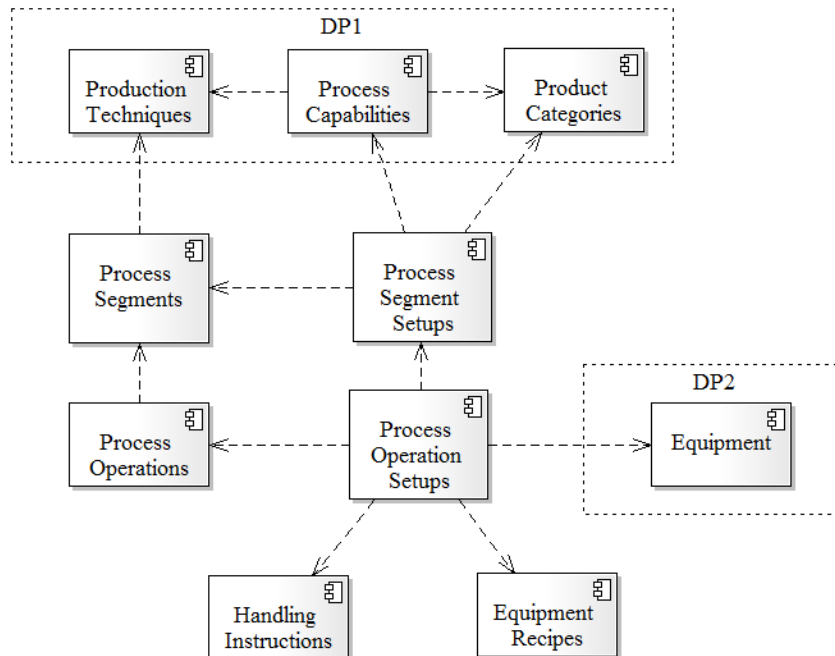


Figure 3.6: Device-specific process-related models of DP₃ and their dependencies.

Process Operations, another model of the whole information model of DP₃, allows modeling of the single activities which make up a *Process Segment*. *Process Operations* require *Process Segments* to which they are assigned and introduce a chronological sequence. Both models, *Process Operations* and *Process Segments*, are device-specific although they are not immediately associated with production resources. However, the structure of *Process Segments* and the existence of *Process Operations* in a production system are strongly related to the applied production resources as already discussed in Chapter 2.2.

The model *Process Segment Setups* enables modeling of the setup of a *Process Segment* which has to be applied across all *Process Operations*. Depending on this setup, different *Process*

Capabilities are enabled through the same *Process Segment*. For instance, a *Process Segment* for warming fat-containing food can be used to satisfy a *Process Capability* for liquidating butter or another *Process Capability* for liquidating chocolate. For this reason, the model *Process Segment Setups* represents the hub to connect *Process Segments* with *Process Capabilities* and also with *Product Categories* which are used as input material of the respective *Process Segment*. In the first case of the example, the input is solid butter and in the second case it is solid chocolate. The applied *Process Segment* remains always the same although the duration of the heating *Process Operation* is different for both *Process Segment Setups*.

Summary of the first decomposition of DP

DP₁ = An information model of product-related information and device-independent process-related information which comprises the following information domains:

1. *Engineering Units (DP₁₋₁)*
2. *Characteristics (DP₁₋₂)*
3. *Specifications and the ability to match them (DP₁₋₃)*
4. *Production Techniques and the ability to generalize them (DP₁₋₄)*
5. *Products and the ability to match and compose them (DP₁₋₅)*
6. *Product Categories and the ability to generalize, match and compose them (DP₁₋₆)*
7. *Process Capabilities and the ability to match them (DP₁₋₇)*

DP₂ = An information model of device-specific process-related information especially for *Equipment (DP₂₋₁)*.

DP₃ = An information model of device-specific process-related information which comprises the following information domains:

1. *Process Segments and their categorization by Production Techniques (DP₃₋₁)*.
2. *Process Segment Setups and their role as setup context of Process Segments to enable particular Process Capabilities if appropriate Product Categories are used as input. (DP₃₋₂)*.
3. *Process Operations and their sequence and membership in Process Segments (DP₃₋₃)*.
4. *Handling Instructions (DP₃₋₄)*
5. *Equipment Recipes (DP₃₋₅)*
6. *Process Operation Setups as setup context of Process Operations, comprising Equipment, Equipment Recipe and Handling Instruction and serving as enhancements of Process Segment Setups (DP₃₋₆)*.

DP₄ = An information model of recommendations which link concepts of DP₁ with concepts of DP₂ and DP₃ (DP₄₋₁).

Table 3.7: Summary of the first decomposition of DP.

The model *Handling Instructions* of DP₃ enables the modeling of instructions for human operators. Complementarily, the model *Equipment Recipe* enables the modeling of recipes which are automatically executed by *Equipment*. For execution of a *Process Operation*, rather likely,

Handling Instructions and *Equipment Recipes* need to be combined for specific *Equipment*. This aspect is considered by the model *Process Operation Setups*.

Summarizing the previous sections, DP₃ manages the majority of device-specific process-related information and particularly it is complementary to DP₂ to cover the statement DP-2 of design parameter DP (Table 3.4). It specifies the information model which is needed to satisfy FR₃ and it contributes, complementary with DP₂, to statement PV-2 of PV (Table 3.5).

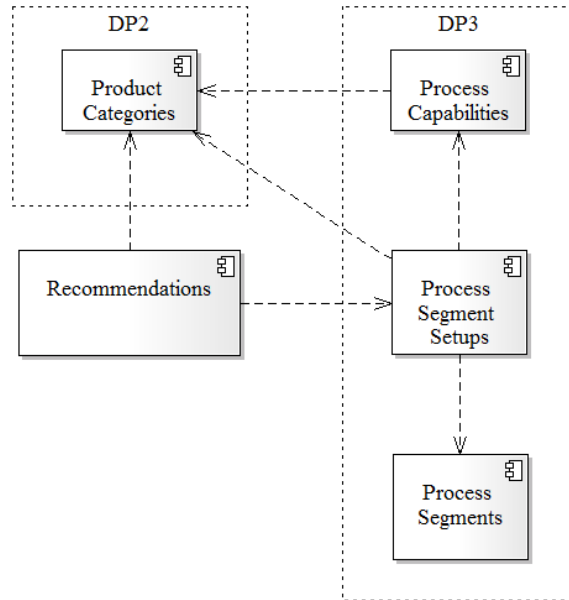


Figure 3.7: Recommendations information model of DP₄ and its dependencies.

Design parameter DP₄ addresses the required recommendations with respect to FR₄. On a very general level *Recommendations* dependent on the one side to concepts of the model *Product Categories* and on the other side to concepts of the model *Process Segment Setups*. For the latter, the model *Recommendations* also depends indirectly on the models *Process Segments* and *Process Capabilities*.

3.2.2.3 Decomposition of root process variable

The first decomposition step of process variables decomposes PV to a set of subordinated PVs which shall be aligned with the DPs on the same decomposition level. In order to ensure an ideal design of the process domain, each PV_i is exactly mapped to a DP_i. Consequently, there should be the same number of PVs and DPs. PV₁, for instance, covers a set of procedures with the intention to manage and to exchange product-related and device-independent process-related information, which is based on the respective information model of DP₁. Generally, a PV_i comprises all procedures which are needed at least in conjunction with the associated DP_i.

In Chapter 3.2.1 the two categories of procedures *Management of Information* and *Exchange of Information* are introduced and their respective abilities are listed. In the sequel of the decomposition process it is necessary to enhance the abilities of *Management of Information*. This category of procedures shall also comprise the ability “to reason new information from existing information based on rules which are provided by the respective information models of the physical domain”. For the specification of PV in Table 3.8 this enhanced specification does already apply.

Summary of the first decomposition of PV

PV_1 = A set of procedures to manage and to exchange product-related and device-independent process-related information which is based on the following information domains:

1. Engineering Units (PV_1-1)
2. Characteristics (PV_1-2)
3. Specifications and reasoning to match them (PV_1-3)
4. Production Techniques and reasoning to determine generalization structures (PV_1-4)
5. Products and reasoning to match them (PV_1-5)
6. Product Categories and reasoning to determine generalization structures as well as to match them (PV_1-6)
7. Process Capabilities and reasoning to match them (PV_1-7)

PV_2 = A set of procedures to manage information about Equipment (PV_2-1).

PV_3 = A set of procedures to manage device-dependent process-related information which is based on the following information domains:

1. Process Segments and their categorization by Production Techniques (PV_3-1).
2. Process Segment Setups and reasoning to link Process Segments to enabled Process Capabilities using appropriate Product Categories as input (PV_3-2).
3. Process Operations and their sequence and membership in Process Segments (PV_3-3).
4. Handling Instructions (PV_3-4)
5. Equipment Recipes (PV_3-5)
6. Process Operation Setups the linkages to Process Operations, comprising Equipment, Equipment Recipe and Handling Instruction and enhanced Process Segment Setups (PV_3-6).

PV_4 = A set of procedures for reasoning of recommendations with quantified penalty and management of recommendations which are linked to the matching concepts (PV_4-1)

Table 3.8: Summary of the first decomposition of PV.

3.2.3 Second decomposition step

3.2.3.1 Decomposition of functional requirement FR_1

Functional requirements, design parameters and process variables are decomposed recursively. The process is again started with functional requirements one after the other. For each functional requirement, the question has to be answered which subordinated FRs are needed in more detail on the next level of decomposition by the associated DP.

FR_1 is therefore decomposed in a way that the specification of DP_1 can be realized. Recalling DP_1 , it specifies an information model which is already split to models DP_1-1 to DP_1-7 . Each specified model of DP_1 causes one separate functional requirement on the second decomposition

level. Decomposition of FR_1 in accordance to these models is resulting in functional requirements $FR_{1,1}$ to $FR_{1,8}$ on the next decomposition level (Table 3.10).

With $FR_{1,1}$, it is requested that *Engineering Units* are managed and exchanged independently from other concepts, which are also summarized as product-related or device-independent process-related information. It makes sense to follow this request, as *Engineering Units* are not only uniform across all involved production systems, but they are uniform across all industries. In fact, it is useful to establish a uniform information model about *Engineering Units*, as it is performed by Hodgson et al. [48]. The potential reuse of this work is discussed later in Chapter 6.

The request of $FR_{1,2}$ is almost similar. It makes sense to use a uniform set of *Characteristics* across all involved production systems. *Characteristics* are associated with one specific *Engineering Unit*. Keeping the information model of *Characteristics* separated, allows managing and exchanging of such entities independently from other product-related or process-related information. Moreover, in the sequel of this work it turns out that *Characteristics* are crucial information items for matching *Product Categories*, *Products* and *Process Capabilities*.

In this context, also the functional requirement $FR_{1,3}$ for management of Specifications is increasingly important. *Specifications* are associated with one specific *Characteristic* and some acceptance criteria. For instance, a *Specification* with *Characteristic* “surface color” is linked with acceptance criterion “red”. However, it could be also more than one acceptance criterion, like the criteria “red” and “dark orange” in order to specify a set of acceptable colors. Such acceptance criteria are targets which much be achieved in order to avoid defective parts. Therefore, this type of *Specification* is called *Specification Target* in the context of K-RAMP.

However, it is even more likely that acceptance criteria are numeric. Quantifying acceptance criteria are the more likely application of *Specifications* in the industry. In the case of quantifying acceptance criteria it is necessary to provide a tolerance band which is delimited by an upper specification limit (USL) and a lower specification limit (LSL) as already described in Chapter 2.4. The width of the tolerance band is usually a compromise between the expected quality of a *Product* and the real capability of the production process to produce the *Product* within this tolerance band. For this purpose *Specification Range* is another concept requested by $FR_{1,3}$, which is associated to one *Characteristic* and up to two limits – the USL and the LSL. Occasionally, only one of the specification limits is applied. *Specification* is the generalization of *Specification Range* and *Specification Target*.

Products, *Product Categories* and *Process Capabilities* are described through *Specifications*. In other words, there is usually more than one *Specification* necessary for a comprehensive description of something. As a consequence, the term “set of *Specifications*” or *Specification Set* is introduced by $FR_{3,1}$. *Specification Sets* collect *Specifications* which utilize a set of disjoint *Characteristics*. *Products*, *Product Categories* and *Process Capabilities* are essentially such *Specification Sets*. Matching and comparing of *Specifications* and *Specification Sets* turn out to be a key for several subsequent requirements and design decisions in order to match information of the original production system and the target production system.

Figure 3.8 shows three *Specification Sets* P_X , P_Y and P_Z . In this example, P_X , P_Y and P_Z represent *Products*, and *Product* is a specialization of *Specification Set* as previously stated. The term *Specification Set* shall be used in the sequel of this example because the described problem is generally applicable. Each *Specification Set* in Figure 3.8 has separate *Specification Ranges* for each *Characteristic* C_1 and C_2 . These *Characteristics* are commonly used by *Specification Ranges* of all involved *Specification Sets* of the example. However, for each *Specification Set* the

Specification Ranges have different LSLs and USLs which are allowed without harming the quality of the respective *Specification Set* (the respective *Product*).

In Figure 3.8, *Specification Ranges* are written in a form where the subscripted letter refers to the *Specification Set* and the superscripted number refers to the applied *Characteristic*. R_Y^1 is the *Specification Range* with *Characteristic* C_1 for *Specification Set* P_Y .

According to Figure 3.8, P_Y is enclosed by P_X because R_X^1 fully encloses R_Y^1 , and R_X^2 fully encloses R_Y^2 . Another *Specification Set* P_Z is not enclosed. The *Specification Ranges* R_X^1 and R_Z^1 are only overlapping each other. This is the most trivial decision-making to determine reusability of *Specification Sets* (whatever the *Specification Sets* specifically are). Having, for instance, a look to P_Y and P_X , the decision is obvious. The majority of measurements in the production process, with respect to characteristics C_1 and C_2 of *Specification Set* P_Y must be within the *Specification Ranges* R_Y^1 and R_Y^2 . Otherwise, according to Chapter 2.4 the underlying process segment is not qualified. There is the assumption that only qualified process segments of the target production system are represented in the information model. So it is rather likely, that if P_X is produced by the same production process the majority of measurements for C_1 and C_2 of *Specification Set* P_X will be within its even wider *Specification Ranges* R_X^1 and R_X^2 as well.

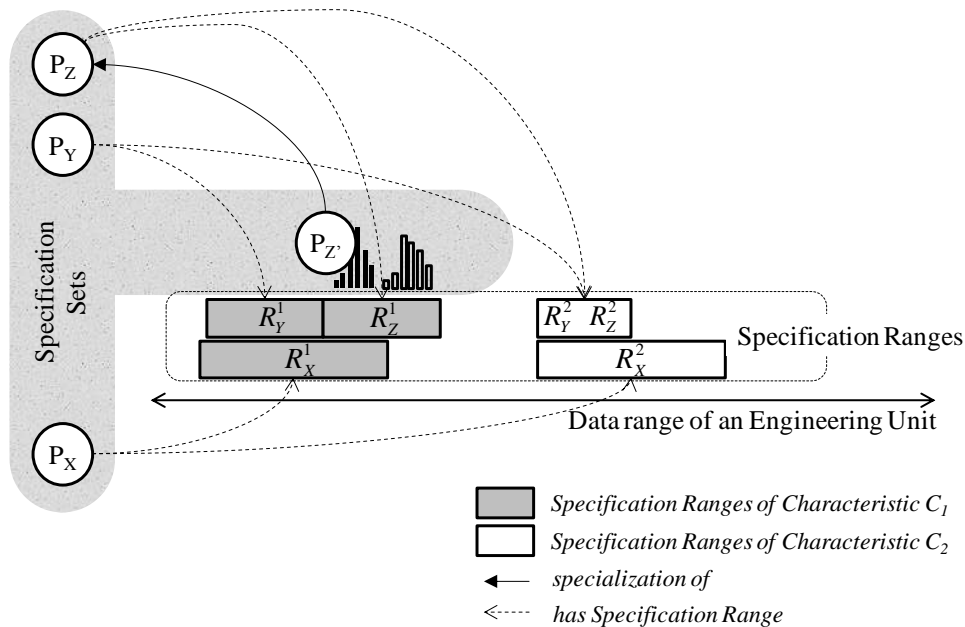


Figure 3.8: Examples of mutually enclosing specification ranges.

However, even *Specification Sets* where the *Specifications* are not fully enclosed by the equivalent *Specifications* of another *Specification Set* can be potential candidates for matching (respectively reuse). This situation is demonstrated by *Specification Set* P_Z in Figure 3.8. Some measurements in the production process, which are related to *Characteristics* of the *Specification Set* P_Z may be within the *Specification Range* R_Z^1 of *Specification Set* P_Z and within the *Specification Range* R_X^1 of the *Specification Set* P_X as well. Therefore, a more special *Specification Set* P_Z' of P_Z can be determined, which is again enveloped by P_X . For this reason R_X^1 and R_Z^1 are overlapping. Due to this reason, it can be also stated that the *Specification Sets* P_X and P_Z are overlapping.

\forall Specifications $s1 \in$ Specification Sets 1 and \forall Specifications $s2 \in$ Specification Sets 2 possible associations between Specifications $s1 R_s s2$			Association between Specification Set 1 and 2 $set1 R_{set} set2$			Cardinality
$R_s=encloses$	$R_s=overlaps$	Different Characteristic	$R_{set}=encloses$	$R_{set}=overlaps$	$R_{set}=partially$	
X			X			$ set1 \leq set2 $
X	X			X		$ set1 \leq set2 $
X		X			X	?
X	X	X			X	?
	X			X		$ set1 \leq set2 $
	X	X			X	?
		X				?

Table 3.9: All possible associations between Specifications of two Specification Sets and the impact on the association between the respective Specification Sets.

In Table 3.9 all possibilities of associations between *Specifications* of two *Specification Sets* and their impact on the association of the respective *Specification Sets* are summarized. For a pair of *Specifications* from two *Specification Sets*, one *Specification* either encloses the other, both *Specifications* overlap each other or the pair is not associated at all. Consequently, the two *Specification Sets* are enclosing each other – if all *Specifications* of the left *Specification Set* enclose the respective *Specifications* of the right hand *Specification Set*. In such a case, the cardinality of the left hand *Specification Sets* is less than or equal to the cardinality of the right hand *Specification Set* because the used set of *Characteristics* of the left hand *Specification Sets* is a subset of the set of *Characteristics* of the right hand *Specification Set*.

The *Specifications* of both *Specification Sets* may overlap each other, resulting in an association “overlaps” between the two *Specification Sets*. This is the case, if there is at least one specification of the right hand specification set only overlapped by its counterpart of the left hand specification set. Some of the *Specifications* of both *Specification Sets* are not related with each other if there is no counterpart using the same *Characteristic*. In this case, the *Specification Sets* can be only partially overlapped – except for the last case where not a single *Specification* of the one *Specification Sets* is associated with a *Specification* of the other *Specification Set*. In this case, there is no association established between the two *Specification Sets*. If two *Specification Sets* are partially overlapped, the association between their cardinalities is not specified because it is not determined how many *Specifications* of the one *Specification Set* and the other *Specification Set* are not associated with each other.

Similar to *Engineering Units* and *Characteristics* it is also useful to provide a separated set of requirements for *Production Techniques*. *Production Techniques* are also uniform across all manufacturing industries to certain extent and can be specialized step by step in accordance with the needs of individual branches. It may be valuable to establish such a uniform set of electronically available *Production Techniques*. There are six basic *Production Techniques* published (see discussion of DP in Chapter 3.2.1), all others of which are derived. Therefore, FR_{1,4} also requires possibilities to build specialization structures of *Production Techniques*.

FR_{1,5} requires the management of *Products*. It is also necessary to determine existing *Products* in the domain of the target production system, which are enveloped by the *Specifications* of a *Product* of the original production system (reusability).

Products are members of *Product Categories*. FR_{1,6} specifies requirements around *Product Category*. *Product Categories* may be specified in a uniform way across all involved production systems and are described by *Specifications*. Moreover, *Product Categories* should be also organized in a generalization structure. For this purpose, it must be considered that a more general *Product Category* fully envelops a more specific *Product Category* based on the logic which is requested for *Specification Sets* by FR_{1,3}. Moreover, a *Product Category* classifies all *Products* which are enveloped by the respective *Product Category*, again with the specified logic of FR_{1,3} because both concepts are specializations of *Specification Set*. Finally, it is also required to compose a structure of *Product Categories* as discussed in conjunction with Figure 3.2.

The management of *Process Capabilities* is required by FR_{1,7}. Moreover, the matching of *Process Capabilities* is required.

Realizing the functional requirements FR_{1,1} to FR_{1,7} fully supports the specification of DP₁ due to the exact matching of the specified requirements with the models as specified through DP₁. Moreover, the total of these functional requirements comprises the full specification of FR₁.

Summary of the decomposition of FR₁

FR_{1,1} = *Manage and exchange knowledge about Engineering Units.*

FR_{1,2} = *Manage and exchange knowledge about Characteristics.*

FR_{1,3} = *Manage and exchange knowledge about Specification Ranges and Specification Targets (generally Specifications), the localization of Specification Ranges to each other and the comparison of Specification Targets in order to associate Specifications as enclosing, overlapping or partially overlapping. Manage and exchange sets of Specifications and associate them due to similarities as enclosing, overlapping or partially overlapping.*

FR_{1,4} = *Manage and exchange knowledge about Production Techniques and their specialization hierarchy.*

FR_{1,5} = *Manage and exchange knowledge about Products, as well as match and compare Products of different sources.*

FR_{1,6} = *Manage and exchange knowledge about Products Categories and their specialization hierarchy. Match and compare Product Categories of different sources. It should be possible to create composition structures of Product Categories. Determine categorization of Products by Product Categories.*

FR_{1,7} = *Manage and exchange knowledge about Process Capabilities. Match and compare Process Capabilities of different sources.*

Table 3.10: Summary of the decomposition of FR₁.

3.2.3.2 Decomposition of functional requirement FR₂

FR₂ shall not be decomposed further, as already clarified during the first decomposition step.

3.2.3.3 Decomposition of functional requirement FR₃

FR_{3,1} requests that *Process Segments* are categorized in *Production Techniques*. It does not request that *Process Segments* are associated directly with input *Product Categories*. This is requested due to FR_{3,2}, which is discussed later. As a consequence, *Process Segments* should not be more than collections of sequences of *Process Operations*. However, also the separation of *Process Segments* to *Process Operations* is requested by FR_{3,3}.

The same *Process Segment*, meaning the same sequence of *Process Operations*, can be used with significantly different results. The changing behavior of *Process Segments* is controlled through *Process Segment Setups* which represent the real enabler for a specific *Process Capability* (FR_{3,2}). A *Process Segment Setup* requires an association with an arbitrary number of *Product Categories*.

The separation of a *Process Segment* to a sequence of *Process Operations* is required through FR_{3,3}. It is useful to manage *Handling Instructions* (FR_{3,4}) and *Equipment Recipes* (FR_{3,5}) separately from the other concepts because entities of both concepts can be used independently and for multiple distinct *Process Operations*. The glue between *Handling Instructions*, *Equipment Recipes*, *Equipment* and the respective *Process Operation* requires a *Process Operation Setup* (FR_{3,6}).

Also the specification of FR₃ is therefore decomposed exactly along the previously described information models of DP₃ (Table 3.11). The functional requirements FR_{3,1} to FR_{3,6} represent the full set of requirements which must be satisfied to realize DP₃. Each decomposed requirement matches exactly to one statement of specification of DP₃. The total of FR_{3,1} to FR_{3,6} is also equivalent to FR₃ on a more detailed level.

Summary of the decomposition of FR₃

FR_{3,1} = *Manage information of Process Segments and their associations to an arbitrary number of Production Techniques.*

FR_{3,2} = *Manage information of Process Segment Setups and their associations to exactly one Process Segment, to exactly one enabled Process Capability and to an arbitrary number of used Product Categories. Determine similarities of Process Capabilities. Match Process Capabilities with Products or Product Categories of different sources by consideration of local Process Segment Setup.*

FR_{3,3} = *Manage information of Process Operations and their associations to maximum one predecessor and to maximum one successors and as membership to exactly one Process Segment.*

FR_{3,4} = *Manage information of Handling Instructions.*

FR_{3,5} = *Manage information of Equipment Recipes.*

FR_{3,6} = *Manage information of Process Operation Setups and their associations to maximum one Handling Instruction, to maximum one Equipment Recipe, to an arbitrary number of Equipment, as well as to exactly one Process Operation.*

Table 3.11: Summary of the decomposition of FR₃.

3.2.3.4 Decomposition of functional requirement FR₄

The functional requirement FR₄ is decomposed to FR_{4,1}. The composition is performed anyway. The reason for this decomposition step is that DP_{4,1} and PV_{4,1} need to be associated with the decomposition of DP₁ and DP₃ in the sequel of the design (Table 3.12).

Summary of the decomposition of FR₄

FR_{4,1} = *Manage and automatically assert recommendations with quantified severities which may be used to associate (sub-)products with existing process segments which are already used to produce similar forerunner (sub-)products.*

Table 3.12: Summary of the decomposition of FR₄.

3.2.3.5 Decomposition of design parameter DP₁

The next design step is the decomposition of design parameters DP_i to multiple DP_{i,j} which are capable to satisfy previously introduced functional requirements F_{i,j}. Additionally, the information models of these design parameters must cover the superior DP_i and provide more detailed information models in order to perform PV_i. Therefore, the same principles apply as already described in the introduction of Chapter 3.2.2.2. This chapter covers the decomposition of DP₁ while the subsequent chapters address the decomposition of DP₃ and DP₄ (DP₂ is omitted with respect to further decomposition).

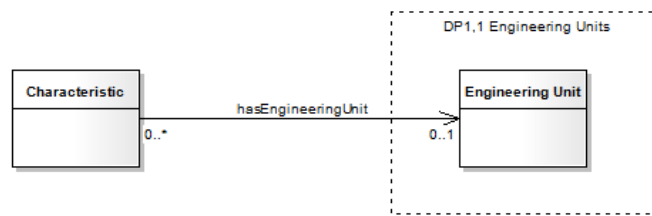


Figure 3.9: Information models of DP_{1,1} and DP_{1,2} and the associations between concepts.

The information model of DP_{1,1} is very simple and comprises only the concept *Engineering Unit*. It is useful, that information which is based on this information model is maintained completely independent of other information domains of K-RAMP because it represents a common foundation to specify quantifying information and data across distributed production systems. This portion of information can be even maintained independently of enterprises across supply chains or across all manufacturing industries as this is, for instance, realized with the QUDT ontology [48]. In order to focus on the immediate objectives of K-RAMP, it is omitted to involve QUDT. However, it might be a useful separate work to consider this idea.

The information model of DP_{1,2} is very simple as well as it only comprises the concept *Characteristic* and its association to *Engineering Unit* of DP_{1,1} (Figure 3.9). Within enterprises or supply chains it is useful to maintain this information model as a common taxonomy in order to apply common *Characteristics* across *Products*, *Product Categories* or *Process Capabilities* of all production systems. In such situations, it is also essential to determine whether two given *Characteristics* are comparable. As a simple logic, two *Characteristics* are comparable if they refer to the same *Engineering Unit*. However, this may still lead to the comparison of *Characteristics* with different semantics. For this reason, *Characteristics* are only comparable with themselves and it is subject to the information modeling across production systems to establish a unified set of *Characteristics* with sufficient in depth semantics of its members.

The information model of DP_{1,3} introduces an abstract concept *Specification* which is always associated with a *Characteristic* of DP_{1,2} (Figure 3.10). In order to distinct between qualifying and quantifying information, there are concepts derived from *Specification* accordingly (*Specification Range*, *Specification Target*). *Specification Target* refers to a set of qualifying information items. Such information items are non numeric or even numeric but with a small set of representatives. Every concept which shall provide target values must be a specialization of *Target Value*. For instance, every *Specification Set* shall be usable as *Target Value*. *Specification Range* represents quantifying information items, which are always numeric. There is an upper specification limit (USL) and a lower specification limit (LSL) possible (see discussion in context with Figure 3.8). At least one of them must be referenced by each *Specification Range*.

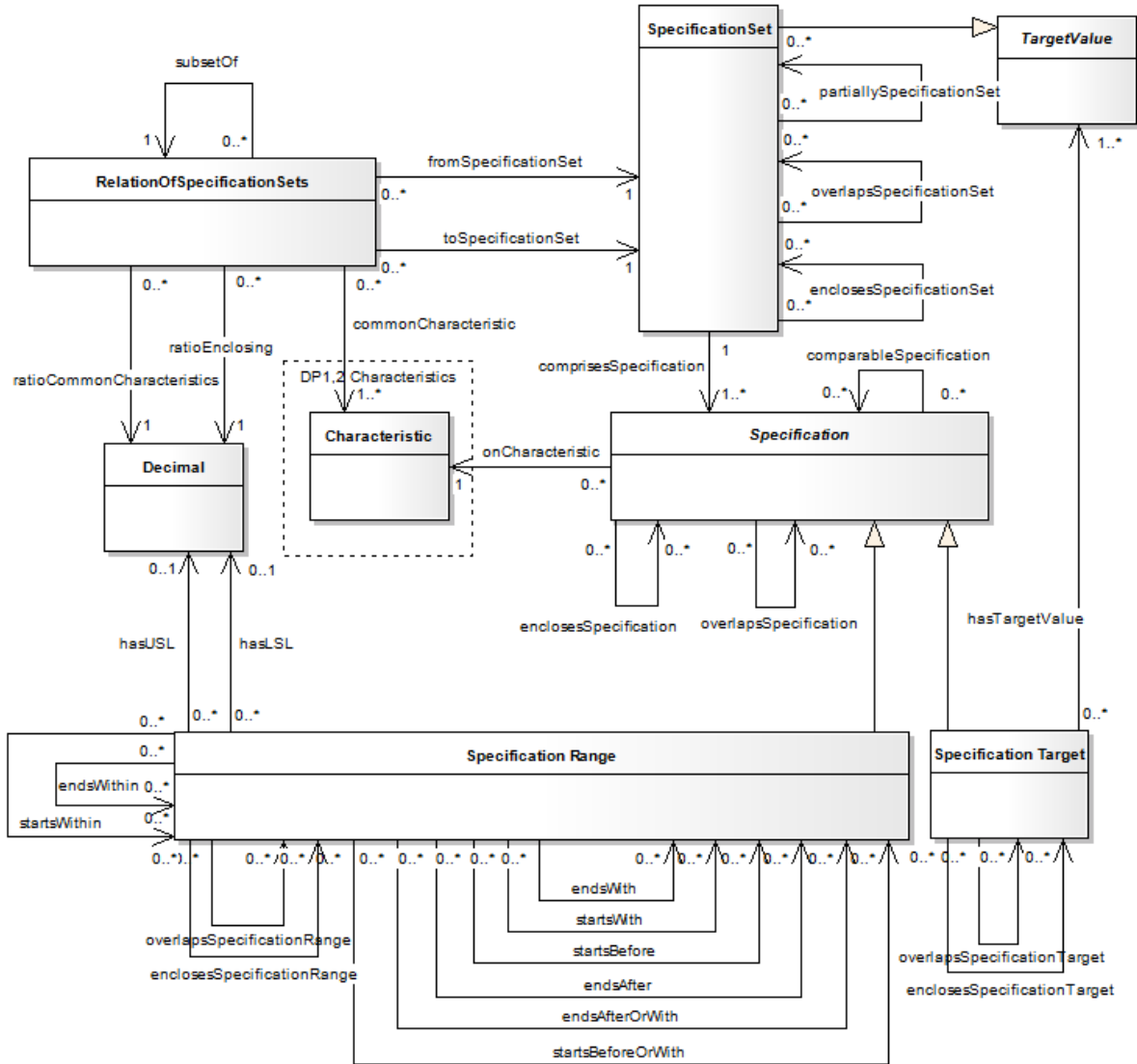


Figure 3.10: Information model of DP_{1,3} and its associations to concepts of DP_{1,2}.

As one example, the *Product Vendor-A-Chocolate* is specified by a *Characteristic* “ratioCacao” with a range between 45 and 60 percent, a *Characteristic* “ratioSugar” within a range of 10 to 12 percent (both are *Specification Ranges*) and a *Characteristic* “taste” which can be qualified as “sweet” and “bitter” (a *Specification Target*).

In the context of FR_{1,3}, matching and comparing of the model *Specification Ranges* is addressed. In order to manage the comparison of *Specification Ranges*, several associations between *Specification Ranges* are introduced for this purpose. A reduced set of such associations is also introduced for *Specification Targets*. The associations between *Specifications* respectively between *Specification Sets* are related to the summary in Table 3.9. The concept *Relation Of Specification Sets* provides additional insight to these associations in case of the relations “encloses”, “overlaps” or “partially” between *Specification Sets*. This insight comprises

1. A ratio of *Specifications* which are enclosing *Specifications* of the other *Specification Set* compared to all relationships between *Specifications* of both *Specification Sets*.

This shall be discussed by example. For instance, for two *Specification Sets* *set1* and *set2* there is the relationship *overlaps (set1, set2) – set1 overlaps set2*. According to Table 3.9, overlaps may also imply a mixture of encloses and overlaps associations between *Specifications*. In the example

set1 may have 3 *Specifications* which enclose *Specifications* of *set2*,

set2 has 1 *Specification* which encloses a *Specification* of *set1*. From the perspective of *set1* this *Specification* overlaps in the other direction.

2 further *Specifications* are overlapping from the perspective of both *Specification Sets*.

1 more *Specification* in *set2* is not associated with any *Specification* in *set1* because no common *Characteristic* is shared.

All in all, this results in $|set1|=6$ and $|set2|=7$.

For *set1 overlaps set2* the *Overlapping Of Specification Set* holds a ratio of enclosing *Specifications* of 0.5 (3 enclosing of 6 in total) while in the opposite direction there is a ratio of 0.143 (1 enclosing of 7 in total) for *set2 partially set1* because from the perspective of *set2* there is one *Specification* which is not associated with any *Specification* of *set1*.

2. A ratio of *Characteristics* which are shared between *Specifications* of both *Specification Sets* compared to the cardinality of the respective *Specification Sets*.

With respect to the previous example, this ratio of common *Characteristics* is 1.0 from the perspective of *set1* because $|set1|=6$ and there are 3 encloses associations and 3 disjoint overlaps associations encountered. From the perspective of *set2*, the ratio is 0.43 because $|set2|=7$ and there is 1 encloses and 2 overlaps association.

3. A list of the shared *Characteristics*.

Relation Of Specification Sets comprises a set of *Characteristics* which are shared by *Specifications* of both overlapping *Specification Sets*. A relationship *subset Of* is introduced between pairs of occurrences of *Relation Of Specification Sets*. One *Relation Of Specification Sets* o_1 is a *subset Of* another *Relation Of Specification Sets* o_2 if $set_1.commonCharacteristics \subseteq set_2.commonCharacteristics$.

The information model of DP_{1,4} introduces the concept *Production Technique* (Figure 3.11). Figure 3.3 discusses the motivation of *Production Techniques* in order to reuse *Process Segments*. For this purpose, the concept *Production Technique* represents a categorization of *Process Segments*. It is still device-independent and is targeting information reuse on a meta-level. The most general *Production Techniques* according to [46, p. 15] or [47] are “Master Forming”,

“Forming”, “Separating”, “Merging”, “Coating” and “Altering”. Within the domain of an enterprise, the existence of a common taxonomy of *Production Techniques* is assumed. For instance, the bakery shop may use a *Production Technique* “Cutting Slices of Dough” which is indirectly a specialization of the basic *Production Technique* “Separating”. Consequently, there is an association “specializes” between *Production Techniques*, which allows the definition of taxonomies of *Production Techniques* including arbitrary levels of specialization.

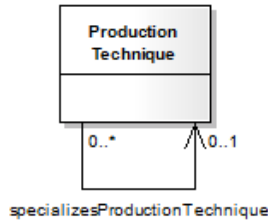


Figure 3.11: Information model of DP_{1,4}.

The information models DP_{1,5} and DP_{1,6} (Figure 3.12) introduce the concepts and associations in order to satisfy FR_{1,5} and FR_{1,6} respectively. The primary focus is on associations in order to represent decomposition structures of both concepts. With respect to *Product Category* also the representation of specialization structures is introduced, while for the concept *Product* the membership in *Product Categories* (is a) is specified.

A *Product Category* is a specialization of *Specification Set* which is specified by DP_{1,3}. Therefore, it is possible to specify a *Product Category* “enrobing chocolate” which has beside others a viscosity between 0.5 and 2.5 Pas and a yield value between 0 and 20 Pa [49]. Every *Product* which is a member of the *Product Category* “enrobing chocolate” must be enclosed by this *Product Category*. As a consequence, a *Product* must be specified at least through the same set of *Characteristics* as the *Product Category* where it belongs to. Therefore, a *Product* “enrobing chocolate” of each vendor or brand must also be specified at least through viscosity and yield value.

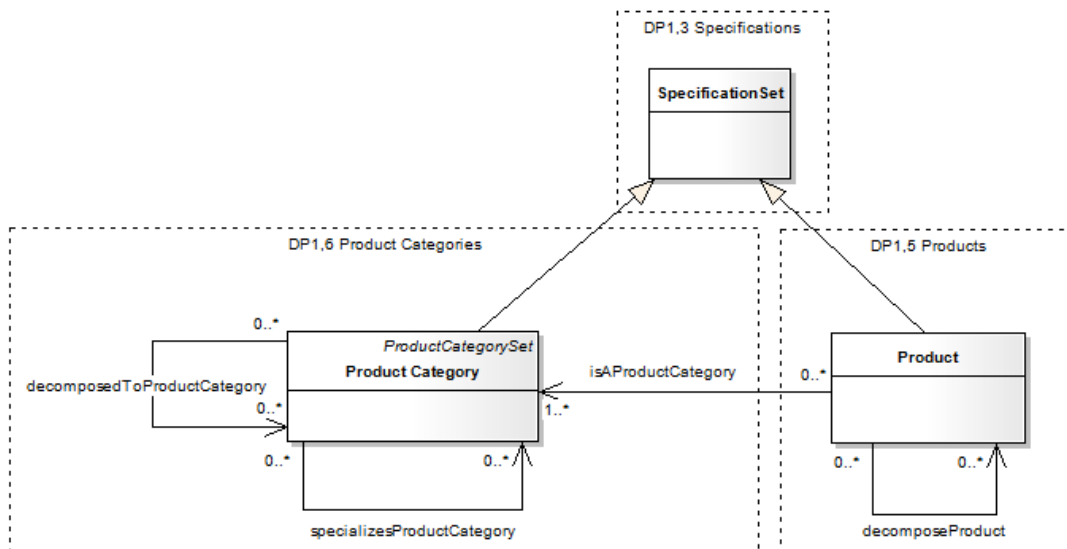


Figure 3.12: Information models of DP_{1,5} and DP_{1,6} and their associations to concepts of DP_{1,3}.

A *Product Category* may be a specialization of another *Product Category*. For instance, a *Product Category* “Chocolate sponge plate” is a specialization of the *Product Category* “Sponge plate”. Specializations of *Product Categories* can be implied through the associations between *Specification Sets*. A *Product Category* which encloses another *Product Category* is a generalization of the enclosed *Product Category*.

The information model of DP_{1,7} contains the concept *Process Capability* and associations with *Product* and *Product Category*, as well as an association with *Production Techniques* of the DP_{1,4} information model (Figure 3.13). *Process Capability* represents the glue between *Products* and *Product Categories* on the one side and *Process Segments* on the other. *Products* and *Product Categories* require *Process Capabilities*, and *Process Segments* satisfy *Process Capabilities* (Figure 3.3). There is also a specialization hierarchy of *Process Capabilities*. One *Process Capability* is a specialization of another *Process Capability* if it is enclosed by the more generic one. The association “encloses” is available for *Process Capabilities* because this concept is derived from *Specification Set*.

A *Product Category* is decomposed by an arbitrary number of other *Product Categories*. It requires a particular *Process Capability*, as introduced by DP_{1,7}, to compose a *Product Category* from these *Product Categories* on a lower level of decomposition. A *Process Capability* composes a *Product Category* from those lower level *Product Categories*. For instance, a *Product Category* “Two-layer chocolate sponge cake with apricot jam, in-between” is decomposed to a *Product Category* “Chocolate sponge plate coated with apricot jam” and a *Product Category* “Chocolate sponge plate”. A *Process Capability* “Stacking two plates of dough” is required to perform the composition. However, this last aspect cannot be covered by the decomposition of DP₁ because it requires the invocation of concepts DP₃.

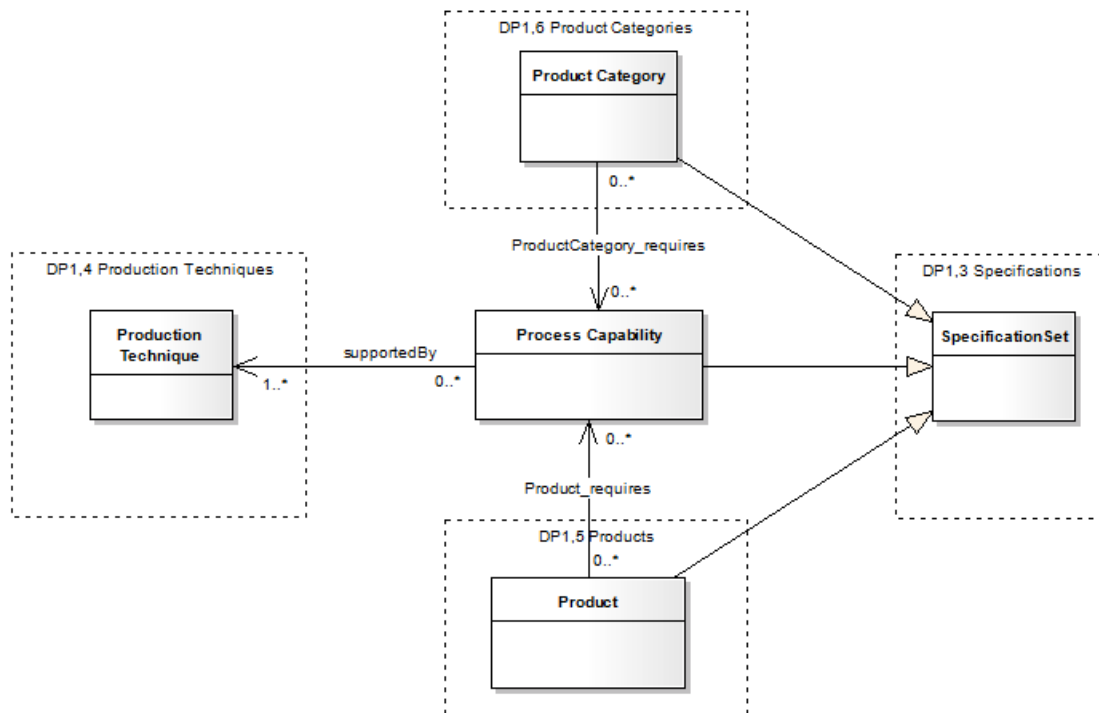


Figure 3.13: Information model of DP_{1,7} and its associations to concepts of DP_{1,3}, DP_{1,4}, DP_{1,5} and DP_{1,6}.

In addition, *Process Capability* is supported by a *Production Technique*. Moreover, one *Process Capability* can be “similar” to another *Process Capability*. This topic was discussed in conjunction with Figure 3.3 and is introduced later in the context of the information model of DP_{3,1} because this aspect also requires the invocation of the device-specific process-related information model.

Products and *Product Categories* require *Process Capabilities*. These associations are asserted with implication rules as specified for DP_{1,7} in Table 3.13. Recalling DP_{1,5}, DP_{1,6} and DP_{1,7} in conjunction with their superior concept of DP_{1,3}, the following conclusions can be taken (see Figure 3.14):

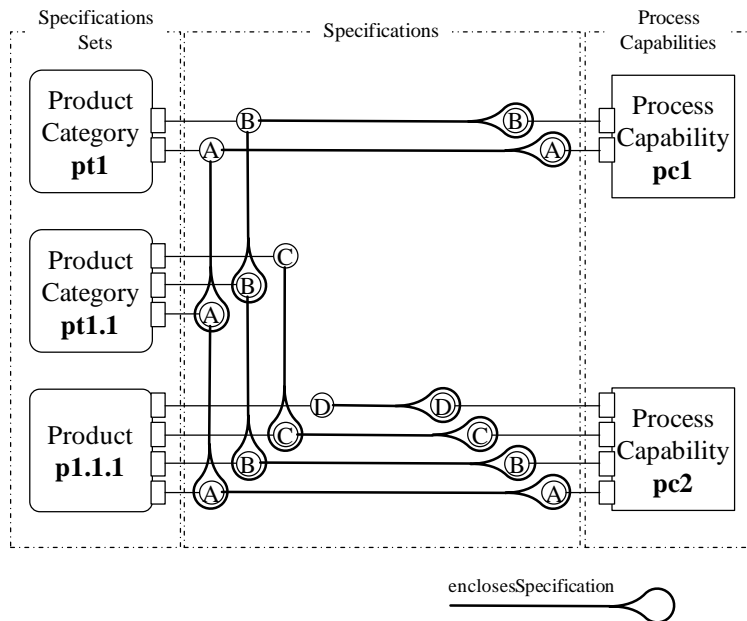


Figure 3.14: Overview of encloses-associations between Product Categories, Products and Process Capabilities.

Products require *Process Capabilities* in order to produce them. These *Process Capabilities* must be enclosed by *Products* in the sense of *SpecificationSets* (e.g., p1.1.1 encloses pc2). This is probably not always the case because there are also *Process Capabilities* which are specified for producing a category of *Products* in general (e.g., p1.1.1 does not enclose pc1). One group of examples comprises capabilities of cooling processes or heating processes where probably the volume and density of the material are the only relevant constraints (e.g., pc1 comprises only *Specifications* A and B).

However, each *Product* has its individually specified shape (e.g., p1.1.1 comprises *Specifications* A, B, C and D). Therefore, the *Product*'s set of specifications is a superset of the *Specifications* of the *Process Capability* (e.g., p1.1.1 \supseteq pc1). According to the *enclosesSpecificationSet*-association of DP_{1,3} consequently the *Product* does not enclose the *Process Capability*, and as further consequence due to DP_{1,7} the *Product* would not require the *Process Capability*. But this decision is only half the story because this particular *Process Capability* is dedicated to a *Product Category* as mentioned before (e.g., pt1 encloses pc1).

How is it possible to determine, whether a categorized *Product* requires this *Process Capability*? The solution is as follows. The *Product* is categorized by a *Product Category* which

may be a specialization of another *Product Category* (e.g., pt1 encloses pt1.1, p isA pt1.1, and due to the implications rules of DP_{1,6} and DP_{1,7} also p isA pt1). The *Product* p1.1.1 does not necessarily enclose pc1. However, both *Specification Sets* are for sure overlapping, and therefore, *Process Capabilities* of *Product Categories* can be possible candidates for their categorized *Products*. Based on this reasoning path also requires-associations between *Products* and *Product Categories* can be determined indirectly.

The specification after decomposition of DP₁ already shows mechanisms in order to reuse existing information. However, the discussed implications are trivial and focused on the specialization of *Product Categories* and *Process Capabilities* due to enclosed *Specification Sets* only. The assumption of this scenario is that a new *Product Category* or a new *Process Capability* has larger quality tolerances (e.g., wider *Specification Ranges*) than existing ones. However, it is also necessary and even more important to address the opposite scenario, as it is more likely in real world.

