Information and Software Technology 53 (2011) 317–343

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Measuring and predicting software productivity: A systematic map and review

Kai Petersen [*]

School of Computing, Blekinge Institute of Technology, Box 520, SE-372 25, Sweden
Ericsson AB, Box 518, SE-371 23, Sweden

ARTICLE INFO

Article history:
Received 20 June 2010
Received in revised form 1 December 2010
Accepted 3 December 2010
Available online 16 December 2010

Keywords:
Software productivity
Software development
Efficiency
Performance
Measurement
Prediction

ABSTRACT

Context: Software productivity measurement is essential in order to control and improve the performance of software development. For example, by identifying role models (e.g. projects, individuals, tasks) when comparing productivity data. The prediction is of relevance to determine whether corrective actions are needed, and to discover which alternative improvement action would yield the best results.
Objective: In this study we identify studies for software productivity prediction and measurement. Based on the identified studies we first create a classification scheme and map the studies into the scheme (systematic map). Thereafter, a detailed analysis and synthesis of the studies is conducted.
Method: As a research method for systematically identifying and aggregating the evidence of productivity measurement and prediction approaches systematic mapping and systematic review have been used.
Results: In total 38 studies have been identified, resulting in a classification scheme for empirical research on software productivity. The mapping allowed to identify the rigor of the evidence with respect to the different productivity approaches. In the detailed analysis the results were tabulated and synthesized to provide recommendations to practitioners.
Conclusion: Risks with simple ratio-based measurement approaches were shown. In response to the problems data envelopment analysis seems to be a strong approach to capture multivariate productivity measures, and allows to identify reference projects to which inefficient projects should be compared. Regarding simulation no general prediction model can be identified. Simulation and statistical process control are promising methods for software productivity prediction. Overall, further evidence is needed to make stronger claims and recommendations. In particular, the discussion of validity threats should become standard, and models need to be compared with each other.

Contents

* Address: School of Computing, Blekinge Institute of Technology, Box 520, SE-372 25, Sweden.
E-mail addresses: kai.petersen@bth.se, kai.petersen@ericsson.com
URLs: http://www.bth.se/besq, http://www.ericsson.com

0950-5849/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2010.12.001

1. Introduction

The focus of software process improvement often is to improve the productivity (or efficiency or performance) of software development. Productivity is commonly defined as the ratio of output divided by input. In software development a variety of outputs were defined in the productivity literature, such as quality and quantity in terms of functions, lines of code, implemented changes, and so forth. In most cases the input is the effort needed for creating the output.

A famous quote says that *"you cannot predict nor control what you cannot measure"* [1]. Hence, software productivity measurement plays an important role in software process improvement activities. It allows to define a baseline for improvement and can be re-evaluated once the improvements have been implemented. Furthermore, productivity measurements are important as benchmarks to determine whether there is a need for improvement to be competitive. Within a company benchmarking is of relevance to identify the best performers and then learn from them.

Productivity prediction is concerned with quantifying how the productivity will be in the future. Predictions of productivity are useful to determine whether corrective actions are needed, e.g. when past data indicates a decline of productivity and the prediction model shows that this trend is likely to continue. Prediction should also assist managers in evaluating improvement alternatives, e.g. how does productivity change with improved testing efficiency.

The quantification of past and future productivity is the subject of this systematic review. The choice to use systematic review to aggregate what we know about software productivity quantification is motivated by the high rigor required by systematic reviews. In all empirical studies it is important to be systematic when analyzing evidence to increase the validity of the study results. To achieve the same rigor in a literature review systematic reviews in software engineering follow a defined process (cf. [2,3] for guidelines of conducting systematic reviews in the context of software engineering). The steps are (1) definition of a search strategy; (2) inclusion and exclusion of articles based on well defined inclusion and exclusion criteria; (3) evaluation of the quality of the articles; (4) extraction of data; and (5) synthesis of evidence. Systematic reviews have the following advantages: (1) they reduce bias due to a well defined process; (2) guidelines of how to aggregate evidence are available; (3) the rigor of the review makes the results more defendable; and (4) the well defined and documented process allows replication. A systematic map structures the area and is the first step towards a systematic review (cf. [4]).

To the best of our knowledge no recent systematic review on software productivity measurement and prediction has been conducted. Hence, there is a need to synthesize the evidence regarding the usefulness and accuracy of existing approaches. The systematic review makes the following contributions:

- Provide a classification scheme of productivity research on measurement and prediction. The classification scheme creates a new view on productivity research by providing a structure of the area. Furthermore, relevant work of future productivity studies can be easily identified.
- Identify the past focus and the change of research trends over time.
- Analyze evidence with regard to usefulness and accuracy to aid practitioners in making informed choices when selecting an approach based on evidence.

Table 1
Question for the mapping study.

ID	Mapping question	Rational
MQ1	Which publication forums are the main targets for productivity research?	The answer to this question is a pointer to where productivity research can be found as well as which are good targets for publication of future studies
MQ2	How has the frequency of approaches related to predicting and measuring productivity changed over time?	The answer to this question shows research trends over time, such as emerging or abandoned approaches to productivity measurement and prediction
MQ3	How is the frequency of published research distributed within the structure of productivity research?	The distribution indicates research focus on specific approaches to productivity measurement and prediction in combination with research methods. The knowledge of papers within categories allows to (1) identify obvious research gaps, and (2) to pinpoint references for related work articles

Table 2
Question for the review study.

ID	Review questions	Rational
RQ1	What evidence is there for the accuracy/usefulness of the prediction and measurement methods?	Evidence for accuracy and usefulness aids practitioners in making informed decisions in choosing a productivity measurement and prediction solution
RQ2	What recommendations can be given to (1) methodologically improve productivity studies, and (2) improve the packaging and presentation of productivity studies?	Identifying improvement opportunities allows to increase the rigor of studies. Improvements in the reporting of the studies aids in future aggregation of evidence

The remainder of the paper is structured as follows: Section 2 presents the related work. The research design of the systematic map and review is described in Section 3. Thereafter, the classification scheme is presented in Section 4. The results of the map are presented in Section 5 and for the review in Section 6. Section 7 discusses the results. Section 8 concludes the paper.

2. Related work

Dale and van der Zee [7] look at productivity measures from different perspectives, i.e. from a development perspective, a user perspective, and a management perspective. The development perspective (including requirements, implementation, and verification and validation) is reflected in lines of code productivity. With regard to lines of code productivity [7] points out that lines of code is only a part of the valuable outputs produced in software development. Furthermore, when comparing between projects it is important to be clear about the definition of the measure. The same applies to the effort related to the produced lines of code. When defining effort the decision has to be made of which staff to include in the analysis (e.g. maintenance staff, developers, testers, etc.). From a user's perspective function points have been introduced, function points being a representation of value delivered to the user. With regard to function points the article points out that costs of the user (e.g. for support and interaction with the development team) have to be taken into consideration as well. The article also discusses productivity factors and points out the risk of measuring too many variables related to productivity. From a managerial perspective the analysis of productivity mainly focuses on monetary aspects, i.e. what value is created for the money made in IT related investments.

Scacchi [8] reviewed a vast amount of software productivity literature. With regard to the identification of productivity factors doubt is raised to flawed analyses in the primary studies. Some examples for problems in the validity of studies were:

- Poor definition of measures (e.g. measure of lines of code verses effort related to all activities of the development life-cycle).
- Unclear root-cause effect for changes in productivity measures for function points (e.g. unknown whether change in productivity was due to re-calibration of subjective productivity factors or due to productivity improvements).
- Ignorance of a combined effect of productivity factors (e.g. managerial and technical aspects).

Furthermore, Scacchi points out that none of the studies he found took large variances in programmer productivity into account. However, when studying large projects the average programmers dominates the productivity while in small projects with few developers the individual differences dominate. Based on these observations a number of recommendations were made:

- In software development many outputs are produced, hence a multivariate analysis is important.
- Data collection should be done throughout the development life cycle. For example, when only collecting data of lines of code productivity problems in early phases (e.g. requirements engineering) will negatively effect lines of code productivity, which is troublesome to discover. The actual coding activity also only accounts for a small part of the overall software development life-cycle. Furthermore, dynamics in the process (such as iterations) play a role in productivity.

Hence, the ability to work with life-cycles and not treating software development as a black box is important.

In more recent years improvements suggested in the review by Scacchi have been addressed. For example, data envelopment analysis has been evaluated to address the fact that software development is multivariate. In addition, simulation has become an important part of software productivity prediction, allowing to consider the software life-cycle. The reviews of software productivity measurement (both being from the early 1990s) show that there is a need for a fresh aggregation of productivity literature.

3. Research design of map and review

3.1. Map and review questions

Mapping and review studies have different aims. The mapping study aims at structuring an area and showing how the work is distributed within the structure. The focus of mapping studies is not to provide recommendations based on the strength of evidence as is the main purpose of systematic reviews. Hence, the mapping related questions are concerned with the structure of the area, and the review related questions with the evidence [4].

Table 1 shows the questions related to the structuring of the research area. The questions as well as the rationals motivating the importance of the question are stated. The questions can be answered through the classification of the research articles identified.

Table 2 states the questions and rational for the systematic review part.

Based on the mapping and review questions the search strategy and paper selection criteria were defined.

3.2. Search strategy and paper selection criteria

Dybå et al. [3] propose four phases to define a comprehensive search strategy. In the first phase a search string is defined to search relevant databases and proceedings for articles answering the research questions. In the second and third phase, inclusion and exclusion criteria are defined which are used to evaluate articles based on title (phase 2) and abstract (phase 3). Possible reasons to exclude a paper are (1) that it is out of scope as it does not belong to the area of software engineering in general or software productivity in particular or (2) that the abstract does not indicate that proper empirical evaluation was used. Finally, the relevant articles are retrieved for an in-depth evaluation.

Phase 1 – Search relevant databases: The definition of the search string is based on population, intervention, comparison and outcome.

- *Population* The population is the development of software. In order to search for the population we use the keywords "software development", "software engineering" and "software process". Different restrictions have been evaluated in order to assure to get papers from the software domain. When only using "software" a high number of papers were returned that were clearly out of scope (e.g. from manufacturing and system performance). At the same time, it was assured that no papers from the reference set got lost with that restriction to population, the reference set being a set of 12 papers we knew should be found by the search.
- *Intervention*: The intervention is the measurement and prediction of software productivity. In order to identify the keywords for the intervention, we looked up synonyms for productivity using the OXID Free Online English Thesaurus and Dictionary website. The identified synonyms for productivity considered are efficiency and performance. As performance delivered too many articles out of scope, we searched for process performance, development performance, project performance and programmer performance instead. Furthermore, the terms measurement, metric, prediction and estimation need to be considered.
- *Comparison*: The focus of the study was not limited to comparative studies. Hence, comparison was not considered in the search strategy.
- *Outcome*: Here, we seek for measurement and estimation methods that are promising with respect to accuracy when measuring or predicting productivity. Accuracy and usefulness of the measurement and prediction approaches can be determined through empirical evidence. Hence, the search string contained words like empirical, validation, evaluation, etc.

The search string used was formulated as follows:

1. *Search in abstracts:* ("software process" OR "software development" OR "software engineering") AND (productivity OR efficiency OR "process performance" OR "development performance" OR "project performance") AND (measure OR metric OR model OR predict OR estimate) AND (empirical OR validation OR evaluation OR experiment OR "case study" OR example OR "action research" OR simulation).
2. *Search in titles:* "software productivity" AND (measure OR metric OR model OR predict OR estimate) AND (empirical OR validation OR evaluation OR experiment OR "case study" OR example OR action research OR simulation).

For the search in titles and keywords the search terms for evaluation were removed from the search string to avoid exclusion of relevant articles. The sets of articles from abstract, title, and keywords were merged and duplicates removed.

As some databases are not able to handle complex search strings, the string can be reformulated using boolean algebra, so that separate terms connected with AND can be entered into the database one after another.

For the selection of databases, we followed the recommendations in [3] as well as Kitchenham and Charters [2] who consider the following databases relevant: Compendex and Inspec, IEEE Explore, ACM Digital Library, ISI Web of Science, Kluwer Online, scienceDirect (Elsevier), SpringerLink, and Wiley Inter Science Journal Finder.

From experiences of previous systematic reviews reported in [3], we know that only Compendex and Inspec, ISIWeb of Science, IEEE Explore and ACM Digital Library deliver unique results. Everything else (like ScienceDirect, SpringerLink, etc.) is indexed by ISI Web of Science, Compendex, and Inspec.

In order to assure that the search is unbiased and covers the software productivity literature two steps have been taken to assure the quality of the search. First, a reference set of articles was created including the articles we knew should be found by the search, and are relevant to be included in the review. The reference set consisted of the following references: [17,20,23,24,26,30,32,34,37,54,55,57]. Secondly, the reference lists of the included articles were reviewed for papers referring to measurement/prediction of software development productivity/performance based on titles. The reference list contained publications related to productivity (e.g. two books on programmer productivity measurement, namely *Measuring programmer productivity and software quality* by L.J. Arthur, and *Programming Productivity* by T. Caspers Jones). However, these books were not included as the scope of this review was peer-reviewed research studies published in journals and conference proceedings. No additional journal and conference papers were identified that were not returned by the search.

Phase 2 and 3 – Exclusion of articles based title and abstract: In this phase, we evaluated the titles and abstracts of the returned studies of phase one. For the *inclusion* of an article it must be *explicitly mentioned* in the abstract that:

- *Topic area:* The paper provides measurements or models (process model, simulation model, etc.) that are able to determine or predict at least one of the following attributes (have to be mentioned) of software development: (1) Performance, productivity or efficiency of software development or software processes or software projects or software developers; (2) Relations between different direct measurements (like size and effort, or reliability and effort, lead-time and resources, or multivariate relations) that are used to characterize the attributes above. That is, a single measure is not enough (e.g., lead time) without relating it to other measures (e.g. effort).
- *Evaluation:* The model or measurement was evaluated in form of proof of concept, experiment, case study, or other empirical research methods to show its applicability and usefulness/accuracy. It is also reasonable to consider papers that are based on real-world data from software repositories. Journal papers that are in scope based on the above definition should always be included for a more detailed evaluation as they are expected to report more mature and in depth results.
- *Type of literature:* The literature was restricted to scientific publications in peer-reviewed journals and conferences. The reason for the restriction was to assure that the included papers were thoroughly reviewed prior to publication, and focused on novel results or the empirical evaluation of results.

From the search results the following studies should be *excluded*:

- *Measurement not main focus:* Measurement and prediction should have the main focus. For example, if productivity is evaluated for different companies to provide an overview of productivity developments, but the measures are not the subject of investigation, then the study should be excluded.
- *Direct measures:* Direct measures are for example lines of code (LOC), they are not combined with other measures and thus no claims regarding productivity are possible.
- *Secondary studies:* The synthesis of evidence is based on the primary studies identified by the search. The secondary studies are excluded from the actual synthesis and results. The main results of the secondary studies can be found in Section 2 to allow for comparisons between this systematic review and previous reviews.
- *Measures for single techniques:* A single technique is for example a software inspection reading technique or a certain testing technique. We are not interested in these measures, as these measurements are on a too fine level of granularity. That is, they do not give an indication how well the software development organization performs, e.g., on the overall process level or within a development phase. However, that does not mean that we exclude measurements on the level of individuals (e.g., programmer productivity) as an individual (or a team) can represent a whole phase or life-cycle in software development. Furthermore, when reviewing a low level of granularity, the number of studies to be reported would not be manageable within a reasonable time frame. However, papers measuring the productivity of a sub-process (be it a requirements, testing, or inspection process) are not to be excluded.
- *No evaluation:* Studies that have not evaluated a solution (be it in an experiment, proof of concept, or a case study) shall be excluded.

The search contained articles published before 2009. The search was piloted, refined, and finalized in the first quarter of 2009. The extraction and analysis of the data was conducted throughout 2009 and early 2010. Thereafter, the manuscript was written and internally reviewed in the first half of 2010.

The inclusion and exclusion criteria are an input to the study selection procedure. For the inclusion and exclusion to be objective the criteria have to be tested to eventually adjust them in order to reduce bias.

3.3. Study selection procedure

The reading of abstracts and titles was done by two researchers. The reason for having two researchers is the reduction of bias when including or excluding articles. Both researchers go through titles and abstracts separately and categorize them as follows:

- *Include:* The researcher is sure that the paper is in scope and that it was properly validated using empirical methods.
- *Exclude:* The researcher is sure that the paper is out of scope or that the validation was insufficient.
- *Uncertain:* The researcher is not sure whether the paper fulfills either the inclusion or exclusion criteria above.

