Network of Excellence

Deliverable D5.2

Identification of Research Gaps in the Common Body of Knowledge
<table>
<thead>
<tr>
<th><strong>Project Number</strong></th>
<th>256980</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project Title</strong></td>
<td>NESSoS</td>
</tr>
<tr>
<td><strong>Deliverable Type</strong></td>
<td>Report</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Deliverable Number</strong></th>
<th>D5.2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title of Deliverable</strong></td>
<td>Identification of Research Gaps in the Common Body of Knowledge</td>
</tr>
<tr>
<td><strong>Nature of Deliverable</strong></td>
<td>R</td>
</tr>
<tr>
<td><strong>Dissemination Level</strong></td>
<td>Public</td>
</tr>
<tr>
<td><strong>Internal Version Number</strong></td>
<td>1.0</td>
</tr>
<tr>
<td><strong>Contractual Delivery Date</strong></td>
<td>September 30, 2012</td>
</tr>
<tr>
<td><strong>Actual Delivery Date</strong></td>
<td>October 15, 2012</td>
</tr>
<tr>
<td><strong>Contributing WPs</strong></td>
<td>WP 5</td>
</tr>
<tr>
<td><strong>Editor(s)</strong></td>
<td>Kristian Beckers (UDE), Stefan Eicker (UDE), Maritta Heisel (UDE), Widura Schwittek (UDE)</td>
</tr>
<tr>
<td><strong>Author(s)</strong></td>
<td>Kristian Beckers (UDE), Stefan Eicker (UDE), Maritta Heisel (UDE), Widura Schwittek (UDE)</td>
</tr>
<tr>
<td><strong>Reviewer(s)</strong></td>
<td>Marianne Busch (LMU), Binh Thanh Nguyen (ETH), Marinella Petrocchi (CNR)</td>
</tr>
</tbody>
</table>
Abstract

This deliverable is devoted to identifying and ameliorating deficiencies in the initial version of the CBK. First, we present a method how to find research gaps based on the content of the CBK, which is based on Kitchenham’s approach to perform different forms of systematic literature reviews. Second, we report on usability tests that we performed on the initial version of the CBK and the suggestions for improvement that resulted from the usability tests.

Keyword List

Interdisciplinarity, Common body of knowledge, Knowledge management, Software engineering, Security engineering, Services computing.
## Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Type of Change</th>
<th>Author(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>Added Section 3</td>
<td>Kristian Beckers</td>
</tr>
<tr>
<td>0.2</td>
<td>Added Sections 2,4</td>
<td>Kristian Beckers</td>
</tr>
<tr>
<td>0.3</td>
<td>Added Sections 1,5</td>
<td>Kristian Beckers</td>
</tr>
<tr>
<td>0.4</td>
<td>Contributed to Sections 1,3,5</td>
<td>Widura Schwittek</td>
</tr>
<tr>
<td>0.5</td>
<td>Improved Sections 2,3,4</td>
<td>Kristian Beckers, Maritta Heisel</td>
</tr>
<tr>
<td>0.5</td>
<td>Improved Sections 1,5</td>
<td>Kristian Beckers, Maritta Heisel</td>
</tr>
<tr>
<td>0.6</td>
<td>Version to be reviewed</td>
<td>Kristian Beckers, Stefan Eicker,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Maritta Heisel, Widura Schwittek</td>
</tr>
<tr>
<td>0.7</td>
<td>Included corrections suggested by first</td>
<td>Kristian Beckers</td>
</tr>
<tr>
<td></td>
<td>internal review</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>Included corrections suggested by second</td>
<td>Kristian Beckers</td>
</tr>
<tr>
<td></td>
<td>internal review</td>
<td></td>
</tr>
<tr>
<td>0.9</td>
<td>Included corrections suggested by third</td>
<td>Kristian Beckers</td>
</tr>
<tr>
<td></td>
<td>internal review</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>Final Version approved by the internal</td>
<td>Kristian Beckers, Stefan Eicker,</td>
</tr>
<tr>
<td></td>
<td>reviewers</td>
<td>Maritta Heisel, Widura Schwittek</td>
</tr>
</tbody>
</table>
# Table of Contents

**LIST OF FIGURES** ................................................................. 9

**LIST OF TABLES** ....................................................................... 11

1 **INTRODUCTION** ..................................................................... 13

2 **EXTENDING THE CONCEPTUAL DESIGN OF THE COMMON BODY OF KNOWLEDGE** . 15
   2.1 Introduction to the CBK ......................................................... 15
   2.2 Extensions to the CBK ............................................................ 16

3 **RESEARCH GAP IDENTIFICATION** ............................................ 19
   3.1 Structured literature surveys according to Kitchenham ................ 19
   3.2 Structured Research Area Analysis .......................................... 21
      3.2.1 Extension/Integration of Kitchenham and CBK .................... 21
      3.2.2 Identifying Research Gaps Using the CBK ......................... 23
      3.2.3 Formalizing Research Area Analysis ................................. 24
   3.3 Realization ............................................................................ 28
   3.4 Related Work ....................................................................... 32
   3.5 Summary and Outlook ............................................................. 32

4 **USABILITY TESTING OF THE COMMON BODY OF KNOWLEDGE** .............. 33
   4.1 Usability Testing of Knowledge Discovery ................................. 33
   4.2 Usability Testing of Adding Knowledge Objects ......................... 35
   4.3 Realizing Improvements .......................................................... 41

5 **CONCLUSIONS** ....................................................................... 43

**BIBLIOGRAPHY** ....................................................................... 45

# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BOK</strong></td>
<td>Body Of Knowledge</td>
</tr>
<tr>
<td><strong>CBK</strong></td>
<td>Common Body of Knowledge</td>
</tr>
<tr>
<td><strong>JVRL</strong></td>
<td>Joint Virtual Research Lab</td>
</tr>
<tr>
<td><strong>KA</strong></td>
<td>Knowledge Area</td>
</tr>
<tr>
<td><strong>KO</strong></td>
<td>Knowledge Object</td>
</tr>
<tr>
<td><strong>NESSoS</strong></td>
<td>Network of Excellence on Engineering Secure Future Internet Software Services and Systems</td>
</tr>
<tr>
<td><strong>NoE</strong></td>
<td>Network of Excellence</td>
</tr>
<tr>
<td><strong>RQ</strong></td>
<td>Research Questions</td>
</tr>
<tr>
<td><strong>SDE</strong></td>
<td>Service Development Environment</td>
</tr>
<tr>
<td><strong>SMW</strong></td>
<td>Semantic MediaWiki+</td>
</tr>
<tr>
<td><strong>SMW-QL</strong></td>
<td>Semantic MediaWiki query language</td>
</tr>
<tr>
<td><strong>SLR</strong></td>
<td>Structured Literature Review</td>
</tr>
<tr>
<td><strong>UML</strong></td>
<td>Unified Modeling Language</td>
</tr>
</tbody>
</table>
List of Figures

Figure 2.1: The CBK entries over time ....................................................... 16
Figure 2.2: The evolved CBK ontology ....................................................... 18
Figure 3.1: Types of literature researches and reasons and goals to perform them ............... 19
Figure 3.2: Process proposed by Kitchenham for undertaking a SLR .......................... 21
Figure 3.3: Extension of the Identify Research Gaps use case .................................. 24
Figure 3.4: Mapping study support realized as SMW-QL query ................................. 30
Figure 4.1: Results for the usability test for knowledge gathering use case of the CBK (1/2) .... 34
Figure 4.2: Results for the usability test for knowledge gathering use case of the CBK (2/2) .... 34
Figure 4.3: Results for the usability test for knowledge object entering use case of the CBK (1/2) ... 35
Figure 4.4: Results for the usability test for knowledge object entering use case of the CBK (2/2) ... 36
Figure 4.5: Diagram of usability principles for which improvements are suggested .......... 36
Figure 4.6: Diagram of functional areas for which improvements are suggested ............... 37
List of Tables

Table 4.1: List of usability principles for which improvements are suggested ................. 37
Table 4.2: Functional areas to which improvements were suggested ............................... 37
Table 4.3: Mapping usability suggestions to functional areas (1/3) ................................. 38
Table 4.4: Mapping usability suggestions to functional areas (2/3) ................................. 39
Table 4.5: Mapping usability suggestions to functional areas (3/3) ................................. 40
1 Introduction

In this deliverable, we present the progress being made concerning the common body of knowledge (CBK) and its current implementation in the SMW+ framework.

The Semantic MediaWiki itself is an extension of the MediaWiki platform, which is quite renowned empowering the popular Wikipedia encyclopedia. The Semantic MediaWiki has been funded in part by projects of the EU Framework Programmes (FP6 & FP7), SEKT\(^1\) and ACTIVE\(^2\), and by the project Halo\(^3\). The Halo core extension in turn is an extension to the Semantic MediaWiki featuring better usability options. The whole conglomerate of the MediaWiki, the SMW extension, and the Halo core extension is called SMW+. In the following, when we speak about SMW+ we will refer to the whole package consisting of the MediaWiki, the Semantic MediaWiki extension and the Halo core extension.

The CBK is intended to support one of the main goals of the NESSoS NoE, namely to create a long-lasting research community on engineering secure software services and systems and to bring together researchers and practitioners from security engineering, service computing, and software engineering.

The overall theme of this deliverable is the identification of deficiencies in the initial version of the CBK. This theme concerns two aspects: first, the identification of research gaps should be supported. This means that by querying the CBK it should be possible to obtain hints about possible research gaps. To support such a functionality by the CBK, we carried over Kitchenham’s work on systematic literature reviews [6] to our ontology-based CBK. To support the identification of research gaps, the ontology had to be adjusted. With the adjusted ontology, we can support mapping studies and problem/gap studies. The ontology underlying the CBK helps to acquire knowledge in a structured way. This makes it possible to formalize queries using relational algebra and to implement the queries in SMW+.

Second, it is intended to open the CBK for the general public during the duration of the NESSoS project. For this to be possible, the usability of the CBK implementation must be of a very high quality. In order to assess and improve the usability of the initial version of the CBK, we conducted two usability tests. The first usability test concerned querying the CBK and resulted in a satisfactory level of usability. The second usability test concerned the entry of new knowledge objects to the CBK. This test revealed numerous points for improvement. We discuss how these suggestions can be taken into account.

The deliverable is organized into three main chapters: in Chapter 2, we present the evolution of the conceptual design of the CBK. We extended the ontology of the CBK to support our structured research gap analysis, and we proposed several improvements to increase the structuring possibilities of the CBK. We present our approach for research gap analysis in Chapter 3, and we report on the results of our usability tests in Chapter 4. Finally, we conclude and raise ideas for future work in Chapter 5.

Background/Foreground The entire work regarding the CBK constitutes foreground. We did not work on the conception or the implementation before the start of NESSoS. Thus, we have no background regarding the CBK.

Relations to other work packages We developed a plugin for the NESSoS SDE tool packaging platform that can display the CBK entry of an SDE tool, inside the SDE platform. The partners from WP2 will ask all contributors to the SDE also to contribute a CBK entry. This shall ensure that the CBK can display an entry for each tool in the SDE. This plugin is described in Deliverable 2.3 and is called CBK search.

We describe the technical enhancements of the CBK in Deliverable 1.3, which includes features for business continuity and a new version of the SMW+ platform. The CBK now is equipped with a manual for entering methods, tools, techniques, and notations. We delivered these enhancements as part of the JVRL population plan.

\(^1\)http://www.sekt-project.com 
\(^2\)http://www.active-project.eu 
\(^3\)http://www.projecthalo.com
2 Extending the Conceptual Design of the Common Body of Knowledge

We provide an introduction to the CBK in Sect. 2.1 and the extensions we developed since the last deliverable in Sect. 2.2.

2.1 Introduction to the CBK

Ontologies are used to capture knowledge about some domain of interest. Our domain is the field of engineering secure software and services. An ontology provides a catalogue of the classes of objects that exist in the domain. Moreover, an ontology consists of relations between these classes, and of the objects contained in the classes. In an earlier work [26] we presented the Ontology for the NESSoS CBK. We present the ontology we use in this paper in Fig. 2.2 as a UML class diagram1, this ontology presents the subset of the CBK, which is relevant for this work. The classes in light grey represent the most relevant classes in our ontology, and the classes in dark grey are classes that inherit from the most relevant classes.

The class KnowledgeArea divides the field of secure software and services into knowledge areas (KA). The central class in our ontology is the class KnowledgeObject, which represents all types of knowledge objects (KO) we want to capture. As examples, we consider the KOs of the types Tool, Method, and Notation. The equally named classes inherit general properties from the class KnowledgeObject. In general, the properties that are inside of a class box are simple properties, e.g., of type String or Boolean, while there also exist structured properties connected to class boxes via associations. Simple properties are, for instance, contextDescription, problemDescription, and solutionDescription, which represent textual descriptions of the context, the tackled problem, and the solution for tools, methods, and notations. These properties are part of the class KnowledgeObject. An example for a structured property is the association publications, which connects the class KnowledgeObject and the class Publication. This property is structured, because every publication consists of a BibTeX entry or links to DBLP2 (bibtexEntriesOrLinksToDBLP), and a flag indicating the importance of a publication (isPrimaryLiterature).

The class CommonTerm has several defined terms, defined in previous work [9], these can be related to terms of KOs. Moreover, some structured properties refer to enumeration types labeled with the UML «enumeration» stereotype, e.g., the association maturityLevel that connects the class KnowledgeObject and the class MaturityLevel. This enumeration type allows us to rate every tool, method, and notation according to its maturity.3

Multiplicities at the association ends specify constraints on the number of elements contained in an association end. For instance, the 1 at the association end of the association maturityLevel describes that each KO has exactly one maturity level. KOs have several relations between each other, e.g., that one KO Uses another. However, these were extended for the contribution of this work and, hence, we explain them in Sect. 2.2.

Specific types of KOs are Notations, which can be supported by Tools (isSupportedBy). A Notation supports a grade of formality (GradeOfFormality). A Tool has a set of functions, and a tool itself can be used by other tools (usedBy) and can support methods (supportsMethods). The specification of technical requirements (technical-Requirements) and an installation guide (installationGuide) are useful for practitioners. A tool can be licenced (licence). The output type of a tool subsumes several outputs a tool provides. The concept of a Function is similar to functions of regular programming languages. Hence, each Function has signature, a number of parameters, a returnType, and a description. Methods can be divided into Activities. They may use specific specification styles (SpecificationStyles), which are pre-defined by the enumeration class SpecificationStyle. Furthermore, the property BasedOnNotations inherited from the class KnowledgeObject is constrained and cannot be empty, as all methods use a notation, sometimes one specifically defined for the method. Activities (Activity) help to structure Methods and to describe workflows based on Inputs, Outputs, and a Description. This means that an activity can use the output of another activity as input. Techniques have just one action, which makes them less complex than Methods. A Notation has at least one grade of formality (gradesOfFormality) according to the enumeration class GradeOfFormality.

Using the presented ontology structure, we can adequately capture and process knowledge in the field of engineering secure software and services. In addition, we evolved the ontology for this work with the elements Term, Note-1

2. http://www.informatik.uni-trier.de/~ley/db/
3. The UML stereotype «enumeration» is used for classes that have a fixed set of attributes, which are referred to by other classes. This use differs from the specification in the UML standard.

In general, enumeration types allow us to pre-define values a property might have.
2.2 Extensions to the CBK

We prepared the NESSoS CBK ontology (see Sect. 2.1) in order to support the concepts of the structured literature reviews from Kitchenham explained in Sects. 3.1 and 3.2.1. Hence, we extended the CBK ontology (see Fig. 2.2) with the classes SubKnowledgeArea, Publication, and Study. SubKnowledgeAreas refer to KnowledgeAreas. This allows a more precise specification parts of KnowledgeAreas. Kitchenham considers a specific kind of Publication that relates to one or more KnowledgeObjects, a so-called Study. This considers a specific problem KnowledgeObjects address, borders of the study, and open questions. In addition, the type of the study can be one of the types mentioned in Sect. 3.1.

We consider keywords and tags in knowledge objects. Keywords are given according to the guidelines of a specific system, e.g., the ACM-keyword-system [29]. Tags are chosen without any restriction and provide the possibility to choose any possible word.

We want to use the CBK ontology for finding research gaps. This requires KnowledgeObjects that do not have a solution description yet, but only problem descriptions. We also included the MaturityLevel None for these KnowledgeObjects. In addition, KnowledgeObjects now have a FutureWork attribute that states research not yet carried out. The difference between these two is that KnowledgeObjects that only have a problem description represent a research area that needs a significant amount of research for providing a solution. The FutureWork attribute in KnowledgeObjects represents possible research that can build upon an existing solution. Researchers that use our approach might look for one or the other.

KnowledgeObjects have relations between each other. These relations are relevant for the investigation of research areas. KOs can be based on other KOs (IsBasedOn), and they can use each other (Uses). In addition, KOs can be used in combination. In this case, one KO Complements another. If KOs can be exchanged, these have a Substitutes relation.

We include the class Term that holds the terms a specific KnowledgeObject uses. The class CommonTerm
contains a set of well-defined terms. A Term of a KnowledgeObject is either broader, synonymous, or narrower than a CommonTerm. This allows a comparison of KnowledgeObjects using the CommonTerms. Without these any comparison would lack precision, because terms and notions differ in KnowledgeObjects.

