Chapter 13
Building Theories from Multiple Evidence Sources

Forrest Shull and Raimund L. Feldmann

Abstract As emphasized in other chapters of this book, useful results in empirical software engineering require a variety of data to be collected through different studies – focusing on a single context or single metric rarely tells a useful story. But, in each study, the requirements of the local context are liable to impose different constraints on study design, the metrics to be collected, and other factors. Thus, even when all the studies focus on the same phenomenon (say, software quality), such studies can validly collect a number of different measures that are not at all compatible (say, number of defects required to be fixed during development, number of problem reports received from the customer, total amount of effort that needed to be spent on rework). Can anything be done to build a useful body of knowledge from these disparate pieces?

This chapter addresses strategies that have been applied to date to draw conclusions from across such varied but valid data sets. Key approaches are compared and the data to which they are best suited are identified. Our analysis together with associated lessons learned provide decision support for readers interested in choosing and using such approaches to build up useful theories.

1. Introduction

Research in software engineering is often concerned with the development of new techniques, methods, or tools for software development. It has long been recognized that the weaknesses and benefits of such technologies can be identified by conducting empirical studies (Basili et al, 1986, 1999). Empirical information is necessary for researchers to refine the technologies, as well as for practitioners to understand when such technologies are likely to be useful. Empirical evidence can never *prove* that a technology will be useful under specific conditions, but such evidence helps build theories to that effect. The more evidence that can be accumulated, and the greater the extent to which the evidence is internally consistent, the more confidence can be had in the theories they support.

F. Shull et al. (eds.), *Guide to Advanced Empirical Software Engineering*.

The chapter by Sjøberg et al. (Sjøberg, 2007a) in this book discusses the difficulty of providing a precise definition of what a "theory" is. However, to avoid misconceptions, we adopt their convention of focusing on empirically-based theories, which are built on the basis of empirical research to offer explanations of why certain phenomena occur. We also adopt their criteria in saying that a good theory is constructed in such a way as to be testable; is supported by evidence, perhaps in the form of empirical studies; has explanatory power; contains the minimum number of concepts and prepositions; is independent of specific settings; and has relevance to the software industry. In accordance with Zelkowitz, we define empirical studies as a general form of research strategy that relies on analysis of the results of application in some context (Zelkowitz, 2001). Empirical studies include for example controlled experiments, case studies, and archival analyses.

Although a theory represents a proposed model of reality, these need not be formal models. An example of a theory that aims to support decision-making by practitioners might be, "When process conformance is good, software formal inspections will find and remove between 60% and 90% of the extant defects in an artifact, under typical conditions in many environments." Theories may also build implicit models by hypothesizing relationships between variables, such as "When applied by very small teams, the cost to apply software formal inspections may be prohibitive."

A single empirical study is a first step towards constructing theories related to the effectiveness of a technique, method, or tool. However, such single studies usually have a low power. The findings become more reliable (and we have greater confidence in the theories they support) if studies are replicated (i.e., are repeated or conducted in different settings). Similar findings in replications increase the confidence in the results. Multiple authors (e.g., Basili, 1999; Miller, 2000; Kitchenham et al, 2004) point out that it is necessary to accumulate the material of many studies to abstract robust and useful theories.

Based on our experiences, we define a "useful" theory as one which satisfies these criteria: (1) There must be traceability to the supporting data, such that a level of confidence is enabled. To have high confidence, there must be a rigorous way of showing which sources of evidence support a theory. (2) The theory must be abstract enough to be useful (i.e., it cannot hold only under certain unusual or unrealistic conditions, but it has to be relevant for some subset of software development projects).

Building theories is difficult, mainly because solid theories need to be supported by a significant body of evidence. But evidence is generated from many different environments, for many different reasons, and there are no universal standards for how to measure aspects of software development. For example, a researcher might want to theorize that a particular practice helps improve software quality. Supporting or extending this theory becomes difficult when some of the evidence on which it is based measures software quality in terms of customer satisfaction, some in terms of number of defects found after delivery, and some using the number of defects removed from work artifacts.

A number of techniques have been applied to accumulate bodies of knowledge and support theories based on them. The techniques range from informal, subjective, and unrigorous to formal, objective, and rigorous. In this chapter, we describe

three such techniques and summarize the process for applying them. Since research techniques, just like development techniques, work well in some contexts and for some goals but not for all, we also assess all of the techniques along a standard set of dimensions to help understand the problems and conditions for which each is most appropriate.

2. Theory Building

After the more general introduction to the problem in the last section, we now take a closer look at the different tasks that need to be accomplished in order to build a useful body of knowledge. First, we will introduce a general process description of how theories can be built using available quantitative and qualitative evidence (Subsection 2.1). Based on these general process steps we will compare and contrast various existing approaches in the following sections. Second, we will identify and discuss a set of quality attributes for a body of knowledge (Subsection 2.2). This set of attributes will allow us to better classify the existing approaches.

2.1. A Process Model for Building Theories

Several approaches exist for how to build a body of knowledge out of discrete pieces of evidence. These approaches vary in specific details, such as the type of evidence considered for the evaluation, or in the way of handling different evidence pieces. However, all approaches need to integrate some essential process steps to be repeatable and systematic: (1) Define the topic, (2) identify search parameters, (3) find evidence, (4) analyze evidence, and (5) integrate evidence. Fig.1 displays how these steps are connected and emphasizes the iterative nature of the process.

For describing the process steps, and the basic activities associated with them, we will use the following schema:
Step number/name: Clearly identifies the process step.
Input: Lists products and preconditions needed to execute the process step.
Actions: Describes the basic activities performed in this process step.
Output: Identifies the products generated by the process and post conditions.
Comments: Provides a practical example of what needs to be done in this step or lists typical issues.

2.1.1. Define Topic

Before we can start collecting any evidence for a theory, the topic of the theory we want to describe needs to be defined. In this first step one has to clearly identify the object(s) that will be described by the theory. Ideally, this description not only

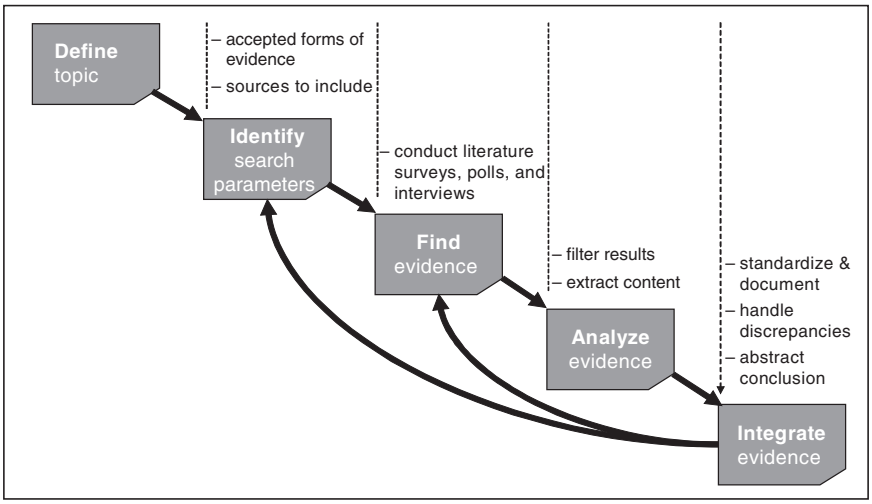

Fig. 1 Basic process steps for building a theory based on multiple pieces of evidence

identifies the topic(s) but also provides the basic definitions for key terms and concepts. Quality attributes, i.e., specific aspects of the object we are interested in, need to be included in this definition process, too. Examples for such quality attributes could be the effectiveness of the object regarding cost or time reduction.

