Technical Report

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Building Empirical Software Engineering Bodies of Knowledge with Systematic Knowledge Engineering

Stefan Biffl[1]
Marcos Kalinowski[2], Fajar J. Ekaputra[1], Estefania Serral[1], Dietmar Winkler[1]

[1]Christian Doppler Laboratory CDL-Flex, Institute of Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstrasse 9-11/188, AT 1040 Vienna, Austria
{stefan.biffl, fajar.ekaputra, estefania.serral, dietmar.winkler}@tuwien.ac.at

[2]Federal University of Juiz de For a, NEnC,
Rua José Kelmer s/n 36036-330 Juiz de Fora, Brazil
kalinowski@ice.ufjf.br

Technical Report No. IFS-CDL 13-03
Issued: October 2013

Building Empirical Software Engineering Bodies of Knowledge with Systematic Knowledge Engineering

Stefan Biffl[1] Marcos Kalinowski[2] Fajar J. Ekaputra[1] Estefanía Serral[1] Dietmar Winkler[1]

[1]Vienna University of Technology
CDL-Flex, Favoritenstr. 9/188
1040 Vienna, Austria
+43 1 58801 18810
<firstname>.<lastname>@tuwien.ac.at

[2]Federal University of Juiz de Fora
NEnC, Rua José Kelmer s/n
36036-330 Juiz de Fora, Brazil
+55 32 2102-3311
kalinowski@ice.ufjf.br

ABSTRACT

[Context] Empirical software engineering (EMSE) researchers conduct systematic literature reviews (SLRs) to build bodies of knowledge (BoKs). Unfortunately, valuable knowledge collected in the SLR process is publicly available in research syntheses reports only to a limited extent, which considerably slows down building BoKs incrementally. [Objective] In this paper, we introduce the Systematic Knowledge Engineering (SKE) approach to support building up EMSE BoKs from empirical studies efficiently. [Method] SKE extends the SLR process and provides a Knowledge Base (KB) to reuse intermediate data extraction results in future research analyses. We evaluated the SKE approach by building a software inspection EMSE BoK KB from knowledge acquired by controlled experiments. We elicited relevant queries from EMSE researchers and systematically integrated information from 30 representative research papers into the KB. [Results] The resulting KB was effective and efficient in answering the relevant queries, enabling knowledge reuse for analyses beyond the results from the SLR process. [Conclusion] The SKE approach showed promising results in the software inspection context and should be also evaluated in other contexts for building EMSE BoKs faster.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software Validation
I.2.4 [**Artificial Intelligence**]: Knowledge Representation

General Terms

Measurement, Experimentation, Theory, Verification.

Keywords

Empirical software engineering, knowledge base, systematic knowledge engineering, systematic literature review, software inspection.

1. INTRODUCTION

Researchers in Empirical Software Engineering (EMSE) collaborate on research topics, such as defect detection methods for software inspection [1], to build up a body of knowledge (BoK). An EMSE BoK includes theory models [2], hypotheses derived from the theory models, and results from empirical studies that test those hypotheses [3], to explain and/or predict EMSE phenomena.

A consequence of the growing number of empirical studies in software engineering is the need to adopt systematic approaches for aggregating research outcomes in order to provide a balanced and objective summary of evidence on a particular topic [4]. In this context, systematic literature reviews (SLRs) have become a widely used research method [5]. An advantage of SLRs is their systematic review scope definition, enabling incremental research on top of the results of previous SLRs in the context of a BoK.

However, currently the main public result of a SLR is, in general, a specific research synthesis report [6], while the accumulated knowledge in the SLR working material, generated from heterogeneous empirical study data sources [7], is not available to other researchers. Therefore, each new SLR project has to overcome knowledge sharing issues [8] and rebuild large parts of the existing knowledge from previous SLRs, which makes building a BoK considerably less efficient and slower than necessary. Moreover, meta-analyses are limited to the presented research synthesis, not allowing other researchers in the EMSE BoK community to explore the underlying extracted information in different ways in order to answer questions related to their specific research goals.

Figure 1 illustrates current challenges in an EMSE BoK community [9] with key stakeholders, artifacts, and technologies. EMSE BoK researchers produce SLR and empirical study reports, available to other researchers and general readership through digital libraries. However, the following research challenges can be observed: (1) data extracted during the SLR process usually stays in a local archive and (2) the SLR report contains a specific research synthesis and there is seldom a way for other BoK researchers to access the intermediate extracted data to apply different analyses and research synthesis methods, or to extend the data in the BoK.

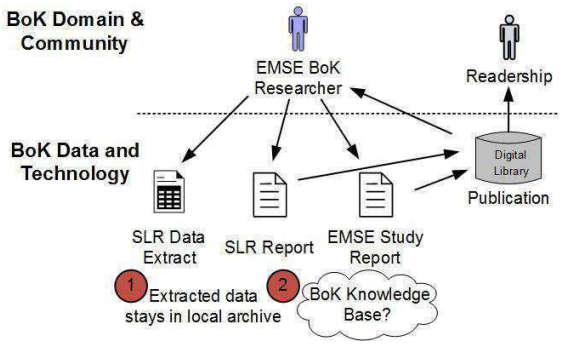

Figure 1. EMSE BoK research challenges.

In this paper we address these challenges by introducing the Systematic Knowledge Engineering (SKE) approach, which supports systematically and efficiently building up EMSE BoKs from empirical studies. The SKE process builds on the SLR process [10], improving data management by storing knowledge on BoK domain concepts that are typically used for theory building [2] from EMSE studies [3] in a Knowledge Base (KB). The semantic technology used in SKE facilitates semantic search, e.g., on synonyms

and related concepts, which goes beyond the syntactic search capabilities of typical digital libraries and search services. SKE enables researchers to identify relevant knowledge in the BoK by querying the KB, which can be incrementally extended [11].

The SKE process and KB allow truly building up knowledge incrementally as researchers can access and reuse knowledge on past studies and integrate their new contributions. Moreover, the KB enables researchers to explore the knowledge aggregated from EMSE study reports through an extensible set of relevant queries. Those queries can be built according to their specific research synthesis needs, based on theory elements in the SKE KB data model, such as hypotheses, response variables, treatments, results, and study findings. For researchers not familiar with the KB tool set, a knowledge engineer can provide an adequate user interface.

For evaluation we instantiated the SKE approach to build an initial Software Inspection EMSE BoK bottom-up based on a specific type of empirical study: controlled experiments. First, we elicited a set of most relevant query candidates in a survey with EMSE BoK researchers. Then we modeled the BoK topics related to software inspection theory and the common data model for controlled experiment data on those topics, defining the relevant information to be extracted. Following the SKE search process, 102 experiments on software inspections were identified. For the evaluation an independent team of six EMSE experts extracted information from the 30 most recent representative papers for data import and query resolution by a knowledge engineer.

Major findings of the evaluation were: 1. The SKE approach was effective in identifying and characterizing similarities and variations of inspection experiments and their results. 2. It was possible to build the software inspection EMSE BoK KB from published controlled experiment data. 3. The SKE KB was effective and efficient in providing answers to the required BoK queries.

Major findings from comparing the SKE process to the corresponding steps in the traditional SLR process, when applied to build EMSE BoKs, were: 1. The SKE KB allows reusing and exploring the underlying data based on semantic search capabilities that are not available for SLR reports, e.g., searching for domain concepts such as effectiveness (and its synonyms) in the hypotheses of empirical studies. 2. The effort to extract data from empirical study reports to spreadsheets is comparable to the data extraction effort in a typical SLR process. However, SKE also requires a knowledge engineer for data modeling, mapping, and providing query facilities. In an EMSE BoK community the overhead of the knowledge engineer can be offset by the benefits of the KB, which can be used among EMSE BoK researchers in and beyond a work group to incrementally build up EMSE BoKs.

The remainder of this paper is organized as follows. Section 2 summarizes related work. Section 3 motivates the research issues. Section 4 introduces the SKE approach and Section 5 presents the evaluation approach. Section 6 discusses the evaluation results and lessons learned. Section 7 summarizes the research results and proposes future research work.