We also provide uses and isBasedOn relations on KOs. In the previous version of the CBK the relations IsBasedOn and Uses were only defined between specific kinds of KOs. Standard is a new kind of KO that represents a conceptual framework for security and risk management standards (see Fig. 2.2). Standard contains several concepts that result in StandardActivities. StandardActivities are a specific kind of Activities. We want to investigate possible relations between standards and other KOs, which can support the implementation of a standard, in the future. The Standard KO is based on the work by Sunyaev [28] and Stoneburner [27]. Sunyaev developed a conceptual model for security standards and Stoneburner a model for risk management standards. We want to add the Common Criteria [14] and the ISO 27001 [13] standards to the CBK in the future.

Our derived framework for security and risk management standards consists of the following StandardActivities. These are activities that result in the required documentation for certification.

**Environment Description** The Environment Description states the scope of the standard. Hence, the environment in which the security system shall be integrated into, e.g., an organization or an Information and Communication Technology (ICT)-based System or combinations of both.

**Stakeholder Description** The Stakeholder Description describes all relevant persons that have a relation to the environment.

**Asset Identification** The Asset Identification for these stakeholders collects all information or resources that have a value for the stakeholders. The assets shall be protected from harm caused by the environment.

**Risk Level Description** For each asset, a Risk Level Description states the impact the loss of an asset has on a stakeholder. Hence, the risk level description classifies the assets into categories according to their significance for the environment.

**Vulnerability and Threat Analysis** The Threat Analysis detects vulnerabilities of the assets. In addition, threats have to be identified that exploit the vulnerabilities.

**Security Property Description** The Security Property Description describes the security properties of the assets. These are security goals, requirements, etc., which describe the security needs of the assets. These descriptions are based upon the results of a Vulnerability and Threat Analysis.

**Risk Determination** The Risk Determination also uses the results of the Vulnerability and Threat Analysis in order to determine the likelihood of an attack on the assets, refining the risk level descriptions into risk values.

**Control Assessment** The Control Assessment evaluates existing security controls and their ability to reduce the risk of an attack towards assets.

**Security Measures** The Security Measures specify a list of new or refined security controls that are required to improve the protection of the assets.

**Security Assessment** The Security Assessment evaluates if the security measures satisfy the security properties of the assets with regard to the Vulnerability and Threat Analysis. This can result in an update of the security measures.

**Risk Acceptance** The Risk Acceptance evaluates if the Security Measures reduce the risk of attacks on assets to acceptable levels.

**Security and Risk Documentation** The security system description finishes with the Security and Risk Documentation.
Figure 2.2: The evolved CBK ontology
3 Research Gap Identification

Getting an overview of existing engineering methods, tools, techniques, standards, and notations for specific fields is of major importance for software engineering researchers. This knowledge is the basis for finding research gaps and problems in this field, which require their attention. Our objective is to develop a technique for finding missing methods, notations and tools in specific knowledge areas.

Researchers usually have to rely on their experience during a research area analysis, which includes the activities of finding research gaps and identifying research areas. This can lead to a biased outcome of a research area analysis. Hence, research gaps or immature research areas might be overlooked repeatedly. In addition, researchers have to find relations between publications, which are sometimes implicit.

In order to ameliorate this situation, we propose a structured approach for research area analysis. This approach utilizes the extensive research of Kitchenham et al. [15, 16, 18, 17, 19, 6, 7, 23] for structured literature reviews. We apply Kitchenham’s methods to the CBK.

Our approach is threefold: We carry over Kitchenham’s research for structured literature reviews to informal queries for the CBK, and we also extend the CBK to support these informal queries. In the next step, we refine these informal queries into formal CBK relations using Codd’s relational algebra [8]. For this purpose, we apply the DOOR method by Allocca et al. [2] for capturing the semantics of relations in ontologies and to formally specify these relations. The technical realization of the CBK is a Semantic MediaWiki+ platform, and we implemented the relational algebra expressions as CBK queries.1

The queries result in tables that show the relations between KAs and KOs. The tables also contain the information of how many KOs are in a KA and what kind of KOs exist in it, e.g., methods, tools, techniques, and notations. These compact results of a query are more effective than analyzing the natural language in publications. Moreover, the creation and execution of a query in the CBK is less time consuming than finding relevant literature for a KA and analyzing it.

This chapter is organized as follows: we explain Kitchenhams’ work about structured literature surveys in Sect. 3.1. We present in Sect. 3.2 a structured research area analysis method, for research gap analysis and immature research area identification. We show in Sect. 3.3 our realization of the method with SMW+ and the CBK. Section 3.4 presents related work. Finally, we summarize and raise ideas for future work in Sect. 5.

3.1 Structured literature surveys according to Kitchenham

As already mentioned, one possible usage of the CBK is to get an overview about one knowledge area or several knowledge areas. This use case and motivation is not limited to the CBK. To gain a structured overview of the state of the art and existing literature before starting new research is one fundamental element of scientific work. For the area of software engineering, Kitchenham was one of the first, who described a structured literature review

1http://www.nessos-project.eu/cbk
process [15]. Over the years this initial process was extended and improved by Kitchenham herself and others [15, 16, 18, 17, 19, 6, 7, 23]. Today it is the best practice approach and standard for undertaking literature reviews in the field of software engineering.

There are several reasons and goals why researchers might want to perform a literature review. And there are also different types of literature reviews, which can serve the different goals. Fig. 3.1 shows a condensed view of findings and statements from different publications in the field of systematic literature reviews in software engineering [15, 16, 18, 17, 19, 6, 7, 23]. The overall reason to do any kind of literature research is Information Gain. This top-level goal can be refined into the goal to get a mere Overview [15]. This overview is not obtained for any particular reasons and the requirements for a literature review regarding completeness and quality are quite unspecific. In contrast, the goals of Aggregation of Results and Search for Research Areas have a well-founded motivation. Moreover, they lead to clear requirements for the quality, completeness and structure of a literature review.

When aggregating results, one might want to Refine New Findings based on the aggregated data [15, 16]. This goal is often the motivation when empirical studies are aggregated. Aggregating small datasets to a big one can lead to new insights. It is also possible that the findings and data of other publications are used to Strengthen Own Results, for example when the own dataset obtained is too small to allow statistically significant conclusions [15, 16]. A last reason for aggregation is to give a Background / Positioning of Own Work [17, 19, 7, 23] and obtain a comprehensive background and rationale for the own research and to connect the results to the related work.

When searching for research areas, a Detection of Problems & Boundaries [15, 16, 19, 6] of a certain method or set of methods can be the goal. The result then allows researchers to define where research is needed to improve (an) existing method(s). Another option is to search for Assumptions or Unvalidated claims [15, 16, 19, 6]. (In)Validating such assumptions and claims helps to improve the understanding of the research field for which they were stated. These two sub-goals aim at finding immature research areas and improve them with further research. In contrast, finding Open Questions and Gaps aims at research fields, where no publications about solutions exist [16, 19, 23]. Such an area is a research gap, and often only an incomplete problem description for this gap exists. For example, several publications state the same question, which they did not tackle in their work. In addition the question is not directly related to this work, but it is a problem for future research.

All types of Literature Reviews support the goal of obtaining an overview. The quality of the overview differs in how structured and planned the literature review was performed. One type is the Structured Literature Review SLR [15, 16, 18, 6]. A SLR is a comprehensive literature review considering a specific research question. The SLR is obtained in a planned and structured way. Moreover the SLR report is structured itself in a certain way. Kitchenham’s method to perform a SLR was developed to find empirical primary studies considering a specific question and to aggregate the data in the first place. This review type supports all kinds of aggregation goals. Additionally, it turned out later that SLRs also make it possible for researchers to find immature research areas [19, 6]. For this kind of goal, Kitchenham’s method has to be adapted slightly as the data to be extracted is no longer the empirical data itself but the findings based on this data. One kind of the SLR is the Tertiary Study [16, 17]. Such a study aggregates the results of other SLRs, hence it relies on secondary studies. Another type of literature review is the Mapping Study [16, 19, 7, 23]. Here it is not the aim to extract any data, but to map studies to research fields or problems. In this way a good background can be established. This kind of study can be the first step to define the boundaries of a SLR. A Problem / Gap Study is a refinement of a mapping study. Hence, a mapping study in which also problems and gaps are discussed results also in a a Problem / Gap Study [16, 23, 7]. A Problem / Gap Study serves to find gaps or to find immature methods. For assessing immaturity, a gap study should be combined with a SLR.

Kitchenham et al. also propose a process to conduct SLRs [15, 16, 18, 6]. It is shown in Fig. 3.2. This process can be also used for mapping studies with some slight adaptions [16, 19, 23]. The process is split up in three major phases. First the Planning phase takes place, followed by the Conducting Review phase, and finally the Reporting Review phase ends the review process.

The Planning starts with an Identification of need for review step, in which it is asked what are the needs to be covered by an SLR and if they substantiate conducting a SLR. The next step is optional as in Commissioning of Review the SLR is tendered to other research groups. This only happens if the group interested in the results is not willing to conduct the review themselves. The first real step towards a SLR is to Formulate RQ (Research Questions). The research questions are the core of an SLR. All later decisions and results are checked against the RQ later on. In Develop Protocol the review itself is planned. The central aim of this step is to establish a common understanding of tasks and related documentation between the conducting researchers. The initial protocol is refined later on and also serves as a documentation of all steps executed. The step Evaluate Protocol is performed to detect misunderstandings, ambiguities, and insufficient definitions. An evaluation can be a test run on a small set of documents or just a recheck by SLR experts. To ease the planning there are predefined Satisfaction Checklists / Questionaries, Question Types,
Question Structure Templates, and Protocol Templates. For Question Structure Templates, Kitchenham and Chaters proposes to use the PICOC criteria framework to structure research questions [16]. PICOC stands for the criteria population, e.g., application area or specific groups of people, intervention, e.g., the method which is of interest, comparison, e.g., the benchmark, outcomes, what is the improvement to be shown, and context, a description of the setting in which the comparison takes place. All these documents serve as an input and guide for certain planning steps. The result of the planning phase are the Research Questions and the Review Protocol. They serve as input to the Conducting Review phase.

The review starts with the Identification of Research, which results in a set of studies which might be relevant. According to defined inclusion and exclusion criteria the step Selection of Primary Studies is performed. The selected studies are then rated in the step Assessment of Study Quality. For those studies with a satisfying quality level the data contained in the studies is extracted in the step Data Extraction and Monitoring. Afterwards the Data Synthesis is performed. The input to this phase are a List of Sources to Search In, Quality Checklist Templates, Extraction Form Templates, and Synthesis Types. Outputs produced in the conducting review phase are the Selected And Rated Studies, the Data Sets extracted form these studies, and the Results and Findings of the data synthesis.

All previously generated outputs serve as an input for the last phase Reporting Review. As external input, Report Templates are given. Based on the inputs, the step Specifying Dissemination Mechanism is executed, where one decides how to spread the result and in which form. Then the report is actually written in the step Write and Format Report. As last activity a Evaluate Report step is performed. A evaluation can be a discussion with experts or a peer review when submitting the report to a journal or conference. The Report is the output of the entired SLR process.

3.2 Structured Research Area Analysis

We present in Sect. 3.2.1 how our work integrates into the approach of Kitchenham described in Sect. 3.1. We have shown the preparation of our ontology in order to support research area analysis in Sect. 2.2 in the previous chapter. We describe how to support research area analysis in Sect. 3.2.2. In Sect. 3.2.3, we specify the relations between the different ontology parts in detail. These relations enable us to substantiate the envisaged research area analysis by the semantics of the knowledge stored in the ontology.

3.2.1 Extension/Integration of Kitchenham and CBK

There are several points of integration for the CBK and the literature review process introduced in Sec. 3.1. These points are the inputs Question Types, Question Structure Templates, and the List of Sources to Search In, and the
process steps Formulate RQ, Identification of Research, and Selection of Primary Studies (see Fig. 3.2). To improve the integration for some of these points, the CBK and / or the literature review process have to be adapted.

**Question Types** The original question types defined by Kitchenham at al are formulated for SLRs [15, 16]. For mapping studies, a selection of questions and their generalization can be found in the works of Kitcheham et al [19], and Petersen et al. [23]. These insights combined with the structure of the CBK (see Fig. 2.2) result in some new question types:

- How many different KOs exist for the KA in question?
- Which KA(s) are covered by a certain KO?
- Which are the problems and future work mentioned for (a) given KA(s)?
- What is the maturity of KOs for (a) given KA(s)?
- What are the main publications for a given KO or KA?

*Question types help to formulate research questions. They give evidence which questions can be of interest and how to formulate them. Moreover, whenever a research question within an actual review maps to one of the questions types given above, this question can be answered by the CBK directly.*

**Question Structure Templates** For the question structure template, Kitchenham and Chaters propose to use the PICOC criteria framework to structure research questions [16] as we already described in Sect. 3.1. But when investigating the CBK meta-model (see Fig. 2.2) it seems to be reasonable to add some criteria. Some of these criteria are an implicit part of the original PICOC criteria, but to have them as explicit criteria ensures that they are specified.

The main addition is to define the knowledge area(s) explicitly, unlike having them implicit in the context. In most cases of conducting a SLR or mapping study, there is a very specific focus on a special part of software engineering. An overview given by Kitchenham et al. shows that evidence [17]. This focus should be captured within the criteria, because some electronic sources support to select knowledge areas [6, 16]. Moreover, for mapping studies these knowledge areas are often used for structuring the report [19, 7, 23]. In the case of the CBK the knowledge areas are one of the core concepts, and searching the CBK utilizes the knowledge areas.

A minor addition is to distinguish between general terms of the population and special terms of the knowledge area(s). The special terms have a great weight when searching and can help to structure the results [6, 19].

*Using this new question structure makes important parts of the questions more explicit. And they ease the use of the CBK, because the separation between general search terms, common terms and knowledge areas are directly reflected in the formulation of the search queries (We will see in Sect. 3.2.3).*

**List of Sources to Search In** The CBK and its searching capabilities have to be added to the list of sources to search in [16, 6]. Adding the CBK itself is trivial, but for the capabilities it has also to be checked which new capabilities the CBK introduces. For example, the missing relations between KOs and between publications, which has been identified as an issue for all existing search sources [6], is explicitly addressed in the CBK.

*Having the CBK in the source list with an explanation of its capabilities helps to plan the research and clarifies in which cases the CBK is of superior use in comparison to other sources.*

**Formulate RQ** Besides the two inputs, the Question Types and the Question Structure Templates, the CBK can directly support the formulation of research questions. The CBK already defines an ontology of knowledge areas. With an extension, it supports sub-knowledge areas (see Sec. 2.2). Those areas can help to focus the research questions, because it gives an orientation how to refine knowledge areas and how knowledge areas are related. To find the focus for the own research is considered challenging without such a support [18, 17, 7]. Additionally the adapted CBK presented in Sec. 2.2 provides the common terms used in these knowledge areas. These terms help to sharpen the questions and to avoid ambiguities. Thus, the basic activities “select question types”, “generate questions”, “structure questions”, as proposed by Kitchenham et al. [16] for the process step Formulate RQ, can be complemented by “obtain and select knowledge areas” and “identify and select common terms”.

*The integration of the CBK and Formulate RQ improves the outcome of the whole literature review process, because it helps to avoid the formulation of imprecise search questions with respect to missing / wrong focus and wrong / ambiguous wording.*
Identification of Research  The ontology of common terms and related synonyms contained in the CBK also helps to formulate the search queries, not only for the CBK, but also for other search sources.

This formulation of search terms is a crucial step and the knowledge about relations between terms and the existence of a synonym list improves this step of finding research a lot [15, 16, 18, 17, 19, 6, 7, 23].

Selection of Primary Studies  For the selection of primary studies the CBK provides information for some comprehensive and sophisticated selection criteria. The CBK already contains information, which is very specific and useful to rate KOs, like the maturity level. And this information is available for all results obtained from the CBK. These criteria are hard to evaluate for results from other search sources [16, 18, 6]. Examples for inclusion criteria:

- Only include KOs and related publications with a certain maturity level
- Only include KOs and related publications with a certain style type
- Only include KOs, which are a core concept of a KA and therefore many other KOs are based on these KOs
- Only include KOs and related publications which span certain KAs
- Only include publications, which are considered as most significant by the editors of KOs

Having a set of precise selection criteria, which can be evaluated for all results, improves the outcome of the whole review process. The probability of excluding relevant studies and the bias caused by including literature of low quality can be reduced. And the whole review process speeds up when using the CBK, because the information for evaluating the criteria is available explicitly.

3.2.2  Identifying Research Gaps Using the CBK

According to the CBK use cases [26] we concentrate on the use case Identify Research Gaps in this paper. We therefore further specified this use case resulting in four specializations of the aforementioned use case (see Fig. 3.1). The roles Administrator and Quality Agent are connected to all use cases. In order to improve the readability of the diagram we left out these connections.

While a Mapping Study and a Problem/Gap Study are supported best by the CBK, it at least partially supports a Structured Literature Review and a Tertiary Study (see Fig. 3.1). In the following, the CBK support for all different kinds of studies, as defined by Kitchenham, is described in more detail. The support, as described in this section, sketches only a first idea, which will be refined in Sect. 3.2.3, where we only consider the well supported Mapping Study and Problem/Gap Study.

It is assumed that all relevant literature of a field is already part of the CBK. Ideally, previously identified open research questions and gaps which are part of the future work or outlook sections of scientific literature are also codified as KOs, containing only a problem description but missing a solution description.