This process step can be triggered for several reasons. Typical examples are the need for decision support on a given topic, or the interest of a researcher to identify missing studies in a certain field. As a result of this step we create a *Theory Topic Definition Document (TTDD)*, which will be the input and basic reference for the following process steps.

For formulating the goal in a more formal manner one might consider a specific template or other structured approaches. The Goal Question Metric (GQM) approach (Basili, 1994b; van Solingen and Berghout, 1999), for instance, provides a specific goal template for describing measurement goals. We have found the GQM goal template, as depicted in Fig.2 useful for helping to specify fairly straightforward theories, since it helps make explicit the object that is being theorized about as well as the properties of interest. Templates that are more comprehensive, for instance, have been proposed in Sjøberg (2007b).

Note that some researchers in the social sciences recommend mapping studies, prior to performing systematic review, in order to identify patterns in the research literature and identify areas suitable for systematic literature review or meta analysis or where more primary studies are needed (Petticrew and Roberts, 2006). This activity, however, may be most relevant under certain conditions or study topics.

Analyze the	[*object*]
for the purpose of	[*purpose*]
with respect to	[*quality aspect*]
from the perspective of	[*view point*]
in the context of	[*context*]

Fig. 2 GQM goal template according to Basili et al

Table 1 Overview of process step 1

Step number and name	❶ Define topic
Input	This process step can be started at any time; no specific input documents are required
Actions	Clearly describe the theory to be developed; provide basic definitions and include quality attributes
Output	Theory Topic Definition Document (TTDD)
Comments	Definitions may not be necessary if the relevant terms are commonly known

In conclusion, the first process step is summarized by using our schema in Table 1.

2.1.2. Identify Search Parameters

Using the concrete topic definition in the TTDD, the next process step in building a theory focuses on the search parameters for finding evidence. The evidence will be the basis for our body of knowledge. Hence, it is crucial to (a) clearly identify acceptable forms of evidence, and (b) describe how we will proceed to find the evidence.

By determining the forms of acceptable evidence it is indirectly determined how rigorous the overall process of building the body of knowledge will be. If, for instance, only the most significant and best documented empirical results will be considered, a highly rigorous process is most likely. The overall rigor becomes more relaxed if, for instance, qualitative evidence such as lessons learned is included.

Possible forms of evidence include: A rigorous empirical study with a comparison of the object under study to other existing practices, a controlled experiment in a research environment, an industrial case study, literature surveys, a qualitative statement of lessons learned, a poll, or even a single person's opinion captured in a white paper or interview. A good overview and classification of possible empirical evidence can be found in (Zelkowitz and Wallace, 1998).

Along with the types of accepted evidence goes the definition of accepted (i.e., trusted) sources for such evidence. Such sources can range from books and archival

journals, where each piece of evidence is peer reviewed, to purely electronic sources on the Internet, which may include promotional material of technology vendors or companies.

Last but not least, we have to identify the possible search process we will use to find the evidence. This is in part connected to the list of accepted sources. For instance, evidence in journals can be found by searching specific internet catalogues of such journals (e.g., IEEE Computer Society Digital Library[1] or The ACM Digital Library[2]) or by a classical library search. A search for evidence on the Internet offers even more possibilities: Which search engines are going to be used? What keywords will be entered? How are the results filtered? In any case, it is necessary to document the intended (and later applied) search process and routines so it becomes obvious and repeatable for others.

As outcome products of our second process step, we generate a *List of Accepted Forms of Evidence (LAFE)*, a *List of Accepted (i.e., trusted) Sources for the Evidence (LASE)*, and a *Search Process Definition (SPD)*. All of these results can consist of separate documents, or can even be included in a single document. They even might be added to the TTDD. However, for our generic process we assume that each document will be handled separately.

We summarize the second process step by using our schema, in Table 2.

2.1.3. Find Evidence

While the first two process steps have been more concerned with the theoretical foundation of the theory building, this third process step marks the start of the practical work. The pieces of evidence for our body of knowledge are retrieved. Therefore, the search is executed as documented in the Search Process Definition (SPD).

Table 2 Overview of process step 2

Step number and name	❷ Identify search parameters
Input	Process ❶ needs to be terminated; complete Theory Topic Definition Document (TTDD)
Actions	Identify and list the accepted forms of (empirical) evidence. Provide an initial list of acceptable sources for the evidence. Describe the search process that will be applied
Output	List of Accepted Forms of Evidence (LAFE); List of Accepted Sources of Evidence (LASE); Search Process Definition (SPD)
Comments	Typically, this process step is executed in an iterative way. As soon as some of the produced documents exist, they may be evaluated and fine-tuned in the following process steps

[1] On-line at http://www.computer.org/portal/site/csdl/index.jsp

[2] On-line at http://portal.acm.org

Table 3 Overview of process step 3

Step number and name	❸ Find evidence
Input	Process step ❷ must have been started; initial versions of LAFE and LASE exist, and SPD has been created
Actions	Execute SPD; conduct literature surveys, polls, and/or interviews
Output	Collection of Retrieved Evidence for Theory (CRET)
Comments	For documentation purposes, the CET should include all of the retrieved evidence pieces that match the LAFE and LASE criteria. A filtering of these results will be conducted in the next process step ❹

This process step could include such activities as performing a literature survey, conducting specific polls, or holding interviews with practitioners and experts. All of the retrieved evidence should be documented in a *Collection of Retrieved Evidence for Theory (CRET)*. This step is summarized in Table 3.

2.1.4. Analyze Evidence

In this process step the potential evidence pieces in the CRET will be analyzed. Therefore, one first has to take a look at the CRET and define the process for the analysis. The process for analyzing the evidence has not necessarily been defined before (e.g., in step 2) because it may be dependant on the evidence itself (e.g., its quantity, quality, completeness, etc.). One also may have to further filter the CRET and prepare the single evidence pieces for the analysis. As part of the analysis activities the content of each evidence piece is extracted and prepared for the inclusion into the body of knowledge. This extraction is based on the defined quality attributes of the TTDD. Specific analysis methods will be discussed in the later sections of this chapter.

As results of this process step one creates a *Documentation of Chosen Analysis Process (DCAP)* and the *Analyzed Evidence for Theory (AET)*. See Table 4 for a summary of these actions and output.

2.1.5. Integrate Evidence

In this last step of the general process for building theories, summarized in Table 5, the actual body of knowledge is described and documented. This includes the clear identification and representation of all found and accepted pieces of evidence, the handling of possible discrepancies in these different evidence pieces, as well as an abstraction from the single evidence pieces. As a result of this final process step we create a *Structured Body of Knowledge (SBK)*.

