CLEI ELECTRONIC JOURNAL, VOLUME 16, NUMBER 1, PAPER 8, APRIL 2013

Are We Prepared for Simulation Based Studies in Software Engineering Yet?

Breno Bernard Nicolau de França, Guilherme Horta Travassos
ESE Group, PESC/COPPE, Federal University of Rio de Janeiro,
Rio de Janeiro, Brazil, 68511
{bfranca, ght}@cos.ufrj.br

Abstract

CONTEXT: Simulation-based studies have been used in different research areas in order to conduct computerized experiments with distinct purposes. However, it seems that Software Engineering simulation studies have been performed in a non-systematic way, using *ad-hoc* experimental design and analysis procedures, i.e., without defining a research protocol and missing information when reporting results. OBJECTIVE: To characterize simulation-based studies and identify the common simulation strategies in Software Engineering. METHOD: To undertake a quasi-Systematic Review. Three online digital libraries (Scopus, Ei Compendex, and Web of Science) are used as sources of information. Information extraction from the primary studies should be captured using a predefined form. Plotted charts and tables should be used together with quantitative/qualitative approaches when possible to support data analysis. RESULTS: From 946 papers, 108 have been included, from which it is possible to identify 19 simulation approaches, 17 domains, 28 simulation models characteristics, 22 output analysis instruments, and 9 procedures for the verification and validation of simulation models in the Software Engineering context. Most dominant approach is System Dynamics in the Software Project Management domain. Replication is not a common behaviour. CONCLUSION: The lack of information regarding most of the simulation-based studies and their models restricts replication, making the results usually specifics and generalization hard. Apart from that, it compromises validity confidence. More research and discussions should be made by the community to increase the reliability and use of simulation-based studies in Software Engineering.

Keywords: Computer Simulation, Software Engineering, Systematic Review.

1 Introduction

Computer simulation has been successfully used in several disciplines, such as Engineering, Economics, Biology, and in the Social Sciences, for many distinct aims [1]. The development of models is also diversified as regards the models' purposes and application domains in Software Engineering (SE), including the use of simulation as a mean of experimentation for models and processes. So, simulation models are used as instruments in experimental studies, called ***simulation based studies*** (SBS). According to Banks, '*Simulation is the imitation of the operation of a real-world process or system over time. Simulation involves the generation of an artificial history of the system, and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented*' [2].

With SBS it is possible to reduce the risks, time and costs of experiments, considering simulations usually run in virtual environments. Apart from that, a simulation is claimed to facilitate the replication of experimental studies and to allow the testing of hypothesis before their implementation into *in vivo* or *in vitro* studies, anticipating the effects of such implementation. However, the development of simulation models for *in virtuo* or *in silico* experiments requires knowledge previously acquired through *in vivo* or *in vitro* studies, configuring a paradox between them [3]. As it is expected, in performing experiments with lower risks, costs, and time, increasing one's research productivity and the scale of the studies through simulation, we need first to organize an evidence-based body of knowledge to understand and characterize the phenomenon of interest, develop valid SE models and only then plan and execute such SBS. Therefore, *in virtuo*/*in silico* studies making use of models built without solid knowledge can always be questioned on their validity and on the confidence of their results. The technical literature provides us with several simulation studies and models based on existing simulation approaches. Examples of such models and studies are given below.

Luckham *et al* [4] present an ADL (Architecture Description Language) called Rapide. This ADL allows system architecture simulation for behavioural analysis. For that, Rapide describes an executable software architecture that allows performing simulations in the early stages of the software development process, before deciding on the implementations. The embedded simulation model, also called execution model, is based on a partially ordered event set, generated with their causes and timestamp that describes the dependence between these events. When the simulation starts, this set is generated and observed by a set of processes (control threads representing the architecture components). These processes respond (execute an action) to an event, and also generate new events through a trigger mechanism.

Arief and Speirs [5] present an approach for automated simulation model generation, based on UML models, in order to foresee system performance. The input model consists of class and sequence diagrams. A parser is responsible for the transformation of UML elements into discrete-event simulation elements (JavaSim). The JavaSim constructs involve: processes, queues, input parameters, pseudo-random variables, and events, among others. Each process represents an instance of a model class, which exchanges messages with other processes until the end of the simulation. System performance results allow evaluating the existence of bottlenecks, processing time, and scale conditions.

The model proposed by Abdel-Hamid and Madnick [6] may be, in the context of Software Engineering, the most refereed system dynamic model. It was proposed to simulate software projects in a wide scope, and it consists of seven subsystems: Human Resource Management treats variables as training, organizational and project turnover, experience level, and developer productivity, among others; Software Production represents project effort allocation; Software Development treats productivity as a complex variable that represents the course of the project, and where there is a potential productivity and actual productivity. The last one is directly affected by motivation and communication variables; Quality Assurance and Rework, which represents the defect generation and detection rates, schedule pressure, and the resulting rework; Tests, which represents the cycle of testing activities in the projects, as well as the impact of non-corrected or non-detected defects in the downstream stages; Controlling, which relates to the measurement of performed tasks, productivity, and rework, among others, apart from the tuning of effort allocation and workload; and Planning, which represents schedule stability (how often a schedule is re-made) and activities ending on schedule.

As regards the simulation model development in SE, our previous experience involves mainly two System Dynamics models. The first one is a scenario-based software project management model to support risk assessment [7]. The second one is related to software evolution, in which the goal is to allow the observation of software quality decay over successive maintenance cycles [8]. Both models were developed using the *Illium* tool, based on a proper System Dynamics language [9]. However, our interests are beyond our previous experience, as it includes understanding how the SE community has developed such simulation model. It started when we looked for a simulation-based solution on the analysis of software architecture tradeoffs in the face of distinct design alternatives, i.e., support for decision-making when a software architect has two or more possible design solutions to evaluate, but depending on the alternative chosen it will positively or negatively interfere with some of the quality attributes of the system under development. Thus, we started to capture knowledge on computer simulations in SE, looking for a methodology to support the development of valid simulation models. In doing so, we expected to increase our orientation in the planning and execution of SBS, including the development of valid simulation models, by characterizing the different simulation approaches previously applied in the SE context.

Systematic Literature Reviews (SLR) are claimed to be a building block for Evidence-Based Software Engineering. It has been seen as a ‘means to evaluating and interpreting the research available that is relevant to a particular research question, topic area, or phenomenon of interest’ [10]. For instance, two related secondary studies [11][12] can be identified in SBS regarding Software Process Simulation.

The survey by Ahmed *et al* [11] relates its results to the state of the practice in simulation in 2008. Such results indicate that software process simulation practitioners are, in general, methodical, work with complex problems (resulting in large scale models), and use a systematic process to develop the simulation models. This work also points out that the simulation modelling process and model evaluation are both the main issues in need of more attention from the community.

The systematic review by Zhang *et al* [12] traces the evolution in Software Process Simulation and Modelling research from 1998 to 2008. The authors analyzed some 200 relevant papers in order to answer their research questions. Among their contributions, the authors highlight: '*Categories for classifying software process simulation models; Research improving the efficiency of Software Process Simulation Models is gaining importance; Hybrid process simulation models have attracted interest as a possibility to more realistically capture complex real-world software processes*'.

Both works do not present results that oppose what we have found. Indeed, we have found supplementary information, as our findings go through the same directions pointed out by previous works, apart from including more findings related to other SE domains, rather than just Software Processes, and experimental issues. The main reason for that, although the research questions brought up by these studies and also by our review are distinct, they share a similar population of studies under investigation.

To expand this perspective, we have undertook a *quasi*-Systematic Review [13] aimed at characterizing SBS reported in the software engineering technical literature. Our review results reinforce the aforementioned ones. Apart from that, we have found supplementary information regarding SBS, making the results less specific. It happened due our broader research questions, which allowed the inclusion of many primary studies considered in [11] [12].

The rest of this paper consists of: Section 2 presents the protocol for the *quasi*-Systematic Review; Section 3 presents our findings and the SBS related results; Section 4 discusses threats to validity; and Section 5 presents the conclusions.

2 Research Protocol

The systematic review document template proposed by Biolchini *et al* [14] guided the review process. As our secondary study has the purpose of characterization, there is no baseline allowing comparisons among studies. Thus, we call this review a *quasi*-Systematic Review [13]. The main research question is 'How have the different simulation approaches presented in the technical literature been applied in simulation studies in Software Engineering?' To answer the research question, we structured the search string, following the PICO strategy [15] and established criteria to include and exclude research papers found in this review:

- **P**opulation: Simulation-based studies in Software Engineering;
- **I**ntervention: Computer simulation models used in studies;
- **C**omparison: None;
- **O**utcome: Model purpose, characteristics, Software Engineering domain, and experimental aspects (experimental design) of the studies.

The outcome matches the information to be extracted. However, some titles and abstracts do not use keywords to identify such information, resulting in low coverage over the controls. Therefore, we decided to suppress the outcome from the search string and consider it in the information extraction stage. Based on the PICO structure, the search string is:

```
(("simulation modeling" OR "simulation modelling" OR "in silico" OR "in virtuo" OR "simulation based study" OR
"simulation study" OR "computer simulation" OR "modeling and simulation" OR "modelling and simulation" OR
"simulation and modeling" OR "simulation and modelling" OR "process simulation" OR "discrete-event simulation" OR
"event based simulation" OR "system dynamics" OR sampling OR "monte carlo" OR "stochastic modeling" OR "agent
based simulation" OR "state based simulation") AND ("software engineering" OR "systems engineering" OR
"application engineering" OR "software development" OR "application development" OR "system development"))AND
("simulation model" OR "discrete event model" OR "event based model" OR "system dynamics model" OR "agent model"
OR "state model"))
```

The controls (Table 1) were captured in a previous *ad-hoc* literature review. The basic selection procedure is based on the paper's title and abstract. For that, a set of inclusion and exclusion criteria is established *a priori*. So, only papers available in the Web; written in English; discussing simulation-based studies; belonging to a Software Engineering domain; and those mentioning one or more simulation models, should be included. Papers not meeting all of these criteria should be excluded. Three researchers (Ri) are involved in the selection of potential relevant papers. R1 did the searching, retrieving the papers, saving their references (including abstract) in the reference manager (*JabRef* tool), adding a field to represent the paper status (I - Included, E - Excluded, D – Doubt), and removing possible duplicates. After that, the first selection (based on the inclusion criteria – reading title and abstract) was done and the status for each paper was assigned. Next, R2 received the *Jabref* file (in BibTeX format) with the references and status information, and reviewed the included and excluded papers. In the case of updating any paper status, R2 set up paper status as D2 and tried to solve queries from R1 by setting the former D's as I2 or E2. Lastly, R3 went through the same R2 procedures, though tagging them as I3 or E3. The papers remaining in D after 3 reviews were included for postanalysis in the information extraction stage. All papers included in the selection stage could be later excluded in the extraction stage, where they are integrally read to improve understanding, clearing any doubts and allowing a better decision on their inclusion or exclusion according to the criteria.