Mapping Study  For a Mapping Study researchers have to specify one or more KAs, one or more sub-KAs or one or more common terms. Additionally, they can constrain the search by providing further search terms. The CBK returns KOs grouped by KAs. For example, depending on the scope of the search, 20 results for one KA can be interpreted as a sign for maturity of a field or immaturity of a field. The number of 20 results for the KA Requirements has a different meaning than 20 results for the sub-KA Goal-oriented Requirements including the search terms Cloud and Law. Different meaning refers to that the value the results have for a researcher when trying to find a research gap. The number of 20 results for a KA might have less value for finding a research gap, because a KA is a broad set that includes numerous KAs. Hence, the number of 20 KAs can mean that there is plenty of research gaps to be discovered. While 20 results for a sub-KA and several search terms might mean to a researcher that the area is well covered with KOs. Hence, a search for a gap in this particular corner might not be worth the researchers time.

Problem/Gap Study  Conducting a Problem/Gap Study is also well supported by the CBK. The researcher specifies one or more KAs, one or more sub-KAs or one or more common terms and provides search terms. In this case, the CBK not only groups KOs along the specified KAs, sub-KAs or common terms as described for the Mapping Study, but extends the search to the following classes and fields of the CBK ontology: ProblemDescription and FutureWork of the class KnowledgeObject, Problem and Border in the class Study. These results
support the creation of a Problem/Gap Study because all relevant information is presented to the researcher in a structured way.

**Structured Literature Review** While the researcher is able to retrieve KOs for the selected KAs, sub-KAs or common terms, and all relevant literature references, can only be regarded as a starting point to conduct a full-fledged Structured Literature Review. A SLR involves an in-depth analysis of the actual literature, which is out of the CBK’s scope. If a CBK extension for a SLR should be designed, the ontology would have to be extended to support all kinds of knowledge a researcher can extract from it. We presume that this would mean a significant increase in the size of the ontology. Moreover, providing an ontology extension for all possible findings of SLR results might not even possible, because researchers vary in the kinds of results they gather from a SLR.

**Tertiary Study** Conducting a Tertiary Study is only supported in theory, because it requires all relevant secondary studies to be part of the CBK. If this is not the case, there is no support for this kind of study using the CBK.

In the following, we only consider the well supported Mapping Study and Problem/Gap Study.

### 3.2.3 Formalizing Research Area Analysis

We now identify and specify relevant relations for the identification of research gaps, making use of the ontology and the knowledge it stores and structures. For the analysis, the relations between different tools, methods, and notations, i.e., different KO types, and KAs are of particular relevance.

Allocca et al. [2] present the DOOR method to capture the semantics of relations between different ontologies and to formally specify these relations. While we partly adopt the DOOR steps to support our approach to identify
and specify relations between different KOs, we abstain from building an ontology of these relations. We use the ontology structure presented in Sects. 2.1 and 2.2 on the one hand for typing the relations and, more importantly, on the other hand to refine the semantics of the relations. We divide our approach into the following three steps:

1. Identify and specify top-level relations
2. Identify and specify variants and sub-relations, and characterize their algebraic properties
3. Compose relations

We will use the following abbreviations in the formalization: KA = Knowledge Area, SKA = SubKnowledgeArea, STY = Style, KO = Knowledge Object, TE = Technique, ME = Method, NO = Notations, TO = Tool, P = Publication, ST = Study, CT = CommonTerm, T = Term, and ML = MaturityLevel.

Top-Level Relations: We propose to use the work of Kitchenham (see Sect. 3.1) as the basis for our work, which we already based use cases in Sect. 3.2.2 on this work. The following relations are abstracted top-level relations that support the kinds of queries sketched in Sect. 3.2.2.

- MappingStudy KA x KO Describes a mapping study as a relation between knowledge areas and knowledge objects.
- MappingStudy SKA x KO Describes a more fine granular mapping study as a relation between sub-knowledge areas and knowledge objects.
- MappingStudy CT x KO Describes a mapping study as a relation between common terms and knowledge objects.
- ProblemGapStudy KA x KO x ST Describes a problem or gap study as a relation between knowledge areas, knowledge objects, and studies.
- ProblemGapStudy SKA x KO x ST Describes a more fine granular problem or gap study as a relation between sub-knowledge areas, knowledge objects, and studies.
- ProblemGapStudy CT x KO x ST Describes a problem or gap study as a relation between common terms and knowledge objects and studies.
- StructuredLiteratureReview KA x P Describes a structured literature as a relation between knowledge areas, knowledge objects, and studies.
- StructuredLiteratureReview SKA x P Describes a more fine granular structured literature as a relation between sub-knowledge areas, knowledge objects, and studies.
- StructuredLiteratureReview CT x P Describes a more fine granular structured literature as a relation between common terms or gap study, knowledge objects, and studies.

Variants, sub-relations, and Algebraic Properties: Variants and sub-relations shed light on various facets of the top-level relations with regard to the structured design of the ontology. We express the relations using relational algebra based upon the work of Codd [8]. We use an extention of the relational algebra [20, 1] that offers aggregation and grouping functionalities. The symbol \( \xi \) groups the output according to specified attributes(s). \( \pi \) projects only specified columns of a table. \( \sigma \) selects rows in a table for which specified boolean expression(s) hold. \( \bowtie \) joins tables according to common attributes. All rows that do not have these attributes are left out. \( \bowtie \bowtie \) joins tables, but also displays rows of the left table that do not have all the common attributes. We use relational algebra, because the algebra expressions can be translated to SMW+ queries in a straightforward way, see Sect. 3.3.

For the specifications of the relations, we assume that the structural design of the ontology presented in Sect. 2.1 and 2.2 is given as tables. Classes that have 1..* cardinalities on both ends of the relation in our ontology require connection tables. Otherwise we would require multiple relations between tables, which is to be avoided during database design. For example, we want to create a table for knowledge objects. One row in the table is allowed to have multiple relations to rows in the knowledge area table. Instead, we would have to create numerous columns for these relations in the knowledge area, because we do not know how many relations we need. Hence, we create a further table for these relations. We denote these tables, which we add in the formalization, with “Connect” and append the names of the classes this table connects. For example, the connection table for the tables KnowledgeObject and KnowledgeArea is stated as: ConnectKAtoKO. These connection tables have two columns, which contain the primary keys of each of the tables they connect.
Inheritance in the ontology is translated into one main table for the superclass and one table for each class that inherits from this class. These classes have a relation to the table that represents the superclass and have only the additional attributes of the inherited class. For example, the superclass KnowledgeObject has the class Tool that inherits from it, and one of the additional attributes is Input. Hence, we create a table KnowledgeObject and a table Tool, which has the attribute Input.

**Searchterm:** We define an algebraic expression ST, which represents a boolean expression for one or more searchterms.

\[
ST ::= ST \odot ST | \kappa = \text{String} | (ST) | \neg ST \\
\odot ::= \land | \lor \\
\kappa ::= \text{Tag} | \text{Keywords} | \text{ExecutiveSummary} | \text{Name}
\]

For example, the expression “Tags=’cloud’ \lor Tags=’law’” can be used with \(\sigma\) for the table KnowledgeObject. This results in a table with all KOs that have ‘cloud’ or ‘law’ as tags.

**MaturityLevel:** We define MLB a boolean expression for the selection of one or more maturity level.

\[
MLB ::= MLB \lor MLB | \text{MaturityLevel} = \text{MLS} \\
MLS ::= \text{None} | \text{ProofOfConceptOrPrototype} | \text{StableOrEvaluatedHasUsedInSomeOrganizations} | \text{WidelyAdoptedInPractice}
\]

**KnowledgeArea:** We define KAB to be a boolean expression for the selection of one or more knowledge areas.

\[
KAB ::= KAB \lor KAB | \text{KnowledgeArea} = \text{KAS} \\
KAS ::= \text{Requirements} | \text{Design} | \text{Implementation} | \text{Maintenance} | \text{ConfigurationManagement} | \text{EngineeringManagement} | \text{EngineeringProcess} | \text{Quality} | \text{Security} | \text{RiskManagement} | \text{Privacy} | \text{Trust}
\]

**Sub-KnowledgeArea:** We define SKAB a boolean expression for the selection of one or more sub knowledge areas.

\[
SKAB ::= SKAB \lor SKAB | \text{Sub-KnowledgeArea} = \text{SKAS} \\
SKAS ::= \text{Goal-oriented Requirements} | \text{Problem-oriented Requirements} | \text{Other}
\]

**CommonTerms:** We define CTB a boolean expression for the selection of one or more common terms.

\[
CTB ::= CTB \lor CTB | \text{CommonTerm} = \text{CTS} \\
CTS ::= \text{Requirement} | \text{System} | \text{Environment} | \text{Asset} | \text{SecurityGoal} | \text{SecurityPolicy} | \text{SecurityRequirements} | \text{Threat} | \text{Attack} | \text{Attacker} | \text{Machine} | \text{Loss} | \text{Risk} | \text{Formality} | \text{Vulnerability} | \text{DomainKnowledge}
\]

**MappingStudy_KAxKO:** We specify queries for mapping studies as database relations. We explain the query in detail starting with the \(\sigma_{ST}\), where we join the tables KA, ConnectKAtoKA, and KO and select rows according to ST. The query filters the resulting table for rows that have the required MLB and KAB. The result is projected onto the columns KnowledgeArea, MaturityLevel, Name, ExecutiveSummary, Tags, and Keywords. The query groups the results according to KnowledgeArea, MaturityLevel, Name, ExecutiveSummary, Tags, and Keywords.

\[
\text{MappingStudy}_{KAxKO} =
\]
sub knowledge areas and maturity level. The relation considers certain search terms, maturity level and sub knowledge areas as input. We specified this relation in the following.

\[
\text{MappingStudy}_{SKAxKO} = \\
\xi_{\text{Sub-KnowledgeArea, MaturityLevel, Name, ExecutiveSummary, Tags, Keywords}} \\
\pi_{\text{Sub-KnowledgeArea, MaturityLevel, Name, ExecutiveSummary, Tags, Keywords}} \\
\sigma_{SKAB}( \\
\sigma_{MLB}( \\
\sigma_{ST}(SKA \bowtie ConnectSKAtoko \bowtie KO)) \\
) \\
) \\
)
\]

\textbf{MappingStudy}_{CTxKO}: To extend the MappingStudy to common terms we define the following relation that returns the name of the knowledge object, the executive summary, the tags and the keywords grouped by common terms and maturity level. The relation considers certain search terms, maturity level and common terms as input. We specified this relation in the following.

\[
\text{MappingStudy}_{CTxKO} = \\
\xi_{\text{CommonTerm, MaturityLevel, Name, ExecutiveSummary, Tags, Keywords}} \\
\pi_{\text{CommonTerm, MaturityLevel, Name, ExecutiveSummary, Tags, Keywords}} \\
\sigma_{CTB}( \\
\sigma_{MLB}( \\
\sigma_{ST}(CT \bowtie T \bowtie KO)) \\
) \\
) \\
)
\]

\textbf{Extending Searchterm}: For a problem gap study we extend the fields of the CBK that can be searched as follows.

\[
\kappa ::= \text{Tag} | \text{Keywords} | \text{ExecutiveSummary} | \text{Name} | \text{FutureWork} | \\
\text{Title} | \text{Problems} | \text{Borders} | \text{OpenQuestions}
\]

\textbf{ProblemGapStudy}_{KAxKOxST}: To perform problem gap studies, we include existing studies in the search relation and enriches the output with problem and future work descriptions. We formalize this relation as an variant of the relation MappingStudy_{KAxKO}. The symbol \( \bowtie \) between KO and (ConnectKOtoto \bowtie P \bowtie ST) causes that also KOs are selected that do not have a publication or study.

\[
\text{ProblemGapStudy}_{KAxKOxST} = \\
\xi_{\text{KnowledgeArea, Name, ExecutiveSummary, FutureWork, Tags, Title, Border, OpenQuestions}} \\
\pi_{\text{KnowledgeArea, Name, ExecutiveSummary, FutureWork, Tags, Title, Border, OpenQuestions}} \\
\sigma_{KAxKOxST} \bowtie KO \bowtie (ConnectKOtoto \bowtie P \bowtie ST))
\]

\textbf{ProblemGapStudy}_{SKAxKOxST}: The second extension of the use case includes existing studies in the search relation and enriches the output with problem and future work descriptions. We formalize this relation as an variant of the relation MappingStudy_{SKAxKO}.

\[
\text{ProblemGapStudy}_{SKAxKOxST} = \\
\xi_{\text{KnowledgeArea, Name, ExecutiveSummary, FutureWork, Tags, Title, Border, OpenQuestions}} \\
\pi_{\text{KnowledgeArea, Name, ExecutiveSummary, FutureWork, Tags, Title, Border, OpenQuestions}} \\
\sigma_{SKAxKOxST} \bowtie KO \bowtie (ConnectKOtoto \bowtie P \bowtie ST)
\]
**ProblemGapStudy\_CTxKOxST**: The second extension of the use case includes existing studies in the search relation and enriches the output with problem and future work descriptions. We formalize this relation as an variant of the relation MappingStudy\_SKAxKO.

\[
\text{ProblemGapStudy\_CTxKOxST} = \\
\xi_{\text{KnowledgeArea}, \text{Name}, \text{ExecutiveSummary}, \text{FutureWork}, \text{Tags}, \text{Title}, \text{Problem}, \text{Borders}, \text{OpenQuestions}}( \\
\pi_{\text{KnowledgeArea}, \text{Name}, \text{ExecutiveSummary}, \text{FutureWork}, \text{Tags}, \text{Title}, \text{Problem}, \text{Borders}, \\
\text{OpenQuestions}( \\
\text{MappingStudy\_CTxKO} \bowtie \bowtie \text{KO} \bowtie (\text{ConnectKOtoP} \bowtie \bowtie \text{P} \bowtie \bowtie \text{ST}) \\
))
\]

**Compose relations**  Complex relations can be composed from simple ones, as shown in the following example.

**MappingStudy\_KAxSKAxKOxCT**: We merge the different mappings for knowledge area, sub-knowledge area, and the common term and define the following relation that returns the name of the knowledge object, maturity level, the executive summary, the tags, and the keywords, grouped by knowledge area, sub-knowledge area and common terms.

\[
\text{MappingStudy\_KAxSKAxKOxCT} = \\
\xi_{\text{KnowledgeArea}, \text{Sub}−\text{KnowledgeArea}, \text{CommonTerm}, \text{MaturityLevel}, \text{Name}, \text{ExecutiveSummary}, \text{Tags}, \\
\text{Keywords}}( \\
\pi_{\text{KnowledgeArea}, \text{Sub}−\text{KnowledgeArea}, \text{CommonTerm}, \text{MaturityLevel}, \text{ExecutiveSummary}, \\
\text{Tags}, \text{Keywords}(\text{MappingStudy\_KAxKO} \bowtie \bowtie \text{MappingStudy\_SKAxKO} \\
\bowtie \bowtie \text{MappingStudy\_CTxKO}))
\]

**ProblemGapStudy\_KAxSKAxKOxCTxST**: Finally we merge the different problem or gap study relations for knowledge area, sub knowledge area, and the common terms and define the following relation that returns the name of the knowledge object, common term, the executive summary, future work, the tags the keywords, title of the study, problem, borders and open questions grouped by knowledge area, sub-knowledge area and common terms. We specified this relation in the following.

\[
\text{ProblemGapStudy\_KAxSKAxKOxCTxST} = \\
\xi_{\text{KnowledgeArea}, \text{Sub}−\text{KnowledgeArea}, \text{CommonTerm}, \text{Name}, \text{ExecutiveSummary}, \text{FutureWork}, \text{Tags}, \\
\text{Keywords}, \text{Title}, \text{Problem}, \text{Borders}, \text{OpenQuestions}}( \\
\pi_{\text{KnowledgeArea}, \text{Sub}−\text{KnowledgeArea}, \text{CommonTerm}, \text{Name}, \text{ExecutiveSummary}, \text{FutureWork}, \\
\text{Tags}, \text{Keywords}, \text{Title}, \text{Problem}, \text{Borders}, \text{OpenQuestions}( \\
\text{ProblemGapStudy\_KAxSKAxKOxCT} \bowtie \bowtie \text{ProblemGapStudy\_SKAxKOxCT} \\
\bowtie \bowtie \text{ProblemGapStudy\_CTxKOxST})
\))
\]

To sum up, we applied the DOOR method for the structured creation of ontology relations to the CBK ontology for implementing the Kitchenham structured research area analysis. First, we defined the top-level relations for mapping studies and problem/gap studies. We formalized these relations, using relational algebra, and we derived further relations from these. In addition, we have shown an example for a composed relation of the previously defined relations. We checked all the relational algebra expressions using the relational tool.\(^\text{2}\) For future semi-automatic use of the relations and in the light of the technical realization (see Sect. 3.3), the composition of relations can be left to the users. For example, the Semantic MediaWiki+ allows its users to easily switch the predicates of the relations on and off to generate a result set as required.