2. RELATED WORK

The quality and speed of building up a body of knowledge (BoK) in an EMSE research area depend on the ability of researchers to discover the existing content in a BoK, e.g., empirical studies investigating a set of hypotheses or variables. Currently, online searching for content is supported by syntactic full-text search on specialized databases, such as digital libraries. However, support for semantic searching and linking between data sources is limited [12]. Thus, researchers may not discover all relevant content.

Even though some effort has been spent on repositories for empirical studies (e.g., CeBASE [13] and ViSEK [14]), they did not show significant progress since their introduction. To our knowledge, there is no related work on using SLR-based study identification and KB integration for bottom up EMSE BoK building to facilitate reuse and semantic search. Therefore, this section describes work related to the theoretical foundations of this main research constructs: Systematic Literature Reviews (Section 2.1) and Knowledge Base Design and Population (Section 2.2). Further, we describe the foundations on Software Inspection EMSE BoKs (Section 2.3) and Empirical Studies on Software Inspections (Section 2.4) as input to the evaluation use case.

2.1 Systematic Literature Reviews

Kitchenham and Charters [10] developed guidelines for performing Systematic Literature Reviews (SLRs) in the software engineering domain. In those guidelines they state that the main reasons for conducting SLRs are (a) summarizing the existing evidence concerning a treatment or technology; (b) identifying gaps in current research in order to suggest areas for further investigation; and (c) providing background in order to appropriately position new research activities.

The first reason is directly related to building BoKs by gathering evidence-based knowledge. In the context of this research, the main advantage of using SLRs is allowing to systematically summarize such knowledge on a specific BoK scope and afterwards enable incremental updates on top of previous SLRs. An example of such updates is available by Kalinowski et al. [15], where four independent SLR trials were conducted in separate years in order to incrementally build evidence-based guidelines on defect causal analysis.

The SLR guidelines by Kitchenham and Charters [10] summarize three main phases of a systematic review: Planning the Review, Conducting the Review, and Reporting the Review. The stages related to each of those phases are: (a) Planning the Review: identification of the need for a review, commissioning a review, specifying the research questions, and developing a review protocol; (b) Conducting the Review: identification of research, selection of primary studies, study quality assessment, data extraction and monitoring, and data synthesis; and (c) Reporting the Review: specifying dissemination mechanisms, formatting the main report, and evaluating the report.

The PICO (population, intervention, comparison, outcome) strategy [16] has been suggested [10] for detailing the research question elements in order to support developing the review protocol. Lessons learned from applying SLRs to the software engineering domain are reported elsewhere [4].

SLRs have become a widely used and reliable research method [5]. However, the contributions of current SLRs are limited to their main public result, in general, in the form of a research paper reporting on a specific research synthesis done by the authors. Unfortunately, reusable SLR packages that include the working material, which holds the accumulated knowledge, are seldom available. The working material includes the data extracted from the primary studies (commonly performed using spreadsheets). Therefore, a new SLR has to rebuild large parts of the existing knowledge, which makes the addition of knowledge less efficient and slower than necessary. Even if such knowledge were availa-

ble, data integration mechanisms to enable making the knowledge available for further use by other EMSE BoK researchers have not been discussed in this context. In summary, the reuse value of SLR knowledge to help building EMSE BoKs is limited.

2.2 Knowledge Base Design and Population

The process of building a knowledge base may be seen as a modeling activity [17]. Building a knowledge base means building a computer model with problem-solving capabilities comparable to a domain expert [17]. For creating a knowledge base, it is essential to capture domain knowledge through content-specific agreements, so both human and knowledge-based systems can access and use the information [18].

For this purpose, formal ontologies have been successfully used since the 1990s [19]. Ontologies can provide standard terminologies and rich semantics to facilitate knowledge sharing and reuse. Technically, an ontology involves a well-formed vocabulary that is machine and human interpretable and defines clearly the terms of the domain and the relationships. OWL DL is the most used language for ontologies as it has the capability of supporting semantic interoperability to exchange and share context knowledge between different systems, and keeps a balance between expressiveness and automatic processing. In addition, ontology enhances searching mechanisms, which may refer to precise semantic concepts rather than simple syntactic keywords, facilitating the use of the knowledge stored in the ontology [18].

Many methodologies have been proposed to design ontologies [20]. A comparative and detailed study of these methodologies can be found at [21]. However, only a few of them consider collaborative and distributed construction of ontologies, such as the Collaborative Design Approach [22], which addresses the issue of collaborative construction of the ontologies by identifying and involving a diverse panel of participants, e,g., in an EMSE BoK.

Once the ontology or the data model of the KB is defined, it is necessary to capture the extracted data from, often heterogeneous, information resources in accordance to the KB. This process is called the KB population, and involves the creation, transformation and integration of individuals (instances) into the KB. In our case, the information resources for creating the EMSE BoK KB are empirical study reports.

The KB population process may cause data integration problems if the different information resources use varying structures to represent the same or overlapping concepts [23]. The Interchange Standard Approach, based on ontologies has been stated as one of the best solution options for semantic integration [24]. However, the currently available tools to manage ontologies usually require ontology experts; therefore ontology non-experts need to be provided with effective and efficient interfaces for both, importing and exporting knowledge (e.g., via spreadsheets), and for querying the knowledge base (e.g., via a web interface that builds on the domain concepts known by the intended users).

2.3 Software Inspection EMSE BoK

Software inspections improve software quality by the analysis of software artifacts, detecting defects for removal before these artifacts are delivered to following software life cycle activities [1]. The traditional software inspection process by Fagan [25] involves a moderator planning the inspection, inspectors reviewing the artifact, a team meeting to discuss and register defects, passing the defects to the author for rework so they can be corrected, and a final follow-up evaluation by the moderator on the need of a new inspection.

Empirical software engineering (EMSE) research groups investigate related topics in empirical studies to build a body of knowledge (BoK) on software engineering. Biffl et al. [9] recently presented an ecosystem view of how research actors interact to build EMSE BoKs. Based on software inspection papers containing empirical studies, we informally estimate this research community to consist of around 150 to 200 researchers conducting empirical studies, with about 10% of them active in conducting SLRs. Given activities in expert networks, we estimate a wider audience of around 1,500 to 3,000 domain experts, practitioners, and researchers, who are interested in software inspection on a general level and want to use and discuss research results.

To support a software inspection BoK community in building up their BoK, specific BoK topics can help to define the scope of knowledge. The IEEE Software Engineering BoK (SWEBoK) [26] breaks down the Testing BoK into the following topics: Fundamentals, Test Levels, Test Techniques, Test-Related Measures, Test Process, and Software Testing Tools. Software inspections relate to test techniques in the IEEE SWEBoK. Nevertheless, they can be seen as a similar topic of interest and a hierarchical structure for them could be useful to facilitate organizing knowledge.

However, a fixed topic breakdown may be difficult since it is possible to apply several variant options of inspections. Laitenberger and DeBaud [27] provide some parameters that help define inspection variant options based on an early literature review on software inspection experiments: inspection artifacts (e.g., requirements, design or code), inspection process (e.g., inspection with or without a group meeting), and inspection methods (e.g., reading techniques). This list of parameters is a good starting point for a software inspection EMSE BoK topic structure.

2.4 Empirical Studies on Software Inspection

Over the years, many contributions on software inspections have been proposed, including alternative inspection processes, inspection methods to improve the effectiveness and efficiency of inspectors in detecting defects, models and guidelines supporting tasks of the inspection process that involve decision making, and tool support [28]. Some knowledge on those contributions has been acquired by conducting empirical studies and can be considered part of a BoK in the software inspection area. Examples of such studies are provided hereafter.

Regarding the inspection process, Votta [29] and others argued that by avoiding synchronous inspection meetings, costs and scheduling conflicts can be reduced without sacrificing inspection effectiveness. Several empirical studies, such as the experiment conducted by Johnson and Tjahjono [30] reinforce this argument.

