software

MDPI

Article

Empirical Formal Methods: Guidelines for Performing Empirical Studies on Formal Methods

Maurice H. ter Beek † **and Alessio Ferrari** *,†

Formal Methods and Tools Lab, ISTI–CNR, Via G. Moruzzi 1, 56124 Pisa, Italy; maurice.terbeek@isti.cnr.it
* Correspondence: alessio.ferrari@isti.cnr.it
† The authors contributed equally to this work.

Abstract: Empirical studies on formal methods and tools are rare. In this paper, we provide guidelines for such studies. We mention their main ingredients and then define nine different study strategies (usability testing, laboratory experiments with software and human subjects, case studies, qualitative studies, surveys, judgement studies, systematic literature reviews, and systematic mapping studies) and discuss for each of them their crucial characteristics, the difficulties of applying them to formal methods and tools, typical threats to validity, their maturity in formal methods, pointers to external guidelines, and pointers to studies in other fields. We conclude with a number of challenges for *empirical formal methods*.

Keywords: formal methods; empirical studies; guidelines

Citation: ter Beek, M.H.; Ferrari, A. Empirical Formal Methods: Guidelines for Performing Empirical Studies on Formal Methods. *Software* **2022**, *1*, 381–416. https://doi.org/10.3390/software1040017

Academic Editor: Li Li

Received: 9 August 2022
Accepted: 21 September 2022
Published: 24 September 2022

1. Introduction

For over two decades, empirical strategies, such as controlled experiments, case studies, surveys, literature reviews, etc., have largely been used to assess software engineering methods and tools, to study software practice, and to summarise research findings [1]. However, empirical studies on formal methods (FM), which are mathematically-based techniques and associated withols that typically target the development of demonstrably dependable software, are notably scarce. This has been highlighted already in 2007, by Höfer and Tichy [2], in their analysis of the status of empirical research in software engineering, and has been further stressed in 2015 by the research agenda of Jeffery et al. [3], calling for a better uptake of empirical methods in FM. A more recent call to arms for the FM community comes from the the manifesto for applicable FM by Gleirscher et al. [4] from 2021. One of the points of the manifesto states that "[FM] effectiveness should be evidenced. For example, it should be demonstrated (e.g., by means of case studies or controlled experiments) what would have been different if a conventional or non-formal alternative had been used instead". Among the expected impacts of the manifesto, it is worth mentioning the "Impact on the Conduct, Writing, and Review of Formal Method Research" and the "Impact on the Evaluation of Future Formal Method Research", both calling for case studies, action research, and controlled experiments. "Following these methods would greatly benefit the FM community". Additionally, Huisman et al. [5] recommend to "Invest time in industrially-relevant case studies in order to understand what techniques are actually needed for industrially-relevant applications".

Despite these statements, FM research remains focused on developing novel techniques, typically tackling more complex problems or performance issues, and tends to remain a method/tool focused discipline, rather than an evidence-based one. Furthermore, by focusing solely on the technical dimension, FM research does not sufficiently take into account human and social factors, which nevertheless affect the usage of FM tools [6]. In other terms, the discipline of *empirical formal methods* still remains a poorly explored avenue. Without demonstrated evidence of effectiveness and applicability, skepticism

about FM remains among practitioners, and the industrial uptake of FM is still limited. We argue that, among other factors, this is also hampered by the absence in the FM community of a sufficient knowledge of the available empirical strategies, their fundamental principles, and typical guidelines. Empirical guidelines exist for software engineering research. The book by Wohlin et al. [7] provides an overview of a selection of strategies (case studies, experiments, surveys, and systematic literature reviews), and gives detailed guidelines for laboratory experiments, mainly involving human subjects. The volume edited by Shull et al. [8] collects a set of articles about guidelines for qualitative methods, experiments, and principles of theory building in research. The more recent volume edited by Felderer and Travassos [9] includes a set of articles that update the body of knowledge in empirical software engineering, accounting for novel paradigms such as reviews of grey literature, Bayesian data analysis, optimisation methods, and data science. Furthermore, summary articles on the topic have been published during the first decade of the century [10,11], as well as complete overviews such as the ABC framework from Stol and Fitzgerald [12], which focuses on identifying the differences between different strategies and explicitly lists their limitations, considering the generalisability of the findings and the level of obtrusiveness of the research setting. Another notable effort of the software engineering community is the Empirical Standards initiative [13], which aims to provide checklists for conducting, and in particular *reviewing* empirical studies. A GitHub repository (https://github.com/acmsigsoft/EmpiricalStandards, accessed on 20 September 2022) is available where essential, desirable, and exceptional attributes are listed for each research strategy, together with links to exemplary papers. Despite this substantial amount of work, we are not aware of any publication that provides a concise and comprehensive set of guidelines with a specific focus on FM research. We argue that this is one of the hurdles that hamper a more widespread application of empirical research in FM.

To address this gap, this paper aims to support future research in *empirical formal methods* with a summary of the main strategies, and a set of guidelines to better apply them in FM. In particular, we consider nine empirical strategies, namely laboratory experiments with software subjects; laboratory experiments with human subjects; usability testing; surveys; qualitative studies—with reference to grounded theory; judgement studies; case studies—arguably including design science and action research; systematic literature reviews; and systematic mapping studies. Though other research strategies exist (cf., e.g., Stol and Fitzgerald [12] for a comprehensive framework), we believe that these can be considered as the most representative and useful for FM researchers. With respect to existing guidelines in software engineering, this paper makes the following main contributions:

- A practical foundation for *empirical formal methods*, aiming to encourage further empirical research in this field;
- A comprehensive overview of research strategies that can be applied to contribute to theory building in FM, together with their fundamental characteristics, and specific threats to validity;
- For each strategy, a reflection of the main difficulties and potential weaknesses for its application in FM;
- For each strategy, pointers to papers within FM and software engineering, as well as an up-to-date list of references providing detailed guidelines;
- An easy-to-use informal guide for selecting the most appropriate empirical strategy (cf. Figure 1).

In the remainder of this paper, we first provide an overview of the main ingredients that each empirical study should contain, such as research questions, data collection and analysis procedures, and threats to validity (Section 2). Then, for each strategy (Sections 3–10), we summarise: their crucial characteristics, the difficulties of applying them to FM and tools, typical threats to validity, their maturity in FM, pointers to external guidelines, and pointers to studies in other fields. Finally, we provide final remarks as well as recommendations on how to choose the most appropriate research strategy for the problem at hand (Section 11).

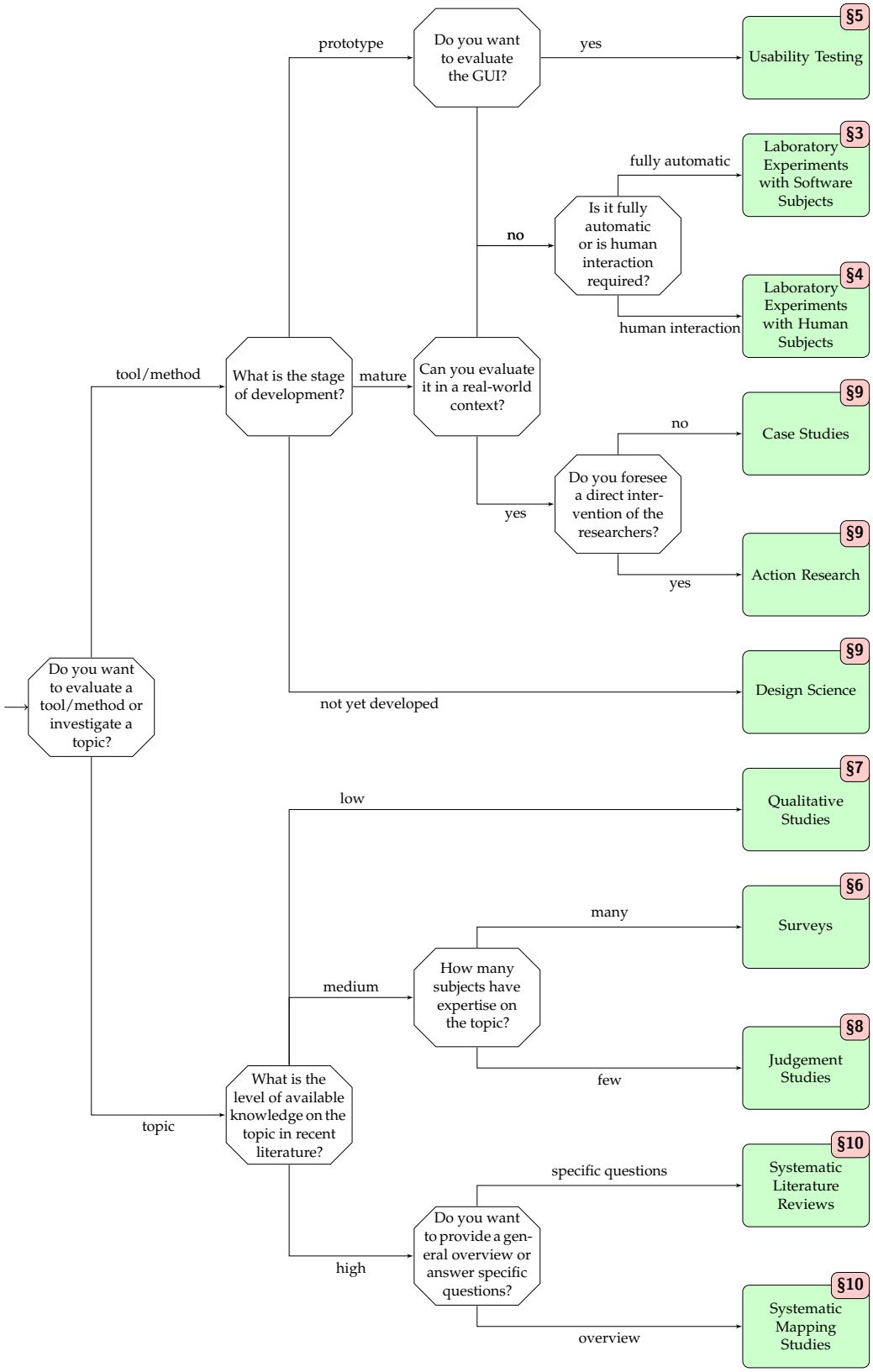

Figure 1. Informal guide to the selection of the most appropriate research strategy.

This paper can be used as a concise reference for FM researchers who want to carry out an empirical study, but do not know where to start from—and do not want to incur in typical pitfalls. Our wish is to facilitate the development of an empirical mindset in the FM community.

2. Fundamental Ingredients

Empirical studies are structured research procedures that aim to derive some theory from the observation of phenomena in a study setting. They apply systematic protocols for data collection and analysis, accompanied by validity procedures to reduce researcher bias and mitigate threats to validity. Empirical studies differ for their degree of realism, the ability to generalise outside the study setting, and the ability to isolate the observed phenomena from exogenous factors. However, they all have in common a general structure to keep in mind, which can be useful also for reporting. The fundamental ingredients of this structure are:

- **Research Questions (RQs):** these are statements in the interrogative form that drive the research. They are useful as a guideline for the researchers, who has a set of clear objectives to address, but also for the reader. The RQs also typically include the *constructs* of interest, which are the abstract concepts (e.g., efficiency, usability) to be investigated through the research.
- **Data Collection Procedure:** since empirical studies stem from data, these need to be collected, and a systematic and repeatable procedure needs to be established. The data collection procedure specifies which are the data sources, and how data is collected. Data are related to the constructs of interest, as one aims to use the data to measure or evaluate such constructs.
- **Data Analysis Procedure:** this specifies how the data is elaborated and interpreted to answer the RQs, thus establishing a chain of evidence that goes from data to constructs of interest. In other terms, the data analysis procedure establishes a link between empirical data and RQs. Both data collection and analysis procedures require to consider possible validity threats, and countermeasures to prevent possible threats need to be established and made explicit.
- **Execution and Results:** these specify how data collection and analysis have been carried out, and what is the specific output of these procedures. This part also systematically answers the RQs, based on the available evidence, while in principle data collection and analysis abstract away from concrete data, here the focus is specifically on the data and their interpretation.
- **Threats to Validity:** these specify what are the possible uncontrolled threats that could have occurred in data collection and analysis, and that could have influenced the observed results. In this part, the researchers should reinstate the mitigation strategies oriented to address typical threats to validity, and acknowledge residual threats. Different threats can typically occur depending on the type of study. Nevertheless, there are three main categories of threats, which in principle apply only to experiments, but that introduce a reasoning framework that can be useful for other types of studies:
  - *Construct Validity:* indicates to what extent the abstract constructs are correctly operationalised into variables that can be quantitatively measured, or qualitative evaluated. To ensure construct validity the researcher should show that the constructs of interests are well-defined and well-understood based on existing literature. Furthermore, the researcher should argue about the soundness of the proposed quantitative measures or evaluation strategies. For example, if a researcher wishes to measure effectiveness of a certain tool T, they should present related literature defining the concept of effectiveness, and defining a sound measure for this construct.
  - *Internal Validity:* indicates to what extent the researcher has ensured control of confounding external factors that could have impacted the results. These factors includes *researcher bias*, i.e., expectations/inclinations of the researcher that may have impacted the study design (e.g., in a questionnaire definition, or in the data analysis), and any aspect related to subjectivity or context-dependency in the production of the results. Internal validity can also be threatened by time-related aspects, e.g., with a *maturation* effect that could occur when the participants perform multiple tasks one after the other, or with *fatigue* effects due to long

experimental treatments. For example, consider the case of comparing two tools A and B on a certain task K. If the subjects first use tool A and then tool B on task K, a learning effect could occur. Indeed, with tool A, they could have learned about task K, and this would have facilitated them in performing the same task when using tool B. To address this issue, the researcher could allocate some subjects only on tool A and others only on tool B.

– *External Validity:* indicates to what extent the results can be applicable to contexts other than the one of the study, or, in other terms, to what extent the results can be considered general, i.e., what is the *scope of validity* of the study.

When selecting a research strategy for an empirical study, and defining a study design, one should consider that there is always a *trade-off* between internal and external validity, and also between the knowledge depth that one could achieve, and the generalisability of the results [12]. For example, an experiment should include realistic elements, but its results are typically hardly applicable to real-world cases as the lab context is largely different from a real context—e.g., time constraints, limited realism of models or programs analysed, and the overall in vitro, fictional context. Conversely, a case study is highly realistic, but its results are specific to the organisation in which the case study is carried out, and can hardly be applicable to other contexts. A survey can achieve a high degree of external validity, as a statistically relevant set of subjects are included, but the degree of internal validity is limited, as one can hardly control the subjectivity of the responses. Furthermore, since surveys are oriented to a large number of subjects, the questions should be easy to understand, which limits the knowledge depth that one can achieve, compared, e.g., with case studies or qualitative studies, in which highly informative interviews are carried out, with the possibility of follow-up questions.

3. Laboratory Experiments with Software Subjects

- **Definition of the Strategy:** a laboratory experiment is a research strategy carried out in a contrived setting in which the researcher wishes to minimise the influence of confounding factors on the study results. In an experiment with software subjects, the researcher typically compares different tools, algorithms or techniques, to collect evidence, e.g., of their efficiency or effectiveness on a certain representative set of problems.
- **Crucial characteristics:** in a laboratory experiment with software subjects, one typically defines measurable constructs to be assessed and used for comparison between different software subjects. More specifically, the researcher identifies the constructs of interest, and how these constructs are mapped into variables that can be quantitatively measured, or, if this is not feasible, qualitatively estimated. The constructs of interest are typically strictly connected with the RQs, and the data are the source information that can be used to answer the RQs. Therefore, the researcher also needs to specify how one aims to collect the data associated with the variables. For example, in a quantitative study one may want to focus on the construct of *effectiveness* of a certain tool, with the RQ: *What is the effectiveness of tool T?* If the tool T is designed to find bugs in a certain artefact, this construct can be measured with the variable *bug identification rate* = number of identified bugs/total bugs. The data collection strategy could consist of measuring the number of bugs found by tool T on a specific dataset given as input (*number of identified bugs*), which contain a pre-defined set of bugs (*total bugs*). The dataset, also called *benchmark*, should be representative of the set of programs that the tool T aims to verify. If the tool T is designed for a specific type of artefact, then the artefacts should vary across different variables that characterise the artefact: e.g., language, size of the artefact, complexity. It is important to always report the characteristics of the dataset across these salient dimensions. Furthermore, to assess whether the effectiveness of a certain software subject is 'good enough', one also needs to define one or more *baselines*, i.e., other tools previously developed, or an artificial

predictor (e.g., random, majority class), that can allow the researchers to state that the software subject overcomes the existing baselines for the given dataset.