To create the SBK several activities have to be performed. First of all the basis for the SBK needs to be documented. This may be simply done by referring to the AET or by integrating the AET evidence pieces into a specific data structure or

Table 4 Overview of process step 4

Step number and name	❹ Analyze evidence
Input	This process step can be started as soon as the first pieces of evidence are added to the CRET. TTDD is used as a basis
Actions	Define suitable process for analyzing the CRET Filter and prepare results from CRET according to process Extract content from evidence based on defined process
Output	Documentation of Chosen Analysis Process (DCAP) Analyzed Evidence for Theory (AET)
Comments	In the general process for building theories we include the DCAP in the analysis step. However, specific process may choose to perform this considerations already as part of the earlier process steps (e.g., step ❶ or ❷)

Table 5 Overview of process step 5

Step number and name	❺ Integrate evidence
Input	This process step can be started as soon as the DCAP is existent and the first pieces of evidence are available in the AET documentation
Actions	Standardize and make evidence available to users Identify and handle discrepancies in the evidence set Create an abstraction that integrates all evidence pieces into a transparent summary
Output	Structured Body of Knowledge (SBK)
Comments	If not enough evidence is available for this process step, it might be considered to redefine the search parameters (step ❷) or repeat the search step ❸

knowledge management system. Ideally, all evidence pieces have similar tendencies or the same findings regarding the quality attributes under study. In this case it is relatively easy to integrate all pieces of evidence into an abstraction. The abstraction is a transparent conclusion that summarizes the findings of all evidence pieces regarding the theory and the quality attributes under evaluation. This abstraction allows users to get a quick overview of the body of knowledge without having to take a look at all evidence pieces. Specific methods for accomplishing this combination and extraction of evidence will be discussed later in this chapter.

Regardless of which integration method is chosen, one important goal is that contradictory findings in the AET are clearly reflected in the final output. For instance, the results of the process so far may show that for seven out of nine pieces of evidence there are clear results that a technology reduces costs. But in the two other pieces of evidence it is reported that there has been no cost reduction or, even worse, that the cost has been increased. This inconsistency needs to be reflected somehow in the abstraction of the body of knowledge.

In analyzing these inconsistencies, it is important to note whether the evidence suggests that certain factors might be responsible for the different results.

For example, if the seven pieces of evidence, which support the idea that the technology reduces costs all come from large projects, and the contradictory evidence comes from small projects, then it is possible to hypothesize that project size influences the effectiveness of the technology. It is important to note that influencing factors may be attributes of the studies as well as attributes of the project; for example, analysts might notice that beneficial effects are seen only in the studies of one researcher and are missing in independent replications.

2.2. Quality Attributes for Classifying Theories

Before we take a detailed look at how different approaches instantiate the general process steps, we introduce some quality attributes that apply to theory building approaches. These quality attributes can be used to:

1. Characterize the specific aspects of a given theory building approach
2. Classify and compare the different theory building approaches so as to select the most suitable

Based on our experiences with decision support and technology transfer, we choose the following eight quality attributes as most relevant to robust and useful theories: (1) Applicability for qualitative data, (2) applicability for quantitative data, (3) scalability, (4) objectivity, (5) fairness, (6) ease of use, (7) openness, and (8) cost.

Since we are only intending to give tendencies on how these quality attributes are met by different approaches to theory building, we will rate each approach for each attribute as either: +, ±, or −. In this scheme a + indicates that the given approach can produce output that is rated well for this attribute, while a – definitely indicates that the approach is not well suited for users to whom this attribute is important. A ± is used in the case where no clear tendencies can be identified.

2.2.1. Applicability for Quantitative Data

This attribute indicates whether or not an approach makes use of quantitative data such as numeric measures of cost, quality, or schedule impact. Approaches that explicitly do not include such information will be indicated by a −, while others which explicitly include them will be indicated by a +.

2.2.2. Applicability for Qualitative Data

This attribute indicates whether or not an approach makes use of qualitative data such as lessons learned, whitepapers, or expert statements and interviews. Some approaches explicitly do not include such information (which will be indicated by a −) while others explicitly include them (indicated by a +).

2.2.3. Scalability

This attribute addresses the question of how easy or hard it is likely to be to find evidence that matches the constraints of the theory-building approach. That is, given the current state of the software engineering literature, does the approach scale up in that it can use a large set of publications as evidence, or is it limited to only a small subset? Obviously this will depend on the particular theory and the desired rigor of the analysis; however, this criterion attempts to give a (subjective) rating of, on balance, how many evidence sources in the software engineering domain will be found that are suitable inputs. A – indicates the approach is defined in such a way that suitable evidence sources will be difficult to find, while a + indicates the approach is designed to be more inclusive.

2.2.4. Objectivity

This attribute expresses how objective the approach is in handling the evidence. It describes the extent to which subjective influences of the person(s) executing the process are excluded. The more objective a process, the more deterministic its output becomes. Hence, this attribute indirectly captures the extent to which the process is repeatable. A + indicates the absence of subjective influences, while a – indicates the potential presence of such influences. A ± is used in the case where no determination can be made.

2.2.5. Fairness

This attribute describes the lack of bias in an approach. While objectivity describes whether repeatable conclusions will be drawn from a given set of evidence, fairness describes whether an approach will collect an appropriate set of evidence on which to base conclusions. Approaches with no bias will be marked with a + while a – indicates that the approach has the potential to include some bias.

2.2.6. Ease of Use

This attribute describes how easily the results can be accessed from a user's perspective. Are results clearly understandable by everyone, or does one need specific knowledge, for example about a domain, to interpret them? We rate outcomes that require no additional knowledge with a + while others which require highly specialized knowledge are rated with –.

2.2.7. Openness

This attribute describes how open the process steps are for the user. Can interested outside parties understand how the results were created? Are intermediate results available so that various process steps can be re-applied by outsiders and the results

checked? Approaches which are explicitly open for users are rated with a + while a – indicates approaches that operate as more of a black-box (end users are guaranteed only to see the inputs and outputs).

2.2.8. Cost

This is the last but definitely not the least important attribute in our list. "Cost" expresses the level of time and effort investment necessary to get results. Regardless of the benefits that can be achieved, some approaches may require substantial work to produce and document the results. In such cases we clearly flag them with a – while approaches with a + have exactly the opposite meaning, namely they are relatively cheap to apply.

3. Approaches to Theory-Building

Given the multiplicity of evidence types in the software engineering literature, it should not be surprising that multiple approaches have been applied to make sense of this information. It is important to note that the software engineering literature should be viewed as being stronger, not weaker, because it incorporates such a wide variety of types of evidence, ranging from a single expert's opinion, to aggregated opinions of multiple experts, to anecdotal case studies, to rigorously measured data from across dozens or hundreds of projects. However, this very disparity makes it hard to aggregate well-supported theories and marshal the supporting evidence in a way that is commonly accepted.

In this section, we introduce several approaches that have been proposed to rigorously and repeatably abstract well-formed theories from such data sets. Each is mapped to the general process described in the last section so as to facilitate comparison.

3.1. Systematic Literature Review

The approach to theory- and knowledge-building which has garnered the most attention recently is the systematic review. The systematic review can be defined as "a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest" (Kitchenham, 2004). It is in short a way to summarize across multiple studies on a given topic what conclusions can be drawn. Note the emphasis on completeness in the above definition ("…all available research…"), which is a major goal of the technique. By taking a highly procedural approach to defining the problem of study and searching the available literature, the technique aims to avoid the danger of selection bias, in which only a subset of studies are canvassed (which just might

happen to be the subset that corresponds to a particular point of view). Systematic review was a key method proposed to support the goal of evidence-based software engineering, as articulated by Kitchenham et al. (2004).