Specific inspection methods have also been evaluated. Maldonado et al. [31], replicated experiments on the effectiveness of the perspective based reading (PBR) technique [32]. Winkler et al. [33] investigated the temporal behavior of defect detection when applying specific techniques.

Regarding models and guidelines, Biffl et al. [34], for instance, proposed and evaluated a cost-benefit model to estimate cost-effectiveness of re-inspections in software development. Walia and Carver [35] and others evaluated capture-recapture models for estimating the number of defects in an artifact and determining whether a re-inspection is necessary.

Tool support has also been addressed by empirical studies, e.g., Kalinowski and Travassos [36] reported on an experiment on a distributed inspection planning support tool. Lanubile et al. also evaluated tool support for distributed inspections [37]. Moreover, Biffl et al. [38] conducted a family of experiments to investigate the effects of groupware on software inspections.

While there are numerous empirical studies on software inspections, their acquired knowledge is currently not organized in the context of an EMSE BoK. Therefore it still takes considerable expertise and effort to identify studies and study results relevant for a given topic in the scope of software inspections.

3. RESEARCH ISSUES

The overall goal of the Systematic Knowledge Engineering (SKE) approach is to support efficiently building up an EMSE BoK incrementally by providing a process for knowledge acquisition and querying. From this goal we derive the following research issues.

Research Issue RI-1: SKE Requirements Analysis. Which queries to knowledge on empirical studies are most relevant to EMSE BoK stakeholders?

EMSE researchers want to synthesize EMSE BoKs but tend to focus on conducting empirical studies and seem to spend much less effort on considering data management to provide their EMSE BoK community with suitable access to the created knowledge [7]. Building on typical hypothesis patterns reported by Sjøberg et al. [2] we conducted an informal survey with EMSE BoK researchers from several active work groups on queries to knowledge on empirical studies that they need for their work. We are aware of the limitations of this survey and see the survey outcome as a preliminary working result, which is still useful to drive the SKE research at this stage, and can be extended by a future survey in a wider scope.

Research Issue RI-2: SKE Process and Data Modeling. How can the traditional SLR process be adapted to support incrementally building EMSE BoKs? Which data elements are necessary to address the most relevant queries of EMSE BoK researchers?

The SKE process builds on the traditional SLR process [10] and on the Collaborative Design Approach (CDA) [22] for knowledge engineering. Key idea is to loosen the tight connection between SLR data extraction and data synthesis in order to allow collecting knowledge from EMSE studies effectively and efficiently as input to a range of research synthesis methods in a BoK community. Second key goal is to enable incrementally building up knowledge in the context of an EMSE BoK. We evaluate the resulting SKE process regarding effort and added benefits.

In a traditional SLR project data extraction and synthesis are, in general, focused on ensuring a traceable research analysis process in a work group, which leads to high efficiency for the project but limited use (and availability) of the intermediate results to the BoK community. To address this shortcoming, SKE aims at designing a common KB data model that captures both, concepts from the BoK domain (e.g., software inspection), and from the selected types of EMSE studies (e.g., controlled experiments). While there are recommendations on modeling measurement data [39], an important issue is how to structure the SKE data model to enable effective and efficient data integration and querying for the BoK community, allowing extensions to the data model and content as research topics in the BoK community evolve. We evaluate the SKE data model regarding the effectiveness to answer the queries of EMSE BoK researchers identified in RI-1.

Research Issue RI-3: SKE Tool Support. Which system architecture and functions are necessary to automate key steps in the SKE process, i.e., efficient data integration and querying?

For automating the SKE process a knowledge base (KB) is a major element to provide the desired semantic capabilities. The SKE KB is based on semantic technology with ontologies. Using ontologies makes the KB model extensible and facilitates semantic search [18]. A data integration approach, such as the Interchange Standard Approach [24], allows heterogeneous spreadsheets to be used for data contributions to the KB. As most EMSE researchers are not experts in semantic technologies, an important issue is to separate the KB administration functions from functions for the EMSE BoK community in order to make the interaction of EMSE researchers with the SKE KB simple, effective, and efficient.

Based on the requirements coming from RI-1 and RI-2 and discussions with SLR researchers the user interface for data import should be based on spreadsheet technology and for querying based on web technologies. We evaluate the tool support regarding the effectiveness and efficiency of data import and querying when applying the SKE process.

4. THE SYSTEMATIC KNOWLEDGE ENGINEERING APPROACH

This section presents the SKE approach and its application to build a Software Inspection EMSE BoK KB based on contributions from controlled experiments. The following subsections provide details on how each research issue (requirements analysis, SKE process and data modeling, and SKE tool support) was addressed.

4.1 SKE Requirements Analysis (RI-1)

Figure 2 shows how the SKE approach addresses the challenges posed in Figure 1 by introducing an EMSE BoK knowledge base (KB), and the role of a knowledge engineer. In this context, EMSE BoK researchers extract data from empirical studies published in digital libraries and have that extracted data integrated into the EMSE BoK KB by the knowledge engineer. Therefore, the knowledge collected is then available for semantic querying from the KB also to the general readership, including BoK researchers. Thus, the KB enables a wider community to identify relevant knowledge on BoK topics from empirical studies and reuse it according to their goals. Moreover, SKE considers allowing to correct, discuss, rate and annotate knowledge to the KB even after data extraction by the BoK community through collective intelligence mechanisms [40].

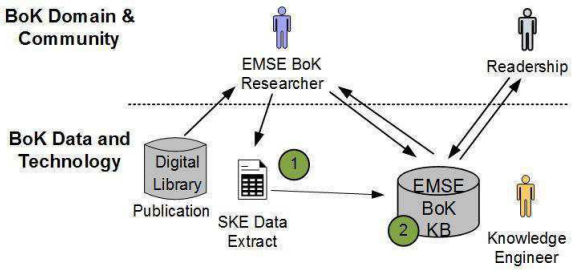

Figure 2. EMSE BoK stakeholders and technology with SKE.

To identify the most relevant query candidates to be answered by the example software inspection EMSE BoK KB, we focused on the EMSE BoK researchers as main stakeholders, who conduct meta-analyses on study reports or conduct empirical studies them-

selves and need to be aware on relevant research in their area. A second stakeholder group is external readers of empirical study reports, both researchers and practitioners, who want to understand the big picture of research relevant for their background.

Based on an informal survey with software inspection EMSE BoK researchers in 6 research groups (located in Vienna, Madrid, Valencia, Ecuador, Chile, and Brazil), we identified a set of query candidates. The selection of most relevant queries was based on a limited budget of value points, which each stakeholder could spend on the query candidates. Then the query candidates were sorted in descending order by the total number of points of each query. Overall, 10 researchers from the 6 research groups contributed to this selection process. The six most relevant stakeholder queries were:

- Q1: Which inspection methods were effective (or efficient) in finding defects in requirements artifacts?
- Q2: What are the results of experiments that report on a given BoK Topic Parameter <BTP>, e.g., inspection method PBR?
- Q3: Which experiments were conducted with the response variable <RV>, e.g., number of defects?
- Q4: Which hypotheses include the domain concept <DC> (and its synonyms), e.g., effectiveness?
- Q5: Which synonyms have been used for domain concept <DC>, e.g., effectiveness?
- Q6: Who are researchers working on topics with response variables in their experiments similar to the domain concept <DC>, e.g., efficiency?

As user interface for querying the EMSE BoK researchers wanted a simple approach with query result sets that are easy to use for further analysis with a range of methods and tools. Therefore, we decided to provide a simple web interface for asking queries, which provides the query results as html text and as spreadsheets, which can then be used as input to a variety of further analyses.

4.2 SKE Process and Data Modeling (RI-2)

Figure 3 compares the phases and stages of the Systematic Literature Review (SLR) [10] and the proposed SKE process. The key innovation of the SKE process comes from decoupling data extraction from data synthesis and integrating extracted data into a KB designed for EMSE BoK building, rather than using the data to apply a particular synthesis method for answering a specific research question in the format of a SLR report (keeping the extracted data from the BoK research community). Thus, the approach allows the community to extend the knowledge gathered during data extraction and reusing it, with the KB's semantic search facilities, as building blocks for a variety of analyses.