- **Weaknesses/Difficulties in FM:** several difficulties may occur when applying this type of strategy in FM. Typically, a software subject is a tool such as, e.g., a model checker or a theorem prover. If the tool requires some interaction with the user, then this can affect the results, as the variable that is measured, e.g., bug identification rate, also depends on the human operator. To address this issue, the experiments should also include design elements that are proper of laboratory experiments with human subjects (cf. Section 4). Another pitfall can occur when the tool uses some random or probabilistic principle, and thus the results of the experiment can vary from one execution to the other. In these cases, the tool should be executed multiple times, and confidence intervals, e.g., on its effectiveness or other performance-related constructs, should be estimated and reported with appropriate p-values [14]. Furthermore, if one aims to report differences between different tools across multiple runs, appropriate statistical tests, e.g., t-test or Mann–Whitney U test, should be also performed, again reporting p-values and evaluating effect size. Another typical difficulty is identifying a baseline. Indeed, formal tools often target specific fine-grained objectives, e.g., *runtime verification* vs. *property proving*, and, in the case of model checkers, can use different modelling languages and different logics for expressing properties. Therefore, the comparison between tools is often hardly possible. In these cases, one can (i) define simple artificial baselines, against which the tools can be compared; (ii) restrict the comparison to the subset of the dataset for which a comparison is possible; and (iii) complement the quantitative evaluation with a qualitative evaluation, involving human subjects in the assessment of the effectiveness of the tool, e.g., with a usability study/judgement study (cf. Sections 5 and 8), a questionnaire provided to users after using the tool, or qualitative effect analysis [15].
- **Typical threats to validity:** typical threats are related to the representativeness of the dataset (*external validity*), the soundness of the research design (*internal validity*), and the definition of variables and associated measures (*construct validity*). An inherent threat of this type of study, as for laboratory experiments in general, is the limited realism, as the lab setting is typically contrived and does not account for real-world aspects, e.g., learning curve required to learn a tool, incremental and iterative interaction with tools, or iterative nature of artefact development, which is normally not captured by a fixed dataset.
- **Maturity in FM:** laboratory experiments with software subjects and in particular tool comparison is relatively mature in FM. Many different competitions exist in which tools are evaluated in terms of performance (evaluation of their usability is rare). Experiments are typically conducted on a representative set of benchmark problems and executed by benchmarking environments like BenchExec [16], BenchKit [17], DataMill [18], or StarExec [19]. The oldest competitions concern Boolean satisfiability (SAT) solvers [20], initiated three decades ago, and Automated Theorem Provers (ATP) [21]. In 2019, 16 competitions in FM joined TOOLympics [22] to celebrate the 25th anniversary of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS). For several years now, TACAS and other FM conferences also feature artefact evaluations to improve and reward reproducibility, following their success in software engineering conferences, where they have been introduced over a decade ago [23,24]. A recent survey on FM ([25], Section 5.9) showed that their adoption in industry would benefit a lot from the construction of benchmarks and datasets for FM.
- **Pointers to external guidelines:** guidelines and recommendations for this kind of studies, also called *benchmarking studies*, are provided by Beyer et al. [16] and Vitek et al. [26]. Guidelines for combining these studies with other assessment methods are part of the DESMET (Determining an Evaluation Methodology for Software Methods and Tools) methodology by Kitchenham et al. [15].

- **Pointers to studies outside FM:** an example in the field of automatic program repair is the study by Ye et al. [27], which applies different automatic program repair techniques to the QuixBugs dataset and extensively report also the characteristics of the dataset. In automated GUI testing, Su et al. [28] compare different tools for the constructs of effectiveness, stability and efficiency. This is also a good example of applying statistical tests to the comparison of different tools. Herbold et al. [29] also uses statistical tests to compare cross-project defect prediction strategies from the literature on a common dataset. Another representative example, in the field of requirements engineering, is the work by Falessi et al. [30]. This can be taken as reference in case a set of building blocks need to be combined to produce different variants to be compared. The work is particularly interesting because it also illustrates the empirical principles that underpin the comparison. It should be noticed that the paper does not use a public and representative dataset as benchmark, but a dataset that is specifically from a company, thus including the laboratory experiment in the context of a case study (cf. Section 9). A similar problem of composition of building blocks is considered also by Maalej et al. [31], in the field of app review analysis, and by Abualhaija et al. [32], in the field of natural language processing applied to requirements engineering. This latter study is particularly interesting as it complements the performance evaluation with a survey with experts.

4. Laboratory Experiments with Human Subjects

- **Definition of the Strategy:** similarly to laboratory experiments with software subjects, an experiment with human subjects is a research strategy carried out in a contrived settings, in which the researcher wishes to minimise the influence of confounding factors on the study results. In these experiments, the typical goal is to evaluate constructs (e.g., understandably, effectiveness) concerning notations, tools, or methodologies, when their evaluation depends on human interaction.
- **Crucial characteristics:** in a laboratory experiment with human subjects, one typically defines measurable constructs to be assessed and used for comparison between different study objects, i.e., notations, tools, or methodologies. More technically, a laboratory experiment with software subjects typically evaluates the effect of certain *independent variables* (e.g., the tool under study, or the experience of the subjects) on other *dependent variables* (e.g., understandability, effectiveness). The researcher thus identifies the constructs of interest, and how these constructs are mapped into variables that can be measured quantitatively or, if this is not feasible, estimated qualitatively. The constructs of interest are typically strictly connected with the RQs, and the data is the source information that can be used to answer the RQs. As for laboratory experiments with software subjects, the researcher also needs to specify how one aims to collect the data associated with the variables. To this end, the researcher typically recruits a set of human subjects, either professional, or, more often, students, and asks them to perform a given task using the objects of the study, i.e., notations, tools, or methodologies. Subjects are typically divided into groups, or *treatments*, the experimental group and the control group. The former uses the object of the study to perform the task. The latter performs the task without using the object of the study. In the task, data is collected concerning the dependent variables, and in relation to the RQs. In these experiments, it is typical to refine the RQs into *hypotheses* to be statistically tested, based on evidence collected from the data of the experiment itself. An experiment with statistical hypothesis testing can be seen as two sequential black boxes, an experiment box, and a statistical assessment box. The first box represents the experiment itself which produces data, while the second one represents the actual procedure of hypothesis testing, which uses the data produced by the experiment box to state to what extent one can be confident that, given the data, the effect observed in the data is *not* due to chance. In the experiment box: (i) the inputs are the so-called *independent variables*, i.e., the variables that the researcher wants to manipulate, for example the

type of tool to be used in the treatment; (ii) the outputs are *dependent variables*, i.e., the variables that represent the constructs that one wishes to observe, e.g., effectiveness, understandability; and (iii) additional input parameters, e.g., maximum time to execute the task, exercise used in the task. These are the *controlled variables* that are not considered independent variables, but that could influence the effect of independent variables on dependent variables, if not properly controlled, and if their effect is not properly cancelled. In the hypothesis testing box, the inputs are all the data associated with the dependent and independent variables, and the main outputs are: (i) the *effect size*, which represents the degree of the observed impact of independent variables on dependent variables; and (ii) the *statistical significance* of the results obtained, which is given by the p-value. The statistical significance roughly indicates how likely it is that the results obtained are due to chance, and not to the treatment. Therefore, lower p-values are preferable, and one normally identifies a significance level, called α, often set to 0.05. When p-value $\leq \alpha$, results are considered significant. In the hypothesis testing box, the output is produced from the input using a certain statistical test (e.g., t-test, ANOVA) that depends on the type of experimental design, and the nature of the variables under study (e.g., rate variables, categorical).

- **Weaknesses/Difficulties in FM:** applying this type of strategy in FM is made hard by the inherent complexity of most FM. A laboratory experiment typically wants to assess the effectiveness of a tool, but the subjects who will use the tool sometimes also need to be trained on the *theory* that underlies the tool, e.g., formal language, notation, and usage itself. This means that laboratory experiments may need to involve 'experts' in FM. However, experts in FM are typically proficient on a specific and well-defined set of approaches (e.g., theorem proving vs. model checking), and even tools [33,34]. Therefore, comparable subjects with similar expertise and in a sufficient number to achieve both statistical power and significance, are hard to recruit, and this makes it difficult to carry out experiments in FM. A possible solution is to focus experiments on fine grained, simple, aspects that can be taught in the time span of a class or a limited tutorial, e.g., a graphical notation or a specific temporal logic. If one wishes to evaluate formal tools, and in particular their user interfaces, it is feasible to perform usability studies (cf. Section 5). These do not normally require a large sample size (in many settings, 10 ± 2 subjects are considered sufficient [35,36], when one adopts specific usability techniques), as they do not aim to assess significance but rather to spot out specific usability pitfalls. If, instead, one wishes to evaluate entire methodologies, it is recommended to decompose them into steps, and design experiments that evaluate one step at the time, e.g., distinguishing between comprehension, modelling phase, verification phase.
- **Typical threats to validity:** typical threats to validity are associated with *construct validity*, i.e., to what extent the constructs are correctly operationalised into variables, *internal validity*, i.e., to what extent the research design is sound, and all possible factors that could have affected the outcome have been properly controlled, *external validity*, i.e., to what extent the results obtained are applicable to other setting, for example a real-world setting, and *conclusion validity*, which specifies to what extent the statistical tests provide confidence on the conclusion. For conclusion validity, one needs to specify: that the assumption of statistical tests are considered and properly assessed—most of the tests (so called *parametric*) assume a normal distribution of the variables; the value of the statistical power of the tests, which can be estimated based on the number of subjects involved, and that gives an indication of how likely it is that one has incorrectly missed an effect between the variables, while an effect is actually present. Similarly to laboratory experiments with software subjects, an inherent threat is the low degree of realism, as human subjects undertake a task in a constrained environment which somewhat simulates how the task would be carried out in the real world. In other terms, the external validity is inherently limited for these types of study, which tend to maximise internal validity.

- **Maturity in FM:** not surprisingly, given the complexity of many FM, laboratory experiments with human subjects are not extremely mature in FM, but quite some experiments exist. Sobel and Clarkson [37] conducted one of the first quasi-experiments, in an instructional setting, where undergraduate students developed an elevator scheduling system—with and without using FM. "The FM group produced better designs and implementations than the control group". Debatably [38,39], this contradicts Pfleeger and Hatton [40], who investigated the effects of using FM in a case study, in an industrial setting, where professionals developed an air-traffic-control information system. They "found no compelling quantitative evidence that formal design techniques *alone* produced code of higher quality than informal design techniques", yet "conclude that formal design, combined with other techniques, yielded highly reliable code". We are also aware of some well-conducted controlled experiments for the comprehensibility of FM like Petri nets [41], Z [42,43], OBJ [44,45], and B [46,47], for a set of state-based (semi-)formal languages like Statecharts and the Requirements State Machine Language (RSML) [48], and for domain-specific methods and languages in business process modelling [49–51], software product lines [52,53] and security [54–56]. Further empirical studies on the usability of such FM would be very welcome. The same holds for human comprehensibility and usability of other well-known FM (e.g., Abstract State Machines (ASM), the Temporal Logic of Actions (TLA), and calculi like the Calculus of Communicating Systems (CCS) and Communicating Sequential Processes (CSP)), even prior to evaluating the effectiveness of tools based on such FM. We note that the formalisms of attack trees and attack-defense trees, popularised by Schneier [57] and formalised by Mauw et al. [58,59], are claimed to have an easily understandable human-readable notation [57,60]. However, as reported in [61,62], there have apparently been no empirical studies on their human comprehensibility. Thus, also in this case laboratory experiments with human subjects would be much needed.
- **Pointers to external guidelines:** Wholin et al. [7] published a book on experimentation in software engineering and the principles expressed in the book also apply to experiments in FM. A practical guide on conducting experiments with tools involving human participants is provided by Ko et al. [63]. Guidelines for analysing families of experiments or replications are provided by Santos et al. [64]. To have more insights on experiment design with human subjects, one can also refer to the Research Methods Knowledge Base (https://conjointly.com/kb/, accessed on 20 September 2022) [65], an online manual primarily designed for social science, but appropriate also for experiments in FM. When psychometric is involved because some questionnaire are used to evaluate certain variables, the reader should refer to the guidelines by Graziotin et al. [66], specific to software engineering research. To acquire background on the statistics used in experiments, the handbook by Box et al. [14] is a major reference. To focus on hypothesis testing, with clear and intuitive guidelines for the selection of the types of tests to apply, one of the main reference is the book by Motulsky [67]. The book is, in principle, oriented to biologists, but the provided guidelines are presented in an intuitive and general way, which is appropriate also for an FM readership. It should be noted that, though widely adopted, hypothesis testing has several shortcomings that have been criticised by the research community. Bayesian Data Analysis has been advocated as an alternative option, and guidelines in the field of software engineering have been provided by Furia et al. [68].
- **Pointers to studies outside FM:** in software engineering it is quite common to use this strategy, for example to evaluate visual/model-based languages, as done for example by the works of Abrahão et al. [69,70], focused on modelling notations. In the evaluation of methodologies, a representative work is the one by Santos et al. [71] on test-driven development. When one wants to focus on single specific methodological step, a reference work is the one by Mohanani et al. [72], about different strategies for framing requirements and their impact on creativity. When the focus is human factors, e.g., competence or domain knowledge, a representative work is the one by

Aranda et al. [73], on the effect of domain knowledge on elicitation activities. Finally a comparison between an automated procedure and a manual one for feature location is presented by Perez et al. [74].

