Survey of Software Development Effort Estimation Taxonomies

Tomás Vera, Sergio F. Ochoa, Daniel Perovich
Computer Science Department, University of Chile
{tvera, sochoa, dperovic}@dcc.uchile.cl

***Abstract*.** Determining the development effort (cost and time) is one of the main challenges of the software projects formulation and management. In order to address such a challenge, several effort estimation models have been proposed since the '80s, as a way to envision the software development effort before and during a project, and also reduce the project risks. As the use of these estimation models involves several requirements, a particular model is not necessarily suitable to support the estimations of every software company or development team. On the other hand, it is not easy for a software company to identify estimation models that are potentially appropriate for them. In order to identify the most prominent software estimation models and their characteristics, we conducted a systematic literature review on taxonomies of effort estimation methods, reported between 2000 and 2017. The goals were mainly two: (1) determining the start-of-the-art in this area and (2) identifying the main variables used to classify these methods. Based on this review, we elaborate on how software companies can use these taxonomies and classification criteria to identify suitable methods to support their estimations, considering some particular features of the company.

1. Introduction

The first reports about software effort estimation methods[1] were published in the '50s. Particularly, between the '50s to '70s this activity was done manually and based on expert judgement. As the software was becoming larger and more complex, the software engineering practices became more protagonist, including the effort estimation of the development of these systems. Thus, several software effort estimation models were proposed to deal with such a requirement; for instance Doty [HER77], SLIM [LAW91], Checkpoint [JOPN97], Price-S [PAR88], SEER [JEN83], and CoCoMO [BOE81]. Figure 1 presents a timeline indicating the definition of the most prominent software development effort estimation (SDEE) methods.

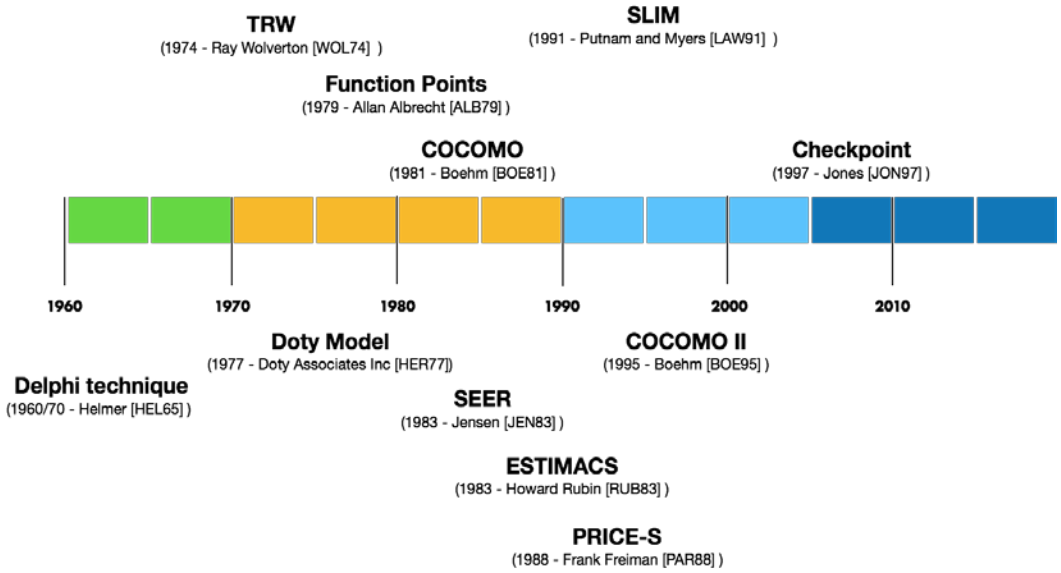

Figure 1. Timeline of the definition of the main SDEE methods.

[1] In this document we use the words "method", "model" and "technique" as synonymous.

The development of multiple SDEE methods during the first five decades (until 2000) helped this area not only to formalize, but also diversify the proposals, generating families of estimation methods. Today, the literature reports more than a hundred of SDEE proposals, grouped in several categories depending on the classification criteria being used. Regardless this variety, most of these methods cannot be directly used by software companies as most methods provide a general guide on how to conduct estimations. In order to get accurate estimations, a software company must choose a method and then adjust or configure it to make it representative of the kind of development effort conducted by the organization. Selecting an appropriate method and adjusting it properly is a complex task that most companies, particularly the small ones, are not usually able to perform given their lack of expertise and guidelines to perform these activities.

Today, the use of an effort estimation model is more a need than an option for most software companies, since the projects are no longer trivial and the competence in this industry is high. For competitiveness reasons, the projects' budgets should be accurate, which implies that the use of a SDEE method tailored for each software company is mandatory. A report of The Standish Group over 25,000 projects shows the consequences of not using a SDE method to support the software estimations [JOH06]. These consequences go from lack of competitiveness and the underestimations, to projects failures and eventually business failures.

In order to reduce the complexity of identifying appropriate SDEE methods for particular software companies, we conducted a systematic literature review (SLR) following the guidelines of Kitchenham and Charters [KIT07]. This study identified the taxonomies and the classification criteria of SDEE methods, as a way to help software companies to easily discard inappropriate methods and also determine the potential candidates according to the companies' features.

The classification criteria allows software companies to pre-select families (or clusters) of SDEE methods, and analyze the potential suitability of each cluster depending on, e.g., if they require historical information, are based on expert judgement, require few or much calibration, or if they are public or protected (commercial). A company can reduce the effort to identify potentially suitable SDEE methods by analyzing the clusters. Therefore, the SLR reported in this paper intended to answer the following research questions:

RQ1 - What are the main taxonomies of SDEE methods reported in the literature? Answering this question will allow identifying the taxonomies available to perform the pre-selection of techniques based on clusters.

RQ2 - What are the most relevant classification criteria used in these taxonomies? Answering this question will allow determining the most transversal clusters according to the reported studies.

This paper is structured as follows. Section 2 describes the methodology followed in this study to perform the SLR. Section 3 presents and discusses the results of this study, highlighting the findings. Section 4 shows the implications of the findings for the industry and the academia. Section 5 presents the limitations of this study, and Section 6 shows the conclusions and the future work.

2. Methodology

In this SLR the author followed the process recommended by Kitchenham and Charters (2007) to conduct these studies in the software engineering domain. This process involves six stages (Fig. 2): (1) definition and review of the research questions, (2) definition of the search strategy and performing the search

process, (3) selection of studies to be analyzed, (4) quality evaluation of the selected studies, (5) data extraction from the studies, and (6) synthesis of the results. Next we explain more in detail these stages and how they were conducted in this study.

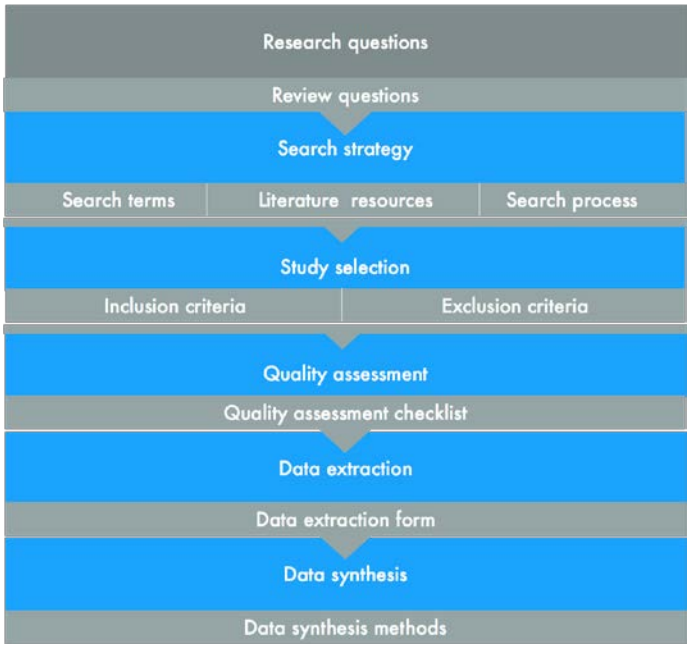

Figure 2. Protocol for conducting SLR in the software engineering domain (from [KIT07]).

2.1. Research questions

Table 1 summarizes the RQs and the motivation to explore them. Once defined the RQs, they were reviewed in order to determine how they would be answered. RQ1 is answered by describing the classifications obtained as result of the searching process and analyzing the structure of each of them; this is presented in Section 3.2. RQ2 is answered by identifying the classification criteria and aggregating them in a way that can be used by practitioners to determine the potential suitability of the SDEE methods to the context of their companies. The answer to RQ2 is presented in Section 3.3.

Table 1. Research questions considered in the SLR.

ID	Research Question	Motivation
RQ1	What are the main taxonomies of SDEE methods reported in the literature?	Determining the main taxonomies allows creating a more comprehensive view of the available proposals, and therefore, increasing the chances to find methods potentially useful for a particular software company, considering the clusters identified in these taxonomies.
RQ2	What are the most relevant classification criteria used in these taxonomies?	Knowing these criteria allows determining the variables that can be used as instruments for evaluating the potential suitability of a certain method, considering the features of a particular software organization.

2.2. Search strategy

The purpose of this stage is to find studies that help answer the stated RQs, and it involves three steps: the definition of the search terms, the definition of the literature resources (i.e., digital libraries to be used) and to conduct the searching process. Next we explain these steps.

2.2.1. Search terms

The definition of the search terms involved several activities in order to obtain the Query Strings (QS) used to retrieve the studies that help answer the RQs. These activities are the following:

1. Identify the most relevant terms considered in the RQs.
2. Identify synonyms or alternatives for these terms.
3. Create the search terms connecting the terms with OR logic operators. This allows finding studies that matches with them.
4. Compose the search terms with an AND operator. This allows finding studies having variants of any search term.
5. Generalize the search terms, if possible, using a wildcard.

Table 2 shows the main query string (QS1) used to look for papers on software effort estimation, and a refined query string (QS2) that select the papers that refer to methods, classifications or taxonomies.

Table 2. Query strings used in the SLR.

Query String - QS1
`(Software OR system* OR project* OR development OR application*) AND (effort OR cost* OR siz*) AND (Estimat* OR Predict* OR Assess* OR Forecast* OR calcula* OR siz* OR measur* OR dimension*)`
Refinement Query String - QS2
`AND (Taxonom* OR Categor* OR Class* OR Process* OR strateg* OR approach* OR method* OR algorithm* OR Metric* OR Unit* OR Review* OR Survey* OR Analys* OR Report* OR Ensemble OR Systematic)`

2.2.2. Literature resources

Four sources of literature were selected based on their relevance for this SLR: IEEE Xplore, ACM Digital Library, Science Direct and Springer Link. The searches considered only publications reported in these libraries since January 2000 to May 2017. The search process in these libraries was automated using a robot, as a way to reduce the effort of conducting the searches and retrieving the resulting information.

2.2.3. Search process

The search and retrieval process was done in two steps by the software robot. In the first one the robot used the QS1 to build the master dataset that then was used in the rest of the SLR process. Given that not all search engines of the digital libraries support a complex query string with logical operators and wildcards, the robot created all the combinations of search terms and resolves all wildcards to concrete terms, and then trigger the search for each combination. The robot processed the HTML in the search responses, identified the articles included in the search results, extracted the metadata of each article, and stored it in a text database in an homogenized format. This homogenization and completion of

metadata was required due to every digital library provides different metadata for the query results. Such a metadata includes: library ID, paper title, abstract, authors, keywords, doi, publication type, year, publisher, and URL to download the paper.

In the second step, the robot applied the QS2 on the metadata stored in the master dataset obtained in the previous step, and produced a list of candidate studies. Such an information was validated by contrasting the search results with a set of previously selected studies, manually identified by the advisors of the author during a previous stage. This validation process shown that every pre-selected study was found and selected by the software robot, indicating the automatic selection was accurate.