Table 1. Control Papers

Reference
Martin, R.; Raffo, D. Application of a hybrid process simulation model to a software development Project. Journal of Systems and Software, Vol. 59, Issue 3, 2001, pp. 237-246
Khosrovian, K.; Pfahl, D.; Garousi, V. GENSIM 2.0: A customizable process simulation model for software process evaluation. LNCS, Vol. 5007, 2008, pp. 294-306
Drappa, A.; Ludewig, J. Simulation in software engineering training. Proc. ICSE 2000, pp. 199-208
Madachy, R. System dynamics modeling of an inspection-based process. Proc. ICSE 1995, pp. 376-386
Al-Emran, A.; Pfahl, D.; Ruhe, G. A method for re-planning of software releases using discrete-event simulation. Software Process Improvement and Practice, Vol. 13, Issue 1, Jan 2008, pp. 19-33
Al-Emran, A.; Pfahl, D.; Ruhe, G. DynaReP: A discrete event simulation model for re-planning of software releases. LNCS, Vol. 4470, 2007, pp. 246-258
Luckham, D. C.; Kenney, J. J.; Augustin, L. M.; Vera, J.; Bryan, D.; Mann, W. Specification and analysis of system architecture using rapide. IEEE TSE, Vol. 21, Issue 4, April 1995, pp. 336-355
Arief, L. B.; Speirs, N. A. A UML tool for an automatic generation of simulation programs. Proc. Second WOSP, 2000, pp. 71-76
Choi, K.; Bae, D.H.; Kim, T. An approach to a hybrid software process simulation using the DEVS formalism. Software Process Improvement and Practice, Vol. 11, Issue 4, July 2006, pp. 373-383

The sources selected for this *quasi*-Systematic Review are online digital libraries available through the CAPES Portal. Furthermore, all of them should allow online searching via a search engine that supports logical expressions, retrieves papers by title, abstract and keywords, and contains papers from several SE domains. Thus, Scopus, EI Compendex and the Web of Science were chosen as they cover most of the main digital libraries for computer simulation and Software Engineering research, and related areas, such as ACM, IEEE, Elsevier, Springer, and WILEY.

After executing the searches, we got 1,492 entries (906 from Scopus, 85 from Web of Science and 501 from EI Compendex) with 546 duplicates (removed), and 946 entries remaining for the selection stage. From these, 150 papers were chosen. In the information extraction stage an extra 28 papers were excluded after detailed reading and 14 papers were not accessible, with 108 papers left for the analysis, including the two secondary studies [11][12] discussed in the Section 1. Information on all the papers was extracted according to the form provided in Table 2 and later organized in the *Jabref* tool.

Table 2. Information Extraction Form

Field *[Extracted Information]*
Paper identification *[Title, authors, source, year, publication type]*
Simulation approach name
Simulation model purpose *[Objective for the model to be developed]*
Study purpose *[Objective for the study to be performed]*
Software Engineering Domain *[Application area]*
Tool support *[Does the model used have any tool support? If so? Which ones?]*
Characteristics related to the model *[Examples: discrete, continuous, deterministic, stochastic]*
Classification or taxonomy for the characteristics *[Does it have any characteristic classification?]*
Simulation model advantages
Simulation model disadvantages
Verification and validation procedure *[V&V techniques used to evaluate the simulation model]*
Analysis procedure *[Output analysis methodology applied to the study]*
Study strategy *[Controlled experiment, case study, among others]*
Paper main results *[Applicability of the approach, accuracy of results]*

The effort required in the selection process was of on 75 hours to read the titles and abstracts and screen them through the selection criteria. Although the number of papers in the selection stage was high, the extraction step required too much effort, considering the full reading of the research papers and some difficulty in identifying information in the reports, such as simulation model characteristics, experimental design, and analysis procedures. This information was sometimes unclear and often scattered in the text.

The quality assessment criteria (Table 3) adopted are related to the information extraction from. Grading for each criterion is set by its relevance in answering the research question. As we are characterizing SBS, experimental features such as study purpose and strategy are of great relevance. Apart from that, as the simulation model is the main instrument in this sort of study, and represents the main threat to study validity, procedures for V&V and analysis are considered equally relevant. The sum of all quality criteria values should add to 10.

Table 3. Quality Criteria

Criteria	Value
[Approach] Does it identify the underlying simulation approach?	*1 pt*
[Model Purpose] Does it explicitly mention the simulation model purpose?	*1 pt*
[Study Purpose] Does it explicitly mention the study purpose?	*1 pt*
[Domain] Is it possible to identify the SE domain in which the study was undertaken?	*1 pt*
[Tool] Does it explicitly mention any tool support?	*0,5 pt*
[Characteristics] Does it mention the characteristics on the simulation model?	*0,5 pt*
[Classification] Does it provide any classification or taxonomy for the characteristics mentioned?	*0,5 pt*
[Advantages] Does it present the advantages of the simulation model?	*0,5 pt*
[Disadvantages] Does it present the disadvantages of the simulation model?	*0,5 pt*
[V&V] Does it provide any verification or validation procedure for the simulation model?	*1 pt*
[Analysis] What are the statistical instruments used in the analysis of the simulation output?	*1 pt*
[Study Strategy + Exp Design] Is it possible to identify the study strategy in which the simulation model was used as an instrument as well as the experimental design?	*1,5 pt (0,5 for strategy + 1pt for exp. design)*

3 A Profile for Simulation Studies in Software Engineering

After information extraction from each of 108[1] research papers (see Appendix A), including title, abstract and body of text, it was possible to identify 88 simulation models, distributed among several Software Engineering research sub-areas, called domains. In brief, we identified 19 simulation approaches, 17 Software Engineering domains, 28 simulation models characteristics, 22 output analysis instruments, 9 procedures for the verification and validation of simulation models, in the context of SE.

As a general characterization, the most dominant simulation approaches in SE are discrete-event and System Dynamics, as seen in Table 4. This statement can be confirmed in [12] for the context of Software Process and Project Simulation. Even in hybrid models, most of the combinations fall in these two approaches. Discrete-event simulation is a mature approach that has for years been shown to succeed in a vast range of research areas. However, System Dynamics seems to have another explanation for its majority over the other ones, specifically in SE: the influence of Abdel-Hamid and Madnick software project integrated model [16] in works, such as [17] and [18].

We also identified different approaches, though most of them relied on discrete or continuous behaviour. For example, Agent-Based Simulation is often claimed to be a distinct abstraction, but software agents are usually marked by continuous behaviours. In fact, these approaches are not separate. Some of them share characteristics. It is also the case of hybrid approaches.

[1] From the 108 research papers found in the review, two of them are secondary studies (one systematic review and one literature survey), and the remaining 106 include primary studies and simulation model proposals.

Table 4. Distribution of Simulation Approaches

Simulation Approach	#Models	#Papers	Simulation Approach	# Models	# Papers
Agent-Based Simulation	2	2	Object-Oriented Simulation	1	1
Conditional Growth	1	1	Proxel-based	1	1
Discrete-Event	**17**	**18**	Qualitative Simulation	2	2
General Continuous-time Simulation	2	2	*quasi*-Continuous Simulation	1	2
General Discrete-time Simulation	1	1	Semi-Quantitative Simulation	1	1
Hybrid Simulation (Continuous + Discrete)	9	11	State-based Simulation	1	1
Hybrid Simulation (PN + DEVS)	2	2	Stochastic Process Algebra	1	1
Knowledge-based	1	1	**System Dynamics**	**37**	**51**
Monte Carlo	1	1	Temporal Parallel Automata	1	1
Not Specified	6	6			

Some simulation approaches are not clearly defined. This issue happened for two reasons: either paper does not explicitly mention the underlying approach or the simulation model specification does not match any known approach. The model presented in [19] seems to be an analytical model, instead of a simulation one, but there are not enough details to confirm it and no simulation approach was mentioned. Another example of a not clearly identified simulation approach is [20], where we were unable to find details on their model and there is no sentence mentioning the simulation approach used to execute simulations. They are grouped into the 'Not Specified' category. Similar behaviour is seen with the 'General Discrete-Time Simulation' and 'General Continuous-Time Simulation' categories, where the model descriptions mentioned the implementations of discrete or continuous time-advancing mechanisms. Unfortunately, we were unable to recognize the specific approach used to build these models. Examples of discrete and continuous-time simulation can be found in [21], where a simulation approach is applied to design and test embedded systems and in [22], where a continuous model for open source development using Test-Driven Development is given, respectively. The occurrences where approaches (Table 4) appeared in one or two papers seem to be investigations on the approach's suitability in simulating Software Engineering systems or processes, as the models were not submitted to any systematic validation procedure.

Simulation models are system or process abstractions specified in a simulation language, representing the concepts involved in the underlying simulation approach. So, when using these models as an instrument for a simulation-based study, simulation tools are needed in order to allow the simulation trials.

For the papers found in this *quasi*-Systematic Review, most of them present an experience using generic-purpose simulation tools, like Vensim, Arena, and iThink. On the other hand, specific simulation tools appear as the authors' implementation of their models; it is the case of SESAM [33], where a tool with an interactive graphical user interface is used to simulate a software project organization for training purposes. Often, specific-purpose tools are used, at most, in two studies. It may occur because the tool only attains the goals of a specific model and of its respective studies. Table 5 presents the distribution of the simulation tools across papers (P) and models (M) captured in this *quasi*-systematic review, as well as their references and the supported simulation approach.

The most frequently used simulation tools, as shown in Table 5 are Vensim, Extend, and iThink. The main reasons for this may relate to their generic purpose and also for simulation approaches supported by these tools, named System Dynamics and Discrete-event simulation.

This work aims at characterizing SBS in many ways. Therefore, a set of 28 characteristics explicitly mentioned within the model descriptions is obtained. Table 6 summarizes the characteristics and their associated description, extracted just as they were presented from the papers found in the review. These characteristics are purposefully presented in a flat format, as they were found in different contexts. As we are unable to infer synonyms or any kind of relationship among them, we decided not to aggregate them at this time.

We saw in several papers that authors do not characterize their models; they just present them or a representative part of them. It is common to only mention the underlying simulation approach (sometimes describing it). For example, in [23] a mention is made of the simulation approach in the quote 'The simulation model was implemented in a modular way using the SD (*System Dynamics*) tool Vensim'. Another example is 'We present a *discrete-time* simulator tailored to software projects which…' [24]. Furthermore, papers also present the model's structural specification and a brief description of its variables, but no specification of its execution mechanism. Among the papers selected, no one gives a classification for simulation models. However, Raffo [25] claims that Software Process Simulation Models rely on three main paradigms: discrete-event, state-based, and System Dynamics, mentioning only some of their features. Also, Zhang *et al* [12] present the simulation 'paradigm' for each paper in their study, but no features are discussed.

Table 5. Simulation Tools - Distribution and References

Simulation Tool	Reference	Simulation Approach	P	M
AnyLogic	www.xjtek.com/anylogic/why_anylogic	Discrete-Event	1	1
ARENA	www.arenasimulation.com	Discrete-Event	1	1
ASADAL/SIM	selab.postech.ac.kr/xe/?mid=selab_link	Discrete-Event	1	1
C-Sim	www.atl.lmco.com/projects/csim	Discrete-time Simulation	1	1
CESIUM	No website found.	Object-oriented Simulation	1	1
DEVSim++	sim.kaist.ac.kr/M5_1.htm	Hybrid Simulation (SD + DEVS) Discrete-Event	3	3
Extend	www.extendsim.com	Hybrid Simulation (SD + DEVS) Discrete-Event System Dynamics	9	9
GENSIM	No website found.	System Dynamics	1	1
iThink	www.iseesystems.com/	System Dynamics	5	4
Matlab	www.mathworks.com/products/matlab/	Discrete-Event	1	1
NetLogo	ccl.northwestern.edu/netlogo	Agent-based Simulation	1	1
PEPA	www.dcs.ed.ac.uk/pepa/tools	Stochastic Process Algebra	1	1
PowerSim	www.powersim.com	System Dynamics	1	1
Professional Dynamo Plus (PD+)	No website found.	System Dynamics	1	1
Prolog	www.swi-prolog.org	Discrete-Event	1	1
QSIM	ii.fmph.uniba.sk/~takac/QMS/qsimHowTo.html	Qualitative Simulation Semi-quantitative Simulation	3	3
ReliaSim	No website found.	Not Specified.	1	1
RiskSim	www.treeplan.com/risksim.htm	Hybrid Simulation (PN + DEVS)	1	1
SEPS	No website found.	System Dynamics	1	1
SES	www.stackpoleengineering.com/software.aspx	Hybrid Simulation (PN + DEVS)	1	1
SESAM	www.iste.uni-stuttgart.de/en/se/forschung/schwerpunkte/sesam	*quasi*-Continuous Simulation	2	1
SIMNET	No website found.	Discrete-Event	1	1
SimSE	www.ics.uci.edu/~emilyo/SimSE	Not Specified	1	1
SLAMSYSTEM	research.microsoft.com/en-us/projects/slam	Discrete-Event	1	1
Statemate Magnum	No website found.	State-based Simulation	1	1
SystemC	www.systemc.org	Not Specified	1	1
Vensim	www.vensim.com	System Dynamics	18	13

The characteristics found in the studies reports (Table 6) were confronted with simulation approaches and SE domains. From this, it is possible to see that the simulation approach selected brings its own characteristics which are transferred to the simulation model.