Summary of the decomposition of DP₁

DP_{1,1} = *An information model of Engineering Units.*

DP_{1,2} = *An information model of Characteristics.*

DP_{1,3} = *An information model of Specifications. Assuming Specifications s₁ and s₂, Characteristics c₁ and c₂, Specification Targets t₁ and t₂, Specification Sets set₁ and set₂, Specification Ranges r₁ and r₂, as well as Relation Of Specification Sets ross.*

*fromSpecificationSet (ross, set₁) ∧ toSpecificationSet (ross, set₂) ∧
DifferentFrom (set₁, set₂) ∧ ratioCommonCharacteristics (ross, nc) ∧ nc=100 ∧
ratioEnclosing (ross, ne) ∧ ne=100 → enclosesSpecificationSet (set₁, set₂)*

*∃ set₁, set₂ ∀ x ∃ y, c. (comprisesSpecification (set₁, x) ∧ comprisesSpecification (set₂, y) ∧
enclosesSpecification (x, y) ∧ onCharacteristic (x, c)) →*

∃ ross. RelationOfSpecificationSets (ross) ∧

*fromSpecificationSet (ross, set₁) ∧ toSpecificationSet (ross, set₂) ∧
ratioEnclosing (ross, 100) ∧ ratioCommonCharacteristics (ross, 100) ∧
commonCharacteristic (ross, c)*

∀ x, y. enclosesSpecificationTarget (x, y) → enclosesSpecification (x, y)

∀ x, y. enclosesSpecificationRange (x, y) → enclosesSpecification (x, y)

comparableSpecification (t₁, t₂) ∧

*∀ x ∃ y. hasTarget (t₁, x) ∧ hasTarget (t₂, y) ∧ (SameAs (x, y) ∨ generalizes (x, y)) →
enclosesSpecificationTarget (t₁, t₂)*

*comparableSpecification (r₁, r₂) ∧ startsBeforeOrWith (r₁, r₂) ∧ endsAfterOrWith (r₁, r₂) →
enclosesSpecificationRange (r₁, r₂)*

∀ x, y. enclosesSpecificationSet (x, y) → encloses (x, y)

∀ x, y. enclosesSpecification (x, y) → encloses (x, y)

*∀ c₁ ∃ c₂. commonCharacteristic (ross₁, c₁) ∧ commonCharacteristic (ross₂, c₂) ∧
SameAs (c₁, c₂) → subsetOf (ross₁, ross₂)*

*fromSpecificationSet (ross, set₁) ∧ toSpecificationSet (ross, set₂) ∧
ratioCommonCharacteristics (ross, n) ∧ n=100 ∧ ratioEnclosing (ross, ne) ∧ ne<100 →*

Summary of the decomposition of DP₁

overlapsSpecificationSet (*set*₁, *set*₂)

$\exists set_1, set_2 \exists x \exists y, c. comprisesSpecification (set_1, x) \wedge comprisesSpecification (set_2, y) \wedge overlapsSpecification (x, y) \wedge onCharacteristic (x, c) \wedge$

$(\forall x' \exists y', c'. x' \neq x \wedge y' \neq y \wedge enclosesSpecification (x', y') \wedge onCharacteristic (x', c')) \rightarrow$

ross. RelationOfSpecificationSets (*ross*) \wedge

fromSpecificationSet (*ross*, *set*₁) \wedge *toSpecificationSet* (*ross*, *set*₂) \wedge

commonCharacteristic (*ross*, *c*) \wedge *commonCharacteristic* (*ross*, *c'*) \wedge

ratioEnclosing (*ross*, $\#x' / (\#x + \#x') * 100$) \wedge *ratioCommonCharacteristics* (*ross*, 100)

...# is the count of respective instances

$\forall x, y. overlapsSpecificationTarget (x, y) \rightarrow overlapsSpecification (x, y)$

$\forall x, y. overlapsSpecificationRange (x, y) \rightarrow overlapsSpecification (x, y)$

comparableSpecification (*t*₁, *t*₂) \wedge $\neg enclosesSpecificationTarget (t_1, t_2) \wedge$

$\exists x \exists y. hasTarget (t_1, x) \wedge hasTarget (t_2, y) \wedge$

$(SameAs (x, y) \vee generalizes (x, y) \vee specializes (x, y) \vee$

$(generalizes (c, x) \wedge generalizes (c, y))) \rightarrow overlapsSpecificationTarget (t_1, t_2)$

comparableSpecification (*r*₁, *r*₂) $\wedge startsBefore (r_1, r_2) \wedge endsWithin (r_1, r_2) \rightarrow$

overlapsSpecificationRange (*r*₁, *r*₂)

comparableSpecification (*r*₁, *r*₂) $\wedge startsWithin (r_1, r_2) \wedge endsAfter (r_1, r_2) \rightarrow$

overlapsSpecificationRange (*r*₁, *r*₂)

onCharacteristic (*s*₁, *c*₁) \wedge *onCharacteristic* (*s*₂, *c*₂) $\wedge SameAs (c_1, c_2) \rightarrow$

comparableSpecification (*s*₁, *s*₂)

$\forall x, y. overlapsSpecificationSet (x, y) \rightarrow overlaps (x, y)$

onSpecificationSet (*ross*, *set*₁) \wedge *onSpecificationSet* (*ross*, *set*₂) $\wedge DifferentFrom (set_1, set_2) \wedge$

ratioCommonCharacteristics (*ross*, *n*) $\wedge n < 100 \rightarrow partiallySpecificationSet (set_1, set_2)$

$\exists set_1, set_2 \forall x, x', x'', x''' \exists y, c. comprisesSpecification (set_1, x) \wedge$

comprisesSpecification (*set*₂, *y*) \wedge *comprisesSpecification* (*set*₂, *x'*) \wedge

comprisesSpecification (*set*₂, *x''*) \wedge *comprisesSpecification* (*set*₂, *x'''*) \wedge

$((enclosesSpecification (x, y) \vee overlapsSpecification (x', y) \vee$

comparableSpecification (*x''*, *y*)) \wedge *onCharacteristic* (*y*, *c*) \vee

$\neg comparableSpecification (x''', y) \rightarrow$

ross. RelationOfSpecificationSets (*ross*) \wedge

fromSpecificationSet (*ross*, *set*₁) \wedge *toSpecificationSet* (*ross*, *set*₂) \wedge

commonCharacteristic (*ross*, *c*) \wedge

ratioEnclosing (*ross*, $\#x / (\#x + \#x' + \#x'' + \#x''') * 100$) \wedge

ratioCommonCharacteristics (*ross*, $(\#x + \#x' + \#x'') / (\#x + \#x' + \#x'' + \#x''') * 100$)

...# is the count of respective instances

$\forall x, y. partiallySpecificationSet (x, y) \rightarrow partially (x, y)$

$\forall r_1, r_2. startsBefore (r_1, r_2) \rightarrow startsBeforeOrWith (r_1, r_2)$

Summary of the decomposition of DP₁

$\forall r_1, r_2. \text{startsWith}(r_1, r_2) \rightarrow \text{startsBeforeOrWith}(r_1, r_2)$

$\forall r_1, r_2. \text{endsAfter}(r_1, r_2) \rightarrow \text{endsAfterOrWith}(r_1, r_2)$

$\forall r_1, r_2. \text{endsWith}(r_1, r_2) \rightarrow \text{endsAfterOrWith}(r_1, r_2)$

$\text{comparableSpecification}(r_1, r_2) \wedge \text{hasLSL}(r_1, \text{lsl}_1) \wedge \text{hasLSL}(r_2, \text{lsl}_2) \wedge \text{lsl}_1 < \text{lsl}_2 \rightarrow \text{startsBefore}(r_1, r_2)$

$\text{comparableSpecification}(r_1, r_2) \wedge \text{hasLSL}(r_1, \text{lsl}_1) \wedge \text{hasLSL}(r_2, \text{lsl}_2) \wedge \text{lsl}_1 = \text{lsl}_2 \rightarrow \text{startsWith}(r_1, r_2)$

$\text{comparableSpecification}(r_1, r_2) \wedge \text{hasUSL}(r_1, \text{usl}_1) \wedge \text{hasUSL}(r_2, \text{usl}_2) \wedge \text{usl}_1 > \text{usl}_2 \rightarrow \text{endsAfter}(r_1, r_2)$

$\text{comparableSpecification}(r_1, r_2) \wedge \text{hasUSL}(r_1, \text{usl}_1) \wedge \text{hasUSL}(r_2, \text{usl}_2) \wedge \text{usl}_1 = \text{usl}_2 \rightarrow \text{endsWith}(r_1, r_2)$

$\text{comparableSpecification}(r_1, r_2) \wedge \text{hasLSL}(r_1, \text{lsl}_1) \wedge \text{hasLSL}(r_2, \text{lsl}_2) \wedge \text{hasUSL}(r_2, \text{usl}_2) \wedge \text{lsl}_1 > \text{lsl}_2 \wedge \text{lsl}_1 < \text{usl}_2 \rightarrow \text{startsWithin}(r_1, r_2)$

$\text{comparableSpecification}(r_1, r_2) \wedge \text{hasUSL}(r_1, \text{usl}_1) \wedge \text{hasUSL}(r_2, \text{usl}_2) \wedge \text{hasLSL}(r_2, \text{lsl}_2) \wedge \text{usl}_1 < \text{usl}_2 \wedge \text{usl}_1 > \text{lsl}_2 \rightarrow \text{endsWithin}(r_1, r_2)$

DP_{1,4} = An information model of Production Techniques. Assuming Production Techniques pq_1 and pq_2

$\forall pq_1, pq_2. \text{specializesProductionTechnique}(pq_1, pq_2) \rightarrow \text{specializes}(pq_1, pq_2)$

$\text{specializes}(x, y) \wedge \text{specializes}(y, z) \rightarrow \text{specializes}(x, z)$

$\text{specializes}(y, x) \rightarrow \text{generalizes}(x, y)$

DP_{1,5} = An information model of Products.

DP_{1,6} = An information model of Product Categories. Assuming Product Categories pt , pt_1 and pt_2 and Products p .

$\text{encloses}(pt, p) \rightarrow \text{isAProductCategory}(p, pt)$

$\forall x, y. \text{isAProductCategory}(x, y) \rightarrow \text{isA}(x, y)$

$\text{encloses}(pt_1, pt_2) \rightarrow \text{specializesProductCategory}(pt_2, pt_1)$

$\text{specializesProductCategory}(x, y) \rightarrow \text{specializes}(x, y)$

$\text{isA}(x, y) \wedge \text{specializes}(y, z) \rightarrow \text{isA}(x, z)$

see DP_{1,4} concerning definition of $\text{specializes}(x, y)$

DP_{1,7} = An information model of Process Capabilities their attributes and associations (to Production Technique, Specification, Product Category and Product). Assuming Product Category pt , Products p and Process Capability pc .

$\text{encloses}(p, pc) \rightarrow \text{Product_requires}(p, pc)$

$\text{encloses}(pt, pc) \rightarrow \text{ProductCategory_requires}(pt, pc)$

Summary of the decomposition of DP₁

$$ProductCategory_requires (pt, pc) \wedge isAProductCategory (p, pt) \rightarrow Product_requires (p, pc)$$

$$\forall p, pc. Product_requires (p, pc) \rightarrow requires (p, pc)$$

$$\forall pt, pc. ProductCategory (pt, pc) \rightarrow requires (pt, pc)$$

Table 3.13: Summary of the decomposition of DP₁.

3.2.3.6 Decomposition of design parameter DP₃

There are design parameters DP_{3,1} to DP_{3,6} as a decomposition of DP₃ in accordance with the functional requirements FR_{3,1} to FR_{3,6}. Each design parameter satisfies the associated functional requirements. Those design parameters also provide the comprehensive information model which is needed for PV₃.

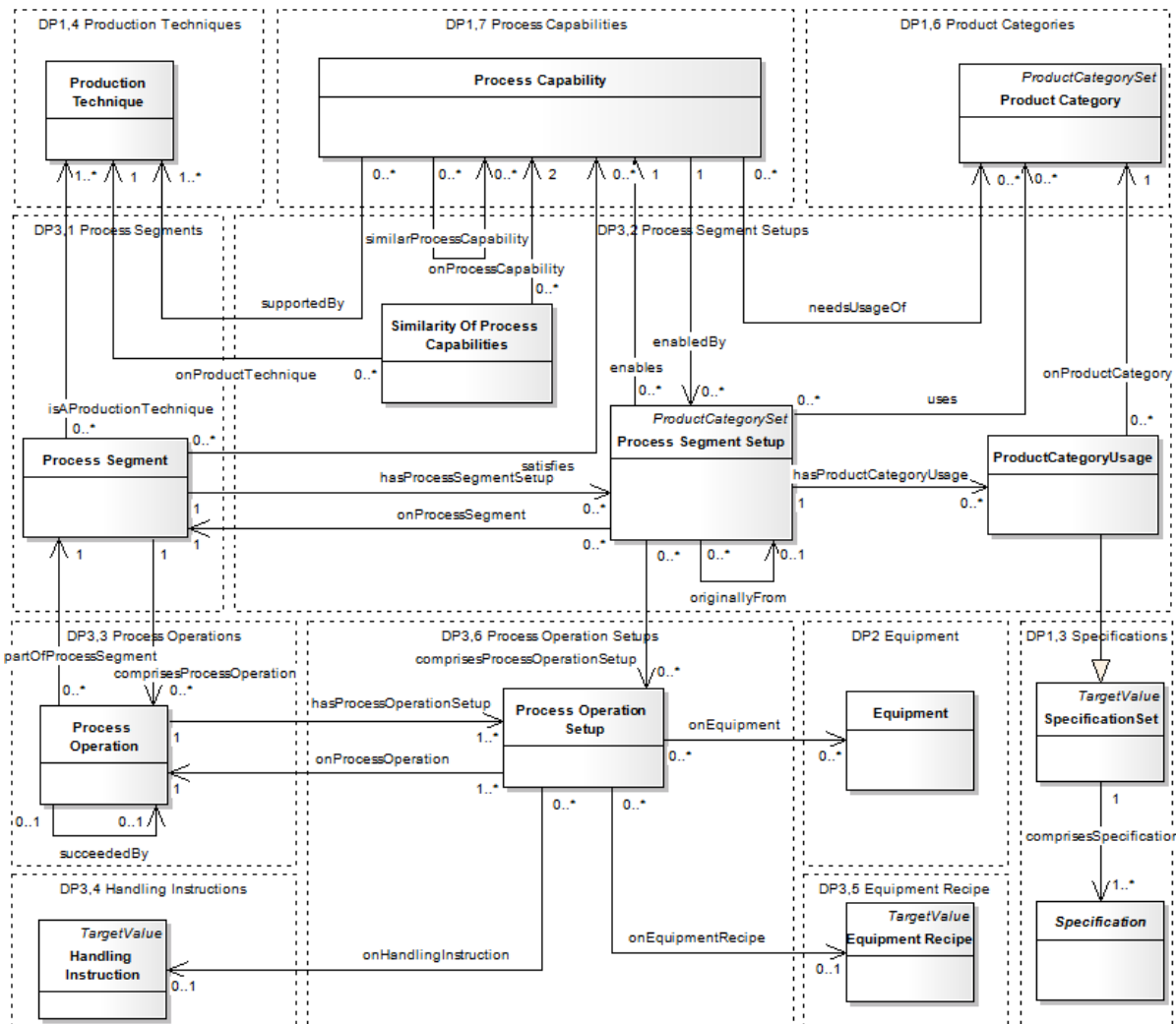


Figure 3.15: The information model as specified by DP_{3,1} to DP_{3,6} and its association to DP_{1,3}, DP_{1,4}, DP_{1,6}, DP_{1,7} and DP₂.

The model of DP_{3,1} (Figure 3.15) introduces the concept *Process Segment* to the information model of DP₃. On the *Process Segment* level, the concept *Process Segment Setup* (DP_{3,2}) comprises settings and instructions which have to be considered across the sequence of *Process Operations* of the *Process Segment*. The *Process Segment Setup* ensures that a *Process Segment* satisfies a particular *Process Capability*. In contrast to *Production Technique*, the concept *Process Segment Setup* provides qualitative information about the support of *Process Capabilities* by *Process Segments*. If a *Process Segment* is a member of a *Production Technique* then it is a potential candidate to satisfy the supported *Process Capability*. If a *Process Segment* has a *Process Segment Setup* which enables a *Process Capability* then the *Process Segment* satisfies this *Process Capability* under the conditions of the *Process Segment Setup*.

A *Process Segment Setup* uses an arbitrary number of *Product Categories*. It is the idea, that the same *Process Segment* can be used to satisfy different *Process Capabilities* just by variation of the used *Product Categories* or the adjustment of some setup parameters. As a consequence, *Product Categories* are used due to *Process Segment Setups* in order to enable a certain *Process Capabilities*. So *Product Categories* do not only require *Process Capabilities* but they are also the input of *Process Segments* (through *Process Segment Setups*) in order to satisfy *Process Capabilities*.

DP_{3,2} introduces the concept *Similarity Of Process Capabilities*. The possibility to determine similarities between *Process Capabilities* through the membership of *Process Segments* in *Production Techniques* is discussed in the context of Figure 3.3. DP_{3,2} introduces the concept *Similarity Of Process Capabilities* to put each pair of affected *Process Capabilities* in relation with the *Production Technique* which led to this assumption.

The concept *Process Operation* and its association to a *Process Segment* is introduced through DP_{3,3}. *Process Segments* represent unique sequences of *Process Operations* in order to satisfy an arbitrary number of *Process Capabilities*. This structure is already device-specific because the sequence of *Process Operations* depends on the capabilities and the structure of the available *Equipment*. A single *Process Operation* may be needed for more powerful *Equipment* of one production system while *Equipment* of another production system may need several sequential *Process Operations* in order to achieve the same result. Within a *Process Segment* there is only a linear sequence of *Process Operations* (no branches, no joins) (Figure 3.16).

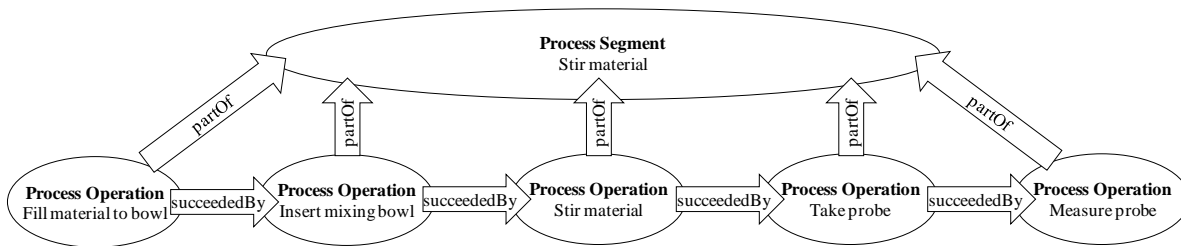


Figure 3.16: Structure of a Process Segment “Stir material”.

Behind every *Process Operation* conceals a more or less extensive series of individual actions, which are executed manually or automatically or as a combination of both. From the perspective of K-RAMP, the detailed structure of those actions is not relevant. Actions to be executed manually by human operators are specified through *Handling Instructions*, while actions to be executed automatically are specified through *Equipment Recipes*. For K-RAMP, only the names of these *Handling Instructions* or *Equipment Recipes* are relevant. This differentiation is justified by the fact that K-RAMP intends to reuse existing instructions and not to create new ones.

Summary of the decomposition of DP₃

DP_{3,1} = *An information model of Process Segments' attributes and associations (to an arbitrary number of process capabilities, to an arbitrary number of production techniques). Assuming Process Capabilities pc₁ and pc₂, a Production Technique pq and a Process Segment ps.*

$\forall x, y. isAProductionTechnique(x, y) \rightarrow isA(x, y).$

DP_{3,2} = *An information model of Process Segment Setups' attributes and associations (to exactly one Process Segment, to an arbitrary number of Process Capabilities, to an arbitrary number of Product Categories, to at least one Specification). Assuming a Product Category pt, a Process Segment ps, a Process Segment Setup pss, a Product Category Usage ptu, a Production Technique pq, Process Capabilities pc, pc₁, pc₂ and Similarity Of Process Capabilities sopc.*

$similarProcessCapability(pc_1, pc_2) \wedge supportedBy(pc_1, pq) \rightarrow$
SimilarityOfProcessCapabilities(sopc), onProductionTechnique(sopc, pq),
onProcessCapability(sopc, pc₁), onProcessCapability(sopc, pc₂)

$supportedBy(pc_1, pq) \wedge supportedBy(pc_2, pq) \wedge DifferentFrom(pc_1, pc_2) \rightarrow$
similarProcessCapability(pc₁, pc₂)

$enabledBy(pc, pss) \wedge onProcessSegment(pss, ps) \wedge isAProductionTechnique(ps, pq) \rightarrow$
supportedBy(pc, pq)

$enables(pss, pc) \rightarrow enabledBy(pc, pss)$

$hasProcessSegmentSetup(ps, pss) \rightarrow onProcessSegment(pss, ps)$

$enabledBy(pc, pss) \wedge uses(pss, pt) \rightarrow needsUsageOf(pt, pc)$

$hasProductCategoryUsage(pss, ptu) \wedge onProductCategory(ptu, pt) \rightarrow uses(pss, pt)$

$hasProcessSegmentSetup(ps, pss) \wedge enables(pss, pc) \rightarrow satisfies(ps, pc)$

DP_{3,3} = *An information model of Process Operations attributes and associations (to maximum one predecessor, to maximum one successor, as membership to exactly one process segment).*

Assuming a Process Operation po and a Process Segment ps

$comprisesProcessOperation(ps, po) \rightarrow partOfProcessSegment(po, ps)$

DP_{3,4} = *An information model of Handling Instructions' attributes.*

DP_{3,5} = *An information model of Equipment Recipes' attributes.*

DP_{3,6} = *An information model of Process Operation Setup attributes and associations (to maximum one Handling Instruction, to maximum one Equipment Recipe, to an arbitrary number of Equipment, to exactly one Process Operation).*

Assuming a Process Operation Setup pos and a Process Operation po

$hasProcessOperationSetup(po, pos) \rightarrow onProcessOperation(pos, po)$

Table 3.14: Summary of the decomposition of DP₃.

Handling Instruction and *Equipment Recipe* are concepts which represent the foundation of such existing information. Their details are developed during qualification processes where

Handling Instructions and *Equipment Recipes* are developed accurately by producing pilot workpieces (see Chapter 2.4). It has to be highlighted that *Handling Instruction* and *Equipment Recipe* are also derived from *Target Value*. It is therefore possible to introduce *Specifications* (e.g. as part of a *Process Capability*) which refers to a particular *Handling Instruction* or *Equipment Recipe*. Beside *Specifications* which describe the process result it is thus also possible to specify a particular instruction which must be followed to achieve this result.

In case of a partially automated production process, a specific *Handling Instruction* (DP_{3,4}) and a specific *Equipment Recipe* (DP_{3,5}) are used in a common context. This common context is represented by a *Process Operation Setup* (DP_{3,6}). A *Process Operation Setup* represents an orchestrated context which needs to be considered by human operators or machines while acting due to a *Process Operation*.

3.2.3.7 Decomposition of design parameter DP₄

DP₄ is decomposed in alignment with the associate functional requirement FR_{4,1}. The yet discussed concepts of matchmaking, particularly in conjunction with DP_{1,3} and DP_{3,2}, are crucial in order to link introduced product-related information of the original production system with existing information of the target production system. This matchmaking is possible on different levels of certainty, and its results shall be the input for target-oriented process qualification. This process qualification adapts the target production system for the production of the introduced product. However, before discussing possible matchmaking scenarios in detail the concepts of *Recommendation* and of *ProductCategorySet* are introduced by DP_{4,1} (Figure 3.17).

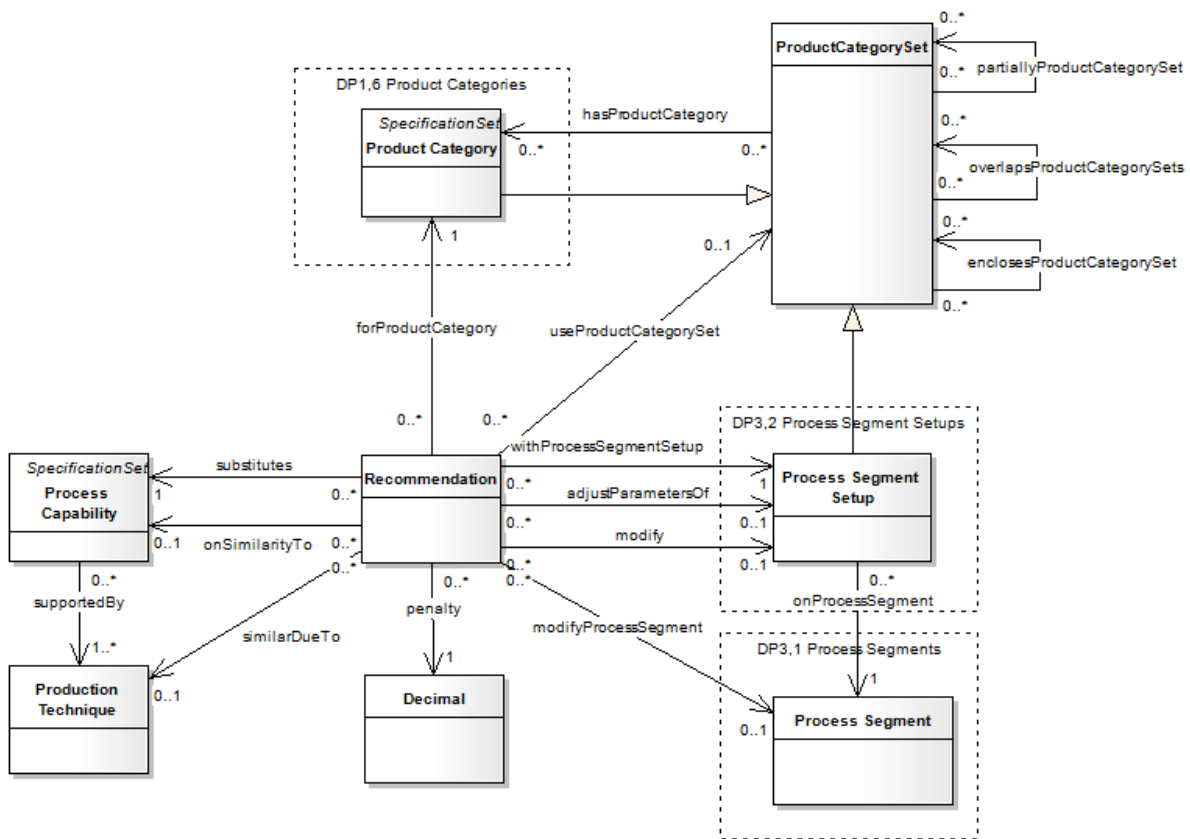


Figure 3.17: Concept of Recommendation and Product Category Set as part of DP_{3,2}.

The associations encloses, overlaps and partially are already known from DP_{1,3}. In the context of *Product Category Sets* the semantics of these associations are extended to sets of *Specification Sets*. However, the particular need for these semantics is limited to sets of *Process Categories* for the purpose of K-RAMP. A *Product Category Set* is a set of *Product Categories* and provides associations (encloses_{PTS}, overlaps_{PTS}, partially_{PTS} where PTS abbreviates *Product Category Set*). *Product Category Sets* are used to match the immediate decomposition structure of *Product Categories* with the used *Product Categories* of *Process Segment Setups*. For this purpose, *Product Categories* and *Process Segment Setups* become specializations of *Product Category Sets* due to DP_{4,1}. *Product Category* is derived from *Product Category Set* because every *Product Category* refers to a set of *Product Categories* which represent its immediate decomposition. *Process Segment Setup* is derived from *Product Category Set* because every *Process Segment Setup* refers to a set of used *Product Categories*.

What are the rules in order to assert those associations? A *Product Category Set* set_1 encloses_{PTS} a *Product Category Set* set_2 if each member of set_1 encloses – mutual enclosing of *Specification Sets* in terms of DP_{1,3} – the equivalent member of set_2 (Table 3.15). This rule requires that each *Product Category* of set_1 encloses an equivalent counterpart in set_2 . That each production system thoroughly specifies each *Product Category* with sufficient entirety, in order to distinct it from other entities in the knowledgebase, is a premise of this approach. The latter ensures that for each member of the one set its counterpart can be determined in the other set.

\forall Specification Sets $set_1 \in$ Product Category Sets 1 and \forall Specification Sets $set_2 \in$ Product Category Sets 2 possible associations between Specification Sets $set_1 R_{set} set_2$			Association between Product Category Sets $ptset_1$ and $ptset_2$	Cardinality		
$R_{set}=\text{encloses}$	$R_{set}=\text{overlaps}$	Different Specifications	$R_{ptset}=\text{encloses}_{PTS}$	$R_{ptset}=\text{overlaps}_{PTS}$	$R_{ptset}=\text{partially}_{PTS}$	
X			X			$ ptset_1 \leq ptset_2 $
X	X			X		$ ptset_1 \leq ptset_2 $
X		X			X	?
X	X	X			X	?
	X			X		$ ptset_1 \leq ptset_2 $
	X	X			X	?
		X				?

Table 3.15: All possible associations between Specifications Sets of two Product Category Sets and the impact on the association between the respective Product Category Sets.

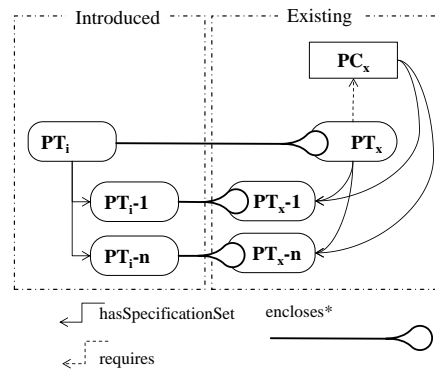
A *Product Category Set* set_1 overlaps_{PTS} with a *Product Category Set* set_2 if at least one member of set_1 overlaps – in terms of *Specification Sets* of DP_{1,3} – the equivalent member of set_2 instead of enclosing it. Again, each *Product Category* of set_1 has an equivalent counterpart in set_2 . Each pair of members of both sets shares *Specifications* with the same *Characteristic* but either their *SpecificationRanges* or their sets of *SpecificationTargets* are only overlapping, as this is already specified in conjunction with DP_{1,3}.

A *Product Category Set* set_1 is partially_{PTS} a *Product Category Set* set_2 if a subset of members of set_1 overlaps or encloses equivalent members of set_2 . In this case, there are members in each set which do not have equivalent counterparts in the other set.

Back to the matchmaking scenarios! The matchmaking scenarios, which are useful for process qualification, are discussed in Table 3.16. The association *enclosesSpecification*, which is already discussed in conjunction with Figure 3.14, is the central concept of this discussion. The same symbols for *Process Capabilities*, *Product Categories* and the association *enclosesPTS* are also applied in the Figures of Table 3.16. Depending on the level of certainty of the matchmaking, a penalty is applied for the respective scenario. The lower the level of certainty, the higher is the applied penalty.

Matchmaking scenarios and levels of certainty

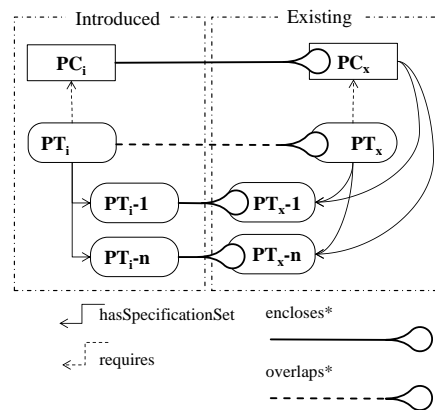
Penalty 0 The introduced *Product Categories* (PT_i) enclose existing *Product Categories* (PT_x). In other words, all *Specification Ranges* and *Specification Targets* of an existing *Product Category* are tighter than the ones of the introduced *Product Category*. In such a case, it is assumed that the existing *Process Capability* (PC_x) of the existing *Product Category* is implicitly enclosed by the introduced *Process Capability* of the introduced *Product Category*, as well. For this reason, it has to be checked whether PT_i encloses_{PTS} the *Process Segment Setups* of the existing *Process Capability* (all decomposing *Product Categories* PT_{i-1} to PT_{i-n} of PT_i enclose their counterparts PT_{x-1} to PT_{x-n} which are used due to the *Process Segment Setups*).



(MS-01)

The recommendation is to link the existing *Process Capability* and the *Process Segment Setup* also with the introduced *Product Category*.

Penalty 0 The introduced *Product Category* (PT_i) overlaps (MS-02) or partially overlaps (MS-03) with an existing *Product Category* (PT_x). Their introduced *Process Capability* encloses the existing *Process Capability*. In this case, only a subset of all *Specifications* of a *Product Category* makes up the *Specification* of the required *Process Capability*. PT_i encloses_{PTS} the *Process Segment Setup* which enables the existing *Process Capability*.



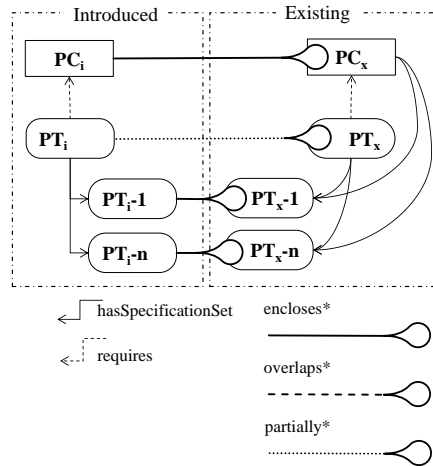
(MS-02)

Same as for the previous case, the recommendation in this case is to link the existing *Process Capability* and the *Process Segment Setup* also for the introduced *Product Category*.

This is also possible for (MS-03) because obviously PC_x uses exactly such *Product Categories* which can compose PT_i , and PC_i is completely satisfied by PC_x . Therefore, it is

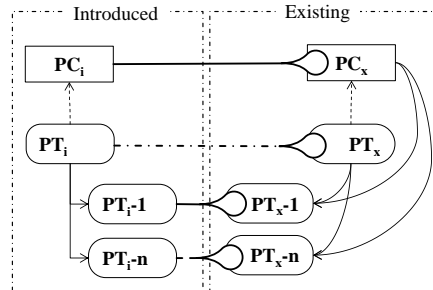
Matchmaking scenarios and levels of certainty

likely that the additional *Specifications* of PT_i and PT_x are not relevant with respect to the selection of PC_x .



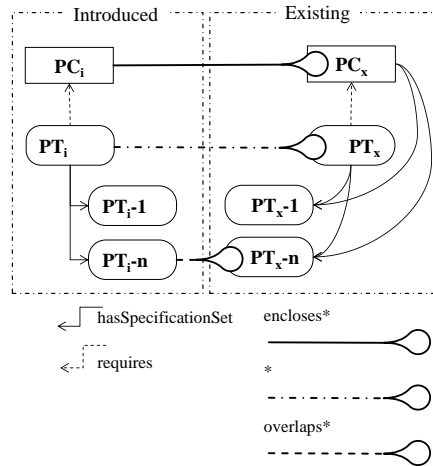
(MS-03)

Penalty 1 The introduced *Product Category* (PT_i) encloses, overlaps or partially overlaps with an existing *Product Category* (PT_x) – this relation is not relevant in this scenario. The respectively introduced *Process Capability* (PC_i) encloses the existing *Process Capability* which is required by PT_x . PT_i overlaps_{PTS} the *Process Segment Setups* which are enabling PC_i . This means that the existing *Process Segment Setup* of PC_i uses *Product Categories* which are very similar to the ones to which PT_i is decomposed (MS-04). Alternatively, PT_i partially_{PTS} overlaps these *Process Segment Setups* (MS-05).



(MS-04)

It is recommended that the existing *Process Capability* and the *Process Segment Setup* are considered as a baseline for setting up an appropriate new *Process Capability* and an appropriate *Process Segment Setup*. As PC_i encloses PC_x the used *Product Categories* have probably no significant impact on the result. Therefore, it is rather likely that only the used *Product Categories* of the new *Process Segment Setup* need to be taken from the decomposition of PT_i keeping the rest of the original *Process Segment Setup* unchanged.



(MS-05)

Matchmaking scenarios and levels of certainty

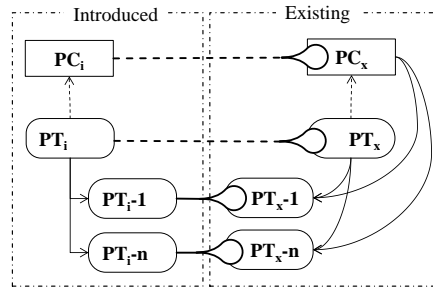
Penalty 2 The introduced *Product Category* (PT_i) overlaps (MS-06) or partially overlaps (MS-07) with the existing *Product Category*, and the required, introduced *Process Capability* overlaps with the respective existing *Process Capability*. PT_i still encloses_{PTS} the *Process Segment Setups* of PC_i .

As a recommendation, the existing *Process Capability* and the *Process Segment Setup* can be considered as a baseline for setting up an appropriate new *Process Capability* and an appropriate *Process Segment Setup*. The relation PC_i overlaps PC_x indicates that some of their *Specifications* only overlap. PT_i encloses_{PTS} the *Process Segment Setup* of PC_x indicates that there is no impact on the deviation of these *Specifications* due to the used *Product Categories*. Depending on the degree of overlapping of the *Process Capabilities*, there may be therefore the chance to adjust setup parameters of the *Process Segment* in order to achieve the *Specifications* of PC_i by keeping the *Process Segment* and the usage of the same *Product Categories* unchanged.

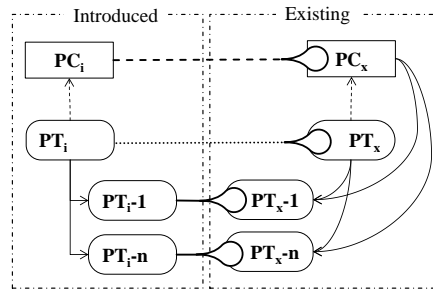
For (MS-07) the same approach is recommended, because PC_i and PC_x represent *Specification Sets* which share the same set of *Characteristics* from the perspective of PC_i . Therefore, it is not essential whether PT_i only partially overlaps PT_x . The relevant *Specifications* for both PT_i and PT_x are obviously covered by their required *Process Capabilities*.

Penalty 3 The introduced *Product Category* (PT_i) overlaps (MS-08 or MS-10) with the existing *Product Category*, and the required introduced *Process Capability* overlaps the respectively existing *Process Capability*. PT_i overlaps_{PTS} (MS-08 or MS-09) or partially_{PTS} overlaps (MS-10 or MS-11) the *Process Segment Setup* of PC_i .

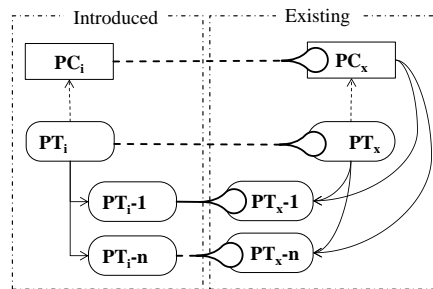
Compared to the previous scenario, in these scenarios the setup parameters of the *Process Segment* and, additionally, the used *Product Categories* may have an impact on the *Specification* of the existing *Process Category*. The introduced and the existing *Process Capabilities* are mutually overlapping each other.



(MS-06)



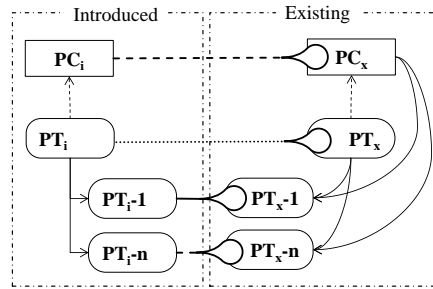
(MS-07)



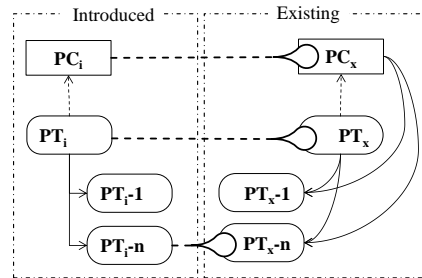
(MS-08)

It is recommended that the existing *Process Capability* and the *Process Segment Setup* are considered as a baseline for setting up an appropriate new *Process Capability* and an appropriate *Process Segment Setup*. However, the used *Product Categories* of the new *Process Segment Setup* need to be taken from the decomposition of PT_i . Moreover, depending on the degree of overlapping of the *Process Capabilities* there may be the need to adjust setup parameters of the *Process Segment* in order to achieve the *Specifications* of PC_i by keeping the *Process Segment*.

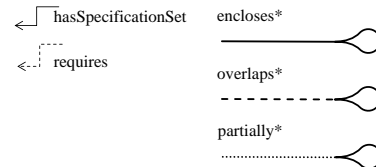
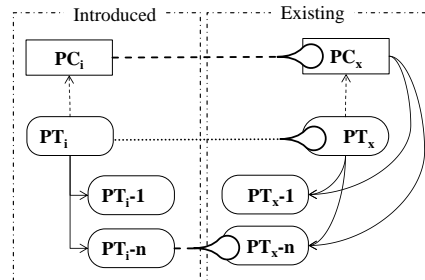
For (MS-09 and MS-11) the same approach is recommended, because PC_i and PC_x represent *Specification Sets* which share the same set of *Characteristics* from the perspective of PC_i . Therefore, it is not essential whether PT_i only partially overlaps PT_x . The relevant *Specifications* for both PT_i and PT_x are obviously covered by their required *Process Capabilities*.



(MS-09)



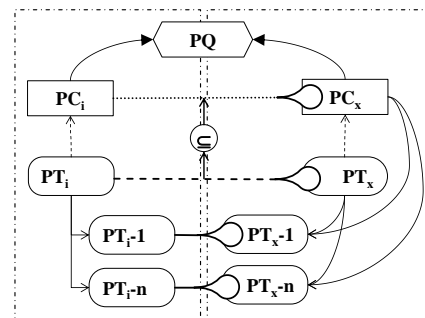
(MS-08)



(MS-09)

Penalty 6 The introduced *Product Category* (PT_i) overlaps (MS-12) or partially overlaps (MS-13) with an existing *Product Category* (PT_x). However, the respectively required introduced *Process Capability* (PC_i) is only partially overlapping with the existing *Process Capability* (PT_x). But PT_i encloses_{PTS} the *Process Segment Setup* of PC_x .

As an essential add-on PC_i and PC_x are considered as similar as they are of the same



(MS-12)

Production Technique (see discussion in conjunction with Figure 3.3).

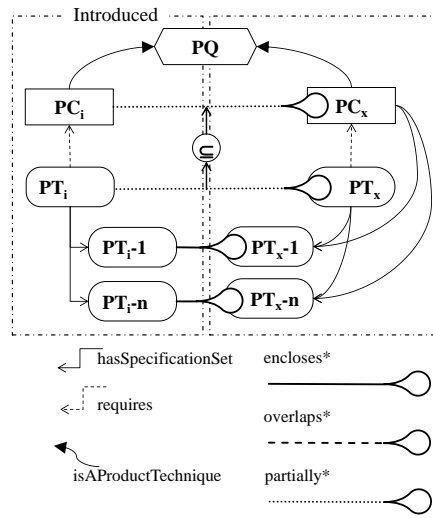
As a recommendation, the existing *Process Capability* and the *Process Segment Setup* can be considered as a baseline for setting up an appropriate new *Process Capability* and an appropriate *Process Segment Setup*. Probably, adjustments of setup parameters are sufficient in order to achieve the specification of PC_i by keeping the *Process Segment* and the usage of the same *Product Categories*. However, due to the determination of similarity through a commonly shared *Production Technique* there is less certainty and thus a higher penalty compared to previous cases where both *Process Capabilities* are really matching.

This recommendation is valid for (MS-12) and (MS-13), because in both cases *Process Capabilities* are considered as similar. Secondly, in both cases the overlapping intersection of *Specifications* between PC_i and PC_x represents a subset of the intersection of *Specifications* of PT_i and PT_x . Thirdly, because PT_i encloses_{PTS} the *Process Segment Setup* of PC_i .

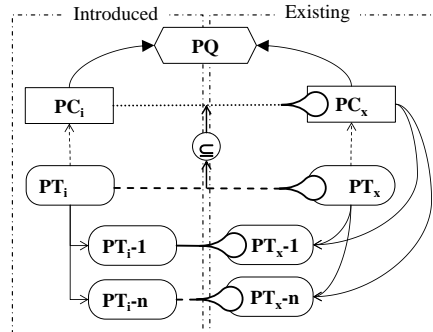
Penalty 15 The introduced *Product Category* (PT_i) overlaps (MS-14 or MS-16) or partially overlaps (MS-15 or MS-17) with an existing *Product Category* (PT_x). The respectively introduced *Process Capability* is only partially overlapping with the existing *Process Capability*. PT_i overlaps_{PTS} (MS-14 or MS-15) or partially_{PTS} overlaps (MS-16 or MS-17) the *Process Segment Setup* of PC_x .

Same as in the previous scenario, PC_i and PC_x are considered as similar as they are of the same *Production Technique*.

It is recommended that the existing *Process Capability* and the *Process Segment Setup* are considered as a baseline for setting up an appropriate new *Process Capability* and an appropriate *Process Segment Setup*. However, the used *Product Categories* of the new *Process Segment Setup* need to be taken from the decomposition of PT_i . Moreover, even the *Process Segment* must be enhanced. Because PC_i and PC_x are only partially overlapping, there are



(MS-13)



(MS-14)

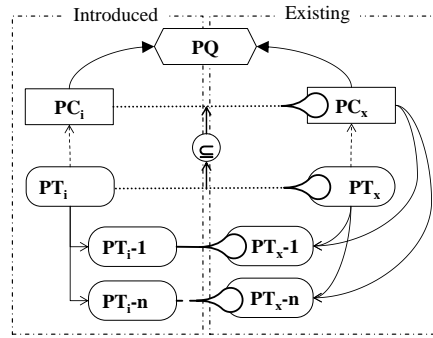
Matchmaking scenarios and levels of certainty

obviously missing *Specifications* from PC_i 's perspective and therefore, rather likely metrology operations (*Process Operations*) are missing at least in the considered *Process Segment* which are gauging according to these missing *Specifications*. These additional metrology operations need to be introduced during the process qualification for enabling the missing *Specifications* according to the needs of PC_i .

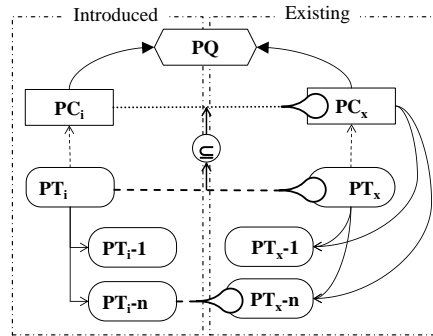
Finally, depending on the ratio of enclosed *Specifications* between both the *Process Capabilities* there is likely the need to adjust some setup parameters of the *Process Segment*.

All measures together may result in an appropriately set up *Process Segment* which likely reuses an adapted copy of the original *Process Segment* and an adapted copy of the original *Process Segment Setup* which uses the decomposition of PT_i in order to meet the *Specification* of PC_i .

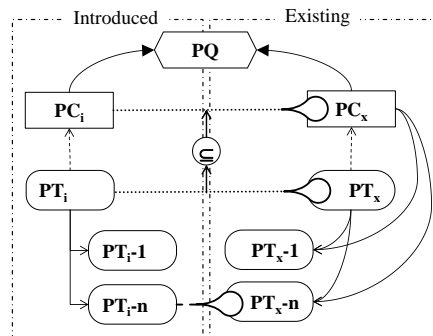
This recommendation is valid for (MS-14) and (MS-15), because in both cases *Process Capabilities* are considered as similar. Secondly, in both cases the intersection of the *Specifications* of PC_i and PC_x represents a subsets of the intersection of the *Specifications* of PT_i and PT_x .



(MS-15)



(MS-16)



(MS-17)

Table 3.16: Matchmaking scenarios and their penalties for process qualification.

The summary of the previously discussed matchmaking scenarios is listed in Table 3.17. It provides a comprehensive overview about the intensity of linkage between pairs of *Product Categories*, *Process Capabilities* or *Product Category Sets* as well as change actions to be

considered during ramp-up and the penalty of the recommendation. The calculation of the penalty is defined as

$$Penalty = n_{PCU} + 2n_{PA} + 4n_{PQ} + 8n_{PS} \dots n_i = \begin{cases} 0 & \dots \text{not relevant} \\ 1 & \dots \text{relevant} \end{cases} \quad (3.1)$$

The higher the penalty, the higher are the duration and costs of the underlying process qualification. It is assumed that the replacement of the *Product Category Usages* (n_{PCU}) has the lowest costs. There are no modifications performed within the existing setup of the *Process Segment* but only the incoming material is modified. Parameter adjustment (n_{PA}) is more expensive because it requires at least the modification of some adjustable parameters of the *Process Segment*, which causes implicitly more expensive and time consuming pilot production runs than in the case of a simple change of incoming material. If similarity of *Process Capabilities* is assumed due to the categorization by a common *Production Technique* (n_{PQ}), some additional penalty points are counted. If it turns out by verification of human experts that the similarity between the *Process Capabilities* is assumed correctly by K-RAMP, then the penalty of the given matchmaking scenario would be 2 instead of 6 or 11 instead of 14 because n_{PQ} can be set to 0. However, this additional penalty is introduced in order to distinct the uncertainty of these matchmaking scenarios from more severe assumptions during the initial recommendation. The most expensive matchmaking scenarios are the ones where modifications of the *Process Segment's* (n_{PS}) sequence are encountered. Therefore, these recommendations are assigned with the highest penalty – meaning the most expensive ramp-up activities.

The implication rules as specified for DP_{4,1} in Table 3.18 can be derived automatically from the settings of Table 3.17 and the explanations of Table 3.16. The quality of relationship between *Product Categories* has an impact on the conditions of the antecedent of these rules: E requires *encloses*, O an *overlaps* and P a *partially* relationship between the introduced *Product Categories* and the existing *Product Categories*. This is a similar situation with *Product Category Set* relationships between the introduced *Product Categories* and the existing *Process Segment Setups* which are enabling existing *Process Capabilities*.

Match-making scenario	Levels of association			changes to consider due to recommendation				Penalty
	Product Category	Process Capability	Product Category Set	Process Segment	Production Technique	Parameter Adjustment	Product Category Usage	
MS-01	E	N/A	E	0	0	0	0	0
MS-02	O	E	E	0	0	0	0	0
MS-03	P	E	E	0	0	0	0	0
MS-04	*	E	O	0	0	0	1	1
MS-05	*	E	P	0	0	0	1	1
MS-06	O	O	E	0	0	1	0	2
MS-07	P	O	E	0	0	1	0	2
MS-08	O	O	O	0	0	1	1	3
MS-09	P	O	O	0	0	1	1	3
MS-10	O	O	P	0	0	1	1	3
MS-11	P	O	P	0	0	1	1	3
MS-12	O	P	E	0	1	1	0	6
MS-13	P	P	E	0	1	1	0	6
MS-14	O	P	O	1	1	1	1	15
MS-15	P	P	O	1	1	1	1	15
MS-16	O	P	P	1	1	1	1	15
MS-17	P	P	P	1	1	1	1	15

E Encloses
O Overlaps
P Partially

Table 3.17: Summary of matchmaking scenarios, changes to be considered by recommendation and penalties.

The corresponding relationships between *Product Category Sets* are *encloses Product Category Set* ($\text{encloses}_{\text{PTS}}$), *overlaps Product Category Set* ($\text{overlaps}_{\text{PTS}}$) and *partially Product Category Set* ($\text{partially}_{\text{PTS}}$). In order to involve the existing *Process Segment Setup*, the chain of relationships *requires* and *enabled By* has to be resolved between the existing *Product Categories* and the existing *Process Segment Setups*. With respect to the relationships between *Process Capabilities* the relationships *encloses*, *overlaps* and *partially* are required accordingly. However, concerning *partially* relationship the conditions are more complex. The relationship *partially* requires that the *Similarity Of Process Capability* entities are resolved in order to invoke the responsible *Production Technique*. Moreover, in such cases it is necessary to verify the *subsetOf* relationship between the common set of *Characteristics* of the *Process Capabilities* and the common set of *Characteristics* of the *Product Categories*.

The consequents of the implication rules which are specified by $\text{DP}_{4,1}$, assert a *Recommendation* entity as well as its relationships *forProductCategory*, *substitute*, *withProcessSegmentSetup* and *penalty*. If the *Product Category Usage* has to be adapted in accordance with the introduced *Product Category* the relationship *useProductCategorySet* is asserted accordingly. If parameter adjustment is required an additional *adjustParameterOf* relationship is asserted. In case of an involved *Production Technique* due to a *partially* relationship between the introduced *Process Capability* and the existing *Process Capability*, *onSimilarityTo* and *similarDueTo* relationships are asserted. A *modify* relationship is asserted if modification of the underlying *Process Segment* needs to be considered.