2.3. Study selection

In this stage participated the author and also two researchers of the area (advisors of the author). The relevance of each study selected by the robot (i.e., that matches with QS1 and QS2) was rated by two participants, and in case of conflicts with these rates, the third participant decides on the paper relevance. The author rated all studies and the advisors played the other roles.

The relevance of a study was defined using inclusion and exclusion criteria. The inclusion criteria considered: (1) studies that group, classify or enumerate SDEE methods based on one or more criterion, (2) studies indicating properties or quality features that can be used to classify these methods, and (3) previous SLR in the same study domain. The exclusion criteria considered: (1) studies that describe a particular SDEE method, (2) studies that involve various SDEE methods, but do not classify or group them, (3) studies that estimate project variables (e.g., product size or team time), but do not the development effort, and (4) studies already stored in the master dataset (i.e., duplicated articles).

Every paper counted on two reviewers, where each reviewer indicated the paper relevance using the inclusion and exclusion criteria after reading its title, abstract and keywords. In some very few cases it was required to quickly review a study to determine its relevance for this SLR. The relevance of each paper was indicated using three values: *relevant*, *irrelevant*, *uncertain*. Relevant means the study satisfies at least one of the inclusion criteria and none of the exclusion ones. Irrelevant means the study satisfies at least an exclusion criterion. A study's relevance is uncertain in any other case.

2.4. Quality assessment

In order to improve the selection done in the previous stage, we created a validation list based on the questions indicated in Table 3. Each question (for each study) was answered as "yes", "partially" and "no", which correspond to 1, 0.5 and 0 respectively, according to the recommendation of Wen et al. [WEN12]. The final score of each study (i.e., its quality indicator) was obtained by adding the scores assigned individually to each quality assessment question for such a paper. Similar to the previous stage, the quality assessment of each study was performed by two people, and in case of conflict the third person decided.

After conducting the quality assessment, we selected studies considered "relevant" and with a quality indicator over the 50% of the perfect score. This decision adheres to the recommendation given in [WEN12].

Table 3. Quality assessment checklist (based on [WEN12]).

ID	Quality Assessment Questions
QA1	Is the study published in a conference or journal relevant in the study domain?
QA2	Has the article clearly defined its goals and contributions?
QA3	Are the paper contributions supported by evidence?
QA4	Is the work useful for the industry?

2.5. Data extraction

Once obtained the final list of studies to be used for answering the RQs, we filled a data extraction form (recommended in [KIT07]) that includes for each study: study identifier, publication year, author name(s), source, URL, article title, and number of citations. The form helps organize and index the information. In our case, such an information was retrieved by the software robot and reviewed by the author. All information required to fill the forms was available in the already collected metadata, except the number of citations that was obtained from Google Scholar. Moreover, a set of key aspects were defined for each research question to help the reviewer identify the parts of the paper that are relevant for the SLR (i.e., for answering the questions). These terms are shown in Table 4.

Table 4. Relevant terms for the RQs.

RQ1 - What are the main taxonomies of SDEE methods reported in the literature?
● Classifications or taxonomies reported in the study
● Coverage of the taxonomy according to the authors
● List of SDEE methods being classified
RQ2 - What are the most relevant classification criteria used in these taxonomies?
● Variables, dimensions or classification criteria reported in the study
● Values reported for each variable/dimension/classification criterion
● Strategy to classify methods using the reported variables/dimensions/classification criteria

2.6. Data synthesis

After performing the information retrieval, such information was grouped and organized to help answer the RQs and determine the generalization of the results. In order to do that, two synthesis approaches were used: narrative and quantitative synthesis. The first one was used to answer RQ1 since it allows summarizing the results, and also visualizing and explaining the taxonomies. For answering RQ2, a quantitative synthesis was used as it allows to visually represent the dimensions that are most frequently used to classify SDEE methods.

3. Obtained Results

Figure 3 illustrates the SLR process and the number of studies retrieved in each stage (the step labels are indicated with circles). The automated search using the first query string (QS1) in the four sources of literature retrieved 1,442,159 documents (including duplicate papers). After removing the duplicated articles and then applying the *QS2,* we obtained 1,303 papers that were analyzed considering the inclusion and exclusion criteria. These criteria were applied after reading the title, abstract and keywords of each article, and only in few cases the paper was fully reviewed. As a result of this step, a set of 35 studies was identified, which was reduced to 17 papers after applying the evaluation criteria presented in Table 3.

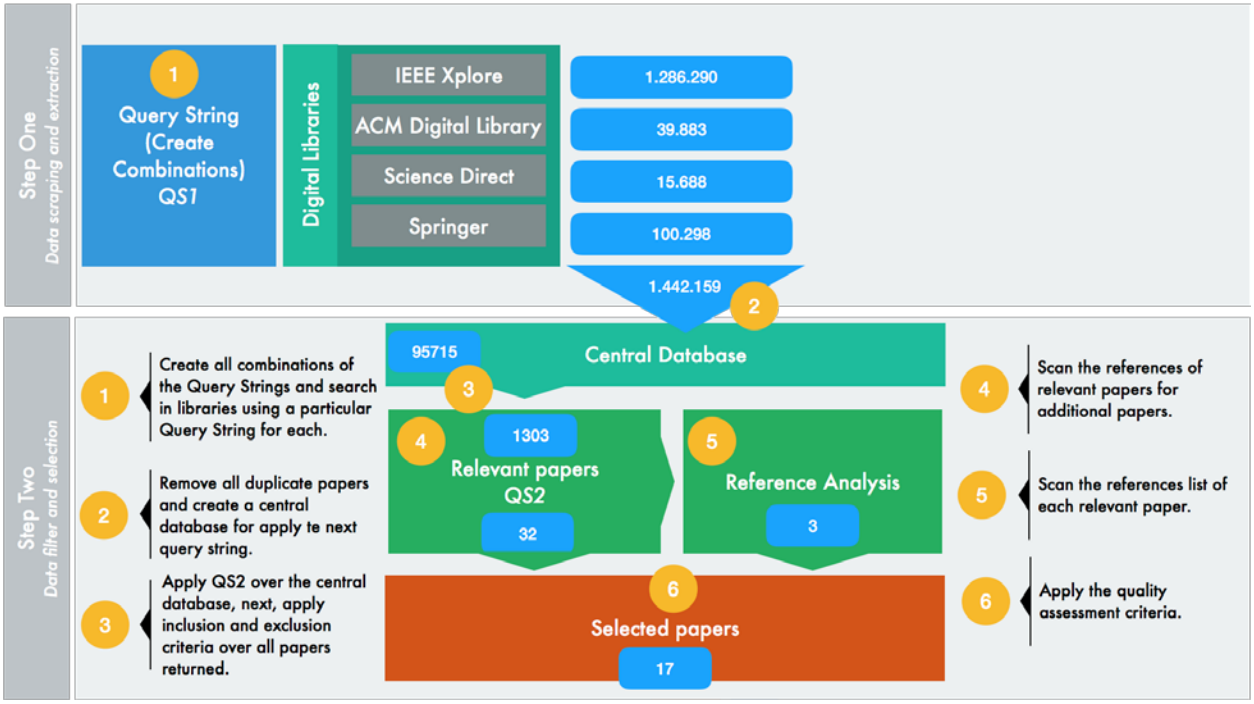

Figure 3. Data extraction strategy (from [KIT07]).

Considering the total of 17 selected articles, 11 of them were published in journals, 3 in conferences, 2 as book chapter and 1 in a symposium. Table 5 shows the sources of the pre-selected articles.

Table 5. Sources of the selected studies

Publication source	Type	# of studies
The Journal of Systems and Software	journal	3
Information and Software Technology	journal	2
Journal of Computer Science and Technology	journal	1
Indian Journal of Science and Technology	journal	1
Annals of Software Engineering	journal	1

IEEE Transactions on Software Engineering	journal	3
International Conference on Global Software Engineering	conference	1
International Conference on Software Engineering (Companion Volume)	conference	1
International Conference on Predictive Models in Software Engineering	conference	1
International Symposium on Empirical Software Engineering	symposium	1
Encyclopedia of Software Engineering	book chapter	1
Advances in Computers	book chapter	1

Figure 4 shows the distribution of papers according to the publication year. The number of publications remains stable and quite low.

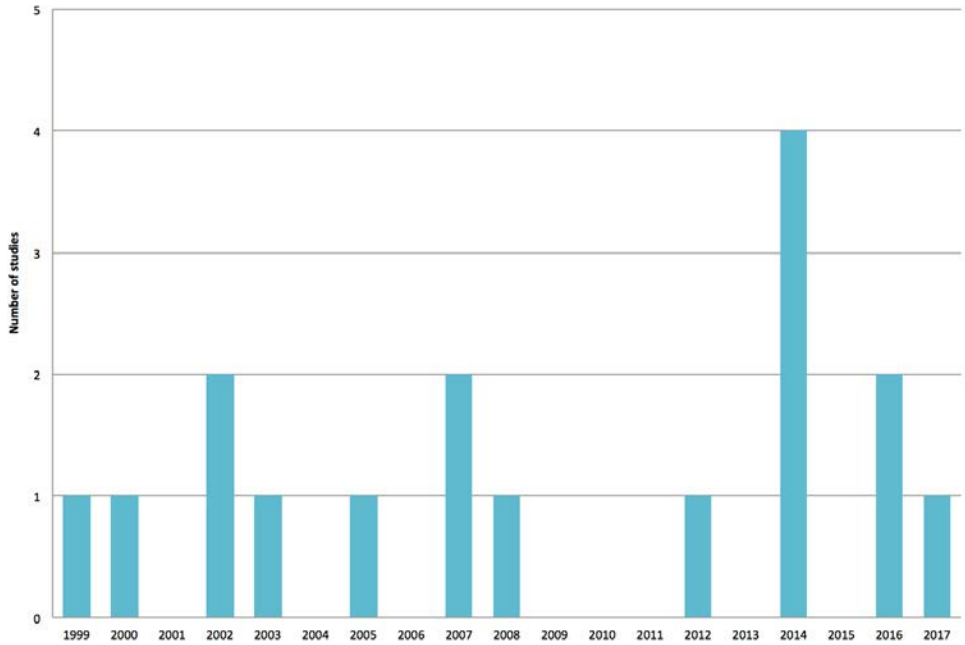

Figure 4. Distribution of selected studies by year.

3.2. SDEE method classifications and taxonomies (RQ1)

Typically the classifications and taxonomies are used to organize knowledge in particular domains, and they can be broad (large coverage) or specific (small coverage). Considering the papers selected for this study, 10 of them (59%) are broad and the other 7 (41%) are focused on describing in detail some of the previous categories. Next we first describe the general taxonomies (Section 3.2.1) and then the specific ones (Section 3.2.2).

3.2.1. General Classifications

The publishing date of the articles reporting general classification ranges from 2000 to 2017. Seven of them (78%) present a flat classification with only one level and the other three articles include several levels and subclassifications. These classifications group the SDEE techniques from four points of view: technique-based, model-based, based on the volume of supporting data, and hybrids. Next we briefly explain each of them.

Classifications based on Techniques

According to Bohem et al. [BOE99], software economics involves two main areas: (1) software cost and schedule estimation, and (2) software decision support. The first one is relevant for this study, and it considers six categories (Fig. 5): expertise-based, model-based, regression-based, composite-Bayesian, learning-oriented, and dynamics-based.