### 3.3 Realization

Our ontology behind the CBK allows us to specify various queries realizing the relations presented in Sect. 3.2.3 using the SMW query language (SMW-QL). SMW-QL was introduced as a comfortable means to query the SMW
\(^\text{2}\)\url{http://galileo.dmi.unict.it/wiki/relational/doku.php}
The SMW+ platform provides an inline syntax to integrate queries into a wiki page and a graphical query builder to support the creation of such queries (see Fig. 3.4). In the following, some of the queries specified previously in relational algebra will be translated into SMW-QL. We start with a simple query referring to the relation \textit{MappingStudy} $\text{KA} \times \text{KO}$, followed by a complex query referring to the relation \textit{ProblemGapStudy} $\text{KA} \times \text{KO} \times \text{ST}$. The query given in Listing 3.1 is read like this: retrieve all KOs that belong to the KA \textit{Security Requirements} and which contain the search term \textit{attacker} and/or \textit{invader} in the executive summary. The search term can be further specified using comparator operators and wildcards. The result is returned as a table. Each row represents one knowledge object, whereas each column represents an attribute specified in the query indicated by the question mark. In our case, the table contains the columns \textit{RefersToKnowledgeArea}, \textit{HasMaturityLevel}, \textit{ExecutiveSummary}, \textit{Tags} and \textit{Keywords}. The table is sorted along the KAs and MaturityLevel. The user is able to customize sorting by clicking on the table’s header. The SMW-QL query given in Listing 3.2 refers to the relation \textit{ProblemGapStudy} $\text{KA} \times \text{KO} \times \text{ST}$, thus supporting a Problem/Gap Study. Therefore, it is necessary to additionally output the attributes \textit{ProblemDescription} and \textit{FutureWork} from the KO class and the attributes \textit{Problem} and \textit{Border} from the Study class.

In contrast to definition of \textit{ProblemGapStudy} $\text{KA} \times \text{KO} \times \text{ST}$ in Sect. 3.2.3, the first SMW-QL query is not reused in this query. While subqueries are in principal possible with SMW-QL, it is recommended to express sub-queries as queries where possible. In this case, it is realized as a flat query, not only because of performance advantages, but also for the sake of simplicity. Instead of having to specify SMW-QL queries, the SMW+ offers another more end-user oriented way to support use case extension 1: the faceted search facility. A combination of several ontology properties serve as criteria or ?facets? for a search activity, e.g., searching KOs of type ?Method? (first facet) that belong to the KA ?Software Requirements? (second facet) and that support the notation ?UML? (second facet, via the ontology property HasMethodNotationFor. This more exploratory approach can be considered an alternative way to support mapping studies (see Fig. 3.5). In Fig. 3.5 we show an example of the results of a mapping study.
Figure 3.4: Mapping study support realized as SMW-QL query
Figure 3.5: Mapping study support realized as faceted search
3.4 Related Work

In the following we present other BOKs and check their ability to be used for research gaps identification and existing tool support for structured literature reviews.

The concept of a codified BOK is not new and can be found in many different disciplines. Compared to our CBK and what we’re aiming at, they all differ in how knowledge is codified, which in turn has an influence on how the BOK can be used. Besides the SWEBOK [5], which represents the state-of-the-art of the software engineering field, the Project Management Body of Knowledge (PMBOK) [24] is another prominent BOK. It covers project management knowledge in general. The PMBOK has influenced many subsequent BOK efforts in the computing disciplines. A more collaborative approach is taken by the two BOK projects Usability BOK\(^3\) and Build Security In\(^4\) by the U.S. Department of Homeland Security, both also fostering user participation to provide content following a bottom-up approach.

Most of the BOKs only exist as books with access possibilities given by the table of contents or a key word index, while others also provide a hypertext system, allowing one to browse content along links, such as the online version of the SWEBOK\(^5\). In contrast to our CBK, all of the BOKs we found cannot be queried elaborately, thus making it impossible to get automated support on identifying research gaps.

Tools for structured literature reviews that are regularly used by the software engineering research community are major search engines and digital libraries such as ACM, CiteSeer, IEEE Xplore, Google Scholar, Science Direct and Web of Science [29]. All of these work similarly by specifying boolean search expressions. While they differ in evaluating search expressions and ranking the results, it can be stated that none of these search engines and digital libraries was created to support structured literature reviews [6], as our dedicated approach does.

3.5 Summary and Outlook

We have formalized the Kitchenham approach for structured literature reviews in relational algebra. Furthermore, we implemented these queries in an SMW+ ontology. Thus, we provide a semi-automatic support for the Kitchenham approach that eases the burden of manual literature reviews.

Our approach offers the following main benefits:

- Systematic execution of mapping and problem/gap studies according to Kitchenham based upon ontologies for specific domains (here: secure software and service engineering)
- A structured approach to analyze a research area
- Improving the outcome of literature studies via structured processing of knowledge using ontologies
- Further analysis of research domains can be executed with little effort

Our approach has the limitation that it cannot detect research gaps that are not part of the content of the CKB. Hence, the quality of the outcome of our work is dependent on the quality and quantity of CBK content. However, the possibility also exists that publications might be overlooked when manually executing a literature review according to Kitchenham. Moreover, research gaps, ideas for future work, etc. that only exist in the heads of researchers also cannot be found by any of these approaches.

The work presented here will be extended to support further, more extensive research questions in the future. Examples are the refinement of new findings or the strengthening of own results. We will also work on further automating our approach. We envision an extension of the approach towards other existing ontologies.

\(^3\)http://www.usabilitybok.org
\(^4\)https://buildsecurityin.us-cert.gov
\(^5\)http://www.computer.org/portal/web/swebok
4 Usability Testing of the Common Body of Knowledge

To manage that third parties contribute to and use the CBK, its usability is of utmost importance. Practical experience and feedback from the NESSoS partners suggested that the usability of the CBK should be improved. To assess the usability of the initial version of the CBK and to identify parts for improvements of the CBK, we conducted usability tests according to the ISO 9241 [10] standard, the Isometrics method [11], and the Thinking Aloud method [22].

In Sect. 4.1 we evaluated the usability of discovering KOs using only the Isometrics method. We evaluated the usability adding KOs to the CBK using the Isometrics method and the Thinking Aloud method in Sect. 4.2. We show suggestions for improvements for the CBK in Sect. 4.2.

4.1 Usability Testing of Knowledge Discovery

The first test covers the Browse KOs use case. We conducted this test with master students participating in a course on the development of secure and safe software at UDE. An initial challenge for participants is that these had to understand the basic principles of the ontologies (e.g. relations between KAs and KOs). They had to describe the basic structure of the CBK. The participants had to learn about this structure of the CBK only by using the system. In addition, participants had to find the knowledge objects Security Adaptation Contract and VeriFast in the CBK. In this test they had to describe two specific attributes of the Security Adaptation Contract. These are the fields context and the problem description. They also had to read the entry of VeriFast and describe it. The tasks should be done in-between fifteen and thirty minutes.

We based our evaluation upon the ISO 9241 [10] standard that covers ergonomics of human-computer interaction. The standard describes general ergonomic principles, which apply to the design of dialogues between humans and information systems, namely:

- Suitability for the task
- Suitability for learning
- Suitability for individualization
- Conformity with user expectations
- Self descriptiveness
- Controllability
- Error tolerance

However, the standard lacks a detailed description of how to assess these general ergonomic principles. Hence, we use the Isometrics method [11], which proposes two questionnaires that are compliant to the ISO 9241. The difference between these questionnaires is that one requires a shorter time to fill out and to evaluate. In addition, the Isometrics method contains a manual for evaluating the results of the questionnaires. The results of these questionnaires can be evaluated according to the ISO 9241 general ergonomic principles. The Isometric method is available free of charge for scientific purposes.\(^1\) We asked the authors for permission to use the method for the evaluation of the CBK during the NESSoS project, and the authors granted permission.

The Isometrics questionnaires contain 76 individual questions, where each question belongs to a specific ergonomic principle. The participants have to rate the quality of specific characteristics of the principles. The rating ranges from 1 to 5, where 5 is the best grade. The participants can also choose not to give a grade and reply that they do not have an opinion on this specific question. The Isometrics method prescribes that questionnaires with more than 20 percent of the answers are that the participant has no opinion have to be excluded from the evaluation. We raised this level to 32 percent, so we had only to eliminate one participant from the evaluation. Hence, the results of the experiment have a threat to validity, because of this increase in allowed no opinion answers. Another threat to the validity of this experiment is that the students might have wanted to impress their teacher. However, the

\(^{1}\)The Isometrics Project: \texttt{http://www.isometrics.uni-osnabrueck.de}
evaluation results are shown in Figs. 4.1 and 4.2. The first figure shows that, except for the principles suitability for individualization, the CBK was evaluated with grades of 4 and above. The overall mean of the results is also above 4. Fig. 4.2 shows additionally that the standard deviation of the participants ranged between 1 and 1.5. We conclude from these results that the usability of the CBK for this use case can be regarded as sufficient.
4.2 Usability Testing of Adding Knowledge Objects

We conducted a second usability evaluation on the CBK for the use case of entering a knowledge object into the CBK. The test persons were a senior researcher and a PhD student. The persons were already familiar with the concept of the CBK. However, the persons were not familiar with the implementation of the CBK.

The experiments were conducted on a large screen and the test persons were each trying to enter a method into the CBK. This first stage of the experiment was conducted using the so-called *Thinking Aloud* method. The method captures participants’ conscious cognitions and emotions that are in relation to their actions during the usability testing. The test person speaks aloud every emotion and cognition during the execution of the test. This method can produce insightful results with only few participants. The examiner documented the spoken words during the usability testing. The use of logging instead of video and sound recordings of the test is called *Discount Usability Testing* [12]. After the experiment a short interview was conducted, and the test persons answered the long version of the Isometrics method [11]. We present the results of the Isometrics questionnaires in Figs. 4.3 and 4.4. The results in Fig. 4.3 are between 2.5 and around 3.5 except for the category suitability for individualization. The controllability and the conformity of user expectations are the highest values with 4 points. The other values are considerably lower. The standard deviation is between less than 1 and 2 points, depicted in Fig. 4.4. The Isometrics method prescribes that questionnaires with more than 20 percent of the answers are that the participant has no opinion have to be excluded from the evaluation. We raised this level to 25 percent, otherwise we would have had to eliminate one participant from the evaluation. The deviation was around 1 for the controllability and the conformance with user expectation principles. However, the error tolerance deviation was significantly higher with 2 points.

**Suggestions for Improvements** The Isometrics questionnaires offer the possibility to write a comment of why a specific rating was given, and the questions also suggest the test persons to state examples for these problems. Furthermore, an importance rating between 1 and 5 was given for each of the answers. The Isometrics manual suggests that only suggestions with a rating of 3 or above should be considered as improvement suggestion. We collected all the suggestions that matched this principles. Furthermore, we collected all the suggestions that were made during the interview and the Thinking Aloud test. The evaluation of the entire usability tests resulted in 85 suggestions for usability improvements for the CBK. Tab. 4.1 and Fig. 4.5 show the principles that have to be improved. The highest numbers of suggestions are in the criteria suitability for the task and self-descriptiveness. The issues in self-descriptiveness concern often manuals and help texts, which can be added with little effort. However,
the issues in the suitability for the task principles concern conceptual problems, and these require a considerably higher effort in addressing. A few issues were also discovered in the controllability, error tolerance, and conformity with user expectations.

We mapped the improvements to several functional areas, shown in Tab. 4.2. The start page area represents the main page of the CBK. The add method area is the entire dialogue add method. The remaining areas refer to specific pages of parts of the add method dialogue. The add method: general is one specific page that is part of the add method dialogue. This area contains e.g. the webpages of KOs. The functional areas are also depicted in Fig. 4.6.

We show the lists of the usability improvements in Tabs. 4.3, 4.4 and 4.5. In these tables we present several
Table 4.1: List of usability principles for which improvements are suggested

<table>
<thead>
<tr>
<th>Usability principle</th>
<th>Number of usability issues found</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suitability for the task</td>
<td>37</td>
</tr>
<tr>
<td>Self-descriptiveness</td>
<td>31</td>
</tr>
<tr>
<td>Controllability</td>
<td>7</td>
</tr>
<tr>
<td>Error tolerance</td>
<td>7</td>
</tr>
<tr>
<td>Conformity with user expectations</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4.2: Functional areas to which improvements were suggested.

<table>
<thead>
<tr>
<th>Functional area</th>
<th>Number of usability issues found</th>
</tr>
</thead>
<tbody>
<tr>
<td>start page</td>
<td>14</td>
</tr>
<tr>
<td>add method</td>
<td>37</td>
</tr>
<tr>
<td>add method: startpage</td>
<td>2</td>
</tr>
<tr>
<td>add method: create method</td>
<td>4</td>
</tr>
<tr>
<td>add method: general</td>
<td>9</td>
</tr>
<tr>
<td>add method: knowledge area</td>
<td>1</td>
</tr>
<tr>
<td>add method: save</td>
<td>7</td>
</tr>
<tr>
<td>add method: help</td>
<td>7</td>
</tr>
<tr>
<td>add method: Image Gallery</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 4.6: Diagram of functional areas for which improvements are suggested

mappings from found usability improvements to functional areas of the CBK and usability principles. We sorted the list according to functional groups. We explain how we plan to address these in the next section. The description column contains citations from the usability test.
Table 4.3: Mapping usability suggestions to functional areas (1/3)

<table>
<thead>
<tr>
<th></th>
<th>Method</th>
<th>Description</th>
<th>Functional Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>start page</td>
<td>Is myDashboard needed?</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>2</td>
<td>start page</td>
<td>Is new page needed?</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>3</td>
<td>start page</td>
<td>Your browser message is irritating</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>4</td>
<td>start page</td>
<td>Explicit support for cbk use cases missing</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>5</td>
<td>start page</td>
<td>Change view is not explained</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>6</td>
<td>start page</td>
<td>Login difficult to find</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>7</td>
<td>start page</td>
<td>Login missing feedback</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>8</td>
<td>start page</td>
<td>login failure message did not explain the cause of the error</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>9</td>
<td>start page</td>
<td>login required warning shall be on the frontpage</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>10</td>
<td>start page</td>
<td>login field very small and hidden</td>
<td>Controllability</td>
</tr>
<tr>
<td>11</td>
<td>start page</td>
<td>login feedback missing</td>
<td>Controllability</td>
</tr>
<tr>
<td>12</td>
<td>start page</td>
<td>Login difficult to locate</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>13</td>
<td>start page</td>
<td>The Domain Dialogue in the Login Field is not explained</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>14</td>
<td>start page</td>
<td>Lots of Text to read after login</td>
<td>suitability for the task</td>
</tr>
<tr>
<td>15</td>
<td>add method: save</td>
<td>The save function won’t work until an email is added</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>16</td>
<td>add method: save</td>
<td>“save” message not clear</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>17</td>
<td>add method: save</td>
<td>“save” functionality did not explain the cause of the error</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>18</td>
<td>add method: save</td>
<td>“save” should always be possible</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>19</td>
<td>add method: save</td>
<td>After pushing save I always end up in the main window</td>
<td>Conformity with user expectations</td>
</tr>
<tr>
<td>20</td>
<td>add method: save</td>
<td>It should be explained that saving is possible at all times.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>21</td>
<td>add method: save</td>
<td>Allow save without webpage or email</td>
<td>suitability for the task</td>
</tr>
<tr>
<td>22</td>
<td>add method: save</td>
<td>After pushing save I always end up in the main window</td>
<td>Conformity with user expectations</td>
</tr>
<tr>
<td>23</td>
<td>add method: help</td>
<td>help text examples partially incomprehensible</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>24</td>
<td>add method: help</td>
<td>Help field shall close automatically</td>
<td>Controllability</td>
</tr>
<tr>
<td>25</td>
<td>add method: help</td>
<td>The help is not everywhere present</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>26</td>
<td>add method: help</td>
<td>More help texts are needed. More structured textfields and predefined values would be helpful.</td>
<td>suitability for the task</td>
</tr>
<tr>
<td>27</td>
<td>add method: help</td>
<td>Some help texts and explanations are missing.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>28</td>
<td>add method: help</td>
<td>Help texts sometimes missing and manual not detailed enough.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>29</td>
<td>add method</td>
<td>Difficult to enter a method</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>30</td>
<td>add method</td>
<td>Too much “garbage” on the screen</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>31</td>
<td>add method</td>
<td>Arrangements of fields makes them difficult to understand</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>32</td>
<td>add method</td>
<td>Help fields incomplete</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>33</td>
<td>add method</td>
<td>Notions not well explained, e.g., role in method</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>34</td>
<td>add method</td>
<td>Structure of methods is insufficient</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>35</td>
<td>add method</td>
<td>Earlier versions of Knowledge Objects shall be kept</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>36</td>
<td>add method</td>
<td>Method detail (activities) shall be added earlier</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>37</td>
<td>add method</td>
<td>add a help text that the field size can be changed</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>38</td>
<td>add method</td>
<td>Notation should offer choices</td>
<td>suitability for the task</td>
</tr>
<tr>
<td>39</td>
<td>add method</td>
<td>I did not understand what method role is</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>40</td>
<td>add method</td>
<td>Publication require a help text. What are related publications?</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>41</td>
<td>add method</td>
<td>Additional information should provide examples or more detailed questions.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>42</td>
<td>add method</td>
<td>It is not possible to easily upload and reference pictures in the work flow.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>43</td>
<td>add method</td>
<td>Fixed wiki style not many modifications possible.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>44</td>
<td>add method</td>
<td>Not all information can be found in the manual.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>45</td>
<td>add method</td>
<td>Due to the usage of a web front-end the browser could be closed without saving.</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>46</td>
<td>add method</td>
<td>There is no undo functionality.</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>47</td>
<td>add method</td>
<td>Some input information was not displayed (Method notations).</td>
<td>Error tolerance</td>
</tr>
<tr>
<td>48</td>
<td>add method</td>
<td>It should be explained that the text fields can be increased in size.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>49</td>
<td>add method</td>
<td>I can add the level of a tool here. That dialogue should be rewritten.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>50</td>
<td>add method</td>
<td>What does Version node mean?</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>50</td>
<td>add method</td>
<td>What does fckLR mean?</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>52</td>
<td>add method</td>
<td>Next step does not work.</td>
<td>Controllability</td>
</tr>
<tr>
<td>53</td>
<td>add method</td>
<td>What exactly do the knowledge areas mean? An explanation is missing.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>54</td>
<td>add method</td>
<td>The about the CBK entries should be linked to the help during the method entry.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>55</td>
<td>add method</td>
<td>What does the common terminology tab mean.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>56</td>
<td>add method</td>
<td>For grade of formality examples are missing. They should be added in the help.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>57</td>
<td>add method</td>
<td>Steps and activities in a method are not well explained in the tool or the manual.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>58</td>
<td>add method</td>
<td>A field should be added that explicitly states that the texts here are copied from a paper.</td>
<td>suitability for the task</td>
</tr>
<tr>
<td>59</td>
<td>add method</td>
<td>Papers shall be referenced in the text from bibtex entries.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>60</td>
<td>add method</td>
<td>I could not add a method notation.</td>
<td>Controllability</td>
</tr>
<tr>
<td>61</td>
<td>add method</td>
<td>A help text should be added for the publication. Should this be publications about the publication.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>62</td>
<td>add method</td>
<td>Relation to other objects shall offer the other objects in the CBK.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>63</td>
<td>add method</td>
<td>Notation should offer choices</td>
<td>suitability for the task</td>
</tr>
<tr>
<td>64</td>
<td>add method</td>
<td>Publication require a help text. What are related publications?</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>65</td>
<td>add method</td>
<td>Relations to other knowledge objects is useful but not implemented well.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>66</td>
<td>add method: startpage</td>
<td>Is the red sentence needed</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>67</td>
<td>add method: startpage</td>
<td>Remove AckLR from page button</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td></td>
<td>Knowledge Area</td>
<td>Image Gallery</td>
<td>Controllability</td>
</tr>
<tr>
<td>---</td>
<td>----------------</td>
<td>---------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>68</td>
<td>add method:</td>
<td>Is the message &quot;this is a minor edit&quot; needed?</td>
<td>Controllability</td>
</tr>
<tr>
<td>69</td>
<td>add method:</td>
<td>A description is missing of how to add a figure</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>70</td>
<td>add method:</td>
<td>What kind of pictures shall I add. An explanation is missing.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>71</td>
<td>add method:</td>
<td>How can I reference graphics I already added? A help text is missing.</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>72</td>
<td>add method:</td>
<td>Licensing field cannot be used after uploading a picture.</td>
<td>Conformity with user expectations</td>
</tr>
<tr>
<td>73</td>
<td>add method:</td>
<td>Remove moralising text</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>74</td>
<td>add method:</td>
<td>Remove postal address of the developer</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>75</td>
<td>add method:</td>
<td>Remove email</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>76</td>
<td>add method:</td>
<td>Add references to paper in every field</td>
<td>Controllability</td>
</tr>
<tr>
<td>77</td>
<td>add method:</td>
<td>do we need the “solutions” field</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>78</td>
<td>add method:</td>
<td>do we need the “consequences” field</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>79</td>
<td>add method:</td>
<td>In the help text of the executive summary it would be nice to name the relevant underlying fields instead to reference them.</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>80</td>
<td>add method:</td>
<td>Dialogue for adding a method difficult to find</td>
<td>Self-descriptiveness</td>
</tr>
<tr>
<td>81</td>
<td>add method:</td>
<td>The summary field should the last that is queried, because I do not know what to summarize</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>82</td>
<td>add method:</td>
<td>Do we need “Facts about method” information on the screen</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>83</td>
<td>add method:</td>
<td>Can we remove “Category form” from the page</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>84</td>
<td>add method:</td>
<td>Do we need “page-state: unprotected”</td>
<td>Suitability for the task</td>
</tr>
<tr>
<td>85</td>
<td>add method:</td>
<td>It is not explained what happens when I enter a method that already exists in create or edit</td>
<td>Self-descriptiveness</td>
</tr>
</tbody>
</table>
4.3 Realizing Improvements