The inclusion and exclusion criteria have been evaluated by the author and a fellow colleague. Prior to the exclusion and inclusion trial the author explained and discussed the criteria with his colleague. Thereafter, both applied the criteria on 100 articles independently and thereafter the results were compared. The results showed high agreement between them with only seven disagreements and three articles where one researcher was uncertain. The high agreement indicates that the inclusion and exclusion criteria were objective. Hence, the primary author made the decision of whether articles should be included or excluded in the map and review for all remaining articles. If the author was uncertain about a specific article it is included for the detailed review to avoid exclusion of relevant work.

3.4. Data extraction

A data extraction form was created and filled in for each article by the author. The data extraction form was split into different sections to be filled in. The following information was captured in the form:

- The first section of the form was concerned with extracting data about the measurement or prediction method. It had to be specified whether the purpose was reactive (productivity measurement) or predictive (productivity prediction). Furthermore, the type of measurement/prediction needed to be named and the construction of the measure described. The researcher ticked the options that apply for the article under review.
- In the second section the type of research was specified (validation research, evaluation research, or solution proposal, see Section 4). As context is an important aspect for generalizability (cf. [9,12]) and comparison of studies, information about the context was narratively described.
- The third section was concerned with the rigor of the study. The quality of the articles was evaluated based on the questions listed in Appendix A. The criteria are divided into two groups, one regarding the model, and one regarding the research group. The model was checked based on whether the construction/structure was described, and whether the variables measured were named. With regard to the research method the research process was covered. Design and conducting the study is covered in the quality checks on data collection, control groups, as well as analysis and reporting of findings. In addition, context was considered as a quality criterion as this was raised as important when judging the results of studies (cf. [12,9]). Similar criteria for checking rigor have been described in [10].
- The fourth section described the outcome of the study regarding the measurements/predictions and their evaluation. In addition, specifically positive/ negative aspects could be described. This section was filled in as free text.

Based on the extracted data the analyses for the map and the review was conducted.

3.5. Analysis

For synthesizing qualitative as well as quantitative evidence Dixon-Woods et al. [13] provide a number of alternatives.

We used content analysis and narrative summary as approaches to integrate evidence, both being presented in Dixon-Woods et al. [13]. Content analysis categorizes data and analyzes frequencies of themes within categories. Hence, it allows to transfer qualitative into quantitative information. The tabulation simplifies the analysis of the evidence. Some categories relevant for this review were already identified in literature and are referred to in the explanation of the classification scheme in Section 4.

Narrative summaries recount findings and describe them. Evidence in diverse forms can be discussed side by side. We do not apply meta-analysis on the findings as this would require the form of evidence to be similar, e.g. when replicating experiments the same hypotheses are tested and hence the means and variances of the experiments are more likely to be comparable. As shown in the classification scheme in Section 4 a variety of evaluation approaches have been used. Consequently, a narrative summary is suitable to synthesize the evidence of this review.

3.6. Threats to validity

The discussion of validity threats is important to be able to judge the strengths and limitations of the study with regard to the validity of the outcome. Relevant threats to validity were researcher bias, exclusion of relevant articles, gray literature, and generalizability.

Single researcher bias in inclusion/exclusion: The main validity threat of this study is the threat of researcher bias. This is the main threat as a single researcher has been conducting the inclusion/exclusion, data extraction, and analysis of the studies. Some actions have been taken to reduce the threat, as mitigation was not possible due to that a single researcher was conducting the study. The actions taken were:

- The review protocol was peer-reviewed with a fellow researcher to assure good understandability and clarity for inclusion/exclusion and data extraction.
- The clarity of inclusion/exclusion criteria has been evaluated on a test-set of articles, showing a high degree of agreement between two researchers conducting the process of inclusion/exclusion independently.
- In the quality assessment minimum criteria have been defined to make the judgement as easy and objective as possible. The minimum criteria led to a yes/no answer in inclusion or exclusion of articles.
- After having done the extraction, the author reviewed the forms and articles once more to check whether something was forgotten, or whether something should be changed.

When conducting systematic reviews or mapping studies alone, the test-retest approach has been proposed and applied in [6]. In this study on software productivity, the test-retest was not conducted as it was not known prior to conducting the systematic review study. Hence, now the test-retest would be largely biased due to the learning effect when conducting the systematic review. However, for future reviews conducted by a single researcher it is an approach to check the objectivity of searches, inclusion/exclusion, and classification.

Exclusion of relevant articles: The risk of excluding relevant articles is reduced by being inclusive, i.e. the researcher included articles for detailed review when in doubt. In addition no articles with lower scores of rigor (see definition in Appendix A) have been excluded from the review. The reason for including the studies is that they still can serve for triangulation. In the synthesis of the evidence the strength of evidence is taken into consideration. Furthermore, there is no restriction for the time-span in which the search was conducted. With regard to the exclusion there is also a risk of researcher bias, given that the majority of articles were reviewed by a single researcher. Even though the objectivity of the inclusion and exclusion criteria applied on the abstract was tested with two researchers (7 disagreements on 100 articles were observed), there is no guarantee that for the remaining articles there would not be more/the same number of disagreements. In order to reduce this threat, only very obvious articles were excluded, such as papers that are not in the area of software engineering, or clearly did not investigate productivity of software development. As a consequence of being inclusive during the screening, the number of full-papers checked (94 articles) was greater than the number of articles actually included in the review (38 articles).

Gray literature and Focus on Software Engineering Databases: Gray literature was not included as it is particularly hard to identify. Instead, we focused on the relevant research databases presented in the systematic review guidelines for software engineering (cf. [2,3]). However, we acknowledge that the inclusion of gray literature could have further increased the validity of our findings. In addition, the search was focused on the recommended databases for software engineering. Hence, software productivity articles that are only indexed in databases related to business administration or other fields might be missing.

Generalizability: The generalizability of the results is limited by the generalizability of the studies included in the review. The context information for each article has been extracted to take into account the degree of generalizability. Petersen and Wohlin [9] defined a checklist to identify relevant context information in industrial software engineering research, which has been used as a guidance when looking for relevant context information.

4. Classification scheme

The classification scheme is a structure of empirical studies on software productivity measurement and prediction. The scheme consists of six facets, namely quantification approach, abstraction, context, evaluation, research method, and measurement purpose (see Fig. 1).

Classification schemes/taxonomies are rated based on a set of quality attributes. A good taxonomy/classification is:

1. *Orthogonality*: There are clear boundaries between categories, which makes it easy to classify.
2. *Defined based on existing literature:* The taxonomy/classification is created based on an exhaustive analysis of existing literature in the field.
3. *Based on the terminology used in literature:* The taxonomy uses terms that are used in existing literature.
4. *Complete*: No categories are missing, so that existing articles can be classified.
5. *Accepted*: The community accepts and knows the classification/taxonomy.

Completeness is indicated as articles could be classified into the scheme without the need of adding further categories. Criteria two to three are supported as the classification and terminology used is based on existing literature (such as the distinction between measurement-based analytical and simulation-based models). Orthogonality is indicated as there was little doubt/uncertainty when classifying. However, the orthogonality has not been evaluated in the form of different persons classifying independently. The acceptance of the classification is not provided, given that it is a new classification scheme. A list of quality criteria for taxonomies/classifications can be found in [11].

4.1. Quantification approach

The quantification facet is a means of structuring the area of productivity measurement with regard to different methods of how to capture the productivity. This is important in order to group methods that are closely related, and by that helps in structuring complex phenomena.

Pfahl and Ruhe [14] provided categories of different types of models to evaluate software development. The types relevant to this systematic review are measurement-based analytical models and dynamic software engineering models.

Measurement-based analytical approaches model the relationship between dependent and independent variables. Here it is important to introduce the concept of constant returns to scale (CRS) and variable returns to scale (VRS). In the CRS case with an

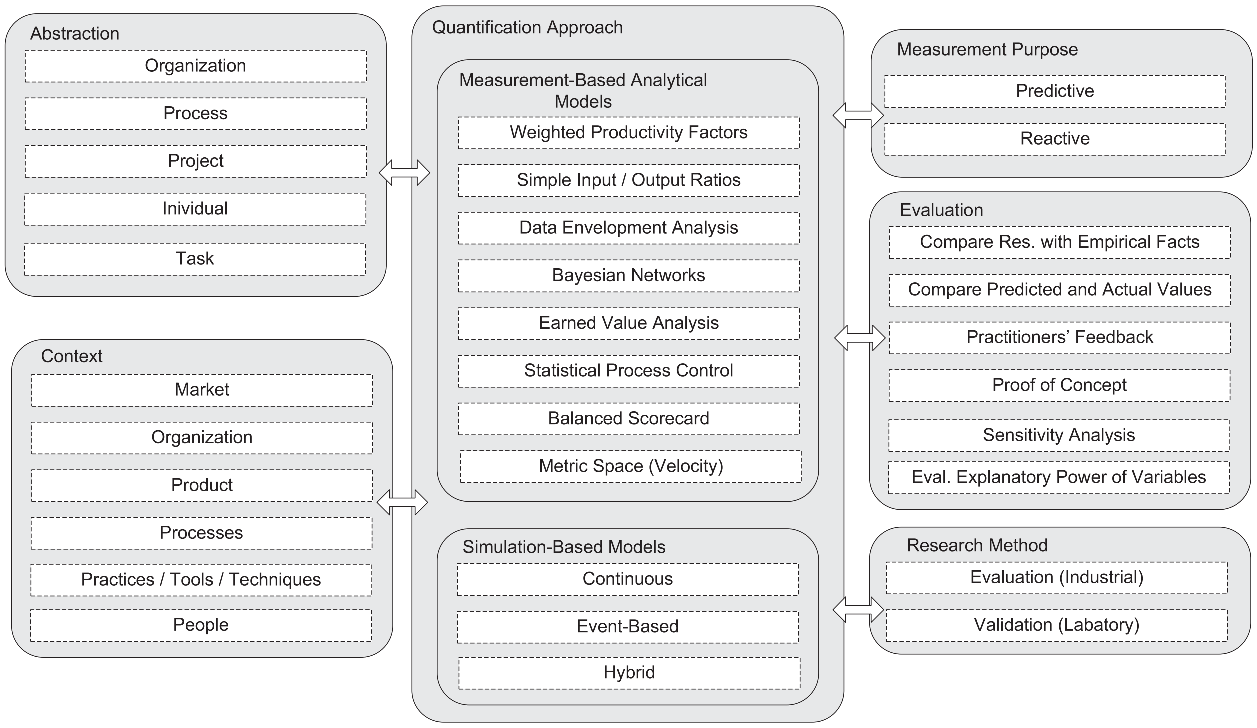

Fig. 1. Classification scheme.

additional unit of input the same amount of output is produced. On the other hand, the VRS case can have either an increase or decrease of outputs with additional units of input.

Dynamic software development models contain a number of sub-models, namely process enactment models, process simulation models, and system dynamic models. The models are used to describe the behavior of software development [14]. The models relevant in this study are process simulation models and system dynamic models. Process simulation models run on process descriptions in form of discrete event and queuing simulation or state-based simulation, they are event-based. System dynamic models follow a continuous simulation approach.

4.1.1. Measurement-based analytical models

Weighted productivity factors: The productivity is calculated by weighting factors influencing the productivity. A common approach to identify weights of independent variables to determine a dependent variable is regression analysis. Regression analysis models the relationship between variables (independent and dependent). In order to establish a model on the relationship between variables there has to be a relation. That is, the independent variables must have an explanatory power on the independent variables (i.e. they must account for the variation in the dependent variable). Different types of regression models are available, such as simple linear regression, non-linear regression, multiple regression, and stepwise regression. Further details on regression based modeling can be found in [15,16]. In addition to regression Pfleeger [15] created an adjustment factor for the average productivity to predict the actual productivity.

Simple Input/Output Ratios: Simple input and output based ratios are based on the classical productivity model, defined as the ratio of one output divided by one input. The model can be extended by adding up several inputs as well as outputs (see Eq. (1)). Observe that the inputs and outputs are not adjusted by a weighting factor.

$$P = \frac{Output_1 + Output_2 + \cdots + Output_p}{Input_1 + Input_2 + \cdots + Input_q} \tag{1}$$

Most common output is a size measure such as Lines of code (LOC) [17–19] or Function points (FP) [20–22]. The most common input is measured as effort used to develop the output [19,18,17,23].

Data envelopment analysis: Data envelopment analysis (DEA) is able to handle multiple inputs and outputs. The inputs and outputs are weighted and their ratio (productivity) should be maximized for each decision making unit. The weights are determined by the method. Data envelopment analysis determines the production frontier. On the production frontier the most productive decision making units (e.g. factories, projects, etc.) can be found, they are referred to as reference projects. The assumption of the analysis is that if firm A can produce Y outputs with X inputs then other firms with a similar amount of inputs should be able to produce the same output. DEA also allows to identify efficient reference projects for each inefficient projects. The reference projects are the ones the inefficient projects should be compared with. Peer values calculated through DEA indicate which reference projects a specific decision making unit should be compared with. The linear programming functions to solve the DEA problems for constant and variable returns to scale are presented in [24,25].

Bayesian belief networks: A Bayesian Belief Network is illustrated as a directed graph. Nodes show variables and the probability of the variables to take a specific value. Between the variables a cause-effect relationship exist. On the top a node for productivity is shown. The productivity can be defined in intervals of productivity values. Based on the probabilities assigned to the variables and the knowledge of cause-effect relationship the probability of achieving a specific productivity can be calculated. The cause-effect relationships are expressed in order to create the model. However, these relationships are not to be confused with empirically obtained knowledge about cause-effect relationships. Instead, it can be either a result of opinions of the model creators, or the model creators base their model on existing empirical studies. The same applies to system dynamics simulation, which is described in the context of simulation-based models. The rules for calculating probabilities and the application of Bayesian Belief Networks in software productivity prediction can be found in [26].

Earned value analysis: The paper introducing earned value analysis to software development can be found in [27]. The earned value is the output computed as the percentage of progress towards the final product. The progress can only be measured with clearly defined milestones. The input is the effort spent from the beginning of the project till the completion of a milestone. The productivity is the ratio of the earned value divided by the milestone cost. The calculation shows the shifts in productivity and the progress.

Statistical process control: Statistic process control uses statistical inferences and control charts to analyze the software process. A commonly used tool from statistical process control is the individual and moving range (I-MR) chart showing the individual values and moving ranges (size of change in the investigated variable) and variance around the mean over time, also referred to as time-series analysis. Furthermore, control limits (usually ±2 or 3 standard deviations away from the mean) are shown. If data points are outside the control limits then this indicates that the process is out of control.

Balanced scorecard: Balanced Scorecard (cf. [28]) is a tool to evaluate an organization based on a long-term perspective (strategies and visions). The scorecard includes different perspectives that are evaluated (e.g. finances, human and management aspects, process perspective, and customer satisfaction). The measures obtained are compared to the target levels defined based on the goals derived from strategies and visions. If there is a deviation from the target corrective actions should be taken.

Metric space A metric space allows to measure the distance between elements of a set or between functions (for further detail on metric spaces see [29]). For example, to compare the productivity of two programmers the distance between their accumulated work functions over time could be calculated (cf. [30]).

4.1.2. Dynamic software development models

For dynamic software development models continuous and event-based models are distinguished.

Continuous simulation: Continuous simulation models describe software development through mathematical equations and graphical representations [31]. The models are able to quantify cause-effect relationships and take probabilities and distribution of data into consideration. For example, a dynamic model could express change of workload over time based on a system model describing cause-effect relations and probabilities based on data distributions [32]. When running the same simulation model with the same calibration each simulation result might yield different results due to the introduction of probabilities in the simulation run.

Event-based simulation: Event-based simulation changes model variables with updates of events. Event-based approaches are state-based and queuing simulation State-based simulation is based on a graphical representation of the development process, modeled in form of petri nets or data flow diagrams and can be used to predict various variables of software development (e.g. cycle-times, costs of development, and software quality) [31]. Queuing theory has been applied to the performance of software processes by representing the process model in as a queuing network with multiple and interconnected servers (see e.g. [33,32]).

Hybrid simulation: State-based simulation and queuing simulation can take dynamic/continuous simulation characteristics when being combined with system dynamics models.