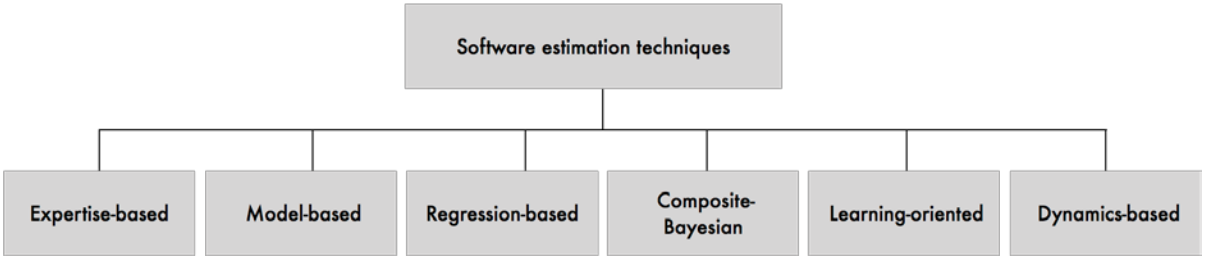

Figure 5. Classification proposed by Bohem et al. [BOE99].

In the category *expertise-based* (or *expert judgment*) techniques the main protagonists are the experts; i.e., the people that know about the business and the technical domain, and are in charge of generating the cost and time estimations using their expertise and doing analogies with their previous projects. The *model-based* techniques (or parametric models [BOE00]) are supported by a model that typically involves a set of variables, parameters and relationships among them. These models need to be set according the company and project features (this also includes the characteristics of the development team) in order to obtain accurate estimations. The *regression-based* estimation methods use linear regressions (or similar) to make an estimate, and they require to count on an important volume of historical information, that should also be relevant for the project being estimated. Moreover, they also require to count on configurations that indicate the correlation between variables of the previous and the current project. The *composite-Bayesian* (also known as hybrid or composite [BOE00]) techniques allows using expertise-based techniques complemented with model-based ones. Thus, the experts can adjust or justify their estimations supported by the extra information provided by particular estimation models. The *learning-oriented* methods are those that include adaptive learning in order to improve their estimations based on the results of previous projects. These techniques represent an alternative to regression-based estimations. The *dynamics-based* methods are used to adjust the estimations for a project development while the project is being run and according to the history of such a project; i.e., these methods are focused on producing re-estimations.

Moløkken et al. [MOL03] use a different perspective and propose a taxonomy of SDEE methods with three categories (Fig. 6): *expert-based*, *model-based* and *other*. Similar to the previous taxonomy, expert-based include techniques where a person estimates using intuition and making analogy with previous projects. The model-based estimation techniques group pure models like SEER [JEN83] and CoCoMO [BOE81], and the “other” category groups the non-pure models like those based in price-to-win [BER92].

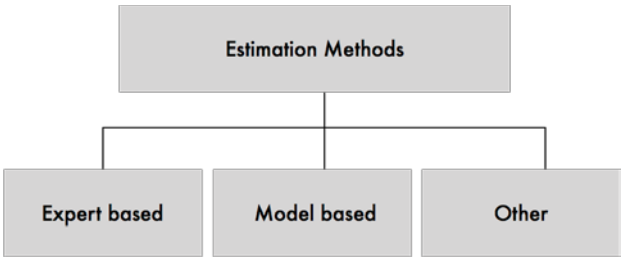

Figure 6. Classification proposed by Moløkken et al. [MOL03].

Jørgensen et al. [JOR07] performed a systematic literature review on software development cost estimations and proposed twelve categories of techniques (Fig. 7). Various of them were identified in previous proposals and already introduced in this document, and the rest of the approaches are briefly explained next.

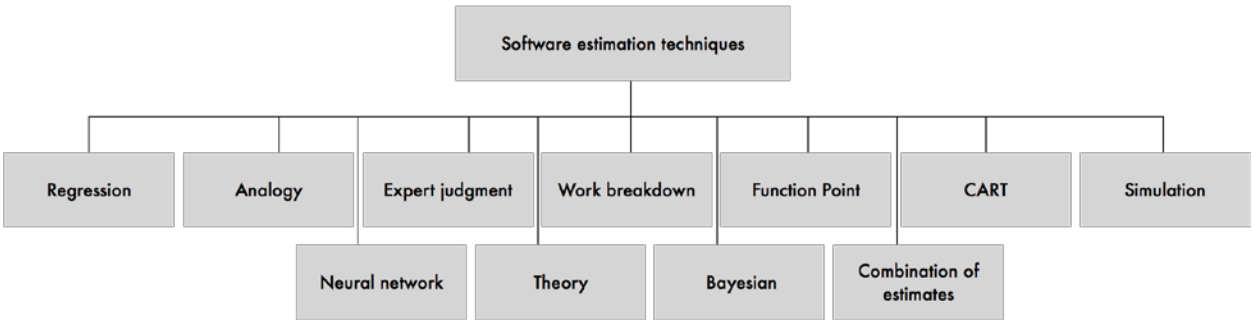

Figure 7. Classification proposed by Jørgensen et al. [JOR07].

The *neural network-based* methods consider the use of these tools and also artificial intelligence to try guessing projects estimations. The methods *based on theory* are constructs derived from theoretical knowledge about the project estimations, like SLIM [LAW91]. The *work breakdown* techniques take advantage of the decomposition of development activities, reducing thus the size of the activities to estimate and generating more accurate estimations. The *function point* methods include variants to the Albretch proposal [ALB79]; e.g., feature points and use case points. These methods use a unit of measurement (function points or similar) to express the effort required to develop a software. The *CART* (Classification and Regression Trees) methods use classification and regression trees to generate the estimations, and the *simulation* techniques include the use of simulation models, like Monte Carlo.

In 2014 Britto et al. [BRI14] performed a systematic literature review on SDEE methods focused on global development, and classified the estimation techniques in three categories (Fig. 8): expert-based, algorithmic-based (or model-based), and artificial intelligence approaches. This taxonomy is general and quite similar to the one proposed by Moløkken et al. [MOL03].

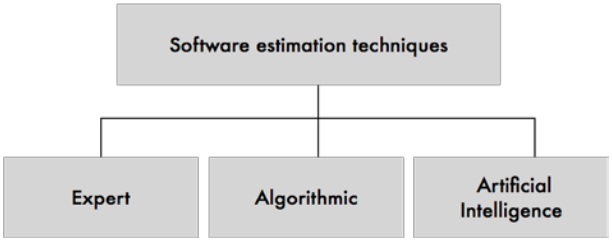

Figure 8. Classification proposed by Brito et al. [BRI14].

Rajper et al. [RAJ16] (Fig. 9) uses the classification proposed in [BOE00] as a basis to report the results of a systematic literature review on SDEE methods. Although these categories are mainly the same than those discussed in the previous works, this proposal introduces a category named “data mining” techniques that use such an approach on a large amount of historical information to generate the estimations. Dejaeger et al. [DEJ12] review several data mining techniques for conducting SDEE. The work of Rajper et al. [RAJ16] also establishes subcategories for the composite techniques by grouping them in three subcategories: machine learning-based, parametric models and Bayesian techniques.

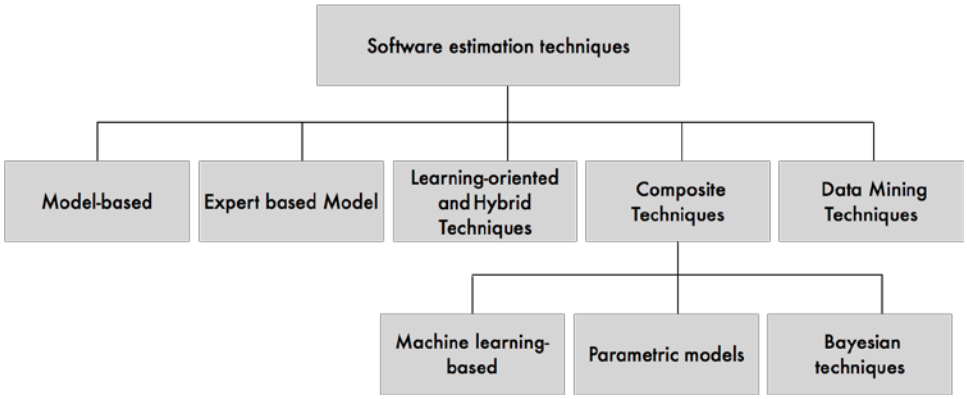

Figure 9. Classification proposed by Rajper et al. [RAJ16].

The last general taxonomy of SDEE was reported by Boehm [BOE17], and it considers six categories of methods (Fig. 10): expert-judgement, analogy, parametric models, resource-limited, reuse-driven, and product line. The last three categories were identified in previous works but never included formally in a taxonomy. Next we briefly explain each of them. The *resource-limited* techniques consider that cost or time in a project are predefined (that is the limited resource), and therefore the technique is used to estimate the non-predefined variable. In this case, project cost and time are considered as independent variables [BOE00-2]. The *reuse-driven* methods allows the reuse of the historical information in project with similar size and characteristics than the new one [BOE00-2, POU96]. This strategy considers tuning of the estimation, considering the effort by phase required in the new project. The product line techniques consider the effort required to develop other products of the product line, as input to estimate the new ones [BOE00-2, POU96].

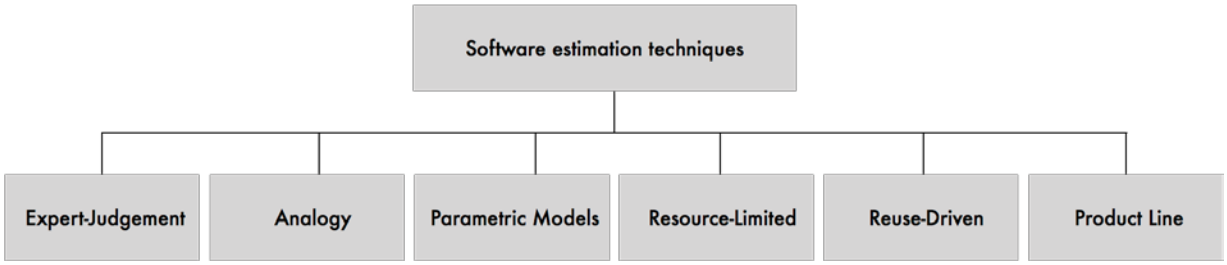

Figure 10. New classification proposed by Bohem [BOE17]

Classifications based on amount of available data

One of the first questions to answer when selecting a technique to estimate a project is how much historical information we have to support such an activity. The answer establishes the set of alternatives that are more suitable to conduct the estimation. Considering this aspect, Myrtveit et al. [MYR05] propose a taxonomy that reclassifies the existing SDEE categories indicating if they require sparse or many data.

As shown in Figure 11, most categories of estimation techniques require an important amount of historical information.

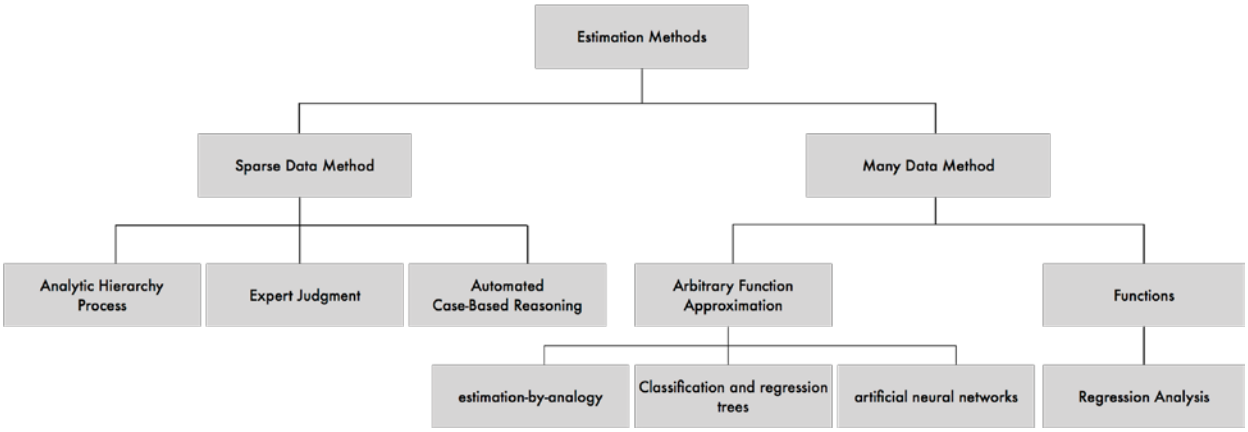

Figure 11. Classification based on the volume of available data (based on [MYR05]).