The application of systematic review to software engineering was inspired by its success in the medical field, a domain in which researchers must also abstract actionable theories and conclusions from among many studies of the same phenomenon.

Procedure. The procedure for the systematic review is described in detail in a technical report compiled by Kitchenham (2004). The major activities, as mapped to our generic process description, are described below, and have been summarized from that source unless otherwise noted. Kitchenham does note that the process is likely to be highly iterative, with many transitions backwards and forwards among the following activities. An important part of this procedure is to document the planned activities for conducting the systematic review as a protocol, to facilitate the review of the plan and ensure that decisions are made so as to support a review that is as repeatable and rigorous as possible.

- *Define topic*. The guidelines state that the process should start from a well-defined question, in which the population, intervention, contrast, outcome, and context of interest have been made explicit. Kitchenham suggests starting in natural language but converting to a structured question as the ideas become refined.
- *Identify search parameters*. Next in the process, researchers must define a repeatable strategy for searching the literature. Doing so requires setting clear criteria for the following issues (among others):

  - Which sources will be searched
  - How sources will be filtered
  - How quality of sources will be assessed
  - What information will be extracted from sources
  - How missing information will be handled

- *Find evidence*. Assuming the search criteria and range of permissible sources have been defined in detail as above, finding evidence is then the process of exhaustively searching all sources for any paper that matches the criteria. Having the search specified in such detail helps ensure that the search process is repeatable, that is, that multiple users conducting a search according to the same criteria would find exactly the same sources.
- *Analyze evidence*. Analyzing the publications found in the search consists of first filtering out unsuitable publications and then extracting the information needed from those remaining.

  - During this round of filtering, only primary studies should be selected for inclusion in the systematic review. That is, researchers should analyze only reports of studies that directly examined the research question. Analysis or synthesis of studies performed by other researchers are not to be included in the study in combination with primary sources. (Such surveys should

themselves be used as pointers to important primary sources or to compare against the final outcome of the systematic review.) An important question is whether certain types of studies or evidence should be excluded from consideration at this point. However, Kitchenham notes that due to the number of studies currently published in software engineering, researchers on most topics will not be able to be so selective: "In software engineering, we will usually accept all levels of evidence. The only threshold that might be viable would be to exclude level 5 evidence [expert opinion] when there are a reasonable number of primary studies at a greater level…" (Kitchenham, 2004). Still, the quality of each study included in the analysis must be assessed so that this can be considered when the results from each study are compared and contrasted during the integration phase.

- From each study that remains after the filtering is performed, the required data for the analysis must be extracted. The guidelines suggest that a template should be defined for each systematic review conducted and applied to each publication, so that complete information is extracted from each and organized consistently.

- *Integrate evidence*. Having defined in earlier phases concrete guidelines for what type of evidence will be included in the systematic review, the guidelines for how the evidence is to be integrated are not as specific. This is likely because the methods which are feasible for each systematic review will depend largely on how much and what type of evidence has been utilized, and on the specific research question under study. Kitchenham does note that conclusions in software engineering will need to be drawn from many different types of studies, but guidelines for combining different types of studies are not given. Although qualitative measures are allowed, it is recommended to convert each to a quantitative measure if at all possible. One way of reporting such results is via a forest plot, which is feasible if all studies measure the same treatment variable in the same units (or using different measures that can be converted to the same units).

Although not mentioned explicitly in our generic process, the systematic review guidelines do contain an addition activity for documenting the review. The justification for having this listed as a separate step is that the systematic review cannot be considered complete until it has been validated; the authors suggest that such validation is likely to happen via peer review. In the event that the report is published as a technical report or some other non-peer reviewed document, it should be made available via the web and a peer review organized for this purpose.

3.1.1. Lessons Learned in Application to Software Engineering

In the software engineering domain, this approach has been applied to a number of different analyses, which are increasing in number each year. After a relatively few applications published in 2004 and 2005, there has been a large increase in 2006 of

the number of systematic reviews, especially in Master's theses and other student work. Some key examples in which systematic review was applied to test a research hypothesis include:

- Jørgensen (2004) conducted a systematic review of studies of estimating software development effort. He found, first, that estimation based on expert judgment was the most often-used approach. The systematic review found 15 different studies comparing expert estimates to estimates produced using more formal models. The results about which estimation approach produced more accurate estimates are inconclusive: five studies found expert judgment more effective; five found formal estimation models more effective; and five found no difference. However, Jørgensen was able to formulate a number of guidelines for improving expert estimation, which are each supported by at least some of the studies surveyed.
- Jørgensen and Moløkken-Østvold (2006) used a systematic review to test an assessment of the prevalence of software cost overruns done by the Standish Group. They investigated whether they could find evidence to support one of the often-cited claims of the 1994 "CHAOS" report, namely that "challenged" software engineering projects reported on average 189% cost overruns. This systematic review found three other surveys of software project costs. The comparison could not be definitive, since the Standish Group did not publish their source data or methodology. However, the researchers found that the conclusions of the Standish Group report were markedly different from the other studies surveyed, raising questions about the report's methodology and conclusions.
- Kitchenham et al. (2006) undertook a systematic review to investigate the conditions under which organizations could get accurate cost estimates from cross-company estimation models, specifically, the conditions under which those cross-company models were more accurate than within-company models. Seven papers were found that represented primary studies on this topic. The results were inconclusive: four found cross-company models were significantly worse than within-company models, while the remainder found that both types of models were equally effective.

Mendes (2005) applied systematic review for a slightly different goal: to assess the level of rigor of papers being published in the field of web engineering. In this case, it was not a single research hypothesis that was being explored; rather, Mendes was assessing the percentage of papers in the field that could be included in a systematic review of any hypothesis in this area, according to criteria for rigor that she set. 173 papers were reviewed and only 5% were deemed sufficiently rigorous, which emphasizes that this approach ensures rigor by being quite restrictive about the quality of papers accepted as input.

Some authors explicitly comment on the difficulty of applying the approach given the state of the software engineering literature. Jørgensen (2004), for example, mentions that few if any of the studies he identified met the criteria of reporting the statistical significance of their results, defining the population sampled, or using

random sampling. For these reasons, it appears to be difficult to define the quality criteria too rigorously, in case the number of studies that can be included become too small to produce interesting results.

Because of the costly nature of applying this approach, some researchers have done some tailoring of the approach in application. For example, even though a best practice is to minimize bias by using two researchers to do the analysis, some researchers who are applying the method feel it is practical to use only one.

3.1.2. Assessment

Systematic review does cover a range of sources from different environments. To describe the conditions for which this analysis approach may best be suited, we examine it in reference to our quality criteria:

- *Applicability for quantitative data*: +
- The literature contains several examples of research questions addressed by systematic review of quantitative evidence sources.