We provide an overview of improvements for the usability of the CBK that we will address. We ordered the received suggestions according to the functional areas used in the previous section.

**start page** We work on providing self-descriptiveness of the CBK. For this purpose, we added a long description of the CBK’s background and goals of the NESSoS project. This long text can be reached via a *About the CBK* entry in the main menu. In addition, we provide a brief description of the CBK’s directly on the main page. We cleaned the start page of several entries that are not needed, e.g., the myDashboard button. Furthermore, we will increase the number of available help texts, e.g., the change view button will get an explanation next to it. The login is required for every action in the CBK, and so far the login button is difficult to find. The user is not told on the start page to login. We will improve this via a message on the start screen and a bigger login button on the front page.

**add method: save** The CBK shall allow to save the changes made on a knowledge object at any point in time. So far, this was not well explained and the save function would not work, if the address and email of a KO were not entered. We decided to change this behavior. In addition, a click on the save button changed the view of the user from the screen *method adding* to the *start page*. However, the change of this behavior of the system would require an in-depth change of the code of the SMW+ platform. Thus, we will not implement this improvement at this time.

**add method: help** We started adding numerous help texts to the CBK before executing the usability tests. However, the tests showed that the texts require further improvements and corrections. In addition, the help texts are not everywhere in the system, and we have to strive for a complete coverage of the CBK.

**add method** We are working on a help entry that will contain detailed manuals providing guidance on how to enter knowledge objects into the CBK. The manuals already exist, which include many screenshots for a better understanding, and are part of the Population Plan document (available in the NESSoS Web page). These were also used during the requirements tests. The manuals have to be revised according to the discovered problems during the usability test, e.g., the explanation of a role in a method has to be revised. Moreover, after the results from the usability test, the usage of the CBK will be different from what it is now. Hence, we have to adapt the manuals accordingly. Another problem of the *add method* dialogue is that the participants of the test found it difficult to enter a method. We aim to improve this by reducing the complexity of the entry, meaning to reduce the number of fields in the dialog. For instance, the email and address field caused problems for the test participants. We will remove these entries and allow the entries of steps of a method earlier in the dialog. Moreover, we also increase the number of help texts in this dialog and reduce the number of buttons and texts on the screen that are not directly useful for the task. In addition, we have addressed some malfunctions of the system. For example, in the current version *method notations* were not displayed on the screen. We also found several improvements that we cannot address. For example, a complaint was that the CBK is a browser-based software and when the browser is closed abruptly all the information is lost. We would have to invest a considerable amount of resources into programming a specific browser extension that can deal with this problem. Hence, we cannot solve this issue.

**add method: start page** The start page of the entry contains several unnecessary texts, which we will remove.

**add method: Knowledge area** The knowledge area field in the *add method* dialog has an entry for “minor edits”. We will discuss if this is useful and possibly remove that field.

**add method: create method** The create method has numerous texts and buttons on the screen that are not relevant for the task. We will remove these in order to simplify the dialog. For example, we had a large “Facts about the method” entry on the page that is not required for this task and will be removed.

**add method: general** The *general* field queries much information that is not directly relevant. We will discuss of how to change the dialog so that the user can enter the most relevant entries first, and we will also reduce the number of required fields for the entry.

**add method: Image Gallery** The image gallery requires a detailed description of how to add a picture to the description of a method. We will provide this description. Furthermore, the *licensing* field of the CBK that is not working properly. This issue will be addressed.
We investigated only the method entry dialog in the usability test. However, we will of course search for the found issues in the other dialogs as well and address them.
5 Conclusions

In this deliverable, we presented the evolution of the Common Body of Knowledge (CBK), a structured approach for research gap identification using the CBK, and the results of our usability tests for the CBK in this deliverable. Our work comprises the following main contributions to improve the quality of the Common Body of Knowledge:

- The ability of KOs allowing users to structure knowledge has been improved. A number of new relations has been introduced that enable complex queries and comparisons on the CBK.

- The CBK introduces a mechanism to conduct Mapping Studies and Problem Gap Studies of KAs and KOs.

- The usability of entering KOs and browsing KOs has been evaluated. Numerous improvements have been suggested.

- The CBK is integrated into the SDE in order to improve collaborative work.

All of the endeavors presented in this deliverable and also our plans for future work have one major goal: The research and practice communities to be addressed and integrated by the NESSoS project should eventually use the CBK. We have to further improve the usability of the CBK and extend its functionality in order to provide incentives for contributors.
Bibliography


A  Published Papers [4] and [25]
Ontology-Based Identification of Research Gaps and Immature Research Areas *

Kristian Beckers\textsuperscript{1}, Stefan Eicker\textsuperscript{1}, Stephan Faßbender\textsuperscript{1}, Maritta Heisel\textsuperscript{1}, Holger Schmidt\textsuperscript{2}, and Widura Schwittek\textsuperscript{1}

\textsuperscript{1} University of Duisburg-Essen, paluno - The Ruhr Institute for Software Technology
\{firstname.lastname\}@paluno.uni-due.de
\textsuperscript{2} ITESYS - Institut für technische Systeme GmbH, Germany
h.schmidt@itesys.de

Abstract. Researchers often have to understand new knowledge areas, and identify research gaps and immature areas in them. They have to understand and link numerous publications to achieve this goal. This is difficult, because natural language has to be analyzed in the publications, and implicit relations between them have to be discovered. We propose to utilize the structuring possibilities of ontologies to make the relations between publications, knowledge objects (e.g., methods, tools, notations), and knowledge areas explicit. Furthermore, we use Kitchenham’s work on structured literature reviews and apply it to the ontology. We formalize relations between objects in the ontology using Codd’s relational algebra to support different kinds of literature research. These formal expressions are implemented as ontology queries. Thus, we implement an immature research area analysis and research gap identification mechanism. The ontology and its relations are implemented based on the Semantic MediaWiki+ platform.

Key words: ontologies, research gaps, knowledge management, facetted search

1 Introduction

Getting an overview of existing engineering methods, tools and notations (referred to as Knowledge Objects – KOs) for specific fields (referred to as Knowledge Areas – KAs) is of major importance for software engineering researchers. This knowledge is the basis for finding research gaps and problems in this field, which require their attention. Our objective is to develop a technique for finding missing methods, notations and tools in specific knowledge areas.

Researchers usually have to rely on their experience during a research area analysis, which includes the activities of finding research gaps and identifying research areas. This can lead to a biased outcome of a research area analysis. Hence, research gaps or immature research areas might be overlooked repeatedly. In addition, researchers have to find relations between publications, which are sometimes implicit.

\* This research was partially supported by the EU project Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS, ICT-2009.1.4 Trustworthy ICT, Grant No. 256980).
In order to ameliorate this situation, we propose a structured approach for research area analysis. This approach utilizes the extensive research of Kitchenham et al. [1–8] for structured literature reviews. We apply Kitchenham’s methods to a special ontology, the Common Body of Knowledge (CBK) of the EU project Network of Excellence (NoE) on Engineering Secure Future Internet Software Services and Systems (NESSoS)\(^3\). One of the major goals of this NoE is the integration of the disciplines of software, service, and security engineering. Hence, the CBK ontology contains information about these areas in numerous KOs that refer to KAs.

Our approach is threefold: We carry over Kitchenham’s research for structured literature reviews to informal queries for the CBK, and we also extend the CBK to support these informal queries. In the next step we refine these informal queries into formal CBK relations using Codd’s relational algebra [9]. For this purpose, we apply the DOOR method by Allocca et al. [10] for capturing the semantics of relations in ontologies and to formally specify these relations. The technical realization of the CBK is a Semantic MediaWiki\(^4\) platform, and we implemented the relational algebra expressions as CBK queries.

The queries result in tables that show the relations between KAs and KOs. The tables also contain the information of how many KOs are in a KA and what kind of KOs exist in it, e.g., methods, tools, techniques, and notations. These compact results of a query are more effective than analyzing the natural language in publications. Moreover, the creation and execution of a query in the CBK is less time consuming than finding relevant literature for a KA and analyzing it.

The paper is organized as follows: we explain background about structured literature surveys and the NESSoS CBK in Sect. 2. We present in Sect. 3 a structured research area analysis method, which contains research gap analysis and immature research area identification. We show in Sect. 4 our realization of the approach for the field of engineering secure software and services using the NESSoS CBK. Section 5 presents related work. Finally, we conclude and raise ideas for future work in Sect. 6.

2 Background

We explain Kitchenham’s structured approach to structured literature reviews in Sect. 2.1 and the basic structure of the NESSoS Common Body of Knowledge in Sect. 2.2.

2.1 Literature Research According to Kitchenham

To gain a structured overview of the state of the art and existing literature before starting new research is one fundamental element of scientific work. For the area of software engineering, Kitchenham was one of the first, who described a structured literature review process [1]. Over the years this initial process was extended and improved by Kitchenham herself and others [1–8].

There are several reasons and goals why researchers might want to perform a literature review. And there are also different types of literature reviews, which can serve

\(^3\) http://www.nessos-project.eu/
\(^4\) http://www.nessos-cbk.org
the different goals. Fig. 1 shows a condensed view of findings and statements from different publications in the field of systematic literature reviews in software engineering [1–8]. The overall reason to do any kind of literature research is Information Gain. This top-level goal can be refined into the goal to get a mere Overview without a specific motivation [1]. In contrast, the goals of Aggregation of Results and Search for Research Areas have a well-founded motivation.

When aggregating results, one might want to Refine New Findings based on the aggregated data [1, 2]. Or the findings and data of other publications are used to Strengthen Own Results [1, 2]. A last reason for aggregation is to give a Background / Positioning of Own Work [4, 5, 7, 8]

When searching for research areas, a Detection of Problems & Boundaries [1, 2, 5, 6] of a certain method or set of methods can be the goal. Another option is to search for Assumptions or Unvalidated claims [1, 2, 5, 6]. These two sub-goals aim at finding immature research areas and improve them with further research. In contrast, finding Open Questions and Gaps aims at research fields, where no publications about solutions exist [2, 5, 8].

All types of Literature Reviews support the goal of obtaining an overview. The quality of the overview differs in how structured and planned the literature review was performed. A special type is the Structured Literature Review (SLR) [1–3, 6]. A SLR is a comprehensive literature review considering a specific research question. Kitchenham’s method to perform a SLR was developed to find empirical primary studies considering a specific question and to aggregate the data in the first place. Additionally it turned out later that SLRs also make it possible for researchers to find immature research areas [5, 6]. A special kind of the SLR is the Tertiary Study [2, 4]. Such a study aggregates the results of other SLRs, hence it relies on secondary studies. Another type of literature review is the Mapping Study [2, 5, 7, 8]. Here it is not the aim to extract any data, but to map studies to research fields or problems. When also the problems and gaps discussed in the studies are obtained while doing the mapping, a Problem / Gap Study as a special kind of mapping study is the result [2, 8, 7]. This kind of study serves to find real gaps or to find immature methods. For assessing immaturity, a gap study should be combined with a SLR.
Kitchenham et al also propose a process to conduct SLRs [1–3, 6]. It is shown in Fig. 2. This process can be also used for mapping studies with some slight adaptions [2, 5, 8]. The process is split up in three major phases. First the Planning phase takes place, followed by the Conducting Review phase, and finally the Reporting Review phase ends the review process.

The Planning starts with an Identification of need for review step. The next step is optional as in Commissioning of Review the SLR is tendered to other research groups. The first real step towards a SLR is to Formulate RQ (Research Questions). The research questions are the core of an SLR. All later decisions and results are checked against the RQ later on. In Develop Protocol the review itself is planned. The step Evaluate Protocol is performed to detect misunderstandings, ambiguities, and insufficient definitions. To ease the planning there are predefined Satisfaction Checklists / Questionaries, Question Types, Question Structure Templates, and Protocol Templates. For Question Structure Templates, Kitchenham and Chaters proposes to use the PICOC criteria framework to structure research questions [2]. PICOC stands for the criteria population, e.g. application area or specific groups of people, intervention, e.g. the method which is of interest, comparison, e.g. the benchmark, outcomes, what is the improvement to be shown, and context, a description of the setting in which the comparison takes place. All these documents serve as an input and guide for certain planning steps. The result of the planning phase are the Research Questions and the Review Protocol. They serve as input to the Conducting Review phase.

The review starts with the Identification of Research, which results in a set of studies which might be relevant. According to defined inclusion and exclusion criteria the step Selection of Primary Studies is performed. The selected studies are then rated in the step Assessment of Study Quality. For those studies with a satisfying quality level the data contained in the studies is extracted in the step Data Extraction and Monitoring. Afterwards the Data Synthesis is performed. The input to this phase are a List of Sources to Search In, Quality Checklist Templates, Extraction Form Templates, and Synthesis Types. Outputs produced in the conducting review phase are the Selected And Rated Studies, the Data Sets extracted from these studies, and the Results and Findings of the data synthesis.
All previously generated outputs serve as an input for the last phase Reporting Review. As external input, Report Templates are given. Based on the inputs, the step Specifying Dissemination Mechanism is executed. Then the report is actually written in the step Write and Format Report. As last activity a Evaluate Report step is performed. The Report is the output of the entire SLR process.