Kläs et al. [KLA08] propose other alternative to such taxonomy and classify the SDEE methods in three categories, considering not only the amount of required data, but also the way in which such information is used. Particularly, the proposed categories are (Fig. 12): *data-driven*, *expert-based* and *hybrid*. The first ones require an important amount of data, which can be used to conduct analogies, or feed or tune a model (including composite models). Models in this category require to count on formal records of the historical information. The second category use few or informally recorded information, and it is the expert who analyzes, prioritizes and interprets it in order to produce an estimation. The last category mixes the two previous ones.

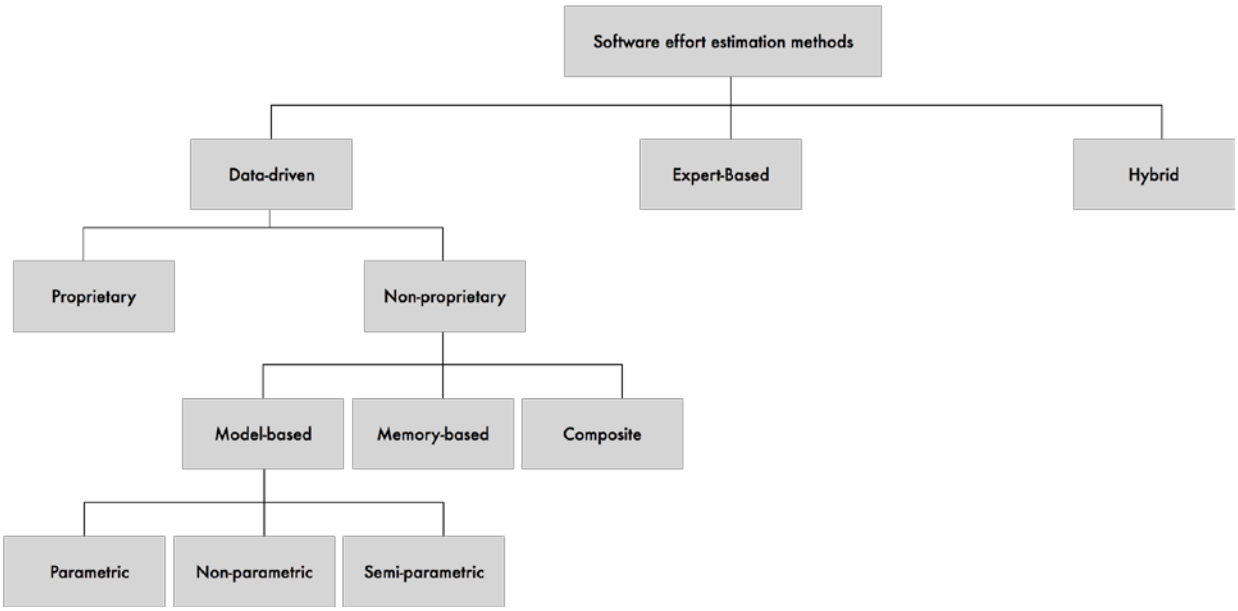

Figure 12. Classification proposed by Kläs et al. [KLA08].

Classifications based on models

Similar to the previous proposal, but using a different reclassification criterion, Briand et al. [BRI02] establish a taxonomy considering if the SDEE techniques corresponds to *model-based* methods (Fig. 13).

This article proposes to classify them in generic and *specific model-based,* which is a categorization that was not reported in previous works. The *generic model-based* techniques are potentially suitable to be used in any work context (project or product), and they can be proprietary or non-proprietary. While in the first case the estimation model and the way of using it are not publicly available, in other case the methods are non-proprietary.

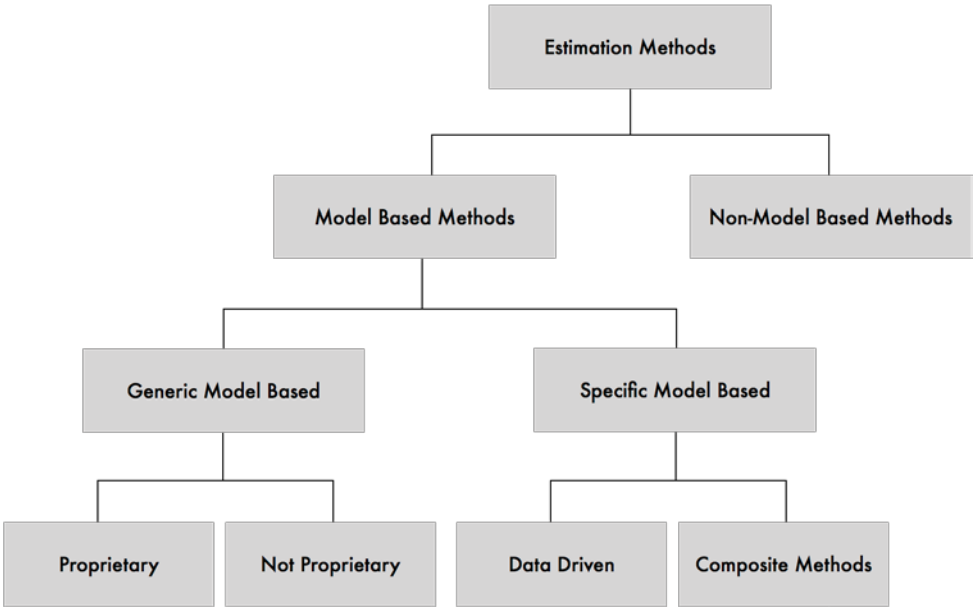

Figure 13. Classification proposed by Briand et al. [BRI02]

Concerning the *specific model-based* techniques, they use models that are valid on particular project contexts. Data driven and composite methods are example of specific model-based techniques. Concerning the *non-model based* methods, they involve the use of one or more techniques that do not involve models (e.g., expert judgement), and require to count on a procedure that indicates how to use the technique. This is particularly required when more than one technique is going to be combined.

3.2.2. Specific classifications

As mentioned before, seven of the selected articles address specific classifications (or categories) and propose sub-classifications for them. The specific categories to be analyzed are the following: *expert judgement*, *semi-parametric approaches*, *machine learning*, *focused on data mining*, *analogy-based*, *ensemble* and *agile techniques*. Next we briefly explain each of them.

Classification of semi-parametric models

Nikolaos et al. [NIK15] address the challenge of classifying semi-parametric SDEE methods (Fig. 14). They use as a basis the taxonomy of Myrtveit et al. [MYR05] (shown in Fig. 11) that identifies two initial classes of techniques: sparse data or many data. The first category includes methods based on expert-judgment or similar, and they were not reviewed in Nikolaos et al. study. The second category considers three dimensions of techniques: parametric, non-parametric and semi-parametric. This last category is the focus of their study, where the authors intend to determine how to combine parametric and non-parametric models in a suitable way. In order to do that they extend the model proposed in [NIK10] by combining linear and non-linear components in two steps. In the first step they suggest to use the following non-parametric methods: analogy-based, locally weighted regression, neural networks and support vector machine. Although the same model is also applied in the second step, it is recommended

that the parametric component is estimated using some of the following regressions: least median of squares, robust m-estimator, least trimmed and ridge.

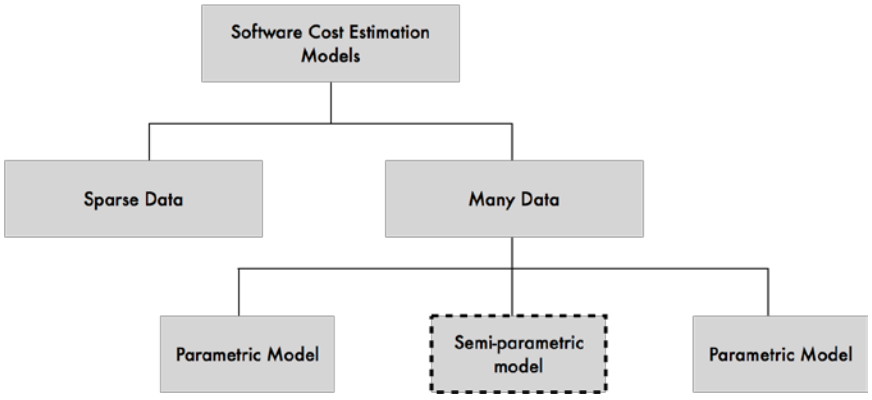

Figure 14. Classification of semi-parametric models (based on [NIK15]).

SDEE methods based on expert judgement

Jørgensen [JOR04] reviews several studies on SDEE methods based on expert judgement and defines twelve ‘‘best practice’’ guidelines (or principles) to conduct this type of estimation: (1) evaluate estimation accuracy, but avoid high evaluation pressure; (2) avoid conflicting estimation goals; (3) ask the estimators to justify and criticize their estimates; (4) avoid irrelevant and unreliable estimation information; (5) use documented data from previous development tasks; (6) find estimation experts with relevant domain background and good estimation records; (7) estimate top-down and bottom-up, independently of each other; (8) use estimation checklists; (9) combine estimates from different experts and estimation strategies; (10) assess the uncertainty of the estimate; (11) provide feedback on estimation accuracy and development task relations; and (12) provide estimation training opportunities.

The use of all or part of these principles lead us to structured estimation methods, where the expert can use historical data and also supporting instruments like validation lists or feedback tables. This category includes most of the methods based on expert judgement that are reported in the literature (e.g., Wideband Delphi or Poker Planning). Contrarily, the lack of principles or processes for conducting the estimations correspond to techniques classified as unaided intuition (‘‘gut feeling’’). In this case, the expert counts on mainly his intuition and memory to conduct such an activity. Figure 15 summarizes this taxonomy.

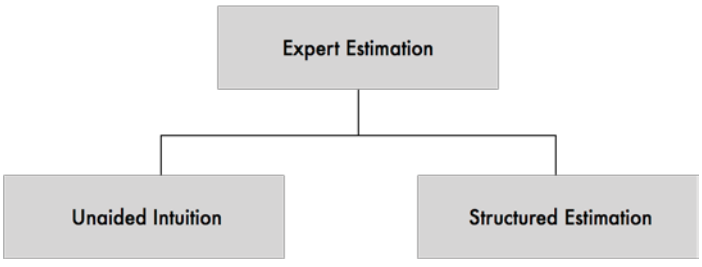

Figure 15. Classification of SDEE methods based on expert judgement (based on [JOR04]).

Classification of combined techniques (ensemble)

Ali Idri et al. [ALI16] show that it is possible to create ensemble estimation methods, when the composed techniques presents certain characteristics. In order to propose an ensemble method, it is required not only to identify the techniques that are going to be used as base for the new method, but also the components from those techniques to be combined and the set of combination rules to be used in the fusion. After that, we are in condition to specify the ensemble technique.

The ensemble methods can be classified in two categories (Fig. 16) [ALI16]: *homogeneous* and *heterogeneous* techniques. The homogeneous ensemble methods can belong to two categories: the *single base* model and the *ensemble learning with base* model. The techniques in the first category use a single method as the base, and at least two configurations. The techniques in the second category include the combination of a base model and a predictor learning model like Bagging [BRE96], negative correlation [YAO99] o random [HAL09]. If we use as base model a combination of two different methods, then we obtain an heterogeneous ensemble method.