5. Usability Testing

- **Definition of the strategy:** usability testing focuses on observing users working with a product, and performing realistic tasks that are meaningful to them. The objective of the test is to measure usability-related variables (e.g., efficiency, effectiveness, satisfaction), and analyse users' qualitative feedback. It can be seen as a laboratory experiment, but (i) with a more standardised design; (ii) with a limited amount of subjects (6 to 12, belonging to 2–3 user profile groups are considered sufficient by Dumas and Redish [75]); (iii) collecting both quantitative and qualitative data; and (iv) whose goal is to identify usability issues, rather than testing hypothesis and achieving statistical significance, which typically require larger samples. If larger groups of subjects are available, though, quantitative results of usability tests can be evaluated with statistical tests.
- **Crucial characteristics:** usability studies can be classified into three main types: (i) heuristic inspections (or *expert reviews*), in which a usability expert critically analyses a product according to pre-defined usability criteria (cf. the list from Nielsen and Molich [76]), without involving users; (ii) cognitive walkthrough, in which a researcher goes through the steps of the main tasks that one expects to perform with a product, and reflects on potential user reactions [77]; and (iii) usability testing, in which users are directly involved. Here we focus on usability testing, which is also the most common and most studied technique. Usability and usability tests are also the topic of the ISO 9241-11:2018 Part 11 standard [78]. In usability tests, the constructs to evaluate, and the related RQs, are pre-defined by the literature, as the researcher typically wants to assess a product according to a set of usability attributes. The usability attributes considered by the ISO standard are *effectiveness* (to what extent users' goals are achieved), *efficiency* (how much resources are used), and *satisfaction* (the user personal judgement with the experience of using the tool). Other possible framing of the usability attributes are the 5E expected from a product, i.e., efficient, effective, engaging (equivalent to satisfaction), *error tolerant*, and *easy to learn* (i.e., time to become proficient with the tool) [79]. Holzinger, instead, considers learnability, efficiency, satisfaction, low error rate (analogous to effectiveness), and also *memorability* (to what extent a casual user can return to work with the tool without a full re-training) [80]. After a selection of the usability attributes (constructs) that the researcher wants to assess, one needs to define the user profile that will be considered in the test. Test subjects will be selected accordingly and a screening and/or pre-test (i.e., a sort of demographic questionnaire) will be carried out to assess that the expected profile is actually matched by the subjects. Data collection is performed through the test itself, which is supposed to last about one hour for each subject. A set of task-based scenarios are defined (e.g., installation, loading a model, modifying a model, verification, etc.), which the user needs to perform with the tool. Typically, a moderator is present at the test, who will interact with the users, instruct them, incrementally assign tasks and tests, and be available for support, if needed. An observer should also be appointed, who will take notes on user's physical and verbal reactions. Video equipment, as well as microphones, logging computers, logging software (e.g., Inputlog (https://www.inputlog.net/overview/, accessed on 20 September 2022), Userlytics (https://www.userlytics.com/, accessed on 20 September 2022), ShareX (https://www.goodfirms.co/software/sharex, accessed on 20 September 2022)), and eye-tracking devices can be used, depending on the available resources. During the usability test sessions, it is highly recommended to ask the participants to *think aloud*, which means verbalising actions, expectations, decisions, and reactions (e.g., "now I am pressing the button to verify the model, I expect it to start verification, and to have the results immediately"; "the tool is stuck, I do not

know if it is doing something or not"; "now I see this result, and I cannot interpret it"). After each task, the user should answer at least the Single Easy Questionnaire (SEQ) test, which means asking how easy was the task in a 7-point scale from 1—Very Difficult to 7—Very Easy. Other short questions about the perceived time required, and the intention to use can also be asked. After completion of all the tasks, the user typically fills a post-test questionnaire, which measure perception-related variables. Several standard questionnaires exist, e.g., SUS (System Usability Scale) and CSUQ (Computer System Usability Questionnaire)—cf. Sauro and Lewis for a complete list [81]. Data from the test are both quantitative and qualitative. Quantitative data include: time on tasks, success or completion rate for the tasks, error rate with recovery from errors, failure rate (no completion or no recovery), assistance requests, search (support from documentation) and other data that can be logged. Results of the post-task and post-test questionnaires, associated with perception-related aspects, are also quantitative. Typical variables evaluated in usability studies, with associated metrics, are reported by Hornbaek [82]. Qualitative data include think aloud and observation. Data analysis for quantitative data consists of assessing to what extent the different rates are acceptable, after establishing expected rates beforehand. Since, in the end, what matters is the user perception, the results of post-task and post-test questionnaire are somehow prioritised in terms of relevance, together with qualitative data. For the SUS test, scores go from 0 to 100, and rates above 68 are considered above average. For qualitative data, coding and thematic analysis can be used, similar to qualitative studies (cf. Section 7). Overall, as for laboratory experiments with human subjects, it is important to establish a clear link between constructs to evaluate and measure variables. Quantitative and qualitative data should be also triangulated to identify further insight. For example, qualitative data may indicate satisfaction in learning the tool, although, e.g., the error rate is high. When more subjects are available for the test, e.g., 30 or more, and one wants to establish statistical relations between variables such as effectiveness, perceived usability and intention to use, one can refer to the Technology Acceptance Model (TAM) [83]. The relation between usability dimensions and TAM are discussed by Lin [84].

- **Weaknesses/Difficulties in FM:** applying usability tests for FM tools is complicated by the typical need to first learn the underlying theory and principles, and then using an FM tool. It is sometimes difficult to separate difficulty in learning, e.g., a modelling language or a temporal logic, from the usability of a model checker. Therefore, the researcher should first assess the learnability of the theory, e.g., with laboratory experiments with human subjects, and afterwards should evaluate the FM tool. Selected subjects should have learned the theory before using the tool, and usability of FM tools should preferably be evaluated also after some time of usage, so that initial learning barriers have already been overcome by users. If one aims to make a first usability test with users that are not acquainted with FM—which can happen if the target users are industrial practitioners—one can present a tool, perform some tasks and use available post-test questionnaires, like SUS, to get a first measurable feedback, as was done in previous studies [85]. When industrial subjects are involved, the difficulty for the researcher is to find problems that are meaningful to the domain of the users, so that these can perceive the potential relevance of the tool. Another difficulty is the typical focus of FM tool developers on the performance of tools, especially for model checkers, with respect to usability aspects, because tools often come from research, which rewards technical aspects instead of user-relevant ones. To be effective, usability tests should be iterative, with versions of an FM tool that are incrementally improved based on the test output. This requires resources specifically dedicated to usability testing. However, if this is not possible, heuristic inspections or cognitive walkthrough should at least be performed on an intermediate version of the tool. Another issuse with FM tools is that these are not websites, but complex systems, which can have several functionalities to test (e.g., simulation or

different types of verification as in many model checkers). Given the complexity, one should focus on the most critical aspects to be tested. Another issue with FM tools is the time that is sometimes required to perform verification which makes a realistic test on a complex model often infeasible. In these cases, it is useful to use the so-called Wizard of Oz method (WOZ) [86], in which the output is prepared and produced beforehand, or part of the interaction is simulated remotely by a human.

- **Typical threats to validity:** the typical *construct validity* threats are generally addressed thanks to the usage of well-defined usability attributes and measures. Particular care should be dedicated to the selection of the subjects, so that these are actually representative of the user group considered in the study. Pre-tests or initial screening can mitigate threats. Additional threats to construct validity are related to the way questionnaires are presented. Depending on the formulation of the tests, error of central tendency (the tendency to avoid the selection of extreme values in a scale), consistent response bias (responding with the same answer to similar questions), and serial position effect (tendency to select the first or final items in a list) need to be prevented. Error of central tendency can be addressed by eliminating central answers, or by asking respondents to explicitly rank items. Consistent response bias can be addressed by using negative versions of the same question, and shuffling the questions. Serial position effect is addressed by shuffling the list of possible answers. To guarantee that the answers to the different questions are a correct proxy of the constructs that one wishes to evaluate, it is also important to perform inter-item correlation analysis [87]. *Internal validity* can be hampered by think-aloud activities, which can influence the behaviour of the user, interaction with the moderators, expectations from the tool and possible rewards given after the activity. These threats cannot be entirely mitigated, but the researcher should clarify the following with the user: (i) what is the status of the tool and the goal of the activity, so that expectations are clear; (ii) interaction should be minimised; (iii) it is the tool that is under evaluation and not the user; and (iv) the reward will be given regardless of the results. Overall, to ensure that the analysis is not biased, it is also important to perform triangulation, that is reasoning about relations between think aloud, observations and post/task and post-test questionnaires. Concerning *external validity*, this can be limited by the low degree of realism given by the test environment, which happens for laboratory experiments. To reduce this, one can perform the test in the real environment, in which the user is typically working, so that interruption, noise and other factors can make the evaluation more realistic.
- **Maturity in FM:** throughout the years there have been efforts to address usability, but it has by no means become standard practice and many FM tools have never been analysed for what concerns their usability. The PhD thesis of Kadoda [88] addresses the usability aspects of FM tools. First, using the usability evaluation criteria proposed by Shackel [89], two syntax-directed editing tools for writing formal specifications are compared in a practical setting; second, using the cognitive dimensions framework proposed by Green and Petre [90], the usability of 17 theorem provers is analysed. Hussey et al. [91] demonstrate usability analysis of Object-Z user-interface designs through two small case studies. In parallel, there have been many attempts at improving the usability of specific FM through the use of dedicated user-friendly toolsets and the like to hide FM intricacies from non-expert users like practitioners, ranging from the SCR Requirements Reuse ($SC(R)^3$) toolset [92] through the IFADIS toolkit [93,94] to FRAMA-C platform [95] and the ASMETA toolset [96] built around the ASM method. Recently, a preliminary comparative usability study of seven FM (verification) tools involving railway practitioners was conducted [85]. The importance of usability studies of FM is confirmed by the recent FM survey by Garavel et al., in which over two-thirds of the 130 experts that participated responded that it is a top priority for FM researchers to "develop more usable software tools" ([25], Section 4.5).
- **Pointers to external guidelines:** for a prescriptive introduction to usability, the reader should refer to the ISO 9241-11:2018 Part 11 standard [78]. A main reference for

usability testing is the handbook by Rubin and Chisnell [97]. Nielsen and Molich provide 10 ways to perform heuristic evaluation [76], while Mahatody et al. [77] report the state of the art of cognitive walkthrough. Quantitative metrics to measure usability attributes are given by Hornbaek [82]. Several resources are made available also via specialised websites (e.g., https://usabilitygeek.com/, accessed on 20 September 2022).

- **Pointers to papers outside FM:** A systematic literature review on usability testing, with references scored by their quality, is provided by Sagar and Anju [98]. A recent work addressing usability of two modelling tools is presented by Planas [99]. For works using TAM, and focusing on the assessment of attributes related to usability, also including understandability of languages, the reader can consider the works by Abrahão et al. [70,100].

6. Surveys

- **Definition of the Strategy:** A survey is a method to systematically gather qualitative and quantitative data related to certain constructs of interests from a group of individuals that are representative of a population of interest. The constructs are concepts that one wants to evaluate, e.g., usability of a certain tool or developers' habits. The population of interest (also *target population* or *population*) is the group of individuals that is the focus of the survey, e.g., users of tool T, companies in a certain area, users of tool T from University A vs. users from University B, potential users of tool T with a background in computer science, etc. Surveys are normally oriented to produce statistics, so their output normally takes a quantitative form. Surveys are typically conducted by means of questionnaires, but they can be also carried out through interviews.
- **Crucial characteristics:** The survey process starts from RQs, and the identification of the constructs of interest just as for the other methods discussed. Then, one needs to characterise the target population, i.e., what are the characteristics of the subjects that will take the survey. Based on these characteristics, the researcher performs *sampling*, which means selecting a subset of subjects that can be considered representative for the population. This is normally carried out with probability sampling, in which subjects are selected according to some probability function (random, or stratified—i.e., based on subgroups of the population) from a sampling frame (i.e., an identifiable list of subjects that in an optimal scenario should cover the entire population of interest, for example the list of e-mail addresses of a company). The sample size required for the survey can be computed considering the size of the target population, desired confidence level, confidence interval and other parameters [101,102]. When designing a survey, one also needs to consider that a relevant portion of the selected subjects, usually about 80–90%, will not respond to the inquiry. Therefore, to have significant results, one needs to plan for a broad dissemination of the survey, so that, even with a low response rate, the desired sample is reached. If personal data is collected, it is also important to make sure to adhere to the GDPR [103] and to present an informed consent to the subjects. In this phase, it is also important to define the data management plan (for a template, check https://ec.europa.eu/research/participants/docs/h2020-funding-guide/cross-cutting-issues/open-access-data-management/data-management_en.htm#A1-template, accessed on 20 September 2022), which includes how the data will be stored and when it will be deleted. After determining the sample size, one needs to design the survey instrument, which can be composed of open-ended and/or close-ended questions. Each type of question has its own advantages and disadvantages, e.g., open-ended questions are richer in information but harder to process, while close-ended questions enable less spontaneous and extensive answers, but are easier to analyse and lead to comparable results between subjects. Regardless of the types of questions selected, a well-designed survey has the following attributes: (i) *clarity*, i.e., to what extent the questions are sufficiently clear to elicit the desired information; (ii) *comprehensiveness*,

i.e., to what extent the questions and answers are relevant and cover all the important information required to answer by the RQs; and (iii) *acceptability*, i.e., to what extent the questions are acceptable in terms of time required to answer them and preservation of privacy and ethical issues. To address these attributes, researchers should perform repeated pilots of the survey instrument, with relevant subjects. If the researcher is not sufficiently confident with the topic of the survey or the type of respondents, it is also useful perform a set of interviews with selected subjects, to better define the questions to be included in the survey instrument. After piloting, the survey can be distributed to the selected sample, and the answers need to be recorded, following the data management plan defined beforehand. Then data is analysed and interpreted. In this phase, researchers should perform some form of *coding* for answers to open-ended questions (cf. Section 7), and should adjust the data considering missing answers. Data analysis and reporting can be performed by first presenting quantitative statistics, with percentages of respondents, possibly followed by more advanced statistical analysis. For example, if RQs concern relationships between variables, statistical hypothesis tests can be performed similar to laboratory experiments with human subjects (cf. Section 4). Other advanced methods include Structured Equation Modeling (SEM), which allows researchers to identify relationships between high-level, conceptual and so-called 'latent' variables (e.g., background, success, industrial adoption), by analysing multiple observable indicators that can be extracted from the survey (e.g., educational degree and current profession can be considered as indicators of background) [104,105].

- **Weaknesses/Difficulties in FM:** similar to the case of laboratory experiments with human subjects, the main issue is the selection of the participants, i.e., the respondents to the survey. FM experts are an inherently limited population, and each expert is specialised in a limited number of methods or tools. In practice, random sampling is often not practicable, and one needs to recruit as many subjects as possible, thus resorting to the so-called *convenience sampling*. Furthermore, the actual population of FM users, which could be the target of a survey about an FM tool, or about FM adoption in general, cannot be known in advance. Therefore, reasonable assumptions and arguments need to be provided to show that the sample of respondents is actually representative of a certain target population. The FM domain also uses technical jargon, which could make questions and answers not sufficiently clear to a sufficiently wide range of potential respondents. Therefore, in some cases the researchers are constrained to ask only general questions, which however limit the degree of insight that one can achieve.
- **Typical threats to validity:** the main threats to validity are associated with *construct validity*, which in this case can be actually measured by using different survey questions to measure the same construct, and then performing an inter-item correlation analysis [87]. This allows the researcher to discard some items related to a certain construct of interest, because the responses do not appear to be correlated with other items associated with the same construct, or because they are not sufficiently discriminative with respect to other items measuring different constructs. In principle, survey research distinguishes between *validity* (criterion, face, content, and construct) and *reliability* [106]. Here, we use the term *internal validity*, to account for the different validity types, in order to make the explanation more intuitive and consistent with respect to the other strategies described. Threats to internal validity concern the way in which the questionnaire is formulated, which could be leading to preferred answers (e.g., all the first answers are checked in a long list of options), and that, if too long, could lead to fatigue effects. The first issue is addressed by shuffling answers between respondents. The second one could be addressed by reducing the length of the questionnaire, and also by shuffling the questions between respondents, so that fatigue effects are compensated. Internal validity can also be affected by systematic response bias, especially in case Likert scales are used. This can occur when similar

questions to measure the same construct are always presented in the same affirmative form. The respondent may simply check the same answer, since the questions look similar. To prevent this, the opposing question format is used, in which the same question is asked in positive and negative form. Shuffling the questions also helps at this regard. In survey research, *external validity* should be maximised, as one wants to collect information about an entire population. Claims about the appropriateness of the sample size should be included to support external validity. An additional threat, typical of surveys, concerns *reliability*, which is to what extent similar results in terms of distribution are obtained if the survey is repeated with a different sample on the same population. In practical scenarios, this means that the questions should trigger the same answers if asked to similar respondents, and can be addressed by verifying that the questions are sufficiently clear by piloting the questionnaire with a subset of respondents.