2.2 NESSoS Common Body of Knowledge

Ontologies are used to capture knowledge about some domain of interest. In our case, that domain is the field of engineering secure software and services. An ontology provides a catalogue of the classes of objects that exist in the domain. Moreover, an ontology consists of relations between these classes, and of the objects contained in the classes. We present the ontology we use in this paper in Fig. 3 as a Unified Modeling Language (UML) class diagram.\(^5\), this ontology presents the subset of the CBK, which is relevant for this work. The classes in light grey represent the most relevant classes in our ontology for this work, and the classes in dark grey are classes that inherit from the most relevant classes.

The class KnowledgeArea divides the field of secure software and services into knowledge areas (KA). The central class in our ontology is the class KnowledgeObject, which represents all types of knowledge objects (KO) we want to capture. As examples, we consider the KOs of the types Tool, Method, and Notation. The equally named classes inherit general properties from the class KnowledgeObject. In general, the properties that are inside of a class box are simple properties, e.g., of type String or Boolean, while there also exist structured properties connected to class boxes via associations. Simple properties are, for instance, contextDescription, problemDescription, and solutionDescription, which represent textual descriptions of the context, the tackled problem, and the solution for tools, methods, and notations. These properties are part of the class KnowledgeObject. An example for a structured property is the association publications, which connects the class KnowledgeObject and the class Publication. This property is structured, because every publication consists of a BibTeX entry or links to DBLP\(^6\) (bibtexEntriesOrLinksToDBLP), and a flag indicating the importance of a publication (isPrimaryLiterature).

The class CommonTerm has several defined terms, and these can be related to terms of KOs. Moreover, some structured properties refer to enumeration types labeled with the UML ≪enumeration≫ stereotype, e.g., the association maturityLevel that connects the class KnowledgeObject and the class MaturityLevel. This enumeration type allows us to rate every tool, method, and notation according to its maturity.\(^7\)

Multiplicities at the association ends specify constraints on the number of elements contained in an association end. For instance, the 1 at the association end of the association maturityLevel describes that each KO has exactly one maturity level. KOs have

\(^5\) http://www.uml.org/
\(^6\) http://www.informatik.uni-trier.de/~ley/db/
\(^7\) The UML stereotype ≪enumeration≫ is used for classes that have a fixed set of attributes, which are referred to by other classes. This use differs from the specification in the UML standard.
\(^8\) In general, enumeration types allow us to pre-define values a property might have.
several relations between each other, e.g., that one KO Uses another. However, these were extended for the contribution of this work and, hence, we explain them in Sect. 3.2.

Specific types of KOs are Notations, which can be supported by Tools (IsSupportedBy). A Notation supports a grade of formality (GradeOfFormality). Methods can be divided into Activities. Activities (Activity) help to structure Methods and to describe workflows based on Inputs, Outputs, and a Description. This means that an activity can use the output of another activity as input. Techniques have just one action, which makes them less complex than Methods.

Using the presented ontology structure, we can adequately capture and process knowledge in the field of engineering secure software and services. In addition, we evolved the ontology for this work with the elements Term, Study, and SubknowledgeArea and we also extended the attributes of KnowledgeObject. We explain these new elements in Sect. 3.2.

### 3  Structured Research Area Analysis

This section describes the main scientific contribution of this paper. We present in Sect. 3.1 how our work integrates into the approach of Kitchenham described in Sect. 2.1. We show the preparation of our ontology in order to support research area analysis in Sect. 3.2. We describe how to support research area analysis in Sect. 3.3. In Sect. 3.4, we specify the relations between the different ontology parts in detail. These relations enable us to substantiate the envisaged research area analysis by the semantics of the knowledge stored in the ontology.
3.1 Extension/Integration of Kitchenham and CBK

There are several points of integration for the CBK and the literature review process introduced in Sec. 2.1. These points are the inputs Question Types, Question Structure Templates, and the List of Sources to Search In, and the process steps Formulate RQ, Identification of Research, and Selection of Primary Studies (see Fig. 2). To improve the integration for some of these points, the CBK and / or the literature review process have to be adapted.

Question Types The original question types defined by Kitchenham at al are formulated for SLRs [1, 2]. For mapping studies, a selection of questions and their generalization can be found in the works of Kitcheham et al [5], and Petersen et al. [8]. These insights combined with the structure of the CBK (see Fig. 3) result in some new question types:

– How many different KOs exist for the KA in question?
– Which KA(s) are covered by a certain KO?
– Which are the problems and future work mentioned for (a) given KA(s)?
– What is the maturity of KOs for (a) given KA(s)?
– What are the main publications for a given KO or KA?

Question types help to formulate research questions. They give evidence which questions can be of interest and how to formulate them. Moreover, whenever a research question within an actual review maps to one of the questions types given above, this question can be answered by the CBK directly.

Question Structure Templates For the question structure template, Kitchenham and Chaters proposes to use the PICOC criteria framework to structure research questions [2] as we already described in Sect. 2.1. But when investigating the CBK meta-model (see Fig. 3) it seems to be reasonable to add some criteria. The main addition is to define the knowledge area(s) explicitly, unlike having them implicit in the context. In most cases of conducting a SLR or mapping study, there is a very specific focus on a special part of software engineering. An overview given by Kitchenham et al. shows that evidence [4]. This focus should be captured within the criteria, because some electronic sources support to select knowledge areas [6, 2]. Moreover, for mapping studies these knowledge areas are often used for structuring the report [5, 7, 8]. In the case of the CBK the knowledge areas are one of the core concepts, and searching the CBK utilizes the knowledge areas. A minor addition is to distinguish between general terms of the population and special terms of the knowledge area(s). The special terms have a great weight when searching and can help to structure the results [6, 5].

Using this new question structure makes important parts of the questions more explicit. And they ease the use of the CBK, because the separation between general search terms, common terms and knowledge areas are directly reflected in the formulation of the search queries (We will see in Sect. 3.4).

List of Sources to Search In The CBK and its searching capabilities has to be added to the list of sources to search in [2, 6]. Adding the CBK itself is trivial, but for the capabilities it has also to be checked which new capabilities the CBK introduces. For example, the missing relations between KOs and between publications, which
has been found as issue for all existing search sources [6], is explicitly addressed in the CBK. A detailed discussion is skipped at this point due to the lack of space.

Having the CBK in the source list with an explanation of its capabilities helps to plan the research and clarifies in which cases the CBK is of superior use in comparison to other sources.

**Formulate RQ** Besides the two inputs, the Question Types and the Question Structure Templates, the CBK can directly support the formulation of research questions. The CBK already defines an ontology of knowledge areas. With an extension, it will support sub-knowledge areas (see Sec. 3.2). Those areas can help to focus the research questions, because it gives an orientation how to refine knowledge areas and how knowledge areas are related. To find the focus for the own research is considered challenging without such a support [3, 4, 7]. Additionally the adapted CBK presented in Sec. 3.2 provides the common terms used in these knowledge areas. These terms help to sharpen the questions and to avoid ambiguities.

The integration of the CBK and Formulate RQ improves the outcome of the whole literature review process, because it helps to avoid the formulation of imprecise search questions with respect to missing / wrong focus and wrong / ambiguous wording.

**Identification of Research** The ontology of common terms and related synonyms contained in the CBK also helps to formulate the search queries, not only for the CBK, but also for other search sources.

This formulation of search terms is a crucial step and the knowledge about relations between terms and the existence of a synonym list improves this step of finding research a lot [1–8].

**Selection of Primary Studies** For the selection of primary studies the CBK provides information for some comprehensive and sophisticated selection criteria. The CBK already contains information, which is very specific and useful to rate KOs, like the maturity level. And this information is available for all results obtained from the CBK. These criteria are hard to evaluate for results from other search sources [2, 3, 6]. Examples for inclusion criteria:
- Only include KOs and related publications with a certain maturity level
- Only include KOs, which are a core concept of a KA and therefore many other KOs are based on these KOs
- Only include publications, which are considered as most significant by the editors of KOs

Having a set of precise selection criteria, which can be evaluated for all results, improves the outcome of the whole review process. The probability of excluding relevant studies and the bias caused by including literature of low quality can be reduced. And the whole review process speeds up when using the CBK, because the information for evaluating the criteria is available explicitly.

### 3.2 Preparing the Ontology

We prepared the NESSoS CBK ontology (see Sect. 2.2) in order to support the concepts of the structured literature reviews from Kitchenham explained in Sects. 2.1 and 3.1.

NESSoS - 256980
Hence, we extended the CBK ontology (see Fig. 3) with the classes SubKnowledgeArea, Publication, and Study.

We consider keywords and tags in knowledge objects. Keywords are given according to the guidelines of a specific system, e.g., the ACM-keyword-system [11]. Tags are chosen without any restriction and provide the possibility to choose any possible word.

We want to use the CBK ontology for finding research gaps. This requires KnowledgeObjects that do not have a solution description yet, but only problem descriptions. We also included the MaturityLevel None for these KnowledgeObjects. In addition, KnowledgeObjects now have a FutureWork attribute that states research not yet carried out. The difference between these two is that KnowledgeObjects that only have a problem description represent a research area that needs a significant amount of research for providing a solution. The FutureWork attribute in KnowledgeObjects represents possible research that can build upon an existing solution. Researchers that use our approach might look for one or the other.

KnowledgeObjects have relations between each other. These relations are relevant for the investigation of research areas. KOs can be based on other KOs (IsBasedOn), and they can use each other (Uses). In addition, KOs can be used in combination. In this case, one KO Complements another. If KOs can be exchanged, these have a Substitutes relation.

We include the class Term that holds the terms a specific KnowledgeObject uses. The class CommonTerm contains a set of well-defined terms. A Term of a KnowledgeObject is either broader, synonymous, or narrower than a CommonTerm. This allows a comparison of KnowledgeObjects using the CommonTerms. Without these any comparison would lack precision, because terms and notions differ in KnowledgeObjects.

3.3 Identifying Research Gaps Using the CBK

While a Mapping Study and a Problem/Gap Study is supported best by the CBK, it at least partially supports a Structured Literature Review and a Tertiary Study (see Fig. 1). In the following, the CBK support for all different kinds of studies, as defined by Kitchenham, is described in more detail. The support, as described in this section, sketches only a first idea, which will be refined in Sect. 3.4, where we only consider the well supported Mapping Study and Problem/Gap Study.

**Mapping Study** For a Mapping Study researchers have to specify one or more KAs, one or more sub-KAs or one or more common terms. Additionally, they can constrain the search by providing further search terms. The CBK returns KOs grouped by KAs. Depending on the scope of the search, 20 results, for example, for one KA can be interpreted as a sign for maturity of a field or immaturity of a field. For example, 20 results for the KA Requirements has a different meaning than 20 results for the sub-KA Goal-oriented Requirements including the search terms Cloud and Law.

**Problem/Gap Study** Conducting a Problem/Gap Study is also well supported by the CBK. The researcher specifies one or more KAs, one or more sub-KAs or one or more common terms and provides search terms. In this case, the CBK not only groups KOs along the specified KAs, sub-KAs or common terms as described for the Mapping Study, but extends the search to the following classes and fields of the Ontology-Based Identification of Research Gaps 9
CBK ontology: ProblemDescription and FutureWork of the class KnowledgeObject, Problem and Border in the class Study. These results support the creation of a Problem/Gap Study because all relevant information is presented to the researcher in a structured way.

Structured Literature Review While the researcher is able to retrieve KOs for the selected KAs, sub-KAs or common terms and, thus, all relevant literature references, can only be regarded as a starting point to conduct a full-fledged Structured Literature Review. A SLR involves an in-depth analysis of the actual literature, which is out of the CBK’s scope.

Tertiary Study Conducting a Tertiary Study is only supported in theory, because it requires all relevant secondary studies to be part of the CBK. If this is not the case, there is no support for this kind of study using the CBK.

3.4 Formalizing Research Area Analysis

We now identify and specify relevant relations for the identification of research gaps, making use of the ontology and the knowledge it stores and structures. For the analysis, the relations between different tools, methods, and notations, i.e., different KO types, and KAs are of particular relevance.

Allocca et al. [10] present the DOOR method to capture the semantics of relations between different ontologies and to formally specify these relations. While we partly adopt the DOOR steps to support our approach to identify and specify relations between different KOs, we abstain from building an ontology of these relations. We use the ontology structure presented in Sects. 2.2 and 3.2 on the one hand for typing the relations and, more importantly, on the other hand to refine the semantics of the relations. We divide our approach into the following three steps:

1. Identify and specify top-level relations
2. Identify and specify variants and sub-relations, and characterize their algebraic properties
3. Compose relations

We will use the following abbreviations in the formalization: KA = Knowledge Area, SKA = SubKnowledgeArea KO = Knowledge Object P = Publication, ST = Study, CT = CommonTerm, T = Term, and ML = MaturityLevel.

Top-Level Relations: The following relations are abstracted top-level relations that support the kinds of queries sketched in Sect. 3.3.

MappingStudy_KAxKO Describes a mapping study as a relation between knowledge areas and knowledge objects.

MappingStudy_SKAxKO Describes a more fine granular mapping study as a relation between sub-knowledge areas and knowledge objects.

MappingStudy_CTxKO Describes a mapping study as a relation between common terms and knowledge objects.

ProblemGapStudy_KAxKOxST Describes a problem or gap study as a relation between knowledge areas, knowledge objects, and studies.
ProblemGapStudy SKAxKOxST  Describes a more fine granular problem or gap study as a relation between sub-knowledge areas, knowledge objects, and studies.

ProblemGapStudy CTxKOxST  Describes a problem or gap study as a relation between common terms and knowledge objects and studies.

Variants, sub-relations, and Algebraic Properties:  Variants and sub-relations shed light on various facets of the top-level relations with regard to the structured design of the ontology. We express the relations using relational algebra based upon the work of Codd [9]. We use an extension of the relational algebra [12, 13] that offers aggregation and grouping functionalities. The symbol $\xi$ groups the output according to specified attribute(s). $\pi$ projects only specified columns of a table. $\sigma$ selects rows in a table for which specified boolean expression(s) hold. $\bowtie$ joins tables according to common attributes. All rows that do not have these attributes are left out. $\bowtie|$ joins tables, but also displays rows of the left table that do not have all the common attributes. We use relational algebra, because the algebra expressions can be translated to SMW+ queries in a straightforward way, see Sect. 4.

For the specifications of the relations, we assume that the structural design of the ontology presented in Sect. 2.2 and 3.2 is given as tables. Classes that have 1..* cardinalities on both ends of the relation in our ontology require connection tables. Otherwise we would require multiple relations between tables, which is to be avoided during database design. For example, we want to create a table for knowledge objects. One row in the table is allowed to have multiple relations to rows in the knowledge area table. Instead, we would have to create numerous columns for these relations in the knowledge area, because we do not know how many relations we need. Hence, we create a further table for these relations. We denote these tables, which we add in the formalization, with “Connect” and append the names of the classes this table connects. For example, the connection table for the tables KnowledgeObject and KnowledgeArea is stated as: ConnectKAtoKO. These connection tables have two columns, which contain the the primary keys of each of the tables they connect.

Inheritance in the ontology is translated into one main table for the superclass and one table for each class that inherits from this class. These classes have a relation to the table that represents the superclass and have only the additional attributes of the inherited class. For example, the superclass KnowledgeObject has the class Tool that inherits from it, and one of the additional attributes is Input. Hence, we create a table KnowledgeObject and a table Tool, which has the attribute Input.

Searchterm:  We define an algebraic expression ST, which represents a boolean expression for one or more searchterms.

\[ ST ::= ST \odot ST \mid \kappa = \text{String} \mid (ST) \mid \neg ST \]
\[ \odot ::= \land \mid \lor \]
\[ \kappa ::= \text{Tag} \mid \text{Keywords} \mid \text{ExecutiveSummary} \mid \text{Name} \]

For example, the expression “Tags="cloud" \lor Tags="law" ” can be used with $\sigma$ for the table KnowledgeObject. This results in a table with all KOs that have ‘cloud’ or ‘law’ as tags.
KnowledgeArea: We define KAB to be a boolean expression for the selection of one or more knowledge areas.

\[
KAB ::= KAB \lor KAB \mid KnowledgeArea = KAS \\
KAS ::= Requirements \mid Design \mid Implementation \mid Maintenance \mid ConfigurationManagement \mid EngineeringManagement \mid EngineeringProcess \mid Quality \mid Security \mid RiskManagement \mid Privacy \mid Trust
\]

We defined Maturity Level (MLB), Sub-KnowledgeArea (SKAB), and Common Terms (CTB) in a similar manner.

MappingStudy_KAxKO: We specify queries for mapping studies as database relations. We explain the query in detail starting with the \( \sigma_{ST} \), where we join the tables KA, ConnectKAtoKA, and KO and select rows according to ST. The query filters the resulting table for rows that have the required MLB and KAB. The result is projected onto the columns KnowledgeArea, MaturityLevel, Name, ExecutiveSummary, Tags, and Keywords. The query groups the results according to KnowledgeArea, MaturityLevel, Name, ExecutiveSummary, Tags, and Keywords.

\[
MappingStudy_KAxKO = \\
\xi_{KnowledgeArea,MaturityLevel,Name,ExecutiveSummary,Tags,Keywords}( \\
\pi_{KnowledgeArea,MaturityLevel,Name,ExecutiveSummary,Tags,Keywords}( \\
\sigma_{KAB}( \sigma_{MLB}( \\
\sigma_{ST}( KA \bowtie ConnectKAtoKO \bowtie KO ))) ) )
\]

We define MappingStudy_SKAxKO, and MappingStudy_CTxKO in a similar manner.