4.2. Context

The context describes the environment in which an object of study (in this case the approach of quantifying productivity) is embedded [9]. In general, industrial and experimental studies are conducted in different contexts, and these contexts describe the situation in which the study results are true". Hence, context as a facet is included to be able to evaluate the generalizability of study results and to support comparability.

In the following the context elements are described and rationales are provided why the elements are relevant when studying productivity measurements. We do not claim completeness of the examples, and depending of what measurement approach or purpose is studied the different context elements might be of relevance.

Market: The market plays a role in productivity as it might constrain productivity variables, e.g. by defining a market-window for release or putting constraints on costs. These are important factors when interpreting and making use of results of productivity measures.

Organization: The organizational units (e.g. projects) need to be characterized and described. This is important when comparing different projects. For example, one should be aware of the size of the project in terms of number of people involved when comparing its productivity.

Product: Product attributes are often used as output measure in the form of size and quality. To compare size one has to be aware of the type of the product (e.g. when counting features or functions these are different for an end-user application or an embedded application interacting with hardware). Quality aspects are of relevance as the constraints on quality have an effect on the productivity outcomes. In addition the maturity of the product plays a role when comparing productivities. When measuring a new product the productivity figures ook quite different in comparison to a mature product that is extended. For example, the new product might look less productive as for the new product more up-front investigation is necessary in comparison to building smaller increments to an already mature and well known product.

Processes: The workflow of software development is important, e.g. the productivity prediction of highly iterative development with unstable requirements is different from prediction with stable requirements and a waterfall approach.

Practices/tools/techniques: Practices, tools, and techniques are, for example, relevant when predicting productivity the change of a practice or introducing a new tool.

People: People represent roles and experiences and are important to consider as they should be supported with the measurements in evaluating and improving software development.

4.3. Abstraction

The abstraction describes on which level the measurements and predictions are carried out. Considering abstraction is important as different entities (e.g. projects, and processes) might have different attributes, which can be measured in different ways. Providing a structure for abstraction also allows to determine the mapping of certain productivity measures to abstraction levels. The abstractions are based on the abstraction levels found in the included productivity articles. The abstractions of organization, project, individual, and task are measured as black-boxes. That means the dependent variables (output) and the independent variables (input) are observed and quantified without having to consider what is happening inside the box.

Organization: Here the productivity of an overall software organization (e.g. development site) is determined.

Project: A project executed by a team is evaluated, the main characteristic of the project being that it is only exists temporarily.

Individual: An individual's (e.g. programmer) productivity is determined.

Task: The productivity of a software development task (e.g. a programming task or requirements specification task) is quantified.

At the process level productivity is evaluated considering what is going on inside the box.

Process: The abstraction of the process considers the interactions between different activities in the software process and takes additional variables into account (e.g. the resources needed to execute an individual activity). It is also common to model the artifacts and their flow through the process.

4.4. Measurement purpose

Two measurement purposes are distinguished, based on the focus of the systematic review. The first purpose is reactive, i.e. to determine the productivity based on data of the past. The second purpose is predictive, meaning to quantify the productivity expected in the future. The motivations of why to measure in a reactive way and to predict have been presented in the introduction.

4.5. Evaluation

The criteria define the measure of success or failure of an intervention, which is essential in judging the outcome of an intervention study (the intervention in this case being productivity prediction and measurement approaches). One part of the data extraction was the documentation of the evaluation results, and how the evaluation has been conducted. The following approaches to evaluation have been identified during the data extraction.

Compare results with empirical facts: Here the quantified productivity is compared with empirical facts. In the case of productivity prediction the predicted impact of improvement actions can be compared to expected outcomes based on empirical facts (see e.g. [34,32]). For example, when evaluating the impact of early defect detection it should be expected that this has a positive impact on productivity related variables.

Compare predicted and actual values: The values determined by the predictive model are compared to actual values observed in industry or experimental studies. Examples of comparisons between actual and predicted data can be found in [35,36,26].

Practitioners feedback: The usefulness and accuracy is determined by presenting the data to practitioners providing feedback. One example is to provide them with a list of productive projects identified by the measures used and ask them if they perceive these projects to be the most productive ones (see e.g. [24,37]).

Proof of concept: The proof of concept is done by applying the measures on a set of data (either from the lab or from industry) and reflecting on it. Proof of concept solely relies on the reflection on the collected data by the researcher, and does not include any other evaluation criteria.

Sensitivity analysis: Sensitivity analysis tests the robustness of the proposed measures and predictions to change in the data set. The sensitivity is, for example, tested by removing extreme values from data data set and then observe whether the mean values change. Little change in the mean values is an indication of the robustness of the model. An example of removing extreme values can be found in [24].

Evaluation of explanatory power of variables: Explanatory power shows to what degree the variance in a dependent variable (e.g. lines of code productivity) can be explained by a number of factors considered in a model. The goal is to achieve as high explanatory power as possible as this indicates the predictive ability of the independent variables over the dependent variable.

4.6. Research method

Wieringa et al. [38] provided categories of how to classify research methods, the relevant classifications for this review being validation research and evaluation research.

Validation research: In validation research solutions are characterized by the novelty and often have not been implemented in practice. Validation research mainly focuses on the evaluation of solutions in the lab (e.g. controlled experiments [39] or simulation of scenarios [32]). Validation studies should in an ideal case always have theoretical discussions of limitations, and problems with the modeling/measurement approach.

Evaluation research: Solutions are implemented in industry and are evaluated in the context in which they are employed. Evaluation research shows how the technique has been implemented in industry (solution implementation). To strengthen the evaluation research the consequences of the solution implementation need to be explored, such as strength and weaknesses. Evaluation research also includes the identification of problems related to practices. Case studies [40,41] and action research [42] are often used to investigate interventions in industry. In productivity research databases containing productivity data from past projects play an important role and as they are based on industrial data they are classified as evaluation research. The data contained in the databases has to be analyzed with care. For example, old datasets often do not report important variables (such as planned and estimated effort).

Distinguishing those groups is important as well, as both approaches have their advantages and disadvantages. In validation research it is often challenging to create realistic situations that allow for generalizability to industrial practice, but it is easier to claim cause-effect relationships due to a controlled environment. Evaluation research is done under realistic conditions, but is not conducted in a controlled environment. Consequently, it is not always clear whether unknown confounding factors influence the study outcome.

5. Mapping results

5.1. Number of identified articles

Fig. 2 shows the number of articles remaining after each step of the article selection process. In total 586 articles were identified in

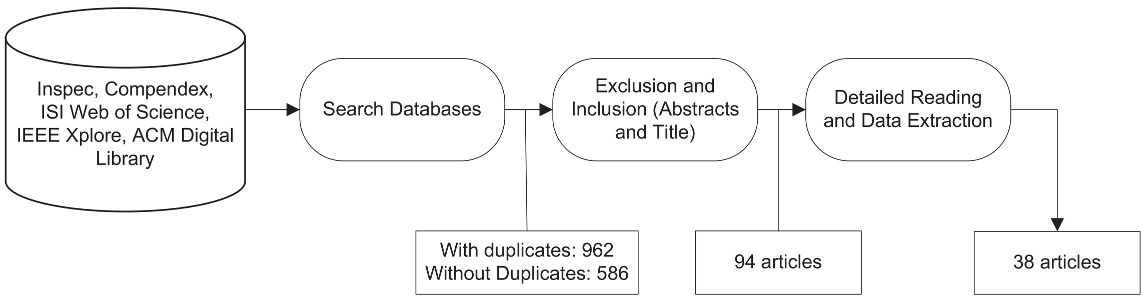
Inspec, Compendex, ISI Web of Science, IEEE Xplore, ACM Digital Library
Search Databases
Exclusion and Inclusion (Abstracts and Title)
Detailed Reading and Data Extraction
With duplicates: 962
Without Duplicates: 586
94 articles
38 articles

Fig. 2. Exclusion of articles and number of primary studies.

the data bases after the removal of duplicates. Thereafter, articles were excluded based on the inclusion and exclusion criteria leaving 94 articles for review. As mentioned in the section describing the threats to validity we were inclusive when deciding which articles to review in detail to not exclude any relevant work from the dataset. That is, when uncertain the article was included for detailed review. Throughout the detailed review further articles were discarded, leaving 38 primary studies being included in the analysis of this review. The reason for excluding a high number of articles is that productivity is mentioned in connection with many studies. However, in the abstract it was not always clear that productivity was not the main focus. Furthermore, studies were excluded if only the measurement or prediction model was theoretically described in the paper without data from an application, be it in the laboratory or in an industrial context.

If studies receive a moderate score in the quality rating regarding scientific rigor we did not exclude them, but rather reported the results together with the quality rating. In the interpretation this allows to reflect on the results with the scientific rigor in mind. Studies with moderate rigor might contain interesting ideas that are worthwhile to be captured and are applicable. Furthermore, in aggregation studies of moderate rigor can serve well for triangulation purposes to support certain findings. No studies with a lower score than three (with eight being the maximum score that could be achieved) were identified (see Table 17 in Appendix A).

5.2. Publication forums (Mapping Question 1)

Table 3 shows the top seven publication forums for software productivity research. The main forums for publication were journals, namely Journal of Systems and Software published by Elsevier and IEEE Transactions on Software Engineering. The second most common forums were the International Conference on Computer Software, the Software Process Improvement and Practice Journal (Wiley & Sons), and the Information and Technology Journal (Elsevier), each having published three papers on quantifying productivity. The International Conference on Software Process and the International Conference on Software Engineering have published two papers related to productivity quantification.

5.3. Trends over time (Mapping Question 2)

Topic areas with only one publication have been omitted from the analysis to increase readability, as the single papers do not allow to see a trend. The analysis of trends showed that the adjustment and weighting of productivity factors to explain a dependent variable has been published in research articles throughout the 1990s and 2000s. For the ratio based analysis we can see that the main publication focus was in the early and late 1990s (five papers in total), with one additional paper after 2005. Data envelopment analysis has been discussed in the early and late 1990s. However, the majority of the publications on the topic have been published in the time-span of 2001–2008 with a total of five papers. Software process control has only few papers in total, which are published throughout the years. Event-based simulation has a publication focus in the late 1990s and early 2000s. Continuous simulation approaches have been published throughout the years and are not concentrated on a specific time period. Two studies were published on hybrid simulation, one being published in the late 1990s and one in the early 2000s. Overall, there are few papers in each category (a maximum of seven papers for data envelopment analysis). Therefore, a trend-analysis is not feasible. Hence, no answer to Mapping Question 2 with respect to trends can be provided.

Table 3
Publication forums.

Rank	Forum	No. of papers
1	Journal of Systems and Software	4
1	IEEE Transactions on Software Engineering	4
3	Intl. Conf. on Computer Softw. and Appl. (COMPSAC)	3
3	Software Process Improvement and Practice Journal	3
5	Journal of Information and Software Technology	2
5	Intl. Conf. on Software Process	2
5	Intl. Conf. on Software Engineering	2

5.4. Research focus (Mapping Question 3)

The approaches investigated with the highest frequency were weighted factors (seven studies), data envelopment analysis (seven studies), ratio based quantification (six studies), event-based simulation (five studies) and continuous simulation (four studies) (see Fig. 3). The quality ratings of each study are shown in detail in Table 17 in Appendix A. The highest scores for rigor can be found for studies investigating weighted factors and data envelopment analysis, both with at least five studies having achieved a score equal or larger than six (see Fig. 3). For the simulation approaches most of the studies achieve a score of four or five. The approaches earned value analysis and metric spaces only have one publication, and at the same time achieve relatively low scores. Studies focusing on the approaches weighted factors, simple ratio, and data envelopment analysis are primarily based on industry application. Simulation is frequently working without getting data directly from industry, i.e. the model is calibrated based on hypothetical data, or the calibration relies on already published data (e.g. from experimentation).

Studies on weighting factors, simple ratio, and data envelopment analysis focus on either projects, individuals or tasks (see Fig. 4). That is, the development of software on these abstractions is treated as a black-box where the relation of depended and independent variables is investigated. On the other hand, simulation looks at the inside of the box, e.g. the workflow of activities, the assignment of work-tasks to individuals, and the flow of development artifacts is important and can change the outcome of the productivity. Two exceptions can be found for the simulation approach. One study proposed a simulation model only based on

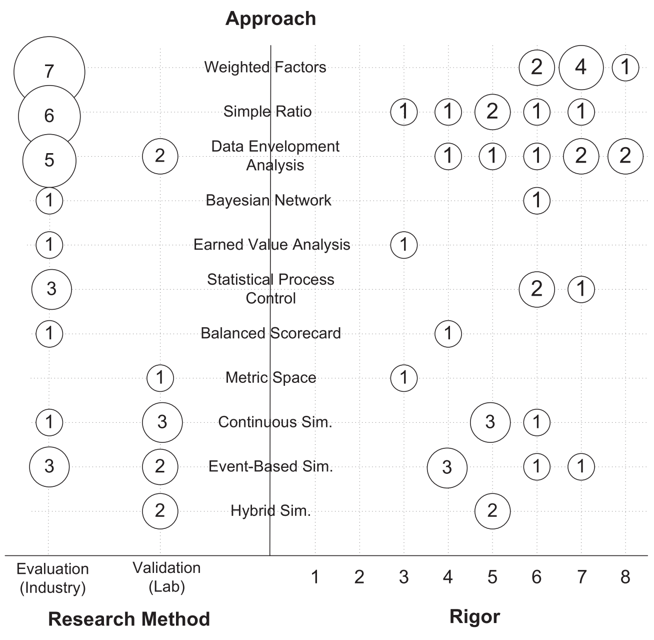

Fig. 3. Quantification approach and evidence.

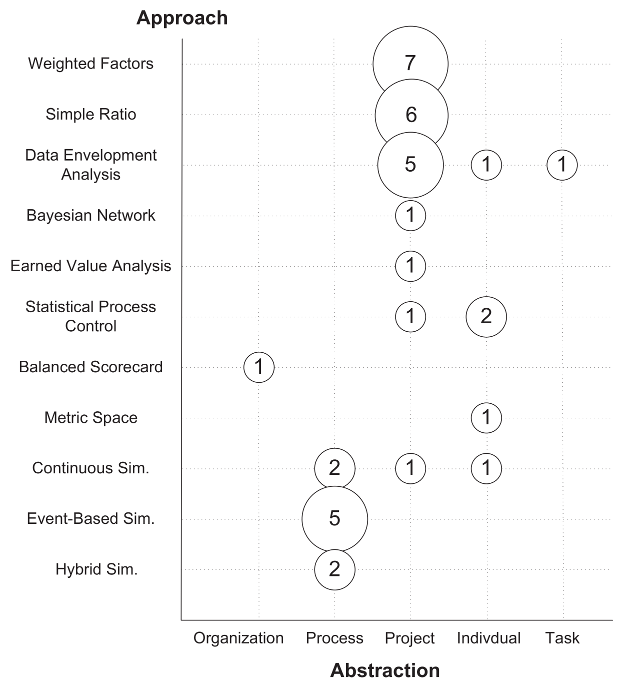

Fig. 4. Quantification approach and abstraction.

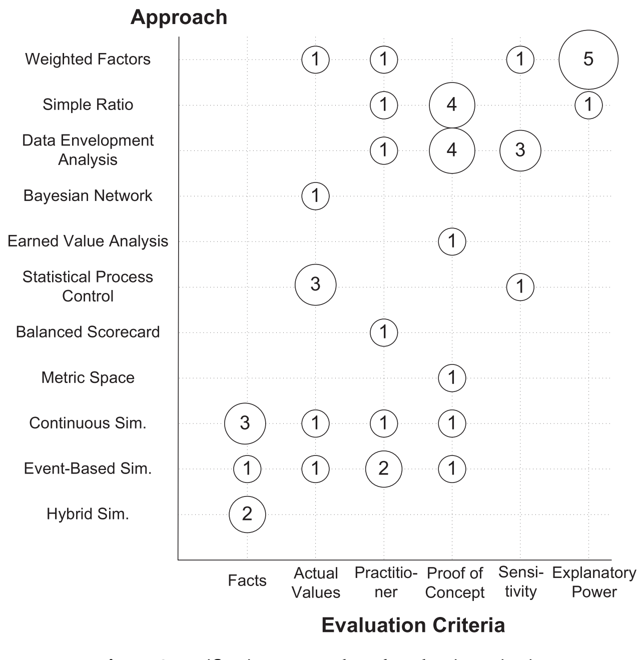

Fig. 5. Quantification approach and evaluation criteria.

inputs and outputs, and another model focuses on modeling the productivity of individuals considering learning while executing development tasks.