- *Applicability for qualitative data*: –
- At the moment, this approach seems less well suited for evidence sources that contain qualitative data. Although methods for qualitative synthesis do exist (e.g., Noblitt and Hare, 1988), none of the applications of systematic review that we could find in the software engineering literature used qualitative data as a substantial source of information. Moreover, the guidelines in this field (Kitchenham, 2004) seem written with quantitative data in mind. It is likely that this will need to be explored further in future applications.
- *Scalability*: –
- An assessment of this attribute would depend on how a given application defines the quality and filtering criteria. However, we can say that applications to date have typically used fairly restrictive criteria. The lessons learned cited above do show that several authors have commented that a fairly small percentage of publications were suitable for inclusion in the systematic reviews that they ran.
- *Objectivity*: +
- The procedure is very well specified. Although key filtering criteria are allowed to be user-defined for each application, and so could theoretically be defined so as to impair the objectivity of the study, this would presumably be caught during the peer review of the study process and results.
- *Fairness*: +
- Fairness is typically high, since the search criteria are to be represented as search queries and repeated in several repositories. The researcher must take all documents matching the query; he or she is not allowed to pick and choose arbitrarily.
- *Ease of use*: +/–
- The procedure and results would be easily accessible to researchers, but the amount of detail in the report would not be user friendly for supporting decisions

by practitioners. This can be mitigated by applying additional effort aimed at creating multiple reports for different audiences, particularly by abstracting actionable guidelines for practitioners from the research (see for example Koyani et al., 2003).

- *Openness*: +
- The amount of detail that is required to be documented and included in the final report of results makes this a very open process. In fact, peer review of each step of the process is called for to ensure quality and rigor in the results.
- *Cost*: –
- Researchers have pointed out that systematic review is effort-intensive and hence high cost: "Systematic reviews require considerably more effort than traditional reviews" (Kitchenham, 2004). Part of this cost is due to the fact that this approach requires extensive and lengthy documentation. It is moreover not well suited for application by a single researcher, since a "best practice" is to use at least two researchers to minimize biases. Although we could find no comprehensive estimate of costs for performing systematic reviews, anecdotally we did hear from researchers who expressed some concern about their expensive nature in comparison to the benefits received. One researcher questioned the wisdom of adopting such techniques from the medical field, which has a research budget many times that of the budget for software engineering.

3.2. Meta-analysis

Meta-analysis is a method for combining data from different datasets collected during different studies, in order to statistically test a hypothesis. By using data from multiple datasets, the meta-analysis allows the investigation of whether the effect under study is robust across multiple contexts. By combining datasets across studies, meta-analysis provides for the statistical test a larger number of data which improves the chances of detecting smaller effect sizes than any test of a single dataset in isolation.

Meta-analysis should be seen as a special case of systematic review, rather than a distinct approach. It follows the same general process of systematically collecting, analyzing, and integrating evidence, but specifies certain techniques that are appropriate when the evidence is expressed in comparable, quantitative metrics.

Both meta-analysis and systematic review have a long history of use in other disciplines. Its applicability to software engineering has been studied relatively recently, as a way of getting greater benefit from the fairly few and expensive studies that are run on software engineering phenomena.

Procedure. The procedure for conducting meta-analysis in software engineering has been specified in previous publications. The information below has been summarized from Miller (Miller, 2000) unless otherwise noted. For purposes of comparison, we discuss the meta-analytic procedure for quantitative data using the same broad steps as we used for the more general systematic review approach. However, since this type of meta-analysis is concerned with a statistical test of

quantitative data, many of the phases can be described in more detail, and require more constraints, than does the general systematic review process.[3] We map these activities to our generic knowledge-building process as follows:

- *Define topic*. The research topic investigated by a meta-analysis should be expressed in the form of a relationship between two variables. Although this is a matter of debate, the conservative approach is that the meta-analysis should be done between two variables only. Separate analyses should be run if there are more than two variables of interest.
- *Identify search parameters*. Although no specific guidelines are given on how to run the search, a number of important constraints govern which sources can be used in the meta-analysis:
  - Meta-analysis requires some knowledge about the individual data sets that it analyzes. Hence, only studies can be used which report the appropriate information regarding the results. If the raw data is not available, then the process requires from each source at least the mean, variance (or standard deviation), number of subjects, and details about the normality of the data. When non-significant results are reported an estimate of the statistical power of the experiment should be included.
  - Independence of the studies is important. Selecting studies among which some dependencies exist can weaken or invalidate the results.
  - Miller notes that "[c]urrently no work exists, which attempts to validate the use of meta-analysis for non-experimental results," and therefore recommends that researchers in software engineering not use evidential data from sources other than experiments in meta-analysis at this time. (The reasoning is that the randomization which takes place in experimental studies eliminates bias and confounding factors within the experimental results.) Thus it may be more appropriate, and is certainly safer, to analyze the results from different types of studies separately and then examine whether they tell a consistent story.
- *Find evidence*. This activity should take the form of an exhaustive literature search aimed at finding all empirical evaluations which describe relationships between the two variables of interest.
- *Analyze evidence*. As some authors have noted, there is a first pass that is necessary over the collected set of sources "to reconcile the primary experiments – i.e., define a common framework with which to compare different studies. This involves defining common terms, hypotheses, and metrics, and characterizing key differences" (Perry et al., 2000). In a second pass, the data must be examined more deeply for:
  - Errors in the individual data sets that could be corrected

[3] We recognize that procedures have been described for meta-analysis of qualitative data, e.g., Paterson et al., 2001, but as we are aware of no instances where they were applied in software engineering research we keep this section focused on quantitative applications.

- Quality of the studies, in order to assign a weighting to each. In order to avoid bias, Miller notes that the recommended practice is to organize an independent panel of experts

- *Integrate evidence*. Having compiled and created a common framework for the individual data sets, integrating the evidence is done by means of running the proper calculation over the data values obtained. This will provide a quantitative, statistically valid answer to the question of whether there is a significant relationship between the two variables of interest. One important note for the analysis is that Miller recommends that meta-analysis not be employed to resolve differences among conflicting results. Meta-analysis was designed to combine results from similar experiments, not to deal with heterogeneous data sets.

3.2.1. Lessons Learned in Application to Software Engineering

In the software engineering domain, this approach has been applied in relatively few cases. Certainly one of the most relevant of these is the study by Miller (2000), in which meta-analysis was applied to abstract conclusions across defect detection experiments (i.e., experiments that ask the question: "Which (if any) defect detection technique is most effective at finding faults?"). This was an important test of meta-analysis in the software engineering domain, as defect detection techniques are among the most often-studied software engineering phenomena. Hence, if sufficient data could not be obtained on this topic, it would be difficult to understand how meta-analysis could be suitable for many other topics in software engineering.

However, the results from Miller's study were inconclusive. On a review of the literature, only five independent studies could be found which had investigated similar enough hypotheses and used similar enough measures to be compared. Upon analysis of the data the results of those studies were so divergent that meta-analysis was not deemed to be applicable. A possible reason for this is that the effectiveness of defect detection techniques is highly dependent upon the types of defects in the artifact being examined; the studies included in Miller's analysis did not describe the defect type information in sufficient detail that a mapping could be made to transform the results onto a common taxonomy. Thus, it could not be assessed whether those studies applied the techniques to defect profiles that were at all comparable.

A related use of this technique in software engineering was the attempt by Hayes to abstract results across five studies of inspection techniques, where four of the studies were either partial or full replications of the first (Hayes, 1999). In this case, the study designs were all very similar, which should have facilitated the ability to draw a common conclusion from this body of information. However, Hayes was forced to conclude that the effect sizes were significantly different across the studies and hence that a meta-analysis was not an appropriate method for reasoning

about the underlying phenomenon. Hayes is able only to speculate about some causes for this – for example, that the studies were run in different cultural contexts and by subjects with different levels of experience – but it is worth noting that these resulting hypotheses may be of as much practical interest to the research community as a successful meta-analysis would have been.