We expected to find whether specific SE domains characteristics influence the selection of a simulation approach. However, no information regarding such influence could be drawn from the extracted characteristics.

Software Process and Project are the most dominant Software Engineering domains in technical literature [26] (Fig. 1). Simulation models can be software process or product-driven. In our classification, we use ‘Software Process’ for simulation models concerned with applying analysis to the whole software development process structure and performance. The analyses of process bottlenecks, activity dependencies, software development optimization, or offering theory to support decision-making and cross-project issues are of the Software Process Simulation interest [27]. On the other hand, Software Project Simulation is concerned with resource (human and material) management, allocation policies, and scheduling and cost issues, among others. Examples of such studies can be found in the series of studies conducted by Abdel-Hamid and Madnick [16] [28-32].

The domain of Software Architecture and Design aggregates simulation models for design issues, for different classes of systems, such as fault-tolerant, embedded, and real-time systems, under the perspective of quality attributes such as reliability and performance. This domain is marked by mostly simulating the product (design specification) rather than the design process. Almost all other SE domains focus on a process-driven perspective, e.g., Quality Assurance [33] and Requirements Engineering [34] processes. In such cases, simulation models are used to provide an impact understanding of these sub-process variables on the whole development process execution.

Table 6. Descriptions of Model Characteristics.

Characteristic	Description
Analytic	Expresses high-level quantitative relationships between input and output parameters through mathematical equations.
Asynchronous	Lacks a centralized coordination of the simulation model.
Bi-directional Simulation (Forward and Reverse)	Uses both types of models together: Forward - models of the system of interest as it evolves forward through time. Reverse - models of the system as it moves in reverse or backward in time.
Causal Relationships	Establishes cause-effect relationships.
Continuous-time	Time constantly advances in small steps, as a continuously differentiable function.
Deterministic	Given a specific input, the output will be the same for every simulation run.
Discrete-time	Timeless steps interleaved with user-defined simulated durations. Timeless execution does not mean the code in one step takes zero time to execute, but the model time used in discrete simulations is frozen during the step.
Dynamic	Cause-effect relationships constantly interact while the model is being executed.
Envelop functions	Given interval bounds on the values of some landmarks and envelopes on the monotonic functions, its Qualitative Differential Equation defines a constraint-satisfaction problem (CSP).
Extensible	Provides an explicit extension point through input and output ports. If any model has a compatible input or output port, the model can be extended.
Feedback loops	Continuous information flow within a system with self-correction property.
Formal	The structural relations between variables must be explicitly and precisely defined.
Fuzzy variables	Uses fuzzy logic in defining model variables.
Hierarchical	Structured in different levels and blocks.
Interactive	Users are responsible for generating control signals or data out of external entities, change system states interactively to situate special conditions, and locate the cause of unexpected behaviour.
Knowledge Representation	Use of knowledge modelling techniques (for instance, Cognitive Maps)
Nonlinear interactions	Interactions between variables not following a linear function.
PI-Calculus	Uses rigorous semantics as described in pi-calculus formalism.
Process-based	A discrete model described as a workflow, different from event-driven.
Qualitative Abstraction	Transforms a sequence of empirical measurements into a pattern.
Qualitative Differential Equations	Parameters of the differential equations do not need to be specified as real numbers. It is sufficient to know whether they are positive or negative, specifying as monotonically increasing and decreasing. Expresses natural types of the incomplete knowledge from real world.
Queue models	Stages modelled as servers, where each server has its (resource) queue.
Rule-based	The model rule part determines how the state is changed with every time step. Each rule consists of a condition and an action part.
Scenario-based	Enables the user to test the effects with several combinations of events in the process. It allows an user to define several scenarios for the system or process.
State-based	Processes an input event, based on its state and condition, and generates an output event and changes its state (state transition).
State based on events	System state is changed only when certain events occur and it does not change between them.
Stochastic	Instead of assigning deterministic values to model parameters and variables these values can be sampled from plausible input distributions.
Synchronous	Opposite of asynchronous.

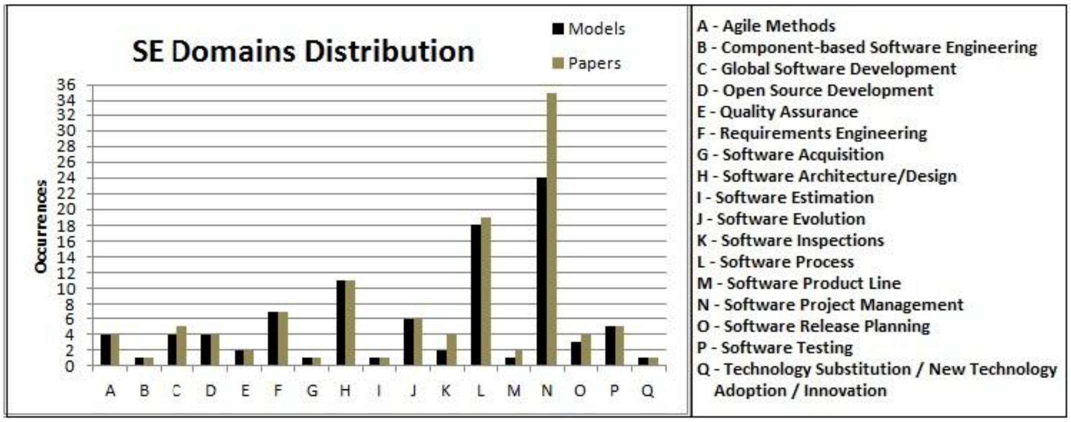

Figure 1. Distribution of Software Engineering Domains

An interesting perspective in this distribution (Fig. 1) is shown in Fig. 2, with the coverage[2] of each simulation approach (Table 4) over different Software Engineering domains.

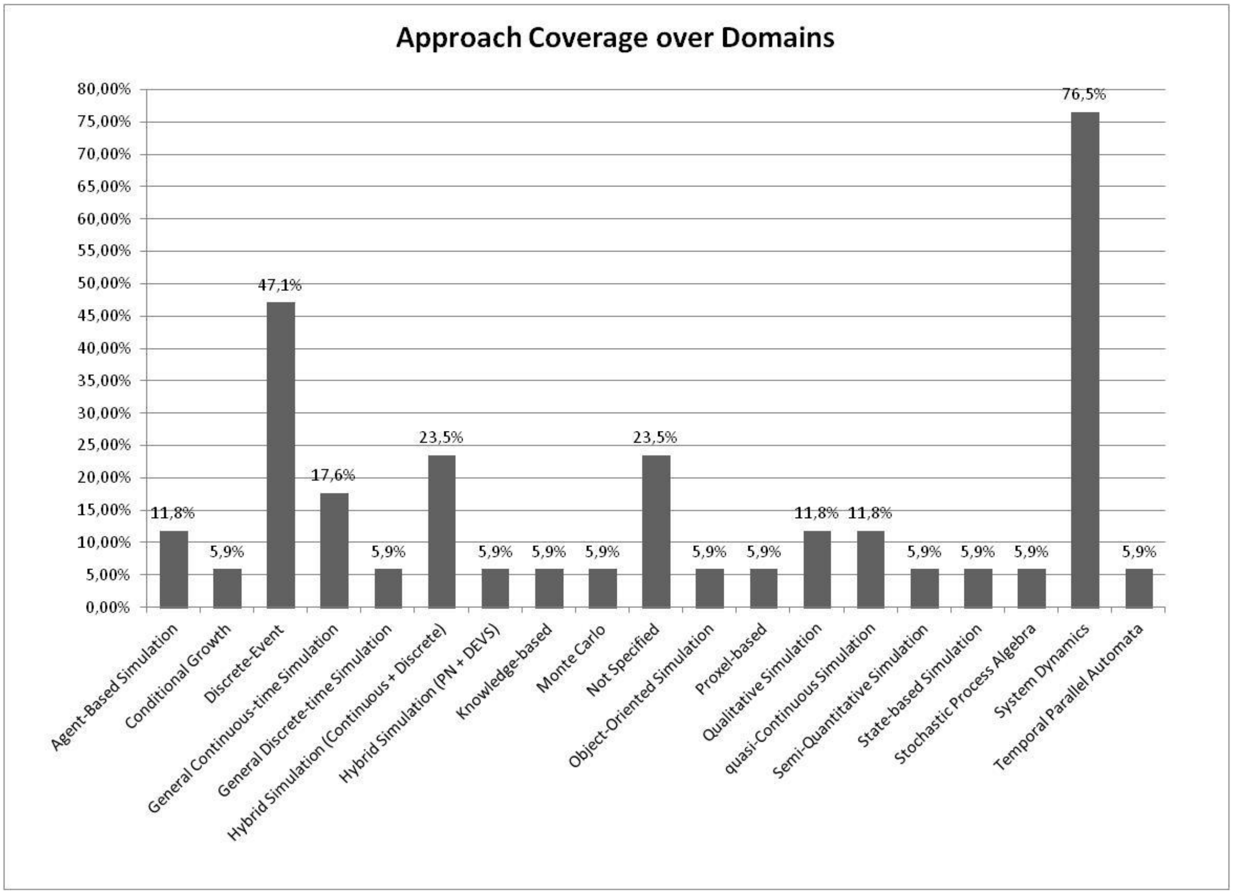

Figure 2. Coverage of Simulation Approaches over Software Engineering Domains

[2] By ‘coverage’ we mean the percentage of elements (SE domains in this case) seen to appear, in a given simulation approach.

The bar chart in Fig. 2 should be read considering each percentage as a number representing the relative occurrence of a specific approach in the space of Software Engineering domains. For example, the Discrete-Event approach appears in 47.1% percent of the Software Engineering domains found in this review, also considering the number of papers and not only models.

Among the different simulation approaches, Discrete-Event and System Dynamics have the most coverage over these domains. Apart from that, Hybrid Simulation, which is mostly a combination of Discrete-Event and System Dynamics, covers 23.5% of the domains. So, the representation of these approaches in Software Engineering, in our sample, can be considered as the main alternative to build a simulation model in this context.

Any simulation model based on observation of a real-world system or process needs to be validated to ensure confidence on its results and compliance with the observed system or process structure/behaviour. The complete list of V&V procedures found and their descriptions is provided in Table 7, along with the number of occurrences in simulation models (M) and research papers (P).

The most dominant V&V procedure is the ‘Comparison against actual results’. It represents an interesting way to verify model ‘accuracy’ in terms of its output results, but many other threats to the study validity (such as *construct* and internal validity) should be evaluated, leading to the combined use of other procedures.