The frequency of studies using different evaluation criteria to judge the approaches are shown in Fig. 5. The total number of studies reported in the figure is larger than total number of studies included in the review, because one paper can make use of several evaluation criteria. The figure shows that all approaches are frequently used, but there are differences between the approaches. The majority of studies focusing on weighted factors evaluate explanatory power. Simple ratio measures and data envelopment analysis most often make use of proof of concept, i.e. the approach is applied on industry data and the evaluation solely relies on the researcher's interpretation of the data showing the applicability of the approach. Simulation strongly focuses on comparing results obtained by the simulation model with known facts and expectations. The figure also shows that many studies on approaches used for prediction do not compare their predictions with actual values, in fact only one study on simulation and one on weighted factors makes use of actual values. Also, only few studies include practitioners in the analysis to provide feedback. Few studies apply triangulation by making use of several criteria (see Table 18, papers [37,24,55,58]).

6. Review results

This section describes the results related to the systematic review questions presented in Table 2. The results of each study are tabulated and the tables are shortly described to highlight the most important results. The detailed narrative summaries of each study are presented in the complementary material to this article [5]. The synthesis and recommendations to practitioners based on the review results can be found in Section 7.

6.1. Measurement-based analytical models

6.1.1. Weighting productivity factors

Table 4 provides an overview of the studies that built predictive models by weighting productivity factors. In this and all forthcoming tables the studies are ordered chronologically. In [43] a regression model based on the variables lines of code (LOC) as well as maximum number of staff was presented and tested in two environments. In a third environment additional variables were introduced. Overall, the conclusion was that there are similarities in the three environments with regard to predictors, the main variables being size and maximum staff. Pfleeger [15] proposed a prediction model which adjusts an average productivity value by an adjustment factor, which pools different cost factors together. The individual analysis steps of the approach are described in the complementary material [5]. The outcome of the study was that the model performs better than COCOMO and Ada-COCOMO considering error margin and hit/miss ratios with respect to actual values. Finnes and Witting [44] tested different forms of models (linear, power, exponential, reciprocal) and found that the reciprocal model fits best for the variables team size and system size. In addition, combined models have been evaluated (see Table 4). Maxwell et al. [17] evaluated predicts using a linear model. They investigated which categorical and non-categorical variables, and a combination thereof, can be used as a predictor for productivity. The variables with the highest explanatory power are shown in the table, a complete list and its explanation is provided in the complementary material (cf. [5]). A multivariate logistic regression model was evaluated by Morasca and Russo [45]. In their logistic regression the dependent variable (Y) was ordinal/nominal, and discretized into classes. The regression was used to predict whether a dependent variable belongs to one of the classes. They found significant predictors for different types of productivity (function point productivity, lines of code productivity, and module productivity) for an entire application and for individual new development requests, the predictors being named in the table. Kitchenham and Mendes [37] used stepwise regression to create a productivity measure based on multiple size measures, the size measure having a significant relationship with the effort. The approach has been successfully applied with positive feedback from practitioners regarding its ability to identify good performers, and at the same time is not sensitive to extreme values. Foulds

Table 4
Quantifying productivity based on weighting factors.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[43]	1987	Stepwise linear regression	Explanatory power of the model in the value of R^2	Management information systems with varying size, language, effort, and number of team members	Project databases from three different environments containing 9, 10, and 19 projects	High similarities in three different environments with regard to predictors; main variables are size and maximum staff; experience and attitude change productivity, but do not explain a large portion of the variance	6
[15]	1991	Adjustment of average productivity by factors	Comparison of actual productivity figures with estimations by models (proposed solution, COCOMO, Ada-COCOMO)	Object oriented development projects at HP	Two project managers per project estimated the cost factors. Data based on a company-wide program for collecting software metrics	Proposed solution predicts within 25% error margin 50% of the time, COCOMO and Ada-COCOMO predicted outside 25% error margin	8
[44]	1994	Linear, power, exponential, reciprocal regression model	Explanatory power for team and system size on function point productivity	Organizations developing data processing systems of different sizes	Interviews with information system managers from 15 organizations	Reciprocal model fits best for team size and system size; for combined effect the model fitting best was the regression of average team size, function points per team member, and ln(function points) against ln(function point productivity)	6
[17]	1996	General linear model	Explanatory power of general linear model	Domains investigated in the survey were military (39%), space (28%), industrial (24%), and others (9%)	99 completed projects from 37 companies and eight European countries; collection of data through questionnaires and telephone interviews	Top three categorical variables: individual company, programming language, type of system; top three non-categorical variables: storage constraints, execution time constraint, use of tools; best combined models: category and language as well as country, category, reliability, and either tools or modern programming practices	7
[45]	2001	Multivariate Logistic Regression	Explanatory power and significance of the coefficients	Mainframe applications (development of new functions as well as maintenance of the mainframe applications)	29 applications (six variations of same application) and hence excluded 88 requests (74 development requests, 41 maintenance request)	*Entire application:* function points and experience were statistically significant predictors for function point productivity, lines of code productivity, and module productivity; *new development requests:* development time, function points, and number of employees are statistically significant predictors for function point productivity and lines of code productivity	7
[37]	2004	Multiple Size Measure/ Regression	Evaluation through practitioner feedback and sensitivity analysis	Web projects	Projects from Tukutuku database and interview with one practitioner	Practitioner agreed with interpretation of productivity of the results for 12 out of 13 projects; the model is stable as removal of high influence projects does not result in major changes, though explanatory power increases	7
[46]	2007	Partial least-square based analysis	Reliability analysis of the factors and its related indicators; bootstrap (stepwise) procedure of partial least-square analysis	Large-scale information system developers, 9 of 11 have more than 1000 employees, 10 organizations more than 50 IT staff, one company with more than 100 developers	110 surveys of information system developers with 60 responses received	Explanatory value of the model derived (measured as R^2) is 0.634. Significant: product and project related factors; insignificant: IT and environmental factors	7

et al. [46] used factors with several variables (indicators) associated to them to create a productivity model. The weights were calculated using partial least-square analysis. The finding was that project and product related factors were significant, while IT and environmental factors were not.

6.1.2. Simple input/output ratios

An overview of productivity measures using simple input/output ratios is provided in Table 5. Yu et al. [18] proposed composed lines of code productivity where lines of code and effort are split into different components, e.g. lines of code is split into new, reused, and changed code. As shown in Table 5 several improvement actions have been triggered by the measurement. Chatman [23] introduced the concept of change point productivity. A change point represents a change on modules on the lowest abstraction level, further details on the approach being provided in the complementary material [5]. The researchers found that change points have advantages over function points (see Table 5), as change points remove technology dependency and are less prone to bias due to more objective counting rules. The study of Maxwell et al.

Table 5
Quantifying productivity based on simple input/output ratio.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[18]	1991	Composed lines of code productivity	Proof of concept through application and observed improvements (though not quantified)	Company: AT&T; system size: several million LOC; effort dev. 75% to effort hardware 25%; system: distributed architecture in a real-time system; programming language.: C; process: waterfall	Project productivity database from AT&T for one product	Concrete improvements made based on productivity measures (e.g. increased stability and completeness of requirements specification, better traceability, etc.)	4
[23]	1995	Change point productivity	Proof of concept by application of the approach in industry and comparative reflection with other methods by the author while introducing productivity measures at the company	Company: SLT	Data source are logs and databases maintained at the company; researcher actively involved in the company in implementing productivity measures	Change points reduce the bias observed in function points; can be easily counted, have high stability; they allow for overlapping process steps, independence of implementation in different programming language, and possibility of automated counting	5
[17]	1996	Lines of code productivity vs. Process productivity	Evaluate whether variables explain variation in the two types of productivity. Claim: process productivity better because it covers a wider range of factors in the equation	Domains investigated in the survey were military (39%), space (28%), industrial (24%), and others (9%)	99 completed projects from 37 companies and eight European countries; collection of data through questionnairs and telephone interviews	Result: no evidence that supports the superity of process productivity measurement	7
[47]	1998	Ratio of use case points and effort	Evaluation through industry feedback (check acceptance of the approach in industry)	Banking domain, Application sizes of 200–4400 use case points. Duration of projects 1–1.5 years	23 projects at the bank. Data collected from an 64 evaluation questionnaires and 11 post benchmark interviews with project managers	*Positive:* method accepted at the studied company and measures considered reliable. *Negative:* scenario modeling in use cases not well understood; projects differ in completeness; use cases not up-to-date; high variety in requirements complexity; ambiguous descriptions complicate reproduction of measures; fragmentation of using different models leads to inconsistencies	3
[20]	2000	Function point productivity	Practitioners feedback and observations made by the researchers	Large-scale information system department with a variety of different applications at of ECHO (company)	Interviews, checking of the databases containing the FP-data, and observations of meetings in which data was discussed	Problems: high fluctuation in measurement results that cannot be explained; inability to compare results with other companies; measurements contradict with perceived productivity; many tasks require effort and are not reflected in function points; user management is not involved	4
[21]	2007	Function point productivity	Evaluation based on measures taken at the company and the reflections of the researchers	Application Management Services group IBM Australia Application Management Services(CMM level 5)	Data set provided by application management services group for researchers to reflect on and provide feedback	Problems: productivity data seldom normally distributed with large spread of data; aggregation of data destabilizes mean; averaging two unrelated variables leads to instable results; when plotting single productivity values important information is lost. Recommendations: use scatter plots for univariate analysis; transform data to achieve normality; analyze data from similar projects	6

[17] is included in two categories because it made two contributions, the first one identifying predictors and the second one evaluating process productivity as a measurement. Process productivity adjusts effort by a skill-factor, and takes development time into account. A comparison of process productivity with the simple lines of code productivity showed that process productivity is not superior in taking into account variances due to productivity factors. Arnold and Pedross [47] evaluated use case point productivity and found positive as well as negative aspects. The positive aspect was the acceptance of the model model, while challenges were related to variance in complexity and the documentation of use cases (see Table 5). Bok and Raman [20] evaluated function point productivity and identified several issues in the studied company, the issues being stated in the table. The issues were mainly related to the counting of function points. Kitchenhem et al. [21] reflected on function point productivity measurements and discussed problems and solutions related to them. The problems identified in their study were mainly related to the comparability of projects and the loss of information when building ratios of two variables. A number of countermeasures have been proposed (see Table 5).

6.1.3. Data envelopment analysis

Studies that evaluated data envelopment analysis as an approach for measuring productivity are summarized in Table 6. Banker et al. [48] proposed a stochastic data envelopment analysis approach, which takes into account that a deviation from the production frontier is not only caused by efficiencies, but also by other factors (e.g. measurement errors). The approach was compared to regression-based analysis, comparing the weights the models produced (see Table 6). The paper makes an important contribution by recognizing that deviation can be attributed to error, but no solution of quantifying of error to inefficiency was proposed. Mahmood et al. [49] evaluated data envelopment analysis CRS. The research-

Table 6
Quantifying productivity based on data envelopment analysis.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[48]	1991	Stochastic Data Envelopment Analysis	Comparison with parametric estimation of coefficients; sensitivity analysis	Financial transaction processing system	65 maintenance projects at Canadian bank	Weights of stochastic DEA and parametric model for production function are similar; value of weights is similar with different ratios of error due to inefficiencies and weights; factors with greatest impact on the objective function can be evaluated by omitting variables (most important factors are use of structured methodology and good response time)	8
[49]	1996	Data Envelopment Analysis CRS/ additive	Proof of concept by applying DEA and reflection on it	Information Systems	78 commercial software projects	*Observations:* magnitude of inefficiency identified (efficient projects outperformed efficient ones by ratio of as much as 10–1); given the split of time between phases it becomes clear which phases need improvement; too little time in one phase can lead to high consumption in another phase; *Implications:* pre-requisite is consistency of measures (e.g. same way of counting inputs and outputs); productivity is multi-dimensional and model need to consider this; focus on time spent to identify improvement opportunities; use and re-iterate DEA to continuously improve productivity	7
[24]	2003	Data Envelopment Analysis CRS/ additive	Sensitivity analysis; confirmation of results with practitioners; comparison with regression based analysis	ERP projects delivering software and business re-engineering	48 completed ERP (multivariate) and Albrecht-Gaffney dat set (univariate)	DEA VRS identifies more reference projects (six) in comparison to CRS (one) indicating that it is not meaningful to compare projects of different sizes; sensitivity analysis indicated robustness of DEA with regard to the production frontier; DEA is better than regression as regression identifies large/small projects as the best/worse performers	8
[50]	2004	Data Envelopment Analysis (CRS and VRS, multivariate)	Comparison of efficiency scores between VRS; evaluation of impact of size; comparison of ratio-based vs. DEA	Maintenance projects fixing the Y2K problem	66 software development projects of a Canadian bank	Observed data shows VRS behavior indicating DEA VRS as appropriate method; size changes production frontier of efficient projects; DEA CRS similar to ratio-based measures while VRS more fine-grained in identifying efficient projects; ratio only considered one input at a time while DEA can handle multiple inputs and outputs	7
[25]	2005	Data Envelopment Analysis (CRS and VRS, multivariate)	Comparison of DEA VRS and CRS; inspection of the data as indicator of whether the strong performers have been identified; ability of DEA to identify reference tasks	10 student exercises evaluated using the personal software process	Student's recordings from their execution of the process	*Observations:* variable and constant returns model agree on most of the efficient exercises; model exhibits decreasing returns to scale; inspecting the original data efficient performers have been identified well. *Problems:* model sensitive to outliers; no comparison to theoretical maximum; no confidence intervals as method is non-parametric	5
[52]	2006	Data Envelopment Analysis VRS	Proof of concept by applying DEA on data and reflection	Domain: bank	158 completed projects from ISBSG dataset and 43 projects from case company	*Observation:* average efficiency score increases with number of outputs; average improvement potential identified is 16-28% depending on selected outputs; up to 53–60% improvement potential identified; important to include time as a variable as projects with very short time require more resources	4
[53]	2007	Data Envelopment Analysis (CRS and VRS, multivariate)	Evaluation of hypotheses regarding ability to identify efficient tasks and reference data sets; comparison of VRS and CRS model; sensitivity analysis	Research and development organization(CMMI level 4)	Data from 30 completed development tasks (recored with SoftPM tool)	Efficient role models as well as reference projects for inefficient tasks could be identified through peer weights; variable returns to scale model more sensitive when identifying efficient tasks; removal of frontiers from data set does not change mean significantly which indicates model stability	6

ers observed inefficiencies and located phases where improvements are needed. The implications of the study are summarized in the table. Stensrud and Myrveit [24] used DEA CRS and VRS to analyze different data sets. They argued that VRS is more sensitive in identifying suitable reference projects, and that DEA is robust with regard to the sensitivity analysis conducted. Furthermore, DEA is better than regression as it is not biased towards the majority of projects. Yan and Paradi [50] compared DEA VRS and DEA CRS models, evaluated the impact of size on the production frontier, and compared DEA VRS with ratio-based measurements. They

Table 7
Quantifying productivity based on bayesian belief networks.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[26]	2003	Bayesian Belief Network	Ability of the model to estimate the productivity correctly for the Cocomo81 dataset	Given by the Cocomo81 dataset	63 projects in the Cocomo81 dataset	52% to till 67% of the projects were classified into the correct intervals (increase to 67% when considering neighboring intervals)	6

Table 8
Quantifying productivity based on earned value analysis.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[27]	1992	Earned Value Analysis	Proof of concept through application	Embedded systems domain; aircraft on-board system	N.A.	Observations: observation of variance of productivity over time possible; earned value analysis allows to detect shifts in development performance; illustration of potential root-causes identified by the solution presented	3

Table 9
Quantifying productivity based on time-series analysis/statistical process control.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[55]	1991	Time-series analysis and prediction based autoregressive model/ Yule-Walker equations	Comparison of predicted and actual data; sensitivity analysis	N.A.	One programmer over 17 sequential programming exercises	*Predicted vs. actual:* actual values close to lower prediction interval. *Sensitivity:* in first scenario large increase and immediate drop in productivity, large initial error with quick decay to reflect the actual productivity; in second scenario large increase, large error in the beginning with quick decay	5
[36]	2005	Statistical process control with dynamic calibration	Process changes: look for actions leading to change and check whether model detects them. Accuracy: compare the error made in estimation with dynamic calibration and analogy from previous studies	Renewal projects (maintenance) in banking domain	Data for relevant variables entered by developers and reviewed weekly by project managers. Improvement actions (events) known and documented	*Process changes:* shifts in process performance successfully detected; shifts in process performance that did not require calibration were also detected. *Accuracy:* DC-SPC was more accurate than prediction by analogy and dynamic calibration	6
[54]	2006	Time-series analysis and prediction based on optimization	Comparison of the model's accuracy with dataset from [55]	N.A.	One programmer over 17 sequential programming exercises	Comparison with Humphrey prediction showed improvement of prediction variance from 0.0447 to 0.0420 (Improvement of 6.04%)	5

found that (1) the data they studied observed DEA VRS behavior, (2) size has an impact on the production frontier when grouping projects into small, medium, and large, and multivariate analysis is not well supported by ratio-based measures. Liping et al. [25] analyzed DEA VRS and CRS applied to development tasks in student projects. Their conclusion was that DEA VRS and CRS agreed on most tasks on the production frontier except one, but the observation of the data indicates decreasing returns to scale. Limitations observed in their discussion were model sensitive to outliers, no comparison to theoretical maximum, and no confidence intervals as the method is non-parametric. Asmid et al. [52] applied DEA and reflected on the outcome. The analysis showed that a large improvement potential was identified and that the average efficiency increased with the inclusion of variables. The authors also pointed out that time should be included as the required resources increase with shorter time frames. Ruan et al. [53] found in their DEA evaluation that efficient role models were identified, and that DEA turned out to be robust with regard to the sensitivity analysis conducted.