From a software ecosystem point of view [9], SKE allows a wider community of researchers, who may even not be familiar with conducting SLRs, to produce research synthesis reports, since they can explore the KB as an input to their analyses and provide feedback to EMSE BoK researchers on their needs. The following subsections detail the SKE process stages and how they can be applied to build a software inspection EMSE BoK KB.

1. Planning EMSE BoK Creation. Similar to conducting a SLR, the first SKE phase starts with identifying the need and commissioning the creation of the EMSE BoK. Since SKE has a predefined purpose of building an EMSE BoK, instead of specifying research questions, SKE just needs to specify the BoK (topics) and the types of empirical studies of interest. In the case of building the inspection EMSE BoK, the BoK was software inspection and the empirical studies of interest were controlled experiments.

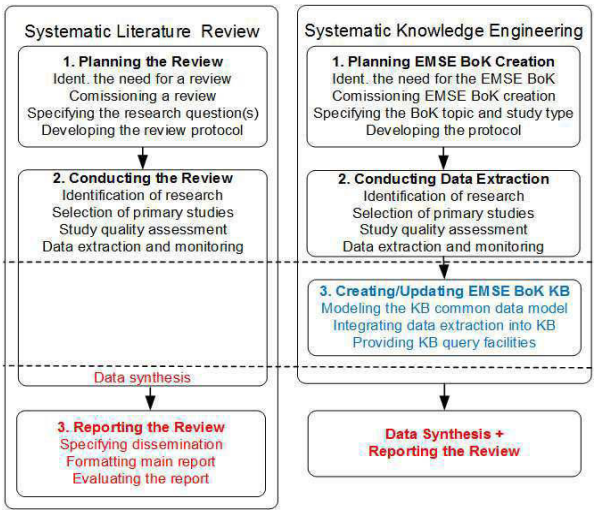

Figure 3. Comparison of the SLR and the SKE processes.

Then, the SKE protocol is built based on a specific configuration of the PICO (population, intervention, comparison, outcome) strategy [16] to derive search strings that can be applied to digital libraries in the "P and I and C and O" format. In this configuration, the population represents the specified BoK or some of its topics, in our case Software Inspection. The intervention represents the specified empirical study types, in our case Controlled Experiments. The comparison is blank and the outcome represents the elements to extract from the empirical studies (e.g., hypotheses, findings), in our case experimental study results.

As in SLRs, the protocol includes the search strategy with the definition of sources of primary studies (e.g., digital libraries), the study selection criteria and procedure, the quality assessment procedures, and the data extraction strategy. In our case a single digital library was chosen: Scopus, which according to [10], claims to be the largest database of abstracts. The study selection and quality assessment criteria were that the study should be an experiment published in a peer-reviewed publication medium.

The search string to be applied on Scopus was derived from the PICO synonyms, adding two specific operators: (i) TITLE-ABS-KEY, avoiding searching in the reference metadata, which would result in false positives; and (ii) W/2, allowing a distance of up to two words between keywords. The resulting search string was:

TITLE-ABS-KEY ((software W/2 (inspection OR "defect detection" OR "reading technique")) AND ("experimental study" OR "experimental evaluation" OR "experiment" OR "empirical study" OR "empirical evaluation") AND ("hypothesis" OR "evidence" OR "finding" OR "result"))

If there is already a BoK KB to build on, the search concepts, including synonyms, can be derived from the BoK KB.

For data extraction, a spreadsheet was prepared to gather relevant experiment data, according to the information to be loaded into the KB common data model on inspection experiments (see Figures 4 to 6). Once the protocol is defined, the next SKE phase, Conducting Data Extraction, can be accomplished.

2. Conducting Data Extraction. This phase consists of following the protocol's search, selection, and assessment strategies for extracting relevant data. In our case, we executed the search string

in Scopus. Overall 156 papers were retrieved. After filtering by title and abstract, a set of 102 papers containing experiments on software inspections were identified, ranging from 1985 to 2013.

A sample consisting of the 30 most recent papers (ranging from 2006 to 2013) was chosen as criteria for data extraction. Six independent local EMSE experts extracted information from those papers into the spreadsheets, with an extra expert acting as a data checker. Data extraction from a paper took on average about 2 person hours. Data checking took additional 0.5 person hours per paper. As data synthesis is in SKE decoupled from data extraction, the goal is integrating the data into a KB so the BoK community can reuse the knowledge. Thus, the next SKE phase concerns creating the EMSE BoK KB.

3. Creating/Updating EMSE BoK KB. In this phase, the knowledge engineer has to design (or update) the KB common data model and then integrate the extracted data. This role is also responsible for providing query facilities. Those facilities allow other researchers to query the KB content and using the results of such queries as input to apply their own research synthesis methods, according to their specific goals.

For building the software inspection EMSE BoK KB, the knowledge engineer initially designed the data model and derived the data extraction spreadsheets to improve the quality and efficiency of integrating the data from the spreadsheets of the six independent data extractors into the inspection EMSE BoK KB. Then the knowledge engineer provided the KB query facilities to address the most relevant stakeholder queries. More details on the KB common data model for hosting information on software inspection experiments follow.

Data Modeling for an EMSE BoK KB. Figure 4 shows an overview on the major areas to be considered in the SKE KB data model in three columns: 1) Researchers who provide publications in the EMSE BoK; 2) the EMSE BoK and the topics that represent the domain concepts and theory in the BoK; and 3) the empirical studies linked to the study data and artifacts. These areas modularize the BoK data model to facilitate data model and content maintenance and reuse.

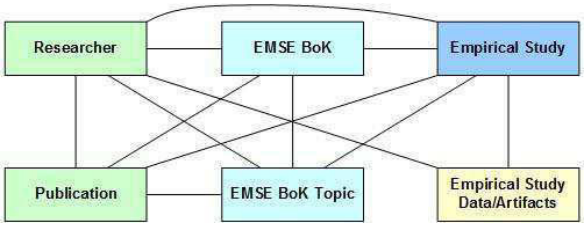

Figure 4. Major areas of the SKE data model.

We used the list of parameters for software inspections by Laitenberger and Debaud [27] (process, inspection method, and artifact) as starting point for the software inspection EMSE BoK topic structure. However, based on an informal overview of knowledge gathered in recent empirical studies on software inspections (see Section 2.4) we found the need to add three additional parameters: inspection activity, model, and tool. To organize the aggregated knowledge in a flexible way and to facilitate future queries, we modeled knowledge with links to topics that represent a combination of such parameters. For instance, knowledge related to requirements inspections (artifact: requirements) applying PBR (inspection method: perspective-based reading/PBR) in the individual preparation (activity: individual preparation) when conducting Fagan inspections (process: Fagan).

Figure 5 illustrates how BoK topics were modeled and linked to empirical study results in order to contribute to build up knowledge in the software inspection EMSE BoK.

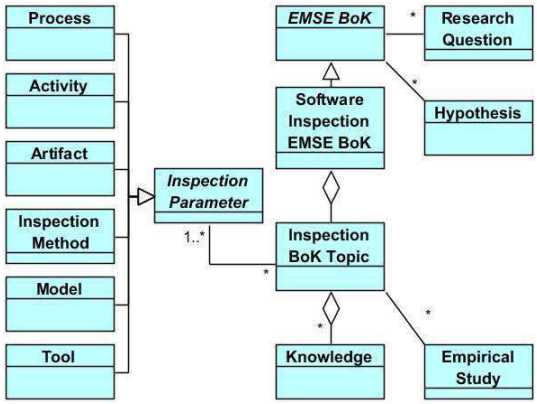

Figure 5. Empirical studies linked to inspection BoK topics to build up knowledge in the EMSE BoK.

Figure 6 shows an overview on the main elements in the data model for building the software inspection EMSE BoK KB. A more detailed data model is available online[5]. The empirical studies (in this case, experiments) are linked to the inspection BoK topic in order to contribute to build up knowledge in the software inspection EMSE BoK. The experiment data itself was modeled to capture the concepts described by Wohlin et al. [3]. For measurement data specific recommendations were followed [39]. Data on the publications and their main findings (based on the experiment results or not) was also included.