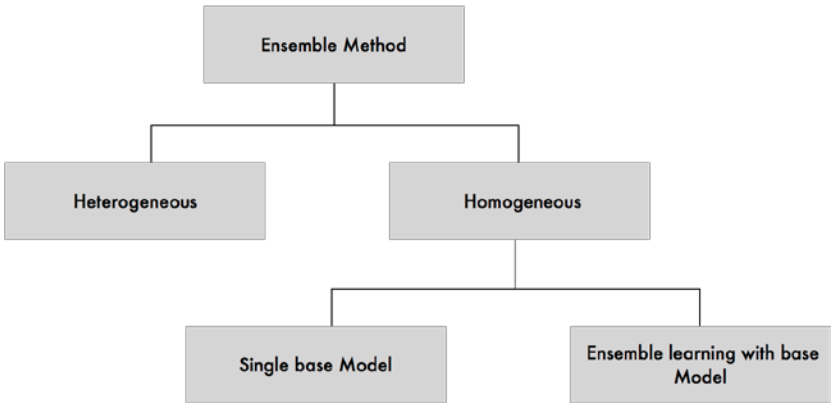

Figure 16. Classification techniques that combine SDEE methods (based on [ALI16]).

Classification of estimation techniques based on machine learning

The “learning-oriented” category was proposed by Boehm and Sullivan [BOE99] and then used in several taxonomies. Particularly, Wen et al. [WEN12] use this category to separate the SDEE methods that use machine learning (ML), in pure form or in hybrid form (Fig. 17). The techniques that involve ML can be pure or hybrid, where the former involves the use of a single ML technique, and the latter combines two or more ML methods. Moreover, Wen [WEN12] identify eight pure ML techniques that are shown in Figure 17.

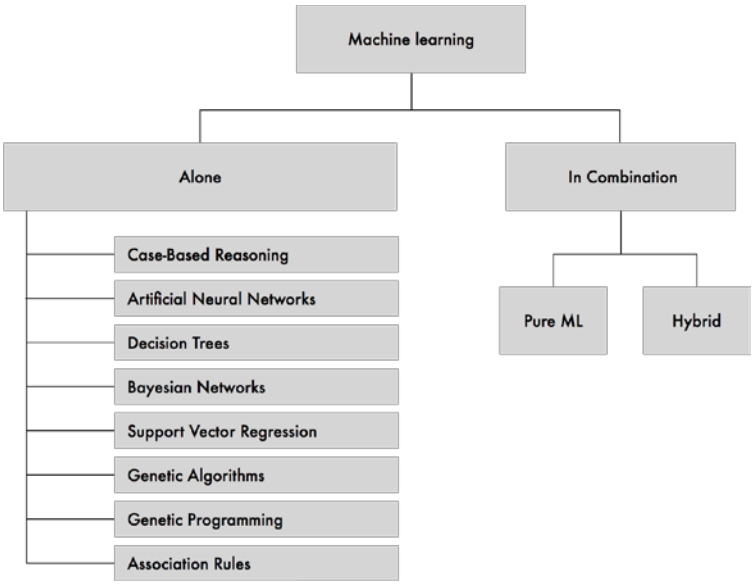

Figure 17. Classification of estimations based on machine learning (based on [WEN12]).

Classifications based on analogies

Figure 18 shows other classification proposed by Ali Idri et al. [IDR15] for SDEE techniques based on analogies, and it is particularly interesting as it is the first proposal detailing such a category. According to their work, it is possible to identify two main groups: analogy-based techniques that are used alone, and those that are combined with ML-based or non-ML-based models. This combination of models should adhere to a set of fusion rules as recommended in [WEN12].

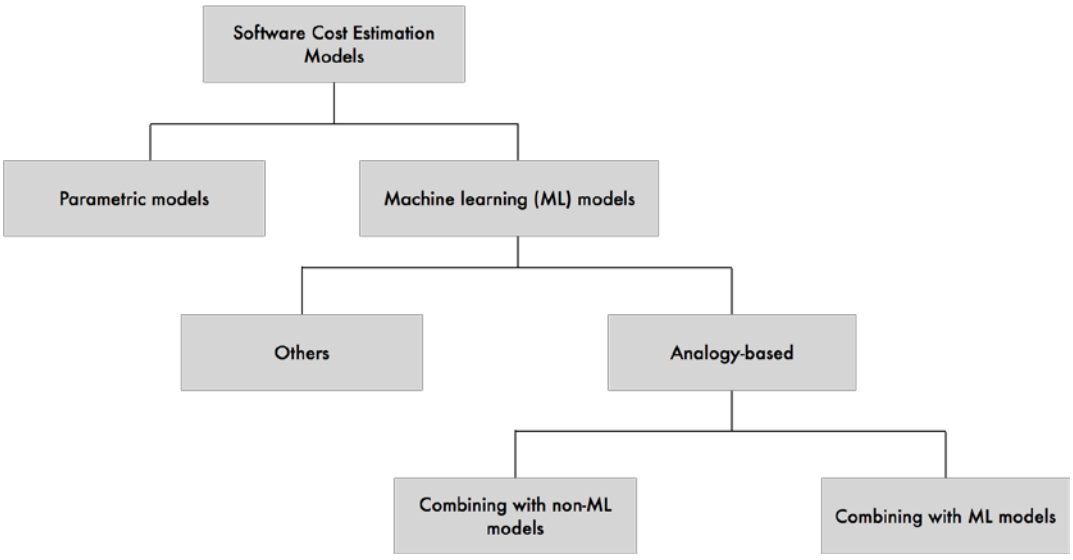

Figure 18. Classification of techniques based on analogies (based on [IDR15]).

SDEE methods focused on data mining

There seems to be no consensus about when a massive data processing technique should be classified as a data mining technique. Dejaeger et al. [DEJ12] shows thirteen techniques that can be included under the data mining umbrella (Fig 19); many of them were also reported by Wen et al. [WEN12].

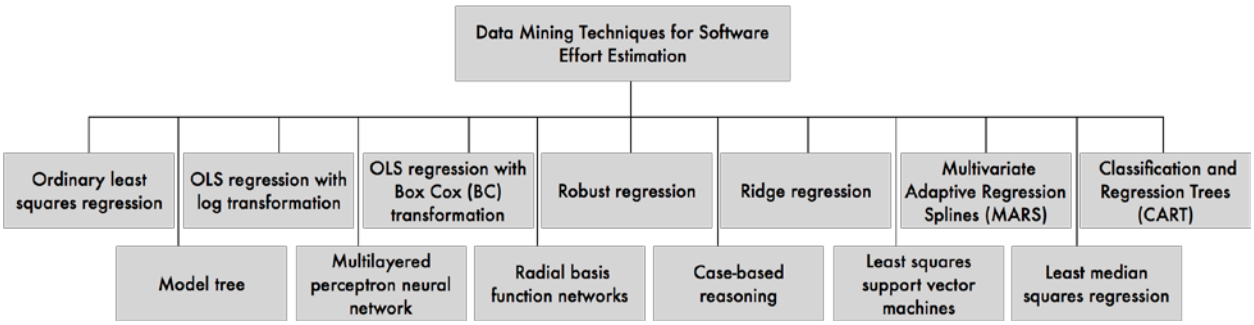

Figure 19. SDEE methods based on data mining (based on [DEJ12]).

According to these authors, the generic SDEE techniques require to use data mining to improve their accuracy; however, counting on such volume of information represents a challenge for most software companies. The work of Dejaeger et al. also indicates the amount of data that is required to get a suitable estimation, and the accuracy that we can expect from every technique. This helps the reader chose the most suitable data mining technique according to the reality of each software company.

SDEE methods for agile development

During the last years there have been many proposals to support estimations in agile software development (ASD); however, few classifications have been reported in the literature. The most interesting one is probably the taxonomy proposed by Usman et al. [USM14] that considers five main categories and specific methods in some of them (Fig. 20). They classification that they propose is the result of a systematic literature review.

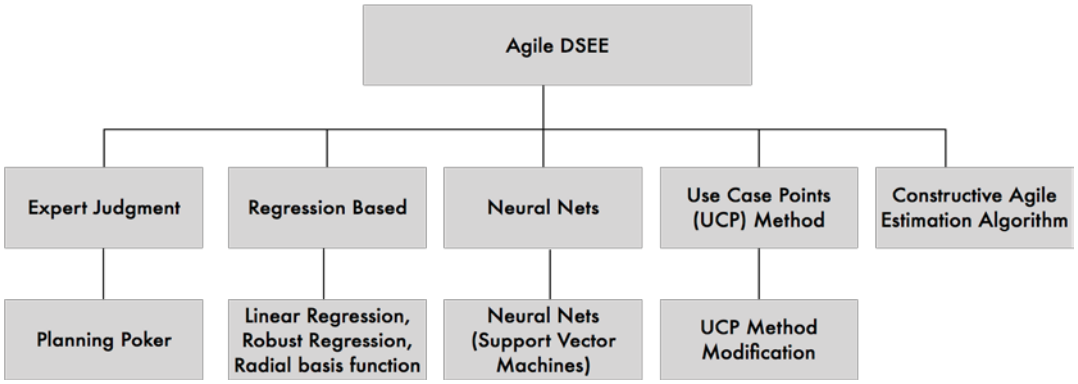

Figure 20. SDEE methods for agile development (based on [USM 14]).

Most of these categories have been already explained in previous taxonomies, but in the context of ASD, some of them have a different flavor. For instance, expert judgement tend to be more collaborative in ASD than in other development scenarios.

An important aspect of the work reported by Usman et al. is the frequency of usage of these techniques. Particularly, expert judgment, planning poker, use case points and variants of this latter are the most frequently used techniques. Other methods like techniques based on regression, neural networks and the constructive agile estimation algorithm are less frequently used. It is important to remark that in this case we are considering only techniques suitable for ASD environments, and therefore, most of the previous ones are not suitable or still not used in this development domain.

3.3. Classification criteria used in the reported taxonomies

The seventeen articles considered in this study used several criteria to classify the SDEE techniques. Fourteen of them consider expert-judgment and the use of parametric/non-parametric models, eleven use artificial intelligence as a classification criterion, nine use combinations (ensemble), eight papers classify the techniques depending on whether they use or not a model (or algorithm) to estimate, five consider the use of regressions or analogies as classification criteria, and four consider the use of intellectual property over the estimation techniques. Next we explain each classification criterion, and then present a list of the 119 SDEE techniques (gathered from this study) classified using these criteria.

3.3.1. Details of classification criteria

Sixteen specific criteria were obtained from the selected studies: expert estimation, data driven, parametric, model-based, regression-based, composite, artificial intelligence, analogy-based, proprietary, ensemble, functionality-based, machine learning, dynamics-based, resource-limited, reuse-driven, and product line. The nine most representative ones (i.e. are included in at least two taxonomies) are enough to create a general classification that allows to catalog properly the collected SDEE techniques (119 in total). Next we explain each criterion.

- *License*. This criterion is used to classify techniques considering whether its implementation details are public, accessible under license or inaccessible. Typically, depending on it, the techniques are classified as *proprietary* or *non-proprietary.*

- *Data.* This criterion indicates how much data is required to use the techniques. Using this criterion, the SDEE methods are classified as requiring *many data, sparse data* or *no data.*

- *Parametric*. Some techniques require to calibrate or configure some parameters before being used for generating an estimation. This criterion determines whether a particular SDEE method requires this process, and consequently, it can be labeled as *parametric, semi-parametric* or *non-parametric.*

- *Expert.* This criterion indicates that a technique is based on the experience or judgment of a person (the estimator). Depending on it, the technique is classified as *expert-based (an expert is required)* or *No need Expert (no expert is required)*.

- *Model*. It determines if a SDEE technique is based or not in a model: *model-based* and *non-model based* respectively.