6.1.4. Bayesian networks

Stamelos et al. [26] introduced Bayesian Belief Networks to predict software productivity. They built their model based on the Cocomo81 dataset and evaluated whether their model is able to classify projects correctly into productivity intervals based on the characteristics of the projects. The intervals were determined by using the log-normal distribution assumed by the COCOMO model. In result the model achieved to classify over 50% of all projects correctly based on the project characteristics. Considering neighboring intervals the accuracy increased further (see Table 7). The complementary material [5] contains further details on the construction of the Bayesian Network used in the study.

6.1.5. Earned value analysis

Kadary [27] introduced earned value analysis as a measure of productivity from a value-adding perspective to the software product throughout the software development life-cycle. The paper illustrates potential root-causes for poor productivity performance that were identified using the earned value analysis as a driver. The author summarized the benefits of the approach in his reflections: (1) the measurement has a focus on delivery of product value rather than focusing on specific activities; (2) the root-cause identification and measurements help to justify improvement actions; and (3) the measure is sensitive in detecting unproductive work (see Table 8).

6.1.6. Statistical process control

Three studies using statistical process control for predictive purposes are shown in Table 9. Humphrey and Singpurwalla [55] introduced time-series analysis and moving averages in order to predict productivity of individual programmers. The authors conclude that their approach evaluated in two different scenarios allows to detect sharp shifts in productivity performance with an initially large error that quickly disappears. The study by Ruan et al. [53] proposes a modification to the approach by Humphrey and Singpurwalla [55] by changing the way the autoregressive parameters are estimated. The change lead to an improvement of 6.04% with regard to the estimation error. Baldassare et al. [36] proposed to combine statistical process control with dynamic calibration (DC), DC being concerned with when an estimation model has to be recalibrated. The result of the study showed that the approach is able to detect shifts in process performance, and it recognizes when an estimation model should be recalibrated. A comparison with other estimation approaches (dynamic calibration without statistical process control, and estimation by analogy) showed that the proposed model is more accurate.

6.1.7. Balanced scorecard

List et al. [56] investigated balanced scorecards as a tool to measure the performance of software development considering different stakeholders. The measures were based on the goals of the stakeholders, each stakeholder representing a dimension. The evaluation through action research showed that the introduction of the scorecard was beneficial. The scorecard became part of everyday work and was the driver for improvement discussions. Problems became visible and actions were taken to tackle them. One

Table 10
Quantifying productivity based on balanced scorecard.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[56]	2005	Balanced Scorecard for Software Engineering	Action research by participation of research team in introduction of balanced scorecard and interviews	Multinational organization in production domain	Observations made throughout action research in company studied and interviews with key process stakeholders one year after introduction	Measurements adopted in everyday work; interviewees reported that they were able to identify problems and improve positively; transparency improved from strategy down to metrics	4

Table 11
Quantifying productivity based on metric spaces.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[30]	2005	Metric Space for programmers based on accumulated work	Evaluation of method in lab based on student performance	Two students conducting parallel programming	Data from students in graduate-level course solving a programming assignment; data collection at each compilation	Approach allowed seeing difference between productivity over time as well as different approaches of achieving the work task	3

Table 12
Quantifying productivity based on continuous/system dynamic models.

Ref.	Year	Approach	Evaluation	Context	Data source	Outcome	Rigor
[57]	1985	Continuous simulation of size and effort	Application on a numerical example to apply dynamic simulation	N.A.	Hypothetical dataset	Model was able to provide confidence intervals for the selected example, the main feature of the model being that it is able to show the probability of occurrence/density with regard to productivities	5
[58]	1997	System dynamics model	Sensitivity analysis by varying multiple input parameters to model; confirmation of model reaction by 23 practitioners and empirical knowledge; comparison of estimation with actuals of a real project	Calibration project: 128,000 LOC, 1621 work weeks, 95 weeks project duration, staff size 18 people. Comparison with actuals: system: the COBE/AGSS system was designed and implemented to support the COBE spacecraft mission, Duration: 2 years; size: 94k LOC estimated and 163kLOC actual; language: Fortran and Assembler; waterfall development	Interviews for model construction; COCOMO data-set for sensitivity analysis; real-world project for comparison with actuals	The reaction of changes to model variables yields expected results and are in-line with empirical evidence (e.g. COCOMO); practitioners confirm behavior of the model for variable change based on experience; no statistically significant difference between actuals and estimation for estimated parameters	6
[59]	1998	Systems dynamic simulation model based on learning	Test whether what-if scenarios lead to expected results	Prediction of productivity for different scenarios based on different knowledge levels and learning abilities of developers, and the type of knowledge required	Hypothetical dataset based on scenarios	Scenarios show expected results, e.g.developers with lower knowledge level are less productive than with high level	5
[34]	2008	System dynamics simulation model based on reusable patterns	Execution of model with hypothetical scenarios investigating whether model is in-line with well known facts	Evaluation of scenario helping manager in deciding on a combination of verification/ validation techniques	Calibration based on empirical data	Results are consistent with empirical evidence when performing verification and validation activities (e.g. early quality assurance increases software quality); model provides flexibility and reuse of patterns to build models efficiently; model takes different dimensions into consideration (i.e. not over-simplifying)	5

Table 13
Quantifying productivity based on event-based models.

Ref.	Year	Approach	Context	Data source	Outcome	Rigor	
[60]	1993	State-based simulation	Model used what-if scenarios on multiple process runs (50 simulation runs) and evaluates whether an expected outcome is achieved	Evaluation of scenario introducing inspection in between coding and testing; complex standard process model as a baseline	Calibration based on empirical data, choice of inspection due to availability of empirical results	Result shows that number of remaining defects is significantly reduced while development time is increased. Error detection capability increased as well. Overall, the results meet expectations	6
[31]	1999	State-based simulation model	Reports of benefits based on industrial application (qualitative presentation)	Domain: Military; development of airborne radar ground; traditional (waterfall based) development process; 71 distinct development steps within the development process	Data as input for model randomly drawn from past simulation runs	Simulation run identified improvements of 92.000 dollar; claimed improvements at the company: simulation supported reaching higher CMM levels, support drives clear definition of metrics, aids in buy-in for management changes, and increases visibility and understanding of the development process	4
[61]	2000	Discrete Event Model linked to data repository	Reports of benefits based on industrial application (qualitative presentation)	Domain: Military; development of airborne radar ground; traditional (waterfall based) development process; 71 distinct development steps within the development process	Data as input for model randomly drawn from past simulation runs	Linkage to data repository allows new predictions and led to increased accuracy	4
[62]	2002	Discrete event-based simulation using control limits for analysis	Reflection of the researchers on the outcome of the simulation	Domain: Military; development of airborne radar ground; traditional (waterfall based); enhancement project of the size of 32k LOC on existing system	Data as input for model randomly drawn from past simulation runs	Model application shows that critical deviations from target values reflected in the control limits be detected through simulation	4
[35]	2003	Discrete Event Model	Process simulation results compared to actual data of the execution of the process in an experiment	Inspection process with nominal teams following a process with overview meeting, preparation, inspection meeting, correction, and follow-up. The inspection processes simulated were based on ad-hoc reading technique vs. perspective-based reading technique	Model calibration based on another experiment of students inspecting a requirements document (slight context change), in total 169 undergraduate students considered	The defect detection rate of the actual experiment verses the simulation for different types of reviewers (checklist, test scenario, design scenario, user scenario) showed minor deviations in team detection rate of defects	7

example improvement was customer communication. In addition, the linkage of measurements to strategies becomes clearer (see Table 10).

6.1.8. Metric space

Numrich et al. [30] suggested a metric space to measure individual programmer productivity, capturing the accumulative work function of each programmer and the calculated distance between programmers. The performance of different programmers was evaluated and the comparison showed that the approach is able to quantify the contribution of programmers and to capture distances between them. The visualization of the accumulated work effort also showed different approaches of how programmers achieve their tasks (see Table 11 and the complementary material for further details).

6.2. Simulation-based models

6.2.1. Continuous/system dynamic models

Table 12 provides an overview of studies evaluating continuous models to predict software productivity. Romeu [57] built a simulator that pools confounding factors into a single random variable that adjusts values of size and effort by the random variable. Running the simulation several times will produce pairs of predicted size and predicted effort. The calculation of the pairs allowed to compute confidence intervals. The approach was applied on a numerical example and was able to show the probability of how likely it is to achieve a certain productivity. Lin et al. [58] introduced a dynamic simulation model consisting of four components, namely production model, schedule model, staff and effort model, and budged/cost model. Functions determine progress, productivity rate, learning factor, etc. The results showed that the model behaved as expected for different improvement scenarios, the results being in line with literature and were agreed on by practitioners. The comparison with actual values did not show a significant difference. Hanakawa et al. [59] built a dynamic simulation model of learning. The model consists of a knowledge, activity, and productivity model, which affect each other. For instance, the productivity is a result of the distance between required and actual knowledge level. The hypothetical scenarios tested with the productivity model led to expected results (see Table 12). Khosrovian et al. [34] proposed a simulation model which makes use of reusable patterns, allowing the construction of a company specific simulation model. The model was evaluated through different verification and validation scenarios. The result was in-line with empirical evidence (e.g. increased quality with early quality ensurance).

6.2.2. Event-based models

An overview of event-based models for productivity prediction is provided in Tables 13 and 14. Raffo [60] introduces a state-based simulation model, consisting of three components, namely functional perspective, behavioral perspective, and organizational perspective. The model was tested to determine the impact of introducing inspections. The outcome showed that the quality

Table 14
Hybrid simulation.

Ref.	Year	Approach	Context	Data source	Outcome	Rigor	
[33]	1999	Queuing based simulation combined with dynamic systems model	Evaluation of theoretical example (waterfall) by testing the effect of a early, middle, and late defect detection policies	Waterfall model consisting of the steps specification, high level and low level design, implementation, system test, and acceptance test	Input fictitious data	Expected results were obtained for the different policies, i.e. focusing effort on early detection reduces lead-time and effort, and changes effort distribution towards early phases	5
[32]	2001	Queuing based simulation combined with dynamic systems model	Evaluation of theoretical example (waterfall) by testing the effect of a early, middle, and late defect detection policies	Waterfall model consisting of the steps specification, high level and low level design, implementation, system test, and acceptance test	Input fictitious data	Requirements stability yields expected impact considering waterfall environment (substantial increase in effort and delivery time, rework percentage, and defect density with a drop in productivity)	5

was improved and error detection capability increased, which was in line with empirical facts. Raffo et al. [31] applied state-based simulation on the same process as presented in study [61]. The hypothetical improvement achieved by the model was 92.000 dollar of cost savings. Claimed improvements due to the model were (1) support in achieving higher CMM levels; (2) model required clear definition of metrics; (3) support in getting a buy-in from management for improvement actions; (4) improved visibility and understandability of the process. Raffo et al. [61] developed a discrete event simulation model. The model consists of the development activities preliminary design, detailed design, and code and unit testing. The model was linked to a corporate database continuously updating the parameters of the model. The researchers reported that the linkage of the data to the repository allowed continuous predictions based on real-time data improving predictive accuracy, though the increase in accuracy was not quantified in the study. Raffo et al. [62] used discrete event simulation using stochastic data for each simulation run, i.e. the data input for the calibration of the model was drawn from probabilities. For the evaluation of the outcome of the model the researchers proposed to define control limits based on management targets. The application of the approach in industry showed that the model allowed identifying deviations from the target values. Regarding the accuracy of the model the researchers mentioned that the model was accurate in predicting past performance, but no numbers are presented. Münch and Armbrust [35] built a simulation model for a software inspection process based on discrete event simulation. The accuracy of the model was evaluated by comparing the results of the simulation run with results of an actual experiment where students used different reading techniques (ad-hoc, perspective-based reading) on a requirements document. The result showed that there were only minor difference in the team detection rate between simulation and actuals.

6.2.3. Hybrid simulation

Donzelli and Iazeolla [33] introduced a simulation model with two abstraction levels. On the first abstraction level the overall development process flow is modeled as a queuing network. On the second abstraction level each activity (server) was designed as a dynamic model. The dynamic model calculated random development size, effort, release time, and injected defects based on a given distribution (in this case a Gaussian-like probability distribution). The performance attributes that can be predicted by the simulation are effort, lead-time, and rework. In the first publication [33] the model was tested for three different strategies for defect detection, i.e. focus on early, middle, and late defect detection. As can be seen in the table providing an overview of the non-deterministic models the outcome showed expected results. The second publication [32] is based on the same model and simulation runs, but in addition evaluates the impact of instable requirements on the performance attributes of the process. The result here was also as expected, given that a waterfall model was implemented in the simulation. For an overview of hybrid simulation studies, see Table 14.

7. Discussion

7.1. Practical implications (reactive)

In the studies focusing in simple input/output ratios a number of problems were raised with regard to the different measurement approaches. The problems were reported for lines of code productivity and function point productivity.

- *Lines of Code Productivity:* As shown in [18] lines of code productivity consists of different components for output (new lines of code, modified lines of code, etc.) and effort (maintenance effort, new development effort, and so forth). Not making explicit which components are considered leads incomparability of measures when benchmarking. Hence, [18] makes an important contribution by making the decomposition of the measures of lines of code and effort explicit. Another problem related to lines of code productivity is the variance of the measure between programming languages, as well as within the same programming language (cf. [23]). The variance within the same language could be due to that a programmer is capable to develop the same functions with fewer lines of code than another programmer, or due to different coding styles. This variance further complicates the use of lines of code productivity in benchmarking.
- *Function Point Productivity:* Function point productivity is subjected to potential bias when measuring the output by counting the function points. Reported problems were that not all development effort is counted as function points (zero function points), the calibration of the model heavily relies on the skills of the counter and is influenced by subjective judgment [23,20]. A consequence was high fluctuation due to measurement error [20]. Kitchenham et al. [21] also reported high fluctuation and observed that the data is often not normally distributed which further increases the variance. However, in the case reported in [21] the main problem discussed was not the measurement error, but the loss of information when aggregating productivity calculations from several projects.
- *General:* A general problem of both function point and lines of code productivity is that measuring two unrelated variables (e.g. size and effort) can be misleading [23,21]. As we saw in previous studies on predictive models not all studies identified size as a predictor of productivity (cf. [46,17]). Furthermore, all approaches are subjected to the risk of lack of rigor when collecting productivity data in an industrial context [20].

In order to address the above mentioned problems a number of improvement proposals have been made, namely change points

[23], process productivity [63], use case points, and a number of suggestions of how to avoid productivity measures to be misleading [21]. Change points as a measure of output [23] removes technology dependency. Furthermore, change points reduce bias due to objective counting rules not relying on subjective ratings. However, to judge to what degree change points really solve the problems inherent in function points and lines of code productivity further empirical studies are needed. Process productivity was proposed as an improvement of lines of code productivity by incorporating the variables skill-factor and development time in the equation. However, as shown in [17] the addition of variables to the equation did not increase the ability of the productivity measure in capturing productivity change due to related productivity factors such as type of system, storage constraints, time constraints, and so forth [17]. Kitchenham et al. [21] stressed that it is important to not only look at the calculated end result of output and input. Instead, the data should better be illustrated in form of a scatter plot to not be misleading. The scatter plot allows seeing and comparing similar projects. When aggregating productivity data the collected measures should be log transformed to make them more normal and to control the large degree of variability.