- **Maturity in FM:** while not particularly mature in FM, some seminal surveys exist. The first systematic survey of the use of FM in the development of industrial applications was conducted by Craigen et al. [107]. This extensive survey is based on twelve 'case studies' from industry and it was widely publicized [108–110]. One of these case studies is also reported in the classical survey on FM by Clarke, Wing et al. [111], together with other 'case studies' in specification and verification. The comprehensive survey on FM by Woodcock et al. [112] reviews the application of formal methods in no less than 62 different industrial projects world-wide. Basile et al. [113] and Ter Beek et al. [114] conducted a survey with FM industrial practitioners from the railway domain, aimed at identifying the main requirements of FM for the railway industry. Finally, Garavel et al. [25] conducted a survey on the past, present and future of FM in research, industry and education among a selection of internationally renowned FM experts, while Gleirscher and Marmsoler [115] conducted a survey on the academic and industrial use of FM in safety-critical software domains among FM professionals from Europe and North America.
- **Pointers to external guidelines:** Guidelines and suggestions specific to the software engineering domain are: the introductory technical report by Linåker et al. [116]; the comprehensive article, especially covering survey *design*, by Kitchenham and Pfleeger [117]; the article by Wagner et al. [118], which has a primary focus on *data analysis* strategies, and related challenges; the checklist by Molleri et al. [119]; the guidelines specific to *sampling* in software engineering, by Baltes and Ralph [120], especially concerning cases in which probabilistic sampling is hardly applicable; the guidelines by Ralph and Tempero about threats to construct validity [121]. More general textbooks on survey research are: the introductory textbook from Rea and Parker [102], covering all the relevant topics in an accessible way; the technical book by Heeringa et al. [122], specific for data analysis; the extensive book on categorical data analysis by Agresti [123], also covering topics that go beyond survey research in a technical, yet accessible way, and including several examples. For SEM, a primary reference is the book by Kline "Principles and Practice of Structural Equation Modeling" [105].
- **Pointers to studies outside FM:** a reference survey, oriented to uncover pain points in requirements engineering, involving several companies across the globe, is the NaPiRE (Naming the Pain in Requirements Engineering) initiative (http://www.re-survey.org/#/home, accessed on 20 September 2022). The results of this family of surveys have been published by Méndez-Fernández et al. [124]. Another recent and rigorous survey, using SEM, is the one by Ralph et al. [125], on the effects of COVID-19 on developers' work. A survey using hypothesis testing based on multiple regression models is the one by Chou and Kao [126], about critical factors on agile software processes. Finally, a survey about modelling practices, also using hypothesis testing but with different types of tests, is the work by Torchiano et al. [127].

7. Qualitative Studies

- **Definition of the Strategy:** qualitative studies aim at collecting qualitative data by means of interviews, focus groups, workshops, observations, documentation inspection or other qualitative data collection strategies [128], and systematically analyse these data. These studies aim at *inducing* theories about constructs, based on the analysis of the data. Constructs and RQs can be defined beforehand, or—less frequently in FM—can emerge from the data themselves. Qualitative studies are typically used when the constructs of interest are abstract, conceptual and hardly measurable (e.g., human factors, social aspects, viewpoints, practices). Qualitative studies include the general framework of Grounded Theory (GT) [129–132], which has recently been specialised for the analysis of socio-technical systems [133].
- **Crucial characteristics:** qualitative studies typically start with general RQs about abstract constructs, and perform *iterations* of data collection and analysis to provide answers to the general RQs. Like surveys, qualitative studies require *sampling* of subjects or objects from which data is collected, while with surveys it is typical to resort to probabilistic sampling, with qualitative studies *purposive sampling* is typically used. With purposive sampling, given the RQ, the researcher samples *strategically*, by selecting the units that, in the given context, are the most appropriate to give different internal perspectives to come to a (locally) complete view and answer the RQ. In qualitative studies, the fundamental characteristic is the qualitative nature of the data analysed, in most of the cases sentences produced by human subjects during interviews. For example, one could formulate a general RQ: *What are the human factors that characterise the understanding of the notation N?*, with associated constructs (i.e., *human factors*, *understanding*). Typically, RQs in qualitative studies, and in particular in GT, are *why* and *how* questions [133], which are oriented to investigate the meaning of the analysed situations. However, *what* questions are also common, especially to induce descriptive theories. Furthermore, in the aforementioned RQ *human factors* and *understanding* are *classes*, not constructs, as in qualitative studies one often aims at providing classifications rather than measuring properties [121]. In this paper, we treat them as constructs to facilitate the reader in making analogies with the more quantitative strategies: while in quantitative studies one measures properties related to the constructs of interest, in qualitative studies one typically creates a classification related to the constructs. Coming back to our RQ, one can answer it by acquiring information through interviews involving novice users of the notation. The interview transcripts will be the qualitative data to be analysed. Data analysis is carried out by means of so-called *thematic analysis* [134,135]. Thematic analysis aims to identify concepts and relations thereof, based on the interpretation of the data. Thematic analysis makes use of *coding*, which means associating essence-capturing labels (called 'codes') to relevant chunks of data (e.g., sentences, paragraphs, lines). The codes represent *concepts*, which are then aggregated into *categories*, which in turn can be aggregated and linked to one another. It should be noted that different terminology is used by different researchers and schools of thought in GT. The terminology used here (e.g., concepts, categories) generally follows from Strauss and Corbin [130]. Later, we refer to coding families, which comes from Glaser [131]. For example, one interviewee may say "*I often suffer from fatigue when reading large diagrams*". This can be coded with the labels *fatigue*, *read*, *large diagrams*. Another interviewee may say: "*I find it hard to memorise all the types of graphical constructs, they are too many, and this makes it frustrating to read the diagrams*". The labels could be: *construct types*, *read*, *frustration*. The concepts *fatigue* and *frustration* could then be aggregated into the more general category, coded as *feelings*. Once concepts and categories are identified, one can identify relationships (hierarchical, causal, similarity, etc.), between concepts and categories. The process is iterative, and through the iterations some concepts and categories may be added or removed. The iterations need to resort to *memos* and *constant comparison*. Memos are notes that the researcher writes to justify codes, reflect on possible relations between

concepts/categories, or about the analysis process itself. Constant comparison means comparing the emerging graph of concepts and categories with the data, so that there is clear evidence of the link between the more abstract categorisation and the data. The process of data collection and analysis is normally carried out until *saturation* is reached, i.e., until no further information appears to emerge from the collection and analysis of novel data. The theory, which answers the RQ, is represented by the conceptualisation that emerges from the data. In our example, the theory is a graph of all human factors affecting different dimensions of understanding different aspects of the notation N. The theory can also be represented by means of different classical coding families, which are typical patterns of concepts and relations thereof [129,131]. This general procedure, which we refer to as *thematic analysis*, is applied as part of the general framework of GT. With GT, data collection and analysis are executed as intertwined and iterative activities, and one applies so-called theoretical sampling to identify subjects to interview or objects to analyse based on the theory that has emerged from the data so far. Here, we do not discuss the GT framework, but we recommend the reader interested in qualitative studies to refer to the guidelines of Hoda [133], which are defined for the software engineering field, and can apply also to FM cases.

- **Weaknesses/Difficulties in FM:** An inherent difficulty in applying qualitative research methods in FM is the type of skills and attitude required from a qualitative researcher, which typically takes a constructivist (humans *construct* knowledge through interaction with the environment) rather than a positivist stance (humans *discover* knowledge through logical deduction from observation) in developing their research. This means that while FM practitioners search for proofs, and assume that mathematical objectivity can, and shall be achieved, qualitative research takes subjectivity and contradiction as intrinsic characteristics of reality. For this reason, the type of profile that is fit to do qualitative research in FM, and therefore has FM competence plus a constructivist mindset, is rare. Another difficulty in FM is the limited application of FM in industrial fields. Qualitative studies in FM should be based on interviews and observations of subjects practicing FM in real-world settings. Since these subjects are limited, one needs to resort to observations and interviews in research contexts, where FM are practiced, tools are developed and interaction with industrial partners takes place. These studies should complement the viewpoint of researchers with that of industrial partners, in order to have a complete, possible contrasting view of the subject matter.
- **Typical threats to validity:** threats to validity in qualitative studies can hardly be categorised according to the classes presented in Section 2. Other validity criteria shall be fulfilled, and different categorisations are provided in the literature; cf., e.g., Guba and Lincoln [136] (*trustworthiness* and *authenticity*) vs. Charmaz [132] (*credibility*, *originality*, *resonance* and *usefulness*) vs. Leung [137] (*validity*, *reliability* and *generalizability*). We refer to Charmaz and Thornberg for a discussion on the topic [138]. Regardless of the type of classification selected, the researcher should ensure that four main practices are followed, which are oriented to ensure that, despite the inherent subjectivity of qualitative research, interpretations are sound and reasonable: (i) clearly report the method adopted for data analysis, with at least one complete example that describes how the researcher passed from data to concepts, categories and relations thereof; (ii) in the results section, report quotes that exemplify concepts/categories and relations thereof; (iii) perform member checking/respondent validation: the researcher needs to (a) agree with the participants that what is transcribed and reported is actually what was meant by the participants; and (b) show the findings to the participants to understand to what extent these are accepted and considered reasonable; and (iv) perform triangulation: this means looking into multiple data sources (e.g., interviews, observations, documents) to corroborate the findings and involving more than one subject

in the data analysis. A reference set of steps for structured triangulation between different analysts is reported in the guidelines by Cruzes [135].

- **Maturity in FM:** qualitative studies are not mature at all. We are only aware of [139], where Snook and Harrison report on five structured interviews, lasting around two hours each, conducted with FM practitioners from different companies, all with some experience of using of FM in real systems (e.g., B, Z, VDM, CCS, CSP, refinement, model checking, and theorem proving). They discuss the impact of FM on the company, its products, and its development processes, as well as their scalability, understandability, and tool support. It is worth mentioning that for the aforementioned systematic survey by Craigen et al. [107,110], the authors conducted 23 interviews involving about 50 individuals in both North America and Europe, lasting from half an hour to 11 h. Moreover, the authors of [140] mention that they interviewed FM practitioners, sponsors, and other technology stakeholders in an informal manner.
- **Pointers to external guidelines:** guidelines for conducting interviews are provided in the book *Social Research Methods* by Bryman [141], which also contains a comprehensive introductory manual with a relevant part on qualitative methods, including GT. For observational studies—which belong to the field of *ethnography* [142]—a relevant reference is Zhang [143]. A primary reference for coding is the book of Saldaña [144]. For GT, one can refer to the already cited article of Hoda [133]—which will be followed by an upcoming manual in the form of a book—and to the guidelines of Stol [145].
- **Pointers to papers outside FM:** Examples of qualitative studies based on interviews are: the one by Ågren et al. [146], studying the interplay between requirements and development speed in the context of a multi-case study; Yang et al. [147], on the use of exectution logs in software development; Strandberg et al. [148], on the information flow in software testing. Example studies using GT in software engineering are: Masood et al. [149], about the difference between Scrum by the book and Scrum in practice; Leite et al. [150], on the organisation of software teams in DevOps contexts.

8. Judgement Studies

- **Definition of the strategy:** a judgement study is a research strategy in which the researcher selects experts on a certain topic and aims to elicit opinions around a set of questions, possibly triggered by some hands-on experience, with the goal of reaching consensus among the experts.
- **Crucial characteristics:** in judgement studies, RQs cover aspects that require specific expertise to be answered, and for which a survey may not provide sufficient insight, or for which research is not sufficiently mature, like, e.g., *What are the main problems of applying FM in industry? In which way can the use of tool T improve the identification of design issues?* In judgement studies, the researcher typically selects a sample of subjects that are are considered *experts* on the topic of interest. The results of a judgement study can be used to drive the design of questionnaires to later be elaborated into surveys. For instance, once one has identified the typical problems of FM in industry, these problems can be presented as possible options to a larger set of participants. Data collection is typically qualitative, and it is performed by means of *focus groups*, brainstorming *workshops*, or *Delphi studies*. With *focus groups*, the experts (normally 8 to 10) participate in a synchronous meeting in which they are asked to provide their viewpoint on a topic of interest. Before the meeting, a moderator and a note taker are initially appointed, and recording, possibly with video, is set up. Before the focus group, the experts can be faced with a reflection-triggering task, for example observing a model of a system, playing with a tool interface or a more complex task (e.g., designing a model with tool T). This latter case typically occurs when one wants to evaluate a certain tool involving the opinion of multiple experts, e.g., building on top of the DESMET project methodology [15]. During the focus group, general warm-up questions are asked, also to elicit the expertise of each expert (e.g., *In which projects did you use FM in industry?*) followed by more specific questions (e.g., *What*

could be the difficulties of using tool T in industry?) and closing with a question for final remarks (e.g., *Do you have something to add?*), after a summary. During focus groups, it is recommended to have a whiteboard, in which the moderator reports the results of the discussion so far. The moderator should make sure that all participants express their opinion, and that consensus is eventually reached—or, if not, contrasting opinions are clearly stated and agreed. Focus groups typically last one hour. If needed, multiple focus groups can be organised in parallel, and participants share the final findings in a plenary meeting. Focus groups can be carried out following the Nominal Group Technique (NGT) designed by Delbecq and Van de Ven [151]. *Workshops* are meetings that include between 10 and 30 participants, and are similar to focus groups in terms of their goal, i.e., brainstorming opinions and reaching consensus. The main difference is that workshops typically address more general questions, and can include different types of experts, with different degrees of expertise, whereas focus groups consist of more homogeneous participants focused on a more specific topic. Workshops can be carried out through adaptations of the NGT technique [151], in which each participant answers a general question using one or more sticky notes (e.g., *What are the problems of applying FM in industry?*). Then the sticky notes are read out loud, explained, attached to a whiteboard by the participants, iteratively grouped and prioritised. In focus groups and workshops, a crucial role is played by the moderator, who needs to ensure that none of the participants overtakes the meeting, and that all participants are able to express their viewpoint. With *Delphi studies*, a large number of experts is normally involved with respect to other methods (i.e., more than 30 subjects) and for longer periods of time (weeks to months), and the goal is to identify best practices or define procedures, aiming also at quantitatively measuring the consensus. The selected experts are individually asked to express their opinions around a certain problem or question, normally in written form and anonymously. The opinions are then shared with the other participants, and discussion takes place in order to reach consensus, similar to a paper reviewing process. The process typically takes place asynchronously. However, in practice, the discussion can also be carried out by means of a dedicated focus group, depending on the goal and the complexity of the RQs. With Delphi studies, multiple rounds of iterations are carried out to reach consensus. The initial round normally consists of an open question oriented to define the items to be discussed in later rounds (e.g., *What are the best practices for introducing FM in industry?*). The second round can give ratings of relevance or agreement to the different items that have been identified collectively (e.g., *How relevant is it to have an internal contact person having some knowledge of modelling?*). Therefore, in this round, quantitative answers are collected. In a third round, participants can re-evaluate their opinions based on the average results of the group. Therefore, in the end, consensus about, e.g., relevance or agreement, can be measured quantitatively. Focus groups, workshops and initial rounds of Delphi studies typically produce qualitative data. Data analysis in all these cases is carried out with thematic analysis, as described in Section 7. Quantitative analysis in Delphi studies aims at establishing that about 75% consensus is reached about the identified items [152].

- **Weaknesses/Difficulties in FM:** no particular difficulties characterise judgement studies in FM, which should therefore actually be encouraged as limited experts are typically available on certain techniques or tools, and the issues under discussion are normally particularly complex, e.g., industrial acceptance or scalability of FM tools. One practical difficulty arises with focus groups and workshops in which one needs to record many different voices, and it is not always easy to reconstruct the event from voice recordings only. Furthermore, poor equipment can make it difficult to record all the voices in a room. Video recording can address part of these issues, together with extensive note taking and transcriptions made early after the meeting.
- **Typical threats to validity:** an inherent threat to validity of judgement studies is the limited generalisability across subjects, given the limited sample, which affects *external*

validity. However, an accurate and extensive selection of experts on the topic of interest can improve external validity by allowing generalisability across responses [12]. Other threats are related to *internal validity*, since the results of the study may be biased by dominant, disruptive or reluctant behaviour of participants. These issues can be addressed by the moderator and by ensuring balanced protocols for participation. Concerning *construct validity*, the main issue resides in the communication of questions, and the definition of a shared terminology. Piloting the study, and defining a common vocabulary beforehand can mitigate this issue. As for data analysis, typical threats of qualitative studies apply here.