To make such comparisons, such as in the early 90s, when Abdel-Hamid and Madnick applied their model to several different environment configurations, it is important to check whether the actual collected data captures the whole context (influence variables and constants) the model requires or assumes to be real, i.e., model parameters and variables should share the same measurement context. Otherwise, it will become a pointless comparison between two distinct contexts that cannot be compared.

Table 7. Verification and Validation Procedures for Simulation Models

Procedure	Description	M	P
Comparison against actual results	It compares the simulation output results against actual output of the same phenomenon. It is likely to be used to measure model accuracy.	**10**	**17**
Comparison against data from literature	It compares the simulation output results against output data (performance) from other studies found in the technical literature. It is likely to be used when there is no complete data at hand.	3	3
Comparison against reference behaviours from technical literature	It compares the simulation output results against trends or expected results often reported in the technical literature. It is likely to be used when no comparable data is available.	6	12
Comparison against other models’ results	It compares the simulation model output results against one another. Controlled experiments can be used to arrange such comparisons.	7	7
Review with experts	It consists of getting feedback from system or process experts, to evaluate if the simulation results are reasonable. This review may be performed using any method, including inspections. It is likely to be used for model validation purposes.	5	5
Interview with experts	It gets feedback from a system or process expert in interviews, to evaluate if the results are reasonable. It is likely to be used for model validation purposes.	3	9
Survey with experts	It gets feedback from a system or process expert in surveys, to evaluate if the results seem reasonable. It is likely to be used for model validation purposes.	3	3
Testing structure and model behaviour	It submits the simulation model to several tests cases, evaluating its responses and traces. It is likely to be used for model verification purpos6es.	4	5
Based on empirical evidence from technical literature	It collects evidence from the technical literature (experimental studies reports) to develop the simulation model.	7	13

The same problem occurs when the comparison is made between simulation output results against data from the technical literature; the latter is seldom available with enough detail to enable valid comparisons. Thus, it is very difficult to ensure the same configuration (input and calibration parameters) for both simulation results and data collected from the technical literature. Ambrosio *et al* [35] use two procedures for model validation: one is a comparison against actual data from a software company, and the other is a comparison against summarized data from different sources extracted from the technical literature.

Lack of data is a known problem in Software Engineering simulation [36], including support for experimentation [37]. So, different efforts are needed in order to increase the validity of simulation studies. An interesting approach is the comparison of simulation results against known reference behaviours[3] in some research areas. In this case, it is possible to analyze whether the simulation model is capable of reproducing consistent results, even if it is not possible to measure accuracy using this procedure. Setamanit *et al* [38] compare the output results of a global software development simulation model against behaviours of GSD projects as described in the technical literature.

Once we have validated models, or at least know them in terms of performance and accuracy, it is possible to compare them. It would be good to have benchmark results and available datasets to perform controlled experiments aimed at comparing models, establishing distinct treatment and control groups to test hypotheses on how independent and dependent variables influence relationships.

Considering the lack of data and reference behaviours, as needed for the previous V&V procedures, it is still useful to conduct reviews, interviews, and surveys with simulation and system-under-study experts. These procedures help towards a better understanding of the simulation model structure and getting insights on how to improve it. It is more like a validation procedure, as model validation gets the customer involved. Choi *et al* [39] mention a feedback review with experts to check a simulation model based on UML for mission-critical real-time embedded system development. Setamanit and Raffo [40] calibrated their model based on information from survey questionnaires as well as in-depth interviews with the technical director, the project manager, and software developers.

The testing model structure and behaviour is related to the application of several test cases to the simulation model. No paper gives details on how to plan and perform these tests.

One of the most important 'V&V procedures' should be considered before the model to be conceived, i.e., it does not work as a test, but brings confidence to the simulation model. The model building methodology should be based on the empirical evidence found in the technical literature; it should rather be based on controlled experiments that can take conclusions on hypotheses involving the simulation model's variables. Anyway, to use results of well-conducted studies (such as controlled experiments, surveys, and case studies) in the conceiving of simulation models would be better than observing the system or process to be simulated in an *ad-hoc* manner. Melis *et al* [41] presented a series of experiments and case studies results on pair programming and test-driven development. These results were used both to determine variables relationships and equations of a System Dynamics model. All procedures found in papers report simulations studies that seem to be applicable to System Dynamics models, i.e., it possibly means that this approach reached a high maturity level, covering several ways to verify and validate a model built under this approach. Or, maybe it is only a matter of a higher sample of studies using System Dynamics. On the other hand, there are approaches with no attempt at verification or validation. Further, the evaluation settings will be discussed as regards the studies done in the papers analyzed. Fig. 4 shows how simulation approaches cover the procedures mentioned before.

In Fig. 4 it is possible to see the low coverage (0%) of a proxel-based simulation approach, for instance. In other words, it means that this approach was never submitted to any V&V procedure in the context of Software Engineering, according to our findings. On the other hand, System Dynamics seems to be the approach that has been evaluated the most (100%) in SE.

Most of the time, output analyses are based on charts; statistical procedures are rarely used, such as hypothesis testing. Perhaps, the lack of a rigorous and systematic approach for SBS influences it. Sequence Run Charts are used mostly for both discrete and continuous data, as well as for well-represented time-sensitive data. Its flexibility associated to the easy understanding may be an explanation for its adoption. Table 8 summarizes this.

Statistical charts are obviously a relevant way for presenting data, but the significance of these results should never be analyzed by only looking at charts. Other statistical instruments are needed, namely significance tests and systematic analysis procedures (such as sensitivity analysis). These other instruments are still underused and misused. Even descriptive statistics have not received the attention needed, for they are good sample summaries.

[3] Examples of reference behaviours in Software Engineering can be 'Brook's Law' for software projects, Lehman's Laws on software evolution, and any process/product repeatable behaviour in a software organization.

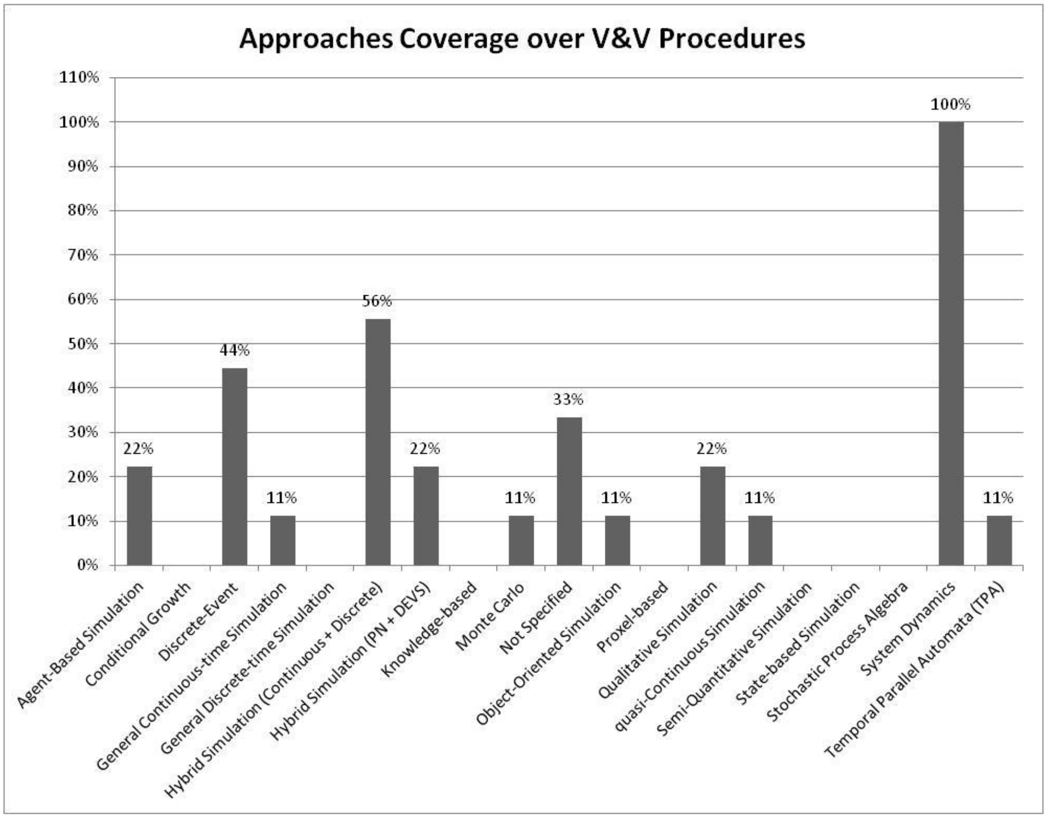

Figure 3. Coverage of Simulation Approaches over V&V Procedures

Table 8. Instruments for Output Analysis

Analysis Instruments	#Models	#Papers	Analysis Instruments	#Models	#Papers
Bar Charts	7	10	Main Effect Chart	1	1
Box Plots	8	9	Outcome Based Control Limits	1	1
Radar Chart	1	1	Pareto Chart	1	2
Distribution Fitting	1	1	Comparison between outputs	3	5
Dot plot	6	6	Probability Distribution Plots	1	1
Gantt Chart	4	5	Scatter plot	1	1
Histograms	4	5	**Sequence Run Charts**	**39**	**48**
Interaction Plots	1	1	Sensitivity Analysis	13	9
ANOVA (and variants)	4	8	t-test	2	5
Confidence Intervals	2	2	Percent relative error	1	1
Descriptive Statistics	6	6	Root Mean Squared Error	1	1

A curious case happens with the Sensitivity Analysis technique: the number of models is greater than the number of papers. It happens as there are some papers promoting the use of this technique to analyze and understand simulation models, identifying the most relevant model factors and input parameters. In these cases, the authors apply sensitivity analysis to more than one simulation model, making some comparisons and presenting different situations that can be found. For example, when models make use of too many input parameters, but just a few really contribute for results. Examples of studies involving sensitivity analysis are [42] using four models and [43] using just one model.

3.1 Study Strategies involving Simulation

Primary studies are presented in 57 of the 108 papers selected. However, there seems to be a misclassification of the study strategies adopted in each study. The remaining reported studies are only examples of model use (assertions or informal studies [44]), with no systematic methodology reported to support their planning, execution, or analysis.

As far as we know, there is no taxonomy or classification schema specific for SBS, and therefore we want to analyze these studies from the perspective of known experimental Software Engineering study strategies. So, we took as a starting point the glossary of terms (http://lens-ese.cos.ufrj.br/wikiese) and the ontology (http://lens.cos.ufrj.br/esee) offered by the eSEE – experimental Software Engineering Environment [45], to classify the observed simulation studies. The research strategies considered are Action Research, Survey, Case Study, Observational Study, and Controlled Study. By definition, Action Research and Observational Study strategies are not applicable to simulation studies, as they require *in loco* observation of the phenomenon under investigation. Thus, we adapted these concepts from the external technical literature to simulation studies, classifying them into: survey (retrospective data; it is important to notice these surveys are not designed as a collection of expert's opinions using forms as instruments, but surveying past historical project data through simulation trials using a simulation model as an instrument), case study (current real data), or controlled experiments.

Several authors call their studies 'experiments'. In fact, these studies are what the technical literature call 'simulation experiments' and they differ from 'controlled experiments'. By 'simulation experiment' we mean a 'test or a series of tests in which meaningful changes are made to the input variables of a simulation model so that we may observe and identify the reasons for changes in the performance measures' [46]. This definition is closer to what we call 'Survey'. In any case, it is difficult to identify hypothesis, experimental design, and also control and treatment groups.