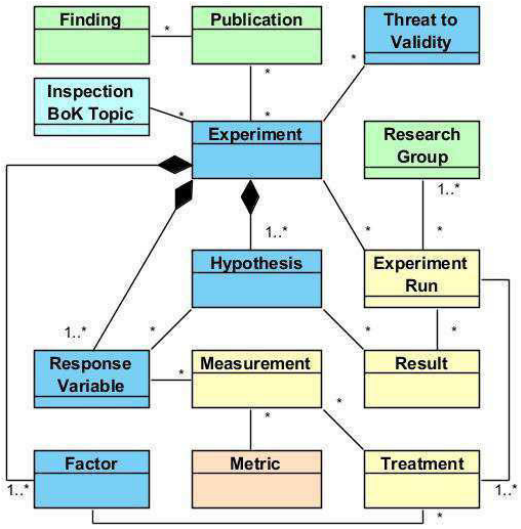

Figure 6. Software inspection EMSE BoK KB common data model, based on [9].

The resulting model allows querying knowledge acquired from experimental studies available on specific inspection BoK topics. For instance, it allows answering queries, such as: "Which hypotheses have been evaluated by experiments on PBR?" In this case, the query would list the hypotheses of all experiments related to BoK topics with parameter "inspection method" equal to PBR (or any of its synonyms). According to specific needs, it is also possible to list the results for each hypothesis in the available experiment run (confirmed/rejected) and information on their

statistical confidence. Moreover, the measurements that led to each of those results can be obtained.

4.3 SKE Tool Support (RI-3)

The knowledge engineer (KE) facilitates the SKE process for non-experts in semantic technology and provides tool support, in particular, for system interfaces and for his own administrative work. Figure 7 shows the EMSE BoK KB and the system interfaces as numbered circles, discussed in the following paragraphs.

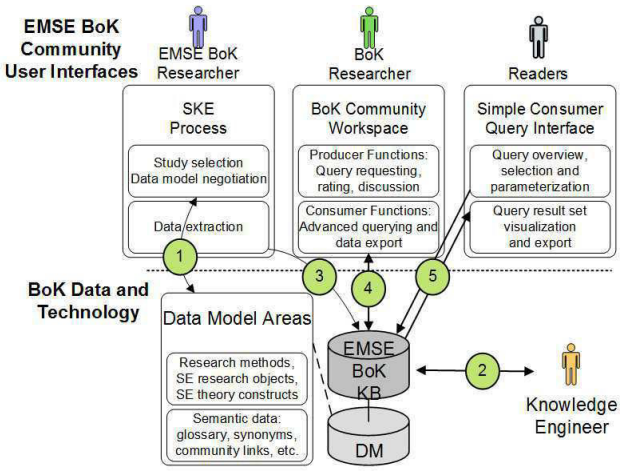

Figure 7. Tool support for SKE process and stakeholders.

1: Data model design. Based on the current data model of the EMSE BoK the KE has to negotiate and agree with the researchers providing data on the data model needs, queries, and necessary data model extensions. We found the Collaborative Design Approach (CDA) [41] useful to guide several stakeholders with heterogeneous data models towards a common data model as basis for data integration.

For the software inspection case, EMSE BoK researchers identified the local specific concepts and the data of each EMSE study that needed to be represented in the data model. Thus, the semantics of the data model were clearly defined so the extracted data could be mapped to the data model. Several workshops were held to facilitate the agreements regarding the data model.

2: KB implementation. The KE created the ontology and queries based on the data model agreed in step 1 with standard tools, such as Protégé[1] and Sparql[2]. Using ontology-specific features, the KE enhanced the KB by facilitating functions related to semantic search and collective intelligence [40].

3: KB population. Based on discussions with SLR researchers, we found that using spreadsheet is the most convenient way for them for data collection and analysis. The KE selected adapted tools for automatically transforming data from most common formats in the EMSE domain into ontologies [42] [43], spreadsheets and relational databases. Options for importing spreadsheets include providing transformations by combining a spreadsheet reader library (e.g., Apache POI[3]) and an ontology library (e.g., Apache Jena[4]).

For the research prototype, we designed spreadsheets as default user interface for data contributions to the KB. For implementing this interface we used Apache Jena and Apache POI.

4: EMSE BoK community workspace. Once there is a substantial EMSE BoK, the EMSE BoK researcher community needs functions to interact with the available knowledge. Producer functions improve the EMSE BoK KB, such as adding links between domain concepts or synonyms. The option of collective intelligence mechanisms [40], such as rating, discussions, and links to external social networks, makes the BoK more valuable as a research tool. Consumer functions provide advanced query options to BoK researchers, who are familiar with semantic technology.

In the research prototype we focus on the population of the SKE KB and provide only initial BoK community functions, such as defining domain concepts in a glossary. Advanced BoK community functions are future work in the scope of this paper.

5: Simple consumer query facilities. The KE provides to non-experts, such as the general readership of a BoK, a simple user interface, which allows selecting queries from a list of most relevant queries, visualizing the query results and exporting the results for analysis. As a simple community function, users can request form the KE new queries, which are not yet provided.

The complete list of software inspection EMSE BoK query candidates (RI-1), the detailed common data model (RI-2) and the EMSE BoK KB prototype (RI-3) are available online[5].

5. EVALUATION RESULTS

This section reports on the evaluation of the results regarding RI-2 and RI-3 based on the required queries coming from RI-1 (see Section 4.1).

5.1 SKE Process Evaluation

For evaluation we applied and measured the SKE process steps with tool support to build a software inspection EMSE BoK KB based on knowledge acquired from controlled experiments. During this evaluation, information was extracted from 30 research papers and integrated into the common KB data model. While the authors of this paper provided the SKE process and data model design, the data extraction was conducted by an independent expert team. The knowledge engineer was one of the authors consulting with external experts on providing the EMSE BoK KB prototype functionality.

SKE Process feasibility. The feasibility of applying each SKE phase (planning, data extraction, and building the EMSE BoK KB) was evaluated. The planned SKE PICO configuration effectively supported the identification of relevant experiments on software inspections. The accuracy of the derived search string was evaluated by comparing its results against the informal software inspection review described in [28], conducted in 2002. The experiments reported in this review and indexed in Scopus were retrieved. Therefore, we see the set of identified papers as representative also for newer inspection experiments.

Data extraction into spreadsheets containing information on software inspection experiments based on the common KB data model was successfully achieved. Thus, the common data model built for software inspection experiments was effective in characteriz-

[1] http://protege.stanford.edu/

[2] http://www.w3.org/TR/rdf-sparql-query/

[3] http://poi.apache.org/

[4] http://jena.apache.org/

[5] http://cdlflex.org/conf/icse14/ske

ing the inspection experiments (model of similarities and variations) and their results.

Finally, concerning EMSE BoK KB creation, it was possible to create the software inspection EMSE BoK KB from published experiment reports. A knowledge engineer conducted the mapping and integration of the data from the local spreadsheets into the common KB data model. In that way, SKE allows building up knowledge incrementally, as researchers can access and reuse past study results and integrate new contributions. Moreover, the resulting EMSE BoK KB was effective and efficient in providing answers to the required queries (further details are provided in the next section).

SKE Process effort. Based on our previous SLR experiences, such as [15], where a series of SLRs was conducted in four consecutive years to build up a BoK on defect causal analysis, we found the overall effort of applying the SKE process comparable to the effort of conducting SLRs (less than 90 person hours). However, the effort of extending the SKE results is likely to be considerably lower (especially if done by other researchers, by allowing directly reusing and extending the extracted data, in the case of [15], as in many, not publicly available). Table 1 shows the effort spent on each of the three SKE phases to build the software inspection EMSE BoK KB and an informal effort comparison to the corresponding SLR phases. Planning takes slightly less effort, since SKE applies a predefined PICO configuration. Although in SKE data extraction from primary studies probably takes somewhat more effort, the overall conduction phase takes about the same effort, since SKE does not apply data synthesis in this phase. Finally, creating the EMSE BoK KB is not considered in SLRs.