A *Recommendation* holds all information which is needed in order to derive an appropriate *Process Segment Setup* for enabling a *Process Capability* which can be required by the introduced *Product Category*. The associations of Figure 3.17 can be interpreted as a recommendation sentence in the form: *For production of a Product Category PT with Process Segment Setup PSS through a Process Capability (accessible through PSS) which substitutes Process Capability PC with a given penalty, use the Product Category Set (usually) PT, by optionally considering the similarity to PC which is assumed with uncertainty due to membership in Production Technique PQ, by optionally adjusting parameters of PSS in order to achieve a supporting Process Segment (accessible through PSS) which optionally needs to be modified.*

Summary of the decomposition of DP_4

$\text{DP}_{4,1}$ = An information model of *Recommendation* and *Product Category Set* including attributes and associations. Assuming a *Recommendation* rc , *Product Category Sets* pts_1, pts_2 , *Product Categories* pti, ptx, pt_1, pt_2 , a *Process Segment Setup* pss , a *Production Technique* pq , *Process Capabilities* pci, pcx and *Similarity Of Process Capabilities* $sopc$.

$\text{encloses}(pti, ptx) \wedge \text{requires}(ptx, pcx) \wedge \text{requires}(pti, pci) \wedge \text{enabledBy}(pcx, pss) \wedge \text{enclosesProductCategorySet}(pti, pss) \rightarrow \text{Recommendation}(rc), \text{forProductCategory}(rc, pti), \text{substitute}(rc, pci), \text{withProcessSegmentSetup}(pc, pss), \text{penalty}(rc, 0)$

$\text{overlaps}(pti, ptx) \wedge \text{requires}(ptx, pcx) \wedge \text{requires}(pti, pci) \wedge \text{encloses}(pci, pcx) \wedge \text{enabledBy}(pcx, pss) \wedge \text{enclosesProductCategorySet}(pti, pss) \rightarrow \text{Recommendation}(rc), \text{forProductCategory}(rc, pti), \text{substitute}(rc, pci), \text{withProcessSegmentSetup}(pc, pss), \text{penalty}(rc, 0)$

$\text{partially}(pti, ptx) \wedge \text{requires}(ptx, pcx) \wedge \text{requires}(pti, pci) \wedge \text{encloses}(pci, pcx) \wedge \text{enabledBy}(pcx, pss) \wedge \text{enclosesProductCategorySet}(pti, pss) \rightarrow \text{Recommendation}(rc), \text{forProductCategory}(rc, pti), \text{substitute}(rc, pci), \text{withProcessSegmentSetup}(pc, pss), \text{penalty}(rc, 0)$

Summary of the decomposition of DP₄

requires (ptx, pcx) \wedge *requires* (pti, pci) \wedge *encloses* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *overlapsProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *useProductCategorySet* (rc, pti), *penalty* (rc, 1)

requires (ptx, pcx) \wedge *requires* (pti, pci) \wedge *encloses* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *partiallyProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *useProductCategorySet* (rc, pti), *penalty* (rc, 1)

overlaps (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *overlaps* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *enclosesProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *adjustParametersOf* (rc, pss), *penalty* (rc, 2)

partially (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *overlaps* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *enclosesProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *adjustParametersOf* (rc, pss), *penalty* (rc, 2)

overlaps (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *overlaps* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *overlapsProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *useProductCategorySet* (rc, pti), *adjustParametersOf* (rc, pss), *penalty* (rc, 3)

partially (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *overlaps* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *overlapsProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *useProductCategorySet* (rc, pti), *adjustParametersOf* (rc, pss), *penalty* (rc, 3)

overlaps (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *overlaps* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *partiallyProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *useProductCategorySet* (rc, pti), *adjustParametersOf* (rc, pss), *penalty* (rc, 3)

partially (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *overlaps* (pci, pcx) \wedge *enabledBy* (pcx, pss) \wedge *partiallyProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *useProductCategorySet* (rc, pti), *adjustParametersOf* (rc, pss), *penalty* (rc, 3)

overlaps (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *partially* (pci, pcx) \wedge *onProcessCapability* (sopc, pci) \wedge *onProcessCapability* (sopc, pcx) \wedge *onProductionTechnique* (sopc, pq) \wedge *fromSpecificationSet* (ross₁, pci) \wedge *toSpecificationSet* (ross₁, pcx) \wedge *fromSpecificationSet* (ross₂, pti) \wedge *toSpecificationSet* (ross₁, ptx) \wedge *subsetOf* (ross₁, ross₂) \wedge *enabledBy* (pcx, pss) \wedge *enclosesProductCategorySet* (pti, pss) \rightarrow *Recommendation* (rc), *forProductCategory* (rc, pti), *substitute* (rc, pci), *withProcessSegmentSetup* (pc, pss), *adjustParametersOf* (rc, pss), *onSimilarityTo* (rc, pci), *similarDueTo* (rc, pq), *penalty* (rc, 6)

partially (pti, ptx) \wedge *requires* (ptx, pcx) \wedge *requires* (pti, pci) \wedge *partially* (pci, pcx) \wedge

Summary of the decomposition of DP₄

$onProcessCapability(sopc, pci) \wedge onProcessCapability(sopc, pcx) \wedge$
 $onProductionTechnique(sopc, pq) \wedge fromSpecificationSet(ross_1, pci) \wedge$
 $toSpecificationSet(ross_1, pcx) \wedge fromSpecificationSet(ross_2, pti) \wedge$
 $toSpecificationSet(ross_1, ptx) \wedge subsetOf(ross_1, ross_2) \wedge enabledBy(pcx, pss) \wedge$
 $enclosesProductCategorySet(pti, pss) \rightarrow Recommendation(rc),$
 $forProductCategory(rc, pti), substitute(rc, pci), withProcessSegmentSetup(pc, pss),$
 $adjustParametersOf(rc, pss), onSimilarityTo(rc, pci), similarDueTo(rc, pq),$
 $penalty(rc, 6)$

$overlaps(pti, ptx) \wedge requires(ptx, pcx) \wedge requires(pti, pci) \wedge partially(pci, pcx) \wedge$
 $onProcessCapability(sopc, pci) \wedge onProcessCapability(sopc, pcx) \wedge$
 $onProductionTechnique(sopc, pq) \wedge fromSpecificationSet(ross_1, pci) \wedge$
 $toSpecificationSet(ross_1, pcx) \wedge fromSpecificationSet(ross_2, pti) \wedge$
 $toSpecificationSet(ross_1, ptx) \wedge subsetOf(ross_1, ross_2) \wedge enabledBy(pcx, pss) \wedge$
 $overlapsProductCategorySet(pti, pss) \rightarrow Recommendation(rc),$
 $forProductCategory(rc, pti), substitute(rc, pci), withProcessSegmentSetup(pc, pss),$
 $useProductCategorySet(rc, pti), adjustParametersOf(rc, pss), onSimilarityTo(rc, pci),$
 $similarDueTo(rc, pq), modify(rc, pss), penalty(rc, 15)$

$partially(pti, ptx) \wedge requires(ptx, pcx) \wedge requires(pti, pci) \wedge partially(pci, pcx) \wedge$
 $onProcessCapability(sopc, pci) \wedge onProcessCapability(sopc, pcx) \wedge$
 $onProductionTechnique(sopc, pq) \wedge fromSpecificationSet(ross_1, pci) \wedge$
 $toSpecificationSet(ross_1, pcx) \wedge fromSpecificationSet(ross_2, pti) \wedge$
 $toSpecificationSet(ross_1, ptx) \wedge subsetOf(ross_1, ross_2) \wedge enabledBy(pcx, pss) \wedge$
 $overlapsProductCategorySet(pti, pss) \rightarrow Recommendation(rc),$
 $forProductCategory(rc, pti), substitute(rc, pci), withProcessSegmentSetup(pc, pss),$
 $useProductCategorySet(rc, pti), adjustParametersOf(rc, pss), onSimilarityTo(rc, pci),$
 $similarDueTo(rc, pq), modify(rc, pss), penalty(rc, 15)$

$overlaps(pti, ptx) \wedge requires(ptx, pcx) \wedge requires(pti, pci) \wedge partially(pci, pcx) \wedge$
 $onProcessCapability(sopc, pci) \wedge onProcessCapability(sopc, pcx) \wedge$
 $onProductionTechnique(sopc, pq) \wedge fromSpecificationSet(ross_1, pci) \wedge$
 $toSpecificationSet(ross_1, pcx) \wedge fromSpecificationSet(ross_2, pti) \wedge$
 $toSpecificationSet(ross_1, ptx) \wedge subsetOf(ross_1, ross_2) \wedge enabledBy(pcx, pss) \wedge$
 $partiallyProductCategorySet(pti, pss) \rightarrow Recommendation(rc),$
 $forProductCategory(rc, pti), substitute(rc, pci), withProcessSegmentSetup(pc, pss),$
 $useProductCategorySet(rc, pti), adjustParametersOf(rc, pss), onSimilarityTo(rc, pci),$
 $similarDueTo(rc, pq), modify(rc, pss), penalty(rc, 15)$

$partially(pti, ptx) \wedge requires(ptx, pcx) \wedge requires(pti, pci) \wedge partially(pci, pcx) \wedge$
 $onProcessCapability(sopc, pci) \wedge onProcessCapability(sopc, pcx) \wedge$
 $onProductionTechnique(sopc, pq) \wedge fromSpecificationSet(ross_1, pci) \wedge$
 $toSpecificationSet(ross_1, pcx) \wedge fromSpecificationSet(ross_2, pti) \wedge$
 $toSpecificationSet(ross_1, ptx) \wedge subsetOf(ross_1, ross_2) \wedge enabledBy(pcx, pss) \wedge$
 $partiallyProductCategorySet(pti, pss) \rightarrow Recommendation(rc),$
 $forProductCategory(rc, pti), substitute(rc, pci), withProcessSegmentSetup(pc, pss),$
 $useProductCategorySet(rc, pti), adjustParametersOf(rc, pss), onSimilarityTo(rc, pci),$

Summary of the decomposition of DP₄

similarDueTo (*rc*, *pq*), *modify* (*rc*, *pss*), *penalty* (*rc*, *l5*)

$\exists pts_1, pts_2 \forall pt_1 \exists pt_2. hasProductCategory (pts_1, pt_1) \wedge hasProductCategory (pts_2, pt_2) \wedge encloses (pt_1, pt_2) \rightarrow enclosesProductCategorySet (pts_1, pts_2)$

$\exists pts_1, pts_2 \exists pt_1 \exists pt_2. hasProductCategory (pts_1, pt_1) \wedge hasProductCategory (pts_2, pt_2) \wedge overlaps (pt_1, pt_2) \wedge (\forall pt_1' \exists pt_2'. pt_1 \neq pt_1' \wedge pt_2 \neq pt_2' \wedge encloses (pt_1', pt_2')) \rightarrow overlapsProductCategorySet (pts_1, pts_2)$

$\exists pts_1, pts_2 \forall pt_1 \exists pt_2. \neg(enclosesProductCategorySet (pts_1, pts_2) \vee overlapsProductCategorySet (pts_1, pts_2)) \wedge$

$hasProductCategory (pts_1, pt_1) \wedge hasProductCategory (pts_2, pt_2) \wedge$

$(encloses (pt_1, pt_2) \vee overlaps (pt_1, pt_2) \vee partially (pt_1, pt_2)) \rightarrow$

$partiallyProductCategorySet (pts_1, pts_2)$

$\forall pt, pt'. decomposedToProductCategory (pt, pt') \rightarrow hasProductCategory (pt, pt')$

$\forall pss, pt. uses (pss, pt) \rightarrow hasProductCategory (pss, pt)$

Table 3.18: Summary of the decomposition of DP₄.

3.2.3.8 Decomposition of process variable PV₁

The decomposition of design parameters consequently leads to the decomposition of process variables. In the sequel, the further decomposition of PV₁ to PV₄ is described considering the already derived decomposition structure and information models of the associated design parameters.

Summary of the decomposition of PV₁

PV_{1,1} = A set of procedures to manage and exchange information about Engineering Units.

PV_{1,2} = A set of procedures to manage and exchange information about Characteristics.

PV_{1,3} = A set of procedures to manage and exchange information about Specification Ranges and Specification Targets, and a procedure to reason the relationships encloses and overlaps between pairs of Specification Targets or Specification Ranges, as well as the relationships encloses, overlaps, partially and subset Of between pairs of Specification Sets.

PV_{1,4} = A set of procedures to manage and exchange information about Production Techniques and a procedure to reason the effect of the transitivity of the relationship specializes.

PV_{1,5} = A set of procedures to manage and exchange information about Products.

PV_{1,6} = A set of procedures to manage and exchange information about Product Categories and a procedure to reason specialization hierarchies of Product Categories based on the relationship encloses between pairs of Product Categories. Moreover, a procedure to reason the relationship isA between Products and Product Categories.

PV_{1,7} = A set of procedures to manage and exchange information about Process Capabilities and a procedure to reason specialization hierarchies of Process Capabilities based on the relationship encloses between pairs of Process Capabilities.

Table 3.19: Summary of the decomposition of PV₁.

All process variables $PV_{1,1}$ to $PV_{1,7}$ comprise sets of procedures to manage and exchange information which is organized in accordance to the information models of the associated design parameters. In addition, $PV_{1,3}$ allows to determine relationships *encloses*, *overlaps* and *partially* between *Specification Ranges* respectively *Specification Targets*. The relationship *encloses* is used by the process variables $PV_{1,5}$ to $PV_{1,7}$. $PV_{1,5}$ determines *isA*-relationships between *Products* and *Product Categories*. The relationships *overlaps* and *partially* are used to determine *Specifications* which have overlapping value range or share common *Characteristics* at least. $PV_{1,6}$ determines the relationship *specializes* between *Product Categories*, and $PV_{1,7}$ determines *specializes*-relationships between *Process Capabilities*.

3.2.3.9 Decomposition of process variable PV_2

As there is no decomposition of DP_2 , there is also no decomposition of PV_2 required.

3.2.3.10 Decomposition of process variable PV_3

PV_3 is the next process variable to be decomposed in order to support the previously performed decomposition of DP_3 . $PV_{3,1}$ as well as $PV_{3,3}$ to $PV_{3,6}$ are trivial and only focused on the management of the information models of the associated design parameters.

$PV_{3,2}$ is of more complexity as it has to provide reasoning logic in order to cover the specified implication rules of $DP_{3,2}$ (Table 3.20).

Summary of the decomposition of PV_3

$PV_{3,1}$ = A set of procedures to manage *Process Segments*.

$PV_{3,2}$ = A set of procedures to manage *Process Segment Setups*. Moreover, a procedure to reason the relationship *satisfies* between *Process Segments* and *Process Capabilities* and the relationship *needs Usage Of* between *Process Capabilities* and *Product Categories*. Furthermore, a procedure is provided which resolves similarities between *Process Capabilities* (Figure 3.3) ($DP_{1,7}$) by using the relationship *isA* to *Production Technique* and the relationship *satisfies* to *Process Capability* for each *Process Segment* as well as the assertion of *Similarity Of Process Capabilities* between similar *Process Capabilities*.

$PV_{3,3}$ = A set of procedures to manage *Process Operations*.

$PV_{3,4}$ = A set of procedures to manage *Handling Instructions*.

$PV_{3,5}$ = A set of procedures to manage *Equipment Recipes*.

$PV_{3,6}$ = A set of procedures to manage *Process Operation Setups*.

Table 3.20: Summary of the decomposition of PV_3 .

3.2.3.11 Decomposition of process variable PV_4

$PV_{4,1}$ is the only process variable which results from the decomposition of PV_4 . It covers primarily the reasoning logic which is needed to cover the specified implication rules of $DP_{4,1}$ (Table 3.21).

Summary of the decomposition of PV₄

PV_{4,1} = A set of procedures to manage Recommendations. Moreover, a set of procedures for reasoning of relationships encloses_{PTS}, overlaps_{PTS} and partially_{PTS} between pairs of Product Category Sets.

Table 3.21: Summary of the decomposition of PV₄.

3.2.4 Product design

The product design is started by establishing a dependency matrix between FRs and DPs as introduced in the previous chapters. For the reason of completeness, the dependency of the information models between each other is checked by answering the question, “Which functional requirement is influenced through the specification of a particular design parameter?”

In Table 3.22 the result of this process is visualized as the so-called product design matrix. As only matrix elements below the main diagonal are populated, a decoupled design is in place.

		DP ₁							DP ₂	DP ₃						DP ₄
		DP _{1,1}	DP _{1,2}	DP _{1,3}	DP _{1,4}	DP _{1,5}	DP _{1,6}	DP _{1,7}		DP _{3,1}	DP _{3,2}	DP _{3,3}	DP _{3,4}	DP _{3,5}	DP _{3,6}	DP _{4,1}
FR ₁	FR _{1,1}	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	FR _{1,2}	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0
	FR _{1,3}	X	X	X	0	0	0	0	0	0	0	0	0	0	0	0
	FR _{1,4}	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0
	FR _{1,5}	X	X	X	0	X	0	0	0	0	0	0	0	0	0	0
	FR _{1,6}	X	X	X	0	X	X	0	0	0	0	0	0	0	0	0
	FR _{1,7}	X	X	X	0	X	X	X	0	0	0	0	0	0	0	0
FR ₂		0	0	0	0	0	0	0	X	0	0	0	0	0	0	0
FR ₃	FR _{3,1}	0	0	0	X	0	0	0	0	X	0	0	0	0	0	0
	FR _{3,2}	X	X	X	X	X	X	X	0	X	X	0	0	0	0	0
	FR _{3,3}	0	0	0	X	0	0	0	0	X	0	X	0	0	0	0
	FR _{3,4}	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0
	FR _{3,5}	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0
	FR _{3,6}	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0
FR ₄	FR _{4,1}	X	X	X	X	X	X	X	0	X	X	0	0	0	0	X

Table 3.22: Product design matrix

3.2.5 Process design

The process design is started by establishing a dependency matrix between DPs and PVs, as introduced in the Chapters 3.2.3.1 to 3.2.3.11. The target of the process design is to determine chronological dependencies between the previously specified process variables. The question, “Which design parameters are directly or indirectly influenced due to the specification of a particular process variable?” has to be answered during the completion of the process design matrix for each combination of PVs and DPs.

		PV ₁							PV ₂	PV ₃						PV ₄
		PV _{1,1}	PV _{1,2}	PV _{1,3}	PV _{1,4}	PV _{1,5}	PV _{1,6}	PV _{1,7}		PV _{3,1}	PV _{3,2}	PV _{3,3}	PV _{3,4}	PV _{3,5}	PV _{3,6}	
DP ₁	DP _{1,1}	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	DP _{1,2}	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0
	DP _{1,3}	X	X	X	0	X	X	X	0	0	X	0	0	0	0	0
	DP _{1,4}	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0
	DP _{1,5}	X	X	X	0	X	0	0	0	0	0	0	0	0	0	0
	DP _{1,6}	X	X	X	0	X	X	0	0	0	0	0	0	0	0	0
	DP _{1,7}	X	X	X	0	X	X	X	0	0	0	0	0	0	0	0
DP ₂		0	0	0	0	0	0	0	X	0	0	0	0	0	0	
DP ₃	DP _{3,1}	0	0	0	X	0	0	0	0	X	0	0	0	0	0	0
	DP _{3,2}	X	X	X	X	X	X	X	0	X	X	0	0	0	0	0
	DP _{3,3}	0	0	0	X	0	0	0	0	X	0	X	0	0	0	0
	DP _{3,4}	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0
	DP _{3,5}	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0
	DP _{3,6}	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0
DP ₄	DP _{4,1}	X	X	X	X	X	X	X	0	0	X	0	0	0	0	X

Table 3.23: Process design matrix.

The result of this process in Table 3.22 shows that there are also matrix elements above the main diagonal. Therefore, a coupled process design is in place. As a consequence [18, pp. 211-213], there are also feedback junctions in the process design.

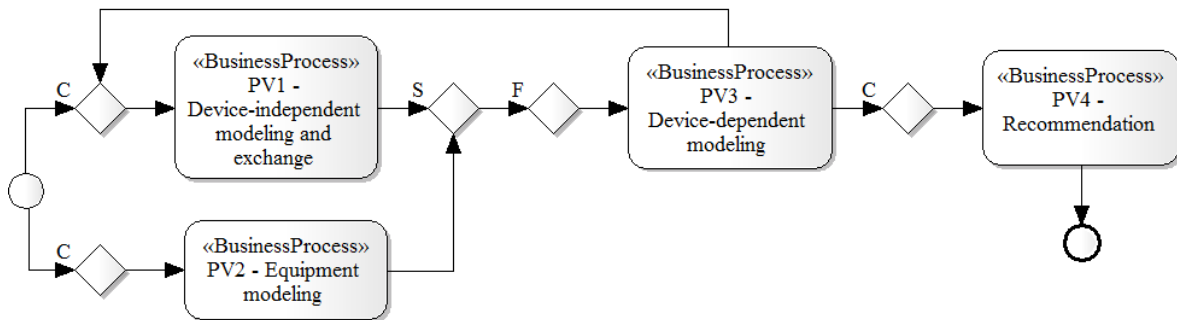


Figure 3.18: Overall flow of K-RAMP process.

The major junctions of process variables and design parameters are highlighted in Table 3.23 with a bolt frame. In Figure 3.18, the overall flow of the K-RAMP process is shown. For easier further discussion, process variables are named from now on. With respect to PV₁ – the process of *Device-independent modeling and exchange* – and for PV₂ – the process of *Equipment modeling* – there are no dependencies on other process variables. The connecting elements “C”, “S” and “F” are abbreviations for “Control junction”, “Summing junction” and “Feedback junction” [18, p. 211].

This is also visible through the process design matrix in Table 3.23, as there are no populated matrix elements in the intersecting areas of both process variables. This is different for PV₃ – the process of *Device-dependent modeling* – as well as for PV₄ – the process of *Recommendation*. Both process variables depend on other process variables. Consequently, this results in a sequence of PV₃ and PV₄ in the overall K-RAMP process as shown in Figure 3.18.

On the second level of decomposition, the detailed sequences of all process variables are derived accordingly. Figure 3.19 illustrates the detailed sequence of *Device-independent modeling and exchange* (PV₁). PV_{1,1} – the process of *Engineering Units modeling and exchange* – as well as PV_{1,4} – the process of *Production Techniques modeling and exchange* – do not depend on other process variables. However, PV_{1,2}, PV_{1,3}, PV_{1,5}, PV_{1,6} and PV_{1,7} are performed in this sequence due to their dependencies according to the process design matrix.

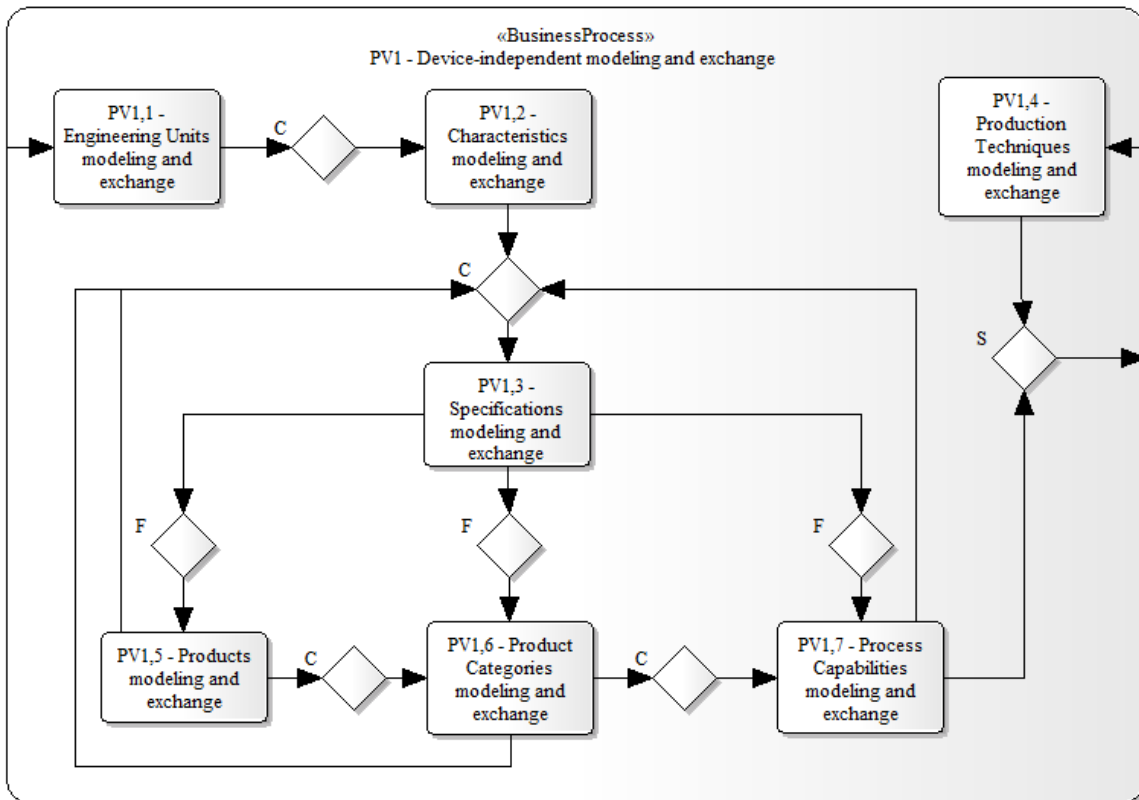


Figure 3.19: Sequence of PV₁ – Device-independent modeling and exchange.

According to the detailed sequence of PV₁ it is not required, that the whole information model represented by DP₁ is exchanged at once between the original production system and the target production system. It is also possible to exchange individual partitions of the information model, namely the information models of DP_{1,1} to DP_{1,7}, as long as this partial exchange follows the sequential structure of PV₁.

PV₂ is not decomposed further. There is no more detailed specification of this process variable required accordingly in the scope of this work.

The detailed sequence of PV₃ is shown in Figure 3.20 and is derived from dependencies of the process design matrix as well. PV_{3,1} – the initial process of *Process Segment modeling* –, as well as PV_{3,3} (*Handling instruction modeling*) and PV_{3,4} (*Equipment Recipe modeling*) are independent

of any other process variables in this decomposition tree. $PV_{3,6}$ does only require $PV_{3,1}$ first, while $PV_{3,5}$ (*Process Operation Setup modeling*) requires $PV_{3,4}$, $PV_{3,3}$, $PV_{3,2}$ to be performed first.

PV_4 is represented by a single encapsulated activity which is also named *Recommendation*. For this reason, it is sufficient to refer to Figure 3.18 in order to localize $PV_{4,1}$ at the position of PV_4 within the overall K-RAMP process.

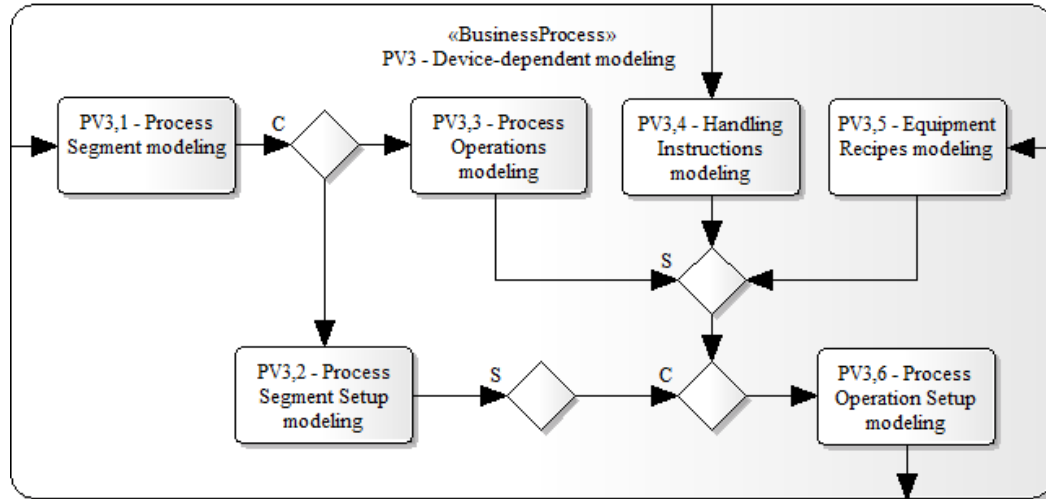


Figure 3.20: Sequence of PV_3 – Device-dependent modeling.

3.2.6 Verification of Question 1

In the previous chapters, a multidisciplinary knowledge base is designed using an axiomatic design approach. The essential customer needs (CA) are derived from (Question 1), and also the root functional requirement (FR) as well as the design constraints C_1 to C_4 , as explained in Chapter 3.2.1. The further derivation of design parameters and process variables as well as the decomposition of FR, DP and PV result in the previously described design of K-RAMP. Does the design support the needs being introduced through (Question 1)?

According to axiomatic design, the most general aspects of specific information models (DP_1 to DP_4) are specified for each functional requirement (FR_1 to FR_4). In Chapter 3.2.2.1, the impact of these main functional requirements on the specified customer needs is explained, and the satisfaction of these main functional requirements through appropriate design parameters is explained in Chapter 3.2.2.2. Implementing the herein specified information models satisfies the associated functional requirements, which consequently satisfies the customer needs. Chapter 3.2.2.3 covers the dynamic part of the knowledge base design and describes procedures (a.k.a. process variables – PV_1 to PV_4) which need to be executed upon the information models of DP_1 to DP_4 . By implementation of the specified process variables, the dynamic behavior of the specified information models is ensured, with the indirect impact on the customer needs.

Axiomatic design applies zig-zagging as an approach for decomposition of design elements of each domain. An interdependent zig-zagging logic is applied during the design of the interdisciplinary knowledge base accordingly. This decomposition is described in the Chapter 3.2.3. Each main functional requirement is decomposed in accordance to partial functional requirements. These partial functional requirements of each functional requirement need to be satisfied by the decomposition elements of the associated design parameter. Additionally, the summary of all decomposed functional requirements is a detailed and comprehensive view to the respective

superior functional requirement. The same decomposition logic is also applied between design parameters and process variables. As logical statements, the dependencies of all design elements which are introduced in the Chapters 3.2.1 to 3.2.3 can be written as follows.

$CA \leftarrow FR \leftarrow DP \cup PV$ can be enhanced by 1st decomposition to

$CA \leftarrow \bigcup_{i=1..4} FR_i \leftarrow [\bigcup_{j=1..4} DP_j] \cup [\bigcup_{k=1..4} PV_k]$ with

$FR_i \leftarrow DP_j \cup PV_k \dots \text{if } i = j = k$

By the 2nd decomposition this statement can be further enhanced to

$CA \leftarrow \bigcup_{i=1..4} \bigcup_l FR_{i,l} \leftarrow [\bigcup_{j=1..4} \bigcup_m DP_{j,m}] \cup [\bigcup_{k=1..4} \bigcup_n PV_{k,n}]$ with

$FR_{i,l} \leftarrow DP_{j,m} \cup PV_{k,n} \dots \text{if } i = j = k \text{ and } l = m = n.$

The design of the multidisciplinary knowledge base K-RAMP can be further specified as the summary of all designed information models and procedures. Therefore, it can be stated that

$K\text{-RAMP} = [\bigcup_{j=1..4} \bigcup_m DP_{j,m}] \cup [\bigcup_{k=1..4} \bigcup_n PV_{k,n}]$ and therefore

$CA \leftarrow \bigcup_{i=1..5} \bigcup_l FR_{i,l} \leftarrow K\text{-RAMP}.$

Resulting from this conclusion, the derived design satisfies (Question 1). Based on the previously described design the next questions to be answered are related to the utilization of features of the Semantic Web. As it can be assumed that the design solves (Question 1), ontology models are developed by consideration of OWL DL features in the sequel of this work.

	Statement	Technical need
S1	An information model of X.	The technical need is a meta-language in order to describe an information model of X.
S2	A procedure to reason new information of X.	The technical need is to reason new information from existing information based on subsets of the information model X plus reasoning rules, which are part of the information model as well.
S3	A set of procedures to manage information X.	The technical need is to perform procedures in order to store new subsets of information based on information model X, or to modify, query and delete existing subsets of information.
S4	A set of procedures to manage and exchange information.	The technical need is the same as for S3 plus procedures to export, transfer and import subsets of information based on information model X.

Table 3.24: Classification of general design statements.

3.3 Ontology models by using the Semantic Web

Based on the general design, in this chapter a specific design is verified which applies dedicated features of the Semantic Web. It is shown that the applied features of the Semantic Web are only able to support a portion of all aspects of the general design. Therefore, the specific design is a hybrid design. For this purpose, the shortcomings of the Semantic Web with respect to the general design are discussed in the sequel of this chapter.

First, the set of chosen features of the Semantic Web by definition are limited to the ones which are recommended by the World Wide Web Consortium and are thus supported by numerous

software products today. This constraint is important as the proposed approach shall be applicable in manufacturing industries. For this reason, the proposed approach shall be supported immediately and with high reliability by a broad variety of existing software products of the Semantic Web's domain.

In particular, the applied features of the Semantic Web are the Resource Description Framework (RDF) [35], the RDF Schema (RDFS) [50], the SPARQL Protocol and RDF Query Language [37], the Web Ontology Language (OWL2) [51] and the Semantic Web Rule Language (SWRL) [52]. Moreover, the described information models must be implemented in a decidable way (constraint C4). For this reason, ontology models of K-RAMP are specified as OWL 2 DL.

DPs	RDF	RDFS	SPARQL	OWL2	SWRL	Reasoning	unsupported	Comment
DP _{1,1}	✓	✓		✓				
DP _{1,2}	✓	✓		✓	✓			
DP _{1,3}	✓	✓		✓	✓		X	<i>Specification reasoning rules for enclosesSpecificationSet, enclosesSpecificationTarget, subsetOf, RelationOfSpecificationSets, overlapsSpecificationTarget</i>
DP _{1,4}	✓	✓		✓	✓			
DP _{1,5}	✓	✓		✓	✓			
DP _{1,6}	✓	✓		✓	✓			
DP _{1,7}	✓	✓		✓	✓			
DP ₂	✓	✓						
DP _{3,1}	✓	✓		✓	✓			
DP _{3,2}	✓	✓		✓	✓		X	<i>Specification reasoning rules for SimilarityOfProcessCapabilities</i>
DP _{3,3}	✓	✓						
DP _{3,4}	✓	✓						
DP _{3,5}	✓	✓						
DP _{3,6}	✓	✓		✓				
DP _{4,1}	✓	✓		✓			X	<i>Specification of reasoning rules for Recommendation, enclosesProductCategorySet, overlapsProductCategorySet, partiallyProductCategorySet</i>
PV _{1,1}			✓					
PV _{1,2}			✓			✓		
PV _{1,3}			✓			✓	X	<i>Reasoning and assertion of enclosesSpecificationSet, enclosesSpecificationTarget, subsetOf, RelationOfSpecificationSets, overlapsSpecificationTarget</i>
PV _{1,4}			✓					
PV _{1,5}			✓			✓		
PV _{1,6}			✓			✓		
PV _{1,7}			✓			✓		
PV ₂			✓					
PV _{3,1}			✓			✓		
PV _{3,2}			✓			✓	X	<i>Reasoning and assertion of SimilarityOfProcessCapabilities</i>
PV _{3,3}			✓					
PV _{3,4}			✓					
PV _{3,5}			✓					
PV _{3,6}			✓			✓		
PV _{4,1}			✓			✓	X	<i>Reasoning and assertion of Recommendation, enclosesProductCategorySet, overlapsProductCategorySet, partiallyProductCategorySet</i>

Table 3.25: Usage (✓) of features of the Semantic Web to satisfy technical needs of design parameters and process variables, as well as not fulfilled technical needs (X).

In order to verify the support of the general K-RAMP design by those features, each process variable and each design parameter, as specified in the previous chapters, has to be determined. As the process variables are performed upon the design parameters and the design parameters are addressing the respective functional requirements it is sufficient to show that

1. the specific design using the Semantic Web covers the technical needs of the general design, and
2. the specified information models are supported by the features of the Semantic Web.

It can be determined, that the majority of statements which make up the specification of the design parameters can be classified by the categories S1 to S4. These categories of statements are summarized in Table 3.24. The standardized features of the Semantic Web are able to handle all those statements in general. This perception leads to the assumption that Semantic Web in general can be used to implement the design of K-RAMP. However, there are some significant language constructs missing in OWL2 DL and SWRL with respect to single reasoning rules. Table 3.25 provides therefore a comprehensive overview of all design parameters and process variables in conjunction with Semantic Web features to be utilized. But also the unsupported language constructs are highlighted.

DP_{1,3} respectively PV_{1,3} (see Table 3.13) requires the invocation of universal quantifiers and existence quantifiers which are not supported by Semantic Web to this extent. Moreover, assertion of new individuals, namely individuals of class RelationOfSpecificationSets, needs to be performed as well as counting of an arbitrary number of matching object properties (e.g. enclosesSpecification) in reasoning rules. Implementing those gaps with Jena rules could be one option. However, Jena's OntModel does not yet fully support OWL2.

The ontology models of K-RAMP are specified in accordance to the information models in Chapter 3.2.3 during the second decomposition step. An ontology model is specified for each design parameter and the mapping is shown in Table 3.26. These ontology models represent the TBOX (terminology box) models of K-RAMP and are completely independent of every manufacturing industry.

Design parameter	Ontology model of K-RAMP	Comment
DP _{1,1}	kr-unit	Engineering Units
DP _{1,2}	kr-char	Characteristics
DP _{1,3}	kr-spec	Specifications
DP _{1,4}	kr-prodn-tech	Production techniques
DP _{1,5}	kr-prod	Products
DP _{1,6}	kr-prod-cat	Product categories
DP _{1,7}	kr-proc-cap	Process capabilities
DP ₂	kr-equip	Equipment
DP _{3,1}	kr-proc	Process segments
DP _{3,2}		Process segment setup
DP _{3,3}		Process operations
DP _{3,4}	kr-hd-instr	Handling instruction
DP _{3,5}	kr-eq-recp	Equipment recipes
DP _{3,6}	kr-props	Process operation setups
DP _{4,1}	kr-rec	Recommendations

Table 3.26: Mapping between design parameters and ontology models of K-RAMP.

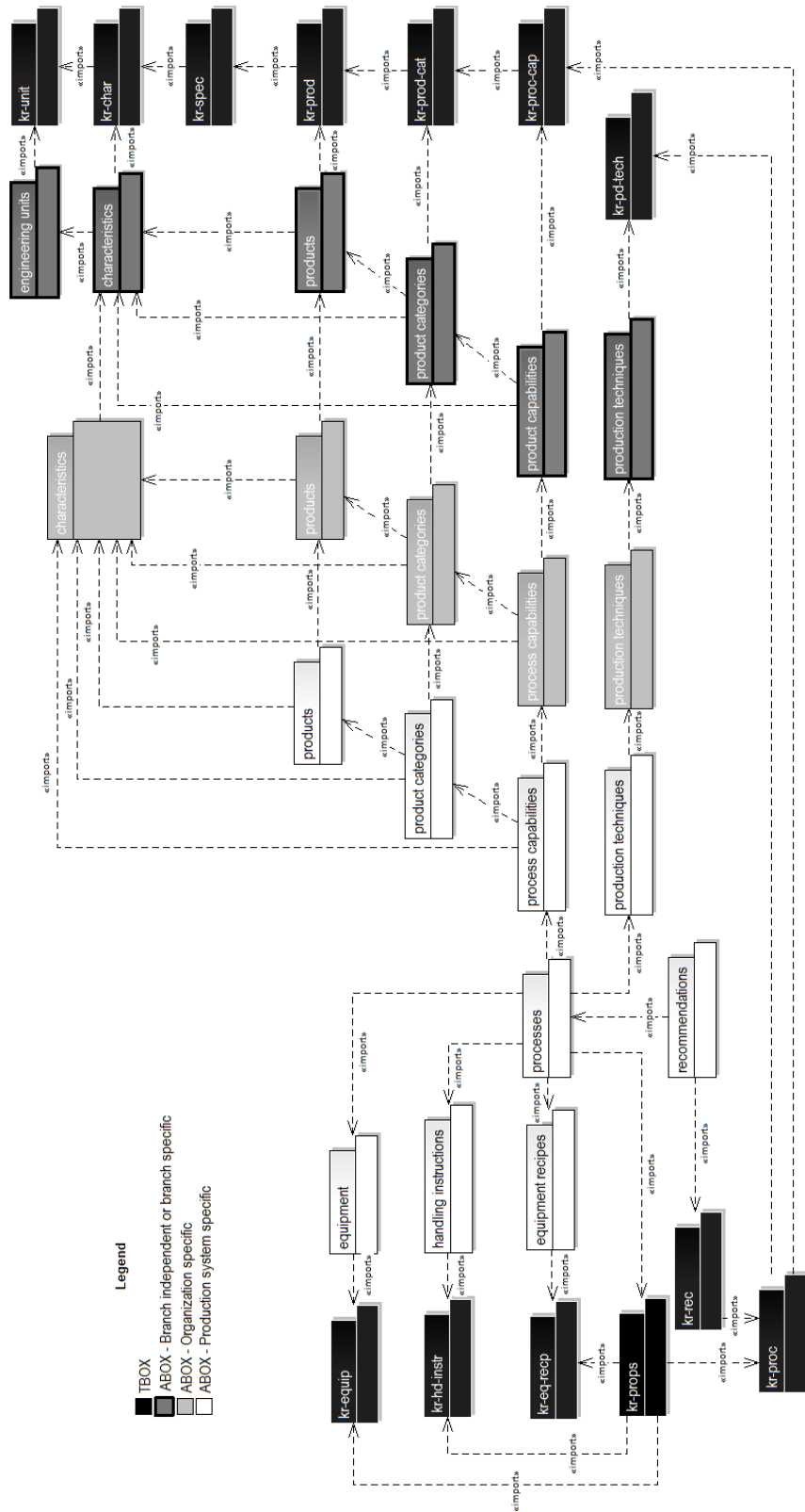


Figure 3.21: A possible structure of TBOX and ABOX ontology models for K-RAMP.

Specific derivatives for particular industries may be specified by ABOX (assertion box) models. These models may also comprise industry-independent terminology (see Chapter 6) for further reuse in additional more specific ABOX models. However, as a first implementation step, such ABOX models may be used within the domain of enterprises or within the domain of directly connected partners. This is useful for the exchange of information between production systems as it is used in the context of a product ramp-up. Moreover, for the same purpose, these models may be used for information exchange between service consumers and service providers. For this reason, the separation in neutral TBOX models and domain-specific ABOX models supports the fulfillment of constraint C_1 .

In Figure 3.21, the recommended structure of the ontology models is shown which shall be based on K-RAMP TBOX models. Branch-specific or general valid ontology models shall be specified on a first layer of derived ABOX models. This step needs to be performed by standardization organizations and is further motivated in Chapter 6.2. More specific ontology models which address the particular purpose of the respective organization or supply chain are derived from the branch-specific layer. It is also possible to specify ontology models for products, product categories, process capabilities or production techniques on the level of each production system. However, it is essential to apply the common set of characteristics.

There are already examples of ontology models with respect to engineering units in place. In order to keep the discussion of this work focused on the topic of product ramp-up, a less complex proprietary ontology model *kr-unit* was introduced.

With respect to production techniques a standardized taxonomy model could be also valuable. However, research for potentially existing work with respect to this topic did not provide any result. Therefore, *k-prodn-tech* is probably the first published approach which covers this topic. It shall be used in Chapter 4 to establish a taxonomy model of an evaluation case study.

For products and product categories there are also some ontology models in place. With UNSPSC [53], the United Nations Standard for Products and Services, a taxonomy model of products and services has been established. The concept of *kr-prod-cat* matches with the concept of the UNSPSC, but while UNSPSC focuses on commerce and thus on a unified classification and description of products and services, *kr-prod-cat* intends to determine similarities between products automatically based on commonly specified characteristics of product categories. However, outside of scope of this work it could be also valuable to merge both ideas in a common ontology model.

Afore mentioned ABOX ontology models are shared between two production systems through two possible approaches. First, it can be assured that the knowledge bases of all production systems are always using the same common ABOX ontology model for engineering units, production techniques, products or product categories. Every modification of these ontology models is performed centrally and published to all involved production systems (*centralized ontology maintenance*). The *centralized ontology maintenance* approach is reasonable for production systems which belong to the same organization.

This approach ensures a commonly applied vocabulary across all production systems within the organization. Secondly, the ABOX ontology models for engineering units, production techniques, products or product categories are maintained separately by each production system, but still based on common TBOX ontology models of K-RAMP (*individual ontology maintenance*). In this case, ambiguous vocabularies across the involved production systems are rather likely. After the transfer of information about engineering units, production techniques, products or product categories from the original production system to the target production system,

algorithms which are published as “ontology matchmaking” or “ontology alignment” [54] must be applied in order to determine equivalences within both vocabularies. As both vocabularies are based on the identical TBOX models, matchmaking is required on the level of individuals only. However, for the purpose of this work the *centralized ontology maintenance* approach is considered.

The overall process about the management and exchange of device-independent information is describe in Figure 3.19. Inside an organization it is useful as a first step to maintain engineering units and characteristics centrally and separately thus ensuring the same vocabulary about this information domain across all production systems (PV_{1,1}, PV_{1,2}). Independently from engineering units and characteristics, production techniques are maintained and exchanged simultaneously (PV_{1,3}). As a second step, products are maintained and exchanged between the original production system and the target production system (PV_{1,5} in conjunction with PV_{1,3}). Product categories are aligned as a third step (PV_{1,6} in conjunction with PV_{1,3}) involving the result of PV_{1,5}. As a forth step, based on common characteristics’ vocabulary, process capabilities are at least partially maintained centrally and shared across both production systems (PV_{1,7} in conjunction with PV_{1,3}).

Organization-specific ABOX models are rather likely specified for products and process capabilities. An ontology model on process capabilities is based on *kr-proc-cap* and comprises process capabilities which are commonly used across an organization and therefore multiple production systems. Such organizations are enterprises of partners in a supply chain. The same is also valid for products, which are specified in an ABOX model as well and thus depend on *kr-prod* with respect to the model structure, on a probably standardized ontology model of product categories (see before) and on the process capabilities of the respective organization.

This leads to the layer of ABOX models, which are specific for a production system. There may be local process capabilities or local products specified as parts of this layer. However, particularly the device-specific information of the production system is comprised in this layer. Not surprisingly, equipments, handling instructions, equipment recipes and a comprehensive ABOX model for processes and control are parts of this layer. It is useful to keep ontology models at least separated to these partial domains, as they are commonly maintained by different groups of experts within a production system.

The implemented structure of TBOX ontology models as well as the layered concept of ABOX ontology models addresses constraint C₁ as it is independent of any industrial sector. But both together provide the foundation of ontology models for specific implementations.

3.4 Design of hybrid components

It is shown in Table 3.25 which parts of the general specification cannot be solved by Semantic Web technologies and thus cannot be specified as integral part of the ontology model. In such situations, alternative technologies are required. This chapter provides a specification how complementary hybrid components are therefore integrated with a Semantic Web knowledge base. The specification of these components is described in a general way in order to avoid specific implementation restrictions. For this reason a possible implementation is provided as pseudo-code.

The implementation of hybrid components in general is disadvantageous. Knowledge, which is bound to such hybrid components, needs to be in place at each knowledge store in order to utilize all features of the underlying ontology model. Consequently, such hybrid components must be distributed separately from the actual ontology. But this is error prone and requires proprietary coding. For this reason, the number of hybrid components is limited to a minimum for K-RAMP.

Process variables (PVs) are implementing procedures. For this reason, all extra coding is associated with one of the PVs where gaps are highlighted in Table 3.25.

In Figure 3.19, $PV_{1,3}$ is in the center of interest as it has to be performed in conjunction with any modifications which are performed for *Products*, *ProductCategories* or *ProcessCapabilities*. In the next sections, the hybrid components of $PV_{1,3}$ are introduced. Table 3.25 provides the overview about the gaps to be closed. Figure 3.19 provides the localization of these hybrid components in the overall context of the K-RAMP process. The first gaps to be covered are with respect to the relationship of *SpecificationTargets* (Figure 3.22). In the sequel of the next sections, it becomes obvious that each invocation of hybrid functions and procedures is followed by standard reasoning of the Semantic Web in order to update the knowledge base's content due to the modifications of the previously executed hybrid functions and procedures.

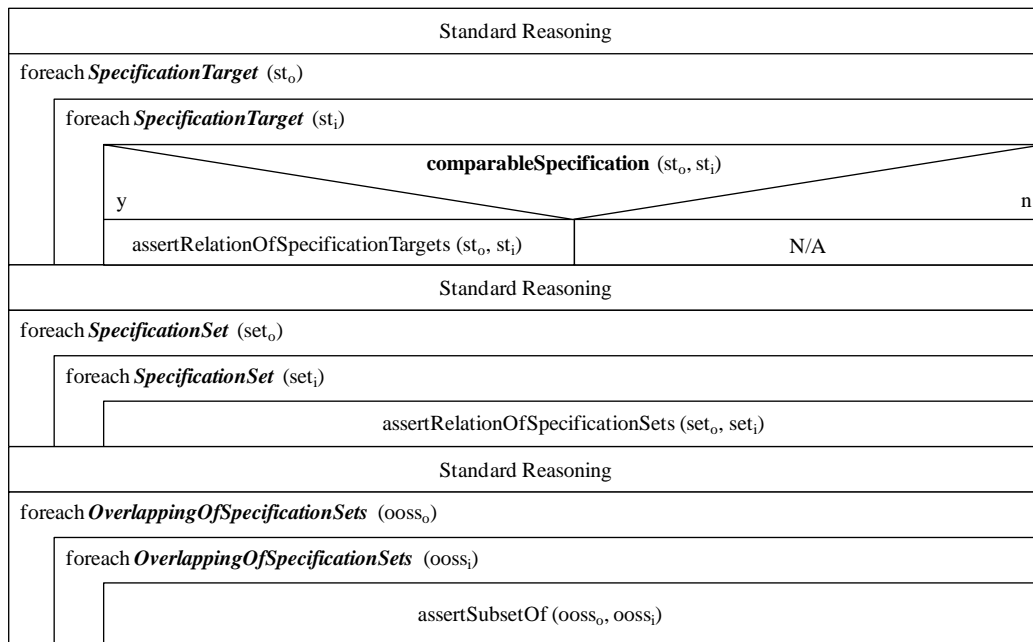


Figure 3.22: Invocation of hybrid functions and procedures to resolve relationships between *SpecificationTargets*, *SpecificationSets* and *RelationOfSpecificationSets*.