- *Analogy.* This criterion determines the feasibility of reusing (partially or completely) previous estimations to generate new ones, by making an analogy between the new project and the past experiences. Depending on it, the SDEE methods can be classified as a*nalogy-based* or *other* (*non-analogy-based).*

- *Regression*. This criterion considers a method as *regression-based* when it uses some type of regression on historical data to generate an estimation. In other case, it is considered as *non-regression-based*.

- *Composite (or ensemble)*. It indicates whether the technique is *pure* or the result of fusioning two or more existing techniques (or part of them). In the latter case the techniques are known as *composite* or *ensemble*.

- *Artificial intelligence*. A SDEE technique is considered as *based on artificial intelligence* when it uses methods or algorithms from such a knowledge domain for generating the estimations. In other case, it is considered as *non-based on artificial intelligence.*

3.3.2. Classification of SDEE techniques

Considering the previous classification criteria and the 119 SDEE techniques identified in the studied papers, we classified each technique using these criteria. We added the value 'unknown' to each criterion to indicate that for a given method, we could not determine how to classify it according to such a criterion. Table 6 shows the summary of this classification; the details are available in the annex. The summary indicate the number of techniques reported by category.

Table 6. Classification of techniques based on suggested criteria.

Classification Criteria	Licence			Data				Parametric				Expert			Model			Analogy			Regression			Composite			Artif. Intellig.		
SDEE Technique / Classification Values	Proprietary	Non-Proprietary	Unknown	Many data	Sparse Data	No Data	Unknown	Parametric	Non-Parametric	Semi-Parametric	Unknown	Need Expert	No Need Expert	Unknown	Model-Based	No Model-Based	Unknown	Analogy-Based	No Analogy-Based	Unknown	Regression-Based	No Regression-Based	Unknown	Composite Technique	Single Technique	Unknown	AI-Based	No AI-Based	Unknown
SUMMARY	17	101	1	102	11	6	0	35	36	19	29	20	98	1	97	20	2	16	91	12	36	59	24	70	32	17	37	63	19

These numbers indicate that most of the reported techniques are non-property, do not need an expert and consequently they require many data. Moreover, most techniques are based on models and do not use analogies, regressions or artificial intelligence mechanisms. It is important to remark that this summary only indicates that most research has been (or is being) done on methods that adhere to such characteristics, that usually correspond to those used to estimate large projects. Moreover, these methods are less people-dependent and also less accurate in considering the relevant historical data. This seems to indicate that the research efforts are more focused on using many data and much computing processing in order to reduce the dependence on subjective components like the estimators (experts) and highly contextualized historical information. This is quite surprising considering the literature reports that expert judgement is the most frequently used SDEE approach [JOR14, MOS00, TRE08, USM14], and Table 6 shows no much research work on it.

Particularly, small and medium-sized companies, that represents most of the software development capacity at global level [ARA10, LAP08, WAN06], have less chance to use techniques based on many data simply because they do not have much historical data to use. They typically use sparse or no data, expert-based (i.e., no model-based) and analogy-based methods to produce their estimations (in blue in Table 6) [MOS00, TRE08, JOR14]. Therefore this study shows an opportunity to do research and improve software estimation support in such an application domain.

4. Discussion

In this section we revisit the stated RQs and provide an answer to them based on the results of this study. Moreover, we discuss the usefulness of these taxonomies for small software companies (less than 50 people). Studying this group of companies is relevant as they represent about 70% of companies in Brazil [WAN06], 95% in the USA [ARA10], and over 80% in Canada [LAP08] and Chile [VAR03, OTE05].

4.1. Revisiting RQ1

After analyzing the 17 articles selected in the systematic literature review, we can provide an answer to the research questions stated in Section 1. Concerning the RQ1 (*What are the main taxonomies of SDEE methods reported in the literature?)* it was not possible to determine one or more "main" taxonomies since the classification criteria they use and the way they represent the categories of SDEE technique is too diverse. However, this study shows some aspects of these taxonomies that are worth remarking.

- *The taxonomy structure*. All taxonomies use a hierarchical structure that leads to overlap SDEE techniques or overgeneralization. Moreover, in many cases the authors include a category "other" (like in [REI17], [IDR15], [MOL03]) that is ambiguous and limits the ability to enumerate the techniques under such a category. Other limitation of using hierarchical taxonomies is that some techniques are ensemble or hybrid, and therefore it is almost impossible to classify them properly in this kind of structure.

- *Poorly contextualized classifications*. Most classifications are stated as independent of the application domain of the estimation techniques, except those that were defined to be used in specific domains. Contextual aspects like the project or company size, type of development approach or business limitations are not considered by these techniques. Considering these aspects would help organizations select appropriate SDEE techniques to their contexts and also transfer the expertise usually required to conduct the estimations [EAR01].

- *Classification coverage*. Most taxonomies were defined using too general categories in order to have large coverage (e.g., [BOE99], [MOL03], [JOR07], [BRI14], [RAJ16], [BOE17], [MYR05]). This allows one to classify almost any technique, but it is not much useful to help organizations identify appropriate SDEE methods to use in a particular context or project. This lack of precision to classify the techniques can be addressed using a feature model in which each method can be classified using multiple criteria, as shown in the table presented in the annex and that we summarized in Table 6.

Summarizing, there is not a taxonomy widely accepted to classify the SDEE techniques (RQ1), and the reasons seem to be two. The first reason is that there are too many relevant and almost independent classification criteria. This means that each technique has a value in each dimension considered in the classification. Considering this fact, it becomes evident that hierarchical representations are not suitable to specify these taxonomies, which is the second reason.

Trying to address the motivation to find an answer to RQ1, this article shows and prioritizes the criteria reported in the literature for classifying the SDEE techniques. Moreover, it also shows the result of using a feature model to classify these techniques (see annex), which seems to be more appropriate that the hierarchical representation, at least for choosing a SDEE method considering the characteristics of any given company or project.

4.2. Revisiting RQ2

Concerning the RQ2 (*What are the most relevant classification criteria of used in these taxonomies?*), the study results indicate that most articles consider the participation of an expert and also the use parametric/non-parametric models, as the main classification criteria. Their relevance is given not only because of their citation frequency (82% of the revised articles considered these classification criteria), but also their position in the hierarchical structure that represents each taxonomy.

It is important to remark that these criteria are relevant for the scientific community, but we do not know their relevance to the software industry. In this sense, a field study is required to answer this question, which should consider several clusters of organizations, e.g., according to company size, maturity level, development approach, and also culture. Several works indicate that expert-based SDEE techniques are the most currently used in the software industry [MOS00, TRE08, JOR14]. However, the literature reports broad or no subclassification for such a category. This represents an opportunity to advance the state of the art, by helping organize the existing knowledge in such a study domain and also identifying the needs that are still not studied by the software engineering community.

4.3. Supporting the selection of SDEE methods for small software companies

In order to determine SDEE methods that are potentially suitable for being used in a certain software organization, it is important to understand the requirements for using each method, and also determine if they can be addressed by the target organization. Recognizing that there is a large variety of small software companies, we characterize these organizations assuming that they have few or no historical information to support their estimations [TRE08], and they use expert-based estimation techniques [MOS00, TRE08, JOR14, USM14]; i.e., they do not use a model and the estimation procedures are public at least for the company personnel. Then, considering these characteristics to filter the requirements of the SDEE techniques shown in the annex, it is possible to obtain the set of estimation methods that matches with these characteristics (Table 7).

Table 7. SDEE techniques potentially suitable for small software organizations.

Techniques	Classificassion Dimensions	Licence			Data				Parametric				Expert			Model			Analogy			Regression			Composite			Artificial Intelligence		
	Techniques / Values	Proprietary	Non-Proprietary	Unknown	Many data	Sparse Data	No Data	Unknown	Parametric	Non-Parametric	Semi-Parametric	Unknown	Need Expert	No Need Expert	Unknown	Model-Based	No Model-Based	Unknown	Analogy-Based	No Analogy-Based	Unknown	Regression-Based	No Regression-Based	Unknown	Composite Technique	Single Technique	Unknown	AI-Based	No AI-Based	Unknown
	Delphi		x				x			x			x				x			x			x			x			x	
	Work Breakdown Structure (WBS)		x				x			x			x				x			x			x			x			x	
	Rule Based			x		x						x	x					x			x			x			x			x
	Guesstimation		x				x			x			x				x			x			x			x			x	
	Estimeeting		x			x				x			x				x		x				x			x			x	
	Planning Game		x				x			x			x				x			x			x			x			x	
	Expert Judgment		x				x			x			x				x			x			x			x			x	
	Analytic Hierarchy Process (AHP)		x			x				x			x				x			x			x			x			x	
	Planning Poker		x				x			x			x				x			x			x			x			x	
	Constructive Agile Estimation Algorithm		x			x			x				x			x				x		x			x				x	
	COBRA – Cost Estimation Benchmarking and Risk Analysis	x				x			x				x			x				x					x				x	
	Colaborative Filtering		x			x					x		x				x		x					x		x				x
	Use Case Points (UCP)		x			x					x		x			x			x				x		x				x	
	UCP Method Modification		x			x					x		x			x			x				x		x				x	
	Bagging + M5P/Model trees (MT)		x			x						x		x		x				x			x		x			x		
	Random + MLP		x			x						x		x		x				x			x		x			x		
	Bagging + Adaptive neuro fuzzy inference system (ANFIS)		x			x						x		x		x				x		x			x				x	
	SUMMARY	17	101	1	102	11	6	0	35	36	19	29	20	98	1	97	20	2	16	91	12	36	59	24	70	32	17	37	63	19
		119			119				119				119			119			119			119			119			119		

Additional filters can then be added to make the search more specific and appropriate for the target organization. In this sense, the table shown in the annex represents an alternative to the reported taxonomies, at least for companies that intend to pre-select a set of SDEE techniques potentially suitable according to the characteristics of the company.

5. Study limitations

Although this study has followed all steps recommended by Kitchenham and Charters [KIT07] to perform systematic literature reviews in the software engineering domain, we have identified some threats to the validity that are discussed next.

Construction validity. In a systematic literature review (SLR) the threats to the construction validity are typically related to the definition of the search strings, which allow identifying the primary studies that are required to answer the stated RQs. This is also conditioned by the source of literature (digital libraries) used as input, the primary studies selection process and the quality criteria used to include or exclude each pre-selected article. Although the process of gathering and selection of studies was followed accurately, there were some few studies that were identified only through a manually search, since the keywords used in the search strings were not included in the title or in the abstract of these articles. In order to mitigate this threat, we manually reviewed the references in the (automatically and manually) selected articles to determine other relevant articles which were not retrieved in the previous process. Such a process does not added new studies to the sample.

Internal validity. This threat is related to the retrieval and analysis of the primary studies. Although all steps of the process were completed successfully, they were conducted on a sample of studies that could have some bias not identified in this work.

External validity. This threat is related to the generalization of the findings. In this sense, we recognize some biases in the studies that were produced by the motivation of the authors for showing particular aspects of their studies. In order to mitigate this threat we reported only information supported by more than one study of the analyzed sample; contradictory information was not reported.

6. Conclusions and future work

This article presents a systematic literature review that aims to determine *what are the main taxonomies of SDEE methods reported in the literature?* (RQ1), and also *what are the most relevant classification criteria used in these taxonomies?* (RQ2). The first question intends to identify the taxonomies available to help software practitioners in the selection of SDEE techniques based on the clusters proposed by the main taxonomies. The second question intends to determine the most transversal clusters of techniques, according to the reported studies.

The SLR adhered to the process proposed by Kitchenham and Charters [KIT07], and it involved four digital libraries relevant for the study domain (IEEE, ACM, Science Direct and Springer). The selection process of primary studies concluded with 17 articles that were read in detail in order to answer the stated RQs.