This way, the term 'control' is often overloaded in SBS. The explicit variation of input parameters (used in techniques such as Sensitivity Analysis) does not mean the model's user has control over the object under investigation. In fact, it just establishes the parameter configuration, which is different from controlling factors as in controlled experiments. When the input parameters variation is performed *ad-hoc*, pointed at understanding model behaviour through the correspondent impact on the output variables, it does not mean to have control over the simulation model. In other words, the output values are not meaningful yet for the real context, as it is not possible to assume that valid output values are being generated without previously observing the actual behaviour in *in vivo* or *in vitro* contexts. Unless the variation of parameters changes model behaviour, what has been done is just to determine the value of a dependent variable for the model behaviour function, given an independent variable value. The same can be said in a comparison of two models: one developed and evaluated against actual collected data and an evolution of the former original model without actual data support. If both models are being compared based on the same data set, such shared data set can be a threat to the validity of the study. The reason is that output values from the modified model do not come from the same measurement context of the original model. So, they are not comparable.

In addition, we proposed the SBS characterization based on the number of involved models and datasets; type of involved dataset (real or artificial data); types (constant or variable) of input parameters and how they are determined (systematically or *ad-hoc*); model calibration or not; and the procedure for the simulation study. On top of that, we applied it to the studies found in our review.

As shown in Fig. 4-A, survey studies are proportional (more than half of the studies) to the procedures pointed in Fig. 4-B, where 'Variation of input parameters to observe the impact on output variables' is the most dominating procedure. The same interpretation can be applied to the percentage of controlled experiments and the procedure 'Comparison against other/modified models results'. These comparisons are made in particular experimental designs with treatment and control groups established to compare the models involved in a fair manner. Another feature of these studies regards how the input parameters are determined (Fig 4-C) for the study, which can be by a preset procedure to generate or choose these values, which will be the same for the whole simulation study, or without any established procedure (*ad-hoc*). Apart from that, it can be constant in the entire trial.

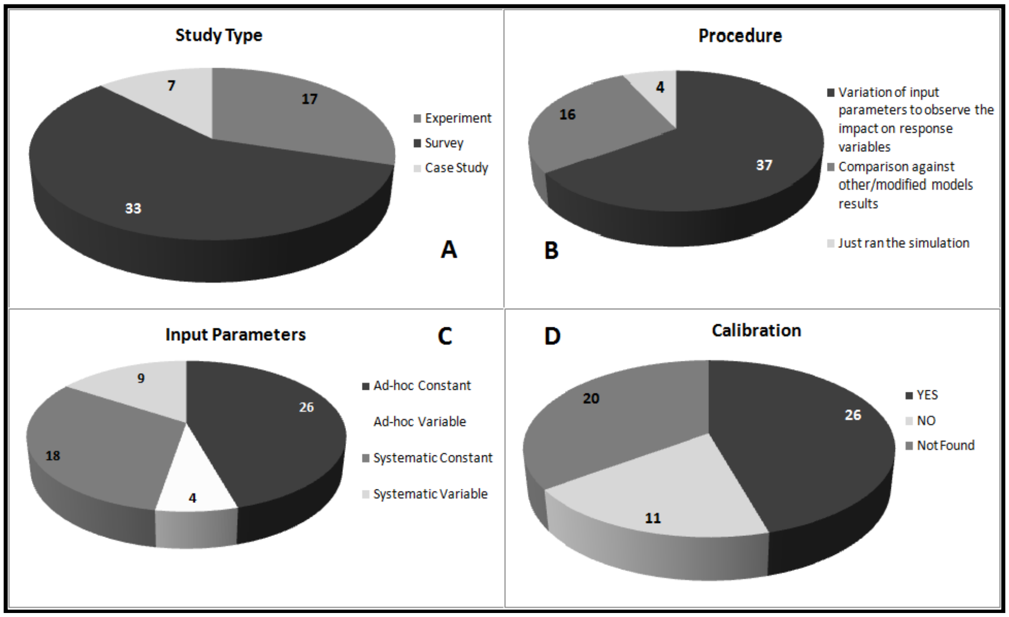

Figure 4. Characterization of simulation-based studies

Model calibration is also important in SBS [47]. In Fig 4-D it is possible to see that many studies do not report this information. Many of the SBS with no calibrated model are based on artificial data. The distribution of simulation studies over simulation approaches is shown in Table 9. Most of the studies use System Dynamics models. We also identified four SBS replications, all of them by the same author. A possible reason for the lack of replication can be insufficient information on the models.

Table 9. Distribution of Simulation Studies over Simulation Approaches

Simulation Approaches	**#Studies**	**Simulation Approaches**	**#Studies**
Agent-Based Simulation	1	Not Specified	5
Discrete-Event	5	Object-Oriented Simulation	1
Hybrid Simulation (Continuous + Discrete)	4	Proxel-Based	1
Hybrid Simulation (PN + DEVS)	1	*quasi*-Continuous Simulation	1
Monte Carlo	1	**System Dynamics**	**36**
General Continuous-time Simulation	1		

Analogous to simulation approaches, we present the studies distributed over Software Engineering domains in Table 10. Software Project Management is the most studied domain in the selected SBS. Apart from that, the combination of System Dynamics models with Software Project Management domain can be found in most of the papers selected in this review. It can indicate that these simulation studies have a higher experience degree and that it is possible to learn SBS features with them.

Table 10. Distribution of Simulation Studies over SE Domains

SE Domain	**#Studies**	**SE Domain**	**#Studies**
Agile Methods	2	Software Inspections	9
Global Software Development	1	Software Process	3
Requirements Engineering	4	**Software Project Management**	**25**
Software Architecture/Design	5	Software Testing	2
Software Evolution	5	Technology Substitution/New technology adoption / Innovation	1

The purpose and goals of simulation models are not clearly defined in the papers analyzed. The same goes for the goals of the studies performed. It is very common to find descriptions mentioning only the problem where the proposed simulation model is involved, but it is very hard to find specific or structured research questions, hypothesis, or even a GQM template, for example, describing the problem or the model. As a first impression, one can believe the models were conceived before the problems, i.e., as we could not find predefined goals and scope, simulation models seem to be developed without a problem to be solved. Actually, this should not be true. However, the way in which SBS have been reported is biasing this conclusion.

Another issue in SBS reports regards the experimental design. In general, it is not reported at all. It is possible, but hard, to identify the independent variables and factors (many times, just parameters) and response variables for the studies where the output data is usually presented in charts. The arrangement is rarely explained, i.e., it is not easy to find answers to questions such as *what are the treatments/levels for each experimental factor? What are the other model input parameters (context variables)? Do they remain constant? What were their initial values for each simulation trial?* Seldom, simple information such as the number of simulation trials is reported. However, the criteria to explain why such number is used are not presented. Without addressing these main issues, the replication and auditing of these studies (or even verifying their results) represent unfeasible tasks, as well as it is hard to compare studies results and to make a benchmark of simulation models, as there is no comparable baseline.

There are some exceptions among the selected SBS. Houston *et al* [42] use DOE (Design of Experiments) to measure the relative contribution of each factor to the variation in response variables, to characterize the behaviour of system dynamic simulation models. Wakeland *et al* [43] propose the use of DOE and BRSA (Broad Range Sensitivity Analysis) to understand the interactions and nonlinear effects at work in the simulation model, i.e., the model logic and behaviour, and in this way, leading to a better understanding of the underlying system/process. However, most of these studies rely on proof of concepts, such as studies, consisting of ***ad hoc*** experimental design, not complying with any reported systematic experimental methodology, and often missing relevant information in their reports.

3.2 Quality Appraisal

Considering the data extracted and analyzed, we assessed the quality of the reports to better understand what kind of information could be missing and look for a possible cause for it [48]. The quality assessment criteria (Table 4) applied to all 108 research papers have direct linkage with the fields of the information extraction form. Such relationship occurs as the main research question for the *quasi*-Systematic Review involves the characterization of SBS.

The mean score is 6.16 (in a [0-10] scale) and standard deviation is 1.41. Apart from that, the minimum obtained score is 3.0 and the maximum score is 9.0. These scores can be construed as lack of important information in the reports. Also, the low standard deviation shows that even the majority of the reports does not offer much more data. The percentage of papers meeting the quality criteria is shown in Fig. 6. Darkest bars indicate information strictly related to SBS issues. For instance, the criteria ‘Does it explicitly mention the study purpose?’ means that in 42.6% of the analyzed papers it was possible to identify the study purpose (goal).

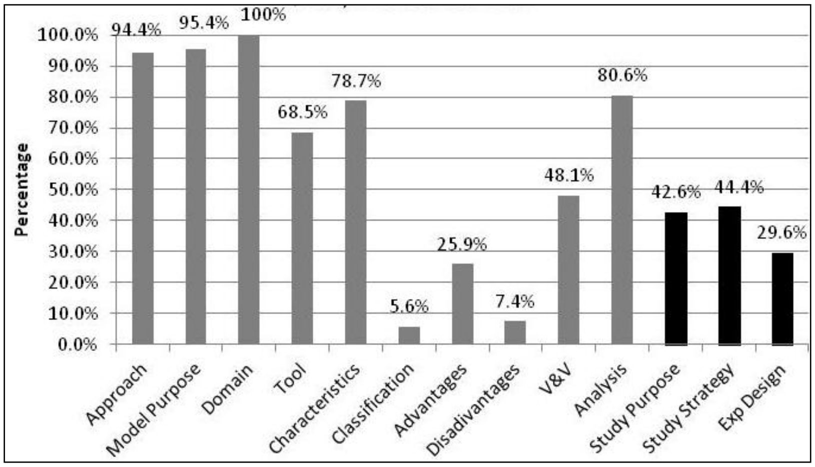

Figure 5. Quality Evaluation Results [48]

All 108 papers provide enough information on the SE domain, where the simulation study is taking place. However, we found just six occurrences (5.6%) of study classification. It does not mean a true classification, but groups of characteristics through which the authors believe it is possible to organize the simulation models. However, these papers do not discuss this issue, they just mention it. Less than half of the papers introduce or discuss simulation model V&V procedures. Information on the studies is also very scarce. Just 48 research papers present a clear goal (what is the prime information to be reported) and only 32 research papers give some information on the experimental design, which often is not fully described. Dependent and independent variables can be found in some of them, mainly perceived in simulation result presentation (charts), without explanation on their arrangement or number of simulation runs. We did not judge the quality or relevance of the definition of goals, experimental design, or adequate data collection procedures. If it is possible to identify the purpose of the study, for instance, but this is not clear, points were assigned to the paper anyway. The same interpretation is also given to experimental design information. In brief, the data typically presented in research protocols is not shown in the SBSs analyzed. Thus, we assume a lack of planning in such studies as it was not reported.

4 Threats to Validity

It is possible to identify threats to this secondary study, as follows.

Keywords. Terminology problems have been reported in many research areas involving Computer Science and Engineering. So, we previously performed an *ad-hoc* review to organize the first set of strings and submitted them to two experts in computer simulation applied to Software Engineering, to minimize the absence of unknown terms. Specific terms for each Software Engineering domain (such as testing, design, requirements, inspections, and others) were not used, but general terms to represent the Software Engineering field were. It may be a threat as it is possible that papers not mentioning the general Software Engineering terms were not included.

Source Selection. Scopus, EI Compendex, and the Web of Science cover the main publication databases for computer simulation and Software Engineering research. A sample of the most important technical literature is enough for characterization purposes, as it is not practical to review all the research papers published on this subject.

Inclusion and Exclusion Criteria. Papers were filtered to exclude those not considering simulation applied to the Software Engineering field. Nine control papers were pre-selected from the *ad-hoc* literature review according to the level of experience of the researchers. The coverage on the control papers was of 67%. From the selected set, 14 papers were unavailable for download.