Extending Searchterm: For a problem gap study we extend the fields of the CBK that can be searched as follows.

\[
... \\
\kappa ::= Tag \mid Keywords \mid ExecutiveSummary \mid Name \mid FutureWork \mid Title \mid Problems \mid Borders \mid OpenQuestions
\]

ProblemGapStudy_KAxKOxST: To perform problem gap studies, we include existing studies in the search relation and enriches the output with problem and future work descriptions. We formalize this relation as an variant of the relation MappingStudy_KAxKO. The symbol \( \bowtie \) between KO and \((ConnectKOtoP \bowtie P \bowtie ST)\) causes that also KOs are selected that do not have a publication or study.

\[
ProblemGapStudy_KAxKOxST = \\
\xi_{KnowledgeArea,Name,ExecutiveSummary,FutureWork,Tags,Title,Problem,Borders,OpenQuestions}( \\
\pi_{KnowledgeArea,Name,ExecutiveSummary,FutureWork,Tags,Title,Problem,Borders,OpenQuestions}( \\
\sigma_{KnowledgeArea,Name,ExecutiveSummary,FutureWork,Tags,Title,Problem,Borders,OpenQuestions}( \\
MappingStudy_KAxKO \bowtie KO \bowtie (ConnectKOtoP \bowtie P \bowtie ST))) )
\]

We define ProblemGapStudy_SKAxKOxST, ProblemGapStudy_CTxKOxST in a similar manner.
Compose relations  Complex relations can be composed from simple ones, as shown in the following example.

**MappingStudy\_KAxSKAxKOxCT:** We merge the different mappings for knowledge area, sub-knowledge area, and the common term and define the following relation that returns the name of the knowledge object, maturity level, the executive summary, the tags, and the keywords, grouped by knowledge area, sub-knowledge area and common terms.

\[
\text{MappingStudy\_KAxSKAxKOxCT} = \\
\xi_{\text{KnowledgeArea}, \text{Sub}-\text{KnowledgeArea}, \text{CommonTerm}, \text{MaturityLevel}, \text{Name}, \text{ExecutiveSummary}, \text{Tags}, \text{Keywords}} (\\
\pi_{\text{KnowledgeArea}, \text{Sub}-\text{KnowledgeArea}, \text{CommonTerm}, \text{MaturityLevel}, \text{ExecutiveSummary}, \text{Tags}, \text{Keywords}} (\text{MappingStudy\_KAxKO} \bowtie \text{MappingStudy\_SKAxKO} \bowtie \text{MappingStudy\_CTxKO}))
\]

**ProblemGapStudy\_KAxSKAxKOxCTxST:** Finally we merge the different problem or gap study relations for knowledge area, sub knowledge area, and the common terms and define the following relation that returns the name of the knowledge object, common term, the executive summary, future work, the tags the keywords, title of the study, problem, borders and open questions grouped by knowledge area, sub-knowledge area and common terms. We specified this relation in the following.

\[
\text{ProblemGapStudy\_KAxSKAxKOxCTxST} = \\
\xi_{\text{KnowledgeArea}, \text{Sub}-\text{KnowledgeArea}, \text{CommonTerm}, \text{Name}, \text{ExecutiveSummary}, \text{FutureWork}, \text{Tags}, \text{Keywords}, \text{Title}, \text{Problem}, \text{Borders}, \text{OpenQuestions}} (\\
\pi_{\text{KnowledgeArea}, \text{Sub}-\text{KnowledgeArea}, \text{CommonTerm}, \text{Name}, \text{ExecutiveSummary}, \text{FutureWork}, \text{Tags}, \text{Keywords}, \text{Title}, \text{Problem}, \text{Borders}, \text{OpenQuestions}} (\text{ProblemGapStudy\_KAxKOxST} \bowtie \text{ProblemGapStudy\_SKAxKOxST} \bowtie \text{ProblemGapStudy\_CTxKOxST}))
\]

To sum up, we applied the DOOR method for the structured creation of ontology relations to the CBK ontology for implementing the Kitchenham structured research area analysis. First, we defined the top-level relations for mapping studies and problem/gap studies. We formalized these relations, using relational algebra, and we derived further relations from these. In addition, we have shown an example for a composed relation of the previously defined relations. We checked all the relational algebra expressions using the relational tool. For future semi-automatic use of the relations and in the light of the technical realization (see Sect. 4), the composition of relations can be left to the users. For example, the Semantic MediaWiki+ allows its users to easily switch the predicates of the relations on and off to generate a result set as required.

### 4 Realization

Our ontology behind the CBK allows us to specify various queries realizing the relations presented in Sect. 3.4 using the SMW query language (SMW-QL). SMW-QL

---

9 http://galileo.dmi.unict.it/wiki/relational/doku.php
Fig. 4: Mapping study support realized as SMW-QL query

was introduced as a comfortable means to query the SMW [14, 15]. The SMW+ platform provides an inline syntax to integrate queries into a wiki page and a graphical query builder to support the creation of such queries (see Fig. 4). In the following, some of the queries specified previously in relational algebra will be translated into SMW-QL. We start with a simple query referring to the relation MappingStudy_KAxKO, followed by a complex query referring to the relation ProblemGapStudy_KAxBxKOxST. The query given in Listing 1.1 is read like this: retrieve all KOs that belong to the KA Security Requirements and which contain the search term attacker and/or invader in the executive summary. The search term can be further specified using comparator operators and wildcards. The result is returned as a table. Each row represents one knowledge object, whereas each column represents an attribute specified in the query indicated by the question mark. In our case, the table contains the columns RefersToKnowledgeArea, HasMaturityLevel, ExecutiveSummary, Tags and Keywords. The table is sorted along the KAs and MaturityLevel. The user is able to customize sorting by clicking on the table’s header. The SMW-QL query given in Listing 1.2 refers to the relation ProblemGapStudy_KAxBxKOxST, thus supporting a Problem/Gap Study. Therefore

### Listing 1.1: Query for supporting a Mapping Study

```sql
{ Ask: [ [Category:KnowledgeObject] ]
  [ RefersToKnowledgeArea: [Security Requirements]]
  [ ExecutiveSummary:˜* attacker * OR [ ExecutiveSummary:˜* invader *]
  } RefersToKnowledgeArea, HasMaturityLevel, ExecutiveSummary, Tags and Keywords.
```

NESSoS - 256980
Ontology-Based Identification of Research Gaps

1. \{ask\} \{[Category: KnowledgeObject] \[Category: Study]\}

2. \{[RefersToKnowledgeArea :: Security Requirements]\}

3. \{[ExecutiveSummary :: attacker\} OR \{[ExecutiveSummary :: invader\}]

4. \{| ? Formula \[Problem \| ? Title \| ? Borders \| ? OpenQuestions \| ? Tags \| ? Keywords \? | ? Problem \| ? Title \| ? Borders \| ? OpenQuestions \| ? Tags \| ? Keywords \?

5. \{sort=RefersToKnowledgeAreas,HasMaturityLevels\}

Listing 1.2: Query for supporting a Problem/Gap Study

it is necessary to additionally output the attributes ProblemDescription and FutureWork from the KO class and the attributes Problem and Border from the Study class.

In contrast to definition of ProblemGapStudy_KAxKOxST in Sect. 3.4, the first SMW-QL query is not reused in this query. While subqueries are in principal possible with SMW-QL, it is recommended to express sub-queries as queries where possible. In this case, it is realized as a flat query, not only because of performance advantages, but also for the sake of simplicity.

5 Related Work

Tools for structured literature reviews that are regularly used by the software engineering research community are major search engines and digital libraries such as ACM, CiteSeer, IEEE Xplore, Google Scholar, Science Direct and Web of Science [11]. All of these work similarly by specifying boolean search expressions. While they differ in evaluating search expressions and ranking the results, it can be stated that none of these search engines and digital libraries was created to support structured literature reviews [6], as our dedicated approach does.

6 Conclusion and Outlook

We have formalized the Kitchenham approach for structured literature reviews in relational algebra. Furthermore, we implemented these queries in an SMW+ ontology. Thus, we provide a semi-automatic support for the Kitchenham approach that eases the burden of manual literature reviews.

Our approach offers the following main benefits:

- Systematic execution of mapping and problem/gap studies according to Kitchenham based upon ontologies for specific domains (here: secure software and service engineering)
- A structured approach to analyze a research area
- Improving the outcome of literature studies via structured processing of knowledge using ontologies
- Further analysis of research domains can be executed with little effort

Our approach has the limitation that it cannot detect research gaps that are not part of the content of the CKB. Hence, the quality of the outcome of our work is dependent on the quality and quantity of CBK content. However, the possibility also exists that
publications might be overlooked when manually executing a literature review according to Kitchenham. Moreover, research gaps, ideas for future work, etc. that only exist in the heads of researchers also cannot be found by any of these approaches.

The work presented here will be extended to support further, more extensive research questions in the future. Examples are the refinement of new findings or the strengthening of own results. We will also work on further automating our approach. We envision an extension of the approach towards other existing ontologies.

References

A Common Body of Knowledge for Engineering Secure Software and Services

Widura Schwittek, Holger Schmidt, Kristian Beckers, Stefan Eicker, Stephan Faßbender, and Maritta Heisel

cpaluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen, Germany
Email: {firstname.lastname}@paluno.uni-due.de

Abstract—The discipline of engineering secure software and services brings together researchers and practitioners from software, services, and security engineering. This interdisciplinary community is fairly new, it is still not well integrated and is therefore confronted with differing perspectives, processes, methods, tools, vocabularies, and standards.

We present a Common Body of Knowledge (CBK) to overcome the aforementioned problems. We capture use cases from research and practice to derive requirements for the CBK. Our CBK collects, integrates, and structures knowledge from the different disciplines on an ontology that allows one to semantically enrich content to be able to query the CBK. The CBK heavily relies on user participation, making use of the Semantic MediaWiki as a platform to support collaborative writing. The ontology is complemented by a conceptual framework, consisting of concepts to structure the knowledge and to provide access to it, and a means to build a common terminology. We also present organizational factors covering dissemination and quality assurance.

Keywords—interdisciplinary, common body of knowledge, knowledge management, software engineering, security engineering, services computing.

I. INTRODUCTION

NESSoS (Network of Excellence on Engineering Secure Future Internet Software Services and Systems) is an EU project comprising 12 partners from academia, industry, and research institutes. It is funded for a duration of 42 months. However, NESSoS aims at constituting and integrating a long-lasting research community on engineering secure software-based services and systems, which outlasts the funding period.

NESSoS is based on the principle of addressing security concerns from the very beginning in system analysis and design, thus contributing to reduce the amount of system and service vulnerabilities and enabling the systematic treatment of security needs through the engineering process. For this approach to work, research from the areas of software engineering (SE), service engineering, and security engineering must be brought together and integrated in such a way that researchers as well as practitioners can easily access and apply the knowledge accumulated in the different areas. To support this goal, we develop a Common Body of Knowledge (CBK).

While existing bodies of knowledge (BOKs) like the Software Engineering Body of Knowledge (SWEBOK) solely rely on books or hypertext systems as a medium, our CBK provides several advantages such as improved flexibility and knowledge access possibilities for its users. Our CBK introduces an ontology that allows users to semantically enrich content. The CBK ontology is based upon a conceptual framework, which introduces a common terminology supporting an increased comprehensibility of research results of different areas.

Another difference between existing BOKs and our CBK is that our CBK heavily relies on user participation realized through a wiki platform. Consequently, the CBK supports collaborative writing and provides mechanisms to build up and update the CBK. Since the CBK will be opened for the public in the future, our approach is complemented by organizational means considering processes such as quality assurance to ensure a high quality of content. The main contribution of this work is to tackle the semantic challenges of finding research in a domain, which we apply to the domain of secure software and service development. However, the contribution can also be applied to other domains.

Note that although the CBK we present in this paper is designed to support the NESSoS project, its underlying concepts are of a general nature. Thus, our CBK concept can be adapted to set up CBKs in other areas. In the current version the ontology is implemented using the SMW+ platform and the members of the NESSoS project currently add content into the SMW+ platform.

The paper is organized as follows: we outline use cases for the CBK in Sect. II. In Sect. III, we present the concepts and an ontology underlying our CBK. We proceed in Sect. IV to briefly present our choice of technology for realizing the CBK. We shape organizational factors of the CBK and report on the CBK’s status in Sect. V. We present related work in Sect. VI. Finally, we conclude and raise ideas for future work in Sect. VII.

1http://www.nessos-project.eu/

2http://www.semantic-mediawiki.org
II. Use Cases

The NESSoS CBK should serve as a toolkit and a handbook for the secure software and services engineering community. It should integrate knowledge grouped into different categories, e.g., the categories method and tool. The CBK should then enable users to be able to insert and retrieve knowledge about individual methods and tools. Therefore, the identification and presentation of relevant use cases constitutes the center of analyzing the functionality and the scope of the CBK. We assume practice and research as well as administrative perspectives for the uses cases, which is reflected in the actors we identified to interact with the CBK. Researchers and practitioners contribute knowledge to the CBK and retrieve knowledge from it. Quality agents review and correct knowledge contained in the CBK and improve its quality. Administrators are responsible for administrative aspects. Note that each of these actors defines a role, i.e., a set of individuals, and these roles might not be disjoint. For instance, a quality agent might be a researcher at the same time.

We present a selection of relevant use cases for the NESSoS CBK in the following:

1) Researchers and practitioners add and modify individuals of content categories.
2) Quality agents and administrators modify and delete individuals of content categories.
3) Researchers, practitioners, and quality agents browse individuals of content categories.
4) Researchers and practitioners search for individuals of content categories according to specific criteria.
5) Researchers and practitioners compare individuals of content categories according to specific criteria.
6) Practitioners discover tool chains according to specific criteria.
7) Researchers identify gaps in practice and in the current research landscape.

Use cases 1 and 2 summarize the relevant operations for adding, deleting, and correcting knowledge stored in the CBK with respect to the different actors. The use cases 3 to 5 refer to "specific criteria", i.e., querying with respect to these criteria has to be explicitly supported by the CBK. For example, we identified the following criteria for methods: notation used, degree of formality, relevance for national and international standards, and the SSDLC (secure software development lifecycle) phase the method supports. We identified similar criteria for other content categories, as presented in Sect. III. Use case 6 represents a query that discovers tools that can be used in combination, i.e., the CBK should enable one to define a notion of compatibility of tools. Use case 7 is a query that could involve a special threshold to define what "gap" means. For instance, gap can mean that there does not exist any individual of a content category that fulfill some criteria, or that there exists a maximum number individuals only.

Based on the uses cases, we present the conceptual, technical, and organizational development of the CBK in the next sections.

III. Conceptual Design

The basic idea behind the structural concept of the CBK is to be able to link arbitrary content categories with each other and to allow actors to browse content along the links between the actual content. Furthermore, the aim is to provide several access possibilities to the CBK, each customized to the target audiences and use cases the CBK addresses. In the following, we introduce the core concepts in Sect. III-A as well as a formalization of these core concepts as an ontology the CBK is based upon in Sect. III-B.

A. Conceptual Framework

The conceptual framework consists of four basic concepts: knowledge objects, knowledge areas, learning trails, and common terminology, which we present in more detail in the following.

A knowledge object (KO) is a content individual, e.g., the concrete description of a method or tool. Each KO can be linked to other KOs, resulting in a network of KOs, which as a whole can be considered as a representation of a BOK of the secure software and services engineering field. The KOs are structured using the four KO types methods, tools, patterns, and notations. We consider them as typical types of contributions to the CBK. This set of KO types is open for extensions in the future.

We adopt the concept of knowledge areas (KA) from the SWEBOK [1] for our CBK, because we regard the fields of security engineering and service engineering as supplements of SE and therefore concerning all SE KAs. KAs span the research field of engineering secure software and services as a whole, dividing it into smaller parts and providing access to subjects of interest in a convenient way. The SWEBOK introduces the differentiation of the SE field into ten KAs on which our KAs are based, e.g., software requirements and software design. In addition, we introduce KAs specific to the fields of security engineering and service engineering based on standard literature. For instance, we introduce the KAs risk analysis and privacy as presented in Anderson's book [2] on security engineering.

Learning trails are a means to provide access to the CBK for different target groups. This idea is based on the fact that content has to be prepared in accordance to the background of the reader. For example, an expert in this area expects more detailed information, whereas a non-expert needs more contextual information in order to be able to understand. Learning trails are designed as moderated tours which guide the reader through a set of KOs, which are considered to be part of a certain topic. Each step builds upon the previous...
step and gives a successive introduction into a topic with respect to the reader’s expert level and background.

The common terminology aims at providing a common language to the community. Thus, it simplifies the translation of a term to another domain with the help of a common term. A common term is a term with a meaning on which an agreement was reached within the community. With the common terminology, we therefore introduce an instrument for defining a common term with a certain meaning and for relating different terms with the same or a similar meaning to this common term. In the opposite direction, the common terminology serves the purpose of a dictionary from which synonyms and translations can be queried. A term does not always have the same meaning of another similar term, so that deviations to the meaning of the common term must be made explicit. In the CBK, this is realized by three different relationship types. A term’s meaning is either synonymical, broader or narrower in relation to another term’s meaning.

More details on the conceptual framework can be found in a paper [3] that specifically focuses on this topic. Next, we formalize the presented CBK concepts using a special CBK ontology.