Concerning the RQ1 this work determined that there is not a widely accepted taxonomy, since there are too many relevant and almost independent criteria to classify the SDEE techniques. Moreover, the hierarchical structure used to represent the taxonomies is not much useful to help identify clusters of techniques potentially useful for a software organization. This is probably true also to organize the knowledge in this study domain. Trying to help practitioners to identify suitable techniques for their organizations or projects, this article presents a matrix, produced as a result of classifying the 119 SDEE techniques reported in the study sample, and illustrates with an example how to use it. This matrix is an alternative to the proposed taxonomies, at least for software practitioners.

Concerning the RQ2, the study results indicate that 82% of the revised articles considered the participation of an expert, and also the use parametric/non-parametric models, as relevant classification criteria for the scientific community. However, it is not clear their relevance for the software industry. In this sense, and as part of the future work, it is required to conduct a field study to determine the relevance of the 16 classification criteria identified in this study.

This work also identified a surprising situation related to the study of expert-based SDEE techniques. Although the literature reports that this type of technique is the most currently used in the software industry [MOS00, TRE08, JOR14], only 20 out of 119 estimation techniques consider the participation of an expert. It is important to try understand the reasons behind this situation in order to align the future research efforts and the industry needs. In this sense, this represents an opportunity for the software engineering community to advance the state of the art in this study area.

References

[ALI16] Ali, I., Hosni, M., Abran, A. 2016. Systematic literature review of ensemble effort estimation. J. Syst. Softw. 118, C, 151-175. DOI: http://dx.doi.org/10.1016/j.jss.2016.05.016.

[ARA10] Aranda, J. Playing to the Strengths of Small Organizations. 2010. Proc. of the 1st Workshop on Requirements Engineering in Small Companies (RESC'10).

[BER92] Bergeron, F., St-Arnaud, J-Y. 1992. Estimation of information systems development efforts: a pilot study. Inf. Management 22, 4, 239-254. DOI=http://dx.doi.org/10.1016/0378-7206(92)90026-C.

[BOE00-2] Boehm W., Clark, Horowitz, Brown, Reifer, Chulani, Madachy, R., Steece, B. 2000. Software Cost Estimation with Cocomo II. Prentice Hall PTR, Upper Saddle River, NJ, USA.

[BOE00] Boehm, W., Abts, C., Chulani. 2000. Software development cost estimation approaches: A survey, S. Annals of Software Engineering 10: 177. https://doi.org/10.1023/A:1018991717352.

[BOE17] Boehm W. 2017. Software cost estimation meets software diversity. In Proceedings of the 39th International Conference on Software Engineering Companion (ICSE-C'17). IEEE Press, Piscataway, NJ, USA, 495-496. DOI: https://doi.org/10.1109/ICSE-C.2017.159.

[BOE81] Boehm, B. 1981. Software Engineering Economics, Prentice Hall, 1981

[BOE95] Boehm, B., Clark, B., Horowitz, E., Westland, C., Madachy, R., Selby, R. 1995. Cost Models for Future Software Life-cycle Processes: COCOMO 2.0. Special Volume on Software Process and Product Measurement, J.D. Arthur and S.M. Henry (Eds.), J.C. Baltzer AG, Science Publishers, Amsterdam, The Netherlands, Vol 1, pp. 45 - 60.

[BOE99] Boehm B. and Sullivan K. 1999. Software Economics: Status and Prospects. Information and Software Technology 41, pp. 937-946.

[BRE96] Breiman, L. 1996. Bagging predictors. Mach. Learn. 24, 2 (August 1996), 123-140. DOI=http://dx.doi.org/10.1023/A:1018054314350.

[BRI02] Briand, L.C., Wieczorek, I. 2002. Software Resource Estimation. In: Marciniak J.J. (ed.), Encyclopedia of Software Engineering, 2:1160-1196, John Wiley & Sons.

[BRI14] Britto R., Freitas V., Mendes E., and Usman M. 2014. Effort Estimation in Global Software Development: A Systematic Literature Review. In Proceedings of the 2014 IEEE 9th International Conference on Global Software Engineering (ICGSE'14). IEEE Computer Society, Washington, DC, USA, 135-144. DOI=http://dx.doi.org/10.1109/ICGSE.2014.11.

[DEJ12] Dejaeger K, Verbeke W, Martens D., and Baesens Bart. 2012. Data Mining Techniques for Software Effort Estimation: A Comparative Study. IEEE Trans. Softw. Eng. 38, 2, 375-397. DOI=http://dx.doi.org/10.1109/TSE.2011.55.

[EAR01] Earl M. 2001. Knowledge Management Strategies: Toward a Taxonomy. J. Manage. Inf. Syst. 18, 1 (May 2001), 215-233.

[HAL09] Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I., H. 2009. The WEKA data mining software: an update. SIGKDD Explor. Newsl. 11, 1, 10-18. DOI=http://dx.doi.org/10.1145/1656274.1656278.

[HEL65] Helmer, O. 1965. Social technology (No. P-3063). Rand Corp. Santa Monica CA. http://www.dtic.mil/docs/citations/AD0460520.

[HER77] Herd J., R., Postak J., N., Russell W., E., Steward K., R. 1977. Software cost estimation study: Study results, Final Technical Report, RADC-TR77-220, vol. I, Doty Associates, Inc., Rockville, MD, pp. 1-10.

[IDR15] Idri A., Amazal F., Abran A. 2015. Analogy-based software development effort estimation: a systematic mapping and review. Inform Softw Technology; Vol. 58: pp.206-230.

[JEN83] Jensen R. 1983. An Improved Macrolevel Software Development Resource Estimation Model. In Judge, G., W. Griffiths, and C. Hill, Learning and Practicing Econometrics, Wiley, New York.

[JOH06] Jim J. 2006. My Life Is Failure: 100 Things You Should Know to Be a Better Project Leader. The Standish Group. Available at: books.google.cl/books?id=M62bCwAAQBAJ&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false. Last visit: Dec. 2017.

[JON97] Jones, C. 1997. Applied Software Measurement. McGraw Hill.

[JOR04] Jørgensen M. 2004. A review of studies on expert estimation of software development effort. J. Syst. Softw. 70, 1-2, 37-60. DOI=http://dx.doi.org/10.1016/S0164-1212(02)00156-5.

[JOR07] Jørgensen M., Shepperd M. 2007. A Systematic Review of Software Development Cost Estimation Studies. IEEE Trans. Softw. Eng. 33, 1, 33-53. DOI=http://dx.doi.org/10.1109/TSE.2007.3.

[JOR14] Jørgensen M. 2014. What We Do and Don't Know about Software Development Effort Estimation. IEEE Software, vol. 31, no. 2, pp. 37-40.

[OTE05] Otero, M. Software in Chile (In Spanish: “Software en Chile”). White paper. IFEMA-COCIM. 2005.

[KAU93] Kauffman, R., R. Kumar, 1993. Modeling Estimation Expertise in Object Based ICASE Environments. Stern School of Business Report, New York University.

[KIT07] Kitchenham B., Charters S. 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering, EBSE 2007-001. Keele University and Durham University Joint Report.

[KLA08] Kläs M., Trendowicz A., Wickenkamp A., Münch J., Kikuchi N., Ishigai Y. 2008. The Use of Simulation Techniques for Hybrid Software Cost Estimation and Risk Analysis. Advances in Computers 74, 115-174.

[LAP08] Laporte, C., Alexandre, S. Y., Renault, A. 2008. Developing International Standards for VSEs. IEEE Computer 41(3), 98–101.

[LAW91] Lawrence P, H.; Myers W. 1991. Measures for Excellence: Reliable Software on Time, Within Budget. Prentice Hall. p. 234. ISBN 978-0-13-567694-3.

[MOL03] Molkken K., Jørgensen M. 2003. A Review of Surveys on Software Effort Estimation. In Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE'03). IEEE Computer Society, Washington, DC, USA, 223-.

[MOS00] Moses, J. 2000. Learning How to Improve Effort Estimation in Small Software Development Companies. In 24th International Computer Software and Applications Conference (COMPSAC'00). IEEE Computer Society, Washington, DC, USA, 522-527.

[MYR05] Myrtveit, Ingunn, Stensrud, Erik, Martin S. 2005. Reliability and Validity in Comparative Studies of Software Prediction Models. IEEE Trans. Softw. Eng. 31, 5 (May 2005), 380-391. DOI=http://dx.doi.org/10.1109/TSE.2005.58.

[NIK10] Mittas N., Angelis L. 2010. LSEbA: least squares regression and estimation by analogy in a semi-parametric model for software cost estimation, Empirical Software Engineering, https://doi.org/10.1007/s10664-010-9128-6.

[NIK15] Nikolaos M., Efi P., Lefteris A., Andreas S. Andreou. 2015. Integrating non-parametric models with linear components for producing software cost estimations. J. Syst. Softw. 99, C, 120-134. DOI: http://dx.doi.org/10.1016/j.jss.2014.09.025.

[PAR88] Park R. 1988. The Central Equations of the PRICE Software Cost Model. Park R., 4th COCOMO Users', Group Meeting.

[POU96] Poulin J. 1996, Measuring Software Reuse, Addison-Wesley.

[Pyt13] Pytel P., Britos P., García-Martínez R. 2013. A Proposal of Effort Estimation Method for Information Mining Projects Oriented to SMEs. Lecture Notes in Business Information Processing, vol 139. Springer, Berlin, Heidelberg.

[RAJ16] Rajper S., Shaikh Z. A., 2016, Software Development Cost Estimation: A Survey , Indian Journal of Science and Technology, Vol 9(31), DOI: http://dx.doi.org/10.17485/ijst%2F2016%2Fv9i31%2F93058.

[REI17] Reinhartz-Berger I., Figl K., Haugen Ø. 2017. Investigating styles in variability modeling: Hierarchical vs. constrained styles, In Information and Software Technology, Vol. 87, pp. 81-102, https://doi.org/10.1016/j.infsof.2017.01.012.

[RUB83] Rubin, H. 1983. Estimacs. IEEE.

[TRE08] Trendowicz A., Münch J., Jeffery R. 2008. State of the Practice in Software Effort Estimation: A Survey and Literature Review. In: Huzar Z., Koci R., Meyer B., Walter B., Zendulka J. (eds) Software Engineering Techniques. CEE-SET 2008. Lecture Notes in Computer Science, vol 4980. Springer, Berlin, Heidelberg.

[TRE14] Trendowicz A., Jeffery R. 2014. Classification of Effort Estimation Methods. In: Software Project Effort Estimation. Springer, Cham. https://doi.org/10.1007/978-3-319-03629-8_6.

[USM14] Usman M., Mendes E., Weidt F., Britto R. 2014. Effort estimation in agile software development: a systematic literature review. In Proceedings of the 10th International Conference on Predictive Models in Software Engineering (PROMISE'14). ACM, New York, NY, USA, 82-91. DOI=http://dx.doi.org/10.1145/2639490.2639503.

[VAC14] Vachik, S., D., Kamlesh D. 2014. Neural network based models for software effort estimation: a review. Artif. Intell. Rev. 42, 2 (August 2014), 295-307. DOI: http://dx.doi.org/10.1007/s10462-012-9339-x.

[VAR03] Varela, M. Diagnose of the IT industry in Chile, (In Spanish: "Diagnóstico de la Industria de TI en Chile"). White Paper, CHILE INNOVA Program. Chilean government, 2003.

[WAN06] Wangenheim, C. G., Anacleto, A., Salviano C. F. 2006. Helping Small Companies Assess Software Processes. IEEE Software, vol. 23, no 1, pp. 91–98.

[WEN12] Wen J., Li S., Lin Z., Hu Y., Huang C. 2012. Systematic literature review of machine learning based software development effort estimation models. Inf. Softw. Technol. 54, 1, 41-59. DOI=http://dx.doi.org/10.1016/j.infsof.2011.09.002.