A final application of meta-analysis in the software domain that is especially worthy of note was a study conducted by Galin and Avrahami (2005). These authors attempted to address the question of whether software quality assurance programs work by conducting a meta-analysis of studies examining the effects of the Capability Maturity Model (CMM) for software. The authors point out that CMM has been one of the most widely-deployed software process improvement methods for an extended number of years, and so would be among the most likely approaches for which sufficient data would exist. For the same reason, this analysis was also a good test of the suitability of meta-analysis for software engineering research. In this case, the results were more positive: 22 studies were found that examined the effects of the CMM on software process improvement and, of these, 19 contained sufficiently detailed quantitative information to be suitable for analysis. The analysis did find substantial productivity gains when organizations achieved the initial improvement levels of the CMM (although data was missing that addressed higher levels of achievement).

In the end, the lesson learned about applying meta-analysis to software engineering seems to be that: "…the heterogeneity of current empirical results is a major limitation in our ability to apply meta-analytic procedures" (Miller, 2000). Because of the large amounts of variation from so many different context variables, which exists in any set of software engineering experiments, we may be unable to generate statistically definitive answers for many phenomena other than those with the largest effect sizes (e.g., organizations going from an undisciplined development process to achieving initial levels of the CMM). This is true even in cases which seem to lend themselves to cross-study analysis, for example, topics for which there is a rich body of studies, some of which may even be replications of one another. For many other topics of interest which do not have such a rich set of studies, which tend to be the ones of most interest to researchers and practitioners, it is still an open question whether the studies undertaken so far are additive and can be combined via meta-analysis to contribute to an eventual body of knowledge.

3.2.2. Assessment

- *Applicability for quantitative data*: +
- When sufficient studies with quantitative results can be found, meta-analysis is the most rigorous way of combining those results.
- *Applicability for qualitative data*: –
- Meta-analysis commonly relies on statistical tests that are not suited for qualitative data. Methods for applying meta-analysis to qualitative analysis have been described but not yet applied in the field of software engineering.

- *Scalability*: +/–
- As with any technique, the number of suitable studies that could be found would depend on how the researcher defines the eligibility criteria. As an example, Miller's case study (Miller, 2000) starts with a relatively loose criteria (that all studies measure the same effect) but notes that it could be tightened, for example by stipulating that only a particular type of study design be used, or that small studies be either dropped from the analysis or given less weight. However, given the relative scarcity of software engineering data, the looser criteria is probably suitable for the field now. Although the study by Galin and Avrahami was able to use 19 out of 22 sources found, the more typical experience in software engineering studies at the moment seems to be that a sufficient number of studies is more difficult to find.
- *Objectivity*: +
- The objectivity of the approach should be seen as quite high: the procedure and statistical methods are very well specified. Different meta-analyses applied to the same datasets will always produce the same answer.
- *Fairness*: +/–
- Since no specific guidelines are given for how researchers should conduct the literature search to find evidence sources, the process will be as fair and unbiased as the researcher's search approach.
- *Ease of use*: –
- The outputs of this approach are aimed more at researchers than at practitioners. Training in statistical methods is necessary in order to apply the technique and interpret the results correctly.
- *Openness*: +/–
- There are no special requirements of the technique with respect to openness. It is to be expected that any serious meta-analysis would be subjected to peer review on its way to publication, and hence should theoretically allow reviewers to replicate the same analysis if desired.
- *Cost*: +/–
- There are no special constraints on cost. There are no special documentation requirements.

3.3. An Experience Portal-Centered Approach

Scientists at the Fraunhofer Center – Maryland developed an approach for accumulating and analyzing disparate evidence sources in 2002, to help the U.S. Department of Defense provide information for a central best practices clearinghouse about software acquisition and development. In contrast to the previous approaches discussed, there is no single comprehensive reference, although details of the approach have been published (Shull and Turner, 2005; Feldmann et al., 2006). The general method which was instantiated in the clearinghouse extends previous knowledge-building approaches used in the Experience Factory method (Basili et al, 1994a)

and is known as EMPEROR (Experience Management Portal using Empirical Results as Organizational Resources).

An important way in which EMPEROR differs from the Experience Factory as well as from systematic reviews and meta-analysis is that it is designed to be executed via a community rather than a single research team. EMPEROR provides a mechanism for users in the field to submit their experiences with a given technology and for such experiences to be reflected in the summarized knowledge. Thus, it aims at abstracting conclusions at a different level than the previously mentioned methods.

This approach was primarily designed for decision support but is also useful for theory generation.

Procedure. The basic procedure for building knowledge through the EMPEROR approach was defined in several papers (Shull and Turner, 2005; Feldmann et al., 2006) and is summarized below. An important distinction from the previous approaches in this chapter is that EMPEROR imposes lower barriers to including information in the analysis, in order to be more inclusive of experiential information from participants. Less-than-rigorous information may therefore be entered as part of the knowledge base although it is labeled as such, and the summarized analysis is checked later to make sure that such information has not been overly relied on in forming conclusions.

- *Define topic*. As with other approaches, EMPEROR requires that the topic of knowledge gathering first be defined. Although this topic definition might be in the form of a hypothesis, it may also be simply a particular practice or technique about which the available evidence should be summarized. In general, topics investigated with this approach are of the form: What is the expected outcome of using a particular practice in a certain environment?
- *Identify search parameters*. Also similar to other approaches, EMPEROR contains a step in which the person applying the process must make explicit which types of evidence will be acceptable to the search and in which venues to look for that evidence. EMPEROR however is less restrictive and allows less rigorous types of evidence to be included (e.g., interviews, experience reports, white papers) both to get a more inclusive survey of the state of the practice and because for many questions sufficient amounts of highly rigorous studies are simply not to be found. This view of the software engineering literature is supported by many of the example applications of meta-analysis and systematic review discussed in previous sections.
- *Find evidence*. The search for the evidence is conducted given the constraints decided upon. When the published literature is found to be significantly lacking, researchers are advised to consider conducting interviews with representative practitioners in order to create additional workable knowledge. For each evidence source, a template is filled out; the information entered in such a template is expected to be largely textual. Where quantitative evidence is found it should be recorded taking special care to record the unit of measure along with the values. It is not expected that all evidence on the same topic will be recorded in the

same measures or in measures that can be translated one to the other. This phase of the procedure may go on for an extended period of time. Evidence may be allowed to accumulate opportunistically, with new templates being filled out as new evidence becomes available. The evidence found so far is made available for interested parties, e.g., at a website that can be updated as new evidence is found.

- *Analyze evidence*. As each evidence template is completed, it is assigned a measure of trustability based upon objective descriptions of how rigorously the practice under investigation was applied, the results were measured, and how results were reported. An example trustability scale (Feldmann et al., 2006) ranks each evidence source on a scale of 1 (signifying anecdotal evidence from a single source) to 20 (sustained and measured evidence that has undergone peer review).
- *Integrate evidence*. When sufficient evidence has been collected, a textual summary is constructed that describes the body of evidence that has been found. The summary is authored by a subject matter expert, that is, someone with sufficient knowledge of the topic area so as to be able to describe the important information from the knowledge accumulated. Before being published, the summary is reviewed by an objective, outside panel consisting of representatives from industry, government, and academia. This panel reviews the summary from the point of view of accuracy and objectivity (especially whether all of the conclusions can be traced back to a statement in the evidence templates) and representativeness (whether the evidence profiles that were used represent environments of interest and whether the evidence sources used do not represent a biased subset of users).