- **Maturity in FM:** we are aware of only one judgement study on FM. In [6], nine different FM tools are analysed by 17 experts with experience in FM applied to railway systems. The study identifies specific strengths and weaknesses of the tools and characterises them by their suitability in specific development contexts.
- **Pointers to external guidelines:** for focus groups, the book by Grueger and Casey [153] is a primary reference, while, for a quicker tour on this methodology, one should refer to Breen [154]. A reflection on focus groups for software enigneering is reported by Kontio et al. [155]. For Delphi studies, the initial guidelines have been proposed by Dalkey and Helmer [156], but over 20 variants exist [157]. For the most commonly used guidelines, we recommend to refer to the survey by Varndell et al. [158]. For the NGT technique, which can be seen as a hybrid between focus groups and Delphi, the reader can refer to the original article [151], to the comparative study between NGT and Delphi by McMillan et al. [159] or to the simple guidelines by Dunham [160]. A good overview of different, group-based, brainstorming techniques is reported by Shestopalov [161].
- **Pointers to papers outside FM:** Delphi studies are not common in FM and not as commo in software engineering research as they are in healthcare and social sciences. In software engineering, Delphi studies have been used for software cost estimation, since their introduction as one of the most suitable techniques in [162], as well as for the identification of the most important skills required to excel in the software engineering industry [163–165]. Another good example using the Delphi method is reported by Murphy et al. [166], in the field of emergency nursing. An example of usage of the NGT technique is presented by Harvey et al. [167]. Focus groups are more frequently used in software engineering, typically to involve industrial participants in the validation of prototypical solutions, cf., e.g., Abbas et al. [168]. An example of a focus group study in software engineering, carried out via online tools, is presented by Martakis and Daneva [169]. An example of a combination of in-person focus groups and workshops in requirements engineering is presented by De Angelis et al. [170].

9. Case Studies, Action Research, and Design Science

- **Definition of the Strategy:** a case study is an empirical inquiry about a certain phenomenon carried out in a real-world context, in which it is difficult to isolate the studied phenomenon from the environment in which it occurs. In the software engineering and FM literature, the term 'case study' is frequently misused, as it often refers to retrospective experience reports with lessons learned, or to exemplary applications of a technique on a specific case [171]. In principle, the researcher does not take an active role in the phenomenon under investigation. When the researcher develops an artefact—tool or method—and applies it to a real-world context, one should design the study as Action Research [172] or Design Science [173]. However, the term *case study* is extremely common, and has established guidelines for reporting [174]. These guidelines are generally applicable also to those cases in which the researcher develops an artefact, applies it to data or people belonging to one or more companies, and possibly refines the artefact based on the feedback acquired through multiple iterations. Therefore, in this paper we will discuss only case study research as unifying

framework, including also those cases in which the researcher actively intervenes in the context, as is common in software engineering and FM.

- **Crucial characteristics:** a crucial characteristic of a case study is the extensive characterisation of the *context* in which the investigation takes place, typically one or more companies or organisations—when more than one are considered, we speak about multiple case study. The researcher needs to clearly specify what is the process typically followed by the company, what are the documents produced, who are the actors involved, which are the tools used to support the process and other salient characteristics. Then, one needs to make explicit the *unit(s) of analysis*, i.e., the case being investigated. The unit can be the entire company, a team, a project, a document, etc., or sets thereof, in case the researcher wants to perform a comparison between different units. Furthermore, one needs to characterise the *subjects* involved in the research, their profile, as well as the *objects*, e.g., documents or artefacts. As for other research inquiries, a case study starts from the RQs. This is also what mainly differentiates a case study from an experience report, which is typically a retrospective reflection, and it is not guided by explicit RQs. In case studies, RQs typically start from the needs of the company in which the study is carried out. The RQs of a case study can include questions related to the application of a certain artefact, e.g., *What is the applicability of tool'T?* with sub-questions: *To what extent can we reduce the bugs by using tool T? To what extent is the performance of tool T considered acceptable by practitioners?* In other cases, the RQs can be related to understand the process, e.g., *What is the process of adoption of FM in the company?* or *What is the process of V&V through FM?* As one can see, RQs in case studies can include both qualitative and quantitative aspects, and therefore quantitative and qualitative approaches are used for data collection and analysis, to answer the RQs. For example, to answer a general RQ such as *What is the applicability of FM in company C?* and associated sub-questions outlined above, one can first measure the performance in terms of bug reduction ensured by the application of FM (*quantitative*) and then interview practitioners to understand if these measures are acceptable (*qualitative*). More specifically, one can first observe the number of bugs in one or more projects carried out without FM, and compare this number with similar projects in which FM are applied—always providing an extensive account of the characteristics of the projects. To understand the perception of practitioners, one can interview them after they have experienced the usage of FM in the projects. Overall, these different types of data contribute to give an answer to the initial RQ. Techniques such as laboratory experiments, usability studies, surveys, qualitative studies and judgement studies can be carried out in the context of case studies. However, one needs to consider the limited data points normally available in case studies, and reasonably adapt the available techniques, applying their principles rather than their full prescriptions, and reduce expectations about generality of the findings.
- **Weaknesses/Difficulties in FM:** a case study requires active participation from industry, and industrial skepticism is the main issue that FM researchers need to face. Even when industrial partners are willing to be involved, researchers need to spend a considerable time understanding the technical application domain and the characteristics of the projects, so that, e.g., a formal model of a product or component can be designed, and the expected properties can be correctly stated. In principle, the researcher should not take part in the application of FM, but this is often impossible, given the complexity and limited maturity of the interface of many a tool. Therefore, it is common to have an interaction scheme in which the researcher develops and verifies formal models, with the support of a main technical contact point from the company, who ensures that the models are faithful to the real system. This communication loop however, should always be completed with a larger involvement of practitioners, who need to assess and confirm a larger applicability of FM in the company, and in general provide some form of structured feedback, e.g., via surveys or interviews. In practice, it is also hard to measure some possibly relevant variables, such as the

learning curve of FM and actual bug reduction, throughout projects that can last for years and have several dimensions of complexity (process phases, change of people involved, budget, time constraint, etc.). To address this, it is acceptable to use data (e.g, requirements) from previous projects as a main source, and to consider substantial portions of products, instead of entire ones. The researcher somehow sacrifices the realism of the case study in favour of something more manageable, keeping in mind that one should always involve practitioners in a structured reflection about possible consequences of the observed case in a real-world environment. Finally, one needs to consider that case studies deal with proprietary data, which companies are typically not inclined to disclose. This problem can be addressed by providing convincing examples and portions of data, and by setting the terms of the collaboration with the company beforehand, including intellectual property management that clearly states what can be published.

- **Typical threats to validity:** case study research is typically evaluated according to *construct*, *internal* and *external* validity. Other types of validity, e.g., from qualitative studies or laboratory experiments are considered, when these strategies are used in the context of case studies. *Construct validity* is concerned with the general constructs and associated measurement instruments used in the study. The researcher should comment on their soundness. *Internal validity* is typically concerned with the list of contextual factors that could have impact on the results. As the characteristics of the subjects and objects involved are specific to the case, and since context and phenomena are hard to be separated in a case study, one needs to provide reasonable mitigation measures to reach some form of objectivity. For example, considering more than one single product or component, and involving multiple practitioners, also with multiple roles and competences. *External validity* is also inherently limited in case studies. However, one should report and discuss what are the dimensions of the case that could make its results generalisable to other contexts, according to the principles of case-based generalisation [175]. For example, safety-critical companies follow highly structured and comparable processes, and railway and avionics are basically oligopolies, following very similar practices. Therefore, a case study in one domain could inform a company in a similar domain, having a comparable degree of maturity. The possibility to generalise, especially based on similarity, drives the need to provide an extensive account of the company or organisational context in a case study paper.
- **Maturity in FM:** not surprisingly, given the difficulties mentioned earlier, case studies and the like are not very mature in FM. However, next to the aforementioned case study by Pfleeger and Hatton [40], who investigated the effects of using FM in an industrial setting in which professionals developed an air-traffic-control information system, there are some examples of case studies developed by academics in close collaboration with practitioners—and also partially carried out inside the companies they work for [176,177]. In particular the railway domain contains a fair number of case studies on applying FM [178–182], among which one of the best known success stories of applying FM in industry [183].
- **Pointers to external guidelines:** the primary reference for case study research is the book by Runeson [174], including also several examples. The reference for action research is the recent book by Staron [172]. We recommend referring to action research when the goal is the actual transformation in the company—e.g., through the introduction of an FM tool—and both researchers and practitioners are active in this transformation, e.g., the former as instructors and the latter as trainees. Finally, for design science, one can refer to the books of Wieringa [173] and Johannesson and Perjons [184]. We recommend referring to design science guidelines when the focus is on the FM tool or interface to be developed.
- **Pointers to papers outside FM:** an example of a case study in the strict sense, i.e., without intervention of the researcher, is the one by Britto et al. [185]. The study

focuses on on-boarding of software developers and, as is common, here the unit of analysis is a single company. A multiple case study, concerning the usage of DevOps in five companies, is presented by Lwakatare et al. [186]. Another reference case is reported by Tomasdottir et al. [187]. The study is about the usage of a static analysis tool by open-source developers. The case is interesting as the unit of analysis is a tool and not a company, and multiple data sources are used, as required by case study guidelines. An iterative case study, in which the researcher performs limited intervention, is presented by Ferrari et al. [188]. The study concerns the incremental development of a natural language processing tool for defect detection in requirements documents. An example paper following action research guidelines is [189], about the application of a machine-learning technique to detect violation of coding guidelines. Finally, for a reference using the design science paradigm, the reader can refer to Manzano et al. [190].

10. Systematic Literature Reviews and Mapping Studies

- **Definition of the strategy:** Systematic Literature Reviews (SLRs) and Systematic Mapping Studies (SMSs) are secondary studies (i.e., analysing other empirical research papers, i.e., primary studies) systematically conducted using search engines of digital libraries. These studies are oriented to ensure *completeness*, in terms of surveyed literature, and *replicability*, thus following well-defined guidelines, and carefully reporting the study protocols. SLRs are typically oriented to survey and summarise *findings* from previous research (e.g., identify reported industrial problems with model-checking tools). SMSs mainly aim to scope a research field, identifying relevant dimensions and categorising the literature accordingly (e.g., summarise the literature about theorem proving for safety-critical systems). However, these studies follow similar guidelines and protocols, and SLRs typically include also a categorisation of existing studies, as do SMSs.
- **Crucial characteristics:** the main RQ of an SMS typically aims to identify the relevant dimensions of a certain field, e.g., *What are the relevant dimensions characterising the literature about FM in avionics?* The RQ is then decomposed into RQs about the demographic distribution of the studies (years, venues, etc.), the type of studies (case study, surveys, lab experiments, etc.), and other aspects, e.g, *What techniques are used?* and *What tools are used?* These questions are typically answered quantitatively, providing statistics about the frequency distribution of the studies. Besides these RQs, which are shared with SMSs, SLRs also provide more in-depth analyses, answering RQs related to the effect of a technology (*What is the effect on the avionic process of introducing model checking?*), its cost (*What is the cost of introducing model checking in avionics?*), its performance (*What is the performance of model-checking tools in avionics?*) and other aspects that are considered relevant. Answering these questions can require qualitative analyses, conducted using thematic analysis and coding (cf. Section 7). An SMS/SLR needs to be justified, i.e., one should provide an account of related reviews that do not address the same RQs, or do not address them systematically (e.g., systematic studies exist on FM for railways, but not for avionics; studies exist on FM for avionics, but are not systematic, or are outdated). Therefore, typical constructs of SMSs/SLRs are the salient characteristics that the researcher wants to extract from the existing literature (e.g., demographic information, type of study, performance, etc.). SLRs/SMSs start from a search string to be used as input for the search engines of specialised digital libraries, which for computer science are IEEE eXplore, Scopus, Web of Science, SpringerLink and ACM Digital Library. The search string is composed of terms that are relevant to the main RQ, connected with AND/OR logical operators. The search conducted via the search engines is called *primary search*. The search string should be able to identify as many relevant studies as possible, and should limit the irrelevant studies retrieved. To this end, pilot searches need to be performed, possibly adding words to the string, if the researcher identifies terminological variations that are typ-

ically used in the literature and enable a more focused search. After retrieving the studies, the researcher needs to select them according to a set of inclusion/exclusion criteria (e.g., removing short papers, removing other secondary studies, removing low quality studies), also removing duplicates that can emerge since multiple search engines are used. The selection activity is typically based on a screening performed on titles and abstracts. To improve replicability, this activity is conducted in parallel by multiple researchers, and disagreement in the selection is resolved through dedicated meetings. Throughout the activities, it is important to always keep track of the number of studies retrieved and selected, and all the search strings used, as different variations of the string may be needed depending on the search engine. To facilitate the removal of duplicates, and the tracing of the whole process, specialised tools like, e.g., Zotero (https://www.zotero.org/, accessed on 20 September 2022) or Mendeley (https://www.mendeley.com/, accessed on 20 September 2022), can be used for support. After the selection, researchers can perform a quality assessment of the studies, according to a dedicated rubric to be defined beforehand. This can be carried out to further limit the number of included studies, to report about their quality, or to answer a subset of the questions considering only high-quality studies. Once the studies have been selected, one performs a *secondary search* to ensure completeness. This means performing forward/backward snowballing (i.e., looking for cited papers, and citing papers, e.g., through Google Scholar), analysing the literature of prominent authors or screening the papers from relevant venues. After the paper selection process has been completed, the researchers extract information according to extraction schemes. These are defined based on the RQs and, in some cases, already outline the predefined options (e.g., the extraction scheme for 'types of studies' would have 'case study', 'survey', etc. as options). When the options are not clear, one initially extracts relevant text from the paper and then identifies the options after thematic analysis. As for the study selection task, this activity should be performed in parallel by multiple subjects. The results of SMSs/SLRs are reported in the form of plots (for frequency-related RQs) and tables (for qualitative RQs). An SMS/SLR is not a mere summary of the literature. It crucial to provide a contribution based on focuses and gaps in the literature, thereby illustrating further avenues for research and providing recommendations to the community.

- **Weaknesses/Difficulties in FM:** SMSs/SLRs in FM share the typical difficulties of other fields like, e.g., software engineering [191,192]. These include the limitations of search engines of digital libraries, and the difficulty of assessing relevance from the title and abstract alone. Additional complexity dimensions are related to the *publication bias*, i.e., certain evidence is not published in scientific venues. To address this issue, multivocal literature reviews with systematic retrieval of *grey literature*, which includes non-peer reviewed articles, such as blog posts, websites, news articles, white papers and similar is recommended [193,194]. Another difficulty, also raised in different points of this paper, is the difficulty of having expertise in multiple FM, which limits the ability to fully understand papers that are within the scope of the RQs, but that cover topics that the researchers do not know in detail. This can be addressed with the involvement of researchers with multiple backgrounds. The reporting of empirical studies in FM does not follow a consistent/standardised structure, and this make it difficult to compare and assess them. The incorrect naming of research strategies, typically 'case study' to refer to examples, is also complicating the evaluation, as different quality criteria and extraction schemes are used depending on the study type.
- **Typical threats to validity:** threats to validity in SMSs/SLRs are partitioned into *study selection validity*, *data validity* and *research validity*. Study selection validity concerns threats in search and filtering, and includes bias in search string construction, the selection of digital libraries and primary studies. Typical mitigations are strategies for cross-checking and study selection involving multiple researches, best effort to achieve completeness through secondary searches and appropriate justification for

the selection of digital libraries. Data validity concerns threats in data extraction and analysis, and include publication bias (i.e., evidence may exist but it is not published in scientific venues), bias in data extraction schemes and subjectivity in classification and extraction. Mitigation strategies are the additional search for grey literature, the reuse of schemes adopted by previous studies, and again, cross-checking involving multiple researchers. Research validity is concerned with threats to the design as a whole, including coverage of the RQs, replicability and generalisability. The threats can be mitigated by sharing the protocol, clarifying the gap with previous reviews and performing a broad search (e.g., without a starting date for the search time interval).