$PV_{1,3}$ is enhanced by proprietary procedures accordingly in order to master universal quantifiers and existence quantifiers but also set operators which are needed for reasoning of individuals of concepts as invoked in Figure 3.22. The functions which have to be implemented with respect to $PV_{1,3}$ are

- assertRelationOfSpecificationTargets
- assertRelationOfSpecificationSets
- assertSubsetOf.

The leading ‘assert’ indicates that the procedures also asserts respective individuals and object properties. The procedure `assertRelationSpecificationTargets` is implemented as specified by (3.1). Its purpose is to determine, whether an object property *enclosesSpecificationTarget* or alternatively *overlapsSpecificationTarget* can be asserted between a pair of *SpecificationTargets*. In order to improve the readability, no optimizations of the algorithms are carried out in general. The bold-written functions within the pseudo-code represent

associations – respectively object properties in Semantic Web terminology – between the respective variables.

Initially, two sets of *Things*, namely setX and setY, are assumed as unspecified. The associated sets of *Things* are gathered for both *SpecificationTargets* ②. For this purpose, the results of the object property *hasTarget* are gathered by setting targ1 respectively targ2 as subject and keeping the object setX respectively setY unspecified. As previously mentioned, also the exemplary call of *hasTarget* (targ1, setX) results in a SPARQL query of the form

```
SELECT ?x ?y
WHERE { ?x kr:hasTarget ?y.
      FILTER (?x = <uri of targ1>) }
```

```
procedure assertRelationOfSpecificationTargets (
    SpecificationTarget targ1, targ2)
begin
    set of TargetValue setX := unspecified;           // (1)
    set of TargetValue setY := unspecified;
    TargetValue x, y, c := unspecified;
    integer cntEncloses, cntOverlaps := 0;
    hasTarget (targ1, setX);                           // (2)
    hasTarget (targ2, setY);
    foreach x in setX do                                 // (3)
        c := unspecified;
        foreach y in setY do
            if x = y or generalizes (x, y) <> empty then
                cntEncloses := cntEncloses + 1;
                exit;
            elseif (generalizes (c, x) <> empty and
                   generalizes (c, y) <> empty) or
                   specializes (x, y) then
                cntOverlaps := cntOverlaps + 1;
                exit;
            end;
        end;
    end;
    if cntEncloses = cardinality (setX) then           // (4)
        assert (enclosesSpecificationTarget, targ1, targ2);
    elseif cntEncloses + cntOverlaps > 0 then
        assert (overlapsSpecificationTarget, targ1, targ2);
    end;
end.
```

(3.1)

Due to the unspecified object, there is also no filter set in the SPARQL query. The SPARQL query results in a list of all resulting tuples (x, y) of the function *hasTarget*. Moreover, the unspecified setX is initialized with a set of all resulting ys thus all resulting individuals of *TargetValue*. Determining the count of *TargetValues* of setX which are identical with or generalizing and thus enclosing *TargetValues* of setY, is performed next ③. As an alternative, it is determined whether *TargetValues* of setX are specializing *TargetValues* of setY or whether both have the same generalization. If the cardinality of setX is equal to the counted enclosing *TargetValues* targ1 encloses targ2. Alternatively, if the summary of counted enclosing and of counted overlapping *TargetValues* is greater than 0 targ1 overlaps targ2 ⑤. Otherwise, there is no relation between both *SpecificationTargets*.

The implementation of `assertRelationOfSpecificationTargets` is straight forward in accordance to the specification of $DP_{1,3}$ in Table 3.13. During the execution of (3.1), a function `assert` is applied which also invokes a SPARQL query with the structure

```
construct { st :enclosesSpecificationTarget st1 }
```

The procedure `assertRelationOfSpecificationSets` (3.2) determines whether from two given *SpecificationSets* `set1` and `set2`, `set1` encloses, overlaps or partially overlaps `set2`. The procedure gathers the *Specifications* ① of a pair of given *SpecificationSets*. The *Specifications* of both sets are verified pairwise whether the *Specification* of `set1` encloses ② or overlaps the *Specification* of `set2` ③. The decision logic follows the specification of $DP_{1,3}$ in Table 3.13. This algorithm is based on some premises which are requested by the underlying information model.

1. Every *Characteristic* is only referred once within the context of a *SpecificationSet*. Because the usage of the same *Characteristics* is prerequisite for being comparable and thus for encloses or overlaps, there can be maximally one pair of *Specifications* in two given *SpecificationSets* which satisfy those conditions. Multiple counting is omitted.
2. *Specifications* are assigned to exactly one *SpecificationSet*. Therefore, it is omitted that the same *Specification* is compared to itself out of two given *SpecificationSets*.

Based on these premises, the occurrence of the relations *encloses* and *overlaps* are counted in this chronological sequence through pair-wise comparison. Moreover, all *Characteristics* which are commonly used between both *SpecificationSets* are collected in `setChar`. These actions result in a summation $cntEncloses + cntOverlaps \leq |setChar| \leq |set1|$. There are three matchmaking scenarios to be encountered for pairs of *SpecificationSets*.

1. **Matchmaking scenario 1 of *SpecificationSets***: If `set1` encloses `set2`, each *Specification* of `set1` encloses a *Specification* of `set2`. The pair-wise comparison results in $0 < cntEncloses = |setChar| = |set1|$ and $cntOverlaps = 0$. The *ratioEncloses* therefore results in $cntEncloses / |set1| = 100$ and *ratioCommonCharacteristics* in $|setChar| / |set1| = 100$ as well, which matches the specification of $DP_{1,3}$.
2. **Matchmaking scenario 2 of *SpecificationSets***: If `set1` overlaps `set2`, which means a mixture of encloses and overlaps of all comprised *Specifications*, the pair-wise comparison results in $0 < cntEncloses + cntOverlaps = |setChar| = |set1|$ with $cntEncloses > 0 \wedge cntOverlaps > 0$. The $cntEncloses < |setChar|$ accordingly. This leads to $ratioEncloses < 100 \wedge ratioCommonCharacteristics = 100$ (see specification of $DP_{1,3}$).
3. **Matchmaking scenario 3 of *SpecificationSets***: If `set1` partially overlaps `set2`, there is a mixture of encloses and overlaps for a subset of *Specifications* of `set1` with respect to *Specifications* of `set2`. But there is also a remaining set of *Specifications* which are either comparable or not comparable with *Specifications* of `set2`. In such a situation, the pair-wise comparison results in constraints $0 < cntEncloses + cntOverlaps < |setChar| < |set1|$ as well as $0 \leq cntEncloses < |set1|$ and $0 \leq cntOverlaps < |set1|$. These constraints lead to $0 < ratioEncloses < 100$ and $0 < ratioCommonCharacteristics < 100$.

Finally, an individual of *RelationOfSpecificationSets* is asserted in accordance to the information model of DP_{1,3} and the specified calculation results of *ratioEncloses* and *ratioCommonCharacteristics* ⑤.

```

procedure assertRelationOfSpecificationSets (
                                SpecificationSet set1, set2)
begin
  integer cntOverlaps := 0;
  integer cntEncloses := 0;
  set of Specification X := unspecified;
  set of Specification Y := unspecified;
  set of Characteristic setChar := empty;
  Characteristic curChar := unspecified;
  RelationOfSpecificationSets ross := unspecified;
  comprisesSpecification (set1, X);
  comprisesSpecification (set2, Y);
  foreach x in X do
    foreach y in Y do
      if encloses (x, y) <> empty then // (2)
        cntEncloses := cntEncloses + 1;
        onCharacteristic (x, curChar);
        setChar := setChar + curChar;
        exit;
      elseif overlaps (x, y) then // (3)
        cntOverlaps := cntOverlaps + 1;
        onCharacteristic (x, curChar);
        setChar := setChar + curChar;
        exit; // (3.2)
      elseif comparable (x, y) then // (4)
        onCharacteristic (x, curChar);
        setChar := setChar + curChar;
        exit;
      end;
    end;
  curChar := unspecified;
end;

if cntEncloses + cntOverlaps > 0 then // (5)
  assert (RelationOfSpecificationSets, ross);
  assert (fromSpecificationSet, ross, set1);
  assert (toSpecificationSet, ross, set2);
  assert (commonCharacteristic, ross, setChar);
  assert (ratioEnclosing, ross,
          cntEncloses / cardinality (set1) * 100);
  assert (ratioCommonCharacteristics, ross,
          cardinality (setChar) / cardinality (set1) * 100);

end;
end.

```

The condition `assertSubSetOf` (3.3) is applied on a given pair of individuals of *RelationOfSpecificationSet* and behaves in accordance to the specification of DP_{1,3} in Table 3.13. For two individual *RelationOfSpecificationSets* which are passed as arguments, the sets of common *Characteristics* are gathered (*commonCharacteristics*) ①.

In Figure 3.20, there is PV_{3,2} for *Process Segment Setup modeling*. Due to this modeling activity, it is also possible to determine similarities between *Process Capabilities* as already discussed in the context of DP_{3,2} in Table 3.14. In the next sections the hybrid component of PV_{3,2} is introduced accordingly. Table 3.25 provides the overview about the gap to be closed. Figure 3.20

provides the localization of the hybrid component in the overall context of the K-RAMP process. The detailed structure of this component is shown in Figure 3.23.

```

procedure assertSubSetOf (RelationOfSpecificationSets ross1, ross2)
begin
  set of Characteristic setX := unspecified;
  set of Characteristic setY := unspecified;
  commonCharacteristic (ross1, setX); // (1)
  commonCharacteristic (ross2, setY); // (2)
  if subset (setX, setY) then
    assert (subsetOf, ross1, ross2);
  end;
end.

```

(3.3)

The procedure `assertSimilarityOfProcessCapabilities` (3.4) asserts an individual of *SimilarityOfProcessCapabilities* due to the occurrence of an existing object property *similarProcessCapability* ①. The implication rule can be described by means of SWRL. However, for further use in conjunction with DP_{4,1} the access to the *ProductionTechnique* is needed which cannot be provided by *similarProcessCapability*.

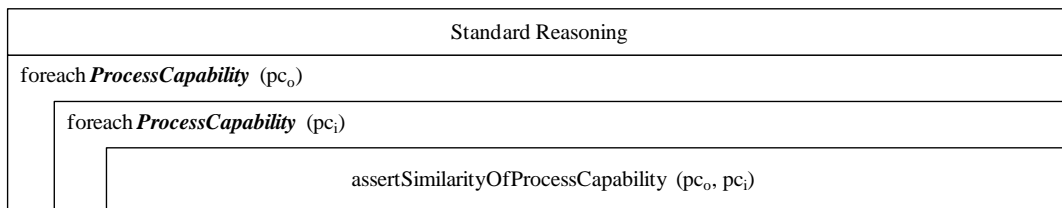


Figure 3.23: Invocation of hybrid functions and procedures to assert *SimilarityOfProcessCapability*.

Because of the precondition that both *ProcessCapabilities* are already asserted as similar, it is sufficient to retrieve the *ProductionTechnique* through one of the passed *ProcessCapabilities* by resolving the object property chain *enabledBy* → *onProcessSegment* → *isAProductionTechnique*. It is assumed that functions *intersection* is provided by the applied framework in accordance to the definition of the respective operator of set theory ②. As a final step, the individual of *SimilarityOfProcessCapability* is asserted in conjunction with it additional object properties and according to the information model of DP_{3,2} ③.

```

procedure assertSimilarityOfProcessCapabilities (
ProcessCapability pc1, pc2)
begin
  set of ProductionTechnique pq1 := unspecified;
  set of ProductionTechnique pq2 := unspecified;
  set of ProductionTechnique inter := unspecified;
  if similarProcessCapability (pc1, pc2) <> empty then // (1)
    supportedBy (pc1, pq1); // (2)
    supportedBy (pc2, pq2); // (3)
    inter := intersection (pq1, pq2);
  end;
  assert (SimilarityOfProcessCapability, sopc); // (4)
  assert (onProductionTechnique, inter);
  assert (onProcessCapability, pc1);
  assert (onProcessCapability, pc2);
end.

```

(3.4)

PV_{4,1} – the *Recommendation* – is localized within PV₄ and according to Figure 3.18. In the next sections the hybrid components of PV_{4,1} are introduced. Table 3.25 provides the overview

about the gaps to be closed. Figure 3.24 shows the detailed structure of the hybrid component to be implemented. Consequently, the functions which have to be implemented with respect to PV_{4,1} are

- assertProductCategorySet
- assertRecommendation

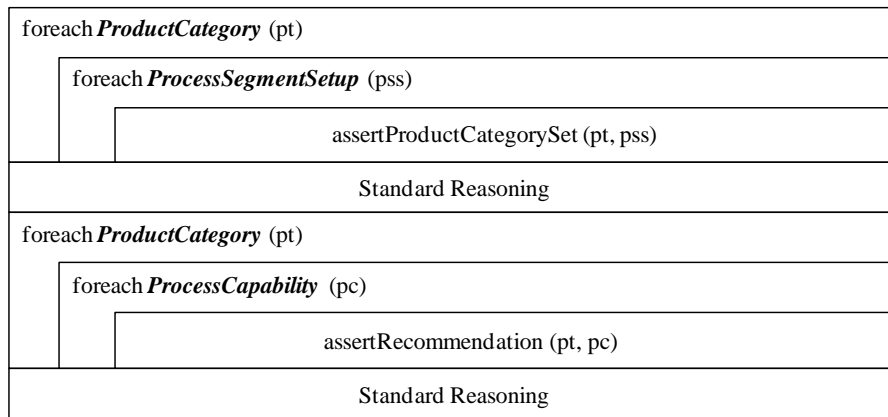


Figure 3.24: Invocation of hybrid functions and procedures to resolve relationships between ProductCategorySets and for assertion of Recommendations.

```

procedure assertProductCategorySet (ProductCategorySet set1, set2)
begin
  set of ProductCategory setPT1 := unspecified;
  set of ProductCategory setPT2 := unspecified;
  ProductCategory pt1, pt2 := unspecified;
  array of integer matrix := empty;
  array of integer diag := empty;

  hasProductCategory (set1, setPT1); // (1)
  hasProductCategory (set2, setPT2);
  foreach pt1 in setPT1 do // (2)
    foreach pt2 in setPT2 do
      if encloses (pt1, pt2) <> empty and
        matrix [pt1, pt2] := 3;
      elseif overlaps (pt1, pt2) <> empty then
        matrix [pt1, pt2] := 2;
      elseif partially (pt1, pt2) <> empty then // (3)
        matrix [pt1, pt2] := 1;
      else
        matrix [pt1, pt2] := 0;
      end;
    end;
  end;
  maximizeTrace (matrix);
  diag := mainDiagonal (matrix);
  if count (diag, 3) = size (diag) then // (3)
    assert (enclosesProductCategorySet, set1, set2);
  elseif count (diag, 2) + count (diag, 3) = size (diag) then
    assert (overlapsProductCategorySet, set1, set2);
  elseif count (diag, 0) < size (diag) then
    assert (partiallyProductCategorySet, set1, set2);
  end;
end.

```

The procedure `assertProductCategorySet` (3.5) expects two *ProductCategorySets*. The procedure implements decisions which are specified by $DP_{4,1}$ (see Table 3.18). The result of `assertProductCategorySet` is either an asserted relation *enclosesProductCategorySet*, *overlapsProductCategorySet*, *partiallyProductCategorySet* or no assertion, depending on determined relations between the members of respectively compared *ProductCategorySets*. Recalling the previous discussion about this topic, the two *ProductCategorySets* are a *ProductCategory* as the first argument and a *ProcessSegmentSetup* as the second argument. However, for keeping the implementation as generic as possible, the common superior concept *ProductCategorySet* is used.

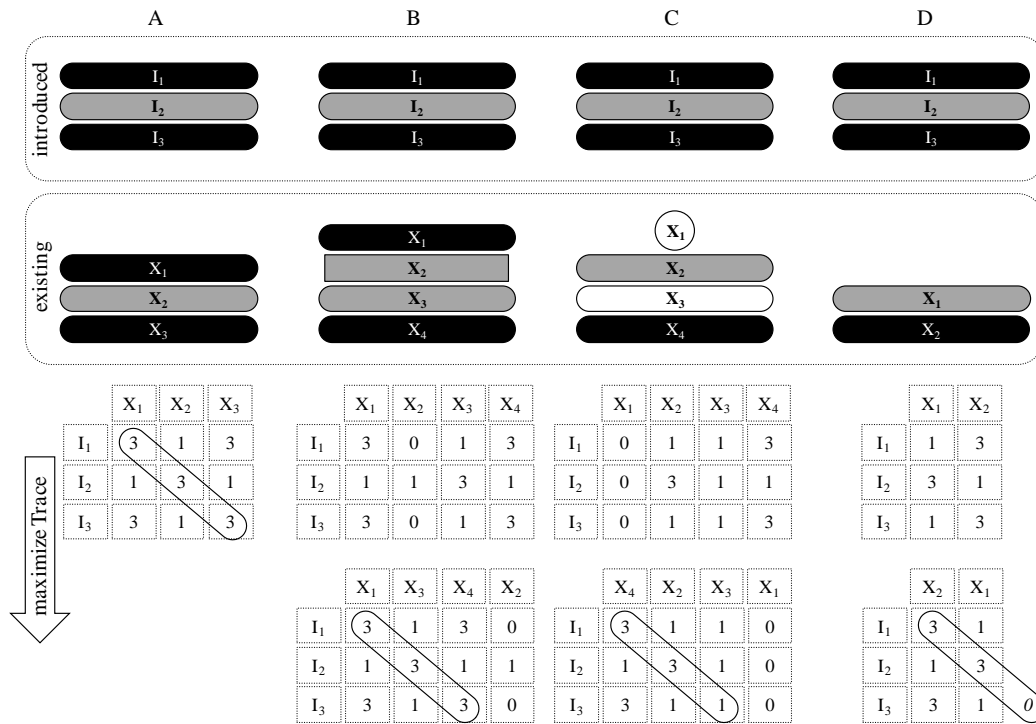


Figure 3.25: Exemplary cases of related *ProductCategorySets*.

The procedure gathers the *ProductCategories* of both sets ①. A nested loop determines the relation between two respective members and initializes a comparison matrix ②. Some examples of possible comparison matrices are shown in Figure 3.25. Example A represents two *ProductCategorySets* where each member of the introduced set encloses at least one member of the existing set. Both sets have the same cardinality. The resulting comparison matrix is squared and its main diagonal is populated with 3 (for encloses relations). Example B is almost similar. However, there is an additional member comprised by the existing set which is not enclosed by any of the members of the introduced set. This member is shifted to the right edge of the comparison matrix while the trace of the matrix is maximized. The existing *ProductCategorySet* of example C comprises a member which is not enclosed, nor overlapped or even partially overlapped by any member of the introduced set. Moreover, only either I_1 or I_3 are determined to enclose the respectively fitting member of the existing set (I_1 in the figure). As best alternative option X_3 is chosen in this example as it overlaps at least partially with I_3 . In example D, there is no counterpart for I_3 because the existing *ProductCategorySet*'s cardinality is smaller than the cardinality of the introduced set. The remaining elements of the main diagonal of a squared matrix are considered 0 (no relation) in such a case.

```

procedure assertRecommendation (
    ProductCategory pti, ProcessCapability pcx)
begin
    set of ProductCategory setPTx := unspecified;
    set of ProcessSegmentSetup setPSS := unspecified;
    set of ProcessCapability setPCi := unspecified;
    ProductCategory ptx := unspecified;
    ProcessSegmentSetup pss := unspecified;
    ProcessCapability pci := unspecified;
    SimilarityOfProcessCapabilities sopc := unspecified;
    RelationOfSpecificationSets ross1, ross2 := unspecified;
    ProductionTechnique pq := unspecified;

    requires (setPTx, pcx); // (1)
    requires (pti, setPCi)
    enabledBy (pcx, setPSS);
    foreach ptx in setPTx do
        foreach pci in setPCi do
            foreach pss in setPSS do
                if enclosesProductCategorySet (pti, pss) <> empty and
                    (encloses (pci, pcx) <> empty or
                     encloses (pti, ptx) <> empty) then
                    assert (Recommendation, rc); // (2)
                    assert (forProductCategory, rc, pti);
                    assert (substitutes, rc, pci);
                    assert (withProcessSegmentSetup, rc, pss);
                    assert (penalty, rc, 0);
                elseif encloses (pci, pcx) <> empty and
                    (overlapsProductCategorySet (pti, pss) <> empty or
                     partiallyProductCategorySet (pti, pss) <> empty) then
                    assert (Recommendation, rc); // (3)
                    assert (forProductCategory, rc, pti);
                    assert (substitutes, rc, pci);
                    assert (withProcessSegmentSetup, rc, pss);
                    assert (useProductCategorySet, rc, pti);
                    assert (penalty, rc, 1);
                elseif overlaps (pci, pcx) <> empty and
                    enclosesProductCategorySet (pti, pss) <> empty and
                    (overlaps (pti, ptx) <> empty or
                     partially (pti, ptx) <> empty) then
                    assert (Recommendation, rc); // (4)
                    assert (forProductCategory, rc, pti);
                    assert (substitutes, rc, pci);
                    assert (withProcessSegmentSetup, rc, pss);
                    assert (adjustParametersOf, rc, pss);
                    assert (penalty, rc, 2);
                elseif overlaps (pci, pcx) <> empty and
                    (overlapsProductCategorySet (pti, pss) <> empty or
                     partiallyProductCategorySet (pti, pss) <> empty)
                    and
                    (overlaps (pti, ptx) <> empty or
                     partially (pti, ptx) <> empty) then
                    assert (Recommendation, rc); // (5)
                    assert (forProductCategory, rc, pti);
                    assert (substitutes, rc, pci);
                    assert (withProcessSegmentSetup, rc, pss);
                    assert (useProductCategorySet, rc, pti);
                    assert (adjustParametersOf, rc, pss);
                    assert (penalty, rc, 3);
                elseif partially (pci, pcx) <> empty and
                    enclosesProductCategorySet (pti, pss) <> empty and
                    (overlaps (pti, ptx) <> empty or

```



```

        partially (pti, ptx) <> empty) then
if similarProcessCapability (pci, pcx) <> empty then
onProcessCapability (sopc, pci + pcx);
onProductionTechnique (sopc, pq);
(fromSpecificationSet + toSpecificationSet) (ross1,
        pci + pcx);
(fromSpecificationSet + toSpecificationSet) (ross2,
        pti + ptx);
if subsetOf (ross1, ross2) then
assert (Recommendation, rc); // (6)
assert (forProductCategory, rc, pti);
assert (substitutes, rc, pci);
assert (withProcessSegmentSetup, rc, pss);
assert (adjustParametersOf, rc, pss);
assert (onSimilarityTo, rc, pci);
assert (similarDueTo, rc, pq);
assert (penalty, rc, 6);
end;
end;
elseif partially (pci, pcx) <> empty and
(overlapsProductCategorySet (pti, pss) <> empty or
partiallyProductCategorySet (pti, pss) <> empty)
and (overlaps (pti, ptx) <> empty or
partially (pti, ptx) <> empty) then
if similarProcessCapability (pci, pcx) <> empty then
onProcessCapability (sopc, pci + pcx);
onProductionTechnique (sopc, pq);
(fromSpecificationSet + toSpecificationSet) (ross1,
        pci + pcx);
(fromSpecificationSet + toSpecificationSet) (ross2,
        pti + ptx);
if subsetOf (ross1, ross2) then
assert (Recommendation, rc); // (7)
assert (forProductCategory, rc, pti);
assert (substitutes, rc, pci);
assert (withProcessSegmentSetup, rc, pss);
assert (modify, rc, pss);
assert (useProductCategorySet, rc, pti);
assert (adjustParametersOf, rc, pss);
assert (onSimilarityTo, rc, pci);
assert (similarDueTo, rc, pq);
assert (penalty, rc, 15);
end;
end;
end;
end;
end;
end;
end.

```

The result of assertProductCategorySet is determined as follows:

1. if there are exactly as many occurrences of *encloses* as there are *ProductCategories* in each set the relation is *enclosesProductCategorySet* – all members of the main diagonal are populated with 3,
2. the relation is *overlapsProductCategorySet* if 1. is not fulfilled and the main diagonal is populated exclusively with 3 or 2 (overlaps),

3. and the relation is *partiallyProductCategorySet* if 2. is not fulfilled and the main diagonal is populated with 1 (partially overlapped), or it is populated with less 0 (no relation) than the cardinality of the introduced set,
4. otherwise – the main diagonal is populated with 0 only – there is no relation asserted between the given sets.

The procedure `assertRecommendation` uses an introduced *ProductCategory* and an existing *ProcessCapability* as input and determines the validity respectively asserts a *Recommendation* for linking both entities (3.6). It determines the validity of implication rules which are specified by $DP_{4,1}$ (see Table 3.18). If all conditions are met, assertions of an individual of *Recommendation* and the respectively necessary object properties are performed. The applied conditions are derived from Table 3.17. There are as many conditions as there are different levels of penalty.

3.5 Architecture overview

The previously discussed design on the level of TBOX and ABOX ontology models as well as the hybrid components together result in an overall architecture of a *K-RAMP knowledge base* which is spread across multiple production systems. Individual knowledge stores, which reside at each production system, are connected through *intranet or internet communication* technologies. This architecture is shown in Figure 3.26.

The architectural center of each *K-RAMP knowledge store* is a *Semantic Web database*. The structure of this database is derived from a structure of dependencies which is similar to the one being discussed in the context of Figure 3.21. The *K-RAMP knowledge store* comprises this *Semantic Web database* and attached hybrid components for *enhances reasoning and asserting*, which are specified in Chapter 3.4. The hybrid components are interacting with the *Semantic Web database* by utilizing *SPARQL* as also already discussed in the previous chapter.

For the purpose of information exchange the *K-RAMP knowledge stores* across all production systems are connected via the internet or intranet through pairs of one *inbound gateway* (incoming information) and one *outbound gateway* (outgoing information). These gateways use *SPARQL* for the interaction with the *Semantic Web database* of the local *K-RAMP knowledge store*.

In order to integrate the local *K-RAMP knowledge stores* with the local production-IT, namely the local *MES*, two complementary application programming interfaces (APIs) are applied for reading respectively for writing of information (*incoming API*, *outgoing API*). Again *SPARQL* is used for the interaction with the *Semantic Web database*. The *MES* or equivalent software is the primary source and target of information at a particular production system. First, it uploads *Product Basic Data* which are converted to the information models being primarily derived from DP_1 and its decomposition. Secondly, it uploads *Process Basic Data* which are converted to the information models being derived from DP_2 and DP_3 and again its decomposition.

The ramp-up team interacts through a user interface (*Ramp-up UI*) with the local *K-RAMP knowledge store*. By means of the *Ramp-Up UI* the ramp-up team is able to gather *Recommendations*. After the recommended activities are performed the initial updates of the linkage between *Product Categories* and *Process Capabilities* or with respect to the structure of *Process Segments* are performed in the local *MES*. Again from the local *MES* the updated information is fed back to the *K-RAMP knowledge store*. The local *MES* thus remains the master of the product basic data and the process basic data.

Due to the standardization of the applied Semantic Web features and the exclusive use of *SPARQL* for information exchange, each production system may use its own off-the-shelf software product with respect to the *Semantic Web database*.

The specific implementation of gateways and APIs is beneficial for overcoming some still unresolved domains of the Semantic Web.

- Security of information access and information update needs to be handled by local means of the respective production-IT and separated from the technical domain of the Semantic Web. However, this part is not covered in more detail in this work.
- Events are needed for the notification of hybrid components in order to perform their proprietarily implemented activities. Also this topic is not covered by this work but highlighted as a need of a comprehensive architecture.

The focus of this work is on the Semantic Web database and the hybrid specifications of some process variables which invoke the component *Enhanced Reasoning and Asserting*. The gateways and APIs as well as the K-RAMP UI are not discussed in more detail. This focus also limits the scope of the evaluation which is described in the next chapter.

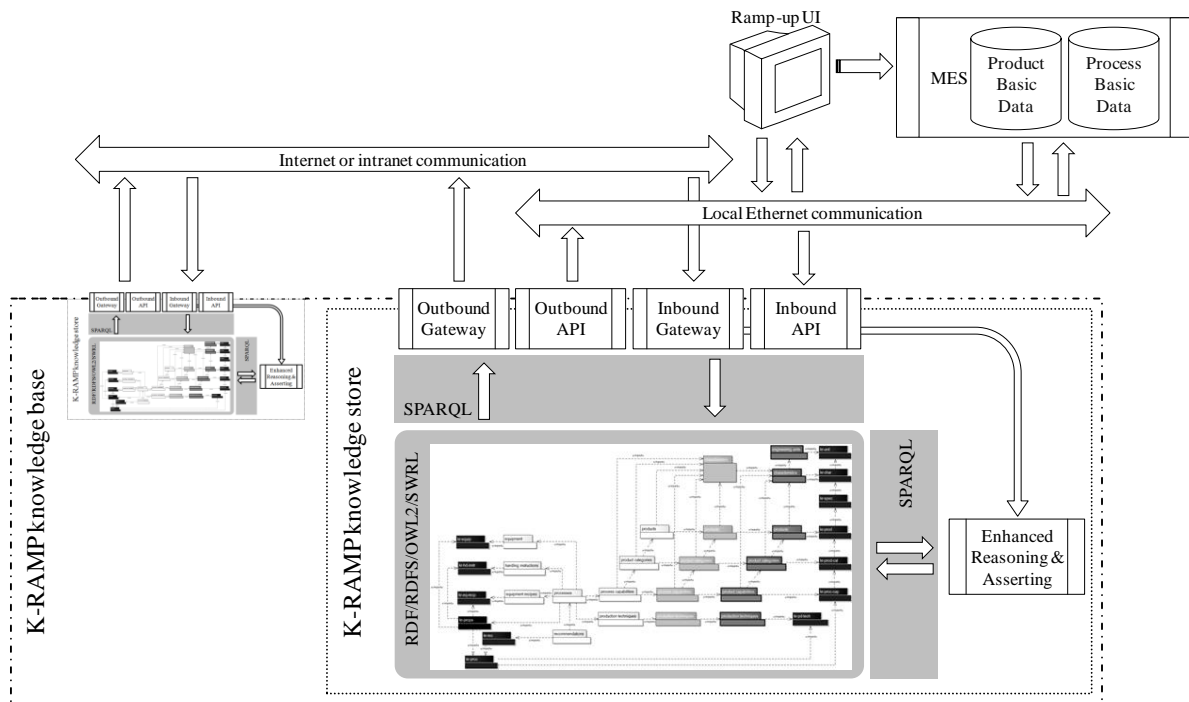


Figure 3.26: Architectural overview of the K-RAMP knowledge base at a single production system.

3.6 Summary

Through the utilization of axiomatic design, the research questions as customer needs are mapped to functional requirements. Further, the functional requirements are converted to design parameters – the information model – and to process variables – the procedures (Objective 1). It is shown that this approach leads to a complete design in order to satisfy (Question 1).

The important design aspects of the K-RAMP information model are a clear separation into a product-related information domain and a process-related information domain. The process-related information domain is further separated into a process-related device-independent information domain and a process-related device-dependent information domain. The product-related information domain and the process-related device-independent information domain are designed to be shared between production systems. The core of these information domains is the information model of *Specifications*.

The information model of *Specifications* comprises means for a structured description of *Products*, *Product Categories* and *Process Capabilities*. Moreover, *Specifications* introduces a set of implications rules in order to determine relationships with graded quality (encloses, overlaps, overlaps partially) between individual *Products*, *Product Categories* or *Process Capabilities* but also between *Product Categories* and *Products* as well as between *Products* or *Product Categories* and *Process Capabilities*. Based on these graded relationships it is possible to perform matchmaking between information models of production systems. Due to the information model and the logic of implication rules also generalizations respectively specializations of concepts can be considered.

Another focus of the process-related device-specific information model is on the implication of relationships between introduced new *Products*, *Product Categories* or *Process Capabilities* and their existing counterparts. The previously mentioned relationships encloses, overlaps and overlaps partially are applied and enhanced by additional rules for this purpose.

In the further course of the preceding chapters, it is shown that particular aspects of the information model are not covered by OWL DL and thus related specifications of the Semantic Web. These aspects are in particular caused by the need for predicate logic with universal quantifiers and existential quantifiers.

Consequently, hybrid components are designed in order to cover partitions of the designed information models which cannot be covered solely by specifications of the Semantic Web. This part of the conceptual phase results in TBOX ontology models of K-RAMP which are mapped to the designed K-RAMP information models and hybrid components which are imperatively implemented. The utilization of ontology models and thus the utilization of the Semantic Web are maximized in this approach. Through this part of conceptual phase, (Objective 2) as well as (Objective 2.1) and (Objective 2.2) are answered. There is only a hybrid approach possible in order to implement the requested knowledge base which answers (Question 2), (Question 3) and (Question 4).

Finally, a hybrid architecture is presented which provides a possible approach in order to implement the K-RAMP information model and the K-RAMP process in a production system (K-RAMP knowledge store). It is also highlighted that the existing MES is integrated with this knowledge store by still keeping the role of a master with respect to the management of product basic data and process basic data.

4 Transfer of a cake recipe

4.1 Overview

This chapter comprises the evaluation of K-RAMP. The evaluation of K-RAMP shall prove that the implementation of the design parameters and process variables by the K-RAMP ontology models in conjunction with the specified hybrid components are able to satisfy the functional requirements. For the purpose of evaluation, a case study is applied which is easy understandable for a majority of readers and is still covering the challenging situations of a real product ramp-up. Therefore, two bakeries are considered. The first, the original bakery, bakes a famous Viennese chocolate cake. The second, the target bakery, is already capable to bake two different cakes. The baking processes of the two target bakery's cakes have partial capabilities to bake the Viennese chocolate cake as well. These capabilities are well chosen in order to demonstrate each design aspect of K-RAMP concerning the reuse of existing production knowledge of the target bakery. The full recipes of all three cakes can be reviewed in Appendix 7.1.

In Chapter 4.2 the evaluation plan is described and how it focuses on the objectives as specified in Chapter 1.4. Chapter 4.3 discusses the setup of the evaluation environment and is followed by Chapter 4.4, where the ABOX ontology models of the cake-baking case study are specified. This chapter provides a good insight about the preparation of *Product Basic Data* and *Process Basic Data* (see Chapter 3.5) which need to be provided by the production-IT outside of the K-RAMP scope. The execution of test cases and their results are discussed in Chapter 4.5.

4.2 Evaluation plan

The evaluation process is embedded in the overall research process (see Figure 1.4). Matchmaking between both sources of knowledge, as shown in the context of the design phase in Chapter 3, is split to a considerable set of matchmaking scenarios (Figure 4.2). A test plan is specified ahead of the evaluation. The comprised test cases specify the expected results with respect to essential combinations of knowledge of both sources. The matchmaking scenarios as derived from the *General Design* are the source for specifying these test cases (*Test Case Specification*). The invocation of all determined matchmaking scenarios is forced by the selected structure of the exemplary information models (test data) of the case study. This structure of the *Created Exemplary Information Model* is therefore driven by the test cases.

Accordingly, the exemplary design knowledge of products and the production knowledge are specified in ABOX ontology models upon the TBOX ontology models which are provided by K-

RAMP. As matchmaking maximally applies standard reasoning of the Semantic Web, Protégé 5.0.0 beta-17 including the Pellet Reasoner Plug-in 2.2.0 are applied as part of the *Created Evaluation Environment* (see also Chapter 4.3). During the execution of the test plan (*Execution of Evaluation*), the reasoning results are verified manually. It is verified whether the expected results are met according to the results which are specified for each test case. The hybrid components of K-RAMP are formally verified and the outcome of this formal verification is loaded manually to the knowledge store of Protégé for verification of Pellet’s reasoning results. The evaluation is considered as being passed successfully, if the matchmaking process results in correct recommendations for each individual test case. The successful passing of the evaluation results in achieved objectives of this work and thus in the answering of the research questions.

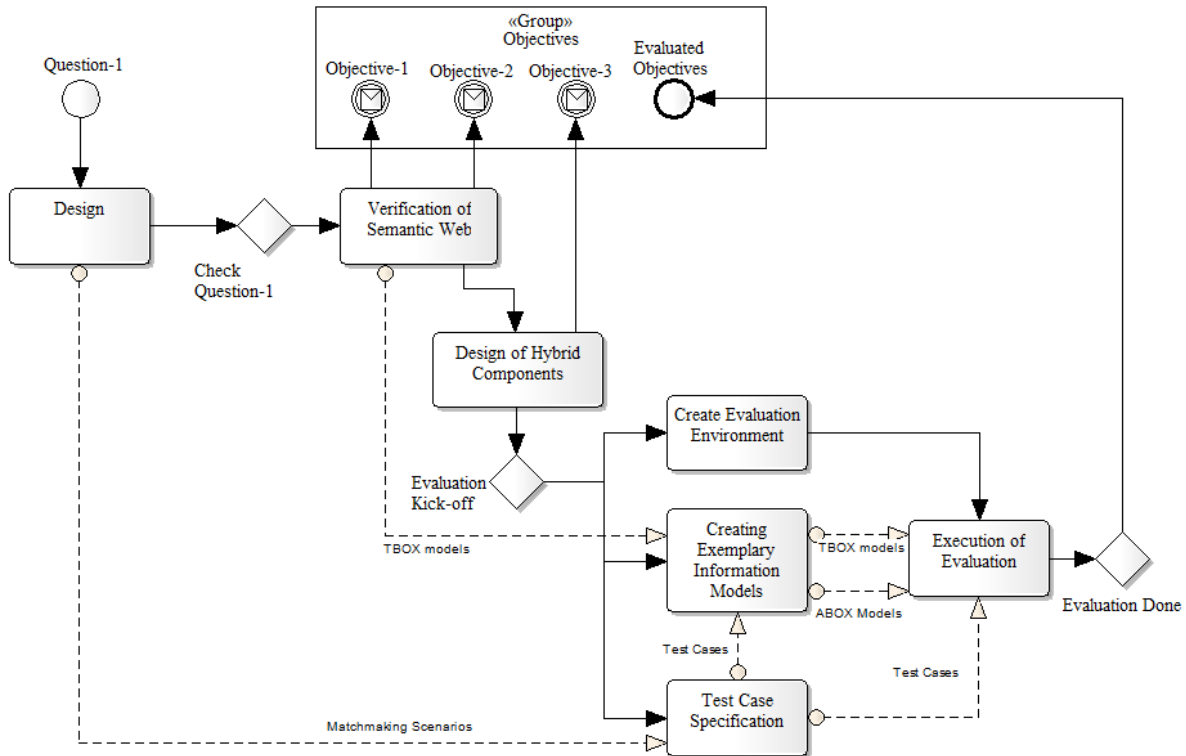


Figure 4.1: Overview of activities for design and evaluation and their association with objectives.

The evaluation plan comprises the test cases to be executed as part of the *Execution of Evaluation*. These test cases are addressing particularly those scenarios which invoke implication rules and assertions of standard Semantic Web database procedures or algorithms of hybrid components. It has to be verified whether

- the implication rules are specified correctly and the underlying information model is complete, and
- the algorithms which shall be implemented by hybrid components are specified correctly.

However, the evaluation plan puts the general purpose of the evaluation to a more comprehensive structure. The evaluation plan follows the sequence of the K-RAMP process as specified in Chapter 3.2.5. The major test groups and subtest groups to be established in accordance with the K-RAMP process are defined in a sequential order as

1. **Test group 1:** Modeling of device-independent information
 - i. Modeling of *ProductionTechniques* (Chapter 4.4.1.2)
 - ii. Modeling of *EngineeringUnits* and *Characteristics* (Chapter 4.4.1.3)
 - iii. Modeling of *Products* and *ProductCategories* (Chapters 4.4.1.4)
 - iv. Modeling of *ProcessCapabilities* (Chapter 4.4.2.2)
2. **Test group 2:** Modeling of *Equipment* and device-dependent information
 - i. Modeling of *ProcessSegments* (Chapter 4.4.2.3)
 - ii. Modeling of *ProcessSegmentSetups* (Chapter 4.4.2.4)
 - iii. Modeling of *Equipment* (Chapter 4.4.3.2)
 - iv. Modeling of *EquipmentRecipes* and *HandlingInstructions* (Chapter 4.4.3.3)
 - v. Modeling of *ProcessOperations* and *ProcessOperationSetup* (Chapter 4.4.3.4)
3. **Test group 3:** Assertion of *ProductCategorySets* and *Recommendations*
 - i. Assertion of *ProductCategorySets* (Chapter 4.5.2)
 - ii. Assertion of *Recommendations* (Chapter 4.5.3)

For each subtest group, a set of evaluation activities has to be performed completely or partially. The completeness of execution of this activity set depends on the complexity of the evaluated model. The complete set comprises the following evaluation activities.

- The ability to prepare the appropriate product basic data and process basic data is performed in order to determine a useful structure for the upload of product basic data and process basic data from the production-IT to the knowledge store. This evaluation activity is utilized to determine premises with respect to the structure of such data. These premises are expected to be fulfilled by the production-IT of the respective production system.
- The generation of ABOX ontology models is performed to determine whether the provided product basic data and process basic data can be transformed to OWL2 statements which are based on the appropriate K-RAMP TBOX ontology models.
- The usage of respective TBOX ontology models determines whether those models are complete in order to store the required information.
- The standard reasoning using OWL2 and SWRL determines whether the OWL2-based and SWRL-based reasoning rules of the K-RAMP TBOX ontology models lead to the expected assertions as specified during the decomposition of DPs.
- The behavior of the hybrid components is determined in order to verify whether the specified algorithms, embedded in the subprocess flows which result from the process design in Chapter 3.2.5, lead to the expected assertions as specified during the decomposition of DPs.

All subtest groups in conjunction with applicable evaluation activities are summarized in Table 4.1. Evaluation activities are performed through formal verification, if this is useful due to the complexity of implemented reasoning rules and assertions. Some subtest groups are simple enough that their evaluation activities are reduced to demonstrations based on examples.

With respect to complex reasoning rules and assertions, representative test data from the cake-baking case study are prepared in order to demonstrate the validity of results. Moreover, formal verifications of the respective rules are performed for such complex rules.

Test group	Subtest group	evaluate ...				
		... ability to prepare product basic data and process basic data	... generation of ABOX ontology model(s)	... ABOX model(s) based on TBOX partial model(s)	... reasoning with OWL and SWRL according to	... behavior of hybrids and assertions according to
1	i	<i>ProductionTechniques specialization</i>	kr-comm	kr-prodn-tech	DP _{1,4}	-
	ii	<i>EngineeringUnits Characteristics</i>	kr-comm kr-orig kr-targ	kr-unit kr-char	-	-
	iii	<i>Products ProductCategories Specifications</i>	kr-comm kr-orig kr-targ	kr-spec kr-prod kr-prod-cat	DP _{1,3} DP _{1,5} DP _{1,6}	PV _{1,3} DP _{1,3}
	iv	<i>ProcessCapabilities Specifications</i>	kr-orig kr-targ	kr-proc-cap	DP _{1,7}	-
2	i	<i>ProcessSegments isAProductionTechnique</i>	kr-orig kr-targ	kr-proc	DP _{3,1}	-
	ii	<i>ProcessSegmentSetups ProductCategoryUsage enableCapability onProcessSegment</i>	kr-orig kr-targ	kr-proc	DP _{3,2}	PV _{3,2} DP _{3,2}
	iii	<i>Equipment</i>	kr-targ	kr-equ	-	-
	iv	<i>EquipmentRecipes HandlingInstructions</i>	kr-targ	kr-req-recp kr-hd-instr	-	-
	v	<i>ProcessOperations ProcessOperationSetup onHandlingInstruction onEquipment onEquipmentRecipe onProcessOperation comprisesProcessOperatio</i>	kr-targ	kr-proc kr-props	DP _{3,3} DP _{3,6}	-
3	i	-	kr-targ-orig	kr-rec	DP _{4,1}	PV _{4,1} - <i>ProductCategorySet</i> DP _{4,1}
	ii	-	kr-targ-orig	kr-rec	-	PV _{4,1} - <i>Recommendation</i> DP _{4,1}

Table 4.1: Overview of test groups and subtest groups including their evaluation actions.

The first complex reasoning rules are considered by the subtest group 1.iii because the whole complexity of relationships between *Specifications* respectively between *SpecificationSets* needs to be invoked the first time in the sequel of the evaluation process. This complexity also requires the invocation of the hybrid specification of PV_{1,3}. The next complex verifications of reasoning rules to be considered during the evaluation process, particularly due to the hybrid specification of PV_{4,1}, are covered by subtest group 3.i. For both cases, several exemplary data sets are prepared in order to demonstrate the results of combinations of input data. Finally, the reasoning rules of subtest group 3.ii are not as complex as the reasoning rules of the previous groups with respect to their specification. However, these reasoning rules and the specified assertions represent the final outcome of the work. Moreover, these reasoning rules comprise the largest variation of possible input data compared to other groups (see Table 3.16) and require most intensive discussion of the results, based on the used input data.

4.3 The evaluation environment

The setup of supporting software tools during the evaluation and the data exchange in-between is specified in the following sections. For specifying the K-RAMP TBOX ontology

models, Protégé 5.0.0 beta-17 is used as editor. The same tool including the Pellet Reasoner Plug-in 2.2.0 is also used for reasoning purposes in conjunction with the ABOX ontology models of the original bakery and the target bakery during the evaluation phase (Figure 4.2). For editing relevant production data of the original bakery and the target bakery as well as for the taxonomies being in common for both bakeries, Microsoft® Excel® 2007 is applied.

Compared with the K-RAMP architecture in the real world, as shown in Figure 3.26, Microsoft® Excel® takes over the role of the MES software as a provider of product basic data and process basic data. Moreover, it generates the respective ABOX ontology models in OWL2 function syntax. The description of the architecture in Chapter 3.5 does already exclude the detailed specification of the input gateway or input API from the scope of this work and thus from the evaluation process. For the purpose of uploading the bakery’s ontology models into the knowledge store – into Protégé –, an alternative approach instead of SPARQL is applied during the evaluation process. As shown in Figure 4.2, the content of original spreadsheet documents (files with xlsx-extension) are transferred to ABOX ontology models in OWL2 functional syntax [55] (files with owl-extension). The transition is performed by simple text conversion and macro programming using means of Microsoft® Excel®.

According to the dependency schema of Figure 3.21, each ABOX ontology model imports the K-RAMP TBOX ontology models. For the sake of easier handling, all partial TBOX ontology models of K-RAMP are enveloped in one comprehensive K-RAMP TBOX model `kr.owl` during the evaluation.

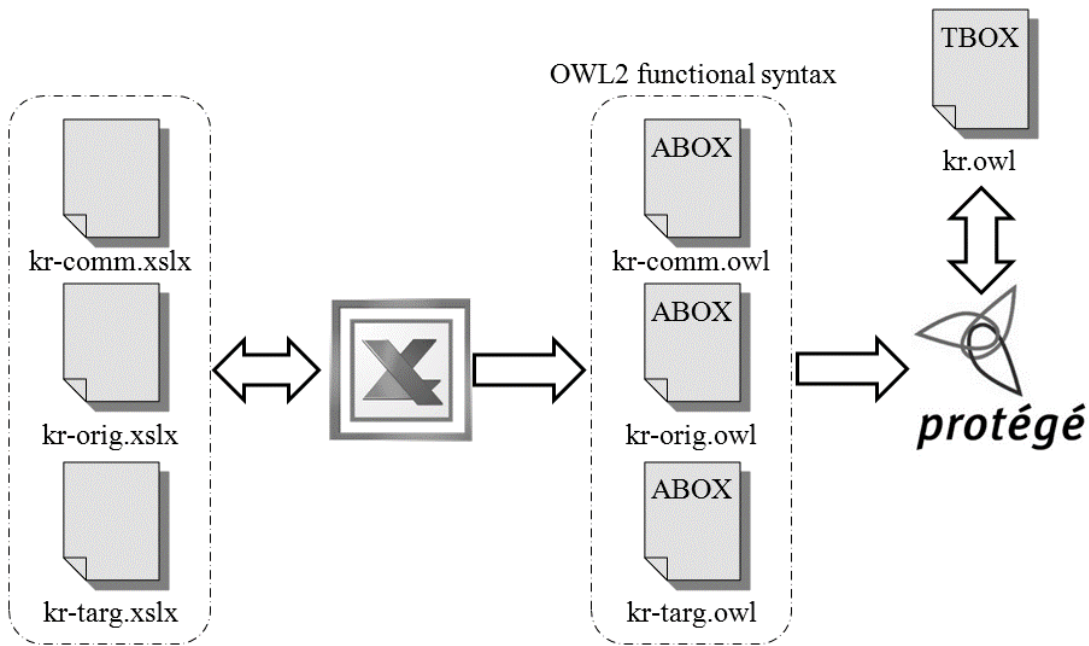


Figure 4.2: Setup of evaluation environment.

Consequently, unique prefixes are used for the URIs across all ontology models. The prefix `kr` is commonly used for all URIs resulting from `kr.owl`. The prefix `kr-comm` stands for the common branch-specific ABOX model of bakeries in general, while `kr-orig` and `kr-targ` are applied as prefixes for URIs of the ABOX models of the original bakery respectively the target bakery. Finally `kr-targ2` is used as prefix of the ontology which results from the integration of `kr-orig` and `kr-targ`.