3.3.1. Lessons Learned in Application to Software Engineering

In the software engineering domain, this approach has been applied so far only in the context of the US Department of Defense's Best Practices Clearinghouse (Dangle et al., 2005). This single project contains analyses of several different practices, however, and hence several different example applications of the technique. These applications range from topics for which experiential data of all kinds is very easy to find (e.g., the costs and benefits of software inspections or spiral development) to topics for which the available data is much more scarce (e.g., the costs and benefits of a process variant known as performance-based earned value management).

As the project repository is currently in an initial phase, the approach will shortly undergo a more thorough evaluation as the project resources are opened up to the user community. Lessons learned will be analyzed and reported on in the near future. Among the most important aspects to be tested in this effort, however, is the question of whether an active community can be built around such a repository and whether it will work to contribute to and refine the evidence collection and hence the summarized information that can be built atop it.

3.3.2. Assessment

It is important to note again that the EMPEROR approach proceeds in a very different manner than the other ones discussed in this chapter. Analyses in this approach are always open to review by the user community, so as to elicit information that may have been missed in the initial review and to allow users to get the benefits of information before the entire review has been completed. Also, rather than take a restrictive approach and allow only the highest-quality evidence to be included in the analysis, EMPEROR will allow less-rigorous types of evidence (e.g., interviews, experiential anecdotes) as long as such evidence is always labeled with an appropriate caveat. Our discussions with our user advisory group has indicated that users are happy to get what guidance is available, as long as they know the appropriate level of confidence to place in it. Given the dearth of highly-rigorous studies that exists on many topics, there seems to be a need for workable interim solutions that can give some guidance.

- *Applicability for quantitative data*: +
- The process makes no special distinction between qualitative and quantitative data; it is equally well suited to both.
- *Applicability for qualitative data*: +
- Because the final summary of abstracted information is text-based, it is very well suited to incorporating qualitative data.
- *Scalability*: +
- The process has been designed to be as inclusive as possible. Any incoming evidence has only to pass a sanity check by a subject matter expert. However, each admitted evidence source is always tagged with an objective indicator of its quality.
- *Objectivity*: –
- The EMPEROR approach is more susceptible to subjectivity than the other approaches. However, it contains safeguards that do try to guard against such problems. For example, because the barriers to entry are low, evidence may be submitted that is anecdotal and subjective. However, this evidence would be tagged as of lower quality and should be marked as of less importance when the summary is created. As another example, the summary itself is a textual summary that needs to combine many disparate sources of evidence and many different measures of a practice's effectiveness. To guard against this, the process requires that the summary is always created by an expert in the topic under study and furthermore, that it be reviewed and accepted (or not) by an outside panel of experts representing different points of view.
- *Fairness*: –
- Similarly to objectivity, the approach is susceptible to bias but contains internal safeguards that attempt to mitigate this. For one example, there are no defined, repeatable search criteria for finding evidence sources. However, by stipulating that the in-process results are always visible to users, the approach allows users who do not see their own experiences represented in the repository to submit

new evidence that includes their own point of view, helping to correct any bias. As a second example, the textual summary may include bias if the included evidence sources exhibit bias. However, the objective outside panel of experts that reviews completed summaries is charged with assessing this. It may also be worth noting that, unlike the other two approaches discussed in this chapter, EMPEROR may suffer less from publication bias (i.e., the threat that negative results on a particular topic, or results that do not match the conventional wisdom, are less likely to be written up or accepted as part of the published literature). EMPEROR avoids this by allowing the submission of less rigorous unpublished experiential data (e.g., via interviews) that attempt to paint a more accurate picture of the state of the practice.

- *Ease of use*: +
  A unique point of the EMPEROR approach is that final vetting of summaries and results is done by representatives who look not only at the accuracy of results but also of the usefulness for the targeted users.
- *Openness*: +
- All in-process evidence and summary information are provided, with traceability links from one to another. Even the scoring models are made explicit, so that users looking to understand why an evidence source received a particular trustability rating can see the underlying scoring model. This openness has advantages that go beyond allowing peer review of the summaries that are produced. The open nature of the EMPEROR approach, as reflected by the requirement to publish in-process reviews, helps to identify areas where more evidence is most important to find. For example, practices for which there is a large degree of anecdotal information are ones which could benefit from a more rigorous study to either confirm or deny the conventional wisdom. The process can also work in the other direction: Practices for which there are a large number of rigorous academic studies but no experiential information from industrial contexts may be good candidates for early adopters in commercial environments to try out.
- *Cost*: +/−
- Another unique aspect is that the EMPEROR approach requires the publication of all materials and results to date, even though the process is ongoing. Thus, end users of the information need not wait until the entire process has been completed to get some benefit. Building up the evidence sets and the resulting summaries can be a costly process, but the entire cost is not required to be paid before any benefit is seen by users of the information.

4. Discussion and Conclusions

For a direct comparison of the approaches, we summarize the evaluations for each of the approaches along our eight quality attributes in Table 6.

The table helps to detect some interesting commonalities and differences among the techniques:

Table 6 Approaches and quality attributes

Quality attributes ⇒	Applicability to quantitative data	Applicability to qualitative data	Scalability	Objectivity	Fairness	Ease of use	Openness	Cost
⇓ Approach								
Systematic review	+	–	–	+	+	+/–	+	–
Meta-analysis	+	–	+/–	+	+/–	–	+/–	+/–
Portal-centered approach	+	+	+	–	–	+	+	+/–

- Basing theories on quantitative data seems to be the "standard" approach to building up theories from across multiple studies, as all of the approaches are designed to abstract theories from quantitative results. However, as has been noted in many of the previous sections, sufficient quantitative data cannot always be found for many topics of interest. For this reason, the additional quality attributes are especially helpful in making decisions about the applicability of approaches for different issues.
- If the majority of experiential information on a topic is expected to be in the form of qualitative data (or quantitative data collected using different incompatible measures), the portal-centered approach is an appropriate choice for combining the evidence sources to abstract a general theory. However, the price to be paid for this ability is a reduction in the rigor (objectivity and fairness) of the resulting conclusions. Although the portal-centered approach includes different levels of quality checking that attempt to remove subjectivity and bias, there is more risk in using this approach than there is for the other approaches, which remove unrigorous evidence by definition.
- Similarly, there is a tradeoff to be had between the inclusiveness of the technique (scalability) and the rigor of the results (fairness and objectivity). The portal-centered approach allows researchers to include less than rigorous evidence sources in the analysis, although the confidence in each is marked with a trustability score. However, again this introduces more risk than approaches which will only accept the most rigorous evidence sources as input. The final decision should of course be based on how much evidence is expected to be available to support interesting and relevant theories on the topic of interest – and the rigor of that decision should be understood and labeled.
- The ease of use attribute helps to highlight a major difference between the portal-centered approach and the other two approaches: The portal-centered approach focuses on providing decision support to practitioners (i.e., providing useful information at the expense of complete rigor), while systematic review

and meta-analysis are focused on providing highly rigorous results (while trading away ease of understandability to practitioners). A related issue is that the portal-centered approach intends to provide information that can support a given decision, not provide a definitive answer to a research question.