Personal Understanding and Bias. Although the undertaking of systematic reviews may impose a lot of extra and manual error-prone work [10], three reviewers were involved in the selection and information extraction to reduce bias.

Classification. As long as no consensual taxonomy or classification schema was identified, we based it on the information presented by each paper. It may cause inaccurate classification. We tried to group terms (approaches, characteristics, analysis and V&V procedures) pursuing a semantically similar definition or description.

Conclusions. Another study limitation is the publication selection bias, as publications rarely contain negative results nor present their weakness. Unfortunately, there is no way to deal with this risk.

Considering all these threats and the caution taken to reduce them, we believe that our review was systematically performed and that it brings some confidence to the results.

5 Discussion

In Software Engineering, there is concern with the development of simulation models. However, the studies undertaken using such models include threats to validity that may compromise their results. Such threats are mainly related to simulation model validity and a lack of a predefined experimental design, as well as an informal output analysis, causing untimely conclusions. Reducing such threats will increase the confidence on the study results and the possibility to adopt SBS as a feasibility assessment strategy before implementing complex solutions.

An important issue identified in this review relates to the quality of the reports found in the technical literature. In other words, the lack of relevant information to improve the understanding of the SBS performed in SE. Such an issue can be managed by guidance on what information is essential for a SBS report in the context of SE. Relevant information such as a precise context definition of where the study is taking place, clear goals and research questions, description of behaviours to be simulated, procedures adopted for model assessment, and the experimental design are examples of useful information that should build a study report.

It is also possible to use the findings of this review as a starting point for future research work involving SBS in the SE community. Thus, we believe that proposals on methods and processes to support the planning and execution of SBS, including model validity aspects and statistical output analysis, are a research challenge for the SE community. All the findings on the simulation approaches, V&V procedures, analysis instruments and so on, may be used to construct such proposals and guide decisions.

Methodological issues comprise not only one but many challenges regarding SBS in the SE context, after the characterization provided by this review, such as, for example:

- The definition of research protocols for SBS. For most of the study strategies applied in SE we can use plans containing context description, goals, observed variables, procedures and instrumentation, subjects' definition and selection. However, these aspects are usually overlooked when performing SBS;
- Proposals and application of V&V procedures for simulation models. This challenge is also mentioned by Ahmed *et al* (2008) in a survey done with modelling and simulation practitioners regarding software processes;
- Practices to reduce threats to validity in SBS. This challenge is strongly related to the validity of simulation models;
- Assessment of the quality and strength of the evidence, as captured in the execution of SBS. Once these studies can be verified, it is important to establish a set of criteria that allows judging how close from reality they are and how the threats to study validity affect the quality of the evidence;
- Replication in simulation-based studies, given the issues regarding relevant information on studies reports, such as: full model description, infrastructure environment, where a simulation has been done, the publication of unavailable input and output datasets, and the experimental procedure containing the steps performed.

6 Concluding Remarks and Future Directions

Simulation approaches are often mentioned and described in the Software Engineering field. Apart from the importance on providing an overview of the underlying simulation approaches, some published approaches are not clearly defined. Therefore, it was not possible to capture the reasons for using one simulation approach or the other. We meant to show the link between simulation approaches and Software Engineering domains/characteristics; however, our findings indicate that model characteristics are mostly driven by the simulation approach, not the SE domain. Simulation-based studies entail the main concerns of our results. There is a lack of rigour in planning studies, assuring model validity prior to performing the studies, and analysis procedures for simulation of the output data. All these issues are supported *ad-hoc* for most SBSs, except for a few studies that have a systematic way of doing this or that activity, but never for all of them together.

These results can be used as a starting point for future research directions that must be addressed by the Software Engineering community when conducting simulation-based studies. Apart from that, the information can be organized as a body of knowledge to support decision-making regarding simulation in Software Engineering. Another direction is to organize the specific terminology for SBS in Software Engineering. Ontologies relating to simulation concepts can help future literature reviews and SBS comprehension.

Based on this, we believe the methodologies that entail SBS and that include model validity assurance and output analysis represent research challenges for software engineers. Apart from that, some discussion is needed on the concrete methods for developing simulation models in SE. Some specific approaches were proposed in the technical literature such as, for example, the IMMoS methodology for the development of System Dynamics models for Software Process Simulation [49]. It represents an important step towards strengthening the SBS. However, we need more. Our future in SE also depends on our capacity to speed observations up as well as acquiring knowledge.

Acknowledgements

The authors would like to thank CNPq for their financial support. We should like to acknowledge and thank Dr. Marco Antonio Araújo for his assistance as second researcher, supporting the selection stage of this work's protocol trial.

References

[1] M. Müller, D. Pfahl, "Simulation Methods," *Guide to Advanced Empirical Software Engineering*, section I, pp. 117-152, New York: Springer-Verlag, 2008.

[2] J. Banks, “Introduction to Simulation,” in *Winter Simulation Conference (WSC’99)*, Phoenix, AZ, USA, 1999.

[3] G. H. Travassos, M. O. Barros, “Contributions of In Virtuo and In Silico Experiments for the Future of Empirical Studies in Software Engineering,” in *WSESE03,* Fraunhofer IRB Verlag, Rome, 2003.

[4] D. C. Luckham, J. J. Kenney, L. M. Augustin, J. Vera, D. Bryan, W. Mann, “Specification and analysis of system architecture using rapide,” *IEEE TSE*, vol. 21, iss. 4, april 1995, pp. 336-355.

[5] L. B. Arief, N. A. Speirs, “A UML tool for an automatic generation of simulation programs,” in *Proc. Second WOSP*, 2000, pp. 71-76.

[6] T. Abdel-Hamid, S. Madnick, *Software Project Dynamics: An Integrated Approach*, Prentice Hall, 1 ed. april, 1991.

[7] M. O. Barros, C. M. L. Werner, G. H. Travassos, “Supporting risks in software project management,” *Journal of Systems and Software*, vol. 70, no.1-2, pp. 21-35, 2004.

[8] V. Monteiro, M. A. Araújo, G. H. Travassos, “Towards a Model to Support in silico Studies regarding Software Evolution,” in *ESEM 2012*, sep. 2012.

[9] M. O. Barros, C. M. L. Werner, G. H. Travassos, “A system dynamics metamodel for software process modeling,” *Software Process: Improvement and Practice*, vol. 7, no. 3-4, pp. 161-172, 2002.

[10] T. Dyba, T. Dingsoyr, G. K. Hanssen, “Applying Systematic Reviews to Diverse Study Types: An Experience Report,” in *ESEM 2007*, 2007, pp.225-234.

[11] R. Ahmed, T. Hall, P. Wernick, S. Robinson, M. Shah, “Software process simulation modelling: A survey of practice,” *Journal of Simulation*, vol. 2, pp. 91 – 102, 2008.

[12] H. Zhang, B. Kitchenham, D. Pfahl, “Reflections on 10 years of software process simulation modeling: A systematic review,” *LNCS*, vol. 5007, pp. 345-356, 2008.

[13] G. H. Travassos, P. M. Santos, P. G. Mian, A. C. Dias Neto, J. Biolchini, “An Environment to Support Large Scale Experimentation in Software Engineering,” in *Proc. 13th IEEE ICECCS*, pp. 193-202, 2008.

[14] J. Biolchini, P. G. Mian, A. C. Natali, G. H. Travassos, “Systematic Review in Software Engineering: Relevance and Utility,” PESC-COPPE/UFRJ, Brazil. Tech. Rep. http://www.cos.ufrj.br/uploadfiles/es67905.pdf (2005.

[15] M. Pai, M. McCulloch, J. D. Gorman, “Systematic Reviews and meta-analyses: An illustrated, step-by-step guide,” *The National Medical Journal of India*, vol. 17, n.2, 2004.

[16] T. Abdel-Hamid, “Understanding the "90% syndrome" in software project management: A simulation-based case study,” *Journal of Systems and Software*, vol. 8, pp. 319-33, 1988.

[17] R. Martin, D. Raffo, “Application of a hybrid process simulation model to a software development project,” *Journal of Systems and Software*, vol. 59, pp. 237– 46, 2001.

[18] B. Lee, J. Miller, “Multi-project management in Software Engineering using simulation modeling,” *Software Quality Journal*, vol. 12, pp. 59-82, Kluwer Academic Publ, 2004.

[19] S. W. Ormon, C. R. Cassady, A. G. Greenwood, “A simulation-based reliability prediction model for conceptual design,” in *Proceedings of the Annual Reliability and Maintainability Symposium*, Philadelphia, PA, United states, 2001, pp. 433 – 436.

[20] E. O. Navarro, A. V. D. Hoek, “Design and evaluation of an educational software process simulation environment and associated model,” in *Proc. of 18th Conference on Software Engineering Education and Training*, CSEE and T 2005, Ottawa, ON, Canada, 2005, pp. 25 – 34.

[21] S. Grillinger, P., Brada, P., Racek, “Simulation approach to embedded system programming and testing,” in *Proc. of 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems*, ECBS 2004, Brno, 2004, pp. 248–254.

[22] I. Turnu, M. Melis, A. Cau, A. Setzu, G. Concas, K. Mannaro, “Modeling and simulation of open source

development using an agile practice," *Journal of Systems Architecture*, vol. 52, no. 11, pp. 610 – 618, 2006.

[23] D. Pfahl, K. Lebsanft, "Using simulation to analyze the impact of software requirement volatility on project performance," *Information and Software Technology*, vol. 42, pp. 1001–1008, 2000.

[24] F. Padberg, "A software process scheduling simulator," in *Proc. ICSE*, pp. 816-817, 2003.

[25] D. M. Raffo, "Software project management using PROMPT: A hybrid metrics, modeling and utility framework," *Information and Software Technology*, vol.47, pp.1009–1017, 2005.

[26] B. Stopford, S. Counsell, "A Framework for the Simulation of Structural Software Evolution," *ACM Transactions on Modeling and Computer Simulation*, vol. 18, 2008.

[27] Y. X. Chen, Q. Liu, "Hierarchy-based team software process simulation model," *Wuhan University Journal of Natural Sciences*, vol. 11, pp. 273– 77, 2006.

[28] T. Abdel-Hamid, "The economics of software quality assurance: A simulation-based case study," *MIS Quarterly: Management Information Systems*, vol. 12, pp. 395-410, 1988.

[29] T. Abdel-Hamid, "Dynamics of software project staffing: A system dynamics based simulation approach," *IEEE Trans. on Software Engineering*, vol. 15, pp. 109–119, 1989.

[30] T. Abdel-Hamid, "Investigating the cost/schedule trade-off in software development," *IEEE Software*, vol. 7, pp. 97-105, 1990.

[31] T. Abdel-Hamid, "A multiproject perspective of single-project dynamics," *Journal of Systems and Software*, vol. 22, pp. 151-165, 1993.

[32] T. Abdel-Hamid, S. Madnick, "Impact of Schedule Estimation on Software Project Behavior," *IEEE Software*, vol. 3, pp. 70–75, 1986.

[33] A. Drappa, J. Ludewig, "Quantitative modeling for the interactive simulation of software projects," *Journal of Systems and Software*, vol. 46, pp. 113 – 122, 1999.

[34] S. Ferreira, J. Collofello, D. Shunk, G. Mackulak, "Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation," *Journal of Systems and Software*, vol. 82, pp. 1568–1577. 2009.

[35] B. G. Ambrosio, J. L. Braga, and M. A. Resende-Filho, "Modeling and scenario simulation for decision support in management of requirements activities in software projects," Journal of Software Maintenance and Evolution, vol. 23, no. 1, pp. 35 – 50, 2011.