4.4 Creation of ontology models

4.4.1 Common taxonomy assumptions

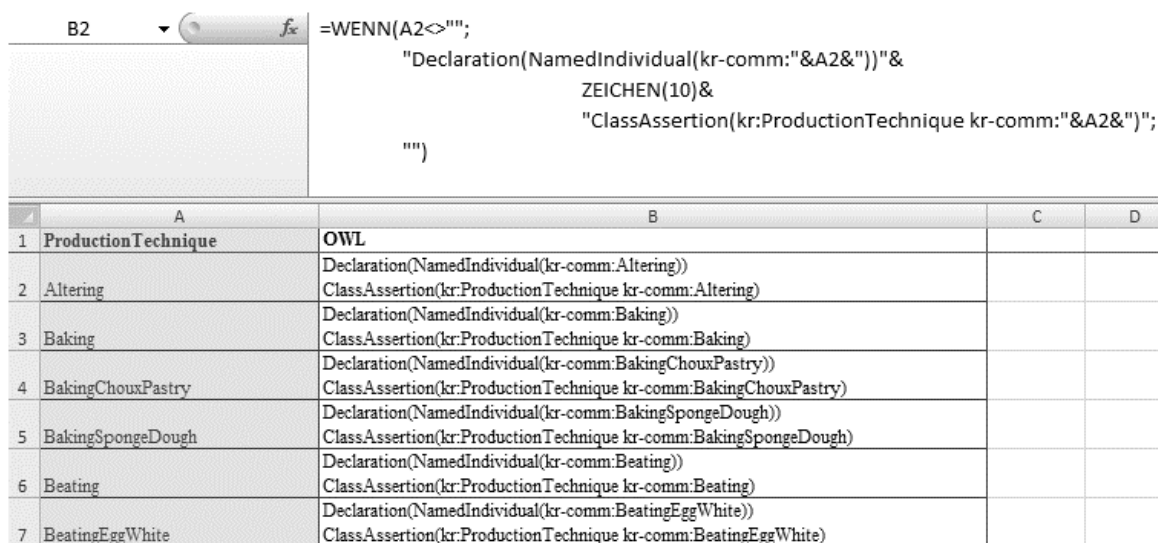
4.4.1.1 Overview

In Chapter 3.3, the concept of K-RAMP TBOX ontology models is discussed and how branch-specific or even branch-independent ABOX ontology models or taxonomies are derived from it. In the sequel of the following subchapter, a common ABOX ontology model `kr-comm` is specified which supports the exchange of information between the original bakery and the target bakery based on branch-specific terminology which is shared between both bakeries. The execution of the subtest groups 1.i, 1.ii and partially 1.iii is described according to Table 4.1.

Subchapter 4.4.1.2 fully covers subtest group 1.ii and therefore the verification of $DP_{1,4}$. The evaluation of $DP_{1,1}$ and $DP_{1,2}$ is described in subchapter 4.4.1.3 thus focusing on *EngineeringUnits* and *Characteristics*. The subchapter 4.4.1.4 addresses parts of the evaluation of $DP_{1,3}$ in conjunction with $DP_{1,5}$ and $DP_{1,6}$. All results of the evaluation process which is performed on `kr-comm` and particularly all conclusions are also applicable on `kr-targ`, `kr-orig` or every other branch-specific ontology model which is specified upon the schema of the used K-RAMP TBOX models.

4.4.1.2 Production Techniques

Figure 4.3 and Figure 4.4 show the definition of *ProductionTechniques* and their hierarchy of specialization. Although one comprehensive TBOX ontology model (`kr`) is applied for evaluation, it has to be mentioned that the definition of *ProductionTechniques* is based on the ontology model `kr-prodn-tech` of Table 3.26.



The image shows a screenshot of an Excel spreadsheet. The top part shows a formula bar with the following text: `=WENN(A2<>""; "Declaration(NamedIndividual(kr-comm:"&A2&"))"& ZEICHEN(10)& "ClassAssertion(kr:ProductionTechnique kr-comm:"&A2&"); ""))`. Below this is a table with 4 columns: A, B, C, and D. Column A contains the names of production techniques, column B contains the corresponding OWL2 functional syntax statements, and columns C and D are empty.

	A	B	C	D
1	ProductionTechnique	OWL		
2	Altering	Declaration(NamedIndividual(kr-comm:Altering)) ClassAssertion(kr:ProductionTechnique kr-comm:Altering)		
3	Baking	Declaration(NamedIndividual(kr-comm:Baking)) ClassAssertion(kr:ProductionTechnique kr-comm:Baking)		
4	BakingChouxPastry	Declaration(NamedIndividual(kr-comm:BakingChouxPastry)) ClassAssertion(kr:ProductionTechnique kr-comm:BakingChouxPastry)		
5	BakingSpongeDough	Declaration(NamedIndividual(kr-comm:BakingSpongeDough)) ClassAssertion(kr:ProductionTechnique kr-comm:BakingSpongeDough)		
6	Beating	Declaration(NamedIndividual(kr-comm:Beating)) ClassAssertion(kr:ProductionTechnique kr-comm:Beating)		
7	BeatingEggWhite	Declaration(NamedIndividual(kr-comm:BeatingEggWhite)) ClassAssertion(kr:ProductionTechnique kr-comm:BeatingEggWhite)		

Figure 4.3: Input of *ProductionTechniques*, the generated statements in OWL2 functional syntax and the generation logic.

Figure 4.3 shows the Microsoft® Excel® form template which is used to enter the names of *ProductionTechniques*. Beside URIs being derived from the names, no additional information with respect to *ProductionTechniques* is needed for the purpose of evaluation. Figure 4.4 shows the

specialization hierarchy of *ProductionTechniques*. Also for both forms, the generated ontology statements as OWL2 functional syntax are shown.

For the purpose of the evaluation, a useful subset of possible *ProductionTechniques* is specified in `kr-comm` which is used at both bakeries and thus represents a common set of *ProductionTechniques* for both. At the most general level, the hierarchy is derived from the basic *ProductionTechniques* “Master Forming”, “Forming”, “Separating”, “Merging”, “Coating” and “Altering”. It is trivial to specify *ProductionTechniques* as this step is fully performed through Semantic Web standard technologies which are provided by Protégé. Therefore, there is no need of in depth evaluation required.

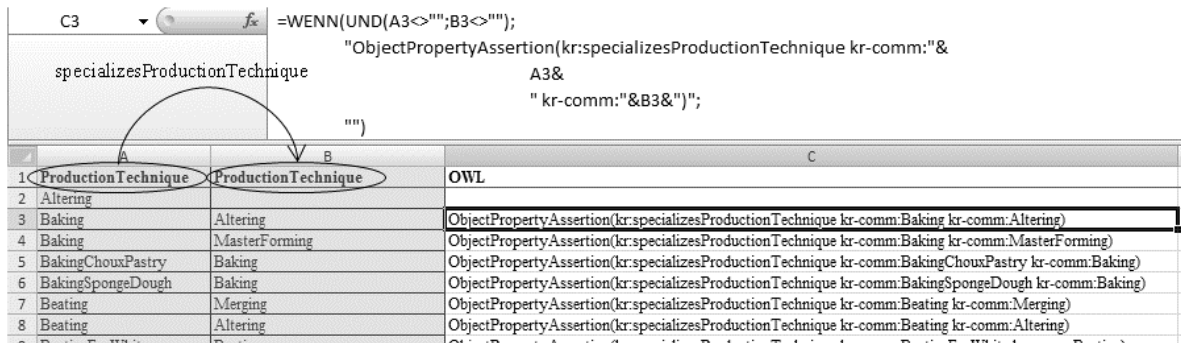


Figure 4.4: Input of specialization structure (*specializesProductionTechnique*), the generated statements in OWL functional syntax and the generation logic.

4.4.1.3 Engineering Units and Characteristics

A common ontology model of *EngineeringUnits* (Figure 4.5) and *Characteristics* (Figure 4.6) is also specified within `kr-comm`. It is assumed accordingly that *Engineering Units* are managed commonly between two production systems.

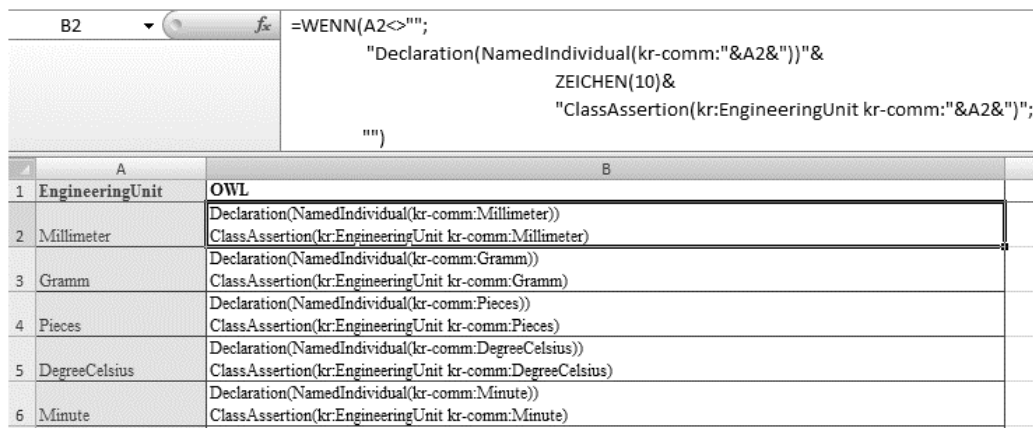


Figure 4.5: Input of *Engineering Units*, the generated statements in OWL2 functional syntax and the generation logic.

With respect to *Characteristics*, the same assumption of commonly shared individuals is made as premise for successful matchmaking between the ontology model of an original production system and a target production system. Even if there is no common management of *Characteristics* in place, stating of equivalence of *Characteristics* of two ontology models from different productions systems could be easily done by using the object property `owl:sameAs`.

The definition of the ABOX model for *EngineeringUnits* within `kr-comm` is based on the TBOX ontology model `kr-unit` of Table 3.26 while the ABOX model for *Characteristics* is based on the TBOX ontology model `kr-char`.

It is trivial to specify *EngineeringUnits* or *Characteristics* and therefore, there is no need of in depth evaluation, as this step is fully performed through Semantic Web standard technologies which are provided by OWL and more general technologies (RDFS, RDF). No reasoning needs to be considered as there are no such rules specified – neither by using OWL2 nor by using SWRL.

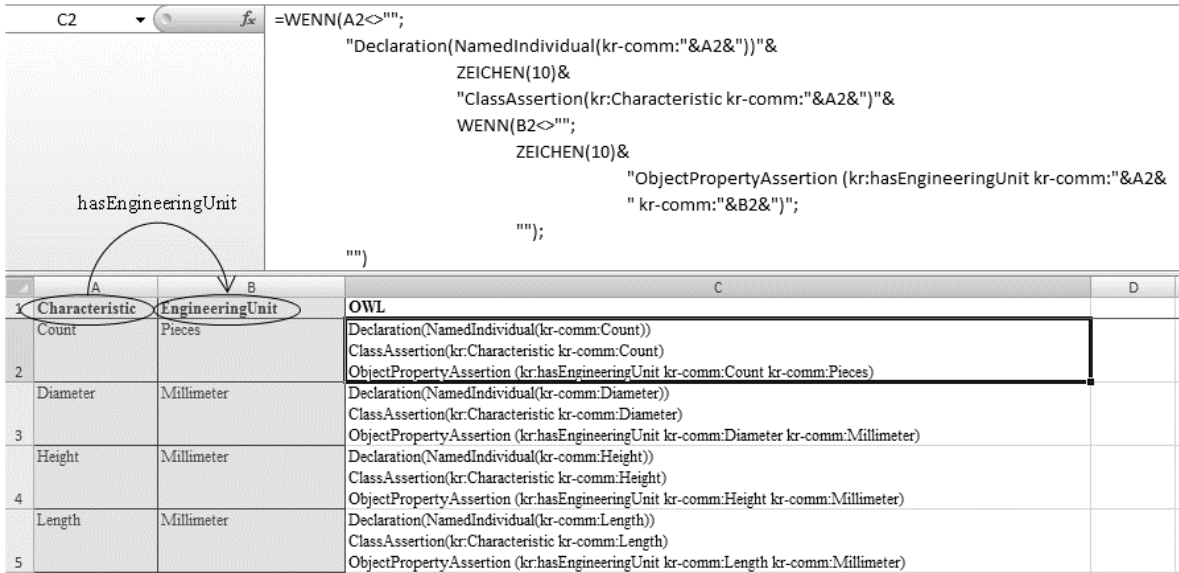


Figure 4.6: Input of Characteristics, the generated statements in OWL2 functional syntax and the generation logic.

4.4.1.4 Products, Product Categories and enclosing of Specification Sets

There is also a common taxonomy of *ProductCategories* (Figure 4.7) which is maintained in respective form templates. For this purpose, a general ontology for *ProductCategories* in the domain of food and beverage (e.g., “Flour”, “Water”, “Milk”, “Custard Powder”, “Dark Chocolate”) is provided as part of `kr-comm`. This ontology is not complete. However, it is sufficiently comprehensive in order to represent a common and realistic base-line for the individually specified *ProductCategory* ontologies of each bakery. The ontology for *ProductCategories* is generated as part of the ABOX model in `kr-comm` and is based on `kr-prod-cat` as listed in Table 3.26.

Beside the generation of individuals of *ProcessCategory*, the aim of this evaluation step is the verification of the implication rule for *specializesProductCategory*. With respect to the specialization of *ProductCategories* there are two possibilities. First, it is possible to assert specialization hierarchies explicitly by asserting the object property *specializesProductCategory* manually. This is useful, if there are no *Specifications* available which allow reasoning due to the object property *encloses*. This case is achieved, if product basic data with respect to *ProductCategories* are only reduced to the step ① as shown in Figure 4.7 followed by an explicit step of maintaining *specializesProductCategory* as shown in Figure 4.8.

However, the more common case is that *ProductCategories* comprise *Specifications*. As specified by DP_{1,3} in Table 3.13, the specialization hierarchy respectively the generalization

SWRL. At least the object property *enclosesSpecificationTarget* is essential in order to imply *enclosesSpecificationSet* and consequently *specializesProductCategory*. The hybrid specification of PV_{1,3}, as shown in Figure 3.22, needs to be invoked and therefore formally verified in the context of this evaluation step.

A	B	C
Product Category	specializesProductCategory	OWL
AlcoholicBeverage	Beverage	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:AlcoholicBeverage kr-comm:Beverage)
AnimalEmulsifier	Emulsifier	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:AnimalEmulsifier kr-comm:Emulsifier)
AnimalFat	EdibleFat	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:AnimalFat kr-comm:EdibleFat)
AnimalFattyFood	FattyFood	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:AnimalFattyFood kr-comm:FattyFood)
AnimalProtein	Protein	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:AnimalProtein kr-comm:Protein)
ApricotJam	Jam	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:ApricotJam kr-comm:Jam)
BakingPowder	Leavening	ObjectPropertyAssertion(kr:specializesProductCategory kr-comm:BakingPowder kr-comm:Leavening)

Figure 4.8: Input of specialization structure (*specializesProductCategory*), the generated statements in OWL2 functional syntax and the generation logic.

After the upload of *ProductCategories* and their *Specifications*, the first step according to the specification of PV_{1,3} is the standard reasoning of the Semantic Web database followed by a pair-wise comparison of *SpecificationTargets*. If a pair can be compared (*comparableSpecification*), the procedure *assertRelationOfSpecificationTarget* is executed. *ProductCategories* can be structured differently and therefore different combinations of such pairs of *ProductCategories* need to be determined during the verification.

Step #	Actions/Assumptions
7	$st_o = st_i = st$
8	<i>assertRelationOfSpecificationTargets</i> (<i>st</i> , <i>st</i>)
8.1	$targ1 \leftarrow st, targ2 \leftarrow st$
8.2	$setX \leftarrow unspecified, setY \leftarrow unspecified, y \leftarrow unspecified, bEncloses \leftarrow false$
8.3	$hasTarget(targ1, setX), hasTarget(targ2, setY) \Rightarrow hasTarget(st, setX), hasTarget(st, setY)$
8.3.1	$hasTarget(st, setX=unspecified) \equiv$ <i>SELECT ?st ?setX WHERE { ?st kr:hasTarget ?setX . FILTER (?st = <uri of st>) }</i>
8.3.1.1	$kr:hasTarget(st, t) \rightarrow \{(st, t_k) : k \geq 1\}$
8.3.1.2	$hasTarget(st, setX=unspecified) \equiv$ <i>SELECT ?st ?setX WHERE { ?st kr:hasTarget ?setX . FILTER (?st = <uri of st>) } \rightarrow \{(st, t_k) : k \geq 1\}</i>
8.3.1.3	$hasTarget(st, setX=unspecified) \rightarrow \{(st, t_k) : k \geq 1\}$
8.3.2	$setX \leftarrow \{t_k : k \geq 1\}$
8.3.3	$hasTarget(st, setY=unspecified) \equiv$ <i>SELECT ?st ?setY WHERE { ?st kr:hasTarget ?setY . FILTER (?st = <uri of st>) }</i>
8.3.3.1	$kr:hasTarget(st, t) \rightarrow \{(st, t_k) : k \geq 1\}$
8.3.3.2	$hasTarget(st, setY=unspecified) \equiv$ <i>SELECT ?st ?setY WHERE { ?st kr:hasTarget ?setY . FILTER (?st = <uri of st>) } \rightarrow \{(st, t_k) : k \geq 1\}</i>
8.3.3.3	$hasTarget(st, setY=unspecified) \rightarrow \{(st, t_k) : k \geq 1\}$
8.3.4	$setY \leftarrow \{t_k : k \geq 1\}$
8.4	$setY = setX \Rightarrow setX \cap setY = setX = setY \Rightarrow setX \cap setY = setY \Rightarrow$ <i>assert(enclosesSpecificationTarget, st, st)</i>
8.4.1	<i>assert(enclosesSpecificationTarget, st, st) \equiv CONSTRUCT { st kr:enclosesSpecificationTarget st }</i>
8.5	$comparableSpecification(st, st) \wedge hasTarget(st, setX) \supseteq hasTarget(st, setY) \Rightarrow enclosesSpecificationTarget(st, st)$

Table 4.2: Verification of *assertRelationOfSpecificationTargets* in case of pair-wise comparison of a *SpecificationTarget* with itself.

In Table 4.2, the verification of a pair-wise comparison of a *SpecificationTarget* with itself is summarized. Gathering the sets of comprised target values (*hasTarget*) for each member of the

pair, consequently results in two sets $setX$ and $setY$ of target values with identical members (8.3.2, 8.3.4) which finally allows the conclusion that $setX$ is also a subset of $setY$ itself. This matches with the specification of $DP_{1,3}$ and leads to the assertion that each *SpecificationTarget* encloses itself.

Step #	Actions/Assumptions
9	$st_o \neq st_i \leftarrow M=\{t_m : kr:hasTarget(st_i, t_m)\} \wedge N=\{t_n : kr:hasTarget(st_o, t_n)\} \wedge 0 = M \cap N < N $
10	$assertRelationOfSpecificationTargets(st_o, st_i)$
10.1	$targ1 \leftarrow st_o, targ2 \leftarrow st_i$
10.2	$setX \leftarrow unspecified, setY \leftarrow unspecified, y \leftarrow unspecified, bEncloses \leftarrow false$
10.3	$hasTarget(targ1, setX), hasTarget(targ2, setY) \Rightarrow hasTarget(st_o, setX), hasTarget(st_i, setY)$
10.3.1	$hasTarget(st_o, setX=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:hasTarget ?y . FILTER (?x = <uri of st_o>) \}$
10.3.1.1	$kr:hasTarget(st_o, t) \rightarrow \{(st_o, t_k) : k \geq 1\}$
10.3.1.2	$hasTarget(st_o, setX=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:hasTarget ?y . FILTER (?x = <uri of st_o>) \} \rightarrow \{(st_o, t_k) : k \geq 1\}$
10.3.1.3	$hasTarget(st_o, setX=unspecified) \rightarrow \{(st_o, t_k) : k \geq 1\}$
10.3.2	$setX \leftarrow \{t_k : k \geq 1\}$
10.3.3	$hasTarget(st_i, setY=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:hasTarget ?y . FILTER (?x = <uri of st_i>) \}$
10.3.3.1	$kr:hasTarget(st_i, t) \rightarrow \{(st_i, t_k) : k \geq 1\}$
10.3.3.2	$hasTarget(st_i, setY=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:hasTarget ?y . FILTER (?x = <uri of st_i>) \} \rightarrow \{(st_i, t_k) : k \geq 1\}$
10.3.3.3	$hasTarget(st_i, setY=unspecified) \rightarrow \{(st_i, t_k) : k \geq 1\}$
10.3.4	$setY \leftarrow \{t_l : l \geq 1\}$
10.4	$N \otimes M \Leftrightarrow setY \otimes setX \Rightarrow \forall y:setY \rightarrow \exists x:setX.x=y \vee generalizes(x, y) \Rightarrow setX \cap setY = \{ \} \Rightarrow$ $ setX \cap setY = 0 \Rightarrow \varepsilon$
10.5	$comparableSpecification(st_o, st_i) \wedge setX \cap setY \Rightarrow$ $comparableSpecification(st_o, st_i) \wedge \forall x \exists y.hasTarget(t_1, x) \wedge hasTarget(t_2, y) \wedge$ $(SameAs(x, y) \vee generalizes(x, y)) = false$
10.6	$comparableSpecification(st_o, st_i) \wedge false \equiv false \Rightarrow \neg enclosesSpecificationTarget(st_o, st_i)$

Table 4.3: Verification of *assertRelationOfSpecificationTargets* in case of pair-wise comparison of *SpecificationTargets* with disjoint sets of target values.

The opposite situation is verified in Table 4.3, namely a pair of *SpecificationTargets* with disjoint sets of target values. In this case $setX$ and $setY$ are disjoint (10.3.2, 10.3.4). There is not a common intersection of $setX$ and $setY$ to be determined if the respective implication rule of $DP_{1,3}$ is recalled.

$$comparableSpecification(t_1, t_2) \wedge \forall x \exists y.hasTarget(t_1, x) \wedge hasTarget(t_2, y) \wedge (SameAs(x, y) \vee generalizes(x, y)) \rightarrow enclosesSpecificationTarget(t_1, t_2)$$

Members x and y of $setX$ respectively of $setY$ must be either identical – what is never the case under the current condition – or x is a generalization of y . Therefore, a non-empty set of *TargetValues* of $setX$ which are enclosing *TargetValues* of $setY$ is possible even if both sets are disjoint.

For this reason, two operators are introduced for the purpose of verification. The operator \cap is treated as an extension of the common intersection operator of set theory while the operator \otimes extends the meaning of the \neq -operator.

$$M = \{m_i | m_i \in \mathbb{N}\}$$

$$\begin{aligned}
N &= \{n_j \mid n_j \in \mathbb{N}\} \\
M \oslash N &= \{m_i \in M \mid \exists n_j \in N. m_i = n_j \vee \text{generalizes}(m_i, n_j)\} \\
M \otimes N &= \{m_i \in M \mid \nexists n_j \in N. m_i = n_j \vee \text{generalizes}(m_i, n_j)\}
\end{aligned}$$

By the use of these operators, the verification of the current procedure can be continued. The relationship disjoint in the current verification scenario means $M \oslash N$. Consequently, setX is no superset of setY and there is also no element in setX which generalizes an element of setY , and for this reason the given *SpecificationTargets* do not enclose each other. Moreover, $M \oslash N$ is empty and therefore the given *SpecificationTargets* do not overlap each other either. Also this result matches with the specification of $\text{DP}_{1,3}$.

Step #	Actions/Assumptions
11	$st_o \neq st_i \Leftarrow M = \{t_m : kr:\text{hasTarget}(st_i, t_m)\} \wedge N = \{t_n : kr:\text{hasTarget}(st_o, t_n)\} \wedge 0 < M \oslash N = N $
12	$\text{assertRelationOfSpecificationTargets}(st_o, st_i)$
12.1	$\text{targ1} \leftarrow st_o, \text{targ2} \leftarrow st_i$
12.2	$\text{setX} \leftarrow \text{unspecified}, \text{setY} \leftarrow \text{unspecified}, y \leftarrow \text{unspecified}, b\text{Encloses} \leftarrow \text{false}$
12.3	$\text{hasTarget}(\text{targ1}, \text{setX}), \text{hasTarget}(\text{targ2}, \text{setY}) \Rightarrow \text{hasTarget}(st_o, \text{setX}), \text{hasTarget}(st_i, \text{setY})$
12.3.1	see 10.3.1
12.3.1.1	see 10.3.1.1
12.3.1.2	see 10.3.1.2
12.3.1.3	see 10.3.1.3
12.3.2	see 10.3.2
12.3.3	see 10.3.3
12.3.3.1	see 10.3.3.1
12.3.3.2	see 10.3.3.2
12.3.3.3	see 10.3.3.3
12.3.4	see 10.3.4
12.4	$ M \oslash N = N \Leftrightarrow \text{setX} \oslash \text{setY} = N \Rightarrow \forall x \in \text{setX} \exists y \in \text{setY}. x=y \vee \text{generalizes}(x,y) \Rightarrow \text{assert}(\text{enclosesSpecificationTarget}, st_o, st_i)$
12.4.1	$\text{assert}(\text{enclosesSpecificationTarget}, st_o, st_i) \equiv \text{CONSTRUCT}\{st_o \text{ kr:enclosesSpecificationTarget } st_i\}$

Table 4.4: Verification of $\text{assertRelationOfSpecificationTargets}$ in case of pair-wise comparison of *SpecificationTargets* with a fully enclosed subset of target values.

Table 4.4 summarizes the verification of the case in which the set of target values of one *SpecificationTarget* fully encloses the set of target values of the other *SpecificationTarget*. This is the case where every element of setY is either identical with or generalized by an element of setX . It is not possible that setY is the empty set. This situation would imply that the second *SpecificationTarget* does not aggregate any target value. However, this is not allowed in accordance with the information model of $\text{DP}_{1,3}$ (see Figure 3.10). As a consequence, the \oslash -operator for both sets is not empty and the cardinality of the \oslash -operator is equal to the cardinality of setY . Therefore, one *SpecificationTarget* targ1 encloses a *SpecificationTarget* targ2 if the aggregated target values of targ2 are either identical with an element of targ1 or generalized by an element of targ1 .

The case being summarized in Table 4.5 represents the situation of two *SpecificationTargets* and thus sets setX and setY of target values where at least one element of setX is identical with an element of setY or generalizes an element of setY or specializes an element of setY or shares a common specialization with an element of setY . This case results in setX and setY with a non empty intersection. This case is related to the implication rule of $\text{DP}_{1,3}$ which is specified as

$$\begin{aligned} & comparableSpecification(t_1, t_2) \wedge \neg enclosesSpecificationTarget(t_1, t_2) \wedge \\ & \exists x \exists y. hasTarget(t_1, x) \wedge hasTarget(t_2, y) \wedge \\ & (SameAs(x, y) \vee generalizes(x, y) \vee specializes(x, y) \vee \\ & (generalizes(c, x) \wedge generalizes(c, y))) \rightarrow overlapsSpecificationTarget(t_1, t_2) \end{aligned}$$

Given the previously introduced sets N and M the respective operator is specified as

$$\begin{aligned} N \square M &= \{n_i \in N \mid \exists m_j \in M. n_i \\ &= m_j \vee generalizes(n_i, m_j) \vee specializes(n_i, m_j) \vee (generalizes(c, m_j) \\ &\wedge generalizes(c, n_i))\} \end{aligned}$$

In conjunction with Table 4.5, the cardinality of this operator is considered as less than or equal to the cardinality of `setY`. A cardinality which is equal to 0 leads to the verification as it is already performed in Table 4.3. This pair of *SpecificationTargets* is considered as overlapped. The conclusion does also match with the specification of `DP1,3`.

It can be thus shown that `assertRelationOfSpecificationTarget` results in correctly asserted relations with respect to the object properties *overlapsSpecificationTarget* and *enclosesSpecificationTarget*. As a consequence, the subsequent step of standard reasoning asserts the respective relations concerning *overlapsSpecification*, *overlaps*, *enclosesSpecification* and *encloses*.

Step #	Actions/Assumptions
13	$st_o \neq st_i \Leftarrow M = \{t_m : kr:hasTarget(st_i, t_m)\} \wedge N = \{t_n : kr:hasTarget(st_o, t_n)\} \wedge 0 < M \square N \leq N $
14	<code>assertRelationOfSpecificationTargets(st_o, st_i)</code>
14.1	$targ1 \leftarrow st_o, targ2 \leftarrow st_i$
14.2	$setX \leftarrow unspecified, setY \leftarrow unspecified, bEncloses \leftarrow false$
14.3	$hasTarget(targ1, setX), hasTarget(targ2, setY) \Rightarrow hasTarget(st_o, setX), hasTarget(st_i, setY)$
14.3.1	see 10.3.1
14.3.1.1	see 10.3.1.1
14.3.1.2	see 10.3.1.2
14.3.1.3	see 10.3.1.3
14.3.2	see 10.3.2
14.3.3	see 10.3.3
14.3.3.1	see 10.3.3.1
14.3.3.2	see 10.3.3.2
14.3.3.3	see 10.3.3.3
14.3.4	see 10.3.4
14.4	$0 < M \square N \leq N \Leftrightarrow 0 < setX \square setY \leq setY \Rightarrow assert(overlapsSpecificationTarget, st_o, st_i)$
14.4.1	<code>assert(overlapsSpecificationTarget, st_o, st_i) ≡ CONSTRUCT { st_o kr:overlapsSpecificationTarget st_i }</code>

Table 4.5: Verification of `assertRelationOfSpecificationTargets` in case of pair-wise comparison between *SpecificationTargets* with a common real subset of target values.

The subsequent step is again a pair-wise comparison. However, this step is performed on the level of *SpecificationSets*. Similar to *SpecificationTargets*, the verification of the pair-wise comparison of *SpecificationSets* has to consider self-pairing, pairing between disjoint *SpecificationSets*, pairing between enclosing *SpecificationSets*, but even two quality levels of overlapping between *SpecificationSets*. The logic of the pseudo code of (3.2), which implements this pair-wise comparison between *SpecificationSets* and the assertion of the appropriate relationship, is verified in the next sections.

Table 4.6 summarizes the verification of the self comparison of one individual of *SpecificationSet*. In this case, the individual set is assigned to both variables `set1` and `set2` (16.1 in Table 4.6). Consequently, the gathering of comprised *Specifications* results in two sets X and Y

which comprise exactly the same mixture of individuals of *SpecificationRanges* or *SpecificationTargets*. Due to the premises being discussed in conjunction with the pseudo code specification of (3.2), each member of X encloses itself as member of Y . Therefore, *Matchmaking scenario 1 of SpecificationSets* as discussed in conjunction with (3.2) is valid (16.1 in Table 4.6) which leads to the assertion of an individual of *RelationOfSpecificationSets* (16.5.1.1 in Table 4.6).

Step #	Actions/Assumptions
15	$set_o = set_i = set$
16	$assertRelationOfSpecificationSets (set, set)$
16.1	$set1 \leftarrow set, set2 \leftarrow set$
16.2	$X, Y, curChar, ross \leftarrow unspecified, setChar \leftarrow \{\}, cntOverlaps, cntEncloses \leftarrow 0$
16.3	$comprisesSpecification (set1, X), comprisesSpecification (set2, Y) \Rightarrow$ $comprisesSpecification (set, X), comprisesSpecification (set, Y)$
16.3.1	$comprisesSpecification (set, X=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:comprisesSpecification ?y . FILTER (?x = <uri of set>) \}$
16.3.1.1	$kr:comprisesSpecification (set, s) \rightarrow \{(set, s_k) : k \geq 1\}$
16.3.1.2	$comprisesSpecification (set, X=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:comprisesSpecification ?y . FILTER (?x = <uri of set>) \} \rightarrow \{(set, s_k) : k \geq 1\}$
16.3.1.3	$comprisesSpecification (set, X=unspecified) \rightarrow \{(set, s_k) : k \geq 1\}$
16.3.2	$X \leftarrow \{s_k : k \geq 1\}$
16.3.3	$comprisesSpecification (set, Y=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:comprisesSpecification ?y . FILTER (?x = <uri of set>) \}$
16.3.3.1	$kr:comprisesSpecification (set, s) \rightarrow \{(set, s_k) : k \geq 1\}$
16.3.3.2	$comprisesSpecification (set, Y=unspecified) \equiv$ $SELECT ?x ?y WHERE \{ ?x kr:comprisesSpecification ?y . FILTER (?x = <uri of set>) \} \rightarrow \{(set, s_k) : k \geq 1\}$
16.3.3.3	$comprisesSpecification (set, Y=unspecified) \rightarrow \{(set, s_k) : k \geq 1\}$
16.3.4	$Y \leftarrow \{s_k : k \geq 1\}$
16.4	$Y = X \Rightarrow \forall x:X \exists y:Y x = y \Rightarrow$ $\forall x:X \exists y:Y \exists c:C \text{ encloses } (x, y) \Rightarrow$ $\forall x:X \exists y:Y \exists c:C \text{ encloses } (x, y) \wedge \text{onCharacteristic } (x, c) \wedge \text{onCharacteristic } (y, c) \Rightarrow$ $ X = Y = C \Rightarrow X > 0 \Rightarrow G(set, set, C, X)$
16.5	$G(set, set, C, X)$
16.5.1	$assert (RelationOfSpecificationSets, ross);$ $assert (fromSpecificationSet, ross, set);$ $assert (toSpecificationSet, ross, set);$ $assert (commonCharacteristic, ross, C);$ $assert (ratioEnclosing, ross, X / set * 100);$ $assert (ratioCommonCharacteristics, ross, C / set * 100);$
16.5.1.1	$CONSTRUCT \{$ $_ :n \text{ rdf:type } kr:RelationOfSpecificationSet.$ $_ :n \text{ kr:fromSpecificationSet } <uri \text{ of } set>.$ $_ :n \text{ kr:toSpecificationSet } <uri \text{ of } set>.$ $_ :n \text{ kr:commonCharacteristic } <uri \text{ of } c1>.$ $_ :n \text{ kr:commonCharacteristic } <uri \text{ of } c2>.$ \dots $_ :n \text{ kr:commonCharacteristic } <uri \text{ of } cn>.$ $_ :n \text{ kr:ratioEnclosing } i1.$ $_ :n \text{ kr:ratioCommonCharacteristic } i2.\}$

Table 4.6: Verification of $assertRelationOfSpecificationSets$ in case of pair-wise comparison of an individual of *SpecificationSet* with itself and assertion of *RelationOfSpecificationSets*.

Based on the verification example in Table 4.6, the behavior of (3.2) with respect to the *Matchmaking scenario 1 of SpecificationSets* for a pair of nonidentical individuals can be determined easily. It is also possible to imply the respective behavior for *Matchmaking scenario 2* and *Matchmaking scenario 3* easily because the major difference between all scenarios is solely step 16.4 in Table 4.6. However, the differences between all three scenarios with respect to the

determination of *ratioEncloses* and *ratioCommonCharacteristic* are already shown in detail in Chapter 3.4. All conclusions with respect to the object parts of other object properties in the domain of *RelationOfSpecificationSets* remain unchanged.

The execution of `assertRelationOfSpecificationSets` is followed by standard reasoning. Due to the asserted individuals of *RelationOfSpecificationSets* and related object properties, this phase of standard reasoning results in assertions which depend on these information entities. According to DP_{1,3}, relations with respect to object properties *enclosesSpecificationSet*, *overlapsSpecificationSet* and *partiallySpecificationSet* are asserted in this phase of standard reasoning.

As a consequence of these assertions, the implication rules which are specified by DP_{1,6} and DP_{1,7} may cause respective assertions as well. With respect to DP_{1,6}, *isAProductCategory* and *isA* but also *specializesProductCategory*, *specializes* and *generalizes* are asserted within the overall context of PV₁.

The final hybrid part of PV_{1,3} asserts subset relations between *RelationOfSpecificationSets*. Pairs of *RelationOfSpecificationSets* are compared for this purpose by invoking the hybrid component `assertSubsetOf`. According to the specification of the implication rule for *subsetOf*, one *RelationOfSpecificationSet* is the subset of another one if it comprises a subset of common *Characteristics* (*commonCharacteristic*) of the other one.

Step #	Actions/Assumptions
17	$ross_o = ross_i = ross$
18	$subsetOf(ross, ross)$
18.1	$ross1 \leftarrow ross, ross2 \leftarrow ross$
18.2	$setX, setY \leftarrow unspecified$
18.3	$commonCharacteristic(ross1, setX), commonCharacteristic(ross2, setY) \Rightarrow commonCharacteristic(ross, setX), commonCharacteristic(ross, setY)$
18.3.1	$commonCharacteristic(ross, setX=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross>) \}$
18.3.1.1	$kr:commonCharacteristic(ross, s) \rightarrow \{(ross, c_k) : k \geq 1\}$
18.3.1.2	$commonCharacteristic(ross, setX=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross>) \} \rightarrow \{(ross, c_k) : k \geq 1\}$
18.3.1.3	$commonCharacteristic(ross, setX=unspecified) \rightarrow \{(ross, c_k) : k \geq 1\}$
18.3.2	$setX \leftarrow \{c_k : k \geq 1\}$
18.3.3	$commonCharacteristic(ross, setY=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross>) \}$
18.3.3.1	$kr:commonCharacteristic(ross, s) \rightarrow \{(ross, c_k) : k \geq 1\}$
18.3.3.2	$commonCharacteristic(ross, setY=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross>) \} \rightarrow \{(ross, c_k) : k \geq 1\}$
18.3.3.3	$commonCharacteristic(ross, setY=unspecified) \rightarrow \{(ross, c_k) : k \geq 1\}$
18.3.4	$setY \leftarrow \{c_k : k \geq 1\}$
18.4	$setY = setX \Rightarrow setX \subseteq setY \Rightarrow G(ross1, ross2) \Rightarrow G(ross, ross)$
18.5	$G(ross, ross)$
18.5.1	$assert(subsetOf, ross, ross)$
18.5.1.1	$CONSTRUCT \{ ross kr:subsetOf ross \}$

Table 4.7: Verification of `assertSubsetOf` in case of pair-wise comparison of an individual of *RelationOfSpecificationSets* with itself and assertion of *subsetOf*.

Three possible scenarios have to be considered during the verification of `assertSubsetOf`. First, a *RelationOfSpecificationSet* is a subset of itself. This should be implicitly the case, as the comprised set of common *Characteristics* is a subset of itself. Secondly,

a *RelationOfSpecificationSets* is not a subset of another *RelationOfSpecificationSets* with disjoint common *Characteristics*. Thirdly, one *RelationOfSpecificationSets* $ross_1$ is a subset of another *RelationOfSpecificationSets* $ross_2$ if the common *Characteristics* of $ross_1$ are a subset of the common *Characteristics* of $ross_2$. In set theory, $ross_1$ with an empty set of *Characteristics* would be always a subset of $ross_2$. However, an empty set of common *Characteristics* is never possible because this is prohibited by the information model of $DP_{1,3}$. This situation is also not possible due to the three *Matchmaking scenarios of SpecificationSets*. Therefore, the third case for the verification of `assertSubsetOf` is limited to subset relations of nonempty sets of common *Characteristics*. For two sets $S_1 = \{c_i \mid \exists ross_1 \forall c_i.commonCharacteristic(ross_1, c_i)\}$ and $S_2 = \{c_j \mid \exists ross_2 \forall c_j.commonCharacteristic(ross_2, c_j)\}$, this leads to the two possible cases $S_1 \subset S_2$ or $S_1 = S_2 \wedge ross_1 \neq ross_2$. The first discussed cases is a special form of $S_1 = S_2$ with $ross_1 = ross_2$.

According to this preliminary discussion, comparison of a *RelationOfSpecificationSets* with itself is verified first. This verification is followed by $ross_1$ being a real subset of $ross_2$, which is then followed by the verification of *RelationOfSpecificationSets* with disjoint sets of common *Characteristics*.

Step #	Actions/Assumptions
19	$\exists ross_o, ross_i \text{ } ross_o \neq ross_i \wedge \forall c_i, c_k : commonCharacteristic(ross_o, c_i) \wedge commonCharacteristic(ross_i, c_k) \wedge c_i \neq c_k$
20	<code>subsetOf(ross_o, ross_i)</code>
20.1	$ross_1 \leftarrow ross_o, ross_2 \leftarrow ross_i$
20.2	$setX, setY \leftarrow unspecified$
20.3	$commonCharacteristic(ross_1, setX), commonCharacteristic(ross_2, setY) \Rightarrow commonCharacteristic(ross_o, setX), commonCharacteristic(ross_i, setY)$
20.3.1	$commonCharacteristic(ross_o, setX=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross_o>) \}$
20.3.1.1	$kr:commonCharacteristic(ross_o, s) \rightarrow \{(ross_o, c_k) : k \geq 1\}$
20.3.1.2	$commonCharacteristic(ross_o, setX=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross_o>) \} \rightarrow \{(ross_o, c_k) :$
20.3.1.3	$commonCharacteristic(ross_o, setX=unspecified) \rightarrow \{(ross_o, c_k) : k \geq 1\}$
20.3.2	$setX \leftarrow \{c_k : k \geq 1\}$
20.3.3	$commonCharacteristic(ross_i, setY=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross_i>) \}$
20.3.3.1	$kr:commonCharacteristic(ross_i, s) \rightarrow \{(ross_i, c_k) : k \geq 1\}$
20.3.3.2	$commonCharacteristic(ross_i, setY=unspecified) \equiv SELECT ?x ?y WHERE \{ ?x kr:commonCharacteristic ?y . FILTER (?x = <uri of ross_i>) \} \rightarrow \{(ross_i, c_k) :$
20.3.3.3	$commonCharacteristic(ross_i, setY=unspecified) \rightarrow \{(ross_i, c_k) : k \geq 1\}$
20.3.4	$setY \leftarrow \{c_k : k \geq 1\}$
20.4	$setY \neq setX \Rightarrow setX \not\subset setY \Rightarrow \epsilon$

Table 4.8: Verification of `assertSubsetOf` in case of pair-wise comparison of different individuals of *RelationOfSpecificationSets* with disjoint sets of common *Characteristics*.

Table 4.7 summarizes the verification of `subsetOf` in case of pair-wise comparison of a *RelationOfSpecificationSet* with itself. As already clarified, gathering the common set of *Characteristics* through `commonCharacteristic` results in `setX` and `setY` (18.3.2, 18.3.4 in Table 4.7) which comprise exactly the same *Characteristics*. Consequently, `setX` is a subset of `setY` and therefore $ross$ is asserted as subset of itself. The similarity with $ross_1 \neq ross_2$ is discussed in the previous sections. As `setX` and `setY`, due to the respective precondition, result again in two sets which comprise exactly the same *Characteristics*, also this general form of the

case is valid and `ross1` is asserted as subset of `ross2`. Throughout the nested loop where `assertSubsetOf` is embedded (see Figure 3.22), also `ross2` is asserted as subset of `ross1` in a separate call of the procedure.

The final case to be verified with respect to potential subset relationship within pairs of *RelationOfSpecificationSets* covers disjoint sets of common *Characteristics* (Table 4.8). As already discussed the gathered `setX` and `setY` of common *Characteristics* are disjoint. Consequently, `setX` is no subset of `setY` and therefore no assertion is performed.

There are no implication rules specified for $DP_{1,3}$ which depends on the existence of relations with respect to *subsetOf*. Therefore, there is also no subsequent standard reasoning required within the context of the hybrid specification of $PV_{1,3}$.

The previously discussed verifications of single steps of $PV_{1,3}$ are fleshed out subsequently by a couple of examples from the cake-baking case study which are listed in Table 4.9. Throughout the following discussion, the leading namespaces of the URIs are omitted. This measure underlines that the discussion is independent of the ontology model. The same conclusions can be made for `kr-comm`, `kr-orig` or `kr-targ` or also for any other ABOX ontology model which is based on the concepts of the K-RAMP models.

Importing the product basic data from Microsoft ® Excel ® to the OWL2 data base is resulting in a series of triples. The most relevant ones are listed as example for the *ProductCategory* “ApricotJam”. The other table entries are treated accordingly.

```
:ApricotJam rdf:type kr:ProductCategory, owl:NamedIndividual;
  kr:SpecificationSet_comprisesSpecification :ApricotJam-Fruite,
    :ApricotJam-RatioFruite,
    :ApricotJam-RatioSweetener;
  kr:specializesProductCategory :Jam .
:ApricotJam-Fruite rdf:type kr:SpecificationTarget, owl:NamedIndividual;
  kr:hasTarget :Apricot;
  kr:onCharacteristic :Fruite .
:ApricotJam-RatioFruite rdf:type kr:SpecificationRange, owl:NamedIndividual;
  kr:hasLSL 45.00;
  kr:hasUSL 60.00;
  kr:onCharacteristic :RatioFruite .
```

The initial reasoning according to the hybrid specification of $PV_{1,3}$ with respect to `:ApricotJam-A-RatioFruite` results in a set of relations *comparableSpecification*, *endWith*, *startsBefore* and other relations (see Figure 3.10) which further concludes

```
:ApricotJam-RatioFruite kr:encloseSpecificationRange
  :StrawberryJam-RatioFruite .
:ApricotJam-RatioFruite kr:encloseSpecificationRange :Jam-RatioFruite .
:ApricotJam-RatioFruite kr:encloseSpecificationRange
  :StrawberryJam-A-RatioFruite .
:ApricotJam-RatioFruite kr:encloseSpecificationRange :ApricotJam-RatioFruite .
:ApricotJam-RatioFruite kr:encloseSpecificationRange :ApricotJam-A-RatioFruite .
:ApricotJam-RatioFruite kr:encloseSpecificationRange :ApricotJam-B-RatioFruite .
```

as well as the respective superior object properties `kr:enclosesSpecification` and `kr:encloses`.

ProductCategory	Characteristic	SpecificationTarget	SpecificationRange		EngineeringUnit
		has Target	has LSL	has USL	
ApricotJam	Fruite	Apricot			
	RatioFruite		45.00	60.00	Percent
	RatioSweetener		55.00	60.00	Percent
Jam	RatioFruite		45.00	60.00	Percent
	RatioSweetener		55.00	60.00	Percent
StrawberryJam	Fruite	Strawberry			
	RatioFruite		45.00	60.00	Percent
	RatioSweetener		55.00	60.00	Percent
BakingPowder	NeutralizingValue		67.00	73.00	
	RateOfReaction		38.00	42.00	Percent
Product	Characteristic	has Target	has LSL	has USL	EngineeringUnit
ApricotJam-A	Fruite	Apricot			
	RatioFruite		50.00	52.00	Percent
	RatioSweetener		56.00	58.00	Percent
	Weight		248.00	252.00	Gramm
ApricotJam-B	Fruite	Apricot			
	RatioFruite		55.00	58.00	Percent
	RatioSweetener		55.00	57.00	Percent
	Weight		125.00	127.00	Gramm
BakingPowder-A	NeutralizingValue		68.00	70.00	
	RateOfReaction		39.00	40.00	Percent
	Weight		15.00	15.00	Gramm
BakingPowder-B	NeutralizingValue		67.00	70.00	
	RateOfReaction		39.00	40.00	Percent
	RatioAir		15.00	15.00	Gramm
StrawberryJam-A	Fruite	Strawberry			
	RatioFruite		50.00	52.00	Percent
	RatioSweetener		56.00	58.00	Percent
	Weight		248.00	252.00	Gramm

Table 4.9: Test data for demonstration of reasoning of *isAProductCategory* and *specializesProductCategory*.

Invoking `assertRelationOfSpecificationTargets` as the next step, performs a pairwise comparison of the *SpecificationTargets* :ApricotJam-Fruite, :StrawberryJam-Fruite, :ApricotJam-A-Fruite, :ApricotJam-B-Fruite, :StrawberryJam-A-Fruite (with respect to the extracted subset in Table 4.9). Recalling the previously proven logic of this procedure, the following relations with respect to *enclosesSpecificationTarget* are asserted.

```
:ApricotJam-Fruite kr:enclosesSpecificationTarget :ApricotJam-A-Fruite .
:ApricotJam-A-Fruite kr:enclosesSpecificationTarget :ApricotJam-Fruite .
:ApricotJam-Fruite kr:enclosesSpecificationTarget :ApricotJam-B-Fruite .
:ApricotJam-B-Fruite kr:enclosesSpecificationTarget :ApricotJam-Fruite .
:ApricotJam-A-Fruite kr:enclosesSpecificationTarget :ApricotJam-B-Fruite .
:ApricotJam-B-Fruite kr:enclosesSpecificationTarget :ApricotJam-A-Fruite .
:ApricotJam-Fruite kr:enclosesSpecificationTarget :ApricotJam-Fruite .
:ApricotJam-A-Fruite kr:enclosesSpecificationTarget :ApricotJam-A-Fruite .
:ApricotJam-B-Fruite kr:enclosesSpecificationTarget :ApricotJam-B-Fruite .
:StrawberryJam-Fruite kr:enclosesSpecificationTarget :StrawberryJam-A-Fruite .
:StrawberryJam-A-Fruite kr:enclosesSpecificationTarget :StrawberryJam-Fruite .
:StrawberryJam-Fruite kr:enclosesSpecificationTarget :StrawberryJam-Fruite .
:StrawberryJam-A-Fruite kr:enclosesSpecificationTarget :StrawberryJam-A-Fruite .
```

The subsequent step of standard reasoning results in respective assertions with regard to the superior object properties `kr:enclosesSpecification` and `kr:encloses`.

The next hybrid step to be invoked is `assertRelationOfSpecificationSets`. In this particular part of the evaluation, the concepts *ProductCategory* and *Product* represent the specializations of *SpecificationSet* which are utilized in order to verify the logic of DP_{1,3}. The determined results can be generalized to all applications of *RelationOfSpecificationSets* on more specific conceptual layers of K-RAMP's information model. These applications of the concept are discussed as part of the further steps of the evaluation process.