- All of the approaches are "open" in that they provide some transparency of the process to interested parties. Both, the systematic review and the portal-centered approach have this as an explicit goal for providing high-quality information.
- All of the approaches are costly; none are cheap to apply. Systematic review may have the most overhead in this regard, as has been commented by multiple researchers who attempted to apply the process guidelines with full rigor. The portal-centered approach is unique in defining useful in-process deliverables that can be published to provide value to users before the final analysis is completed.

As indicated by this comparison, there is no single approach that is capable of meeting all of the quality attributes. A major theme that comes through in the analysis is that full rigor is in tension with the ability to include all types of empirical information and provide easy-to-understand conclusions aimed at practitioners. A key challenge for the future may lie in managing these tradeoffs better, that is, in finding new approaches that combine aspects of the approaches discussed in this paper, to yield positive ratings along more of the quality attributes.

Ongoing research is attempting to address exactly this issue, for example by providing relatively easy-to-use approaches for converting qualitative data into the quantitative data that is usable by meta-analysis and systematic review (Port et al., 2006), or by providing easy-to-use approaches for combining different studies that retain more rigor (Mohagheghi and Conradi, 2006). As this work is fairly new and has not yet been applied in many contexts, it is an open question of how successful it will be in marrying rigor with a less costly, more practical approach. However, such exploration is necessary if we as a field are to aim for truly robust approaches to theory building that can best leverage the multiplicity of kinds and types of existing empirical evidence.

References

Basili, V.R., Selby, R., and Hutchens, D., (1986) Experimentation in software engineering. *IEEE Transactions on Software Engineering*, 12(7): 733–743.

Basili, V.R., Caldiera, G., and Rombach, H.D., (1994a) Experience factory. In *Encyclopedia of Software Engineering*, John, J. Marciniak, (ed.) Vol. 1, Wiley, New York, pp. 469–476.

Basili, V.R.,. Caldiera, G., and Rombach, H.D., (1994b) Goal question metric paradigm. In *Encyclopedia of Software Engineering*, John, J. Marciniak, (ed.) Vol. 1, Wiley, New York, pp. 528–532.

Basili, V.R., Shull, F., and Lanubile, F., (1999) Building knowledge through families of experiments. *IEEE Transactions on Software Engineering*, 25(4): 456–474.

Dangle, K., Dwinnell, L., Hickok, J., and Turner, R., (2005) Introducing the department of defense acquisition best practices clearinghouse. *CrossTalk*, 18(5): 4–5.

Feldmann, R., Shull F., and Shaw, M., (2006) Building decision support in an imperfect world. *Proceedings of International Symposium on Empirical Software Engineering (ISESE)*, Vol. II, Rio de Janeiro, Brazil, pp. 33–35.

Galin D. and Avrahami, M., (2005) Do SQA programs work – CMM work. A meta analysis. *Proceedings of IEEE International Conference on Software – Science, Technology and Engineering (SwSTE05)*, Herzelia, Israel, pp. 95–100.

Hayes, W., (1999) Research synthesis in software engineering: a case for meta-analysis. *Proceedings of the Sixth International Software Metrics Symposium (METRICS'99)*, Boca Raton, FL, p. 143.

Jørgensen, M., (2004) A review of studies on expert estimation of software development effort. *Journal of Systems and Software*, 70(1–2): 37–60.

Jørgensen, M., and Moløkken-Østvold, K. J., (2006) How large are software cost overruns? Critical comments on the Standish group's CHAOS reports. *Information and Software Technology*, 48(4): 297–301.

Kitchenham, B. (2004) *Procedures for Performing Systematic Reviews*, Joint Technical Report, Keele University TR/SE-0401 and NICTA 0400011T.1.

Kitchenham, B., Dybå, T., and Jørgensen, M., (2004) Evidence-based software engineering. *Proceedings of the International Conference on Software Engineering*, Edinburgh, UK, pp. 273–281.

Kitchenham, B., Mendes, E., and Travassos, G. H., (2006) Systematic review of cross- vs. within-company cost estimation studies. *Proceedings of the Evaluation & Assessment in Software Engineering (EASE)*, pp. 89–98.

Koyani, S.J., Bailey, R.W., and Nall, J.R., (2003) Research based web design and usability guidelines. National Cancer Institute. Available for download at http://usability.gov/pdfs/guidelines.html.

Mendes, E., (2005) A systematic review of web engineering research. *Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering*, Noosa Heads, Australia, pp. 408–418.

Miller, J., (2000) Applying meta-analytical procedures to software engineering experiments. *Journal of Systems and Software*, 54: 29–39.

Mohagheghi, P., and Conradi, R., (2006) Vote-counting for combining quantitative evidence from empirical studies – An example. *Proceedings of International Symposium on Empirical Software Engineering (ISESE)*, Vol. II, Rio de Janeiro, Brazil, pp.24–26.

Noblitt, G.W., and Hare, R.D, (1988) *Meta-Ethnography: Synthesizing Qualitative Studies (Qualitative Research Methods)*, Sage Publications Ltd., Thousand Oaks, CA..

Paterson, B., Thorne, S., Canam, C., and Jillings, C., (2001) *Meta-Study of Qualitative Health Research: A Practical Guide to Meta-Analysis and Meta-Synthesis*, Sage Publications Inc, Thousand Oaks, CA.

Perry, D., Porter, A., and Votta, L., (2000) Empirical studies of software engineering: a roadmap. *Proceedings of International Conference on Software Engineering*, Limerick, Ireland.

Petticrew, M. and Roberts, H., (2006) *Systematic Reviews in the Social Sciences. A Practical Guide*, Blackwell Publishing, Oxford.

Port, D., Kazman, R., Nakao, H., Hoshino, N., and Miyamoto, Y., (2006) Investigating a constructive scorecard model for creating meaningful quantitative data from qualitative inputs. *Proceedings of International Symposium on Empirical Software Engineering (ISESE)*, Vol. II, Rio de Janeiro, Brazil, pp. 27–29.

Shull, F. and Turner, R., (2005) An empirical approach to best practice identification and selection: the US department of defense acquisition best practices clearinghouse. *Proceedings of International Symposium on Empirical Software Engineering (ISESE)*, Noosa Heads, Australia, pp. 133–140.

Sjøberg, D.I.K, Dybå, T., Anda, B.C.D., and Hannay, J.E., (2007a) Building theories in software engineering. In *Advanced Topics in Empirical Software Engineering: A Handbook*, Shull, F., Singer, J., and Sjøberg, D.I.K (eds.), Springer, Berlin.

Sjøberg, D.I.K., (2007b) Documenting theories. In *Experimental Software Engineering Issues: Assessment and Future*, Basili, V.R., Rombach, D., Schneider, K., Kitchenham, B., Pfahl, D. and Selby, R, (eds.), Springer-Verlag, Berlin Heidelberg, pp. 111–114.

van Solingen, R. and Berghout, E., (1999) *The Goal/Question/Metric Method*, McGraw-Hill Education, New York.

Zelkowitz, M., (2001) Models for industrial validation of new technology. ISERN workshop at Strathclyde University. Available via http://isern.iese.de/network/ISERN/pub/meetings/Glasgow2001/Agenda.htm.

Zelkowitz, M. and Wallace, D., (1998) Experimental models for validating technology. *IEEE Computer*, 31(5), pp. 23–31.