- **Maturity in FM:** maturity of SMSs and SLRs is increasing, as witnessed by two SMSs and two SLRs published this year. As far as we know, the work by Ferrari and Ter Beek [195] is the first SMS on FM focusing on applications of FM in the railway domain. It considers 328 high-quality studies published during the last 30 years, which are classified according to their empirical maturity, the types of FM applied and railway specific aspects, identifying recent trends and the characteristics of the studies involving practitioners. Moreover, the work by Filho et al. [196] is the first SLR on the use of FM to evaluate security and energy in IoT, including wireless sensor networks, which can be used as a guide for which FM and tools to use. It considers 38 high-quality studies from a period of 15 years and the findings include a clear predominance of the use of manual proof methods, Dolev-Yao-like attack models, and the AVISPA tool [197]. Furthermore, the work by Mishra and Mustafa [198] is claimed to be the first SLR on FM focusing on the security requirement specification. It considers 88 studies from the last 20 years and it is observed that model checking is preferred over theorem proving, while it remains a research challenge to effectively use FM in a cost-effective and time-saving way. Finally, as far as we know, the work by Zahid et al. [199] is the first SMS on the use of FM, including semi-formal methods, during the requirements engineering of industrial cyber–physical systems. It considers 93 studies from the last decade and it is shown that safety and timing requirements have been extensively analysed and verified, but there is a lack of work on key phases like requirements elicitation and management, while also the adoption of industrial standards is largely missing and so are methods to handle the currently critical concerns of privacy and trust requirements. Additionally, worth mentioning are some earlier studies, in particular in specific application domains [200–204], as well as on teaching FM [205].
- **Pointers to external guidelines:** Kitchenham [206] provides the primary reference for conducting SLRs in software engineering and the guidelines are generally appropriate also for SMSs. For SMSs, however, the guidelines from Petersen [207,208] are a valid alternative. For guidelines on how to select and assess an 'optimal' search string, the interested reader can refer to Zhang [209]. For guidelines on how to perform snowballing, the reader should refer to Wholin [210]. For multivocal literature reviews to include grey literature, the reader should refer to Garousi et al. [194]. Guidelines for reporting threats to validity are made available by Ampatzoglou et al. [211]. Finally, guidelines to update SLRs are discussed by Wohlin et al. [212].
- **Pointers to papers outside FM:** a recent example of an SLR is the work by Dąbrowski et al. [213]. Another example, which also includes a quality checklist, is the one by Bano and Zowghi [214], about the practice of user involvement in software development. A good SMS, on software engineering for AI, is the one by Martinez et al. [215]. Another example, with a rigorous evaluation of agreement between researchers, is the work by Horkoff et al. [216], about goal-oriented requirements engineering. Finally, for multivocal literature reviews, a representative example is the study by Garousi et al. [217]. A more recent work, using reference guidelines [194], is the one by Sheuner et al. [218].

11. Discussion and Conclusions

Research in FM has traditionally focused on developing novel techniques, improving the performance of FM tools, and tackling ever complex problems. On the other hand, evidence-based, empirically grounded studies are currently limited. To foster a better uptake of *empirical formal methods* in the community, this paper presents a summary of the main empirical research strategies as they are defined in software engineering, and specifically adapted for their application in FM. Of course, several challenges still exist. In particular: the complexity of the theory behind FM tools, which leads to difficulties in performing usability testing or experiments with human subjects with a sufficient background; the wide differences among tools, which makes it hard to compare them on real-world benchmarks; the development status of many tools, which are not sufficiently mature to be evaluated by industrial practitioners, thus hampering case study research; the fact that FM experts are often specialised on a single tool, thereby making it hard to find sufficient subjects to perform an in-depth survey about a certain tool, and also an experiment with human subjects having comparable backgrounds. Some recommendations on how to overcome these issues in practice are presented throughout the paper. Our study represents a reference guide for researchers who want to approach a FM problem with an empirical mindset, and want to soundly evaluate FM, their tools, or systematically study FM practice and literature. Guidelines for the selection of the specific strategy to choose are not different from those of software engineering research, cf. Easterbrook et al. [11]. The choice primarily depends upon the types of research questions to be addressed, and the maturity of the knowledge (i.e., existing theories) about the constructs of interest. An informal diagram to guide the reader in the selection of the most appropriate research strategy, also considering the peculiarity of FM research, is reported in Figure 1. For example, usability testing should be considered when GUI-related aspects are the main concern. Laboratory experiments are recommended when one wants to study relationships between fine-grained, measurable variables, as, e.g., performance. Case studies, action research, and design science are appropriate when one wants to evaluate specific theories or artefacts in a real-world context. Qualitative studies are more appropriate when less knowledge is available and one wants to have a first understanding of a topic, and possibly generate more detailed research questions. Surveys can be applied after qualitative studies to have a quantitative overview about the topic of interest. Judgement studies, instead, are more suitable when the topic of interest is sufficiently defined, but either few subjects are expert on it, or one wishes to have more informative, qualitative content, with respect to the data that could be collected with a survey. Finally, systematic literature reviews and mapping studies should be performed when the researcher has limited knowledge about a topic, which has however been extensively studied in the literature. Of course, the choice of the type of strategy to adopt also depends on the access to resources, e.g., students or professionals as subjects, and their availability.

Author Contributions: Conceptualization, M.H.t.B and A.F.; methodology, A.F.; investigation, M.H.t.B. and A.F.; writing—original draft preparation, M.H.t.B. and A.F.; writing—review and editing, M.H.t.B. and A.F.; visualization, M.H.t.B. and A.F.; project administration, M.H.t.B.; funding acquisition, M.H.t.B. and A.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the MIUR-PRIN 2020TL3X8X project T-LADIES (Typeful Language Adaptation for Dynamic, Interacting and Evolving Systems).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ACM	Association for Computing Machinery
ANOVA	Analysis of Variance
ASM	Abstract State Machines
ASMETA	ASMs mETAmodelling
ATP	Automated Theorem Provers
AVISPA	Automated Validation of Internet Security Protocols and Applications
CCS	Calculus of Communicating Systems
COVID-19	Coronavirus Disease 2019
CSP	Communicating Sequential Processes
CSUQ	Computer System Usability Questionnaire
DESMET	Determining an Evaluation Methodology for Software Methods and Tools
FM	Formal Method(s)
FRAMA-C	Framework for Modular Analysis of C programs
GDPR	General Data Protection Regulation
GT	Grounded Theory
GUI	Graphical User Interface
IEEE	Institute of Electrical and Electronics Engineers
IFADIS	Integrated Framework for the Analysis of Dependable Interactive Systems
ISO	International Standards Organization
NGT	Nominal Group Technique
RQ	Research Question
RSML	Requirements State Machine Language
SAT	Boolean satisfiability
SC(R)^3	Software Cost Reduction Requirements Reuse
SEM	Structured Equation Modeling
SEQ	Single Easy Questionnaire
SLR	Systematic Literature Review
SMS	Systematic Mapping Study
SUS	System Usability Scale
TACAS	Tools and Algorithms for the Construction and Analysis of Systems
TAM	Technology Acceptance Model
TLA	Temporal Logic of Actions
VDM	Vienna Development Method
WOZ	Wizard of Oz method

References

1. Zhang, L.; Tian, J.; Jiang, J.; Liu, Y.; Pu, M.; Yue, T. Empirical Research in Software Engineering—A Literature Survey. *J. Comput. Sci. Technol.* **2018**, *33*, 876–899. [CrossRef]
2. Höfer, A.; Tichy, W.F. Status of Empirical Research in Software Engineering. In Proceedings of the International Workshop on Empirical Software Engineering Issues: Critical Assessment and Future Directions, Dagstuhl Castle, Germany, 26–30 June 2006; Basili, V.R., Rombach, H.D., Schneider, K., Kitchenham, B.A., Pfahl, D., Selby, R.W., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4336, pp. 10–19. [CrossRef]
3. Jeffery, D.R.; Staples, M.; Andronick, J.; Klein, G.; Murray, T.C. An empirical research agenda for understanding formal methods productivity. *Inf. Softw. Technol.* **2015**, *60*, 102–112. [CrossRef]
4. Gleirscher, M.; van de Pol, J.; Woodcock, J. A Manifesto for Applicable Formal Methods. 2021. Available online: https://arxiv.org/abs/2112.12758 (accessed on 22 September 2022).
5. Huisman, M.; Gurov, D.; Malkis, A. Formal Methods: From Academia to Industrial Practice. A Travel Guide. 2020. Available online: https://arxiv.org/abs/2002.07279 (accessed on 22 September 2022).
6. Ferrari, A.; Mazzanti, F.; Basile, D.; ter Beek, M.H.; Fantechi, A. Comparing Formal Tools for System Design: A Judgment Study. In Proceedings of the 42nd ACM International Conference on Software Engineering (ICSE'20), Seoul, Korea, 27 June–19 July 2020; pp. 62–74. [CrossRef]

7. Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M.C.; Regnell, B.; Wesslén, A. *Experimentation in Software Engineering*; Springer: Berlin/Heidelberg, Germany, 2012. [CrossRef]
8. Shull, F.; Singer, J.; Sjøberg, D.I.K. (Eds.) *Guide to Advanced Empirical Software Engineering*; Springer: Berlin/Heidelberg, Germany, 2008. [CrossRef]
9. Felderer, M.; Travassos, G.H. (Eds.) *Contemporary Empirical Methods in Software Engineering*; Springer: Berlin/Heidelberg, Germany, 2020. [CrossRef]
10. Kitchenham, B.A.; Pfleeger, S.L.; Pickard, L.; Jones, P.W.; Hoaglin, D.C.; Emam, K.E.; Rosenberg, J. Preliminary Guidelines for Empirical Research in Software Engineering. *IEEE Trans. Softw. Eng.* **2002**, *28*, 721–734. [CrossRef]
11. Easterbrook, S.; Singer, J.; Storey, M.D.; Damian, D.E. Selecting Empirical Methods for Software Engineering Research. In *Guide to Advanced Empirical Software Engineering*; Shull, F., Singer, J., Sjøberg, D.I.K., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 285–311. [CrossRef]
12. Stol, K.J.; Fitzgerald, B. The ABC of software engineering research. *ACM Trans. Softw. Eng. Methodol.* **2018**, *27*, 1–51. [CrossRef]
13. Ralph, P. (Ed.) Empirical Standards for Software Engineering Research. 2020. Available online: https://arxiv.org/abs/2010.03525 (accessed on 22 September 2022).
14. Box, G.E.P.; Hunter, J.S.; Hunter, W.G. *Statistics for Experimenters: Design, Innovation, and Discovery*; Wiley: Hoboken, NJ, USA, 2005.
15. Kitchenham, B.; Linkman, S.; Law, D. DESMET: A methodology for evaluating software engineering methods and tools. *Comput. Control Eng. J.* **1997**, *8*, 120–126. [CrossRef]
16. Beyer, D.; Löwe, S.; Wendler, P. Reliable benchmarking: Requirements and solutions. *Int. J. Softw. Tools Technol. Transf.* **2019**, *21*, 1–29. [CrossRef]
17. Kordon, F.; Hulin-Hubard, F. BenchKit, a Tool for Massive Concurrent Benchmarking. In Proceedings of the 14th International Conference on Application of Concurrency to System Design (ACSD'14), IEEE, Tunis La Marsa, Tunisia, 23–27 June 2014; pp. 159–165. [CrossRef]
18. Petkovich, J.C.; de Oliveira, A.B.; Zhang, Y.; Reidemeister, T.; Fischmeister, S. DataMill: A distributed heterogeneous infrastructure for robust experimentation. *Softw. Pract. Exp.* **2016**, *46*, 1411–1440. [CrossRef]
19. Stump, A.; Sutcliffe, G.; Tinelli, C. StarExec: A Cross-Community Infrastructure for Logic Solving. In Proceedings of the 7th International Joint Conference on Automated Reasoning (IJCAR'14), Vienna, Austria, 19–22 July 2014; Demri, S., Kapur, D., Weidenbach, C., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8562, pp. 367–373. [CrossRef]
20. Järvisalo, M.; Berre, D.L.; Roussel, O.; Simon, L. The International SAT Solver Competitions. *AI Mag.* **2012**, *33*, 89–92. [CrossRef]
21. Sutcliffe, G. The CADE ATP System Competition-CASC. *AI Mag.* **2016**, *37*, 99–101. [CrossRef]
22. Bartocci, E.; Beyer, D.; Black, P.E.; Fedyukovich, G.; Garavel, H.; Hartmanns, A.; Huisman, M.; Kordon, F.; Nagele, J.; Sighireanu, M.; et al. TOOLympics 2019: An Overview of Competitions in Formal Methods. In Proceedings of the 25th International Conference on Tools and Algorithms for the Construction and Analysis of Systems: TOOLympics (TACAS'19), Prague, Czech Republic, 6–7 April 2019; Beyer, D., Huisman, M., Kordon, F., Steffen, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11429, pp. 3–24. [CrossRef]
23. Krishnamurthi, S. Artifact Evaluation for Software Conferences. *ACM Sigsoft Softw. Eng. Notes* **2013**, *38*, 7–10. [CrossRef]
24. Krishnamurthi, S.; Vitek, J. The Real Software Crisis: Repeatability as a Core Value–Sharing experiences running artifact evaluation committees for five major conferences. *Commun. ACM* **2015**, *58*, 34–36. [CrossRef]
25. Garavel, H.; ter Beek, M.H.; van de Pol, J. The 2020 Expert Survey on Formal Methods. In Proceedings of the 25th International Conference on Formal Methods for Industrial Critical Systems (FMICS'20), Vienna, Austria, 2–3 September 2020; ter Beek, M.H., Ničković, D., Eds. Springer: Berlin/Heidelberg, Germany, 2020; Volume 12327, pp. 3–69. [CrossRef]
26. Vitek, J.; Kalibera, T. R^3–Repeatability, Reproducibility and Rigor. *ACM Sigplan Not.* **2012**, *47*, 30–36. [CrossRef]
27. Ye, H.; Martinez, M.; Durieux, T.; Monperrus, M. A comprehensive study of automatic program repair on the QuixBugs benchmark. *J. Syst. Softw.* **2021**, *171*, 110825. [CrossRef]
28. Su, T.; Wang, J.; Su, Z. Benchmarking automated GUI testing for Android against real-world bugs. In Proceedings of the 29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE'21), Athens, Greece, 23–28 August 2021; pp. 119–130. [CrossRef]
29. Herbold, S.; Trautsch, A.; Grabowski, J. A Comparative Study to Benchmark Cross-Project Defect Prediction Approaches. *IEEE Trans. Softw. Eng.* **2018**, *44*, 811–833. [CrossRef]
30. Falessi, D.; Cantone, G.; Canfora, G. Empirical Principles and an Industrial Case Study in Retrieving Equivalent Requirements via Natural Language Processing Techniques. *IEEE Trans. Softw. Eng.* **2013**, *39*, 18–44. [CrossRef]
31. Maalej, W.; Kurtanovic, Z.; Nabil, H.; Stanik, C. On the automatic classification of app reviews. *Requir. Eng.* **2016**, *21*, 311–331. [CrossRef]
32. Abualhaija, S.; Arora, C.; Sabetzadeh, M.; Briand, L.C.; Traynor, M. Automated demarcation of requirements in textual specifications: A machine learning-based approach. *Empir. Softw. Eng.* **2020**, *25*, 5454–5497. [CrossRef]