The previously proven logic of `assertRelationOfSpecificationSets` causes the assertion of individuals of *RelationOfSpecificationSets*. Some representative examples are discussed next. Recalling the *Specifications* of *ProductCategory* "ApricotJam" and the *Products* "ApricotJam-A" and "StrawberryJam-A" in Table 4.9, the following two individuals of *RelationOfSpecificationSets* are representing information about the relation between "ApricotJam" and "ApricotJam-A" respectively between "ApricotJam" and "StrawberryJam-A".

```
:ApricotJam-ApricotJam-A rdf:type kr:RelationOfSpecificationSets,
                           owl:NamedIndividual;
  kr:ratioEncloses 100.0;
  kr:ratioCommonCharacteristics 100.0;
  kr:fromSpecificationSet :ApricotJam;
  kr:toSpecificationSet :ApricotJam-A;
  kr:commonCharacteristic :Fruite,
                           :RatioFruite,
                           :RatioSweetener .

:ApricotJam-A-ApricotJam rdf:type kr:RelationOfSpecificationSets,
                              owl:NamedIndividual;
  kr:ratioEncloses 25.0;
  kr:ratioCommonCharacteristics 75.0;
  kr:fromSpecificationSet :ApricotJam-A;
  kr:toSpecificationSet :ApricotJam;
  kr:commonCharacteristic :Fruite,
                           :RatioFruite,
                           :RatioSweetener .

:ApricotJam-StrawberryJam-A rdf:type kr:RelationOfSpecificationSets,
                                owl:NamedIndividual;
  kr:ratioEncloses 67.0;
  kr:ratioCommonCharacteristics 100.0;
  kr:fromSpecificationSet :ApricotJam;
  kr:toSpecificationSet :StrawberryJam-A;
  kr:commonCharacteristic :Fruite,
                           :RatioFruite,
                           :RatioSweetener .

:StrawberryJam-A-ApricotJam rdf:type kr:RelationOfSpecificationSets,
                                owl:NamedIndividual;
  kr:ratioEncloses 0.0;
  kr:ratioCommonCharacteristics 75.0;
  kr:fromSpecificationSet :StrawberryJam-A;
  kr:toSpecificationSet :ApricotJam;
  kr:commonCharacteristic :Fruite,
                           :RatioFruite,
                           :RatioSweetener .
```

Another example comprises the *Specifications* of the two *Products* "ApricotJam-A" and "BakingPowder-A", which should not be significantly related with each other. The following two

individuals of *RelationOfSpecificationSets* are representing the information of their mutual relations.

```
kr-comm:ApricotJam-A-BakingPowder-A rdf:type kr:RelationOfSpecificationSets,
                                     owl:NamedIndividual;
kr:ratioEncloses 0.0;
kr:ratioCommonCharacteristics 25.0;
kr:fromSpecificationSet kr-comm:ApricotJam-A;
kr:toSpecificationSet kr-comm:BakingPowder-A;
kr:commonCharacteristic kr-comm:Weight .
```

```
kr-comm:BakingPowder-A-ApricotJam-A rdf:type kr:RelationOfSpecificationSets ,
                                     owl:NamedIndividual;
kr:ratioEncloses 0.0;
kr:ratioCommonCharacteristics 33.0;
kr:toSpecificationSet kr-comm:ApricotJam-A;
kr:fromSpecificationSet kr-comm:BakingPowder-A;
kr:commonCharacteristic kr-comm:Weight .
```

The step of standard reasoning to follow `assertRelationOfSpecificationSets` involves the so asserted individuals of *RelationOfSpecificationSet* for reasoning of further immediate relations between *SpecificationSets*. Due to the specification of DP_{1,3}, these are relations with respect to the object properties *enclosesSpecificationSet*, *overlapsSpecificationSet* and *partiallySpecificationSet* as well as *encloses*, *overlaps* and *partially*. Keeping the specialization hierarchies of *Products* and *ProductCategories* in mind, the specification of DP_{1,6} is reasoning relations with respect to the object properties *specializesProductCategory*, *specializes*, *isAProductCategory* and *isA*.

Based on the previous examples, the resulting relations are listed in the sequel. It has to be kept in mind, that there are significantly more relations due to the sole content of Table 4.9, including self-references.

```
:ApricotJam kr:enclosesSpecificationSet :ApricotJam-A .
:ApricotJam kr:overlapsSpecificationSet :StrawberryJam-A .
:ApricotJam-A kr:partiallySpecificationSet :ApricotJam .
:ApricotJam-A kr:partiallySpecificationSet :BakingPowder-A .
:StrawberryJam-A kr:overlapsSpecificationSet :ApricotJam-A .
:StrawberryJam-A kr:partiallySpecificationSet :ApricotJam .
:BakingPowder-A kr:partiallySpecificationSet :ApricotJam-A .
:ApricotJam kr:encloses :ApricotJam-A .
:ApricotJam kr:overlaps :StrawberryJam-A .
:ApricotJam-A kr:partially :ApricotJam .
:ApricotJam-A kr:partially :BakingPowder-A .
:StrawberryJam-A kr:overlaps :ApricotJam-A .
:StrawberryJam-A kr:partially :ApricotJam .
:BakingPowder-A kr:partially :ApricotJam-A .
:ApricotJam-A kr:isAProductCategory :ApricotJam .
:ApricotJam-A kr:isA :ApricotJam .
```

The final step according to the hybrid specification of PV_{1,3} is the determination and assertion of relations between individuals of *RelationOfSpecificationSets* with respect to the object property *subsetOf*. Considering the previous examples of relations concerning *encloses*, *overlaps* and *partially*, the following assertions are resulting from this step for instance.

```
:BakingPowder-A-ApricotJam-A kr:subsetOf :BakingPowder-A-ApricotJam-A .
:BakingPowder-A-ApricotJam-A kr:subsetOf :ApricotJam-A-BakingPowder-A .
:ApricotJam-A-ApricotJam kr:subsetOf :ApricotJam-A-ApricotJam .
:ApricotJam-A-ApricotJam kr:subsetOf :ApricotJam-A-StrawberryJam-A .
```



```
:ApricotJam-StrawberryJam-A kr:subsetOf :ApricotJam-A-StrawberryJam-A .  
:ApricotJam-A-BakingPowder-A kr:subsetOf :ApricotJam-A-BakingPowder-A .
```

In the bold printed example, `:ApricotJam-StrawberryJam-A` has `kr:commonCharacteristics :Fruite, :RatioFruite, :RatioSweetener`, and `:ApricotJam-A-StrawberryJam-A` has `kr:commonCharacteristics :Fruite, :RatioFruite, :RatioSweetener` and `:Weight`. After this step, the correct behavior of `PV1,3` in conjunction with `PV1,5` and `PV1,6` is proven. Also `PV1,1` and `PV1,2` as well as `PV1,4` are successfully evaluated. While these subtest groups can be performed in the common ontology model `kr-comm`, the final part of test group 1 – namely subtest group 1.iv – utilizes `kr-orig` already.

4.4.2 Ontology of original bakery

4.4.2.1 Overview

In the context of the original bakery's ABOX ontology model (`kr-orig`), the structure of the Viennese chocolate cake and, moreover, the required *Products* and *Process Capabilities* are specified as far as this information is transferred to the target bakery (a.k.a. the target production system). Information which shall not be exchanged with the target bakery is not relevant during the evaluation and therefore omitted in `kr-orig`. In the following subchapters, the specified information is described and the execution of subtest group 1.iv is described.

The evaluation phase which is discussed in this chapter is still focused on the device-independent part of the specified information model (decomposition of `DP1`) and procedures (decomposition of `PV1`). This evaluation phase continues the evaluation phase which is described in Chapter 4.4.1.

4.4.2.2 Process Capabilities and enclosing of Specification Sets

While the correctness of `PV1,3` is already proven during the previously described subtest groups in conjunction with `PV1,5` and `PV1,6`, this subchapter particularly addresses the relations which are required between *Products* respectively *ProductCategories* and *ProcessCapabilities* in conjunction with *SpecificationSets*. The utilized logic of *SpecificationSets* is exactly the same as it is also used in the previous subtest groups. Therefore, the following sections are purely focussing on `PV1,7` as well as the related information model of `DP1,7`. Moreover, this subchapter provides an insight to the preparation of device-independent process-related production basic data.

The form template of Microsoft® Excel® which is used for the preparation of *ProcessCapabilities* has the same structure as the one for *Products* respectively the one for *ProductCategories* (Figure 4.9). The only difference is due to two different sources of applicable target values. It is possible to apply target values which are already imported from `kr-comm`, or it is possible to use locally asserted target values. In this phase of the evaluation process, the local production system is represented by `kr-orig`, the ontology model of the original bakery.

Loading *Process Capability* data in conjunction with *Product Category* data and *Product* data to the Semantic Web database triggers the execution of `PV1` according to the already performed evaluation of the previous chapters. Table 4.10 shows some example data which are used throughout the next section.

The *ProcessCapability* “BakingCircularSpongeDough-240x50mm” is exemplified asserted as follows. The topics to be discussed are independent of the ontology model, as long as this ontology

model is based on the concepts of the K-RAMP TBOX models. Therefore, the namespace of the URIs is again omitted and replaced by “:”, although the discussed examples are materialized within `kr-orig`. But in these examples the namespace `kr-comm` indicates where individuals of the common ontology model are utilized.

	A	B	C	D	E	F
1	Process Capability	Characteristic (kr-comm)	Target (kr-comm)	Target (kr-orig)	LSL	USL
2	BakingCircularSpongeDough-240x50mm					
3	<i>BakingCircularSpongeDough-240x50mm</i>	<i>Diameter</i>			239	241
4	<i>BakingCircularSpongeDough-240x50mm</i>	<i>Height</i>			23	27
5	<i>BakingCircularSpongeDough-240x50mm</i>	<i>Weight</i>			815	820
6	<i>BakingCircularSpongeDough-240x50mm</i>	<i>Dough</i>				
7	<i>BakingCircularSpongeDough-240x50mm</i>	<i>Dough</i>	<i>SpongeDough</i>			
8	<i>BakingCircularSpongeDough-240x50mm</i>	<i>ColorOfDough</i>				
9	<i>BakingCircularSpongeDough-240x50mm</i>	<i>ColorOfDough</i>		<i>DarkBrown</i>		

Figure 4.9: Input of ProcessCapabilities.

Process Capability	Characteristic	SpecificationTarget	SpecificationRange		Engineering Unit
		has Target	has LSL	has USL	
BakingCircularSpongeDough-240x50mm	Diameter		239	241	Millimeter
	Height		23	27	Millimeter
	Weight		815	820	Gramm
	Dough	SpongeDough			
	ColorOfDough	DarkBrown			
RoomTempering-22degC-SpongeDough-ChocolateGloss	Coating	DarkChocolateCouverture			
	CoatingThickness		3	5	Millimeter
BeatingYolkButterCreme-460g-LiquidDarkChocolate	RatioFat		20	26	Percent
	RatioWater		3	5	Percent
	RatioAir		30	40	Percent
	RatioChocolate		12	18	Percent
	RatioYolk		8	10	Percent
Product	Characteristic	has Target	has LSL	has USL	Engineering Unit
VienneseChocolateCake-Circular-245x60mm-700g	Diameter		244	246	Millimeter
	Height		59	61	Millimeter
	Weight		690	710	Gramm
	Dough	ChocolateSpongeDough			
	ColorOfCoating	DarkBrown			
	ColorOfCoating	Brown			
	Coating	DarkChocolateCouverture			
	Filling	ApricotJam			
	CoatingThickness		3	5	Millimeter
	DoughDensity		0.3	0.5	GrammPerCubicCentimeter
ProductCategory	Characteristic	has Target	has LSL	has USL	Engineering Unit
BakedVienneseDough	Dough	ChocolateSpongeDough			
	ColorOfDough	DarkBrown			
DoughWithDarkChocolateCouverture	Coating	DarkChocolateCouverture			
	CoatingThickness		3	5	Millimeter

Table 4.10: Test data for demonstration of reasoning of *Product_requires* and *ProductCategory_requires*.

```

:BakingCircularSpongeDough-240x50mm rdf:type
    kr:ProcessCapability, owl:NamedIndividual;
kr:SpecificationSet_comprisesSpecification
    :BakingCircularSpongeDough-240x50mm-ColorOfDough,
    :BakingCircularSpongeDough-240x50mm-Diameter,
    :BakingCircularSpongeDough-240x50mm-Dough,
    :BakingCircularSpongeDough-240x50mm-Height,
    :BakingCircularSpongeDough-240x50mm-Weight .

:BakingCircularSpongeDough-240x50mm-ColorOfDough rdf:type
    kr:SpecificationTarget, owl:NamedIndividual;
kr:onCharacteristic kr-comm:ColorOfDough;
kr:hasTarget :DarkBrown .

:BakingCircularSpongeDough-240x50mm-Diameter rdf:type
    kr:SpecificationRange, owl:NamedIndividual;
kr:onCharacteristic kr-comm:Diameter;
kr:hasLSL 239;
kr:hasUSL 241 .

:BakingCircularSpongeDough-240x50mm-Dough rdf:type
    kr:SpecificationTarget, owl:NamedIndividual;
kr:onCharacteristic kr-comm:Dough;
kr:hasTarget kr-comm:SpongeDough .

:BakingCircularSpongeDough-240x50mm-Height rdf:type
    kr:SpecificationRange, owl:NamedIndividual;
kr:onCharacteristic kr-comm:Height;
kr:hasLSL 23;
kr:hasUSL 27 .

:BakingCircularSpongeDough-240x50mm-Weight rdf:type
    kr:SpecificationRange, owl:NamedIndividual;
kr:onCharacteristic kr-comm:Weight;
kr:hasLSL 815;
kr:hasUSL 820 .

```

By performing `assertRelationOfSpecificationSets` respective individuals of *RelationOfSpecificationSets* are asserted. The correctness of this procedure is already demonstrated in the previous chapters. Some exemplary assertions are listed in the sequel and are used for further discussion.

```

:BakedVienneseDough-BakingCircularSpongeDough-240x50mm rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
kr:ratioEncloses 100.0;
kr:ratioCommonCharacteristics 100.0;
kr:commonCharacteristic kr-comm:ColorOfDough, kr-comm:Dough;
kr:toSpecificationSet :BakingCircularSpongeDough-240x50mm;
kr:fromSpecificationSet :BakedVienneseDough .

:VienneseChocolateCake-Circular-245x60mm-700g-BakingCircularSpongeDough-240x50mm
    rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
kr:ratioEncloses 22.0;
kr:ratioCommonCharacteristics 44.0;
kr:commonCharacteristic kr-comm:Diameter, kr-comm:Dough, kr-comm:Height,
    kr-comm:Weight;
kr:toSpecificationSet :BakingCircularSpongeDough-240x50mm;
kr:fromSpecificationSet :VienneseChocolateCake-Circular-245x60mm-700g .

```

The standard reasoning as specified by DP_{1,3} results to assertions of *enclosesSpecificationSet*, *overlapsSpecificationSet* or *partiallySpecificationSet* respectively to reasoning of the superior object properties *encloses*, *overlaps* and *partially*.

```

:VienneseChocolateCake-Circular-245x60mm-700g kr:partiallySpecificationSet
      :BakingCircularSpongeDough-240x50mm .
:VienneseChocolateCake-Circular-245x60mm-700g kr:partiallySpecificationSet
      :RoomTempering-22degC-SpongeDough-ChocolateGloss .
:BakedVienneseDough kr:enclosesSpecificationSet
      :BakingCircularSpongeDough-240x50mm .
:VienneseChocolateCake-Circular-245x60mm-700g kr:partially
      :BakingCircularSpongeDough-240x50mm .
:VienneseChocolateCake-Circular-245x60mm-700g kr:partially
      :RoomTempering-22degC-SpongeDough-ChocolateGloss .
:BakedVienneseDough kr:encloses :BakingCircularSpongeDough-240x50mm .
:BakedVienneseDough kr:ProductCategory_requires
      :BakingCircularSpongeDough-240x50mm .
:BakedVienneseDough kr:requires :BakingCircularSpongeDough-240x50mm .

```

According to these assertions, there is the *ProductCategory* “BakedVienneseDough” which requires the *ProcessCapability* “BakingCircularSpongeDough-240x50mm”. This assumption is correct. “BakingCircularSpongeDough-240x50mm” guarantees the delivery of a “Dough ChocolateSpongeDough” with “DarkBrown” color. However, the result of these assertions is not perfect.

The situation of the *Product* “VienneseChocolateCake-Circular-245x60mm-700g” shall be discussed in a broader context for this purpose. Not visible due to the example data, “VienneseChocolateCake-Circular-245x60mm-700g” is composed from the subordinated *Product* “VienneseChocolateCake-Circular-245x60mm-700g-LiquidGloss”. The *ProcessCapability* “RoomTempering-22degC-SpongeDough-ChocolateGloss” actually guarantees a solid gloss at room temperature in case of “CoatingThickness” between 3 and 5 millimeter. But the *Specifications* of the *Product* and the *ProcessCapability* deviate too much.

Recalling the *Specifications* of “VienneseChocolateCake-Circular-245x60mm-700g”, also information about the composition structure of the *Product* can be found, like “Dough”, “ColorOfCoating”, “DoughDensity” or “Filling”. In the current situation, these *Specifications* cause a low ratio of common *Characteristics* and enclosed *Specifications*. However, *Specifications* with the same values are rather likely also part of subproducts. Eliminating these *Specifications* may thus increase the chance to match with *ProcessCapabilities*. There is still a hidden part of the truth with this approach because subsequent *ProcessCapabilities*, due to subsequent composition steps, may still require these *Specifications*. Choosing the *Specifications* of a *Product* by focusing on an increased chance of matching with *ProcessCapabilities* is therefore no good approach.

What about the *ProcessCapability*’s perspective? In the particular case of the *ProcessCapability* “RoomTempering-22degC-SpongeDough-ChocolateGloss” and its utilized *Specifications* it becomes obvious that this *ProcessCapability* guarantees a solid “DarkChocolateCouverture” in case of “CoatingThickness” between 3 to 5 millimeters. The dough of the cake or its size is of no relevance for this *ProcessCapability*. All *Products* of a potential *ProductCategory* “DoughWithDarkChocolateCouverture” are guaranteed. The introduction of a respective *ProductCategory* is therefore useful. The respective assertions of individuals are listed below, and it is obviously a *SpecificationSet* which encloses “RoomTempering-22degC-SpongeDough-ChocolateGloss” and “VienneseChocolateCake-Circular-245x60mm-700g”.

```

:DoughWithDarkChocolateCouverture rdf:type
      kr:ProductCategory, owl:NamedIndividual;

```

```

kr:SpecificationSet_comprisesSpecification
    :DoughWithDarkChocolateCouverture-Coating,
    :DoughWithDarkChocolateCouverture-CoatingThickness .

:DoughWithDarkChocolateCouverture-Coating rdf:type
    kr:SpecificationTarget, owl:NamedIndividual;
kr:onCharacteristic kr-comm:Coating;
kr:hasTarget kr-comm:DarkChocolateCouverture .

:DoughWithDarkChocolateCouverture-CoatingThickness rdf:type
    kr:SpecificationRange, owl:NamedIndividual;
kr:onCharacteristic kr-comm:CoatingThickness;
kr:hasLSL 3;
kr:hasUSL 5 .

```

The following individuals of *RelationOfSpecificationSets* can be asserted accordingly.

```

:DoughWithDarkChocolateCouverture-RoomTempering-22degC-SpongeDough-
ChocolateGloss rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
kr:ratioEncloses 100.0;
kr:ratioCommonCharacteristics 100.0;
kr:commonCharacteristic kr-comm:Coating, kr-comm:CoatingThickness;
kr:fromSpecificationSet :DoughWithDarkChocolateCouverture;
kr:toSpecificationSet :RoomTempering-22degC-SpongeDough-ChocolateGloss .

:DoughWithDarkChocolateCouverture-VienneseChocolateCake-Circular-245x60mm-700g
    rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
kr:ratioEncloses 100.0;
kr:ratioCommonCharacteristics 100.0;
kr:commonCharacteristic kr-comm:Coating, kr-comm:CoatingThickness;
kr:fromSpecificationSet :DoughWithDarkChocolateCouverture;
kr:toSpecificationSet :VienneseChocolateCake-Circular-245x60mm-700g .

```

The standard reasoning on rules which are specified by DP_{1,3} but also DP_{1,6} and DP_{1,7} are leading now to the following additional relations with respect to *enclosesSpecificationSet*, *encloses*, *ProductCategory_requires*, *Product_requires* and *requires*.

```

:DoughWithDarkChocolateCouverture kr:enclosesSpecificationSet
    :RoomTempering-22degC-SpongeDough-ChocolateGloss .
:DoughWithDarkChocolateCouverture kr:enclosesSpecificationSet
    :VienneseChocolateCake-Circular-245x60mm-700g .
:DoughWithDarkChocolateCouverture kr:encloses
    :RoomTempering-22degC-SpongeDough-ChocolateGloss .
:DoughWithDarkChocolateCouverture kr:encloses
    :VienneseChocolateCake-Circular-245x60mm-700g .
:DoughWithDarkChocolateCouverture kr:requires
    :RoomTempering-22degC-SpongeDough-ChocolateGloss .
:VienneseChocolateCake-Circular-245x60mm-700g kr:isAProductCategory
    :DoughWithDarkChocolateCouverture .
:VienneseChocolateCake-Circular-245x60mm-700g kr:isA
    :DoughWithDarkChocolateCouverture .
:VienneseChocolateCake-Circular-245x60mm-700g kr:Product_requires
    :RoomTempering-22degC-SpongeDough-ChocolateGloss .
:VienneseChocolateCake-Circular-245x60mm-700g kr:requires
    :RoomTempering-22degC-SpongeDough-ChocolateGloss .

```

When a *ProcessCapability* can be used for multiple *Products* in general, an appropriate *ProductCategory* should be introduced which conforms to the *Specifications* of the

ProcessCapability. The essential aim of the specification of DP_{1,7} is the implication of statements of the form *Product requires ProcessCapability* respectively *ProductCategory requires ProcessCapability*. There are two paths for the implication of *Product requires ProcessCapability*.

1. Immediate reasoning path – if a *Product* encloses a *ProcessCapability*.
2. Distant reasoning path – if a *Product* is categorized by a *ProductCategory* and this *ProductCategory* encloses a *ProcessCapability*. It shall be highlighted that if a *Product* is categorized by a *ProductCategory* it is also categorized by all its generalizations.

The immediate reasoning path is based on the fact that *Product* and *ProcessCapability* are both subclasses of *SpecificationSet*. According to the specification of DP_{1,7}, a *Product* or a *ProductCategory* implicitly require a *ProcessCapability* if they enclose this *ProcessCapability*.

D2		fx		=WENN(UND(A2<>"";ODER(B2<>"";C2<>"")); "Declaration(NamedIndividual(kr-orig:"&A2&"))"&ZEICHEN(10)& "ClassAssertion(kr:ProcessSegment kr-orig:"&A2&"))"&ZEICHEN(10)& "ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:"&A2& WENN(B2<>"";" kr-orig:"&B2;" kr-comm:"&C2)&")"; ""))
A	B	C	D	
1 ProcessSegment	isAProductionTechnique (kr-orig)	isAProductionTechnique (kr-comm)	OWL	
2 Baking-InternalFormer	BakingChocolateSpongeDough		Declaration(NamedIndividual(kr-orig:Baking-InternalFormer)) ClassAssertion(kr:ProcessSegment kr-orig:Baking-InternalFormer) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:Baking-InternalFormer kr-orig:BakingChocolateSpongeDough)	
3 BoilingLiquid-HotPlate		BoilingWaterbasedLiquid	Declaration(NamedIndividual(kr-orig:BoilingLiquid-HotPlate)) ClassAssertion(kr:ProcessSegment kr-orig:BoilingLiquid-HotPlate) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:BoilingLiquid-HotPlate kr-comm:BoilingWaterbasedLiquid)	
4 Coating-Spatula		CoatingWithCouverture	Declaration(NamedIndividual(kr-orig:Coating-Spatula)) ClassAssertion(kr:ProcessSegment kr-orig:Coating-Spatula) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:Coating-Spatula kr-comm:CoatingWithCouverture)	
5 Coating-SpatulaFull		CoatingWithCouverture	Declaration(NamedIndividual(kr-orig:Coating-SpatulaFull)) ClassAssertion(kr:ProcessSegment kr-orig:Coating-SpatulaFull) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:Coating-SpatulaFull kr-comm:CoatingWithCouverture)	
6 Coating-Pour		CoatingWithCouverture	Declaration(NamedIndividual(kr-orig:Coating-Pour)) ClassAssertion(kr:ProcessSegment kr-orig:Coating-Pour) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:Coating-Pour kr-comm:CoatingWithCouverture)	
7 Folding-DoughScraper		FoldingMixtureAndEggWhite	Declaration(NamedIndividual(kr-orig:Folding-DoughScraper)) ClassAssertion(kr:ProcessSegment kr-orig:Folding-DoughScraper) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:Folding-DoughScraper kr-comm:FoldingMixtureAndEggWhite)	
8 StiringLiquid-HotPlate		StiringWaterBasedLiquid	Declaration(NamedIndividual(kr-orig:StiringLiquid-HotPlate)) ClassAssertion(kr:ProcessSegment kr-orig:StiringLiquid-HotPlate) ObjectPropertyAssertion(kr:isAProductionTechnique kr-orig:StiringLiquid-HotPlate kr-comm:StiringWaterBasedLiquid)	

Figure 4.10: Input of *ProcessSegments* and their assignment to *ProductionTechniques*, the generated statements in OWL2 functional syntax and the generation logic.

The distant reasoning path is specified in detail in Chapter 3.2.3.5 and is proven due to the previous sections. Consequently, the complete test group 1 is proven.

4.4.2.3 Categorization of Process Segments

The *ProcessSegments* of the original bakery satisfy particular *ProcessCapabilities* which are required by the specific *Products* and *ProductCategories* at the original bakery. Figure 4.9 shows

how *ProcessCapabilities* are asserted for `kr-orig` accordingly. The detailed structure of the *ProcessSegments* is not of any interest with respect to the transfer of production knowledge. However, the categorization schema of the *ProcessSegments* through *ProductionTechniques* can be used to determine similarities between *ProcessCapabilities* of different production systems (Figure 3.3) but also already between *ProcessCapabilities* of the same production system. For this reason, *ProcessSegments* of the original bakery and their categorization with *ProductionTechniques* are prepared as part of `kr-orig` (Figure 4.10) and are discussed in the following sections in order to perform subtest group 2.i. This step does not require extensive evaluation but it results in prerequisite information needed in the context of subtest group 2.ii.

For the purpose of evaluation, with one exception, *ProductionTechniques* of `kr-comm` are applied. The exceptional *ProductionTechnique* “BakingChocolateSpongeDough” is used to verify that the specialization structure of *ProductionTechniques* is also applied correctly during reasoning for similar *ProcessSegments* across two ontology models. It is actually equivalent to verify this behavior between `kr-comm` and `kr-orig` or between `kr-orig` and `kr-targ` because the essential logic is provided by the commonly applied TBOX ontology models of K-RAMP in accordance with the specification of DP_{3,1}.

According to Figure 4.10, *ProcessSegments* are asserted and categorized explicitly through *ProductionTechniques* of the original bakery or of the commonly specified *ProductionTechniques* of `kr-comm`. Some exemplary assertions which are resulting from uploading this ontology model to the Semantic Web database are listed in the sequel. The namespace of `kr-orig` is again omitted.

```
:Baking-InternalFormer rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isAProductionTechnique :BakingChocolateSpongeDough .
:BeatingMixture-HandMixer rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isAProductionTechnique kr-comm:BeatingEggWhite .
:BeatingMixture-KitchenMachine rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isAProductionTechnique kr-comm:BeatingMixture .
:BoilingLiquid-HotPlate rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isAProductionTechnique kr-comm:BoilingWaterbasedLiquid .
:Coating-Pour rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isAProductionTechnique kr-comm:CoatingWithCouverture .
```

Standard reasoning based on implication rules which are specified by DP_{3,1} are leading to further assertions also along the generalization hierarchy of *ProductionTechniques*.

```
:Baking-InternalFormer rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA :BakingChocolateSpongeDough .
:Baking-InternalFormer rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA :Baking .
:BeatingMixture-HandMixer rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:BeatingEggWhite .
:BeatingMixture-HandMixer rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:Beating .
:BeatingMixture-KitchenMachine rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:BeatingMixture .
:BeatingMixture-KitchenMachine rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:Beating .
:BoilingLiquid-HotPlate rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:BoilingWaterbasedLiquid .
:BoilingLiquid-HotPlate rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:Boiling .
:Coating-Pour rdf:type kr:ProcessSegment, owl:NamedIndividual;
    kr:isA kr-comm:CoatingWithCouverture .
```

```
:Coating-Pour rdf:type kr:ProcessSegment, owl:NamedIndividual;
kr:isA kr-comm:Coating .
```

Subtest group 2.i can demonstrate easily the upload of *ProcessSegments* the assignment of *ProductionTechniques* and the reasoning of relations with respect to *isA*.

4.4.2.4 Similarities of ProcessCapabilities

According to the information model of DP_{3,2} (Figure 3.15), *ProcessSegmentSetups* are the hub for reasoning on several implication rules which can be still evaluated within the context of a single production system. For the purpose of this evaluation, there is no need for matchmaking between ontology models of the original production system and the target production system. The next verification steps are therefore all in conjunction with *ProcessSegmentSetups* and are addressing subtest group 2.ii.

Starting from uploading *ProcessSegmentSetup* data to the Semantic Web database, the correctness of implication rules which are specified by DP_{3,2} is verified. As the final step of this evaluation phase, the successful reasoning and assertion of individuals of *SimilarityOfProcessCapabilities* is verified. As there are also similar *ProcessCapabilities* within the original production system the evaluation can be already performed in the context of *kr-orig*. Because all reasoning logic is either part of the TBOX ontology model or the hybrid components of PV_{3,2}, the result of this evaluation step is also applicable for *kr-targ* or for the combination of both ontology models after matchmaking.

Figure 4.11 shows an excerpt of the input form in Microsoft® Excel® which is used to specify *ProcessSegmentSetups* of *kr-orig*. First, each useful combination of a *ProcessSegment* and a *ProcessCapability* is connected by at least one *ProcessSegmentSetup*. The URIs of the connected *ProcessSegment* and *ProcessCapability* are used in combination with a sequential number (num) in order to generate the URI of the *ProcessSegmentSetup*①. The *ProductCategoryUsages* are specified next②, and this step is followed by the formal description of their detailed *Specifications*③.

After uploading of these process basic data to the Semantic Web database, a step of standard reasoning is performed in accordance to the hybrid specification of PV_{3,2}. This step of standard reasoning results in a series of asserted relations which are finally leading to the essential outcome of this step – the assertion of relations with respect to *needsUsageOf* and *similarProcessCapability*. An excerpt of these assertions is listed in the sequel.

```
:Baking-InternalFormer kr:hasProcessSegmentSetup
    :Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1 .

:Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1
    rdf:type kr:ProcessSegmentSetup,
            owl:NamedIndividual;
kr:enables :BakingCircularSpongeDough-240x50mm;
kr:hasProductCategoryUsage
    :Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-
    LiquidVienneseDough .

:BeatingMixture-HandMixer kr:hasProcessSegmentSetup
    :BeatingMixture-HandMixer4BeatingEggWhites-215g-Sugar-1 .
```

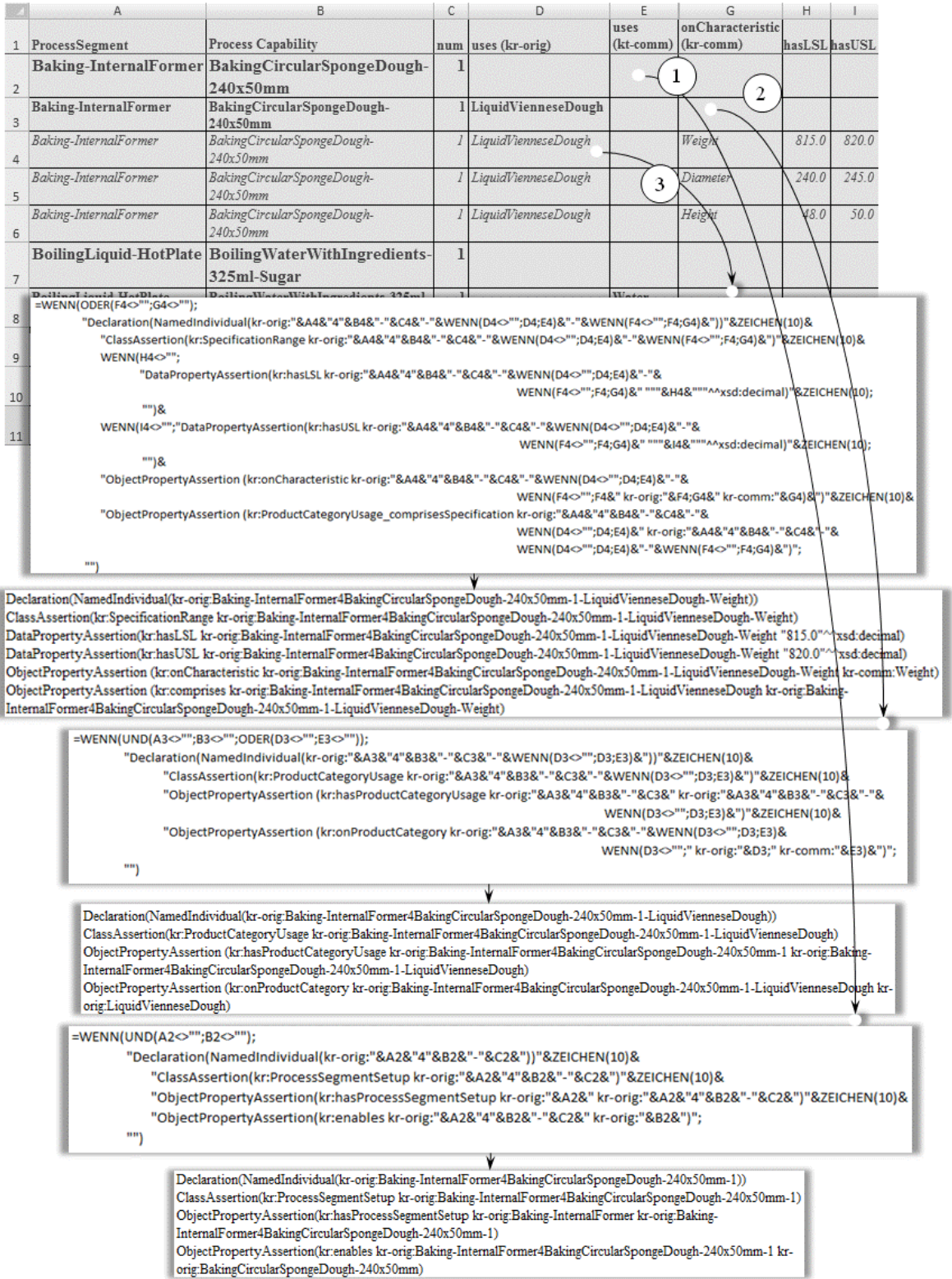



Figure 4.11: Input of ProcessSegmentSetups, ProductCategoryUsages, Specifications and the generation logic and resulting OWL2 functional syntax.

```

:BeatingMixture-HandMixer4BeatingEggWhites-215g-Sugar-1
    rdf:type kr:ProcessSegmentSetup,
           owl:NamedIndividual;
kr:enables :BeatingEggWhites-215g-Sugar;
kr:hasProductCategoryUsage
    :BeatingMixture-HandMixer4BeatingEggWhites-215g-Sugar-1-EggWhite,
    :BeatingMixture-HandMixer4BeatingEggWhites-215g-Sugar-1-
    GranulatedSugar .

:BeatingMixture-KitchenMachine kr:hasProcessSegmentSetup
    :BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1,
    :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1,
    :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-
    LiquidDarkChocolate-1 .

:BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1
    rdf:type kr:ProcessSegmentSetup,
           owl:NamedIndividual;
kr:enables :BeatingButter-270g-Sugar-VanillaFlavor;
kr:hasProductCategoryUsage
    :BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1-
    PowderSugar,
    :BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1-
    SoftButter,
    :BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1-
    VanillaSugar .

:BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1
    rdf:type kr:ProcessSegmentSetup,
           owl:NamedIndividual;
kr:enables :BeatingYolkButterCreme-330g;
kr:hasProductCategoryUsage
    :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1-
    ButterCremeForVienneseDough,
    :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1-Yolk .

:BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-LiquidDarkChocolate-1
    rdf:type kr:ProcessSegmentSetup,
           owl:NamedIndividual;
kr:enables :BeatingYolkButterCreme-460g-LiquidDarkChocolate;
kr:hasProductCategoryUsage
    :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-
    LiquidDarkChocolate-1-LiquidDarkChocolate,
    :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-
    LiquidDarkChocolate-1-YolkButterCremeForVienneseDough .

```

These exemplary assertions underline once more the role of *ProcessSegmentSetups*. The *ProcessSegment* “BeatingMixture-KitchenMachine” has three different *ProcessSegmentSetups*. Each of them enables another *ProcessCapability* and invokes the usage of material from different *ProductCategories*. Upon the previously listed uploaded process basic data, the following assertions are performed due to standard reasoning based on the specification of DP_{3,2}.

```

:Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1
    kr:uses :VienneseDough;
kr:onProcessSegment :Baking-InternalFormer .

:BeatingMixture-HandMixer4BeatingEggWhites-215g-Sugar-1 .
kr:uses :GranulatedSugar, :EggWhite;
kr:onProcessSegment :BeatingMixture-HandMixer .

```

```

:BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1 .
  kr:uses :PowderSugar, :SoftButter, :VanillaSugar;
  kr:onProcessSegment :BeatingMixture-KitchenMachine .

:BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1
  kr:uses :ButterCremeForVienneseDough, :Yolk;
  kr:onProcessSegment :BeatingMixture-KitchenMachine .

:BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-LiquidDarkChocolate-1
  kr:uses :LiquidDarkChocolate, :YolkButterCremeForVienneseDough;
  kr:onProcessSegment :BeatingMixture-KitchenMachine .

:Baking-InternalFormer kr:satisfies :BakingCircularSpongeDough-240x50mm .

:BeatingMixture-HandMixer kr:satisfies :BeatingEggWhites-215g-Sugar .

:BeatingMixture-KitchenMachine kr:satisfies
  :BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-VanillaFlavor-1,
  :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1,
  :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-
  LiquidDarkChocolate-1 .

:BakingCircularSpongeDough-240x50mm
  kr:enabledBy :Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1;
  kr:supportedBy :BakingChocolateSpongeDough;
  kr:needsUsageOf :LiquidVienneseDough .

:BeatingEggWhites-215g-Sugar
  kr:enabledBy :BeatingMixture-HandMixer4BeatingEggWhites-215g-Sugar-1;
  kr:supportedBy :BeatingMixture;
  kr:needsUsageOf :GranulatedSugar, :EggWhite .

:BeatingButter-270g-Sugar-VanillaFlavor
  kr:enabledBy :BeatingMixture-KitchenMachine4BeatingButter-270g-Sugar-
  VanillaFlavor-1;
  kr:supportedBy :BeatingMixture;
  kr:similarProcessCapability
  :BeatingYolkButterCreme-330g,
  :BeatingYolkButterCreme-460g-LiquidDarkChocolate;
  kr:needsUsageOf :PowderSugar, :SoftButter, :VanillaSugar .

:BeatingYolkButterCreme-330g
  kr:enabledBy :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-330g-1;
  kr:supportedBy :BeatingMixture;
  kr:similarProcessCapability
  :BeatingYolkButterCreme-460g-LiquidDarkChocolate,
  :BeatingButter-270g-Sugar-VanillaFlavor;
  kr:needsUsageOf :ButterCremeForVienneseDough, :Yolk .

:BeatingYolkButterCreme-460g-LiquidDarkChocolate
  kr:enabledBy :BeatingMixture-KitchenMachine4BeatingYolkButterCreme-460g-
  LiquidDarkChocolate-1;
  kr:supportedBy :BeatingMixture;
  kr:similarProcessCapability
  :BeatingYolkButterCreme-330g,
  :BeatingButter-270g-Sugar-VanillaFlavor;
  kr:needsUsageOf :LiquidDarkChocolate, :YolkButterCremeForVienneseDough .

```

Based on this listing, it can be seen how applying of the reasoning rules of DP_{3,2} results in relations concerning *similarProcessCapability* between *ProcessCapabilities* as well as *needsUsageOf* between *ProcessCapabilities* and *ProductCategories*. Both represent the foundation

for subsequent evaluation steps. However, it is not yet discussed in detail how *needsUsageOf* is resolved.

This shall be discussed by example of the *ProductSegmentSetup* “Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1” which has a *ProductCategoryUsage* “Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-LiquidVienneseDough” according to one of the previous listings. The original input data of this example are also shown in Figure 4.11. The *Specifications* of this *ProductCategoryUsage* result in the following assertions.

```
:Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-LiquidVienneseDough
    rdf:type kr:ProductCategoryUsage,
           owl:NamedIndividual;
    kr:ProductCategoryUsage_comprisesSpecification
      :Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-
      LiquidVienneseDough-Diameter,
      :Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-
      LiquidVienneseDough-Height,
      :Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-
      LiquidVienneseDough-Weight;
    kr:onProductCategory :LiquidVienneseDough .

:Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-LiquidVienneseDough-
Diameter
    rdf:type kr:SpecificationRange,
           owl:NamedIndividual;
    kr:hasLSL 240.0;
    kr:hasUSL 245.0;
    kr:onCharacteristic kr-comm:Diameter .

:Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-LiquidVienneseDough-
Height
    rdf:type kr:SpecificationRange,
           owl:NamedIndividual;
    kr:hasLSL 48.0;
    kr:hasUSL 50.0;
    kr:onCharacteristic kr-comm:Height .

:Baking-InternalFormer4BakingCircularSpongeDough-240x50mm-1-LiquidVienneseDough-
Weight
    rdf:type kr:SpecificationRange,
           owl:NamedIndividual;
    kr:hasLSL 815.0;
    kr:hasUSL 820.0;
    kr:onCharacteristic kr-comm:Weight .
```

The previous exemplary assertions explains, how the connection to used *ProductCategories* is originally established. Moreover, it highlights the structure of *ProductCategoryUsages* as sets of *Specifications*.

The final step to be considered with respect to the hybrid specification of PV_{3.2} is the execution of the procedure *assertSimilarityOfProcessCapabilities* (see code listing (3,4)). The procedure is implemented straight forward and does not invoke any new features. Therefore, a formal verification of the procedure is omitted. The procedure is invoked in a nested loop ensuring pair-wise processing of all individuals of *ProcessCapability*.

Due to the first condition, the procedure is only executed for pair members with asserted similarity (see listings above). If similarity is asserted, relations with respect to *supportedBy* are gathered through SPARQL queries for both *ProcessCapabilities*. This is already proven as being

correct based on equivalent functions during processing of subtest group 1.iii. According to the information model of DP_{3,2}, there may be multiple *ProductionTechniques* as part of the result. However, both sets of *ProductionTechniques* may not be the same. But because of the asserted similarity, there are common members with both sets of *ProductionTechniques*. By intersection of both sets, the common set of *ProductionTechniques* is achieved. This common set causes the similarity of both *ProcessCapabilities*. Based on this information, it is now possible to assert an individual of *SimilarityOfProcessCapability* and relations with respect to the needed object properties. This step finalizes the execution of subtest group 2.ii, proves the correct execution of PV_{3,2} and provides the foundation for the subsequent evaluation steps. An exemplary snippet, which results from the similarity between “BeatingButter-270g-Sugar-VanillaFlavor” and “BeatingYolkButterCreme-330g”, is listed in the sequel.

```
:BeatingButter-270g-Sugar-VanillaFlavor-BeatingYolkButterCreme-330g
    rdf:type kr:SimilarityOfProcessCapability,
            owl:NamedIndividual;
    kr:onProductionTechnique :BeatingMixture;
    kr:onProcessCapability
        :BeatingButter-270g-Sugar-VanillaFlavor,
        :BeatingYolkButterCreme-330g .

:BeatingYolkButterCreme-330g-BeatingButter-270g-Sugar-VanillaFlavor
    rdf:type kr:SimilarityOfProcessCapability,
            owl:NamedIndividual;
    kr:onProductionTechnique :BeatingMixture;
    kr:onProcessCapability
        :BeatingButter-270g-Sugar-VanillaFlavor,
        :BeatingYolkButterCreme-330g .
```

4.4.3 Ontology of target bakery

4.4.3.1 Overview

The specification of product-related information and device-independent process-related information of *kr-targ* is equivalent to the described approach for *kr-orig* or *kr-comm* in the previous chapters. The specific product basic data of the target bakery are generated as *kr-targ* ontology model and uploaded to the Semantic Web database as already described for the test group 1 and the subtest groups 2.i and 2.ii. These aspects are not discussed in the following subchapters again. The essential evaluation steps of the subtest groups 2.iii to 2.v, which are documented in the following sections, are focusing on device-specific process-related information of the target bakery. As already highlighted for the previous evaluation steps, also the succeeding evaluation steps are valid for every ontology model which is derived from the concepts of the K-RAMP TBOX models. The resulting information model of the following evaluation steps is the foundation of the final test group which shall then evaluate the reuse of information after matchmaking of *kr-orig* and *kr-targ*.

4.4.3.2 Equipment

The first part of not yet uploaded process basic data comprises information about available equipment. Therefore, *kr-targ* is initialized with asserted equipment respectively with devices which are available at the target bakery (Figure 4.12). This step is purely based on standard specifications of the Semantic Web without any K-RAMP-specific reasoning logic. Therefore, no detail verification of this step is necessary. For this reason, the respective activities which are verified by subtest group 2.iii are considered as behaving correctly.

B2	=WENN(A2<>""; "Declaration (NamedIndividual (kr-targ:"&A2&"))"&ZEICHEN(10)& "ClassAssertion (kr:Equipment kr-targ:"&A2&")"; "")
A	B
1 Equipment	OWL
2 Oven-1	Declaration (NamedIndividual (kr-targ:Oven-1)) ClassAssertion (kr:Equipment kr-targ:Oven-1)
3 Oven-2	Declaration (NamedIndividual (kr-targ:Oven-2)) ClassAssertion (kr:Equipment kr-targ:Oven-2)
4 HotPlate-1	Declaration (NamedIndividual (kr-targ:HotPlate-1)) ClassAssertion (kr:Equipment kr-targ:HotPlate-1)
5 HotPlate-2	Declaration (NamedIndividual (kr-targ:HotPlate-2)) ClassAssertion (kr:Equipment kr-targ:HotPlate-2)
6 HotPlate-3	Declaration (NamedIndividual (kr-targ:HotPlate-3)) ClassAssertion (kr:Equipment kr-targ:HotPlate-3)
7 KitchenMachine-1	Declaration (NamedIndividual (kr-targ:KitchenMachine-1)) ClassAssertion (kr:Equipment kr-targ:KitchenMachine-1)
8 KitchenMachine-2	Declaration (NamedIndividual (kr-targ:KitchenMachine-2)) ClassAssertion (kr:Equipment kr-targ:KitchenMachine-2)

Figure 4.12: Input of Equipment, the generated statements in OWL2 functional syntax and the generation logic.

Some exemplary individuals of *Equipment* are listed in the sequel.

```
:Oven-1 rdf:type kr:Equipment, owl:NamedIndividual .
:Oven-2 rdf:type kr:Equipment, owl:NamedIndividual .
:HotPlate-1 rdf:type kr:Equipment, owl:NamedIndividual .
:HotPlate-2 rdf:type kr:Equipment, owl:NamedIndividual .
:HotPlate-3 rdf:type kr:Equipment, owl:NamedIndividual .
:KitchenMachine-1 rdf:type kr:Equipment, owl:NamedIndividual .
:KitchenMachine-2 rdf:type kr:Equipment, owl:NamedIndividual .
:HandMixer-1 rdf:type kr:Equipment, owl:NamedIndividual .
:HandMixer-2 rdf:type kr:Equipment, owl:NamedIndividual .
```

4.4.3.3 Equipment Recipes and Handling Instructions

Consequently, *EquipmentRecipes* and *HandlingInstructions* are specified as shown in Figure 4.13. For the purpose of evaluation, it is sufficient to use a *xsd:string*-type data property *hasContent* for storing descriptions of *EquipmentRecipes* or *HandlingInstructions* respectively. In real-world applications, it is possible without limitation of completeness to use a different data type – like *xsd:base64Binary*. Also for *EquipmentRecipes* and *HandlingInstructions* only standard specifications of the Semantic Web are applied during the upload to the Semantic Web database. According to the specification of DP_{3,3} and DP_{3,4}, there is no reasoning logic specified, and therefore no particular verification is necessary. For this reason, the respective activities which are verified by substest group 2.iv are considered as behaving correctly.