[36] D. Raffo, T. Kaltio, D. Partridge, K. Phalp, J. F. Ramil, "Empirical Studies Applied to Software Process Models," *Empirical Software Engineering*, vol. 4, iss. 4, pp. 353-369, 1999.

[37] R. E. Garcia, M. C. F. Oliveira, J. C. Maldonado, "Genetic Algorithms to Support Software Engineering Experimentation," in *IV International Symposium on Empirical Software Engineering*, 2005, v. 1. p. 488-497.

[38] S. Setamanit, W. Wakeland, D. Raffo, "Using simulation to evaluate global software development task allocation strategies," *Software Process Improvement and Practice*, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom, vol. 12, pp. 491 – 503, 2007.

[39] K. Choi, D. Bae, T. Kim, "An approach to a hybrid software process simulation using the DEVS formalism," *Software Process Improvement and Practice*, vol. 11, pp. 373 – 383, 2006.

[40] S. Setamanit, D. Raffo, "Identifying key success factors for globally distributed software development project using simulation: A case study," *Lecture Notes in Computer Science*, Leipzig, Germany, vol. 5007 LNCS, pp. 320 – 332, 2008.

[41] M. Melis, I. Turnu, A. Cau, G. Concas, "Evaluating the impact of test-first programming and pair programming through software process simulation," *Software Process Improvement and Practice*, vol. 11, pp. 345 – 360, 2006.

[42] D. X. Houston, S. Ferreira, J. S. Collofello, D. C. Montgomery, G. T. Mackulak, D. L. Shunk, "Behavioral

characterization: Finding and using the influential factors in software process simulation models," *Journal of Systems and Software*, vol. 59, pp. 259-270, 2001.

[43] W. W. Wakeland, R. H. Martin, D. Raffo, "Using Design of Experiments, sensitivity analysis, and hybrid simulation to evaluate changes to a software development process: A case study," *Software Process Improvement and Practice*, vol. 9, pp. 107–119, 2004.

[44] M. V. Zelkowitz, "Techniques for Empirical Validation. Empirical Software Engineering Issues, *LNCS*, vol. 4336, Springer-Verlag Berlin Heidelberg, pp. 4 – 9. 2007.

[45] V. P. Lopes, "Repositório de Conhecimento de um Ambiente de Apoio a Experimentação em Engenharia de Software," *Dissertação de Mestrado*. PESC-COPPE/UFRJ. 2010. (in Portuguese)

[46] A. Maria,"Introduction to Modeling and Simulation" in *Proc. of 1997 WSC*, 1997.

[47] T. I. Ören, "Concepts and Criteria to Assess Acceptability of Simulation Studies: A Frame of Reference," *Simulation Modeling and Statistical Computing*, Communications of the ACM, vol. 24, n. 4. April 1981.

[48] B. B. N. França, G. H. Travassos, "Reporting Guidelines for Simulation-Based Studies in Software Engineering," in *16th International Conference on Evaluation & Assessment in Software Engineering – EASE 2012*, Ciudad Real, Spain, May, 2012, pp. 156-160.

[49] D. Pfahl, G. Ruhe, "IMMoS: a methodology for integrated measurement, modelling and simulation," *Software Process Improvement and Practice*, vol. 7, pp. 189-210, Wiley, 2002.

[50] D. Pfahl, M. Klemm, G. Ruhe, "A CBT module with integrated simulation component for software project management education and training," *Journal of Systems and Software*, vol. 59, no. 3, pp. 283 – 298, 2001.

[51] V. Garousi, K. Khosrovian, D. Pfahl, "A customizable pattern-based software process simulation model: Design, calibration and application," *Software Process Improvement and Practice*, vol. 14, no. 3, pp. 165 – 180, 2009.

[52] K. G. Kouskouras, A. C. Georgiou, "A discrete event simulation model in the case of managing a software project," *European Journal of Operational Research*, vol. 181, no. 1, pp. 374 – 389, 2007.

[53] M. Ruiz, M. Ramos, I. Toro, "A Dynamic Integrated Framework for Software Process Improvement," *Software Quality Journal*, vol. 10, no. 2, pp. 181–194, 2002.

[54] D. Williams, T. Hall, M. Kennedy, "A framework for improving the requirements engineering process management," *Software Quality Journal*, vol. 8, no. 2, pp. 133–147, 1999.

[55] M. Höst, B. Regnell, C. Tingström, "A framework for simulation of requirements engineering processes," in *EUROMICRO 2008 – Proc. of the 34th EUROMICRO Conference on Software Engineering and Advanced Applications*, SEAA 2008, Parma, 2008, pp. 183–190.

[56] N. Hanakawa, K. Matsumoto, K. Torii, "A knowledge-based software process simulation model," *Annals of Software Engineering*, vol. 14, no. 1–4, pp. 383–406, Dec. 2002.

[57] C. Chen, C. Wang, Y. Lee, Y. Lee, "A process-oriented system dynamics model for software development project prediction," in *Proc. of 4th International Conference on Networked Computing and Advanced Information Management*, NCM 2008, Gyeongju, Korea, Republic of, 2008, vol. 2, pp. 126 – 131.

[58] T. L. Landers, H. A. Taha, C. L. King, "A reliability simulation approach for use in the design process," *IEEE Transactions on Reliability*, vol. 40, no. 2, pp. 177 – 181, 1991.

[59] Z. Wang, W. Haberl, A. Herkersdorf, M. Wechs, "A simulation approach for performance validation during embedded systems design," *Communications in Computer and Information Science*, Porto Sani, Greece, vol. 17 CCIS, pp. 385 – 399, 2008.

[60] S. Cook, R. Harrison, P. Wernick, "A simulation model of self-organising evolvability in software systems," in *Proc. of the 2005 IEEE International Workshop on Software Evolvability*, Budapest, Hungary, 2005, pp. 17 – 22.

[61] S. Balsamo, M. Marzolla, “A Simulation-Based Approach to Software Performance Modeling,” in *Proc. of the Joint European Software Engineering Conference (ESEC) and SIGSOFT Symposium on the Foundations of Software Engineering* (FSE-11), Helsinki, Iceland, 2003, pp. 363 – 366.

[62] N. Smith, A. Capiluppi, J. F. Ramil, “A study of open source software evolution data using qualitative simulation,” *Software Process Improvement and Practice*, vol. 10, no. 3, pp. 287–300, 2005.

[63] D. Pfahl, A. Al-Emran, G. Ruhe, “A system dynamics simulation model for analyzing the stability of software release plans,” *Software Process Improvement and Practice*, Southern Gate, Chichester, West Sussex, PO19 8SQ, United Kingdom, vol. 12, pp. 475 – 490, 2007.

[64] S. Kuppuswami, K. Vivekanandan, P. Rodrigues, “A system dynamics simulation model to find the effects of XP on cost of change curve,” *Lecture Notes in Computer Science*, vol. 2675, pp. 54–62, 2003.

[65] D. Pfahl, N. Koval, G. Ruhe, “An experiment for evaluating the effectiveness of using a system dynamics simulation model in software project management education,” in *International Software Metrics Symposium*, Proceedings, London, United kingdom, 2001, pp. 97 – 109.

[66] T. Abdel-Hamid, F. Leidy, “An Expert Simulator for Allocating the Quality Assurance Effort in Software Development,” *Simulation*, vol. 56, no. 4, pp. 233–240, Apr. 1991.

[67] D. Pfahl, O. Laitenberger, J. Dorsch, G. Ruhe, “An Externally Replicated Experiment for Evaluating the Learning Effectiveness of Using Simulations in Software Project Management Education,” Empirical Software Engineering, vol. 8, no. 4, pp. 367 – 395, 2003.

[68] L. Williams, A. Shukla, A. I. Anton, “An initial exploration of the relationship between pair programming and Brooks’ law,” in *Proc. of the Agile Development Conference*, ADC 2004, Salt Lake City, UT, United states, 2004, pp. 11 – 20.

[69] S. Ur, E. Yom-Tov, P. Wernick, “An open source simulation model of software development and testing,” in *Lecture Notes in Computer Science*, Haifa, Israel, 2007, vol. 4383 LNCS, pp. 124 – 137.

[70] T. Thelin, H. Petersson, P. Runeson, C. Wohlin, “Applying sampling to improve software inspections,” Journal of Systems and Software, vol. 73, no. 2, pp. 257 – 269, 2004.

[71] F. Alvarez, Guillermo A., Cristian, “Applying simulation to the design and performance evaluation of fault-tolerant systems,” in *Proc. of the IEEE Symposium on Reliable Distributed Systems*, Durham, NC, USA, 1997, pp. 35–42.

[72] K. Kang, K. Lee, J. Lee, G. Kim, “ASADAL/SIM: An incremental multi-level simulation and analysis tool for real-time software specifications,” *Software Practice & Experience*, vol. 28, no. 4, pp. 445–462, Apr. 1998.

[73] R. Madachy, B. Boehm, J. A. Lane, “Assessing hybrid incremental processes for SISOS development,” *Software Process Improvement and Practice*, Southern Gate, Chichester, West Sussex, United Kingdom, 2007, vol. 12, pp. 461 – 473.

[74] T. Häberlein, “Common structures in system dynamics models of software acquisition projects,” *Software Process Improvement and Practice*, vol. 9, no. 2, pp. 67–80, 2004.

[75] S. Park, K. Choi, K. Yoon, D. Bae, “Deriving software process simulation model from spem-based software process model,” in *Proc. of Asia-Pacific Software Engineering Conference*, APSEC, Nagoya, Japan, 2007, pp. 382 – 389.

[76] P. Sooraj, P. K. J. Mohapatra, “Developing an inter-site coordination index for global software development,” in *Proc. of 3rd IEEE International Conference Global Software Engineering*, ICGSE 2008, Bangalore, India, 2008, pp. 119 – 128.

[77] C. Lauer, R. German, J. Pollmer, “Discrete event simulation and analysis of timing problems in automotive embedded systems,” in *IEEE International Systems Conference Proceedings*, SysCon 2010, San Diego, CA, United states, 2010, pp. 18 – 22.

[78] R. Madachy, B. Khoshnevis, "Dynamic simulation modeling of an inspection-based software lifecycle processes," *Simulation*, vol. 69, no. 1, pp. 35 – 47, 1997.

[79] H. Rahmandad, D. M. Weiss, "Dynamics of concurrent software development," *System Dynamics Review*, vol. 25, no. 3, pp. 224–249, Sep. 2009.

[80] A. Al-Emran, D. Pfahl, and G. Ruhe, "DynaReP: A discrete event simulation model for re-planning of software releases," *Lecture Notes in Computer Science*, Minneapolis, MN, United states, 2007, vol. 4470, pp. 246 – 258.

[81] D. Rodríguez, M. Á. Sicilia, J. J. Cuadrado-Gallego, D. Pfahl, "e-learning in project management using simulation models: A case study based on the replication of an experiment," *IEEE Transactions on Education*, vol. 49, no. 4, pp. 451–463, 2006.

[82] D. Pfahl, O. Laitenberger, G. Ruhe, J. Dorsch, T. Krivobokova, "Evaluating the learning effectiveness of using simulations in software project management education: Results from a twice replicated experiment," *Information and Software Technology*, vol. 46, no. 2, pp. 127 – 147, 2004.

[83] S. Setamanit, W. Wakeland, D. Raffo, "Exploring the impact of task allocation strategies for global software development using simulation," *Lecture Notes in Computer Science*, Shanghai, China, 2006, vol. 3966 LNCS, pp. 274 – 285.