[WOL74] Wolverton R. W. 1974. The Cost of Developing Large-Scale Software. IEEE Trans. Comput. 23, 6 (June 1974), 615-636. DOI=http://dx.doi.org/10.1109/T-C.1974.224002.

[YAO99] Yao X., Liu Y. 1999. Ensemble learning via negative correlation, Neural Networks, Volume 12, Issue 10, pp. 1399-1404, http://dx.doi.org/10.1016/S0893-6080(99)00073-8.

Annex

Table 8 presents the details of the SDEE techniques classified according to the criteria defined in Section 3.3.1. This table is the result of a classification of the techniques performed using a feature model, which allows us to determine the features of an estimation method, considering simultaneously several classification criteria. These features were obtained based on the characteristics of the methods reported in the papers used in this study, using the original report of each method only when necessary.

Table 8. SDEE techniques SDEE techniques classified according to the criteria defined.

Classificassion Dimensions	Licence			Data				Parametric				Expert			Model			Analogy			Regression			Composite			Artificial Intelligence		
Techniques / Values	Proprietary	Non-Proprietary	Unknown	Many data	Sparse Data	No Data	Unknown	Parametric	Non-Parametric	Semi-Parametric	Unknown	Need Expert	No Need Expert	Unknown	Model-Based	No Model-Based	Unknown	Analogy-Based	No Analogy-Based	Unknown	Regression-Based	No Regression-Based	Unknown	Composite Technique	Single Technique	Unknown	AI-Based	No AI-Based	Unknown
Delphi		x				x			x			x				x			x			x			x			x		
Work Breakdown Structure (WBS)		x				x			x			x				x			x			x			x			x		
Rule Based			x		x						x	x					x			x			x			x				x
Guesstimation		x				x			x			x				x			x			x			x			x		
Estimeeting		x			x				x			x				x		x				x			x			x		
Planning Game		x				x			x			x				x			x			x			x			x		
Expert Judgment		x				x			x			x				x			x			x			x			x		
Analytic Hierarchy Process (AHP)		x			x				x			x				x			x			x			x			x		
Planning Poker		x				x			x			x				x			x			x			x			x		
Constructive Agile Estimation Algorithm		x			x			x				x			x				x		x			x				x		
Checkpoint			x	x				x					x		x				x			x			x				x	
Estimacs	x			x				x					x		x				x			x			x				x	
PRICE-S	x			x				x					x		x				x			x			x				x	
SEER-SERM	x			x				x					x		x				x			x			x				x	
SLIM		x		x				x					x		x				x			x			x				x	
Softcost	x			x				x					x		x				x			x			x				x	
COCOMO		x		x				x					x		x				x			x			x				x	
COCOMO II		x		x				x					x		x				x			x			x				x	
COSTAR	x			x				x					x		x				x			x			x				x	
Cost Xpert	x			x				x					x		x				x			x		x					x	
Estimate Professional	x			x				x					x		x				x			x		x					x	
SELECT Estimator	x			x				x					x		x				x			x		x					x	
StepWise ANOVA	x			x				x					x		x			x			x			x					x	
Fuzzy COCOMO		x		x				x					x		x				x		x	x		x			x			
SEEM-SER		x		x				x					x		x	x			x			x		x					x	
Generic Artificial Neural Networks (GANN)		x		x				x					x		x				x			x			x		x			
Cerebellar Model Arithmetic Computer (CMAC)		x		x				x					x		x				x			x			x		x			
Predator-Pray		x		x				x					x		x				x			x			x		x			
COCOMO/COSTAR	x			x				x					x		x				x			x			x				x	
Knowledge Plan		x		x				x					x		x					x			x			x				x
True-S	x			x				x					x		x					x			x			x				x
Bayesian analysis		x		x					x				x		x					x		x		x					x	
Case Studies		x		x					x				x		x					x			x		x	x				
Artificial Neural Networks (ANN)		x		x					x				x		x					x			x		x					x
Decision Trees		x		x					x				x		x					x			x		x					x
Bayesian Networks		x		x					x				x		x					x			x			x				
Support Vector Regression (SVR)		x		x					x				x		x					x	x				x					x
Genetic Algorithms		x		x					x				x		x					x			x	x						x
Genetic Programming		x		x					x				x		x					x			x	x						x
Association Rules		x		x					x				x		x			x					x	x						x
Madachy's dynamic (Abdel-Hamid and Madnick)		x		x					x				x		x			x					x		x					x
COSMIC	x			x					x				x		x				x				x		x					
COSMIC-FFP	x			x					x				x		x				x				x	x						
SPQR/100	x			x				x					x		x				x			x			x					
SoftCost-R	x			x				x					x		x				x			x			x					
Classification and Regression Trees (CART)		x		x					x				x		x				x				x		x		x			
Optimized Set Reduction (OSR)		x		x					x				x		x				x				x		x		x			
Stepwise ANOVA – Stepwise Analysis of Variance		x		x					x				x		x				x			x		x					x	
Ordinary Least Squares Regression (OLS)		x		x				x					x		x				x		x				x				x	
COBRA – Cost Estimation Benchmarking and Risk Analysis	x				x			x				x			x				x					x				x		
Estor	x			x				x					x		x			x				x			x				x	
COCOMO-U		x		x				x					x		x				x			x			x				x	
Neuro-fuzzy System		x		x						x		x			x				x			x			x				x	
ANGEL		x		x							x		x			x		x					x			x				x
Agile COCOMO II		x		x				x					x					x				x			x				x	
Incremental Development Productivity Decline (IDPD)		x		x							x			x			x			x			x			x				x
Fuzzy Decision Tree		x		x						x		x			x				x			x			x				x	
Rule Induction		x		x						x		x			x				x			x			x				x	

Classificassion Dimensions	Licence			Data				Parametric				Expert			Model			Analogy			Regression			Composite			Artificial Intelligence		
Techniques / Values	Proprietary	Non-Proprietary	Unknown	Many data	Sparse Data	No Data	Unknown	Parametric	Non-Parametric	Semi-Parametric	Unknown	Need Expert	No Need Expert	Unknown	Model-Based	No Model-Based	Unknown	Analogy-Based	No Analogy-Based	Unknown	Regression-Based	No Regression-Based	Unknown	Composite Technique	Single Technique	Unknown	AI-Based	No AI-Based	Unknown
Fuzzy Rules		x		x						x		x			x				x			x			x			x	
Evolutionary Algorithms (EA) Regression		x		x					x				x		x				x		x				x			x	
Generic Program		x		x						x			x		x				x			x			x	x			
Hierarchical Decision Rules (HIDER)		x		x						x			x		x				x		x				x	x			
Estimation by analogy		x		x					x				x		x			x					x			x				x
Locally weighted regression		x		x					x				x		x				x		x			x			x			
Evolutionary Algorithms (EA) & Multiple Regression Splines (MARS)		x		x						x			x		x				x		x			x			x			
Classification and Regression Trees (CART) + Ordinary Least Squares Regression (OLS)		x		x						x			x		x				x			x		x			x			
Automatic Case Elicitation (ACE)		x		x					x				x			x			x				x		x					x
GRACE		x		x					x				x			x			x				x			x				x
BRACE		x		x					x				x			x			x				x			x				x
AQUA		x		x					x				x			x			x				x			x				x
Fuzzy Case Based Reasoning (CBR)		x		x					x				x			x			x				x		x			x		
Colaborative Filtering		x			x					x		x				x		x					x			x				x
Classification and Regression Trees (CART) + Case Based Reasoning (CBR)		x		x						x			x		x				x			x			x			x		
Optimized Set Reduction (OSR) + Ordinary Least Squares Regression (OLS)		x		x					x				x		x				x			x			x				x	
Neural Networks + Cluster Analisys		x		x					x				x		x				x			x			x			x		
AVN		x		x						x			x		x			x			x			x						x
OLS regression with log transformation		x		x					x				x		x				x		x				x			x		
OLS regression with Box Cox (BC) transformation		x		x					x				x		x				x			x			x			x		
Ridge regression		x		x				x					x		x				x			x			x			x		
Least median squares regression		x		x				x					x		x				x		x				x			x		
MARS		x		x					x				x		x				x		x				x			x		
Model tree		x		x				x					x			x			x			x			x		x			
Multilayered perceptron neural network		x		x				x					x			x			x			x			x		x			
Radial basis function networks		x		x				x					x			x			x			x			x		x			
Least squares support vector machines		x		x				x					x			x			x		x				x			x		
Use Case Points (UCP)		x			x					x		x			x			x				x			x			x		
UCP Method Modification		x			x					x		x			x			x				x			x			x		
Bagging + M5P/Regression trees (RT)		x		x							x		x		x				x		x				x		x			
Bagging + M5P/Model trees (MT)		x			x						x		x		x				x			x			x		x			
Bagging + Multilayer perceptron (MLP)		x		x							x		x		x				x		x				x		x			
Bootstrapping + MLP		x		x							x		x		x				x		x				x		x			
Bagging + Linear regression (LR)		x		x							x		x		x				x		x				x			x		
Bagging + Support vector regression (SVR)		x		x							x		x		x				x		x				x			x		
Bagging + Radial basic function (RBF)		x		x							x		x		x				x		x				x			x		
Bagging + RT		x		x							x		x		x				x		x				x			x		
Negative Correlation learning (NCL) + MLP		x		x							x		x		x				x		x				x			x		
Random + MLP		x			x						x		x		x				x			x			x		x			
Bagging + Adaptive neuro fuzzy inference system (ANFIS)		x			x						x		x		x				x		x				x			x		
Case based-reasoning (CBR) + Estimation by analogy (EBA)		x		x							x		x		x			x				x			x		x			
Multiple additive regression trees (MART)		x		x							x		x		x				x		x				x		x			
Multilayer perceptron (MLP)		x		x							x		x		x				x			x			x		x			
Random forests		x		x							x		x		x				x			x			x		x			
Stochastic gradient boosting technique (SGB)		x		x							x		x		x				x		x				x		x			
Iterated bagging + CBR		x		x							x		x		x				x		x				x			x		
Gaussian process(GaP) + MLP + RBF + SVR + k-nearest neighbors (K-NN) + Locally weighted Learning (LWL) + Bagging (fast decision tree) + Additive regression with decision stump + Random subspace + Decision stump + M5P + Conjunctive rule (CR) + Decision Table		x		x							x		x		x				x		x				x		x			
MLP + SVR + K-NN + RT + RBF		x		x							x		x		x				x		x				x		x			
		x		x							x		x		x				x		x				x		x			
		x		x							x		x		x				x		x				x		x			
		x		x							x		x		x				x		x				x		x			
L.R + ANN		x		x							x		x		x				x		x				x		x			
MLP + ANFIS + SVR		x		x							x		x		x				x		x				x		x			
Multi linear regression (MLR) + Back-propagation neural networks (BPNN) + RBF + Dynamic evolving neural-fuzzy inference system (DENFIS) + Threshold-acceptance-based neural network (TANN) + SVR		x		x							x		x		x				x		x				x		x			
Group method of data handling (GMDH) + Genetic programming (GP) + Counter propagation neural network (CPNN)		x		x							x		x		x				x		x				x		x			
COCOMO + Function points (FP) + SLIM		x		x						x			x		x				x			x			x			x		
Function points + SLIM		x		x						x			x		x				x			x			x			x		
Cost or Schedule as Independent Variable (CAIV, SAIV)		x		x						x			x		x				x		x				x			x		
Equivalent Size		x		x						x			x		x			x				x			x			x		
Adjusted for %Design,Code,Test Modified, Understandability		x		x						x			x		x			x				x			x			x		
% Development for Reuse; % Development with Reuse		x		x						x			x		x			x				x			x			x		
SUMMARY	17	101	1	102	11	6	0	35	36	19	29	20	98	1	97	20	2	16	91	12	36	59	24	70	32	17	37	63	19