Some exemplary individuals of *EquipmentRecipe* and *HandlingInstruction* are listed in the sequel.

```
:Oven-170degC-Convection rdf:type kr:EquipmentRecipe, owl:NamedIndividual;
    kr:hasRecipeContent
        "set: target temperature: 170degC; duration: 10min;
        ventilation: on"^^xsd:string .

:Oven-PostHeat rdf:type kr:EquipmentRecipe, owl:NamedIndividual;
    kr:hasRecipeContent "set: heating: off; ventilation: on"^^xsd:string .
```


C2		
A	B	C
1	Equipment Recipe	Content
		OWL
2	Oven-170degC-Convection	set: target temperature: 170°C; duration: 10min; ventilation: on
		Declaration (NamedIndividual (kr-targ:Oven-170degC-Convection)) ClassAssertion (kr:EquipmentRecipe kr-targ:Oven-170degC-Convection) DataPropertyAssertion (kr:hasRecipeContent kr-targ:Oven-170degC-Convection "set: target temperature: 170°C; duration: 10min; ventilation: on"^^xsd:string)
3	Oven-PostHeat	set: heating: off; ventilation: on
		Declaration (NamedIndividual (kr-targ:Oven-PostHeat)) ClassAssertion (kr:EquipmentRecipe kr-targ:Oven-PostHeat) DataPropertyAssertion (kr:hasRecipeContent kr-targ:Oven-PostHeat "set: heating: off; ventilation: on"^^xsd:string)
4	HotPlate-Level-5	set: target temperature: 130°C
		Declaration (NamedIndividual (kr-targ:HotPlate-Level-5)) ClassAssertion (kr:EquipmentRecipe kr-targ:HotPlate-Level-5) DataPropertyAssertion (kr:hasRecipeContent kr-targ:HotPlate-Level-5 "set: target temperature: 130°C"^^xsd:string)

C2		
A	B	C
1	Handling Instruction	Content
		OWL
2	HowTo-PreHeat-Oven-170degC	(1) switch to 170degC; (2) switch on convection; (3) wait for 10 minutes
		Declaration (NamedIndividual (kr-targ:HowTo-PreHeat-Oven-170degC)) ClassAssertion (kr:HandlingInstruction kr-targ:HowTo-PreHeat-Oven-170degC) DataPropertyAssertion (kr:hasHandlingInstructionContent kr-targ:HowTo-PreHeat-Oven-170degC "(1) switch to 170degC; (2) switch on convection; (3) wait for 10 minutes"^^xsd:string)
3	HowTo-MainHeat-Oven-ChouxPastry	(1) pre-heat at 170degC; (2) leave oven door ajar for 10 minutes; (3) close oven door; (4) continue for 45 minutes
		Declaration (NamedIndividual (kr-targ:HowTo-MainHeat-Oven-ChouxPastry)) ClassAssertion (kr:HandlingInstruction kr-targ:HowTo-MainHeat-Oven-ChouxPastry) DataPropertyAssertion (kr:hasHandlingInstructionContent kr-targ:HowTo-MainHeat-Oven-ChouxPastry "(1) pre-heat at 170degC; (2) leave oven door ajar for 10 minutes; (3) close oven door; (4) continue for 45 minutes"^^xsd:string)
4	HowTo-MainHeat-Oven-SpongeDough	(1) pre-heat at 170degC; (2) leave oven door ajar for 10 minutes; (3) close oven door; (4) continue for 45 minutes
		Declaration (NamedIndividual (kr-targ:HowTo-MainHeat-Oven-SpongeDough)) ClassAssertion (kr:HandlingInstruction kr-targ:HowTo-MainHeat-Oven-SpongeDough) DataPropertyAssertion (kr:hasHandlingInstructionContent kr-targ:HowTo-MainHeat-Oven-SpongeDough "(1) pre-heat at 170degC; (2) leave oven door ajar for 10 minutes; (3) close oven door; (4) continue for 45 minutes"^^xsd:string)

Figure 4.13: Input of Equipment Recipes (top) and Handling Instructions (bottom), the generated statements in OWL2 functional syntax and the generation logic.

```

:HotPlate-Level-5 rdf:type kr:EquipmentRecipe, owl:NamedIndividual;
  kr:hasRecipeContent "set: target temperature: 130degC"^^xsd:string .
:HowTo-PreHeat-Oven-170degC rdf:type
  kr:HandlingInstruction, owl:NamedIndividual;
  kr:hasHandlingInstructionContent
    "(1) switch to 170degC; (2) switch on convection; (3) wait for 10
    minutes"^^xsd:string .

:HowTo-MainHeat-Oven-ChouxPastry rdf:type
  kr:HandlingInstruction, owl:NamedIndividual;
  kr:hasHandlingInstructionContent
    "(1) pre-heat at 170degC; (2) leave oven door ajar for 10 minutes;
    (3) close oven door; (4) continue for 45 minutes"^^xsd:string .

:HowTo-MainHeat-Oven-SpongeDough rdf:type
  kr:HandlingInstruction, owl:NamedIndividual;
  kr:hasHandlingInstructionContent

```

"(1) pre-heat at 170degC; (2) leave oven door ajar for 10 minutes;
 (3) close oven door; (4) continue for 45 minutes"^^xsd:string .

4.4.3.4 Process Operations and Process Operations Setup

One key feature of K-RAMP is the determination of reusable existing sequences of *ProcessOperations*, including their setups. For evaluation purposes, for the target bakery the existing sequences of *ProcessOperations* as part of *ProcessSegments* need to be prepared, including their individual *ProcessOperationsSetups* and associated *Equipment*, *HandlingInstructions* or *EquipmentRecipes* (Figure 4.14).

The example in Figure 4.14 shows a *ProcessSegment* “Baking-SheetInOven” which comprises a *ProcessOperation* “PreHeating” followed by a *ProcessOperation* “MainHeating”, “PostHeating” and “RetractFromBakingTray”. The *ProcessOperation* “MainHeating”, for instance, has two different *ProcessOperationSetups* (Num 1 and Num 2). Both *ProcessOperationSetups* are distinguished through the assigned *HandlingInstruction* – one for baking choux pastry and the other for baking sponge dough. However, both are using the same *EquipmentRecipe* for two different ovens.

Based on these sequences of *ProcessOperations* and their *ProcessOperationSetups*, an additional enhancement of the *ProcessSegmentSetups* is required. This is shown exemplarily in Figure 4.15. The appropriate *ProcessOperationSetups* are assigned for each *ProcessSegmentSetup*. For instance, the setup of *ProcessSegment* “Baking-SheetInOven” for satisfying the *ProcessCapability* “BakingChouxPastry-30x150x40” requires the *ProcessOperationSetup* with the trailing number 1 for the *ProcessOperation* “MainHeating”. But a different setup of the same *ProcessSegment* is used for satisfying the *ProcessCapability* “BakingCircularSpongeDough-240mm”. This setup requires the *ProcessOperationSetup* with the trailing number 2 for “MainHeating”. Considering Figure 4.14 ②, the *ProcessOperationSetup* with number 1 refers to the *HandlingInstruction* “HowTo-MainHeat-Oven-ChouxPastry”, while number 2 refers to the *HandlingInstruction* “HowTo-MainHeat-Oven-SpongeDough”.

For better understanding of the representation as ontology, the discussed exemplary process basic data are listed in Turtle syntax in the sequel.

```
:Baking-SheetInOven rdf:type kr:ProcessSegment, owl:NamedIndividual;
  kr:isAProductionTechnique kr-comm:BakingDough;
  kr:hasProcessSegmentSetup
    :Baking-SheetInOven4BakingChouxPastry-30x150x40-1,
    :Baking-SheetInOven4BakingCircularSpongeDough-240mm-1 .

:Baking-SheetInOven-PreHeating rdf:type
  kr:ProcessOperation, owl:NamedIndividual;
  kr:partOfProcessSegment :Baking-SheetInOven;
  kr:succeededBy :Baking-SheetInOven-MainHeating .

:Baking-SheetInOven-PreHeating-1 rdf:type
  kr:ProcessOperationSetup, owl:NamedIndividual;
  kr:onProcessOperation :Baking-SheetInOven-PreHeating;
  kr:onHandlingInstruction :HowTo-PreHeat-Oven-170degC;
  kr:onEquipment :Oven-1, :Oven-2;
  kr:onEquipmentRecipe :Oven-170degC-Convection .

:Baking-SheetInOven-MainHeating rdf:type
  kr:ProcessOperation, owl:NamedIndividual;
  kr:partOfProcessSegment :Baking-SheetInOven;
  kr:succeededBy :Baking-SheetInOven-PostHeating .
```



```

:Baking-SheetInOven-MainHeating-1 rdf:type
    kr:ProcessOperationSetup, owl:NamedIndividual;
kr:onProcessOperation :Baking-SheetInOven-MainHeating;
kr:onHandlingInstruction :HowTo-MainHeat-Oven-ChouxPastry;
kr:onEquipment :Oven-1, :Oven-2;
kr:onEquipmentRecipe :Oven-170degC-Convection .

:Baking-SheetInOven-MainHeating-2 rdf:type
    kr:ProcessOperationSetup, owl:NamedIndividual;
kr:onProcessOperation :Baking-SheetInOven-MainHeating;
kr:onHandlingInstruction :HowTo-MainHeat-Oven-SpongeDough;
kr:onEquipment :Oven-1, Oven-2;
kr:onEquipmentRecipe :Oven-170degC-Convection .

:Baking-SheetInOven-PostHeating rdf:type
    kr:ProcessOperation, owl:NamedIndividual;
kr:partOfProcessSegment :Baking-SheetInOven;
kr:succeededBy :Baking-SheetInOven-RetractFromBakingTray .

:Baking-SheetInOven-PostHeating-1 rdf:type
    kr:ProcessOperationSetup, owl:NamedIndividual;
kr:onProcessOperation :Baking-SheetInOven-PostHeating;
kr:onHandlingInstruction :HowTo-SwitchOff-Oven;
kr:onEquipment :Oven-1, :Oven-2 ;
kr:onEquipmentRecipe :Oven-PostHeat .

:Baking-SheetInOven-RetractFromBakingTray rdf:type
    kr:ProcessOperation, owl:NamedIndividual;
kr:partOfProcessSegment :Baking-SheetInOven .

:Baking-SheetInOven-RetractFromBakingTray-1 rdf:type
    kr:ProcessOperationSetup, owl:NamedIndividual;
kr:onProcessOperation :Baking-SheetInOven-RetractFromBakingTray;
kr:onHandlingInstruction :HowTo-RetractFromBakingTray-Oven;
kr:onEquipment :Oven-1, :Oven-2 .

:Baking-SheetInOven-RetractFromBakingTray-2 rdf:type
    kr:ProcessOperationSetup, owl:NamedIndividual;
kr:onProcessOperation :Baking-SheetInOven-RetractFromBakingTray;
kr:onHandlingInstruction :HowTo-RetractFromBakingTray-WithPieRing-Oven;
kr:onEquipment :Oven-1, :Oven-2 .

```

The underlying information models of this evaluation step, namely $DP_{3,3}$ and $DP_{3,6}$, comprise two trivial inverse relations only. Therefore, the TBOX ontology model $kr\text{-proc}$ primarily applies Semantic Web standard specifications. For this reason, there is also no further evaluation necessary with respect to the correctness of the behavior of PV_3 at all.

4.4.3.5 Intermediate Summary

The target bakery's ontology model $kr\text{-targ}$ also comprises all information items which are discussed previously in the context of $kr\text{-orig}$. However, as both ABOX ontology models are based on the same TBOX ontology models including the same reasoning rules and hybrid components the correctness of these aspects of $kr\text{-targ}$ can be assumed as proven because of the successful verification of $kr\text{-orig}$ in the previous chapters. Recalling the evaluation plan in Table 4.1, the complete test group 2 has been executed successfully.

F3 =WENN(UND(A3<>"";B3<>"";C3<>"";D3<>""); "ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:"&A3&"4"&B3&"-"&C3& " kr-targ:"&A3&"-"&D3&"-"&E3&"); ")						
	A	B	C	D	E	F
	ProcessSegment	Process Capability	num	Process Operation	Process Operation Setup Num	OWL
1	Baking-SheetInOven	BakingChouxPastry-30x150x40	1			
2	Baking-SheetInOven	BakingChouxPastry-30x150x40	1	PreHeating	1	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingChouxPastry-30x150x40-1 kr-targ:Baking-SheetInOven-PreHeating-1)
3	Baking-SheetInOven	BakingChouxPastry-30x150x40	1	MainHeating	1	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingChouxPastry-30x150x40-1 kr-targ:Baking-SheetInOven-MainHeating-1)
4	Baking-SheetInOven	BakingChouxPastry-30x150x40	1	PostHeating	1	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingChouxPastry-30x150x40-1 kr-targ:Baking-SheetInOven-PostHeating-1)
5	Baking-SheetInOven	BakingChouxPastry-30x150x40	1	RetractFrom BakingTray	1	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingChouxPastry-30x150x40-1 kr-targ:Baking-SheetInOven-RetractFromBakingTray-1)
6	Baking-SheetInOven	BakingCircularSponge Dough-240mm	1			
7	Baking-SheetInOven	BakingCircularSpongeDough-240mm	1	PreHeating	1	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingCircularSpongeDough-240mm-1 kr-targ:Baking-SheetInOven-PreHeating-1)
8	Baking-SheetInOven	BakingCircularSpongeDough-240mm	1	MainHeating	2	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingCircularSpongeDough-240mm-1 kr-targ:Baking-SheetInOven-MainHeating-2)
9	Baking-SheetInOven	BakingCircularSpongeDough-240mm	1	PostHeating	1	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingCircularSpongeDough-240mm-1 kr-targ:Baking-SheetInOven-PostHeating-1)
10	Baking-SheetInOven	BakingCircularSpongeDough-240mm	1	RetractFrom BakingTray	2	ObjectPropertyAssertion(kr:comprisesProcessOperationSetup kr-targ:Baking-SheetInOven4BakingCircularSpongeDough-240mm-1 kr-targ:Baking-SheetInOven-RetractFromBakingTray-2)
11						

Figure 4.15: Assignment of ProcessSegmentSetups and the appropriate ProcessOperationSetups, the generated statements in OWL functional syntax and the generation logic.

4.5 Evaluation of matchmaking and recommendations

4.5.1 Combination of the original bakery and the target bakery

The previous evaluation steps are only executed on the isolated ABOX ontology models of the original bakery or the target bakery. The correctness of essential building blocks for the matchmaking of two ontology models by the involved K-RAMP ontology models and the hybrid specification of K-RAMP processes can be demonstrated though. However, in the context of matchmaking of both ontology models *kr-orig* and *kr-targ* an evaluation is still required. The following subchapters address test group 3 and therefore the assertion of relations between *ProductCategorySets* followed by the assertion of individual *Recommendations*.

The starting point of the evaluation is an ontology model which comprises *kr-orig* and *kr-targ*. This starting point is achieved by creating an ontology model *kr-targ2* which imports *kr-orig* and *kr-targ* directly as well as *kr-comm* and *kr-all* indirectly. The reasoning results of the combined ontology *kr-targ2* are then verified within Protégé. Like a zipper, in the course of evaluation, the entanglement of both ontology models is thus verified.

As already highlighted in Chapter 4.4.1, some ABOX ontology models are considered as common across both bakeries. For this purpose, PV_{1,1} (*Engineering Units modeling and exchange*), PV_{1,2} (*Characteristics modeling and exchange*) and PV_{1,4} (*Production Techniques modeling and exchange*) are excluded from the verification steps of this chapter.

4.5.2 Relations between Product Category Sets

This subchapter verifies the correctness of the procedure `assertProductCategorySet`, to which is also referred as (3.5) in the sequel. The procedure is embedded in a nested loop which iterates through all *ProductCategories* and all *ProcessSegmentSetups* of a Semantic Web database. *ProductCategories* and *ProcessSegmentSetups* are considered as *ProductCategorySets* in the sequel of this chapter. In the following sections, one representative out of “all *ProductCategories*” is considered as the set $PT^i = \{f_i : i \leq |PT^i|\}$ (the introduced *ProductCategorySet*), and one representative of “all *ProcessSegmentSetups*” is considered as the set $PT^x = \{g_i : i \leq |PT^x|\}$ (the existing *ProductCategorySet*). Introducing indices for the sets’ elements enable the positioning of each element within the set.

Each pair (PT^i, PT^x) is passed to `assertProductCategorySet`. The possible relations with respect to object properties are *enclosesProductCategorySet*, *overlapsProductCategorySet* and *partiallyProductCategorySet* in accordance to the specification of DP_{4,1}. For the purpose of the verification of correctness, assumptions for such pairs of *ProductCategorySets* are made accordingly. The procedure expects *ProductCategorySets* as input arguments. The comparison of an individual of *ProductCategorySet* with itself is not part of the verification, as individuals of *ProductCategory* and individuals of *ProcessSegmentSet* are considered to be disjoint.

The assumption therefore always starts from two disjoint pair items. Moreover, the logic of this procedure is built on asserted relations with respect to the object properties *encloses*, *overlaps* and *partially* between *SpecificationSet* which are members of the passed *ProductCategorySets*. These relations are also not symmetric and are determined from the perspective of the members of PT^i .

	encloses = E	overlaps = O	partially overlaps = P
C-1	$\forall f_i \in PT^i \exists g_i \in PT^x f_i E g_i$	ε	ε
C-2	ε	$\forall f_i \in PT^i \exists g_i \in PT^x f_i O g_i$	
C-3	ε	ε	$\forall f_i \in PT^i \exists g_i \in PT^x f_i P g_i$
C-4	$\forall f_i \in PT^i \exists g_i \in PT^x (f_i E g_i \wedge \neg f_i O g_i) \vee (\neg f_i E g_i \wedge f_i O g_i)$		ε
C-5	ε	$\forall f_i \in PT^i \exists g_i \in PT^x (f_i O g_i \wedge \neg f_i P g_i) \vee (\neg f_i O g_i \wedge f_i P g_i)$	
C-6	$\forall f_i \in PT^i \exists g_i \in PT^x (f_i E g_i \wedge \neg f_i P g_i) \vee (\neg f_i E g_i \wedge f_i P g_i) \dots$ no overlaps		
C-7	$\exists f_i \in PT^i \exists g_i \in PT^x (f_i E g_i \wedge \neg f_i O g_i \wedge \neg f_i P g_i) \vee (\neg f_i E g_i \wedge f_i O g_i \wedge \neg f_i P g_i) \vee (\neg f_i E g_i \wedge \neg f_i O g_i \wedge f_i P g_i)$		
C-8	ε	ε	ε

Table 4.11: Possible cases for the verification of `assertProductCategorySet`.

The cases as listed in Table 4.11 are considered as complete set of possibilities of the verification. Each expression applies the same index i for elements of PT^i and PT^x indicating that only the elements along the main diagonal of the comparison matrix are determined. All other relations between elements f_i and g_j with $i \neq j$ are considered to be weaker than the respective

relation along the main diagonal. For each case with respect to the code snippet (3.5), it has to be shown how the comparison matrix is structured by (3.5)②, and it has to be verified whether the decisions at (3.5)③ are performed correctly.

Case C-1 assumes that every element of PT^i encloses an element of PT^x . There are no overlaps or partially overlaps accordingly. As a consequence, the nested loop of (3.5)② populates the main diagonal with 3 only, and the count of items with value 3 on the main diagonal is equal to the total number of elements of the main diagonal according to (3.5)③. Correctly, this case leads to an assertion of *enclosesProductCategorySet*.

Case C-2 assumes that every element of PT^i overlaps an element of PT^x . There are no enclosing or partially overlapping relations. The main diagonal of the comparison matrix is populated with values 2 only ($count(diag, 2) = size(diag)$). Therefore, $count(diag, 3)$ in the code snippet of (3.5) results to 0 and thus $count(diag, 3) \neq size(diag)$. However, $count(diag, 2) + count(diag, 3) = count(diag, 2) + 0 = count(diag, 2) = size(diag)$. Therefore, the relation *overlapsProductCategorySet* is assumed correctly.

Case C-3 assumes that every element of PT^i partially overlaps an element of PT^x thus populating the whole main diagonal with values 1. Therefore, $count(diag, 3)$ and $count(diag, 2)$ result to 0, which results to $count(diag, 3) \neq size(diag)$ and further $count(diag, 3) + count(diag, 2) \neq size(diag)$. However, as the complete main diagonal is populated with values 1 $count(diag, 1) = size(diag)$ and as a consequence $count(diag, 0) = 0$ and therefore $count(diag, 0) < size(diag)$. This leads to the correct assertion of *partiallyProductCategorySet*.

Case C-4 assumes that every element of PT^i either enclose elements of PT^x or overlap such elements. The nested loop (3.5)② thus populates the main diagonal of the comparison matrix exclusively with values 3 or 2. The cases C-1 and C-2 are intentionally excluded from C-4. Therefore, $count(diag, 3) < size(diag)$ does not satisfy the first condition but $count(diag, 3) + count(diag, 2) = size(diag)$. This leads to the assertion of *overlapsProductCategorySet*.

Case C-5 assumes that every element of PT^i either overlaps elements of PT^x or partially overlaps such elements. The nested loop (3.5)② thus populates the main diagonal of the comparison matrix exclusively with values 2 or 1. The cases C-2 and C-3 are intentionally excluded from C-5. Therefore, $count(diag, 3) = 0$ does not satisfy the first condition and $count(diag, 3) + count(diag, 2) = 0 + count(diag, 2) < size(diag)$ does not satisfy the second condition either. However, as the complete main diagonal is exclusively populated with values 2 or 1 $count(diag, 2) + count(diag, 1) = size(diag)$ and as a consequence $count(diag, 0) = 0$ and therefore $count(diag, 0) < size(diag)$. This leads to the correct assertion of *partiallyProductCategorySet*.

Case C-6 assumes that every element of PT^i either encloses elements of PT^x or partially overlaps such elements. The nested loop (3.5)② thus populates the main diagonal of the comparison matrix exclusively with values 3 or 1. The cases C-1 and C-3 are intentionally excluded from C-6. Therefore, $count(diag, 3) < size(diag)$ does not satisfy the first condition and $count(diag, 3) + count(diag, 2) = count(diag, 3) + 0 < size(diag)$ does not satisfy the second condition either. However, as the complete main diagonal is exclusively populated with values 3 or 1 $count(diag, 3) + count(diag, 1) = size(diag)$ and as a consequence $count(diag, 0) = 0$ and therefore $count(diag, 0) < size(diag)$. This leads to the correct assertion of *partiallyProductCategorySet*.

Case C-7 assumes that there is at least one element of PT^i which either encloses an element of PT^x , overlaps or partially overlaps such an element. The nested loop (3.5)② thus populates the main diagonal of the comparison matrix with at least one value of the range 3, 2 or 1. The cases

C-1 to C-6 are intentionally excluded from C-7. Therefore, $count(diag, 3) < size(diag)$ does not satisfy the first condition and $count(diag, 3) + count(diag, 2) < size(diag)$ does not satisfy the second condition either. However, the complete main diagonal is at least populated with one value not equal to 0 and as a consequence $count(diag, 0) < size(diag)$ is true. This leads to the correct assertion of *partiallyProductCategorySet*.

Case C-8 assumes that there is no element of PT_i which is related to an element of PT_x. The nested loop (3.5)Ⓜ thus populates the main diagonal of the comparison matrix exclusively with values 0. Therefore, $count(diag, 3) = 0 < size(diag)$ does not satisfy the first condition and $count(diag, 3) + count(diag, 2) = 0 < size(diag)$ does not satisfy the second condition either. Moreover, $count(diag, 0) = size(diag)$ thus violates the third condition. Correctly, this leads to no assertion.

From a theoretical stand point performing correctly though, the behavior of the procedure `assertProductCategorySet` is also demonstrated by applying it on some examples. The following snippets from `kr-targ2` represent exemplary individuals of *RelationOfSpecificationSets*. The prefixes `kr-orig-` and `kr-targ-` in the URIs of these individuals are applied in order to distinct the URI-portions of the two involved *SpecificationSets*.

```
:kr-orig-SmoothLiquidApricotJam-kr-targ-SmoothLiquidApricotJam rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
    kr:ratioEncloses 100.0;
    kr:ratioCommonCharacteristics 100.0;
    kr:commonCharacteristic kr-comm:RatioFruite, kr-comm:RatioSweetener;
    kr:toSpecificationSet kr-targ:SmoothLiquidApricotJam;
    kr:fromSpecificationSet kr-orig:SmoothLiquidApricotJam .

:kr-orig-CircularSliceOfVienneseDough-kr-targ-CircularMeranerDoughSlice rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
    kr:ratioEncloses 0.0;
    kr:ratioCommonCharacteristics 100.0;
    kr:commonCharacteristic kr-comm:Dough, kr-comm:ColorOfDough;
    kr:toSpecificationSet kr-targ:CircularMeranerDoughSlice;
    kr:fromSpecificationSet kr-orig:CircularSliceOfVienneseDough .

:kr-orig-StiffEggWhite-kr-targ-BeatenSweetEggWhite rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
    kr:ratioEncloses 100.0;
    kr:ratioCommonCharacteristics 100.0;
    kr:commonCharacteristic kr-comm:RatioAir, kr-comm:RatioEggWhite,
        kr-comm:RatioSugar;
    kr:toSpecificationSet kr-targ:BeatenSweetEggWhite;
    kr:fromSpecificationSet kr-orig:StiffEggWhite .

:kr-orig-YolkChocolateButterCreme-kr-targ-YolkButterCreme rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
    kr:ratioEncloses 20.0;
    kr:ratioCommonCharacteristics 80.0;
    kr:commonCharacteristic kr-comm:RatioFat, kr-comm:RatioWater,
        kr-comm:RatioAir, kr-comm:RatioYolk;
    kr:toSpecificationSet kr-targ:YolkChocolateButterCreme;
    kr:fromSpecificationSet kr-orig:YolkButterCreme .

:kr-orig-WheatBakingFlour-kr-targ-FlourBakingPowderMixture rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
    kr:ratioEncloses 100.0;
    kr:ratioCommonCharacteristics 100.0;
    kr:commonCharacteristic kr-comm:RatioMineralNutrients,
```

```

kr-comm:RatioProtein, kr-comm:RatioWetGluten,
kr-comm:ZelenyIndex, kr-comm:RatioGrain;
kr:toSpecificationSet kr-targ:FlourBakingPowderMixture;
kr:fromSpecificationSet kr-orig:WheatBakingFlour .

```

As already discussed and verified in the context of Chapter 4.4.2.2, these assertions result in further reasoning of respective relations with respect to *enclosesSpecificationSet*, *overlapsSpecificationSet* or *partiallySpecificationSet*. The respective results due to the previous snippet is shown in the below.

```

kr-orig:SmoothLiquidApricotJam kr:enclosesSpecificationSet
kr-targ:SmoothLiquidApricotJam .
kr-orig:CircularSliceOfVienneseDough kr:overlapsSpecificationSet
kr-targ:CircularMeranerDoughSlice .
kr-orig:StiffEggWhite kr:enclosesSpecificationSet
kr-targ:BeatenSweetEggWhite .
kr-orig:YolkChocolateButterCreme kr:partiallySpecificationSet
kr-targ:YolkButterCreme .
kr-orig:WheatBakingFlour kr:enclosesSpecificationSet
kr-targ:FlourBakingPowderMixture .

```

DP_{4,1} specifies the information model which is implemented by the TBOX model *kr-rec*. Due to this model, individuals of *ProductCategories* and of *ProcessSegmentSetups* implicitly become individuals of *ProductCategorySets* as well. A few examples are listed in the sequel.

```

kr-orig:LiquidVienneseDough rdf:type
kr:ProductCategory, kr:ProductCategorySet, owl:NamedIndividual;
kr:requires kr-orig:FoldingVienneseDough-820g;
kr:decomposedTo kr-comm:WheatBakingFlour,
kr-orig:StiffEggWhite,
kr-orig:YolkChocolateButterCreme;
kr:hasProductCategory kr-comm:WheatBakingFlour,
kr-orig:StiffEggWhite,
kr-orig:YolkChocolateButterCreme .

kr-targ:Folding-DoughScraper4FoldingSpongeDough-840g-1 rdf:type
kr:ProcessSegmentSetup, kr:ProductCategorySet, owl:NamedIndividual;
kr:uses kr-targ:FlourBakingPowderMixture,
kr-targ:BeatenSweetEggWhite,
kr-targ:YolkButterCreme;
kr:hasProductCategory kr-targ:FlourBakingPowderMixture,
kr-targ:BeatenSweetEggWhite,
kr-targ:YolkButterCreme .

kr-orig:CircularSliceVienneseDoughCoatedWithApricotJam rdf:type
kr:ProductCategory, kr:ProductCategorySet, owl:NamedIndividual;
kr:requires
kr-orig:CoatingCircularSpongeDoughCover-240mm-ApricotJam;
kr:decomposedTo kr-orig:CircularSliceOfVienneseDough,
kr-orig:SmoothLiquidApricotJam;
kr:hasProductCategory kr-orig:CircularSliceOfVienneseDough,
kr-orig:SmoothLiquidApricotJam .

kr-targ:Coating-Spatula4CoatingCircularSpongeDoughCover-240mm-ApricotJam-1
rdf:type
kr:ProcessSegmentSetup, kr:ProductCategorySet, owl:NamedIndividual;
kr:uses kr-targ:CircularMeranerDoughSlice,
kr-targ:SmoothLiquidApricotJam;
kr:hasProductCategory kr-targ:CircularMeranerDoughSlice,
kr-targ:SmoothLiquidApricotJam .

```

	kr-targ: FlourBakingPowder Mixture	kr-targ: BeatenSweetEgg White	kr-targ: YolkButterCreme
kr-comm: WheatBakingFlour	3	0	0
kr-orig: StiffEggWhite	0	3	1
kr-orig: YolkChocolate ButterCreme	0	1	1

Table 4.12: Comparison matrix of kr-orig:LiquidVienneseDough vs. kr-targ:Folding-DoughScraper4FoldingSpongeDough-840g-1.

The comparison matrix of Table 4.12 results from the question, how *ProductCategories* which compose kr-orig:LiquidVienneseDough are related to the *ProductCategories* which are used by kr-targ:Folding-DoughScraper4FoldingSpongeDough-840g-1. According to the previously asserted relations between *SpecificationSets*, kr-comm:WheatBakingFlour encloses kr-targ:FlourBakingPowderMixture for instance. Therefore, the respective item of the comparison matrix is set to 3. The same considerations are also made for the remaining items of the main diagonal. The remaining items of the matrix are set to 0 or 1 in accordance to determined relations. Recalling the behavior of assertProductCategorySet, this comparison matrix results in assertion

```
kr-orig:LiquidVienneseDough kr:partiallyProductCategorySet
kr-targ:Folding-DoughScraper4FoldingSpongeDough-840g-1 .
```

	Kr-targ: SmoothLiquidApricotJam	Kr-targ: CircularMeranerDoughSlice
Kr-orig: SmoothLiquidApricotJam	3	0
Kr-orig: CircularSliceOfViennese Dough	0	2

Table 4.13: Comparison matrix of kr-orig:CircularSliceVienneseDoughCoatedWithApricotJam vs. kr-targ:Coating-Spatula4CoatingCircularSpongeDoughCover-240mm-ApricotJam-1.

Another example is shown by the comparison matrix of Table 4.13 namely the comparison between kr-orig:CircularSliceVienneseDoughCoatedWithApricotJam and kr-targ:Coating-Spatula4CoatingCircularSpongeDoughCover-240mm-ApricotJam-1. This comparison matrix leads to an assertion as follows.

```
kr-orig:CircularSliceVienneseDoughCoatedWithApricotJam
kr:enclosesProductCategorySet
kr-targ:Coating-Spatula4CoatingCircularSpongeDoughCover-240mm-ApricotJam-1 .
```


The subtest group 3.i is successfully completed due to the previous verification and the demonstrated examples based on the cake-baking case study.

4.5.3 Assertion of Recommendations

The previous evaluation steps are preparing the foundation of the final subtest group 3.ii. This subtest group verifies the behavior of the procedure `assertRecommendation` as part of the hybrid specification of $PV_{4.1}$. The needed conditions for appropriate assertion of *Recommendations* are already discussed in conjunction with Table 3.17 as well as in the context of the code snippet (3.6). Therefore, this evaluation step is started with an exemplary scenario. The considered pair of individuals of this test scenario is `kr-orig:LiquidVienneseDough` and `kr-targ:FoldingSpongeDough-840g`. What has to be considered in order to utilize the existing *ProcessCapability* for the production of *Products* of the introduced *ProductCategory*?

According to the code snippet (3.6)①, existing *ProductCategories* which require the considered *ProcessCapability* are determined. In the same way, all the newly introduced *ProcessCapabilities* of the considered *ProductCategory* are being investigated. With respect to the considered *ProcessCapability*, also the set of enabling *ProcessSegmentSetups* is gathered.

Recalling the exemplary pair, this step of the procedure leads to the existing *ProductCategory* `kr-comm:LiquidSpongeDough` which requires `kr-targ:FoldingSpongeDough-840g`, the new introduced *ProcessCapability* `kr-orig:FoldingVienneseDough-820g` which is required by the new introduced `kr-orig:LiquidVienneseDough` and the existing *ProcessSegmentSetup* which enables the *ProcessCapability* `kr-targ:FoldingDoughScraper4FoldingSpongeDough-840g-1`.

As there is one individual for each set in this particular example, the nested loop of the procedure is executed only once. The individual `kr-orig:LiquidVienneseDough` (the variable `pti`) is linked to `kr-targ:FoldingDoughScraper4FoldingSpongeDough-840g-1` (the variable `ps`) via `kr:partiallyProductCategorySet` (see discussion in conjunction with Table 4.12). Therefore, one of the matchmaking scenarios MS-06, MS-07, MS-10, MS-11, MS-16 or MS-17 of Table 3.17 applies.

The following Turtle snippet indicates that the new introduced *ProcessCapability* (variable `pci`) overlaps with the existing *ProcessCapability* (variable `pcx`), thus reducing the possible set of matchmaking scenarios to MS-10 and MS-11.

```
:kr-orig-FoldingVienneseDough-820g-kr-targ-FoldingSpongeDough-840g rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
    kr:ratioEncloses 0.0;
    kr:ratioCommonCharacteristics 100.0;
    kr:commonCharacteristic kr-comm:Weight;
    kr:toSpecificationSet kr-targ:FoldingSpongeDough-840g;
    kr:fromSpecificationSet kr-orig:FoldingVienneseDough-820g .

kr-orig:FoldingVienneseDough-820g kr:overlapsSpecificationSet
    kr-targ:FoldingSpongeDough-840g .
```

In accordance with Table 3.17, it has to be determined whether the introduced *ProductCategory* `kr-orig:LiquidVienneseDough` (variable `pti`) either overlaps or partially overlaps with the existing *ProductCategory* `kr-comm:LiquidSpongeDough` (variable `ptx`). This decision matches also with the remaining partial condition of the code snippet

(3.6)⑤. The given relation between both *ProductCategories* from the perspective of *pti* is listed below.

```

:kr-orig-LiquidVienneseDough-kr-targ-LiquidSpongeDough rdf:type
    kr:RelationOfSpecificationSets, owl:NamedIndividual;
kr:ratioEncloses 0.0;
kr:ratioCommonCharacteristics 88.0;
kr:commonCharacteristic kr-comm:RatioAir, kr-comm:RatioFat,
    kr-comm:RatioYolk, kr-comm:RatioEggWhite,
    kr-comm:RatioFlour, kr-comm:RatioWater,
    kr-comm:Viscosity;
kr:toSpecificationSet kr-targ:LiquidSpongeDough;
kr:fromSpecificationSet kr-orig:LiquidVienneseDough .

kr-orig:LiquidVienneseDough kr:overlapsSpecificationSet
    kr-targ:LiquidSpongeDough .

```

Therefore, matchmaking scenario MS-11 of Table 3.17 is in place and according to the procedure `assertRecommendation` the code snippet at (3.6)⑤ is utilized for respective assertions. These assertions result in the following OWL-statements in Turtle.

```

kr-targ2:Rec-1 rdf:type kr:Recommendation, owl:NamedIndividual;
kr:forProductCategory kr-orig:LiquidVienneseDough;
kr:substitutes kr-targ:FoldingVienneseDough-820g;
kr:withProcessSegmentSetup
    kr-targ:Folding-DoughScrapper4FoldingSpongeDough-840g-1;
kr:useProductCategorySet kr-orig:LiquidVienneseDough;
kr:adjustParametersOf
    kr-targ:Folding-DoughScrapper4FoldingSpongeDough-840g-1;
kr:penalty 3 .

```

As a sentence, `kr-targ2:Rec-1` can be expressed as: With low (3) penalty *LiquidVienneseDough* can be produced by substituting *FoldingVienneseDough-820g* with *FoldingSpongeDough-840g* on *Folding-DoughScrapper* (comment: *ProcessCapability* and *ProcessSegment* gathered through *withProcessSegmentSetup*). For this purpose *YolkChocolateButterCreme*, *StiffEggWhite* and *WheatBakingFlour* (comment: used *ProductCategories* gathered through *useProductCategorySet o hasProductCategory*) must be used and the setup parameters and handling instructions of *Folding-DoughScrapper4FoldingSpongeDough-840g-1* may be adjusted.

By combining each introduced *ProductCategory* with each existing *ProcessCapability*, such *Recommendations* are asserted for every case which satisfies one of the conditions as implemented by `assertRecommendation`. The validity of all other conditions of (3.6) can be verified easily in conjunction with Table 3.17 and the specifications of $DP_{4,1}$. The gathering of information as well as the particular assertions are implemented straight forward and do not need further verification. Test group 3 and thus the whole evaluation plan is described accordingly and provides a seamless sequence of implications and assertions which finally leads to the intended recommendations for product ramp-up teams.

4.6 Summary

By performing the cake-baking case study, it can be demonstrated that recommendations at a target production system can be derived from the matchmaking of product-related knowledge and device-independent process-related knowledge about the Viennese chocolate cake with the

comprehensive knowledge of a target production system. For this purpose several ABOX ontology models have been designed which describe the information base of the original bakery and the target bakery as well as a commonly shared information base about *Characteristics*, *Product Categories* or *Production Techniques*.

The evaluation plan is split into major test groups which are dedicated to particular test cases. The most test cases are also applicable within the isolated scope of each information base. Therefore, the modeling of *Production Techniques*, *Engineering Units*, *Characteristics*, *Products*, *Product Categories* and *Specification Sets* in general as well as reasoning on these portions of the information model are verified within the scope of the common information base (kr-comm ontology model). The results of this evaluation step are also applicable for any other information base within the evaluation environment.

The second major test group is performed on the information base of the original bakery. It comprises the modeling of *Process Capabilities* and *Process Segments* as well as reasoning of similarities between *Process Capabilities* and the categorization of *Process Segments*. There is again no specific need to use the information base of the original bakery for this purpose because also these evaluation results are applicable for all other information bases. The same test group applies also the information base of the target bakery in order to evaluate the modeling and reasoning on device-specific aspects, like *Process Segment Setup*, *Product Category Usage*, *Equipment*, *Equipment Recipe*, *Handling Instruction*, *Process Operation* or *Process Operation Setup*. This portion of the information model is usually also available at the original production system. Therefore, all evaluation results are applicable for every other information base as well.

The third major test group envelops the final parts of the matchmaking process. The reasoning of relations between *Product Category Sets* is verified in the scope of this test group as well as the assertion of *Recommendations*.

Throughout the evaluation, best practices are demonstrated with respect to the preparation of product basic data and process basic data for upload to the knowledge store. Applying the case study on the K-RAMP ontology models and the K-RAMP process proves the achievement of (Objective 2) and its subordinated objectives thus answering (Question 4) and consequently (Question 2). The ontology model and its hybrid components are not only described but also their appropriate behavior is verified within the scope of a representative case study.

5 Reflection

5.1 Overview

In this chapter, the results of the evaluation of K-RAMP are reflected to the objectives and the research question. The conclusions of the evaluation shall be discussed therefore in Chapter 5.2. The fact that K-RAMP is evaluated on a hypothetic transfer of a cake-baking recipe from one bakery to another requires a discussion about the applicability of the approach for production systems in general. This is discussed in Chapter 5.3. K-RAMP requires comprehensive and careful management of product basic data and process basic data. However, this is not an obvious assumption which is applicable to the majority of current production systems. Premises for the implementation of K-RAMP are therefore also discussed in Chapter 5.3.

5.2 Evaluation versus objectives

Already the technical discussion of Chapter 3.2.6 is highlighting that features of the Semantic Web can be utilized in order to implement an inter-disciplinary knowledge base for recommendation of knowledge reuse or adjustment of reusable information in the context of a product ramp-up. It can be also shown, that nowadays the most common specifications of the Semantic Web – namely XML, XSD, RDF, RDFS, OWL2, SWRL and SPARQL – are covering a significant part of the required basic functionality for this purpose. Referring to the summary in Table 3.25, it can be thus confirmed that a major part of K-RAMP can be implemented by a Semantic Web database which is implemented in accordance with these specifications. However, Table 3.25 also implies that there are logical expressions required which are not covered by the Semantic Web. These additional functions require the implementation of complementary hybrid components. Therefore, (Question 3) cannot be answered. However, (Question 4) can be answered due to the design of appropriate ontology models and complementary hybrid components of K-RAMP which are verified successfully by using an exemplary case study. Consequently, (Question 2) is answered with the limitations of (Question 4).

Due to the verification of (Question 1) in Chapter 3.2.6 and the implemented and evaluated case study of K-RAMP, the implication is valid that a respective multi-disciplinary knowledge base can be designed and implemented by a hybrid architecture which maximizes the involvement of the Semantic Web.

5.3 Applicability of K-RAMP in real production

5.3.1 Generalization of the evaluation of K-RAMP and limitations

Is the applied case study of cake-baking representative for real production environments? The following sections address this topic and provide a respective answer. First, there is an attempt to demonstrate the ability to generalize the cake-baking case study to other applications in real production. Secondly, premises are discussed which have to be satisfied for a successful application of K-RAMP in the respective real-world production system.

For the purpose of generalization of the cake-baking case study, representative real-world applications are selected and compared with it. The cake-baking case study was taken as commonly understood example and to avoid overflowing introductions of particular manufacturing industries. Actually, the cake-baking case study itself is also a subset of a real-world application, which is the food and beverage industry. Therefore, while working on examples of this case study, it was determined that experiences from the other manufacturing industries perfectly match with the case study.

While semiconductor manufacturing is representative for production techniques like separating, coating and altering of material, cake-baking does also invoke additional production techniques like master forming, forming and merging. Assembly processes (electronics assembly, mechanical assembly of components, machines or vehicles) represent another broad portfolio of industrial manufacturing. All mentioned real-world applications and the exemplary cake-baking case study are classified as discrete manufacturing, respectively the production of distinct items. However, there is also the class of continuous production processes, which produces undifferentiated products. Representatives of this class are parts of food and beverage or the chemical industry (e.g., production of beverages before bottling into distinct items, production of gases, granular or powder materials). The usability of K-RAMP for continuous production processes shall be subject to future research.

For every branch of the manufacturing industry – independently whether discrete manufacturing or process manufacturing – the following organizational elements have to be considered as part of the product definition and the specification of the respective production process.

- The specification of the products' functions and their quality metrics.
- The categorization of products as product categories for easier generalization and in order to achieve a product portfolio which is scalable easier.
- The hierarchical composition structure of products or product categories not only in the sense of assembly structure but also in the sense of changing material conditions from one process segment to the next.
- The specification of sequences of appropriate process operations and their organization as process segments.
- The classification of process segments through production techniques. In case of process operations within one process segment which follow different production techniques the most significant one shall be chosen for categorization of the whole process segment.
- The specification of handling instructions, in order to create, treat or gauge each individual subproduct.

- The specification of equipment recipes (physical or logical equipment setup), in order to create, treat or gauge each individual subproduct.

Axiomatic design, which is also used as design approach of K-RAMP, highlights these planning elements for the design of functional requirements, design characteristics and the underlying production process. The common applicability of axiomatic design is highlighted in [18] for a variety of designed products and production systems. The design elements of axiomatic design can be mapped to aforementioned organizational elements as shown in Table 5.1. The general applicability of the K-RAMP information model is therefore supported due to the exact mapping of the specification-related aspects of the K-RAMP information model to the structure of axiomatic design.

Planning element	Axiomatic design element
Product functions	Functional requirements
Specifications of products and product categories	Specifications of functional requirements
Product categories	Design parameters
Specifications of products and product categories	Specifications of design parameters
Hierarchical decomposition structure of products and product categories	Decomposition of design parameters
Process capabilities	Process variables
Specifications of process capabilities	Specifications of process variables

Table 5.1: Mapping of organizational elements and axiomatic design elements.

The previously listed planning elements are supported by the K-RAMP information model in order to achieve the intended target. The general validity of the need of these planning elements is also mentioned as part of value engineering (e.g., [56, pp. 14-23]) where the development of the values for products is aligned closely with appropriate planning of the production process. Vajna and Burchardt are summarizing available models and approaches for integrated product development with similar results [57]. And the ISO 9001 standard requires an integrated product planning, product realization and respective metrology as well [58, pp. 7-14].

It is also notable that the PABADiS consortium [59, pp. 77-85] derived almost the same data structures for the specification of capabilities and products as it is also specified in K-RAMP. As PABADiS's data structure is already accepted since more than one decade, it can be concluded that the ontology models of `kr-unit`, `kr-char`, `kr-spec`, `kr-prod` and `kr-prod-cat` are also sufficiently complete for the same field of application.

Kluge [60, pp. 77-103] describes a generic description method which is based on a capability model in order to plan the setup of modular assembly systems. He also applies the separation between the capabilities of resources and the decomposition structure of products. Both aspects are specified through features which can be mapped to the *Specifications* in the K-RAMP information model. Kluge focuses on a generic level of comparison between features. In the next sections it shall be demonstrated that the K-RAMP information model and reasoning applies on such an assembly scenario also by the use of more detailed specifications.

For this purpose, Figure 5.1 shows two exemplary product categories "A" and "B" which shall be mounted by a modular assembly system as describe by Kluge. Product categories are used instead of products because there may be additionally scalable features of the products which allow their grouping in such categories. The product category "standardized M4 head tapping

screw with inside hexagon” shall be applied for mounting the product categories “A” respectively “B”. Furthermore, “B” shall be considered as the new product category while “A” is some forerunner product category.

Based on Figure 5.1, it is demonstrated how the K-RAMP information model is also applicable for assembly problems. This is not surprising as the K-RAMP information model is derived from common methods for the specification of products. An important management aspect to be highlighted is the identification of features across both product categories. The positions of the screws represent characteristics which are shared between the specifications of both product categories. The new product category “B” uses the same identifiers for positions (“S-1-X”, “S-1-Y”, “S-2-X”, “S-2-Y”, “Torque”) where this is semantically useful. Consequently, these specifications need to be considered already during the design of the new product category. As a suggestion, the central management of specifications should be therefore also involved in the design process of the product categories.

The reasoning process of K-RAMP determines partial overlapping between “A” and “B” from the perspective of the new product category “B” because “S-1-X”, “S-1-Y”, “S-2-X”, “S-2-Y” and also supposed “Torque” are enclosed but there is no “S-3-X” or “S-3-Y”.

With respect to process capabilities (Figure 5.2), in the discussed example, the process segments “Drilling tapped screw thread”, “Drilling screw thread” and “Screwing hexagon head tapping screw” are already available and are used for drilling holes for “S-1” and “S-2” in the angle iron (product category “A-1”), the panel (product category “A-2”) respectively for mounting the angle on the panel (product category “A”). Process capabilities are understood as capabilities of the process segment and not of the individual resource. Even if there is only one resource available, process operations and process segments shall be introduced which finally enable dedicated process capabilities.

The process capabilities for drilling (“PC-A1” to “PC-B2”) are specified by the identification of the CNC-program (“Program”) to be used (equipment recipe applied as specification target) as well as by the ranges of the positions where both screw threads can be placed (“A-S-1-X” to “A-S-2-Y” and “P-S-1-X” to “P-S-2-Y” as specification ranges) due to the logic of the program.

It has to be balanced how much information of an applied handling instruction of equipment recipe is made visible as part of the specification structure of a process capability. In the particular case it is required that existing process capabilities are detected during matchmaking through the identification of the program and the position of the drilled holes. In a very efficient implementation, the equipment recipe is parameterized by the specification structure and the MES is able to combine the equipment recipe automatically with the specification of the respective product. In a less efficient implementation, it must be ensured that the specification structure matches with the logic of the equipment recipe.

The incoming and outgoing product categories “Angle”, “Panel”, “Drilled Angle”, “Drilled Panel” and “Assembled Part” are superior product categories of “A-1”, “B-1”, “A-2”, “B-2”, “A” and “B”. The reasoning process of K-RAMP determines the existing process capabilities for product category “A-1” and “A-2” as partially overlapping with the requested process capabilities of “B-1” respectively “B-2”. Although, “Drilling tapped screw thread” and “Drilling screw thread” are both drilling threads the reasoning process will determine “PC-A1” as the better fit for “PC-B1” than “PC-A2” because there is a higher ration of common characteristics. Also the used and the required set of product categories is at least partially overlapping because “A-1” and “B-1” are both specializations of the product category “Angle”, and “A-2” and “B-2” are specializations of the product category “Panel” accordingly. With respect to the matchmaking between “PC-A” and

“PC-B”, the used product categories are partially overlapping because currently two screws are used while three screws are required for the assembly of “B”.

The resulting recommendation follows the matchmaking scenario MS-17. From the perspective of the product category “B”, there is a partial overlapping with the product category “A”. Moreover, the required process capability (e.g., “PC-B”) is partially overlapping with the existing process capability “PC-A”, and on the level of product category sets there is also a partial overlapping between the decomposition structure of “A” and the used product categories of “PC-A” (one screw less is used). The recommendation is reasonable, because an additional hole requires some modification of the process segment setup and also of the process operations where the quality of the holes is measured. Also the fact has to be included that additionally gauged data need to be collected. The impact on the existing process is rather deep.

In the opposite direction, assuming that “A” is the new product category, the recommended matchmaking scenario is MS-05 because “A” is enclosing “B”, and the decomposition is partially overlapping. The process capability “PC-A” encloses the process capability “PC-B”. Also in this case, the lower penalty sounds reasonable because it is significant less effort to remove information items from an existing setup (drilling the third hole, screwing the third screw) then to add new information.