Table 1. SKE inspection EMSE BoK KB process effort.

SKE Phase and Effort (person hrs)		Effort Description	SKE vs SLR Effort
Planning EMSE BoK Creation	8	Building the protocol.	↓ (- 10% to -5%)
Conducting Data Extraction	80	Filtering primary studies and extracting data from primary studies.	↔ (-5% to +5%)
Creating EMSE BoK KB (Population)	< 0.05	Integrating data into the KB. This task is automated and the effort is not significant.	N/A

SKE Process added value. We are aware that SKE and SLR processes have different purposes (SLRs usually seek for evidence-based answers to research questions, while SKE focuses on building EMSE BoK KBs) and that one does not replace the other. However, since SLRs are widely used [5], we ascertain that, given the similar effort, it makes sense to apply SKE for building EMSE BoKs. Moreover, in this specific context, when compared to SLRs, SKE presents the following benefits:

- SKE integrates extracted data into a KB to facilitate the reuse of the aggregated knowledge by other researchers according to their specific goals. SLRs usually focus on specific research syntheses and extracted data is mostly stored in local spreadsheets, seldom publicly available.
- SKE facilitates building up knowledge incrementally by integrating new extracted data into the KB. Moreover, SKE, when supported by collective intelligence mechanisms[40], allows correcting, discussing, rating and annotating knowledge to the KB even after data extraction by the BoK community.
- The SKE KB allows exploring the underlying data, derived from analyzing selected publications, using semantic search capabilities that cannot be performed on SLR reports. For instance: "Which results were obtained for hypotheses investigated in BoK topic <BT> using the response variable <RV> (or any of its synonyms)?"

It is important to state that SKE requires a knowledge engineer for data modeling, mapping and providing query facilities to achieve the above listed benefits of establishing an extensible EMSE BoK KB, when compared to building static EMSE BoK reports.

5.2 SKE Data Model Evaluation

Having evaluated the feasibility of applying the SKE process, interest sprouts in evaluating the resulting software inspection EMSE BoK KB and its underlying common data model against the required queries coming from RI-1. Therefore, the knowledge engineer formulated the queries in the KB language so that results for them could be obtained. To evaluate these results, an independent researcher built a set of query test cases based on the 30 papers included in the KB.

SKE Query effectiveness. Figure 8 shows the data model entities and their attributes used to answer the required queries. Based on the data model and the extracted data the KE was able to formulate KB queries to answer the queries Q2 to Q6 (see the list of queries in Section 4.1) for listing experiment results, experiments, hypotheses, synonyms, and research groups. The test cases for Q2 to Q6 were passed successfully with the query results, which can be accessed in the online prototype[5].

However, query Q1, coming from practitioners, was not formulated in terms of data model elements and, therefore, could not be answered directly.

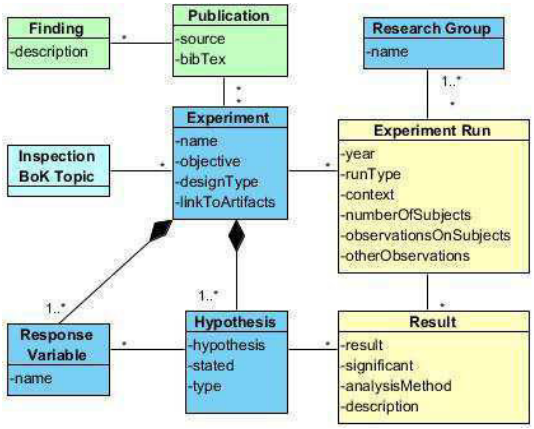

Figure 8. Data model entities used to answer selected queries.

Q1: *"Which inspection methods were effective (or efficient) in finding defects in requirements artifacts?"* needed to be translated by the KE and EMSE BoK experts to the terms of the underlying data model. The first BoK expert decision was to focus on experiments that reported on effectiveness or efficiency (or synonyms) in their hypotheses or response variables and that were related to BoK Topics associated to inspection methods and to the artifact type requirements. For allowing semantic search on synonyms the additional domain concepts KB technology was used. The test case built for this query was related to identifying the right set of experiments and passed successfully.

Then, to answer the query, the KE decided to split the query into two separate queries Q1.1 and Q1.2 that would provide an over-

view to allow getting the knowledge of interest out of those experiments. Q1.1 was to list the hypotheses and their results in all experiment runs conducted on such experiments. Q1.2 was to show the findings of all papers related to them.

Result of Q1.1 are shown in Figure 9. The results enable stakeholders to see which inspection methods were reported as effective. In fact, it was possible to observe, that PBR was more effective than checklist or ad-hoc approaches for requirements inspections in all experiment runs. Moreover, stakeholders are able to see other focused and interesting information. For instance, that defects detected with PBR are more evenly distributed over the document, when compared to using checklists. It was also possible to export the results as input to an external analysis to see to what extent the contexts of the experiments fits to the context of the intended software inspection application. Regarding Q1.2, it provided an insightful overview on the major findings of the papers on effectiveness of inspection methods.

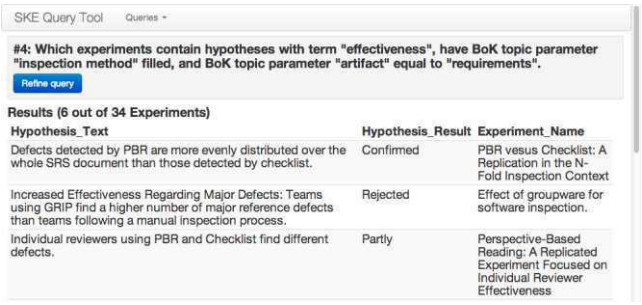

Figure 9. Prototype screenshot answering the query Q1.1.

Therefore, the evaluation of all required queries passed successfully in the scope of the SKE research prototype.

5.3 SKE Tool Support Evaluation

This evaluation concerned the effectiveness and efficiency of the support for data importing and querying.

SKE tool support for data importing. Regarding the provided tool support's architecture and functions (see Section 4.3), in our experience the user interface was effective in enabling contributions from researchers (submitting data extraction sheets, integrated by the KE using a created ontology spreadsheet importer) and efficient by requiring low effort and little time for such contributions to be imported. At all, data from 6 spreadsheets, containing data on 30 experiments and including over 6.000 data elements (cells) were integrated into the KB in less than 3 minutes.

However, it is important to state that SKE required setup effort by the KE for creating the KB's ontology model, creating the spreadsheet importer, providing the query facilities and developing a suitable user interface. Table 2 shows the KE's SKE setup effort.

Table 2. KE Involvement Effort.

KE SKE Setup Effort (person hours)		Effort Description
Creating EMSE BoK KB ontology model	16	Specifying the ontology model based on the common data model.
Spreadsheet Importer Creation	40	Developing the spreadsheet importer tool based on Apache Jena and POI.
Query creation	32	Translating queries into SPARQL.
UI Development	40	Developing a querying user interface.

SKE tool support for querying. A simple web interface prototype for asking queries was developed, allowing stakeholders to easily get query results as html text and as spreadsheets, which can then be used as input for further analyses. As discussed in Section 5.2, the identified relevant stakeholder queries could be effectively answered. Concerning efficiency, processing each query took less than 20 seconds. Although the KB hosted data from only 30 experiments, we believe that efficiency will not be a problem for the proposed architecture and tool support when building up EMSE BoKs from empirical studies.

6. DISCUSSION

In this section we discuss for each research issue the evaluation results, possible limitations, and lessons learned.

6.1 RI-1: SKE Requirements Analysis

Looking at the EMSE BoK community requirements from a software ecosystem point of view [9], the SKE approach and its resulting KB support users well in locating EMSE study reports and reusing their results according to specific needs of different stakeholders, similar to a Semantic Directory [12]. However, it is important to state that support for meta-analyses and for applying specific research syntheses methods are out of the SKE scope. The current SKE scope is also limited to building BoKs from reports of finished EMSE studies, not considering work in progress and raw data, which can be included if access to the study repositories of EMSE research groups becomes available.