The simple ratio measures are univariate. As mentioned earlier software development is characterized as multivariate with multiple inputs and outputs. A solution to capture multiple outputs through multiple size measures was proposed by Kitchenham and Mendes [37]. They showed how to construct a ratio based productivity model taking multiple outputs into consideration and weight them. The data set was based on web projects, which means that outputs such as number of web pages and number of images were considered. However, there is no reason to assume that the model cannot be used for different types of software systems. The study investigated a high number of projects and provides a promising way of considering multiple outputs in ratio based productivity analysis. However, the accuracy of the model was only evaluated based on one practitioner's perception, which means that further studies are needed to further validate the approach.

Data envelopment analysis is solely used as a reactive method to software productivity and is able to consider multiple inputs and multiple outputs. That is, the strength of data envelopment analysis is the ability to handle multivariate analysis. In the case of univariate analysis a production frontier can be identified by observing a scatter plot, also making sure that only comparable projects are benchmarked [24]. However, in the multivariate case a visual inspection is not possible. Hence, the method provides a solution for the multivariate case as no visual examination is possible [24,49]. Furthermore, the applications have shown that the method is able to identify a production frontier as well as reference projects [24,52,25,50,49]. Three issues have been raised. First, data envelopment analysis is non-parametric. A consequence of this is that it is unknown whether the deviation from the production frontier is due to measurement errors or inefficiencies [48,24]. As pointed out in [24] it is important to be aware of this, but at the same time no solution is available as we do not know how much of the deviation from the production frontier can be attributed to error and how much to inefficiencies. Another issue is the consideration of size. As shown in [24] the production frontier changes when clustering projects based on size (e.g. large, medium, small). Consequently, it might be useful to cluster projects and identify the frontier for each cluster. However, the studies have also shown that DEA is able to identify reference projects and the relevance of the projects measured by the peer value [24], which shows the projects most relevant for comparison for the overall dataset. Another issue is the sensitivity of the method to outliers. For the studies included in this review the method was deemed robust as the removal of frontier projects did not change the remaining projects on the frontier [24] and the mean efficiency scores stayed stable [53]. Two studies compared data envelopment analysis with other approaches, namely regression analysis [21] and ratio-based measurement [50]. With regard to regression data envelopment analysis was considered superior as regression does not provide peer values and was not able to identify all reference projects [24]. Data envelopment analysis was also considered superior to simple ratio-based analysis, data envelopment analysis being more fine-grained and sensitive in identifying relevant reference projects. Ratio based analysis also suffers from the problem that large projects will be extreme performers and appear to be either efficient or inefficient. This problem appears if data is analyzed directly, and not with the help of scatter plots. Furthermore, when trying to capture a multivariate case in ratio analysis multiple ratios have to be calculated for each output providing multiple indicators making it difficult to judge which projects are the most efficient ones [50,24,49]. Ratio based analysis has the underlying assumption of constant returns to scale. In the analyzed data sets variable returns to scale were used as well [53,24,50], in particular the variable returns to scale version of data envelopment analysis was more sensitive in capturing frontier projects [24]. Similar efficiency scores obtained by variable returns to scale and constant returns to scale point to similarities. In practice, a large deviation between the two was identified [50]. Given that the variable return to scale models performed better in identifying a production frontier than the constant return to scale models, we could assume that data envelopment analysis also performs better than ratio-based analysis. However, it is not clear whether the underlying data set can be characterized as having variable or constant returns of scale. One explanation for observing variable returns of scales could be that the outer limit of the data is distributed differently. Overall, there are many methodological issues hindering us in order to decide whether there exist economies of scale in software development (see, e.g., [51]), such as selection of hypotheses tests, partitioning of data sets, or the consideration/treatment of underlying assumptions made by productivity measurement approaches. Hence, no definitive answer can be given here.

Based on the observations we can provide the following recommendations for practitioners:

- In the univariate case it is important to be aware of high variances and difficulties when comparing productivities. Hence, it is important to carefully document the context (e.g. coding guidelines, programming language, etc.) to be able to compare between projects. Furthermore, the comparison should not be done solely on the productivity value, but it is recommended to draw a scatter diagram based on inputs and outputs to assure comparability of projects with respect to size. In other words, the calculated end-result of ratios hides information.
- When comparing it should be made clear what output and input contains, as provided in the example of lines of code productivity. It is important to be explicit about which lines of codes are included (e.g. new, modified, blanks, etc.). Otherwise, it is not clear whether a comparison between two projects would be fair.
- When possible use multivariate analysis given the data is available as throughout the software process many outputs are produced. Otherwise, the productivity is biased towards one activity (e.g. number of lines of code produced). For the analysis of the multivariate case data envelopment analysis can be recommended based on the evidence, specifically with respect to the quality of the studies considering rigor. The main concern regarding data envelopment analysis was its sensitivity to outliers and the violation of assumptions made by DEA. However, in the evaluations the method turned out to be robust with regard to the production frontier. Also, practitioners agreed

Table 15
Reactive approaches summary.

Approach	Advantages	Drawbacks	Evaluation summary
Simple ratio	• Provides insight in comparability between projects for the univariate case when plotted in form of scatter plots [21] • High acceptance in industry and hence easy to get buy-in for implementation in practice [47] • Was identified as a driver for concrete process improvements (e.g. increased stability of requirements specifications, increased traceability) [18]	• Potentially measuring two unrelated variables (size and effort) (see diagreement of [17] vs. [43,44]) • Misleading in benchmarking and identifying role models when analyzed directly [21] • Subjectivity of counting output (function points as a frequently used approach) and input (what effort to count) [23,20] • Productivity data often not normally distributed [21], with high variance [21,20] • Some tasks requiring effort are not reflected in measure (e.g. the case in function points) [20] • Ratio based measurement does not account for probabilities of error in measurement [24]	• Different domains and contexts in each paper on ratio-based measurement: real-time systems [18], military/space/industrial [17], application management/information systems [21,20] and banking domain [47] • Different outputs considered in the studies (Two studies on function point productivity [20,21], two on lines of code productivity [18,17], one on change point productivity [23], and one on use case point productivity [47]) • Most studies older than 10 years, see Table 5 • Mixed quality in studies ranging from a score of three to a score of seven (see Table 17) • All studies have been conducted in an industrial context (evaluation research), see Table 18 • Most studies in this category used proof of concept as evaluation criterion (see Table 18)
DEA	• Multivariate analysis (multiple outputs, multiple inputs) is supported, which has been demonstrated in all DEA studies included • Supports the identification of role models based on production functions (as illustrated in all approaches), where the correction of the identification was confirmed by practitioners in one study [24] • Robustness to outliers (production frontier does not change when conducting sensitivity analysis) [24,48,53]	• Software engineering data sets are likely to violate assumptions of DEA (no account for measurement error, all relevant input variables probably not captured, data points often do not represent independent projects with autonomy regarding inputs) [51] • DEA generally known to be sensitive to outliers [25], which is in conflict with observation made in the studies (see advantages)	• Primary information systems investigated (financial transaction processing [48], ERP [24], banking information system [49,50,52]) • Uncertainty of whether to use the VRS or CRS version of DEA, also primarily used non-parametric DEA (only [48] used stochastic approach) • High numbers for rigor (five out of seven studies have a score of seven or higher), see Table 17 • Four studies use proof of concept as evaluation criterion [49,50,25,52], three studies use sensitivity analysis [48,24,53], one practitioner feedback [24] • New and old data sets used in the studies (three after 2000, and four before 2000), see Table 6 • Majority of studies (five) are based on industrial data, see Table 18

with the result produced by DEA with respect to the identification of the right role models. At the same time practitioners should keep in mind that when using DEA with data from software industry, assumptions made by DEA are likely to be violated (see Table 15).

- Generally, it is important that managers are aware of validity threats in the measures when conducting a comparison, such as subjective counting rules, different contexts, different goals with respect to market strategy, and so forth. In additions practitioners should take the message with them that when interpreting data they should do so with care and an awareness of the potential biases and noises in the data due to measurement error.
- With regard to evidence, it is apparent that different forms of productivity models using different variables have been studied (see Table 15). Hence, general recommendations with strong foundations on empirical evidence cannot be provided. This would only be possible when similar approaches with similar variables are evaluated in different contexts. Overall, it should also be noted that the studies did not come to conclusions that contradict each other. Only one case is contradictory, namely that DEA is sensitive to outliers. However, in the studies included sensitivity analysis showed robustness of the DEA approach (see also Table 15). However, in order to generalize this conclusion, more studies of DEA would be needed.

The studies on approaches only appearing once for reactive measurement (cf. [27,56,30]) all have low scores for rigor and hence should be considered as good ideas of how to measure productivity, but need more substantial evaluation before they can be recommended.

For an overall summary of advantages, drawbacks, and a summary of the evaluation of the approaches see Table 15.

7.2. Practical implications (predictive)

Studies on weighting productivity factors focused on information systems with the purpose of supporting management (cf. [46,45,43]). The studies approached the problem of identifying productivity in a similar manner by evaluating the explanatory power of the model. It is shown that all models provide a high explanatory power for the variance of productivity in their environments, which is true for the studies by Foulds et al. [46], Morasca and Russo [45], Maxwell et al. [17] and Jeffery [43] (see Table 4). The studies were based on the ideas in regression analysis. The studies identified factors explaining the variability of the productivity in the context studied, which is an indicator for their predictive ability. The following variables were identified:

- Project factors (management and leadership, user participation, quality assurance, resource availability, test standards) and product factors (documentation availability, batch vs. real-time systems) have significant impact (measured on Likert scale) [46].
- Function points, experience, availability of development time are significant predictors [45].
- Lines of code and maximum staff as dominating factors, and average team attitude to the project and average procedural language experience with minor impact on the productivity [43].
- Company, development language, type of system (category), storage constraints, time constraints, tools, and reliability requirements [17].
- System size and team size [44].

It is visible that the more classical studies from the 1980s and early 1990s [46,43] both share a measure of size and experience in their model, while the new study from 2007 [46] adds other factors that were discussed more recently for software development, such as user participation being considered important in an agile development context [64], which is a development paradigm that emerged in the early 2000s. The model of Maxwell found that experience and size did not explain much of the variability in productivity for a large set of projects [17], but they provided several factors that were relevant when studying projects across domains and types of systems. Hence, the results show that in order to provide good knowledge about predictors for productivity further studies are needed. In particular, new factors have to be considered with the change in developing software, and old factors need to be re-evaluated in the new context. In conclusion no recommendation can be given of which predictor variables to include when predicting productivity based on a regression model in today's software development contexts. As shown in Maxwell the productivity seems to vary with the context with regard to domains (space, military, industrial applications), types of applications, and the use of programming languages and tools. Therefore, aiming for a generic productivity model seems challenging. In order to determine whether a regression model is sufficient as a prediction method future studies should not only focus on identifying the model with the highest explanatory power, but also apply it for predictive purposes. That is, the model should be used to predict future projects and the goodness of fit between the prediction and the actual observations. One way of doing this with existing data is to randomly select a sub-set of projects (e.g. 70% of all data sets), build a prediction model, and then apply it on the remaining 30%. The study of Pfleeger [15] adjusted average productivity by a composition of cost factors. The strength of the study was its comparison of the predictive accuracy with other models. The findings indicate that the method is potentially accurate and achieved good predictive accuracy in the studied context (see Table 4), and at the same time the study was published quite recently. Due to the low number of participants making the prediction based on the model conclusion validity is threatened, as recognized by the authors. Consequently, further validation is needed to make conclusive recommendations.

The prediction of a time-series based on an auto-correlation assumption was able to detect sharp shifts in productivity changes quickly with similar error margins for two different scenarios [55]. A modification of the approach in [54] led a an increase in predictive accuracy [54]. The approach illustrated in [36] showed that time-series analysis/control charts are useful in supporting the decision of whether to recalibrate prediction models. The evaluation showed that the proposed approach worked well in detecting shifts in process performance. In particular the dynamic calibration approach together with control charts provided more accurate predictions than two other approaches, namely pure dynamic calibration and prediction by analogy. However, the studies of [55,54] relied on a single programmer and only presented few scenarios which limits the generalization of the approach. Overall, process control charts as a basis for prediction show promising results, but further evaluations of the approaches are required.

The continuous simulation models were structured in different ways with regard to which components and variables were considered in the models. The solution presented in [57] was very limited with regard to variables as only size and effort were simulated. The models yielded expected results that concur with facts that have been found in the literature [58,59,34]. A limitation is that the studies [57,59,34] did not use the simulation to predict based on an actual industrial case. Only one study [58] compared the simulated results with actual data, showing a very promising outcome as there was very little difference between actuals and the simulation. Hence, this study is an indication for the predictable capability of the results, but further studies are needed to strengthen the evidence.

The event-based models using queuing/discrete event or state-based simulation [31,61,62] all applied the simulation models in an industrial context and provided positive results, such as the support of simulation helps to achieve higher CMM levels [31], aids in getting a buy-in for change initiatives [31], aid in understandability [31], and accurate prediction of past performance [62]. The study presented in [62] reported that predictive accuracy could be improved when continuously updating the simulation model by linking it to a corporate data repository for collected measures. However, the accuracy reported in the studies was not quantified. Only one study using discrete event simulation compared the simulated results with actual data showing that the simulation was very accurate in its predictions [35]. As with the continuous model the study of [35] further supports the conclusion that simulation is capable of predictive accuracy.

Hybrid simulation [33,32] combined queuing simulation with continuous simulation. The simulation was based on fictional data and predicted improvements based on a scenario of changing verification policies, and requirements stability. The impact of the changes resulted in expected results (e.g. focusing effort on early verification reduces lead-time and effort). Hybrid approaches need further validation based on real industrial data in order to make a recommendation with regard to their ability to accurately predict performance.

With regard to the improvement scenarios the results are limited as similar scenarios were evaluated, namely different verification and validation strategies. One possible reason is the good availability of empirical data for the calibration of the model, and knowledge of the effect of different quality assurance strategies on outcome variables such as time and defects. At the same time the ability of the models for different types of improvement scenarios still needs to be shown as well. As different approaches were presented a comparison between them would be of use to learn which way of modeling productivity is more accurate in a variety of industrial contexts.

Based on the observations we can provide the following recommendations for practitioners:

- No generic prediction model can be recommended as studies do not clearly agree on what are the predictors for software productivity. In fact, the predictors might differ between contexts. Hence, companies need to identify and test predictors relevant to their context.
- For the regression analysis we cannot conclude that regression leads to accurate prediction as it has not actually been used for predictive purposes in the identified studies. Instead, the model fitting the existing data best was discovered. In consequence, to be able to give a recommendation on predictive accuracy of regression for software productivity the model should be built on a sub-set of data points, and then predict the remaining data points. Thereafter, the difference between prediction and actual values should be observed and measured.
- Simulation overall provided promising results and the simulation models provided impressive results when being compared with actual data in two studies (cf. [58,35]). Hence, simulation can be suggested as a promising approach. As different simulation approaches were not directly compared with each other no recommendation can be given of which one to choose.
- Time-series analysis/ statistical process control also shows good results in identifying sharp shifts in process performance, as well as shifts due to changes in the process. However, as pointed out earlier few data points have been evaluated and hence the empirical evidence should be seen as preliminary.
- With regard to the evaluation we can also observe that different approaches have been evaluated in different contexts and studies have different ages, which makes generalizability of results

Table 16
Predictive approaches summary.