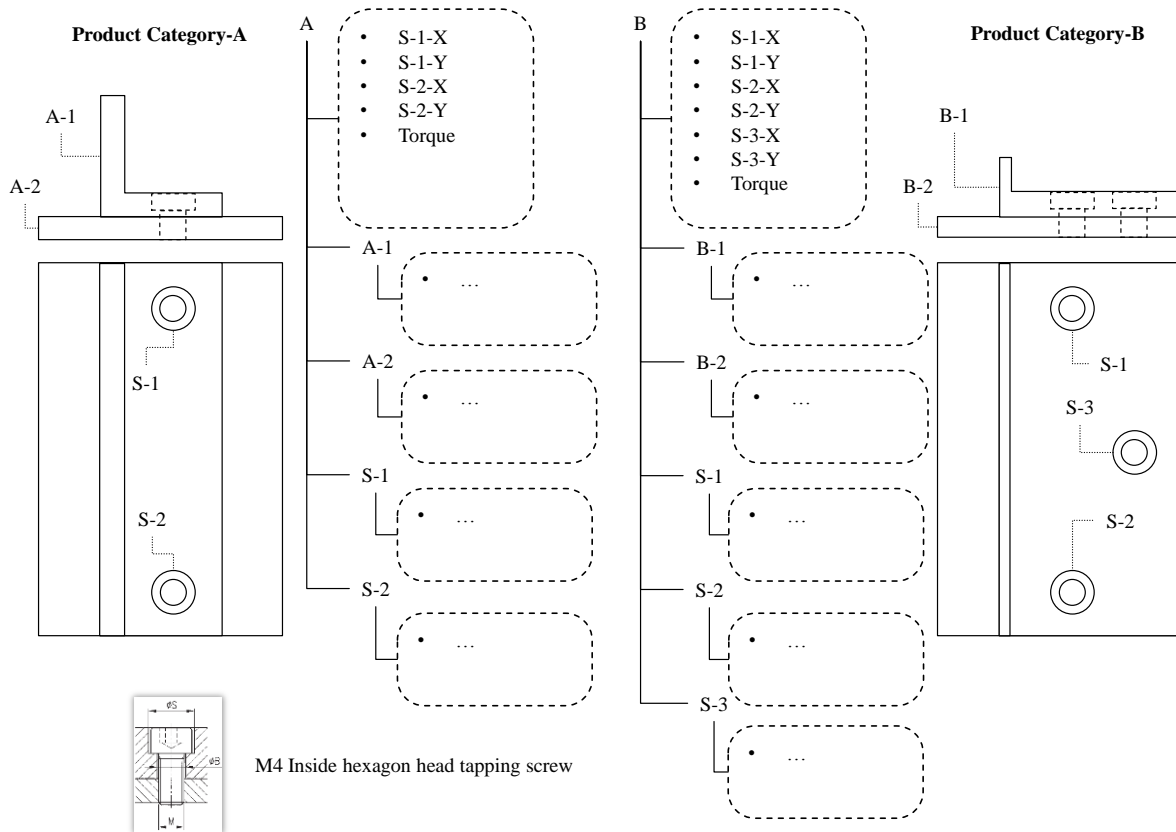


Figure 5.1: Example product categories to be mounted in assembly line.

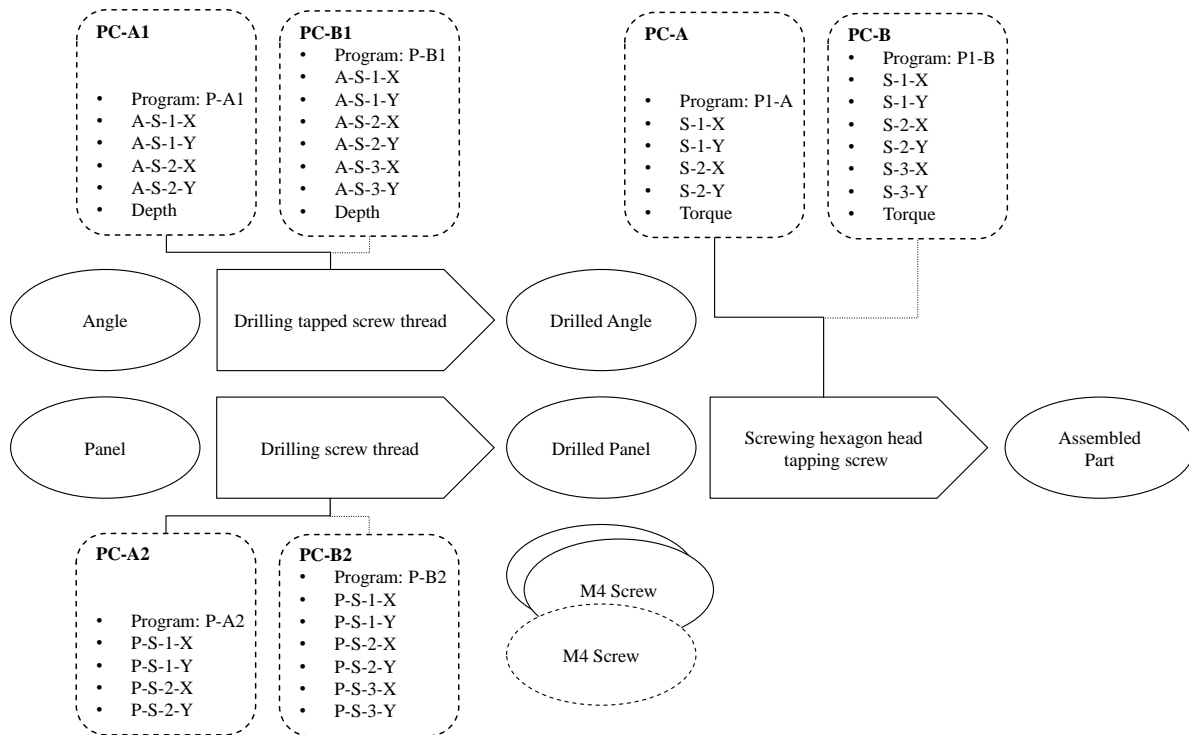


Figure 5.2: Example process segments for assembly of product categories of Figure 5.1.

The previous example explains how the K-RAMP information model and the K-RAMP reasoning process can be applied on an assembly scenario. Due to the recommended process segments, the underlying process operations and process operation setups as well as the appropriate assembly equipment can be derived by the ramp-up team.

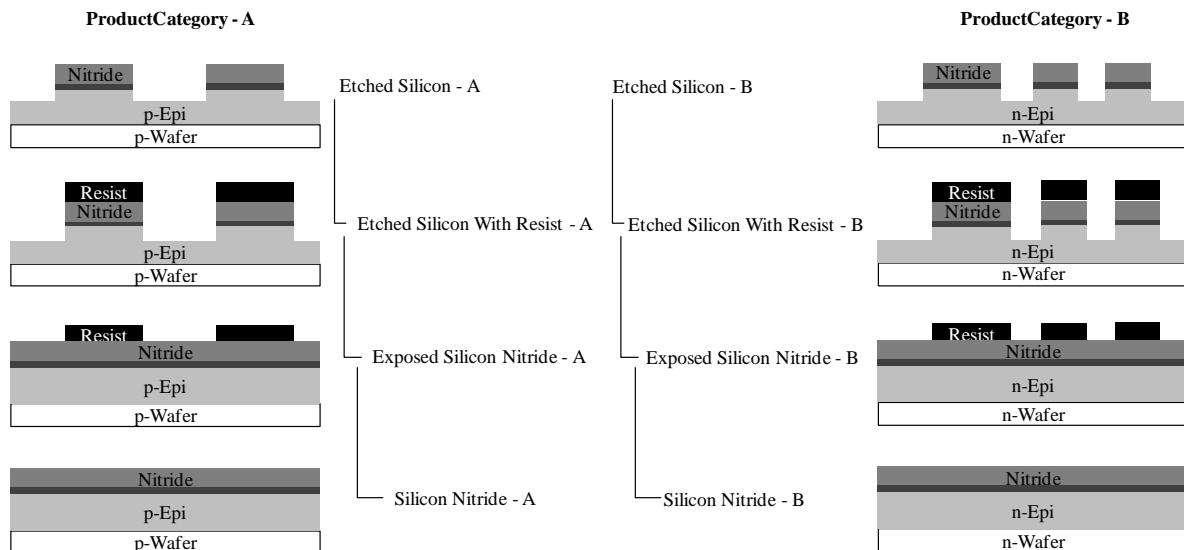


Figure 5.3: Example product categories of semiconductors – source [61, p. 587].

The K-RAMP information model and reasoning process is also applicable for semiconductor manufacturing. This shall be demonstrated by the next example. Figure 5.3 shows the

decomposition hierarchy of the lowest layers of a semiconductor wafer. There shall be two different product categories which are distinct from each other due to the different types of substrate (p-type and n-type).

Although, a semiconductor wafer is not the result of an assembly process as discussed in the previous example the decomposition structure can be organized in a similar way. Same as in the assembly example, each product category comprises its individual specifications. The specifications of the product categories “Exposed Silicon Nitride-A” and “Exposed Silicon Nitride-B” may share characteristics like “THK-Resist” or “Reticle-Id” which specify the thickness of the layer of photo resist respectively the identification of the photo mask.

Obviously, the same pattern can be applied to specify the process segments, product categories and products of a semiconductor production system as it was applied earlier for an assembly process. Due to this perception, the generalization of the K-RAMP concept appears to be valid. But also by comparison with research on more conceptual layers, this assumption can be confirmed.

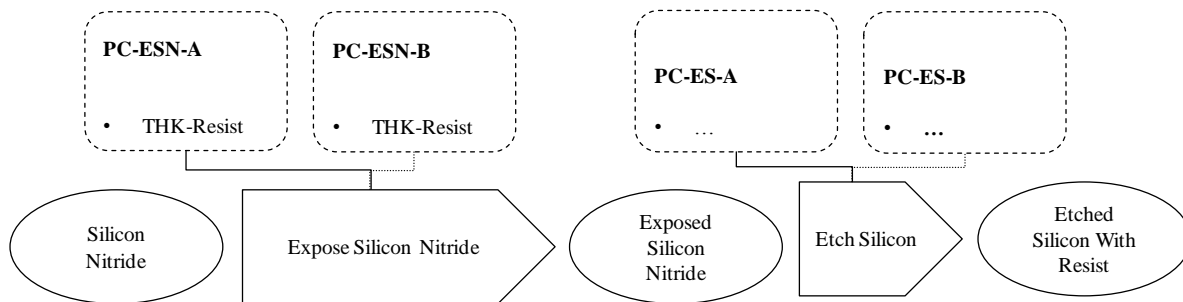


Figure 5.4: Example process segments for exposure and etch of product categories of Figure 5.3.

Mertens [62, pp. 64-146] introduces a metamodel for the management of feature-based information. The described structure of the metamodel [62, p. 65] perfectly maps to the structure of the ontology models of K-RAMP. Mertens describes the structure of machines, services and service-oriented systems. As machines are products, the model described by Mertens can be mapped to the need of K-RAMP for the description of products (a.k.a. feature carrier in Mertens’ terminology) and product categories (a.k.a. feature carrier type). The consideration of services and service-oriented systems in this approach can be mapped to process capabilities in the K-RAMP information model. The reasoning of encloses relations between specifications is supported by [62, pp. 84-85] in order to achieve “a semiorder and comparison on this order”. It can be therefore concluded that the designed structure of K-RAMP’s information model is of general validity. Moreover, Mertens work is complementary in order to develop a specific management system of specifications and specification sets in a proper way.

The K-RAMP TBOX ontology models enable the specification of branch-specific ABOX ontology models or taxonomies. K-RAMP satisfies the initially stated constraint C_1 accordingly. The evaluation of K-RAMP demonstrates how such branch-specific models are specified. In the same way, it is possible to establish enterprise-specific or even branch-specific models for any branch of manufacturing industries accordingly.

While planning the sequence of a manufacturing process, also interdependencies between process operations and between process segments need to be considered. The same is also true with respect to time constraints. How does K-RAMP deal with these constraints? K-RAMP does not need to deal with those constraints at all! The following constraints shall be discussed in the next sections.

- Contamination: After some equipment has processed some particular material, it cannot be used for other materials without cleaning. Otherwise, a negative impact on the quality of results has to be faced. An example of this constraint with respect to the cake-baking case study is the beating of yolk and egg white. It is not possible to beat egg whites with the mixer which has been just used for beating yolk. It must be cleaned before. However, it is possible to use the mixer for beating yolk after it was used for beating egg white without cleaning. Another example is the contamination of process chambers with copper during the processing of semiconductors.
- Time constraint: After a particular process step, it is necessary to consider strict timing constraints in order to avoid an impact on the quality of the subsequent process operation or process segment. Recalling the cake-baking case study again, after melting the chocolate there are only a few minutes for coating the chocolate cake with the liquid chocolate. Otherwise, the liquid chocolate is cooling down again. In the semiconductor industry, there are oxidation effects. For this reason, subsequent deposition processes must be sometimes performed within strict time windows.
- Simultaneous activities: Sometimes, two activities must be executed simultaneously due to technical needs. For instance, folding yolk butter cream and beaten egg whites must be performed simultaneously with sieving of flour on the folded dough.

These constraints are mastered by production control systems (MES) or by appropriate planning of process segments. With respect to contamination, cleaning processes are out of scope of K-RAMP. K-RAMP recommends, how a particular process segment shall be setup by reusing an existing process segment and some modifications (e.g., alternating the used incoming material or a mechanical part of the equipment). However, it is part of the information base of the MES to consider incompatibilities between process operations or process segments along the execution of a manufacturing process. If such contamination occurs, the MES must trigger a cleaning operation of the device before it can be used for the next process operation.

If time constraints need to be considered between process operations then the time constraint shall be handled within the scope of a single process segment. Usually, such time constraints are already mastered this way. For instance, two process operations for deposition which are applied on semiconductor wafers are linked in an immediate sequence in order to avoid oxidation effects. A metrology operation terminates the process segment afterwards. From the perspective of K-RAMP, process segments are the atomic elements which can be considered for reuse. By following the previously introduced pattern, such time constraints can be mastered without limitation of the value of K-RAMP. Recalling the example about melting chocolate and coating the chocolate cake with the liquid chocolate, it may be useful to setup a process segment “coating with liquid chocolate” with the process operations “melting” and “coating”. For the reasoning of K-RAMP, it is not relevant whether this process segment is applied for coating the Viennese chocolate cake or Eclairs (another kind of sweets). From the perspective of the process segment setup and the process capability, the used product category and the handling instruction are varied but the sequence of time critical process operations remains unchanged. It may be useful, to apply the process operation “melting” also for other purposes than coating. Also this need can be handled easily by introducing another process segment “melting” which comprises only the process operation “melting”.

Simultaneous activities can be considered as a specific form of time constraints (time constraint 0). Therefore, also simultaneous activities are organized within a common process segment as described before.

According to its current design, K-RAMP recommends matchmaking scenarios based on penalty points. This concept demonstrates possibilities to consider efforts for the adaptation of reusable knowledge. These penalty points are also derived from encloses and overlaps relations between information items. Currently, those relations do not consider the quality of enclosing or overlapping. For instance, two specification ranges which are almost the same but still overlapped are considered with the same quality as two specification ranges with minimal overlapping. Considering this quality of overlapping, is part of future enhancements of the K-RAMP information model and the K-RAMP process.

The described reasoning process and information model do not consider cost-related features of recommended subproducts or process segments. Further research needs to be performed in order to integrate those features to the penalty systematic as well.

5.3.2 Presupposed architecture of production ICT

Nowadays, there is a broad and branch-independent common understanding about the planning elements which are used in K-RAMP. However, what are the detailed premises which need to be realized in order to integrate an interdisciplinary knowledge base aligned with the K-RAMP concept across multiple production systems? Such an environment shall be named “K-RAMP application” in the sequel.

As highlighted in the previous sections, automated recommendation concerning the reuse and the adjustment of reusable information in case of a new product’s ramp-up comes with a set of premises.

- All process segments deliver metrology data. Consequently, there are process capabilities specified by characteristics which are determined in this context.
- Process capabilities are managed within production systems, consequently decoupling device-dependent process-related information from product-related information. This aspect contributes to the vertical integration of product-related information and process-related information.
- A commonly managed hierarchy of product categories which allows integration of product-related information horizontally along the supply chain.
- A categorization of products through product categories, consequently matching products and subproducts along the supply chain (horizontal integration of information) due to their membership in common product categories.
- Products, product categories and process capabilities are described uniquely as specification sets, thus enabling matchmaking between individuals of these concepts through the discussed relations encloses and overlaps. This matchmaking comprises vertical and horizontal integration of information.
- A taxonomy of production techniques across involved production systems enables the categorization or process segments, thus an approach to a common process-related taxonomy across production systems. Consequently, this aspect contributes to the horizontal integration of process-related information.
- A categorization of process segments through production techniques
- A commonly managed set of characteristics – and consequently a commonly managed set of engineering units – across involved production systems is a further contribution to

horizontal integration. Actually, this aspect is the core of all opportunities for matchmaking being listed previously.

- A comprehensive management system for features (a.k.a. as specifications and specification sets in K-RAMP) [62].

Such premises have to be implemented on the organizational level of a production system first. From a technical perspective an MES or equivalent production ICT for such management of product basic data and process basic data is presumed. This is commonly managed in the domain of computer integrated manufacturing (CIM), namely

- computer aided design (CAD),
- product data management (PDM),
- computer aided production planning (CAPP).

The architectural overview of K-RAMP (see Figure 3.26) is an adequate starting point for answering this question.

In any case and according to ISO 9001 [58, pp. 2-3], the explicit control of documents and records and therefore the explicit control of information which is imported from the K-RAMP knowledge store is presupposed. Usually, a workflow management component, respectively a change management component of one of afore mentioned components, is utilized for this purpose.

For the original production system it is only required to upload (export) product information and device-independent process-related information to the local K-RAMP knowledge store. For the target production system the full integration is necessary. First, it is necessary to upload (export) product-related information as well as device-independent process-related information from the MES to the local K-RAMP knowledge store. Secondly, device-dependent process-related information is uploaded. As already highlighted in Chapter 3.5, only recommendations are provided by the K-RAMP UI component. However, the overall maintenance of product basic data and process basic data remains in the responsibility of the MES.

Decoupling of the existing production ICT from the K-RAMP knowledge base through a API, enables information exchange between heterogeneous production ICT environments. For this reason K-RAMP does support constraint C_2 .

For the purpose of better management of information domains, the ontology models can be even more fragmented, as this was done as part of the evaluation. It may be useful to create separate ABOX ontology models for each product, for each process segment, for groups of equipment, or even ABOX ontology models which are aligned with the context of comprehensive change processes of the CMS. As all these ontology models are based on the same set of K-RAMP TBOX ontology models, all results of the evaluation still remain valid.

It is possible to export these ABOX ontology models in the format of OWL/XML syntax. It is important in general that also the reasoning results are downloaded from the K-RAMP knowledge store as part of this information stream. By using OWL/XML it is possible to use standard XML libraries for programming the parsing and the generation of OWL/XML streams.

5.3.3 Quality of provided product and process basic data

In which formats and with which quality must product basic data and process basic data be provided? The documented input forms which are applied during the evaluation (see Chapter 4.4) provide already a good insight.

5.3.4 Organizational premises

5.3.4.1 Unified taxonomy models

The evaluated concept of K-RAMP does also presume some organizational premises as already listed above. It considers unified taxonomy models for engineering units, production techniques and product categories. This step is essential from an organizational perspective within the boundaries of a company and it may be useful between customers and suppliers (supply chain) as well. Such unified taxonomy models improve the reasoning quality of K-RAMP as there is no risk of fuzzy matching of similar entities at the very first step of matchmaking at the target production system.

However, if there is no unification feasible (e.g., along the supply chain) it may be possible to apply ontology alignment strategies as they are, for instance, introduced by Ehrig [63]. Such ontology alignment strategies may introduce additional uncertainty at the root of K-RAMP's matchmaking process. For instance, two individuals may be considered as the same characteristic or production technique. If this assumption cannot be made with 100% certainty, there is a risk of totally wrong recommendations. Consequently, organizational measures in order to unify taxonomy models or ontology models as already discussed in Chapter 3.3 are recommended.

5.3.4.2 Design of products for reuse

Designing of products based on standardized and reusable products is a good design principle. Advanced manufacturing industries, like semiconductor manufacturing or automotive manufacturing, are building their products already on reusable modules. The results of K-RAMP are the more complete the more comprehensive principles of reusable design are applied for products. For instance, Ong et al. are summarizing possible design principles for the design of product platforms which should be considered for the organization of product designs [16, pp. 81-112] simultaneously with the implementation of K-RAMP.

5.4 Summary

It can be shown that the design of K-RAMP answers the research questions of this work. However, there is no ontology model which is solely based on the specification of Semantic Web. Therefore, the research questions have been answered by a hybrid design.

In the previous sections, it is demonstrated that the result of the applied case study can be also adopted for real-world scenarios. Moreover, several complementary research work and publications support this assumption.

The aim of K-RAMP is the recommendation of reusable production knowledge for the production of a new product. Technical constraints (e.g., contamination) which cause specific logistic planning at single machines are subject to the production control system (a.k.a. MES). Time constraints or constraints with respect to the need for simultaneous actions need to be considered in the planning phase of process segments. Such measures are already taken today by

manufacturing enterprises in order to master complexity and flexibility of their production systems.

There are premises which need to be considered for an effective implementation of K-RAMP. A comprehensive listing of these premises is provided in Chapter 5.3.2. Moreover, it is recommended that a MES is in place which implements the management of product basic data and process basic data of the respective production system.

6 Summary

and future work

6.1 General conclusions

K-RAMP shows promising results and should contribute to more deterministic and more efficient product ramp-up processes. Recent initiatives, like *Industrie 4.0* or the *IIoT*, are envisioning the digitalization of the product value chain. This vision comprises the product design process and the product ramp-up process between enterprises which are participating to the product's value chain, probably on the level of cloud manufacturing. The search for suppliers of subproducts or production services and matchmaking of the product-related and the process-related information fragments of all participants shall become an important need of this vision. Keeping this need in mind, K-RAMP is not only beneficial within the boundaries of an enterprise but also between enterprises along the value chain.

However, as also considered by the mentioned initiatives, there is tremendous need for standardization and unification on all technical and organizational levels [64]. K-RAMP is introducing some additional aspects which are addressed in Chapter 6.2.

K-RAMP presumes the management of process capabilities as a contract between the product design and the setup of production equipment. Depending on the architecture of the production equipment, automated upload of CNC-programs (a.k.a. equipment recipes) available. Alternatively, the production equipment provides human machine interfaces (HMI) which are used by operating personnel in order to setup the machine and based on handling instructions. Also this strategy of equipment setup requires an architecture which enables the control of the process through an HMI. The third strategy considers engineering or reengineering of the production equipment or parts of the production equipment. As this is out of scope of K-RAMP, Chapter 6.3 addresses ideas in order to integrate K-RAMP with research work in the domain of equipment engineering.

6.2 Need for unified taxonomy and ontology models

As already mentioned in the previous chapters, the unification of particular taxonomy models or ontology models is highly recommended. Such unification simplifies the search for products and product categories as well as for production services and matchmaking between both domains. Due to *Industrie 4.0* or the *IIoT*, the need for respective predictability of product ramp-up involves multiple enterprises along a digitalized value chain. This envisioned development raises the need for unification and standardization of taxonomy models or ontology models in aforementioned domains. As the K-RAMP information models and the K-RAMP process are implemented by the help of such models, these models are proposed as an initial point of such initiatives.

In the context of the K-RAMP design, specific information domains are isolated as useful for unification. These information domains are specified by the information models of DP_{1,1} (*Engineering Units*), DP_{1,2} (*Characteristics*), DP_{1,4} (*Production Techniques*), DP_{1,6} (*Product Categories*), DP_{1,5} (*Products*) and DP_{1,7} (*Process Capabilities*). The respective ontology models are *kr-unit*, *kr-char*, *kr-spec*, *kr-prodn-tech*, *kr-prod*, *kr-prod-cat* and *kr-prod-cap* may serve as TBOX of standardized ABOX ontology models.

Ontology models for engineering units are already researched through the QUDT ontology [48] or the OM ontology [65]. The OM ontology or the QUDT ontology could replace the currently proposed *kr-unit* and *kr-char* ontology of K-RAMP by integrating it with the *kr-spec* ontology. Complementarily, there is the need to unify the terminologies of engineering units and characteristics on a branch-independent level and to make it available on the base of aforementioned ontology models (Figure 6.1). Some industries, like the semiconductor industry, have recently started to unify characteristics on the level of equipment data collection [66] or for a wider range of applications. The identities of particular equipment parameters become unified across all equipment suppliers. With respect to production techniques, it would be useful to establish taxonomy models for effective classification of production services as it is demonstrated by the K-RAMP concept. Currently, there is no awareness about such initiatives.

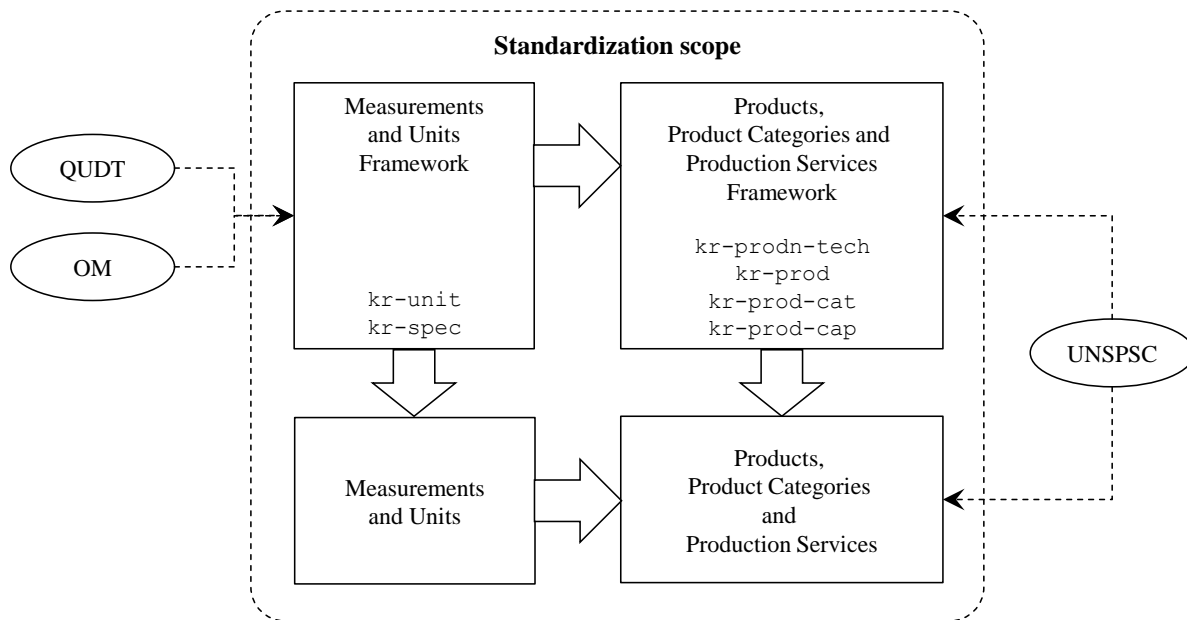


Figure 6.1: Proposed scope and organization for standardization.

The initiative UNSPSC of the United Nations Organization (UNO) provides a taxonomy [53] for the unification of products and services. The proposed taxonomy may be considered for standardized products, product categories and process capabilities on the base of the TBOX models which are introduced by K-RAMP. Complementary to a hierarchical structure, such concepts need to be specified with much more detail. Such specifications, regulations and laws are already in place for different industrial branches but without a unique structure, for instance, there are domestic and EU regulations about chocolate products or preserves, or there are international standards about screws and other assembling aids. These detailed specifications are not yet considered by UNSPSC or another more enhanced ontology. Partially, such standards are used in the cake-baking ontology models in order to demonstrate that such formalization of product standards and product regulations is achievable.

For the support in digital supply chains, there is the need to provide a cloud-based platform where product providers and production service providers are able to submit public specifications of their products and services. From the perspective of consumers of this platform, a matchmaking process as it is introduced by K-RAMP helps to bring proposed products and production services in synch with designed products which need to be manufactured.

6.3 Integration with the equipment engineering domain

In the discussed context, K-RAMP presumes machine architectures where equipment recipes are uploaded to automated equipment, or the setup of equipment is performed by operating personnel through HMIs. A broader interpretation of equipment recipes or handling instructions is the specification of the physical setup of the involved equipment. The handling instruction may, for instance, request a drill with a certain diameter. A human operator reads this handling instruction and mounts the requested drill manually. This interpretation is still within the scope of K-RAMP.

However, even more complex modification of equipment may be required due to the needs of a new product. Such modifications may require reengineering or new engineering of equipment. In this case the equipment engineering team needs to be involved to the product ramp-up as well. In such special cases, it is useful to integrate existing research work of Moser et al. [30] [29] and AutomationML [67] with the ontology models of K-RAMP.

7 Appendix

7.1 Cake recipes

7.1.1 Viennese Sachertorte

In this chapter the recipe of the Viennese Sachertorte is cited in completely according to [68] and can be used in order to refer from this original recipe to the specified ABOX-ontology model of the original bakery. Ingredients for a springform pan with 22 - 24 cm diameter are

- 140 g butter at room temperature
- 110g icing sugar
- core of a ½ vanilla pod
- 6 egg yolks
- 6 egg whites
- 130 g cooking chocolate
- 110 g granulated sugar
- 140g plain flour
- 200 g apricot jam
- Butter and flour for the mold

For the chocolate couverture the following ingredients are needed

- 200 g granulated sugar
- 125 ml of water
- 150 g chocolate

The cake is made according to the following procedure. Whipe softened butter with powder sugar and vanilla until creamy. Gradually add the egg yolks and stir to thick foamy mass. Meanwhile, melt the chocolate in a bain and then stir. Beat egg whites with sugar until the stiff egg white is shiny and can be cut with resistance. This stiff egg white is put on the foamy mass, sieving flour on it and fold carefully with a wooden spoon.

Place baking paper at the bottom of the springform pan, spread out the edge with butter and dust with flour. Add the mass, smooth it and bake it in the oven (preheated) at 170 ° C for one hour. The oven door can slightly open during the first 10-15 minutes. The cake is fully baked when a gentle pressure with the finger is gently replied.

Then place the mold upside down and leave to cool (room temperature). After about 20 minutes remove the paper, turn around and leave to cool completely in the form in order to level the surface. Remove from the mold and divide horizontally with a knife. Stir the slightly warmed jam smooth, spread on both cake bases and put them together again. Coat on the outer sides slightly with jam and let it dry.

For the icing boil sugar and water for 5-6 minutes, then allow to cool (approx. 35°C). Melting chocolate in bain, gradually stir into the treacle to a thick, smooth glaze. Pour the lip-warm glaze in one move on the cake and spread smooth with the fewest possible strokes. Let it dry until the glaze has solidified.

7.1.2 Meraner Torte

In this chapter the recipe of the Meraner Torte is cited in completely according to [69, p. 22] and can be used in order to refer from this recipe to the specified ABOX-ontology model of the target bakery.

7.1.3 Eclairs

In this chapter the recipe of Eclairs is cited in completely according to [69, p. 94] and can be used in order to refer from this recipe to the specified ABOX-ontology model of the target bakery.

7.2 Dictionary

CAD	See Computer aided design
CAPP	See Computer aided production planning
CAQ	See Computer aided quality
Computer aided design	Software for designing a particular product by the help of computers.
Computer aided production planning	Software for planning of production processes including required process capabilities, resource management including the assignment of process capabilities to production machines or skills to human operators or the availability of production resources at a certain time (scheduling of production resources), as well as the management of handling instructions or machine recipes. The same software is also capable to drive the path of each workpiece through the production system, by active invocation of appropriate production resources at the right time.
Computer aided quality	A comprehensive software systems which collect quality data, provide functions for data analysis as well as workflows for escalation management.
Discrete production	Discrete production processes are producing individually countable and measurable workpieces of products (e.g., cars, cell phones, integrated circuits). It is the opposite of flow production where fluids, gases, powder material is produced, which cannot be separated into individual occurrences.
First pass yield	The first pass yield (FPY) is calculated as the portion of defect-free parts of a specific subproduct which are passing a particular process segment of the overall production process [27, pp. 179-180] at the first pass (without rework).
FPY	See first pass yield
Goods receipt	A process segment where incoming material of external suppliers is measured before it is delivered to the internal production process.
ICT	See Information and communication technology
Information and communication technology	Computer systems, hardware and logical network in order to run a production system.
Manufacturing execution system	See Production control system.
MES	See manufacturing execution system.
Metrology step	A process operation where samples of the manufactured workpieces are measured in order to determine whether they still meet the designed characteristics.
Process average	The average value over all samples of a process segment which are measured with respect to a particular characteristic.
Process job	One single execution of a process operation or a process segment. Due to this assignment of process jobs on two layers, a process job on the process segment level owns all process jobs which are executed in its context on the process operation level.
Process operation	One processing step or metrology step.
Process segment	An arbitrary sequence of processing steps followed by one or more metrology steps
Processing step	A process operation where a workpiece or, more generally, material is treated.
Production control system	A software system which is part of the production-IT and responsible for the dispatching of material within the production system. It ensures that a product is produced in accordance to a specified process plan and all required information is at the right machine or work place together with the dispatched material.
Production resource	Man and machines which are treating, measuring, transporting or storing workpieces in a production system.

Production system	All production systems, when viewed at the most abstract level, might be said to be “transformation processes”—processes that transform resources into useful goods and services. The transformation process typically uses common resources such as labour, capital (for machinery and equipment, materials, etc.), and space (land, buildings, etc.) to effect a change. Economists call these resources the “factors of production” and usually refer to them as labour, capital, and land. Production managers refer to them as the “five M’s”: men, machines, methods, materials, and money. [70]
Production volume	The number of workpieces which are created of a particular product.
RDBMS	Relational database management system
RTY	See Rolled throughput yield
Rolled throughput yield	The so called rolled throughput yield (RTY) represents the portion of all produced workpieces which are passing the overall production process at the first pass, which means that there is no potential rework required in order to correct defects [27, pp. 179-180]. The RTY is calculated for each product which is produced in the production system.
SameAs	An association between two entities which specifies their identity. SameAs (a, b) implies that a is identical with b.
Unified Resource Identifier	A globally unique identification for each entity which is comprised in a Semantic Web ontology.
URI	See Unified Resource Identifier
Yield	A ratio between the good workpieces, which have passed the quality assurance, and the total number of workpieces. There are different types of yields. The two being used in this document are the first pass yield (FPY) and the rolled throughput yield (RTY).

7.3 Bibliography

- [1] ISA, ANSI/ISA-95.00.01-2000: Enterprise Control System Integration - Part I: Models and Terminology, Research Triangle Park, North Carolina, USA: ISA–The Instrumentation, Systems, and Automation Society; ISBN: 1-55617-727-5, 2000.
- [2] SEMATECH, Inc., Computer Integrated Manufacturing (CIM) Framework Specification Version 2.0, Technology Transfer # 93061697J-ENG, 1998.
- [3] VDI/VDE - Verein deutscher Ingenieure/Verein der Elektrotechnik Elektronik Informationstechnik e. V., VDI/VDE 3682 - Formalisierte Prozessbeschreibungen [eng. Formalisierte Prozessbeschreibungen], Düsseldorf: Verein deutscher Ingenieure, September 2005.
- [4] M. Slamanig and H. Winkler, "Wissensmanagement bei Produktwechselprojekten [eng. Knowledge management in product change projects]," *ZWF Wissensmanagement (www.zwf-online.de - document number ZW110416)*, no. 10, pp. 893-900, 2010.
- [5] E. Abele, T. Meyer, U. Näher, G. Strube and R. Sykes, Global Production: A Handbook for Strategy and Implementation, Berlin, Heidelberg: Springer; ISBN 978-3-540-71652-5, 2008.
- [6] S. Kinkel and S. Maloca, "Modernisierung der Produktion - Flexibilitäts- und Stabilitätsstrategien in der deutschen Industrie [eng. Modern Production - Flexibility and stability strategies in the German industry]," December 2010. [Online]. Available: <http://www.isi.fraunhofer.de/isi-wAssets/docs/i/de/pi-mitteilungen/pi54.pdf>. [Accessed 6 May 2016].
- [7] J. Schmidt-Dilcher and H. Minssen, "Ewige Baustelle PPS - Strukturkonservative Rationalisierungsmuster im Maschinenbaubetrieb [en. infinite construction site CAPP - structurally conservative rationalization patterns in systems engineering units]," in *Kooperationsnetze, flexible Fertigungsstrukturen und Gruppennetzwerke - Ein interdisziplinärer Ansatz [en: cooperation networks, flexible manufacturing structures and group networks - an interdisciplinary approach]*, Opladen, Leske + Budrich; ISBN 978-3-322-97326-9, 1996, pp. 83-117.
- [8] M. Slamanig and H. Winkler, "Management of product change projects: a supply chain perspective," *International Journal of Services and Operations Management*, vol. 11, no. No. 4, pp. 481-500; ISSN 1744-2370 (Print), 1744-2389 (Online), 2012.
- [9] P. Kurttila, M. Shaw and P. Helo, "Model Factory concept – Enabler for quick manufacturing capacity ramp-up," in *European Wind Energy Conference and Exhibition, EWEC - Volume 4*, 2010.
- [10] J. C. Fransoo and A. G. de Kok, What Determines Product Ramp-up Performance? A Review of Characteristics Based on a Case Study at Nokia Mobile Phones, Beta, Research School for Operations Management and Logistics; ISSN 1386-9213, 2007.
- [11] C. Terwiesch, R. E. Bohn and K. S. Chea, "International product transfer and production ramp-up: a case study from the data storage industry," in *R&D Management 31, 4*, Oxford, UK and Malden MA, USA, Blackwell Publishers Ltd., 2001, pp. 435-451.
- [12] C. Terwiesch and Y. Xu, "The Copy Exactly Ramp-up Strategy: Trading-off Learning with Process-Change," 2003.

- [13] K.-J. Bjelkenäs, „Ramp up in manufacturing - A coming winner?“, [Online]. Available: http://www1.mtov.lth.se/ulimage/exjobb/article_karljohan_bjalckenas.pdf. [Zugriff am 19 07 2012].
- [14] A. Bramley, D. Brissaud, D. Coutellier and C. McMahon, *Advances in Integrated Design and Manufacturing in Mechanical Engineering*, Netherlands: Springer, 2005.
- [15] Sang Jin Oh, Seoul National University, "A Study of the Foundry Industry Dynamics," Massachusetts Institute of Technology, 2010.
- [16] S. K. Ong, Q. L. Xu and A. Y. C. Nee, *Design Reuse in Product Development Modeling, Analysis and Optimization*, Singapore: World Scientific Publishing Co. Pte. Ltd.; ISBN-13 978-981-283-262-7, 2008.
- [17] N. P. Suh, *The Principles of Design*, Oxford: Oxford University Press; ISBN 0-19-504345-6, 1990.
- [18] N. P. Suh, *Axiomatic Design - Advances and Applications*, New York, Oxford: Oxford University Press, ISBN 978-0-19-513466-7, 2001.
- [19] F. Mariano, G. P. Asunción and N. Juristo, "METHONTOLOGY: From Ontological Art Towards Ontological Engineering," *AAAI Technical Report*, pp. 33-40, 1997.
- [20] ISA, ANSI/ISA-95.00.02-2001: *Enterprise Control System Integration - Part 2: Object Model Attributes*, Research Triangle Park, North Carolina, USA: ISA-The Instrumentation, Systems, and Automation Society; ISBN: 1-55617-773-9, 2001.
- [21] ISA, ANSI/ISA-95.00.03-2005: *Enterprise Control System Integration - Part3: Activity Models of Manufacturing Operations Management*, Research Triangle Park, North Carolina, USA: ISA-The Instrumentation, Systems, and Automation Society; ISBN: 1-55617-955-3, 2005.
- [22] D. Nenni and P. McLellan, *FABLESS: The transformation of the semiconductor industry*, SemiWiki.com; ISBN: 978-1-4675-9307-6, 2013.
- [23] P. Macleod, *A Review of Flexible Circuit Technology and its Applications*, PRIME Faraday Partnership; ISBN1-84402-023-1, 2002.
- [24] M. Hüttenrauch and M. Baum, *Effiziente Vielfalt: Die dritte Revolution in der Automobilindustrie* (engl. Efficient variety: the third revolution of the automotive industry), Berlin Heidelberg: Springer; ISBN 978-3-540-72115-4, 2008.
- [25] F. W. Breyfogle III, *Implementing Six Sigma - Second Edition*, Austin, Texas: John Wiley and Sons, Inc., ISBN 0-471-26572-1, 2003.
- [26] E. Dietrich and A. Schulze, *Statistische Verfahren zur Maschinen- und Prozessqualifikation* [eng. Statistical Methods for Machine and Process Qualification], Hanser, Ed., Munich, Vienna: ISBN 3-446-22077-1, 2003.
- [27] J. Wappis and B. Jung, *Taschenbuch - Null-Fehler-Management - Umsetzung von Six Sigma* [engl. Zero-defects management - Implementation of six sigma], 3 ed., Munich, Vienna: Carl Hanser Verlag; ISBN 978-3-446-42262-9, 2010.
- [28] ISO - International Standards Organization, "ISO 21747-2006 - Statistical methods - Process performance and capability statistics for measured quality characteristics," March 2007. [Online]. Available: <http://www.beuth.de/langanzeige/DIN-ISO-21747/de/93888258.html>. [Accessed 17 July 2011].

- [29] T. Moser, S. Biffel, W. D. Sunindyo and D. Winkler, "Integrating Production Automation Expert Knowledge Across Engineering Stakeholder Domains," in *IEEEExplore Digital Library*, Krakow, 2010.
- [30] T. Moser, R. Mordinyi, D. Winkler, M. Melik-Merkumians and S. Biffel, "Efficient Automation System Engineering Process Support Based on Semantic Integration of Engineering Knowledge," in *16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2011)*, 2011.
- [31] K. Konrad, M. Hoffmeister, M. Zapp, A. Verl und J. Busse, „Enabling Fast Ramp-Up of Assembly Lines through Context-Mapping of Implicit Operator Knowledge and Machine-Derived Data,“ *International Federation for Information Processing*, pp. 163-174, 2012.
- [32] S. Ruegsegger, A. Wagner, J. Freudenberg and D. Grimard, "Optimal Feedforward Recipe Adjustment for CD Control in Semiconductor Patterning," in *Characterization and Metrology for ULSI Technology: 1998 International Conference*, 1998.
- [33] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist - Effective Modelling in RDFS and OWL*, Elsevier; ISBN 978-0-12-285965-5, 2011.
- [34] W3C - The world wide web consortium, "RDF 1.1 Primer - W3C Working Group Note," 24th June 2014. [Online]. Available: <http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/>. [Accessed March 2015].
- [35] R. Cyganiak, D. Wood, M. Lanthaler, G. Klyne, J. J. Carroll and B. McBride, "RDF 1.1 Concepts and Abstract Syntax," W3C - World Wide Web Consortium, 25th February 2014. [Online]. Available: <http://www.w3.org/TR/rdf11-concepts/>. [Accessed 30th April 2015].
- [36] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider and S. Rudolph, "OWL 2 Web Ontology Language Primer (Second Edition)," 11th December 2012. [Online]. Available: <http://www.w3.org/TR/owl2-primer/>. [Accessed March 2015].
- [37] S. Harris, A. Seaborne and E. Prud'hommeaux, "SPARQL 1.1 Query Language," W3C - World Wide Web Consortium, 21st March 2013. [Online]. Available: <http://www.w3.org/TR/2013/REC-sparql11-query-20130321/>. [Accessed 30th April 2016].
- [38] Forschungsunion, National academy of science and engineering, "Recommendations for implementing the strategic initiative INDUSTRIE 4.0 - Final report of the Industrie 4.0 Working Group," April 2013. [Online]. Available: http://forschungsunion.de/pdf/industrie_4_0_final_report.pdf. [Accessed 8th March 2015].
- [39] Lehrstuhl für Prozesstechnik of Aachener Verfahrenstechnik at RWTH Aachen University, Germany, "OntoCAPE.org," December 2011. [Online]. Available: <http://www.ontocape.org/>. [Accessed 13 June 2014].
- [40] J. Morbach, A. Wiesner and W. Marquardt, "OntoCAPE.org," December 2011. [Online]. Available: <http://www.ontocape.org/>. [Accessed 13 June 2014].
- [41] D. Winkler and S. Biffel, "Process Automation and Quality Management in Multi-Disciplinary Engineering Environments," Vienna University of Technology, 03 2010. [Online]. Available: http://cdl.ifs.tuwien.ac.at/files/CDL_M1_03%20Process_Automation_Quality_M

anagement%20Research%20100321.pdf. [Accessed 10 03 2013].

- [42] R. Willmann, E. Serral, W. Kastner and S. Biffl, "Shortening of Product Ramp-up by using a Centralized Knowledge base," in *Industrial Informatics 2013*, 2013.
- [43] R. Willmann, S. Biffl and E. Serral Asensio, "Determining qualified production processes for new product ramp-up using semantic web," in *Proceedings of the 14th International Conference on Knowledge Technologies and Data-Driven Business (I-KNOW)*, Graz, 2014.
- [44] OMG - Object Management Group, "Documents Associated With Unified Modelling Language™ (UML) Version 2.5," 2016. [Online]. Available: <http://www.omg.org/spec/UML/2.5/>. [Accessed 11 07 2016].
- [45] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*; 3rd Edition, Upper Saddle River, NJ 07458: Prentice Hall PTR; ISBN 0-13-148906-2, 2005.
- [46] R. Koether and W. Rau, *Fertigungstechnik für Wirtschaftsingenieure (engl. production engineering for industrial engineering and management) - 3rd edition*, München: Hansa; ISBN 978-3-446-41274-3, 2008.
- [47] Deutsches Institut für Normung e. V., DIN 8580:2003-09: Manufacturing processes - Terms and definitions, division, Beuth, 2003.
- [48] R. Hodgson, P. J. Keller, J. Hodges and J. Spivak, "QUDT - Quantities, Units, Dimensions and Data Types Ontologies," 18th March 2014. [Online]. Available: <http://qudt.org/>. [Accessed 1st May 2016].
- [49] aequor, Cargill, Droste Holland, Dutch Cocoa, IFP, MARS netherlands, Olam Cocoa, SOL, vbz.nl, "cocoa & chocolate online," [Online]. Available: <http://www.cacaochocolade.nl/main.php?lng=1&p=inhoud&h=5&g=1&s=7&z=0&sp=.> [Accessed 25 3 2016].
- [50] D. Brickley, R. V. Guha and B. McBride, "RDF Schema (RDFS) - Release 1.1," W3C - World Wide Web Consortium, 25th February 2014. [Online]. Available: <http://www.w3.org/TR/2014/REC-rdf-schema-20140225/>. [Accessed 30th April 2016].
- [51] W3C OWL Working Group, "OWL 2 Web Ontology Language - Document Overview (Second Edition)," W3C - World Wide Web Consortium, 11 December 2012. [Online]. Available: <http://www.w3.org/TR/owl2-overview/>. [Accessed 30 April 2015].
- [52] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosz and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," W3C - World Wide Web Consortium, 21st May 2004. [Online]. Available: <http://www.w3.org/Submission/SWRL/>. [Accessed 30th April 2016].
- [53] United Nations Development Programme, „UNSPSC(R) - United Nations Standard Products and Services Code,“ GS1 US, Inc., 2014. [Online]. Available: <https://www.unspsc.org/>. [Zugriff am 14 05 2016].
- [54] J. Euzenat and P. Shvaiko, "Ontology Matching: State of the Art and Future Challenges," in *IEEE Transactions on Knowledge and Data Engineering - Volume: 25 (Issue: 1) (p158 - 176)*; ISSN-1041-4347, 2011-12-13.

- [55] B. Motik, P. F. Patel-Schneider and B. Parsia, "OWL 2 Web Ontology Language - Structural Specification and Functional-Style Syntax (Second Edition)," World Wide Web Consortium (W3C), 11 December 2012. [Online]. Available: <http://www.w3.org/TR/owl2-syntax/>. [Accessed 13 June 2016].
- [56] European Committee for Standardization, Value management, European Committee for Standardization - EN 12973:2000-06, 2000.
- [57] S. Vajna and C. Burchardt, "Modelle und Vorgehensweisen der Integrierten Produktentwicklung," in *Integrated Design Engineering - Ein interdisziplinäres Modell für ganzheitliche Produktentwicklung [eng. An interdisciplinary model for comprehensive product development]*, Berlin Heidelberg, Springer Vieweg; ISBN 978-3-642-41103-8, 2014, pp. 3-47.
- [58] ISO - International Standards Organization, ISO 9001:2008 - Quality management systems - Requirements, 2008.
- [59] A. Lüder, J. Peschke, T. Sauter, A. Klostermeyer, R. Blume, A. Bratoukhine, D. Diep, F. Murnier, R. Bastillie, K. Sohr, S. Deter, W. Nehr, A. Vontas and U. Hess, "PAPADiS - Plant automation based on distributed systems - Work Package 6 - Assessment and Evaluation," The PAPADiS consortium, 2003.
- [60] S. J. Kluge, Methodik zur fähigkeitsbasierten Planung modularer Montagesysteme [eng. methodology for capability-based planning of modular assembly systems], Heimsheim: Jost-Jetter Verlag; ISBN 978-3-939890-81-2, 2011.
- [61] H. Xiao, Introduction to Semiconductor Manufacturing Technology - Second Edition, Bellingham, Washington: SPIE PRESS; ISBN 978-0-8194-9092-6, 2012.
- [62] M. Mertens, Verwaltung und Verarbeitung merkmalsbasierter Informationen: vom Metamodell zur technologischen Realisierung [eng. Management and processing of feature-based information: from a metamodel to a technical implementation], Düsseldorf: VDI Verlag; ISBN 978-3-18-520708-2, 2011.
- [63] M. Ehrig, Ontology Alignment - Bridging the Semantic Gap, Springer; ISBN 978-0-387-36501-5, 2007.
- [64] VDE - Verband der Elektrotechnik Elektronik Informationstechnik e. V., Die deutsche Normungs-Roadmap Industrie 4.0 [eng. the German standardization roadmap Industry 4.0], 2013.
- [65] H. Rijgersberg, M. van Assem and J. Top, "Ontology of Units of Measures and Related Concepts," *Semantic Web - Linked Data for science and education - Volume 4 Issue 1*, pp. 3-13, 01 2013.
- [66] Semiconductor Equipment and Material International, SEMI E164-0313 - Specification for EDA Common Metadata, Semiconductor Equipment and Material International, 2013.
- [67] AutomationML consortium, "<AutomationML/> The Glue of Seamless Automation Engineering - Whitepaper AutomationML - Part 5 - Communication," www.automationml.org, 2014.
- [68] Kainberger, Ruth; stadt-wien.at, "Sachertorte Originalrezept," Stadt Wien, 2016. [Online]. Available: <http://www.stadt-wien.at/lifestyle/essen-trinken/sachertorte-originalrezept.html>. [Accessed 15 05 2016].

- [69] Salzer, Ueberreuter - Wien, Backen macht Freude - Rezepte Nr. 1-93, Wien: Oetker GmbH Eigenverlag.
- [70] M. Tanenbaum and W. K. Holstein, ""Production System"; Encyclopaedia Britannica Online," Encyclopædia Britannica, Inc, 2016. [Online]. Available: <http://www.britannica.com/technology/production-system..> [Accessed 5 May 2016].