33. Steffen, B. The Physics of Software Tools: SWOT Analysis and Vision. *Int. J. Softw. Tools Technol. Transfer* **2017**, *19*, 1–7. [CrossRef]
34. Garavel, H.; Mateescu, R. Reflections on Bernhard Steffen's Physics of Software Tools. In *Models, Mindsets, Meta: The What, the How, and the Why Not?* Margaria, T., Graf, S., Larsen, K.G., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11200, pp. 186–207. [CrossRef]
35. Hwang, W.; Salvendy, G. Number of people required for usability evaluation: The 10 ± 2 rule. *Commun. ACM* **2010**, *53*, 130–133. [CrossRef]
36. Macefield, R. How To Specify the Participant Group Size for Usability Studies: A Practitioner's Guide. *J. Usability Stud.* **2009**, *5*, 34–45.
37. Sobel, A.E.K.; Clarkson, M.R. Formal Methods Application: An Empirical Tale of Software Development. *IEEE Trans. Softw. Eng.* **2002**, *28*, 308–320. [CrossRef]
38. Berry, D.M.; Tichy, W.F. Comments on "Formal Methods Application: An Empirical Tale of Software Development". *IEEE Trans. Softw. Eng.* **2003**, *29*, 567–571. [CrossRef]
39. Sobel, A.E.K.; Clarkson, M.R. Response to "Comments on 'Formal Methods Application: An Empirical Tale of Software Development'". *IEEE Trans. Softw. Eng.* **2003**, *29*, 572–575. [CrossRef]
40. Pfleeger, S.L.; Hatton, L. Investigating the Influence of Formal Methods. *IEEE Comput.* **1997**, *30*, 33–43. [CrossRef]
41. Moher, T.G.; Mak, D.C.; Blumenthal, B.; Leventhal, L.M. Comparing the Comprehensibility of Textual and Graphical Programs: The Case of Petri Nets. In Proceedings of the 5th Workshop on Empirical Studies of Programmers (ESP'93), Palo Alto, CA, USA, 3–5 December 1993; Cook, C.R., Scholtz, J.C., Spohrer, J.C., Eds.; Ablex: Norwood, NJ, USA, 1993; pp. 137–162.
42. Finney, K.; Rennolls, K.; Fedorec, A.M. Measuring the comprehensibility of Z specifications. *J. Syst. Softw.* **1998**, *42*, 3–15. [CrossRef]
43. Snook, C.F.; Harrison, R. Experimental comparison of the comprehensibility of a Z specification and its implementation in Java. *Inf. Softw. Technol.* **2004**, *46*, 955–971. [CrossRef]
44. Neary, D.S.; Woodward, M.R. An Experiment to Compare the Comprehensibility of Textual and Visual Forms of Algebraic Specifications. *J. Vis. Lang. Comput.* **2002**, *13*, 149–175. [CrossRef]
45. Carew, D.; Exton, C.; Buckley, J. An empirical investigation of the comprehensibility of requirements specifications. In Proceedings of the International Symposium on Empirical Software Engineering (ISESE'05), IEEE, Noosa Heads, Australia, 17–18 November 2005; pp. 256–265. [CrossRef]
46. Razali, R.; Snook, C.F.; Poppleton, M.; Garratt, P.W.; Walters, R.J. Experimental Comparison of the Comprehensibility of a UML-based Formal Specification versus a Textual One. In Proceedings of the 11th International Conference on Evaluation and Assessment in Software Engineering (EASE'07), Keele, UK, 2–3 April 2007; Kitchenham, B.A., Brereton, P., Turner, M., Eds.; BCS: Swindon, UK, 2007. [CrossRef]
47. Razali, R.; Snook, C.F.; Poppleton, M.R. Comprehensibility of UML-Based Formal Model: A Series of Controlled Experiments. In Proceedings of the 1st ACM International Workshop on Empirical Assessment of Software Engineering Languages and Technologies (WEASELTech'07), Atlanta, GA, USA, 5 November 2007; pp. 25–30. [CrossRef]
48. Zimmerman, M.K.; Lundqvist, K.; Leveson, N.G. Investigating the Readability of State-Based Formal Requirements Specification Languages. In 643Proceedings of the 24th ACM International Conference on Software Engineering (ICSE'02), Orlando, FL, USA, 19–25 May 2002; pp. 33–43. [CrossRef]
49. Sarshar, K.; Loos, P. Comparing the Control-Flow of EPC and Petri Net from the End-User Perspective. In Proceedings of the 3rd International Conference on Business Process Management (BPM'05), Nancy, France, 5–8 September 2005; van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3649, pp. 434–439. [CrossRef]
50. Mendling, J.; Reijers, H.A.; Cardoso, J. What Makes Process Models Understandable? In Proceedings of the 5th International Conference on Business Process Management (BPM'07), Brisbane, Australia, 24–28 September 2007; Alonso, G., Dadam, P., Rosemann, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4714, pp. 48–63. [CrossRef]
51. Reijers, H.A.; Mendling, J. A Study Into the Factors That Influence the Understandability of Business Process Models. *IEEE Trans. Syst. Man Cybern. Part A* **2011**, *41*, 449–462. [CrossRef]
52. Reinhartz-Berger, I.; Figl, K.; Haugen, Ø. Comprehending Feature Models Expressed in CVL. In Proceedings of the 17th International Conference on Model-Driven Engineering Languages and Systems (MoDELS'14), Valencia, Spain, 28 September–3 October 2014; Dingel, J., Schulte, W., Ramos, I., Abrahão, S., Insfrán, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8767, pp. 501–517. [CrossRef]
53. Reinhartz-Berger, I.; Sturm, A. Comprehensibility of UML-based software product line specifications—A controlled experiment. *Empir. Softw. Eng.* **2014**, *19*, 678–713. [CrossRef]
54. Labunets, K.; Massacci, F.; Paci, F.; Tran, L.M.S. An Experimental Comparison of Two Risk-Based Security Methods. In Proceedings of the IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM'13), Baltimore, MD, USA, 10–11 October 2013; pp. 163–172. [CrossRef]

55. Labunets, K.; Paci, F.; Massacci, F.; Ruprai, R.S. An Experiment on Comparing Textual vs. Visual Industrial Methods for Security Risk Assessment. In Proceedings of the 4th IEEE International Workshop on Empirical Requirements Engineering (EmpiRE'14), Karlskrona, Sweden, 25 August 2014; pp. 28–35. [CrossRef]
56. Labunets, K.; Massacci, F.; Paci, F. On the Equivalence Between Graphical and Tabular Representations for Security Risk Assessment. In Proceedings of the 23rd International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ'17), Essen, Germany, 27 February–2 March 2017; Grünbacher, P., Perini, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10153, pp. 191–208. [CrossRef]
57. Schneier, B. Attack Trees. *Dr. Dobb's J. Softw. Tools* **1999**, 24, 21–29.
58. Mauw, S.; Oostdijk, M. Foundations of Attack Trees. In Proceedings of the 8th International Conference on Information Security and Cryptology (ICISC'05), Seoul, Korea, 1–2 December 2015; Won, D., Kim, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3935, pp. 186–198. [CrossRef]
59. Kordy, B.; Mauw, S.; Radomirovic, S.; Schweitzer, P. Foundations of Attack-Defense Trees. In Proceedings of the 7th International Workshop on Formal Aspects of Security and Trust (FAST'10), Pisa, Italy, 16–17 September 2010; Degano, P., Etalle, S., Guttman, J.D., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6561, pp. 80–95. [CrossRef]
60. Amenaza Technologies Limited. *The SecuITree® BurgleHouse Tutorial (a.k.a., Who wants to be a Cat Burglar?)* 2006. Available online: https://www.amenaza.com/downloads/docs/Tutorial.pdf (accessed on 22 September 2022).
61. Gadyatskaya, O.; Trujillo-Rasua, R. New Directions in Attack Tree Research: Catching up with Industrial Needs. In Proceedings of the 4th International Workshop on Graphical Models for Security (GraMSec'17), Santa Barbara, CA, USA, 21 August 2017; Liu, P., Mauw, S., Stølen, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; Volume 10744, pp. 115–126. [CrossRef]
62. Eisentraut, J.; Holzer, S.; Klioba, K.; Kretínský, J.; Pin, L.; Wagner, A. Assessing Security of Cryptocurrencies with Attack-Defense Trees: Proof of Concept and Future Directions. In Proceedings of the 18th International Colloquium on Theoretical Aspects of Computing (ICTAC'21), Nur-Sultan, Kazakhstan, 6–10 September 2021; Cerone, A., Ölveczky, P.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 12819, pp. 214–234. [CrossRef]
63. Ko, A.J.; LaToza, T.D.; Burnett, M.M. A practical guide to controlled experiments of software engineering tools with human participants. *Empir. Softw. Eng.* **2015**, *20*, 110–141. [CrossRef]
64. Santos, A.; Vegas, S.; Oivo, M.; Juristo, N. A Procedure and Guidelines for Analyzing Groups of Software Engineering Replications. *IEEE Trans. Softw. Eng.* **2019**, *47*, 1742–1763. [CrossRef]
65. Trochim, W.M.K. The Research Methods Knowledge Base. 2022. Available online: https://conjointly.com/kb/ (accessed on 20 September 2022).
66. Graziotin, D.; Lenberg, P.; Feldt, R.; Wagner, S. Psychometrics in Behavioral Software Engineering: A Methodological Introduction with Guidelines. *ACM Trans. Softw. Eng. Methodol.* **2021**, *31*, 1–36. [CrossRef]
67. Motulsky, H. *Intuitive Biostatistics: A Nonmathematical Guide to Statistical Thinking*; Oxford University Press: Oxford, UK, 2013.
68. Furia, C.A.; Feldt, R.; Torkar, R. Bayesian Data Analysis in Empirical Software Engineering Research. *IEEE Trans. Softw. Eng.* **2019**, *47*, 1786–1810. [CrossRef]
69. Abrahão, S.; Gravino, C.; Insfrán, E.; Scanniello, G.; Tortora, G. Assessing the Effectiveness of Sequence Diagrams in the Comprehension of Functional Requirements: Results from a Family of Five Experiments. *IEEE Trans. Softw. Eng.* **2013**, *39*, 327–342. [CrossRef]
70. Abrahão, S.; Insfrán, E.; Carsí, J.A.; Genero, M. Evaluating requirements modeling methods based on user perceptions: A family of experiments. *Inf. Sci.* **2011**, *181*, 3356–3378. [CrossRef]
71. Santos, A.; Vegas, S.; Dieste, O.; Uyaguari, F.; Tosun, A.; Fucci, D.; Turhan, B.; Scanniello, G.; Romano, S.; Karac, I.; et al. A family of experiments on test-driven development. *Empir. Softw. Eng.* **2021**, *26*, 1–53. [CrossRef]
72. Mohanani, R.; Turhan, B.; Ralph, P. Requirements Framing Affects Design Creativity. *IEEE Trans. Softw. Eng.* **2021**, *47*, 936–947. [CrossRef]
73. Aranda, A.M.; Dieste, O.; Juristo, N. Effect of Domain Knowledge on Elicitation Effectiveness: An Internally Replicated Controlled Experiment. *IEEE Trans. Softw. Eng.* **2016**, *42*, 427–451. [CrossRef]
74. Pérez, F.; Echeverría, J.; Lapeña, R.; Cetina, C. Comparing manual and automated feature location in conceptual models: A Controlled experiment. *Inf. Softw. Technol.* **2020**, *125*, 106337. [CrossRef]
75. Dumas, J.S.; Redish, J. *A Practical Guide to Usability Testing*; Intellect: Exeter, UK, 1999.
76. Nielsen, J.; Molich, R. Heuristic evaluation of user interfaces. In Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'90), Seattle, WA, USA, 1–5 April 1990; pp. 249–256. [CrossRef]
77. Mahatody, T.; Sagar, M.; Kolski, C. State of the Art on the Cognitive Walkthrough Method, Its Variants and Evolutions. *Int. J. Hum. Comput. Interact.* **2010**, *26*, 741–785. [CrossRef]
78. *ISO 9241-11:2018*; Ergonomics of Human-System Interaction–Part 11: Usability: Definitions and Concepts; ISO: Geneva, Switzerland, 2018.

79. Quesenbery, W. The Five Dimensions of Usability. In *Content and Complexity: Information Design in Technical Communication*; Albers, M.J., Mazur, M.B., Eds.; Taylor & Francis: London, UK, 2003; Chapter 4, pp. 81–102. [CrossRef]
80. Holzinger, A. Usability engineering methods for software developers. *Commun. ACM* **2005**, *48*, 71–74. [CrossRef]
81. Sauro, J.; Lewis, J.R. Standardized usability questionnaires. In *Quantifying the User Experience: Practical Statistics for User Research*; Sauro, J., Lewis, J.R., Eds.; Morgan Kaufmann: Amsterdam, The Netherlands, 2016; Chapter 8, pp. 185–248. [CrossRef]
82. Hornbæk, K. Current practice in measuring usability: Challenges to usability studies and research. *Int. J. Hum. Comput. Stud.* **2006**, *64*, 79–102. [CrossRef]
83. Davis, F.D. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Q.* **1989**, *13*, 319–340. [CrossRef]
84. Lin, C. Exploring the relationship between technology acceptance model and usability test. *Inf. Technol. Manag.* **2013**, *14*, 243–255. [CrossRef]
85. Ferrari, A.; Mazzanti, F.; Basile, D.; ter Beek, M.H. Systematic Evaluation and Usability Analysis of Formal Methods Tools for Railway Signaling System Design. *IEEE Trans. Softw. Eng.* **2021**. [CrossRef]
86. Kelley, J.F. An empirical methodology for writing user-friendly natural language computer applications. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'83), ACM, Boston, MA, USA, 12–15 December 1983; pp. 193–196. [CrossRef]
87. Campbell, D.T.; Fiske, D.W. Convergent and discriminant validation by the multitrait-multimethod matrix. *Psychol. Bull.* **1959**, *56*, 81–105. [CrossRef]
88. Kadoda, G.F. Formal Software Development Tools (An Investigation into Usability). Ph.D. Thesis, Loughborough University, Loughborough, UK, 1997.
89. Shackel, B. Ergonomics in Design for Usability. In Proceedings of the 2nd Conference of the British Computer Society, Human Computer Interaction Specialist Group on People and Computers: Designing for Usability, York, UK, 23–26 August 1986; Cambridge University Press: Cambridge, UK, 1986; pp. 44–64.
90. Green, T.R.G.; Petre, M. Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework. *J. Vis. Lang. Comput.* **1996**, *7*, 131–174. [CrossRef]
91. Hussey, A.; MacColl, I.; Carrington, D.A. Assessing Usability from Formal User-Interface Designs. In Proceedings of the 13th IEEE Australian Software Engineering Conference (ASWEC'01), Canberra, Australia, 27–28 August 2001; pp. 40–47. [CrossRef]
92. Chechik, M. SC(R)^3: Towards Usability of Formal Methods. In Proceedings of the 8th Conference of the Centre for Advanced Studies on Collaborative Research (CASCON'98), IBM, Toronto, ON, Canada, 30 November–3 December 1998; pp. 8:1–8:16.
93. Loer, K.; Harrison, M.D. Towards Usable and Relevant Model Checking Techniques for the Analysis of Dependable Interactive Systems. In Proceedings of the 17th International Conference on Automated Software Engineering (ASE'02), IEEE, Edinburgh, UK, 23–27 September 2002; pp. 223–226. [CrossRef]
94. Loer, K.; Harrison, M.D. An integrated framework for the analysis of dependable interactive systems (IFADIS): Its tool support and evaluation. *Autom. Softw. Eng.* **2006**, *13*, 469–496. [CrossRef]
95. Maroneze, A.; Perrelle, V.; Kirchner, F. Advances in Usability of Formal Methods for Code Verification with Frama-C. *Electron. Commun. Eur. Assoc. Softw. Sci. Technol.* **2019**, *77*, 1–6. [CrossRef]
96. Arcaini, P.; Bonfanti, S.; Gargantini, A.; Riccobene, E.; Scandurra, P. Addressing Usability in a Formal Development Environment. In *Revised Selected Papers of the International Workshops at the 3rd World Congress on Formal Methods (FM'19)*; Sekerinski, E., Moreira, N., Oliveira, J.N., Ratiu, D., Guidotti, R., Farrell, M., Luckcuck, M., Marmsoler, D., Campos, J., Astarte, T., et al., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 12232, pp. 61–76. [CrossRef]
97. Rubin, J.; Chisnell, D. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*; Wiley: Hoboken, NJ, USA, 2008.
98. Sagar, K.; Saha, A. A systematic review of software usability studies. *Int. J. Inf. Technol.* **2017**, 1–24. [CrossRef]
99. Planas, E.; Cabot, J. How are UML class diagrams built in practice? A usability study of two UML tools: Magicdraw and Papyrus. *Comput. Stand. Interfaces* **2020**, *67*, 103363. [CrossRef]
100. Abrahão, S.; Insfrán, E.; González-Ladrón-de-Guevara, F.; Fernández-Diego, M.; Cano-Genoves, C.; de Oliveira, R.P. Assessing the effectiveness of goal-oriented modeling languages: A family of experiments. *Inf. Softw. Technol.* **2019**, *116*, 106171. [CrossRef]
101. Ryan, T.P. *Sample Size Determination and Power*; Wiley Series in Probability and Statistics; Wiley: Hoboken, NJ, USA, 2013.
102. Rea, L.M.; Parker, R.A. *Designing and Conducting Survey Research: A Comprehensive Guide*; Wiley: Hoboken, NJ, USA, 2014.
103. European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), 2016.
104. Kaplan, D. *Structural Equation Modeling: Foundations and Extensions*; *Advanced Quantitative Techniques in the Social Sciences*; SAGE: Thousand Oaks, CA, USA, 2008; Volume 10. [CrossRef]
105. Kline, R.B. *Principles and Practice of Structural Equation Modeling*; Guilford Press: New York, NY, USA, 2015.