Moreover, the relevant queries of our evaluation were chosen focusing on a survey with a specific type of stakeholder, i.e., EMSE BoK researchers. Other stakeholders may have different backgrounds and goals. Jedlitschka et al. [44], for instance, gathered information needs from managers in empirical studies with 175 participants from industry. According to their findings, there are three categories of information needs for these stakeholders from experiment reports: technology, context, and impact.

Based on their work, we believe that the required information needs could be gathered by semantic queries applied to the SKE KB. After all, in the software inspection EMSE BoK KB, technologies are related to the development phase and products (artifacts) on which they are used by the common data model's BoK topics and their parameters (Figure 5). The experimental context is captured in the experiment runs. Finally, the impact of the technology may be identified based on semantic search on the hypotheses that have been evaluated, their response variables and results. However, addressing practitioners' needs has to be evaluated formally in a future study.

6.2 RI-2: SKE Process and Data Modeling

The SKE process evaluation showed the feasibility of building a software inspection EMSE BoK KB from controlled experiments. During the process, the common KB data model was considered effective in characterizing the inspection experiments and their results.

Moreover, the overall effort was comparable to the effort of conducting a SLR and, for the specific purpose of building EMSE BoKs, several advantages of using SKE were identified, such as the extensibility and reuse of knowledge from extracted data for further analyses according to specific stakeholder goals. However, it is important to state that the SKE and SLR processes are complementary, not competing, approaches. The suitability of applying one or the other will depend on the specific research goals. In fact, if an EMSE BoK is built following SKE, the resulting KB may be used as input to an SLR protocol to identify relevant re-

search on a specific topic, exploring semantic search facilities usually not available in digital libraries. SLR data extraction sheets, on the other hand, can be used as input for extending the data model and the knowledge contained in SKE's KB.

6.3 RI-3: SKE Tool Support

Finally, concerning the KB architecture and functions (see Section 4.3), the simple user interface of the online research prototype[5] and the data integration approach enabled researchers to reuse the extracted data, to contribute building up additional knowledge, and to query for knowledge with low effort. Using ontologies facilitates extensions of the underlying KB common data model and semantic search. The querying capabilities were found efficient in the evaluation on answering the EMSE BoK researchers' most relevant queries.

Moreover, SKE considers allowing to correct, discuss, rate and annotate knowledge in the KB by the BoK community even after data extraction through collective intelligence mechanisms [40]. The extensibility and the collective intelligence mechanisms are important elements of the SKE approach, given that for knowledge engineering, modeling is a cyclic process, where new observations may lead to a refinement, modification or completion of the already built-up model [17]. However, collective intelligence mechanisms were not the main focus of the implemented research prototype (an early prototype with still limited user interaction functionality) and need to be explored in future work.

6.4 Threats to Validity and Lessons learned

The performed evaluation faces some threats to validity. A major threat is related to the decision of applying the approach on building a software inspection EMSE BoK based on study reports from controlled experiments. Software inspections are widely spread in academia and industry and many empirical studies have been conducted in this area, which may facilitate building up an EMSE BoK. Additionally, SKE can also be applied for gathering knowledge from other types of empirical studies and such feasibility was not evaluated and needs to be investigated.

A related threat is that data was extracted from the 30 most recent identified experiment reports. For older reports data extraction could be more difficult, since the community may have gained maturity on reporting experiments. Finally, the relevant queries used to evaluate the resulting KB were obtained from an informal survey with 10 EMSE BoK researchers. Based on the BoK research prototype it is now possible to collect requests for queries from a wider set of researchers in the EMSE BoK community.

Main lessons learned are related to effort, and success and risk factors for effectively and efficiently applying SKE. Whereas the effort was comparable to conducting SLRs, we noticed that the data extraction effort was slightly higher and that the role of a data checker (also considered in SLRs) is very important to assure the quality of information loaded into the KB. We assume that the data extraction effort can be reduced by providing proper training on how to extract the information of the common data model into spreadsheets. The effort of the data checker could be reduced by using walkthroughs in which the data extractors present the extracted information from each empirical study report.

A major success factor is properly involving a knowledge engineer for data mapping and integration and for providing query facilities. The overhead of this new role is likely to be offset by the benefits obtained by the established KB. An identified risk factor, on the other hand, is not having the EMSE BoK community involved for long-term collection and use of data. To mitigate this risk, incentives could be provided and the benefits of contributing should be clear, similar to other community networks.

7. CONCLUSIONS

In this paper we introduced the Systematic Knowledge Engineering (SKE) approach, which supports building up EMSE BoKs from empirical studies. The SKE process (see Section 4.2) builds on parts of the Systematic Literature Review (SLR) process and provides a Knowledge Base (KB) as storage for extracted data. By decoupling data extraction from data synthesis and providing a KB, SKE allows the community to extend gathered knowledge and reusing it with semantic search facilities, as building blocks for a variety of analyses. SKE also considers collective intelligence mechanisms to make the knowledge more useful to the BoK community.

The SKE process was evaluated in the application context of building a software inspection EMSE BoK KB from knowledge acquired through empirical study reports on controlled experiments. During this evaluation, information was extracted from 30 research papers and integrated into the KB. The resulting software inspection EMSE BoK KB prototype is available online[5]. Main evaluation results were:

- SKE's suggested PICO configuration supported identifying relevant experiments on software inspections.
- Data extraction into spreadsheets containing information on software inspection experiments based on the common data model was successfully achieved.
- It was possible to create the software inspection EMSE BoK KB from published experiment reports.
- The KB was effective and efficient in answering stakeholder queries. Additionally, it allows answering queries that, in general, cannot be answered based on SLR reports.
- The prototype application indicated that SKE enables knowledge reuse (by applying queries) for analysis and meta-analysis purposes. Moreover, new knowledge, i.e., new data from literature, can be included in the KB as a foundation for a growing EMSE BoK KB.

The SKE approach showed promising results in the software inspection context and should also be evaluated in other contexts. The overall effort of applying SKE is comparable to the effort of conducting SLRs and, for the specific purpose of building EMSE BoKs, several advantages of using SKE could be identified.

The long-term goal of this research is to support efficiently building EMSE BoKs. We propose the following future work. 1. Investigate the extent to which different stakeholder needs can be satisfied by querying an EMSE BoK KB. 2. Integrate a more complete set of empirical studies on software inspections into the BoK KB to enable an overview on the BoK theory contributions. 3. Instantiate and evaluate SKE in other BoK contexts. 4. We plan to set up a platform to allow building on the collective intelligence of the EMSE BoK community in the context of an EMSE BoK KB.

ACKNOWLEDGMENTS

This work was supported by the Christian Doppler Forschungsgesellschaft, the BMWFJ - Austria, and CNPq - Brazil. We thank the researchers who participated in the survey, the EMSE experts responsible for data extraction, and Angelika and Jürgen Musil for their support in providing the glossary functionality.

REFERENCES

[1] B. Boehm and V. R. Basili, "Software Defect Reduction Top 10 List," IEEE Computer, vol. 34, no. 1, pp. 135–137, 2001.

[2] D. I. K. Sjøberg, T. Dybå, B. C. D. Anda, and J. E. Hannay, "Building Theories in Software Engineering," in in Guide to Advanced Empirical Software Engineering SE - 12, F. Shull, J. Singer, and D. K. Sjøberg, Eds. Springer London, 2008, pp. 312–336.

[3] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, "Experimentation in Software Engineering," Springer, 2012.

[4] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," Journal of Systems and Software, vol. 80, no. 4, pp. 571–583, Apr. 2007.

[5] S. MacDonell, M. Shepperd, B. Kitchenham, and E. Mendes, "How Reliable Are Systematic Reviews in Empirical Software Engineering?," IEEE Transactions on Software Engineering, vol. 36, no. 5, pp. 676–687, Sep. 2010.

[6] D. S. Cruzes and T. Dybå, "Research synthesis in software engineering: A tertiary study," Information and Software Technology, vol. 53, no. 5, pp. 440–455, May 2011.