B. An Ontology for a Common Body of Knowledge

Ontologies are used to capture knowledge about some domain of interest. We use the OWL (Web Ontology Language)\(^4\) terminology in the following. An ontology describes concepts and relations between them. In OWL, a concept is specified in terms of a class, i.e., a set of individuals. An individual represents a concrete object in the domain in which we are interested. In general, in OWL a relation is specified as a property, which represents a binary relation between individuals. Object properties and data properties are distinguished: the former represent relations between individuals or classes, and the latter represent relations between individuals or classes and data values. Both, classes as well as properties can be structured to build complex constructs.

In the following, we present the current ontology underlying the NESSoS CBK in Fig. 1. This CBK ontology can be considered as the explicit description of the domain of creating a CBK for engineering secure software and services. Since the interrelated network of KOs is supported by the CBK ontology, it allows the user to browse or query this network and to possibly uncover new correlations (see use cases in Sect. II). The depicted CBK ontology is a snapshot from the time this paper is written since the CBK is an artifact that is constantly evolving and enhanced by the community participating in this CBK endeavor.

1) Ontology Creation: The CBK ontology which we present in the following sections was created taking two forces into account: the use cases defined in Sect. II and the format for describing design patterns [4].

Along the use cases we started analyzing typical KOs, e.g., looking at concrete descriptions of tools and methods, and we developed ontology classes, data properties, and object properties. For each entity of our domain we had to decide to either model it as a class or as a property. For example, we decided to model an activity of a method as a class, because an activity from our use case point of view is defined as an aggregation of name, input and output artifacts, etc., for which one property would not be sufficient. We created a criteria catalog in order to systemize the decision process for each entity.

Design patterns in the field of SE usually follow a typical structure, essentially consisting of the parts “context description”, “problem description”, “solution description” and “consequences description”. We decided to not only choose this pattern structure as a form to codify Patterns, but also to structure the descriptions of Method, Tool, and Notation. This decision was mainly taken because design patterns are a well-known and widely accepted reuse concept in the SE community, the pattern-based KO structure is easy to recognize and thus improves understanding. Additionally, specifying the problem solved by a KO individual and the corresponding context explicitly supports access for a broader audience. An overview of all pattern-related properties with short descriptions is given in Table III.

2) Ontology Classes: The result of the ontology creation process is depicted in Fig. 1, which we describe in detail in the following. We use a UML (Unified Modeling Language)\(^4\) class diagram to represent the underlying text-based OWL notation. In the CBK ontology, each concept from our conceptual framework (see Sect. III-A) is represented by an ontology class, namely KnowledgeObject, KnowledgeArea, LearningTrail, and CommonTerm. They are highlighted by a light gray background color. The KnowledgeObject class is further specified by the more specialized subclasses Tool, Method, Pattern, and Notation representing different KO types as described in Sect. III-A. They are highlighted by a dark gray background color. Furthermore, classes are introduced that mainly serve as supplements for the main concept classes. On the one hand, we have the supplement classes Publication, UsageExample, Image, Link, ToolFunction, LearningTrailStep, Role, TargetGroup, and Activity as a container for complex data structures, which cannot be modeled as properties. On the other hand, the classes KnowledgeArea, MethodMaturityLevel, ToolMaturityLevel, and SSDLCPhase represent enumeration types, each containing a predefined set of individuals listed inside the corresponding classes in Fig. 1. This means that they define a fixed range of values, which can be referenced by other classes. For instance, the individual

\(^3\)http://www.w3.org/TR/owl2-overview/

\(^4\)http://www.omg.org/spec/UML/2.3/
“MagicUWE” of the class Tool references the predefined individual “(1) ProofOfConceptOrPrototype” from the class ToolMaturityLevel out of in total four possible values.

3) Ontology Properties: Our ontology defines relations in terms of object properties (between classes) and data properties (between classes and primitive data types). In the following, we present different kinds of properties as examples.

Table I shows properties central to the KO sub-class Method, i.e., this class is the domain of these properties. A method can be supported by a tool or a number of tools (HasMethodSupportFor). A method might consist of sub methods (HasMethodSubMethod), and it can be the successor of other methods (HasMethodPredecessor). Each method consists of at least one activity (HasMethodActivity).

Our ontology describes properties for the KO sub-class Tool, some of which we present in Table II. A tool can be supported by a method or a number of methods (HasToolSupportFor), and it might use other tools (HasToolUses).

Some properties are common to all kinds of KOs. Therefore, we introduce these properties with the class KnowledgeObject as a domain in Table III. We describe each KO by its context (HasKOContextDescription), problem (HasKOProblemDescription), solution (HasKOSolutionDescription), and consequences (HasKOCOnsequencesDescription). A KO can be associated with KAs (HasKOKA). A KO might be a synonym term (HasKOSynonymTerm), a broader term (HasKOBroaderTerm), or a narrower term (HasKONarrowerTerm) compared to a common term.

As an example for those classes that we introduced for structuring purposes, we present some Activity class properties in Table IV. An activity must have a name (HasActivityName) and a description (HasActivityDescription).
An activity might use input artifacts (HasActivityInput) and can generate output artifacts (HasActivityOutput). These properties represent textual descriptions; hence they are String properties.

As another example for a class with a structuring purpose, we present the class LearningTrailStep using the data properties shown in Table V. The first learning trail step is associated with the LearningTrail class. Except the first and the last step, learning trail steps have a predecessor (HasLearningTrailStepPredecessor) and successor (HasLearningTrailStepSuccessor) step. Moreover, each learning trail step has a description (HasLearningTrailStepDescription) and might refer to a number of KOs (HasLearningTrailStepReferenceToKO).

In the following section, we present knowledge queries utilizing the various properties.

4) Knowledge Queries: The properties defined in our ontology allow us to specify various knowledge queries realizing the use cases presented in Sect. II.

Use cases 3 and 5 (the ones referring to specific criteria) in Sect. II can be realized by queries of the form

Property\_Name value Criterion

For example, the query

HasMethodMaturityLevel value ProofOfConceptOrPrototype

retrieves all methods validated by a proof of concept or prototype implementation. Using conjunction, more elaborate queries can be developed. For example, the query

HasMethodMaturityLevel value ProofOfConceptOrPrototype and HasKORelevanceForStandards value CommonCriteria

further restricts the previous example query by considering only methods that support the Common Criteria standard. The other use cases rely on the technical realization of the CBK. Especially the use cases 6 and 7 (discover tool chains and identify research and practice gaps) have to be implemented using additional technical means (see Sect. IV).

After the presentation of the ontology, we continue with the technical realization and the organizational support of the CBK.

IV. Technology

We have realized the CBK using the the Semantic MediaWiki, an extension of the MediaWiki platform that is quite renowned empowering the popular Wikipedia encyclopedia. The Semantic MediaWiki has been partially funded by the project Halo, which in turn is an extension to the Semantic MediaWiki featuring better usability options. The whole conglomerate of the MediaWiki, the Semantic MediaWiki extension, and the Halo core extension is called SMW+.

SMW+ is an open source semantic enterprise wiki for creating and sharing knowledge based on web-oriented collaboration and quick authoring and provision of content. This means that each web page is in principle editable by anyone. It also includes the possibility to version and to discuss the content. Quality is ensured by the masses of people reading and working with the content. The wiki principle enables them to instantly correct or report wrong information or vandalism. Smaller organizations lack of this mass effect, but can ensure quality by organizational means or access control assignable to groups or individuals. Access

---

### Table I

<table>
<thead>
<tr>
<th>Name</th>
<th>Range</th>
<th>Mult.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasMethodSupportFor</td>
<td>tool</td>
<td>U.*</td>
</tr>
<tr>
<td>HasMethodSubMethod</td>
<td>Method</td>
<td>0..*</td>
</tr>
<tr>
<td>HasMethodPredecessor</td>
<td>Method</td>
<td>U.*</td>
</tr>
<tr>
<td>HasMethodSuccessor</td>
<td>Method</td>
<td>U.*</td>
</tr>
<tr>
<td>HasMethodActivity</td>
<td>Activity</td>
<td>1.*</td>
</tr>
</tbody>
</table>

**METHOD PROP. (EXCERPT)**

### Table II

<table>
<thead>
<tr>
<th>Name</th>
<th>Range</th>
<th>Mult.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasToolSupportFor</td>
<td>Method</td>
<td>0..*</td>
</tr>
<tr>
<td>HasToolUses</td>
<td>Tool</td>
<td>0..*</td>
</tr>
</tbody>
</table>

**TOOL PROP. (EXCERPT)**

### Table III

<table>
<thead>
<tr>
<th>Name</th>
<th>Range</th>
<th>Mult.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasKOContextDescription</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td>HasKOPopulationDescription</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td>HasKOSolutionDescription</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td>HasKOCertaintyDescription</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td>HasKOKA</td>
<td>KnowledgeArea</td>
<td>U.*</td>
</tr>
<tr>
<td>HasKOsynonymTerm</td>
<td>CommonTerm</td>
<td>0..*</td>
</tr>
<tr>
<td>HasKOBroaderTerm</td>
<td>CommonTerm</td>
<td>0..*</td>
</tr>
<tr>
<td>HasKNarrowerTerm</td>
<td>CommonTerm</td>
<td>0..*</td>
</tr>
</tbody>
</table>

**KNOWLEDGEOBJECT PROP. (EXCERPT)**

### Table IV

<table>
<thead>
<tr>
<th>Name</th>
<th>Range</th>
<th>Mult.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasActivityName</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td>HasActivityDescription</td>
<td>String</td>
<td>1</td>
</tr>
<tr>
<td>HasActivityInput</td>
<td>String</td>
<td>0..*</td>
</tr>
<tr>
<td>HasActivityOutput</td>
<td>String</td>
<td>0..*</td>
</tr>
</tbody>
</table>

**ACTIVITY PROP.**

---


6[http://www.semantic-mediawiki.org](http://www.semantic-mediawiki.org)

7[http://www.projecthalo.com](http://www.projecthalo.com)
control functionality is not part of the MediaWiki but is a feature of SMW+.

The major feature of the SMW+ extension is the possibility to add semantic annotations to wiki pages. Semantics are meta-data that allow better processing, searching, and reusing of knowledge. The semantic annotations are formalized by an ontology as presented in Sect. III-B. This allows the SMW+ to realize queries such as the ones developed in Sect. II without the need to look at each relevant wiki page and calculate the result manually. Thus, semantically enriched wiki pages make it possible to query the wiki like a database. Aggregated data is kept up-to-date, can be explored interactively, and can be visualized.

The concept of semantic forms helps to create pages with predefined annotated data, because the content of form fields is automatically annotated in the resulting wiki page. Forms are generated with the help of wizards. By choosing a class and defining a subset of attributes from the ontology, a form is generated for a specific class with input fields according to the selected attributes.

In SWM+, every part of the ontology is represented by a wiki page. A wiki page exists for each class, each property, and each individual. Creating our CBK ontology in SMW+ literally means creating wiki pages for each element presented in Sect. III. After creating classes and properties, class individuals have to be provided.

KO individuals (see Sect. III), the actual contents of the CBK, such as concrete methods, tools, patterns, and notations are provided by community members. Semantic forms are supposed to support the creation of such KO individuals for a certain class. Semantic forms are initially generated along the properties of a class. This means that for each property a corresponding input element for filling in the property value is placed on this form. The generated form can be adapted afterwards to implement user interface specialties such as grouping the generated input elements into areas or tabs. We have therefore generated and adapted semantic forms for each KO subclass so that community members are able to contribute KO individuals more comfortably through these forms.

All KA individuals were created along the KAs defined in Sect. III-A. On the one hand, KA individuals can be referenced during the KO individual creation; on the other hand, all KO individuals referencing a KA are queried and presented, when browsing a certain KA individual wiki page.

To create a learning trail individual (see Sect. III-A), a description of the learning trail step, its predecessor and successor (if applicable), and references to KOs (if applicable) must be provided.

In conjunction with SMW+, we use the open source tool Protége8 to edit our CBK ontology. Since both products are able to handle the OWL ontology format, we established a round trip workflow in which individuals are provided through the SMW+, which are then imported into Protége. In Protége we are able to fine-tune and test the ontology and queries. Adaptations to the ontology or queries are then imported back into the SMW+, where semantic forms and individuals have to be regenerated and customized again depending on the adaptations made.

V. PROCESSES AND STATUS

A CBK has the greatest benefit, if it is complete, up-to-date, and valid. Especially in the beginning of such a project this is not the case, leading to low acceptance and low user participation, if launched for the public too early. We therefore conceived three phases, each with a different focus and participation style in order to work against this effect: The first phase is a planning phase, in which all discussed aspects are considered while preparing the initial CBK structure and planning. During an inception phase, content is provided by a closed user group, consisting of experts from different areas within the engineering of secure software and services field. The result of this second phase comprises a sound CBK content base providing a complete, up-to-date, and validated state-of-the-art of this interdisciplinary research field. The run phase is marked by the launch of the CBK for the general public in terms of reading and writing. At this point in time, the CBK should provide a complete overview of the research field of engineering of secure software and services.

More details on the organizational factors can be found in paper [3] that specifically focuses on this topic.

Considering the NESSoS CBK, we already finished the planning phase (months 1 to 6), and we are currently in the very first part of the inception phase (at month 12; the inception phase will run until month 30). After that the NESSoS CBK will made available to the public9.

\footnotetext[8]{{http://protege.stanford.edu/}}

\footnotetext[9]{{http://www.nessos-project.eu/cbk}}
Figure 2 shows the continuing growth of KOs in the CBK over the first twelve months of the NESSoS project. It also shows how the KOs (68 in total at month 12) are divided into the four currently available KO types. Most of the KOs (28) are techniques closely followed by tools (24). The other KOs are split into notations (9) and methods (7).

VI. RELATED WORK

The concept of a codified BOK is not new and can be found in many different disciplines. Compared to our CBK they all differ in how they were created and in how knowledge is codified.

Besides the already introduced SWEBOK [1], the Project Management Body of Knowledge (PMBOK) [5] is a prominent BOK. It covers project management knowledge in general. The PMBOK has influenced many subsequent BOK efforts in the computing disciplines. In the security field, BOKs exist with different content-wise focal points promoted by both industry and governments. Examples are the (ISC)² Common Body of Knowledge [6] created by the non-profit organization “International Information Systems Security Certification Consortium, Inc.” and the Information Technology Security Essential Body of Knowledge [7]. Both serve the purpose of defining a basis for curricula and profession profiles, as well as certification programs for training IT security personnel. Planned but not yet realized efforts towards BOKs in the SE field are, for instance, the Body of Knowledge for Software Quality Measurement [8], the Common Body of Knowledge for Information Security [9], the Requirements Engineering Body of Knowledge (REBOK) [10], [11], and the Computer Engineering Technology Body of Knowledge [12]. A more collaborative approach is taken by the two BOK projects Usability BOK [10] and Build Security In [11] by the U.S. Department of Homeland Security, both also fostering user participation to provide content following a bottom-up approach.

In contrast to our CBK, all of the BOKs presented so far cannot be queried elaborately. Most of the BOKs only exist as books with access possibilities given by the table of contents or a keyword index, while others also provide a hypertext system, allowing one to browse content along links, such as the online version of the SWEBOK [12], the IEEE Body of Knowledge on Services Computing [13] and the Guide to the Systems Engineering Body of Knowledge (G2SEBoK) [14]. Moreover, the presented BOKs were created top-down, i.e., an expert team was formed or authors were chosen to write articles. Our approach comprises a top-down phase, but also a bottom-up phase in which the CBK is opened to the public in terms of reading and writing (see Sect. V).

The need for defining a common terminology for different interdisciplinary communities led to a large number of publications in this area. These include the work by Firesmith [13] for safety and security and by Fabian et al. [14] for SE and security, to name only two examples. Similar to our common terminology concept, their approaches define taxonomies relating fundamental notions across the different disciplines, and they specify to what extent notions from one discipline can be translated into notions of the other discipline. The main difference to our work is that Firesmith and Fabian et al. do not complement their results by further concepts such as KAs, learning trails, etc. to create a CBK.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an approach to structure knowledge of a domain using an ontology. The novelty of the approach is that the knowledge can be queried using the defined structure. Thus, we do not limit the search for knowledge to keyword index searches. In the current example we support the bringing together the fields of SE, service engineering, and security engineering based on a CBK. The main contributions of our approach are:

- KOs allow users to structure knowledge such as best practices and research results according to their type. Provided content is semantically enriched in an automated manner. This allows users to run complex queries on the CBK.
- The CBK introduces a mechanism to group knowledge into KAs. This provides access to the CBK via a hierarchical taxonomy and represents a valuable instrument to discover gaps in research and practice.
- The common terminology helps the community to find a common language of the different disciplines, and to define and use translations.
- Learning trails provide access to the CBK for a broader audience, practitioners and researchers in particular.
- User participation is supported by adequate processes and by the chosen framework SMW+, which might lead to a more up-to-date, a more comprehensive, and a sustained CBK.

In the future, we plan to include first feedback from the community in a revised version of the CBK ontology in order to better identify gaps in the field and to improve the possibility of comparing KOs. Moreover, we want to include user rating mechanisms in our CBK concept.

ACKNOWLEDGMENT

This research was partially supported by the EU project Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS, ICT-2009.1.4 Trustworthy ICT, Grant No. 256980).
REFERENCES


[4] E. Gamma, R. Helm, R. E. Johnson, and J. Vlissides, Design Patterns - Elements of Reusable Object-Oriented Software. Addison Wesley, 1995.