[84] R. Lazarova-Molnar, S., Mizouni, "Floating task: Introducing and simulating a higher degree of uncertainty in project schedules," in *Proc. of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, WET ICE, Larissa, 2010, pp. 136–141.

[85] K. Khosrovian, D. Pfahl, V. Garousi, "GENSIM 2.0: A customizable process simulation model for software process evaluation," in *Lecture Notes in Computer*, Leipzig, Germany, vol. 5007, pp. 294 – 306, 2008.

[86] H. Zhang, R. Jeffery, L. Zhu, "Hybrid modeling of test-and-fix processes in incremental development," *Lecture Notes in Computer Science*, Leipzig, Germany, vol. 5007, pp. 333 – 344, 2008.

[87] D. Merrill, J. S. Collofello, "Improving software project management skills using a software project simulator," in *Proc. of Frontiers in Education Conference*, Pittsburgh, PA, USA, 1997, vol. 3, pp. 1361 – 1366.

[88] R. Madachy, "Integrated modeling of business value and software processes," *Lecture Notes in Computer Science*, Beijing, China, 2006, vol. 3840 LNCS, pp. 389 – 402.

[89] S. Sakthivel, R. Agarwal, "Knowledge-Based Model Construction for Simulating Information Systems," *Simulation*, vol. 59, no. 4, pp. 223–236, Oct. 1992.

[90] N. Hanakawa, S. Morisaki, K. Matsumoto, "Learning curve based simulation model for software development," in *Proc. International Conference on Software Engineering*, Kyoto, Jpn, 1998, pp. 350 – 359.

[91] N. Powell, A., Murdoch, J., Tudor, "Modeling risk-benefit assumptions in technology substitution," *Lecture Notes in Computer Science*, vol. 4470, pp. 295–306, 2007.

[92] I. Collofello, James S., Zhen, Yang, Tvedt, John D., Merrill, Derek, Rus, "Modeling software testing processes," in *Proc. of International Phoenix Conference on Computers and Communications*, Scottsdale, AZ, USA, 1996, pp. 289–293.

[93] R. R. Levary, C. Y. Lin, "Modelling the software development process using an expert simulation system having fuzzy logic," *Software - Practice and Experience*, vol. 21, no. 2, pp. 133 – 148, 1991.

[94] J. Lee, B., Miller, "Multi-project software engineering analysis using systems thinking," *Software Process Improvement and Practice*, vol. 9, no. 3, pp. 173–214, 2004.

[95] R. Koci, V. Janouek, "OOPN and DEVS formalisms for system specification and analysis," in *Proc. of 5th International Conference on Software Engineering Advances*, ICSEA 2010, Nice, France, 2010, pp. 305 – 310.

[96] A. Al-Emran. D. Pfahl, "Operational planning, re-planning and risk analysis for software releases," *Lecture Notes in Computer Science*, Riga, Latvia, 2007, vol. 4589, pp. 315 – 329.

[97] G. Waters, G., Linington, P., Akehurst, D., Utton, P., Martin, “Permabase: Predicting the performance of distributed systems at the design stage,” *IEEE Software*, vol. 148, no. 4, pp. 113–121, 2001.

[98] M. Nonaka, L. Zhu, M. A. Babar, M. Staples, “Project cost overrun simulation in software product line development,” *Lecture Notes in Computer Science*, Riga, Latvia, 2007, vol. 4589, pp. 330 – 344.

[99] M. Nonaka, Z. Liming, M. A. Babar, M. Staples, “Project delay variability simulation in software product line development,” *Lecture Notes in Computer Science*, Minneapolis, MN, United states, 2007, vol. 4470, pp. 283 – 294.

[100] H. Zhang, M. Huo, B. Kitchenham, R. Jeffery, “Qualitative simulation model for software engineering process,” in *Proc. of the Australian Software Engineering Conference*, ASWEC, Sydney, Australia, 2006, vol. 2006, pp. 391 – 400.

[101] H. Zhang, B. Kitchenham, “Semi-quantitative simulation modeling of software engineering process,” *Lecture Notes in Computer Science*, vol. 3966, pp. 242–253, 2006.

[102] A. Drappa, J. Ludewig, “Simulation in software engineering training,” in *Proc. of International Conference on Software Engineering*, Limerick, Ireland, 2000, pp. 199 – 208.

[103] M. Wu, H. Yan, “Simulation in software engineering with system dynamics: A case study,” *Journal of Software*, vol. 4, no. 10, pp. 1127 – 1135, 2009.

[104] B. Zhang, R. Zhang, “Simulation model of team software process using temporal parallel automata,” in *Proc. 4th International Conference on Internet Computing for Science and Engineering*, ICICSE 2009, Harbin, China, 2010, pp. 283 – 285.

[105] D. Pfahl, A. Al-Emran, G. Ruhe, “Simulation-based stability analysis for software release plans,” *Lecture Notes in Computer Science*, Shanghai, China, 2006, vol. 3966, pp. 262 – 273.

[106] N. Celik, H. Xi, D. Xu, Y. Son, “Simulation-based workforce assignment considering position in a social network,” in *Proc. of Winter Simulation Conference*, Baltimore, MD, United states, 2010, pp. 3228 – 3240.

[107] C. Caulfield, G. Kohli, S. P. Maj, “Sociology in software engineering,” in *ASEE Annual Conference Proc.*, Salt Lake City, UT, United states, 2004, pp. 12685 – 12697.

[108] C. L. . Wernick, P., Hall, T., Nehaniv, “Software evolutionary dynamics modelled as the activity of an actor-network,” *IET Software*, vol. 2, no. 4, pp. 321–336, 2008.

[109] D. M. Raffo, W. Harrison, J. Vandeville, “Software process decision support: Making process tradeoffs using a hybrid metrics, modeling and utility framework,” *ACM International Conference Proceeding Series*, Ischia, Italy, 2002, vol. 27, pp. 803 – 809.

[110] I. Rus, J. Collofello, P. Lakey, “Software process simulation for reliability management,” *Journal of Systems and Software*, vol. 46, no. 2, pp. 173 – 182, 1999.

[111] D.M. Raffo, J.V. Vandeville, R. H. Martin, “Software process simulation to achieve higher CMM levels,” *Journal of Systems and Software*, vol. 46, no. 2, pp. 163–172, 1999.

[112] P. Wernick, M. M. Lehman, “Software process white box modelling for FEAST/1,” *Journal of Systems and Software*, vol. 46, no. 2, pp. 193 – 201, 1999.

[113] C. Y. Lin, T. Abdel-Hamid, J. S. Sherif, “Software-Engineering Process Simulation model (SEPS),” *Journal of Systems and Software*, vol. 38, no. 3, pp. 263 – 277, 1997.

[114] R. Madachy, B. Boehm, J. A. Lane, “Spiral lifecycle increment modeling for new hybrid processes,” *Lecture Notes in Computer Science*, Shanghai, China, 2006, vol. 3966 LNCS, pp. 167 – 177.

[115] J. Zhai, Q. Yang, F. Su, J. Xiao, Q. Wang, M. Li, “Stochastic process algebra based software process simulation modeling,” *Lecture Notes in Computer Science*, Vancouver, Canada, 2009, vol. 5543, pp. 136 – 147.

[116] D.M. Raffo, S. Setamanit, “Supporting software process decisions using bi-directional simulation,” *International Journal of Software Engineering and Knowledge Engineering*, vol. 13, no. 5, pp. 513–530, 2003.

[117] R. J. Madachy, "System dynamics modeling of an inspection-based process," in *Proc. of International Conference on Software Engineering*, Berlin, Germany, 1995, pp. 376–386.

[118] F. Stallinger, P. Grunbacher, "System dynamics modelling and simulation of collaborative requirements engineering," *Journal of Systems and Software*, vol. 59, no. 3, pp. 311 – 321, 2001.

[119] G. Kahen, M. M. Lehman, J. F. Ramil, P. Wernick, "System dynamics modelling of software evolution processes for policy investigation: Approach and example," *Journal of Systems and Software*, vol. 59, no. 3, pp. 271–281, 2001.

[120] F. Huq, "Testing in the software development life-cycle: Now or later," *International Journal of Project Management*, vol. 18, no. 4, pp. 243–250, 2000.

[121] T. Wernick, P., Hall, "The impact of using pair programming on system evolution: A simulation-based study," in *IEEE International Conference on Software Maintenance*, ICSM, Chicago, IL, 2004, pp. 422–426.

[122] A. Avritzer, E. J. Weyuker, "The role of modeling in the performance testing of e-commerce applications," *IEEE Transactions on Software Engineering*, vol. 30, no. 12, pp. 1072 – 1083, 2004.

[123] G. Ruhe, A. Eberlein, D. Pfahl, "Trade-off analysis for requirements selection," *International Journal of Software Engineering and Knowledge Engineering*, vol. 13, pp. 345 – 366, 2003.

[124] K. Choi, S. Jung, H. Kim, D. Bae, D. Lee, "UML-based modeling and simulation method for mission-critical real-time embedded system development," in *Proc. of the IASTED International Conference on Software Engineering*, Innsbruck, Austria, 2006, vol. 2006, pp. 160 – 165.

[125] N. Smith, A. Capiluppi, J. Fernandez-Ramil, "Users and developers: An agent-based simulation of open source software evolution," *Lecture Notes in Computer Science*, Shanghai, China, 2006, vol. 3966, pp. 286 – 293.

[126] M. Ruiz, I. Ramos, M. Toro, "Using dynamic modeling and simulation to improve the COTS software process," *Product Focused Software Process Improvement*, vol. 3009, F. Bomarius and H. Iida, Eds. Springer-Verlag Berlin, 2004, pp. 568–581.

[127] D. Pfahl, A. Birk, "Using simulation to visualise and analyse product-process dependencies in software development projects," *Product Focused Software Process Improvement*, vol. 1840, F. Bomarius and M. Oivo, Eds. Springer-Verlag Berlin, 2000, pp. 88–102.

[128] D. Sycamore, J. S. Collofello, "Using system dynamics modeling to manage projects," in *Proc. IEEE Computer Society's International Computer Software and Applications Conference*, Phoenix, AZ, USA, 1999, pp. 213 – 217.

[129] E. Katsamakas, N. Georgantzas, "Why most open source development projects do not succeed?" in *First International Workshop on Emerging Trends in FLOSS Research and Development*, FLOSS'07, Minneapolis, MN, United states, 2007.

Appendix A – List of papers captured in the *quasi*-Systematic Review by year

Year	References	No. of papers
1986	[32]	1
1988	[16] [28]	2
1989	[29]	1
1990	[30]	1
1991	[31] [58] [66] [93]	4
1992	[89]	1
1995	[117]	1

1996	[92]	1
1997	[71] [78] [87] [113]	4
1998	[72] [90]	2
1999	[33] [54] [110] [111] [112] [128]	6
2000	[23] [102] [120] [127]	4
2001	[17] [19] [42] [50] [65] [97] [118] [119]	8
2002	[53] [56] [109]	3
2003	[24] [61] [64] [67] [116] [123]	6
2004	[18] [21] [43] [68] [70] [74] [82] [94] [107] [121] [122] [126]	12
2005	[20] [25] [60] [62]	4
2006	[22] [27] [39] [41] [81] [83] [88] [100] [101] [105] [114] [124] [125]	13
2007	[38] [52] [63] [69] [73] [75] [80] [91] [96] [98] [99] [129]	12
2008	[11] [12] [26] [40] [55] [57] [59] [76] [85] [86] [108]	11
2009	[34] [51] [79] [103] [115]	5
2010	[77] [84] [95] [104] [106]	5
2011	[35]	1