[7] S. Biffl, E. Serral, D. Winkler, O. Dieste, N. Juristo, and N. Condori-Fernández, "Replication Data Management: Needs and Solutions - An evaluation of conceptual approaches for integrating heterogeneous replication study data," in 2013 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement - *ESEM '13 (accepted for publication)*, 2013.

[8] F. Shull, M. G. Mendoncça, V. Basili, J. Carver, J. C. Maldonado, S. Fabbri, G. H. Travassos, and M. C. Ferreira, "Knowledge-Sharing Issues in Experimental Software Engineering," Empirical Software Engineering, vol. 9, no. 1/2, pp. 111–137, Mar. 2004.

[9] S. Biffl, E. Serral, J. Musil, D. Winkler, O. Dieste, E. Fonseca, and N. Juristo, "An Ecosystem View on Empirical Software Engineering Research," Journal on Information and Software Technology (in review), no. Special Issue on Software Ecosystems, 2013.

[10] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Review in Software Engineering," Keele University Technical Report - EBSE-2007-01, 2007.

[11] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering – A systematic literature review," Information and Software Technology, vol. 51, no. 1, pp. 7–15, Jan. 2009.

[12] S. Biffl, E. Serral, J. Musil, and D. Winkler, "A Semantic Directory for Empirical Software Engineering Research," in Proc. of Euromicro 2013 SEAA Work in Progress (submitted)., 2013.

[13] B. Boehm and V. Basili, "The cebase framework for strategic software development and evolution," in Third International Workshop on Economics-Driven Software Engineering Research (EDSER-3 2001), 2001.

[14] B. Hofmann and V. Wulf, "Building Communities among software engineers: the VISEK approach to intra- and inter-organizational learning," Advances in Learning Software Organizations, pp. 25–33, 2003.

[15] M. Kalinowski, D. N. Card, and G. H. Travassos, "Evidence-Based Guidelines to Defect Causal Analysis," IEEE Software, vol. 29, no. 4, pp. 16–18, Jul. 2012.

[16] M. Pai, M. McCulloch, J. D. Gorman, N. Pai, W. Enanoria, G. Kennedy, P. Tharyan, and J. M. Colford, "Systematic reviews and meta-analyses: an illustrated, step-by-step guide.," The National medical journal of India, vol. 17, no. 2, pp. 86–95, 2004.

[17] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," Data & Knowledge Engineering, vol. 25, no. 1–2, pp. 161–197, Mar. 1998.

[18] J. Ye, L. Coyle, S. Dobson, and P. Nixon, "Ontology-based models in pervasive computing systems," The Knowledge Engineering Review, vol. 22, no. 04, pp. 315–347, Dec. 2007.

[19] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing?," International Journal of Human-Computer Studies, vol. 43, no. 5–6, pp. 907–928, Nov. 1995.

[20] O. Corcho, M. Fernández-López, and A. Gómez-Pérez, "Methodologies, tools and languages for building ontologies. Where is their meeting point?," Data & Knowledge Engineering, vol. 46, no. 1, pp. 41–64, Jul. 2003.

[21] N. K. Y. Leung, S. K. Lau, J. Fan, and N. Tsang, "An integration-oriented ontology development methodology to reuse existing ontologies in an ontology development process," in Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services - iiWAS *'11*, 2011, p. 174.

[22] C. W. Holsapple and K. D. Joshi, "A collaborative approach to ontology design," Communications of the ACM, vol. 45, no. 2, pp. 42–47, Feb. 2002.

[23] S. Bergamaschi, S. Castano, and M. Vincini, "Semantic Integration of Semistructured and Structured Data Sources," SIGMOD Record, vol. 28, no. 1, pp. 54–59, 1999.

[24] N. F. Noy, "Semantic integration," ACM SIGMOD Record, vol. 33, no. 4, p. 65, Dec. 2004.

[25] M. E. Fagan, "Design and code inspections to reduce errors in program development," IBM Systems Journal, vol. 15, no. 3, pp. 182–211, 1976.

[26] A. Abran, J. Moore, P. Bourque, R. L. Dupuis, and L. Tripp, Software Engineering Body of Knowledge– SWEBOK, trial version. IEEE-Computer Society Press, 2001.

[27] O. Laitenberger and J.-M. DeBaud, "An encompassing life cycle centric survey of software inspection," Journal

of Systems and Software, vol. 50, no. 1, pp. 5–31, Jan. 2000.

[28] A. Aurum, H. Petersson, and C. Wohlin, “State-of-the-art: software inspections after 25 years,” Software Testing, Verification and Reliability, vol. 12, no. 3, pp. 133–154, Sep. 2002.

[29] L. G. Votta, “Does every inspection need a meeting?,” ACM SIGSOFT Software Engineering Notes, vol. 18, no. 5, pp. 107–114, Dec. 1993.

[30] P. M. Johnson and D. Tjahjono, “Assessing software review meetings,” in Proceedings of the 19th international conference on Software engineering - ICSE *’97*, 1997, pp. 118–127.

[31] J. C. Maldonado, J. Carver, F. Shull, S. Fabbri, E. Dória, L. Martimiano, M. Mendonça, and V. Basili, “Perspective-Based Reading: A Replicated Experiment Focused on Individual Reviewer Effectiveness,” Empirical Software Engineering, vol. 11, no. 1, pp. 119–142, Feb. 2006.

[32] F. Shull, I. Rus, and V. Basili, “How perspective-based reading can improve requirements inspections,” Computer, vol. 33, no. 7, pp. 73–79, Jul. 2000.

[33] D. Winkler, S. Biffl, and K. Faderl, “Investigating the Temporal Behavior of Defect Detection in Software Inspection and Inspection-Based Testing,” in Product-Focused Software Process Improvement, 2010, pp. 17–31.

[34] S. Biffl, B. Freimut, and O. Laitenberger, “Investigating the cost-effectiveness of reinspections in software development,” in International Conference on Software Engineering ICSE´01, 2001, pp. 155–164.

[35] G. S. Walia and J. C. Carver, “Evaluation of capture-recapture models for estimating the abundance of naturally-occurring defects,” in Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM *’08*, 2008, p. 158.

[36] M. Kalinowski and G. H. Travassos, “A computational framework for supporting software inspections,” in Proceedings. 19th International Conference on Automated Software Engineering, 2004., 2004, pp. 46–55.

[37] F. Lanubile, T. Mallardo, and F. Calefato, “Tool support for geographically dispersed inspection teams,” Software Process: Improvement and Practice, vol. 8, no. 4, pp. 217–231, Oct. 2003.

[38] S. Biffl, P. Grünbacher, and M. Halling, “A family of experiments to investigate the effects of groupware for software inspection,” Journal of Automated Software Engineering, vol. 13, no. 3, pp. 373–394, Jul. 2006.

[39] B. A. Kitchenham, R. T. Hughes, and S. G. Linkman, “Modeling software measurement data,” IEEE Transactions on Software Engineering, vol. 27, no. 9, pp. 788–804, 2001.

[40] J. Musil, A. Musil, and S. Biffl, “Elements of Software Ecosystem Early-Stage Design for Collective Intelligence Systems,” in *1st Int’l Workshop on Software Ecosystem* Architectures. Colocated with the 9th Joint Meeting of ESEC/FSE, 2013.

[41] C. W. Holsapple and K. D. Joshi, “A collaborative approach to ontology design,” Communications of the ACM, vol. 45, no. 2, pp. 42–47, Feb. 2002.

[42] A. Grünwald and T. Moser, “umlTUowl - A Both Generic and Vendor-specific Approach for UML to OWL Transformation,” in SEKE 2012, 2012, pp. 730–736.

[43] O. Kovalenko, E. Serral, and S. Biffl, “Towards Evaluation and Comparison of Tools for Ontology Population from Spreadsheet Data,” in I-SEMANTICS 2013, 9th Int. Conf. on Semantic Systems (accepted for publication), 2013.

[44] A. Jedlitschka, N. Juristo, and D. Rombach, “Reporting experiments to satisfy professionals’ information needs,” Empirical Software Engineering, Aug. 2013.