Approach	Advantages	Drawbacks	Evaluation summary
Weighting factors	• Considers probabilities (e.g. measurement error or model error likely to occur in software development [51]) • Supports in understanding factors that lead to variance in the outcome variable • Robustness with regard to removal of extreme values [37] • Predictive accuracy found to be high in the studies based on explanatory power (all studies)	• Which factors are identified is highly context dependent (Disagreement on predictors, e.g. between studies [17] and [43,44] disagree on whether size is a predictor) • Limitation to one outcome variable (software engineering is multivariate on inputs and outputs)	• Regression applied to a variety of domains: Web projects [37], information systems [43,15,44–46], and military/space/others [17] • Different forms of regression employed leads to few studies per regression approach • Overall high scores on evidence (all articles have scores greater than five, see Table 17) • All studies have been conducted in an industrial context (evaluation research) • Four of the studies are from before 1995 (relatively old data sets), see Table 4 • No standard set of variables that is considered in every study hinders in arriving at a general understanding of predictors (see Table 4) • Primary criterion for evaluation is limited to explanatory power (see Table 18)
Process control	• High predictive accuracy observed in comparison to other models [55,36,54] • Approaches presented in [55,36] are accurate at detecting sharp shifts in process performance	• Specific cases in practice (processes with very high variance that are out of control) might be hard to predict with the given approaches (no evidence) • Software process control charts make assumptions about the distribution of the data (underlying normal distribution) to determine control limits	• Individual computer science student projects [55,54] and one study in the banking domain [36] • Three different approaches (dynamic calibration, Yule-Walker, optimization), see Table 9 • Overall median scores (5–6), see Table 17 • Comparison with actual performance [55,36,54] and sensitivity analysis [55] • One old study (1991) and two relatively new ones (2005 and 2006), see Table 18 • Two lab studies [55,54] and one industrial [36]
Simulation	• Simulation predicts with high accuracy when compared to actuals [58,35] • Simulation produces results that are in-line with empirical evidence (i.e. they predict expected results) [59,34,60,33,32] • Driver for definition and clear definition of metrics [31] • Tool for getting buy-in for changes from management [31] • Supports in understanding/reflecting on the development process [31]	• High model building costs and little understanding of how much a good model requires (depending on abstraction/simulation goals) • Requires profound knowledge in building models in tools (e.g. few simulation patterns for software simulation available to support practitioners)	• Industrial studies restricted to military [31,61,62] and space [58]. Scenarios of lab applications on verification and validation techniques [60,35,33,32,34] • Several studies on each type of simulation model (see Tables 12–14), but models different as built for individual companies/simulation purposes • Majority of studies received medium scores on rigor (eight studies have a score of four or five), Three studies have a score of six or greater (see Table 17) • Comparison with facts is the primary way of evaluating lab studies, Study [58] is a role model in the sense that it uses several criteria in one study • Both newer and older studies identified, six studies before 2000, five studies in or after 2000 • Four industrial and seven lab studies (see Table 18)

challenging. Also, there were limitations with regard to evaluation criteria (i.e. most of the time one criterion was used per study). In addition, for simulation very similar scenarios have been evaluated (focus on verification and validation activities), see Table 16. Overall, for practitioners that means that we have indicators for strenghts and weaknesses for the different approaches, but no strong evidence. As said earlier, for that similar approaches have to be evaluated in a variety of contexts, so that more data points (studies) in each category are available.

Bayesian Belief Networks [26] have also been identified as a promising approach. However, due to that only one study is available further investigation is needed.

For an overall summary of advantages, drawbacks, and a summary of the evaluation of the approaches see Table 16.

7.3. Future directions impacting productivity measurements

In recent years two paradigms have been emerged which will likely influence productivity measurement and prediction, namely agile software development and value-based software engineering.

In agile software development the underlying assumption is that the planning horizon is very limited due to frequent changes in the requirements [64]. To deal with the frequent changes different practices have been introduced, such as time-boxing [65,66]. When using time-boxing the time is fixed (e.g. project is not allowed to last longer than 30 days) and the scope (sets of requirements to be developed) has to be selected in a way that it can be realized within the time box and the given resources. This is a change from prediction done in traditional software development where the scope is determined and based on the scope the productivity is predicted. In the time-box situation the scope has to be predicted to fit into the time-box run by a project with limited resources (e.g. agile and lean recommend small team sizes with a maximum number of team members). As a result this situation has a fixed resource pool and fixed time-box with a variable scope to be selected. Furthermore, in agile development detailed predictions should have a shorter planning horizon (e.g. planning one or two sprints ahead). In consequence, the prediction of productivity changes in comparison to traditional approaches where the overall scope related to a product is estimated early on in the project (predictions done for next couple of sprints with variable scope and fixed time). However, prediction might become more complex from a coordination perspective where many projects running in a time-box have to be coordinated, which adds managerial effort [67]. We cannot say whether productivity measurement and prediction will be harder or easier in the agile context overall. What is important for future work is that the change of the ways of working in agile projects needs to be considered when developing prediction and measurement approaches. For example, relationships between variables might be different compared to more classical software development approaches, such as plan-driven development.

With the emergence of value-based software engineering [68] the content of an output would not be treated equally, i.e. one output (such as function points) would contain parts of the output being of higher value, and parts of the output being of lower value. Both, value-based approaches and productivity assessment are always important to be considered in software organizations. With regard to productivity and value generation, there might be a potential benefit in combining them instead of seeing them as two separate entities, as we might discover that much of the output is not valuable (e.g. if features are developed, but never used). This would provide insights whether the effort invested provides a valuable outcome. Consequently, the target would not be to produce as much as possible with as little effort as possible, but produce as much value as possible with as little effort as possible.

Eighteen productivity studies included in this review have been published before 2000, and hence do not take recent developments in software engineering research and practice into consideration. For example, for the weighting of productivity factors for prediction four out of seven studies were conducted in the 1980s or 1990s. In consequence, a re-evaluation of factors in the current software engineering context is important. It is also of interest to continuously evaluate new factors that lead to variances in software productivity.

Another direction is the combination of reactive and predictive approaches. One possible example is the combination of simulation and data envelopment analysis. We believe that this is a useful combination as the simulations were of multivariate character with multiple inputs and outputs, which is well supported by data envelopment analysis. For example, the simulation could determine to what degree projects move closer to the production frontier for a variety of improvement scenarios. As observed earlier, in this context it is important to test the accuracy of the prediction to a broader set of improvement scenarios as the previous focus was mainly on quality assurance strategies.

7.4. Improvements to studies in productivity research

Fig. 6 shows the frequency of studies having fulfilled the different quality criteria for evaluating the rigor of studies. The figure shows two major areas for potential improvement, namely validity and control group. In the area of validity it is important to capture limitations of the results, so that it is easy to judge the strength of the empirical evidence. An overview of validity threats for different types of studies can be found in [41,69,70] for case study research and [39] for experiments in a software engineering context.

An additional improvement needed is to conduct more comparative studies. This is important to be able to provide recommendations of which approach to select. As was apparent in this review several approaches were able to achieve reasonable results, but there is little indication of which approach performs better than another with few exceptions (cf. [15,17,48,24,50,54]). Furthermore, comparison of the same approaches in similar contexts would support quantitative meta-analysis [71] when aggregating evidence.

Observing Fig. 5 and Table 18 it becomes apparent that only few studies apply different evaluation criteria to judge the accuracy and usefulness of the approach. We would like to point out that multiple criteria would strengthen the results of the studies, e.g. comparison with actual data would capture how precise the method predicts, and practitioner feedback would provide feedback on the practical usage and the consequences taken based on the description.

Another important improvement in reporting productivity studies is to become more consistent in the context descriptions. The studies included in this review all described the context, but have put different emphasis. In response to that we proposed context elements to be covered and explained why they are important for studies on software productivity (see Section 4).

The extraction of relevant information was particularly difficult due to the way many of the studies were reported. That is, information about context, analysis, validity, and so forth was scattered around the papers and hence was troublesome to identify. Therefore, as an improvement to the reporting it is essential to provide the information relevant for systematic reviews in designated sections, thereby making the data extraction for researchers much more efficient. All information with regard to the research methodology (context, data collection methods, analysis methods, validity threats) should be easily identifiable in individual sections. For different research methods reporting guidelines have been provided, for case studies these can be found in [70] and for experiments in [39]. More general information of what is important to report (e.g. context) can be found in [12] and [9].

With regard to the identification of studies in the search and inclusion/exclusion phase structured abstracts would help in identifying studies of relevance as they assure that all important information is available in the abstract (such as background, research method, data sources, and conclusions). Evidence for the usefulness of structured abstracts has been reported in [72].

To understand the return of investment with regard to introducing metric programs focusing on software productivity it is important to report the effort/cost needed in order to implement the metrics in industrial practice. That way, the up-front effort needed to achieve the benefits reported in studies is made explicit. The effort and cost is an important decision criterion for companies when implementing metrics.

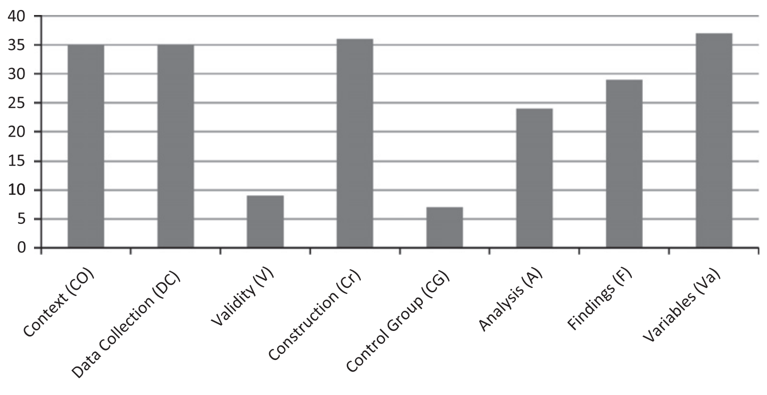

Fig. 6. Overview of quality rating.

8. Conclusion

This systematic review is a synthesis of studies focusing on software productivity measurement and prediction. In total 38 articles were identified with 22 of them being related to prediction and 16 to reactive measurement. In addition to the detailed review a systematic map was created to link the measurement and prediction approaches to evidence, research methods, and abstraction level of measurements. Two main questions were answered in the systematic review part.

What evidence is there for the accuracy/usefulness of the prediction and measurement methods? With regard to reactive measurement the review suggests that simple ratio measures are misleading, and hence should be evaluated with care. One way of avoiding loss of important information is to draw the data in form of a scatter plot capturing the values of input and output for different projects. As software development is multivariate in nature data envelopment analysis was proposed and in the evaluation has turned out to be a robust approach for comparing projects with each other. One of the major benefits of data envelopment analysis is that for inefficient projects it is able to identify the most appropriate reference projects to compare with. In a comparison with ratio-based measurement as well as regression-based measurement data envelopment analysis turned out to be more capable in identifying efficient and inefficient projects. Furthermore, managers using productivity measurements should be aware of potential validity threats related to comparability of measures and measurement error. With regard to productivity prediction the findings suggest that no generic prediction model can be recommended as studies disagree on the factors generating variance in productivity measures. Two simulation studies showed very promising results in simulating a real environment as there was no statistical difference between actual and simulated productivity values. Statistical process control and time-series analysis have shown to be promising approaches in detecting and predicting shifts in productivity.

What recommendations can be given to (1) methodologically improve productivity studies, and (2) improve the packaging and presentation of productivity studies? From a methodological perspective we recommend that: (1) Studies should make use of multiple evaluation criteria as a means for triangulation; (2) Validity threats need to be explicitly addressed; (3) Different approaches should be compared with each other to be able to provide recommendations which one to choose. With regard to the packaging and presentation the studies could improve in the following ways: (1) Become more consistent in the way of describing the context and strive for a high coverage of context elements; (2) A description of information relevant for systematic reviews should be stated in designated sections to ease in data extraction; (3) The use of structured abstracts is encouraged, aiding in the identification and selection of studies.

Besides the answers to the research questions future directions for productivity research were discussed. Important ones to name

Table 17
Quality ratings of studies.

Approach	Ref.	Co	DC	V	Cr	CG	A	F	Va	$\sum$ (max. 8)
Weighting	[43]	1	1	0	1	0	1	1	1	6
	[15]	1	1	1	1	1	1	1	1	8
	[44]	1	1	0	1	0	1	1	1	6
	[17]	1	1	0	1	1	1	1	1	7
	[45]	1	1	1	1	0	1	1	1	7
	[37]	1	1	1	1	0	1	1	1	7
	[46]	1	1	1	1	0	1	1	1	7
Simple ratio	[18]	1	1	0	1	0	0	0	1	4
	[23]	1	1	0	1	0	0	1	1	5
	[17]	1	1	0	1	1	1	1	1	7
	[47]	1	1	0	0	0	0	1	0	3
	[20]	1	1	0	1	0	0	1	1	5
	[21]	1	1	0	1	0	1	1	1	6
Data envelopment analysis	[48]	1	1	1	1	1	1	1	1	8
	[49]	1	1	1	1	0	1	1	1	7
	[24]	1	1	1	1	1	1	1	1	8
	[50]	1	1	0	1	1	1	1	1	7
	[25]	1	1	0	1	0	0	1	1	5
	[52]	1	1	0	1	0	0	0	1	4
	[53]	1	1	0	1	0	1	1	1	6
Bayesian network	[53]	1	1	0	1	0	1	1	1	6
Earned value analysis	[27]	1	0	0	1	0	0	0	1	3
Statistical process control	[55]	0	1	0	1	0	1	1	1	5
	[36]	1	1	0	1	0	1	1	1	6
	[54]	0	1	0	1	1	1	0	1	5
Balanced scorecard	[56]	1	1	0	1	0	0	0	1	4
Metric space	[30]	0	1	0	1	0	0	0	1	3
Continuous simulation	[57]	1	1	0	1	0	1	0	1	5
	[58]	1	1	0	1	0	1	1	1	6
	[59]	1	1	0	1	0	0	1	1	5
	[34]	1	1	0	1	0	0	1	1	5
Event-based simulation	[60]	1	1	0	1	0	1	1	1	6
	[31]	1	1	0	1	0	0	0	1	4
	[61]	1	1	0	1	0	0	0	1	4
	[62]	1	1	1	0	0	0	0	1	4
	[35]	1	1	1	1	0	1	1	1	7
Hybrid	[33]	1	0	0	1	0	1	1	1	5
	[32]	1	0	0	1	0	1	1	1	5

Table 18
Classification.

Ref.	Approach	Purpose	Abstraction	Evaluation criteria	Research method
[43]	Weighting	Predictive	Project	Explanatory power	Evaluation (Industrial)
[15]	Weighting	Predictive	Project	Comparison with actuals	Evaluation (Industrial)
[44]	Weighting	Predictive	Project	Explanatory power	Evaluation (Industrial)
[17]	Weighting	Predictive	Project	Explanatory power	Evaluation (Industrial)
[45]	Weighting	Predictive	Project	Explanatory power	Evaluation (Industrial)
[37]	Weighting	Reactive	Project	Practitioner feedback, Sensitivity analysis	Evaluation (Industrial)
[46]	Weighting	Predictive	Project	Explanatory power	Evaluation (Industrial)
[18]	Simple ratio	Reactive	Project	Proof of concept	Evaluation (Industrial)
[23]	Simple ratio	Reactive	Project	Proof of concept	Evaluation (Industrial)
[17]	Simple ratio	Reactive	Project	Explanatory power	Evaluation (Industrial)
[47]	Simple ratio	Reactive	Project	Proof of concept	Evaluation (Industrial)
[20]	Simple ratio	Reactive	Project	Practitioner feedback	Evaluation (Industrial)
[21]	Simple ratio	Reactive	Project	Proof of concept	Evaluation (Industrial)
[48]	Data envelopment analysis	Reactive	Project	Sensitivity analysis	Evaluation (Industrial)
[49]	Data envelopment analysis	Reactive	Project	Proof of concept	Evaluation (Industrial)
[24]	Data envelopment analysis	Reactive	Project	Sensitivity analysis, practitioner feedback	Evaluation (Industrial)
[50]	Data envelopment analysis	Reactive	Project	Proof of concept	Evaluation (Industrial)
[25]	Data envelopment analysis	Reactive	Individual	Proof of concept	Validation (Lab)
[52]	Data envelopment analysis	Reactive	Project	Proof of concept	Evaluation (Industrial)
[53]	Data envelopment analysis	Reactive	Task	Sensitivity analysis	Validation (Lab)
[53]	Bayesian network	Predictive	Project	Comparison with actuals	Evaluation (Industrial)
[27]	Earned value analysis	Reactive	Project	Proof of concept	Evaluation (Industrial)
[55]	Statistical process control	Predictive	Individual	Comparison with actuals, sensitivity analysis	Evaluation (Industrial)
[36]	Statistical process control	Predictive	Project	Comparison with actuals	Evaluation (Industrial)
[54]	Statistical process control	Predictive	Individual	Comparison with actuals	Evaluation (Industrial)
[56]	Balanced scorecard	Reactive	Organization	Practitioner feedback	Evaluation (Industrial)
[30]	Metric space	Reactive	Individual	Proof of concept	Validation (Lab)
[57]	Continuous/system dynamics	Predictive	Project	Proof of concept	Validation (Lab)
[58]	Continuous/system dynamics	Predictive	Process	Comparison with facts, comparison with actuals, practitioner feedback	Evaluation (Industry)
[59]	Continuous/system dynamics	Predictive	Individual	Comparison with facts	Validation (Lab)
[34]	Continuous/system dynamics	Predictive	Process	Comparison with facts	Validation (Lab)
[60]	Event-based	Predictive	Process	Comparison with facts	Validation (Lab)
[31]	Event-based	Predictive	Process	Practitioner feedback	Evaluation (Industry)
[61]	Event-based	Predictive	Process	Practitioner feedback	Evaluation (Industry)
[62]	Event-based	Predictive	Process	Proof of concept	Evaluation (Industry)
[35]	Event-based	Predictive	Process	Comparison with actuals	Validation (Lab)
[33]	Hybrid simulation	Predictive	Process	Comparison with facts	Validation (Lab)
[32]	Hybrid simulation	Predictive	Process	Comparison with facts	Validation (Lab)

are the use of agile methods, and the emergence of value-based software engineering. As many studies have been conducted before the year 2000 productivity factors need to be re-evaluated in currently operating development organizations. We also suggested a way of how to combine predictive and reactive approaches to productivity quantification.

Acknowledgments

I would like to thank Claes Wohlin for helping in testing the inclusion and exclusion criteria, and for reviewing the manuscript. Thanks also go to Eva Norling, Kent Pettersson and the whole team of the BTH library for helping in retrieving articles that otherwise would have been very hard to get.

Appendix A. Overview of quality rating for studies

The quality rating checked whether the following aspects related to the rigor of the study have reported, the answer to the questions being either yes (valued=1) or no (value=0).

- *Context (Co):* Has the context been described? *Minimum criteria:* In evaluation research at least the application domain should be characterized. In simulation studies the application scenario simulated should be described.
- *Data collection (DC):* Were the data collection methods described and are they justified? *Minimum criteria:* The data source is named (e.g. specific project repositories, interview sources) and/or the data collection method has been presented (e.g. industrial case study with interviews with description of interview goal/what questions were asked).
- *Validity (V):* Were validity threats to the study (e.g. a limitation with regard to the reliability of the collected data) discussed? *Minimum criteria:* The articles raises one or more major threats to validity related to the four types o validity (construct validity, conclusion validity, external validity, internal validity).
- *Construction (Cr):* Was there a description of the steps of how the model was constructed/the structure of the model? *Minimum criteria:* The process and reasoning for the construction of the model is defined.
- *Control group (CG):* Was there a control group/comparison with another model? *Minimum criteria:* Two approaches to productivity assessment are used and the text contains a reflection on the comparison between approaches (e.g. discussion).
- *Analysis (A):* Did the authors explain how the analysis/interpretation of the collected data was done? *Minimum criteria:* At least statistical approaches and/or evaluation steps the researcher took should be named and shortly described.
- *Findings (F):* Was there a reflection/discussion of the results obtained by the productivity measure? *Minimum criteria:* A discussion section discusses the benefits and limitations of the approaches.
- *Variables (Va):* Were the variables of the model/measurement defined? *Minimum criteria:* The variables used in the model and their measurement scale is defined.

The rating on each of the included articles is shown in Table 17.

Appendix B. Overview of classification of each study

The classification was done based on the scheme presented in this paper. For each of the included articles the classifications regarding approach, purpose, abstraction, evaluation criteria, and research method are summarized in Table 18. The information from this table was used as input for the mapping of the articles.

References

[1] N.E. Fenton, S.L. Pfleeger, Software Metrics: A Rigorous and Practical Approach, second ed., PWS, London, 1997.
[2] B. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, Tech. Rep. EBSE-2007-01, Software Engineering Group, School of Computer Science and Mathematics, Keele University, July 2007.
[3] T. Dybå, T. Dingsøyr, G.K. Hanssen, Applying systematic reviews to diverse study types: an experience report, in: Proceedings of the First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), 2007, pp. 225–234.
[4] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: Proceedings of the 10th International Conference on Evaluation and Assessment in Software Engineering (EASE 2008), 2008, pp. 1–10.
[5] K. Petersen, Supplementary Material to: A Systematic Review of Performance Measurement of Software Development, 2010 <http://sites.google.com/site/kaipetersen79/SP.pdf>.
[6] B. Kitchenham, What's up with software metrics? – A preliminary mapping study, Journal of Systems and Software 83 (1) (2010) 37–51 (1991) 293–321.
[7] J. Dale, H. van der Zee, Software productivity metrics: who needs them?, Information and Software Technology 34 (11) (1992) 731–738
[8] W. Scacchi, Understanding software productivity: a knowledge-based approach, International Journal of Software Engineering and Knowledge Engineering 1 (3) (1991) 293–321.
[9] K. Petersen, C. Wohlin, Context in industrial software engineering research, in: Proceedings of the 3rd International Symposium on Empirical Software Engineering (ESEM 2009), 2009, pp. 401–404.
[10] M. Ivarsson, T. Gorschek, A method for evaluating rigor and industrial relevance of technology evaluations, Empirical Software Engineering (2011), doi:10.1007/s10664-010-9146-4.
[11] S.L. Lough, A taxonomy of computer attacks with applications to wireless networks, Ph.D. Thesis, Virginia Polytechnic Institute and State University, 2001.
[12] B. Kitchenham, S.L. Pfleeger, L. Pickard, P. Jones, D.C. Hoaglin, K.E. Emam, J. Rosenberg, Preliminary guidelines for empirical research in software engineering, IEEE Transactions on Software Engineering 28 (8) (2002) 721–734.
[13] M. Dixon-Woods, S. Agarwal, D. Jones, B. Young, A. Sutton, Synthesising qualitative and quantitative evidence: a review of possible methods, Journal of Health Services Research and Policy 10 (1) (2005) 45–53.
[14] D. Pfahl, G. Ruhe, Immos: a methodology for integrated measurement, modelling and simulation, Software Process: Improvement and Practice 7 (3–4) (2002) 189–210.
[15] S.L. Pfleeger, Model of software effort and productivity, Information and Software Technology 33 (3) (1991) 224–231.
[16] J. Cohen, Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences, L. Erlbaum Associates, Mahwah, NJ, 2003.
[17] K. Maxwell, L.V. Wassenhove, S. Dutta, Software development productivity of european space, military, and industrial applications, IEEE Transactions on Software Engineering 22 (10) (1996) 706–718.
[18] W.D. Yu, D.P. Smith, S.T. Huang, Software productivity measurements, in: Proceedings of the 15th International Conference on Computer Software and Applications Conference (COMPSAC 1991), 1991, pp. 558–564.
[19] B.W. Boehm, Software Engineering Economics, Prentice-Hall, Englewood Cliffs, NJ, 1981.
[20] H.S. Bok, K. Raman, Software engineering productivity measurement using function points: a case study, Journal of Information Technology 15 (2000) 79–90.
[21] B.A. Kitchenham, D.R. Jeffery, C. Connaughton, Misleading metrics and unsound analyses, IEEE Software 24 (2) (2007) 73–78.
[22] Çigdem Gencel, O. Demirörs, Functional size measurement revisited, ACM Transactions on Software Engineering Methodology 17 (3) (2008).
[23] V. Chatman, Change-points: a proposal for software productivity measurement, Journal of Systems and Software 31 (1) (1995) 71–91.
[24] E. Stensrud, I. Myrtveit, Identifying high performance erp projects, IEEE Transactions on Software Engineering 29 (5) (2003) 398–416.
[25] D. Liping, Y. Qiusong, S. Liang, T. Jie, W. Yongji, Evaluation of the capability of personal software process based on data envelopment analysis, in: Proceedings of the International Software Process Workshop (SPW 2005), Springer-Verlag, Berlin, Germany, 2005.
[26] I. Stamelos, L. Angelis, P. Dimou, E. Sakellaris, On the use of bayesian belief networks for the prediction of software productivity, Information and Software Technology 45 (1) (2003) 51–60.
[27] V. Kadary, On application of earned value index to software productivity metrics in embedded computer systems, in: Proceedings of the Conference on Computer Systems and Software Engineering (CompEuro 92), 1992, pp. 666–670.
[28] R.S. Kaplan, D.P. Norton, The Balanced Scorecard: Translating Strategy into Action, Harvard Business School Press, Boston, Mass., 1996.
[29] M.O. Searcoid, Metric Spaces, Springer, New York, 2006.
[30] L. R.W. Numrich, V.R. Hochstein, Basili, A metric space for productivity measurement in software development, in: Proceedings of the Second International Workshop on Software Engineering for High Performance Computing System Applications (SE-HPCS 2005), ACM, New York, NY, USA, 2005, pp. 13–16.
[31] D. Raffo, J. Vandeville, R.H. Martin, Software process simulation to achieve higher cmm levels, Journal of Systems and Software 46 (2–3) (1999) 163–172.
[32] P. Donzelli, G. Iazeolla, A hybrid software process simulation model, Software Process: Improvement and Practice 6 (2) (2001) 97–109.
[33] P. Donzelli, G. Iazeolla, A software process simulator for software product and process improvement, in: Proceedings of the 1st International Conference on Product Focused Software Process Improvement (PROFES 1999), 1999, pp. 525–538.
[34] K. Khosrovian, D. Pfahl, V. Garousi, Gensim 2.0: a customizable process simulation model for software process evaluation, in: Proceedings of the International Conference on Software Process (ICSP 2008), 2008, pp. 294–306.
[35] J. Münch, O. Armbrust, Using empirical knowledge from replicated experiments for software process simulation: A practical example, in: Proceedings of the International Symposium on Empirical Software Engineering (ISESE 2003), 2003, pp. 18–27.
[36] M.T. Baldassarre, N. Boffoli, D. Caivano, G. Visaggio, Improving dynamic calibration through statistical process control, in: Proceedings of the 21st IEEE International Conference on Software Maintenance (ICSM 2005), 2005, pp. 273–282.
[37] B. Kitchenham, E. Mendes, Software productivity measurement using multiple size measures, IEEE Transactions on Software Engineering 30 (12) (2004) 1023–1035.
[38] R. Wieringa, N.A.M. Maiden, N.R. Mead, C. Rolland, Requirements engineering paper classification and evaluation criteria: a proposal and a discussion, Requirements Engineering 11 (1) (2006) 102–107.
[39] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslen, Experimentation in Software Engineering: An Introduction (International Series in Software Engineering), Springer, 2000.
[40] K.M. Eisenhardt, Building theories from case study research, Academy of Management Review 14 (4) (1989) 532–550.
[41] R.K. Yin, Case Study Research: Design and Methods, third ed., Sage Publications, 2003.
[42] B. Somekh, Action Research: A Methodology for Change and Development, Open University Press, Maidenhead, 2006.
[43] D.R. Jeffery, Software development productivity model for mis environments, Journal of Systems and Software 7 (2) (1987) 115–125.
[44] G.R. Finnie, G.E. Wittig, Effect of system and team size on 4gl software development productivity, South African Computer Journal (11) (1994) 18–25.
[45] S. Morasca, G. Russo, An empirical study of software productivity, in: Proceedings of the 25th International Computer Software and Applications Conference (COMPSAC 2001), 2001, pp. 317–322.
[46] L.R. Foulds, M. Quaddus, M. West, Structural equation modeling of large-scale information system application development productivity: the hong kong experience, in: Proceedings of the 6th IEEE/ACIS International Conference on Computer and Information Science (ACIS-ICIS 2007), 2007, pp. 724–731.
[47] M. Arnold, P. Pedross, Software size measurement and productivity rating in a large-scale software development department, in: Proceedings of the 20th International Conference on Software Engineering (ICSE 1998), IEEE Computer Society, 1998, pp. 490–493.
[48] R.D. Banker, S.M. Datar, C.F. Kemerer, Model to evaluate variables impacting the productivity of software maintenance projects, Management Science 37 (1) (1991) 1–18.
[49] M.A. Mahmood, K.J. Pettingell, A.L. Shaskevich, Measuring productivity of software projects: a data envelopment approach, Decision Sciences 27 (1) (1996) 57–81.
[50] Z.J. Yang, J.C. Paradi, Dea evaluation of a y2k software retrofit program, IEEE Transactions on Engineering Management 51 (3) (2004) 279–287.
[51] B. Kitchenham, The question of scale economies in software – why cannot researchers agree?, Information and Software Technology 44 (1) (2002) 13–24
[52] M. Asmild, J.C. Paradi, A. Kulkarni, Using data envelopment analysis in software development productivity measurement, Software Process Improvement and Practice 11 (6) (2006) 561–572.
[53] L. Ruan, Y. Wang, Q. Wang, M. Li, Y. Yang, L. Xie, D. Liu, H. Zeng, S. Zhang, J. Xiao, L. Zhang, M.W. Nisar, J. Dai, Empirical study on benchmarking software development tasks, in: Proceedings of the International Conference on Software Process (ICSP 2007), 2007, pp. 221–232.
[54] R. Li, W. Yongji, W. Qing, S. Fengdi, Z. Haitao, Z. Shen, Arimammse: an integrated arima-based software productivity prediction method, in: Proceedings of the 30th International Conference on Computer Software and Applications Conference (COMPSAC 2006), vol. 2, 2006, pp. 135–138.
[55] W.S. Humphrey, N.D. Singpurwalla, Predicting (individual) software productivity, IEEE Transactions on Software Engineering 17 (2) (1991) 196–207.

[56] B. List, R.M. Bruckner, J. Kapaun, Holistic software process performance measurement from the stakeholders' perspective, in: Proceedings of the 16th International Workshop on Database and Expert Systems Applications (DEXA 2005), 2005, pp. 941–947.

[57] J.L. Romeu, A simulation approach for the analysis and forecast of software productivity, Computers and Industrial Engineering 9 (2) (1985) 165–174.

[58] C.Y. Lin, T. Abdel-Hamid, J.S. Sherif, Software-engineering process simulation model (seps), Journal of Systems and Software 38 (3) (1997) 263–277.

[59] N. Hanakawa, S. Morisaki, K. Matsumoto, A learning curve based simulation model for software development, in: Proceedings of the 20th International Conference on Software Engineering (ICSE 1998), 1998, pp. 350–359.

[60] D. Raffo, Evaluating the impact of process improvements quantitatively using process modeling, in: Proceedings of the Conference of the Center for Advanced Studies on Collaborative research (CASCON 1993), IBM Press, 1993, pp. 290–313.

[61] D. Raffo, W. Harrison, J. Vandeville, Coordinating models and metrics to manage software projects, Software Process: Improvement and Practice 5 (2–3) (2000) 159–168.

[62] D.M. Raffo, W. Harrison, J. Vandeville, Software process decision support: making process tradeoffs using a hybrid metrics, modeling and utility framework, in: Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE 2002), ACM, New York, NY, USA, 2002, pp. 803–809.

[63] L.H. Putman, W. Myers, Software Metrics: A Practitioner's Guide to Improved Product Development, Chapman and Hall, London, 1993.

[64] K. Beck, Embracing change with extreme programming, IEEE Computer 32 (10) (1999) 70–77.

[65] C. Larman, Agile and Iterative Development: A Manager's Guide, Pearson Education, 2003.

[66] A.S. Koch, Agile software development: evaluating the methods for your organization, Artech House, Boston, 2005.

[67] K. Petersen, C. Wohlin, A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case, Journal of Systems and Software 82 (9) (2009) 1479–1490.

[68] S. Biffl, Value-Based Software Engineering, first ed., Springer, New York, NY, 2005.

[69] C. Robson, Real World Research: A Resource for Social Scientists and Practitioner-Researchers, second ed., Blackwell, Oxford, 2002.

[70] P. Runeson, M. Höst, Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering 14 (2) (2009) 131–164.

[71] F.M. Wolf, Meta-Analysis: Quantitative Methods for Research Synthesis, Sage Publications, 1986.

[72] D. Budgen, B.A. Kitchenham, S.M. Charters, M. Turner, P. Brereton, S.G. Linkman, Presenting software engineering results using structured abstracts: a randomised experiment, Empirical Software Engineering 13 (4) (2008) 435–468.

Kai Petersen is an industrial PostDoc at Ericsson AB and Blekinge Institute of Technology. He received his M.Sc. and Ph.D. in Software Engineering from Blekinge Institute of Technology. Before starting the Ph.D. studies he worked as a research assistant at University of Duisburg Essen, focusing on software product-line engineering and service-oriented architecture. His current research interests are empirical software engineering, software process improvement, lean and agile development, software testing and security, and software measurement.