106. Taherdoost, H. Validity and Reliability of the Research Instrument; How to Test the Validation of a Questionnaire/Survey in a Research. *Int. J. Acad. Res. Manag.* **2016**, *5*, 28–36. [CrossRef]
107. Craigen, D.; Gerhart, S.; Ralston, T. *Industrial Applications of Formal Methods to Model, Design and Analyze Computer Systems: An International Survey*; Advanced Computing and Telecommunication Series; William Andrew: Norwich, NY, USA, 1995. [CrossRef]
108. Craigen, D.; Gerhart, S.L.; Ralston, T. An International Survey of Industrial Applications of Formal Methods. In Proceedings of the 7th Z User Workshop, London, UK, 14–15 December 1992; Bowen, J.P., Nicholls, J.E., Eds.; Springer: Berlin/Heidelberg, Germany, 1992; pp. 1–5. [CrossRef]
109. Craigen, D. Formal Methods Technology Transfer: Impediments and Innovation. In Proceedings of the 6th International Conference on Concurrency Theory (CONCUR'95), Philadelphia, PA, USA, 21–24 August 1995; Lee, I., Smolka, S.A., Eds.; Springer: Berlin/Heidelberg, Germany, 1995; Volume 962, pp. 328–332. [CrossRef]
110. Craigen, D.; Gerhart, S.L.; Ralston, T. Formal Methods Reality Check: Industrial Usage. *IEEE Trans. Softw. Eng.* **1995**, *21*, 90–98. [CrossRef]
111. Clarke, E.M.; Wing, J.M. Formal Methods: State of the Art and Future Directions. *ACM Comput. Surv.* **1996**, *28*, 626–643. [CrossRef]
112. Woodcock, J.; Larsen, P.G.; Bicarregui, J.; Fitzgerald, J. Formal methods: Practice and experience. *ACM Comput. Surv.* **2009**, *41*, 19:1–19:36 [CrossRef]
113. Basile, D.; ter Beek, M.H.; Fantechi, A.; Gnesi, S.; Mazzanti, F.; Piattino, A.; Trentini, D.; Ferrari, A. On the Industrial Uptake of Formal Methods in the Railway Domain: A Survey with Stakeholders. In Proceedings of the 14th International Conference on Integrated Formal Methods (iFM'18), Maynooth, Ireland, 5–7 September 2018; Furia, C.A., Winter, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2018; Volume 11023, pp. 20–29. [CrossRef]
114. ter Beek, M.H.; Borälv, A.; Fantechi, A.; Ferrari, A.; Gnesi, S.; Löfving, C.; Mazzanti, F. Adopting Formal Methods in an Industrial Setting: The Railways Case. In Proceedings of the 3rd World Congress on Formal Methods: The Next 30 Years (FM'19), Porto, Portugal, 7–11 October 2019; ter Beek, M.H., McIver, A., Oliveira, J.N., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11800, pp. 762–772. [CrossRef]
115. Gleirscher, M.; Marmsoler, D. Formal Methods in Dependable Systems Engineering: A Survey of Professionals from Europe and North America. *Empir. Softw. Eng.* **2020**, *25*, 4473–4546. [CrossRef]
116. Linåker, J.; Sulaman, S.M.; Maiani de Mello, R.; Höst, M. *Guidelines for Conducting Surveys in Software Engineering*; Technical report; Department of Computer Science, Lund University: Lund, Sweden, 2015.
117. Kitchenham, B.A.; Pfleeger, S.L. Personal Opinion Surveys. In *Guide to Advanced Empirical Software Engineering*; Shull, F., Singer, J., Sjøberg, D.I.K., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 63–92. [CrossRef]
118. Wagner, S.; Méndez, D.; Felderer, M.; Graziotin, D.; Kalinowski, M. Challenges in Survey Research. In *Contemporary Empirical Methods in Software Engineering*; Felderer, M.; Travassos, G.H., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; pp. 93–125. [CrossRef]
119. Molléri, J.S.; Petersen, K.; Mendes, E. An empirically evaluated checklist for surveys in software engineering. *Inf. Softw. Technol.* **2020**, *119*, 106240. [CrossRef]
120. Baltes, S.; Ralph, P. Sampling in software engineering research: A critical review and guidelines. *Empir. Softw. Eng.* **2022**, *27*, 94:1–94:31. [CrossRef]
121. Ralph, P.; Tempero, E.D. Construct Validity in Software Engineering Research and Software Metrics. In Proceedings of the 22nd ACM International Conference on Evaluation and Assessment in Software Engineering (EASE'18), Christchurch, New Zealand, 28–29 June 2018; pp. 13–23. [CrossRef]
122. Heeringa, S.G.; West, B.T.; Berglund, P.A. *Applied Survey Data Analysis*; Statistics in the Social and Behavioral Sciences; Taylor & Francis: London, UK, 2020.
123. Agresti, A. *Categorical Data Analysis*; Wiley: Hoboken, NJ, USA, 2012.
124. Fernández, D.M.; Wagner, S.; Kalinowski, M.; Felderer, M.; Mafra, P.; Vetrò, A.; Conte, T.; Christiansson, M.T.; Greer, D.; Lassenius, C.; et al. Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice. *Empir. Softw. Eng.* **2017**, *22*, 2298–2338. [CrossRef]
125. Ralph, P.; Baltes, S.; Adisaputri, G.; Torkar, R.; Kovalenko, V.; Kalinowski, M.; Novielli, N.; Yoo, S.; Devroey, X.; Tan, X.; et al. Pandemic programming. *Empir. Softw. Eng.* **2020**, *25*, 4927–4961. [CrossRef] [PubMed]
126. Chow, T.; Cao, D. A survey study of critical success factors in agile software projects. *J. Syst. Softw.* **2008**, *81*, 961–971. [CrossRef]
127. Torchiano, M.; Tomassetti, F.; Ricca, F.; Tiso, A.; Reggio, G. Relevance, benefits, and problems of software modelling and model driven techniques—A survey in the Italian industry. *J. Syst. Softw.* **2013**, *86*, 2110–2126. [CrossRef]
128. Lethbridge, T.C.; Sim, S.E.; Singer, J. Studying Software Engineers: Data Collection Techniques for Software Field Studies. *Empir. Softw. Eng.* **2005**, *10*, 311–341. [CrossRef]
129. Vollstedt, M.; Rezat, S. An Introduction to Grounded Theory with a Special Focus on Axial Coding and the Coding Paradigm. In *Compendium for Early Career Researchers in Mathematics Education*; Kaiser, G., Presmeg, N., Eds.; ICME-13 Monographs; Springer: Berlin/Heidelberg, Germany, 2019; pp. 81–100. [CrossRef]
130. Corbin, J.M.; Strauss, A. Grounded Theory Research: Procedures, Canons, and Evaluative Criteria. *Qual. Sociol.* **1990**, *13*, 3–21. [CrossRef]
131. Glaser, B.G. *Theoretical Sensitivity: Advances in the Methodology of Grounded Theory*; Sociology Press: Mill Valley, CA, USA, 1978.

132. Charmaz, K. *Constructing Grounded Theory: A Practical Guide through Qualitative Analysis*; SAGE: Thousand Oaks, CA, USA, 2006.
133. Hoda, R. Socio-Technical Grounded Theory for Software Engineering. *IEEE Trans. Softw. Eng.* **2021**. [CrossRef]
134. Braun, V.; Clarke, V. Using thematic analysis in psychology. *Qual. Res. Psychol.* **2006**, *3*, 77–101. [CrossRef]
135. Cruzes, D.S.; Dybå, T. Recommended Steps for Thematic Synthesis in Software Engineering. In Proceedings of the 5th International Symposium on Empirical Software Engineering and Measurement (ESEM'11), IEEE, Banff, AB, Canada, 22–23 September 2011; pp. 275–284. [CrossRef]
136. Guba, E.G.; Lincoln, Y.S. Competing Paradigms in Qualitative Research. In *Handbook of Qualitative Research*; Denzin, N.K., Lincoln, Y.S., Eds.; SAGE: Thousand Oaks, CA, USA, 1994; Chapter 6; pp. 105–117.
137. Leung, L. Validity, reliability, and generalizability in qualitative research. *J. Family Med. Prim. Care* **2015**, *4*, 324–327. [CrossRef]
138. Charmaz, K.; Thornberg, R. The pursuit of quality in grounded theory. *Qual. Res. Psychol.* **2021**, *18*, 305–327. [CrossRef]
139. Snook, C.F.; Harrison, R. Practitioners' views on the use of formal methods: An industrial survey by structured interview. *Inf. Softw. Technol.* **2001**, *43*, 275–283. [CrossRef]
140. Bloomfield, R.E.; Craigen, D.; Koob, F.; Ullmann, M.; Wittmann, S. Formal Methods Diffusion: Past Lessons and Future Prospects. In Proceedings of the 19th International Conference on Computer Safety, Reliability and Security (SAFECOMP'00), Rotterdam, The Netherlands, 24–27 October 2020; Koornneef, F., van der Meulen, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2000; Volume 1943, pp. 211–226. [CrossRef]
141. Bryman, A. *Social Research Methods*; Oxford University Press: Oxford, UK, 2016.
142. Sharp, H.; Dittrich, Y.; de Souza, C.R.B. The Role of Ethnographic Studies in Empirical Software Engineering. *IEEE Trans. Softw. Eng.* **2016**, *42*, 786–804. [CrossRef]
143. Zhang, H.; Huang, X.; Zhou, X.; Huang, H.; Babar, M.A. Ethnographic Research in Software Engineering: A Critical Review and Checklist. In Proceedings of the 27th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE'19), Tallinn, Estonia, 26–30 August 2019; pp. 659–670. [CrossRef]
144. Saldaña, J. *The Coding Manual for Qualitative Researchers*; SAGE: Thousand Oaks, CA, USA, 2021.
145. Stol, K.J.; Ralph, P.; Fitzgerald, B. Grounded Theory in Software Engineering Research: A Critical Review and Guidelines. In Proceedings of the ACM 38th International Conference on Software Engineering (ICSE'16), Austin, TX, USA, 14–22 May 2016; pp. 120–131. [CrossRef]
146. Ågren, S.M.; Knauss, E.; Heldal, R.; Pelliccione, P.; Malmqvist, G.; Bodén, J. The impact of requirements on systems development speed: A multiple-case study in automotive. *Requir. Eng.* **2019**, *24*, 315–340. [CrossRef]
147. Yang, N.; Cuijpers, P.J.L.; Schiffelers, R.R.H.; Lukkien, J.; Serebrenik, A. An Interview Study of how Developers use Execution Logs in Embedded Software Engineering. In Proceedings of the 43rd IEEE International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP'21), Madrid, Spain, 25–28 May 2021; pp. 61–70. [CrossRef]
148. Strandberg, P.E.; Enoiu, E.P.; Afzal, W.; Sundmark, D.; Feldt, R. Information Flow in Software Testing—An Interview Study With Embedded Software Engineering Practitioners. *IEEE Access* **2019**, *7*, 46434–46453. [CrossRef]
149. Masood, Z.; Hoda, R.; Blincoe, K. Real World Scrum A Grounded Theory of Variations in Practice. *IEEE Trans. Softw. Eng.* **2022**, *48*, 1579–1591. [CrossRef]
150. Leite, L.A.F.; Pinto, G.; Kon, F.; Meirelles, P. The organization of software teams in the quest for continuous delivery: A grounded theory approach. *Inf. Softw. Technol.* **2021**, *139*, 106672. [CrossRef]
151. Delbecq, A.L.; Van de Ven, A.H. A Group Process Model for Problem Identification and Program Planning. *J. Appl. Behav. Sci.* **1971**, *7*, 466–492. [CrossRef]
152. Keeney, S.; Hasson, F.; McKenna, H. Consulting the oracle: Ten lessons from using the Delphi technique in nursing research. *J. Adv. Nurs.* **2006**, *53*, 205–212. [CrossRef]
153. Krueger, R.A.; Casey, M.A. *Focus Groups: A Practical Guide for Applied Research*; SAGE: Thousand Oaks, CA, USA, 2014.
154. Breen, R.L. A Practical Guide to Focus-Group Research. *J. Geogr. High. Educ.* **2006**, *30*, 463–475. [CrossRef]
155. Kontio, J.; Bragge, J.; Lehtola, L. The Focus Group Method as an Empirical Tool in Software Engineering. In *Guide to Advanced Empirical Software Engineering*; Shull, F., Singer, J., Sjøberg, D.I.K., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 93–116. [CrossRef]
156. Dalkey, N.; Helmer, O. An Experimental Application of the DELPHI Method to the Use of Experts. *Manag. Sci.* **1963**, *9*, 458–467. [CrossRef]
157. Mullen, P.M. Delphi: Myths and reality. *J. Health Organ. Manag.* **2003**, *17*, 37–52. [CrossRef] [PubMed]
158. Varndell, W.; Fry, M.; Lutze, M.; Elliott, D. Use of the Delphi method to generate guidance in emergency nursing practice: A systematic review. *Int. Emerg. Nurs.* **2021**, *56*, 100867. [CrossRef] [PubMed]
159. McMillan, S.S.; King, M.; Tully, M.P. How to use the nominal group and Delphi techniques. *Int. J. Clin. Pharm.* **2016**, *38*, 655–662. [CrossRef] [PubMed]
160. Dunham, R.B. *Nominal Group Technique: A Users' Guide*; Technical report, University of Wisconsin-Madison: Madison, WI, USA, 1998.
161. Shestopalov, S. Organizing Brainstorming Workshops: A Designer's Guide. *Smashing Mag.* **2019**.
162. Fairley, R. *Software Engineering Concepts*; Series in Software Engineering and Technology; McGraw-Hill: New York, NY, USA, 1985.

163. Keil, M.; Lee, H.K.; Deng, T. Understanding the most critical skills for managing IT projects: A Delphi study of IT project managers. *Inf. Manag.* **2013**, *50*, 398–414. [CrossRef]
164. Holtkamp, P.; Jokinen, J.P.P.; Pawlowski, J.M. Soft competency requirements in requirements engineering, software design, implementation, and testing. *J. Syst. Softw.* **2015**, *101*, 136–146. [CrossRef]
165. Groeneveld, W.; Jacobs, H.; Vennekens, J.; Aerts, K. Non-cognitive Abilities of Exceptional Software Engineers: A Delphi Study. In Proceedings of the 51st Technical Symposium on Computer Science Education (SIGCSE'20), ACM, Portland, OR, USA, 11–14 March 2020; pp. 1096–1102. [CrossRef]
166. Murphy, J.P.; Rådestad, M.; Kurland, L.; Jirwe, M.; Djalali, A.; Rüter, A. Emergency department registered nurses' disaster medicine competencies. An exploratory study utilizing a modified Delphi technique. *Int. Emerg. Nurs.* **2019**, *43*, 84–91. [CrossRef]
167. Harvey, N.; Holmes, C.A. Nominal group technique: An effective method for obtaining group consensus. *Int. J. Nurs. Pract.* **2012**, *18*, 188–194. [CrossRef]
168. Abbas, M.; Ferrari, A.; Shatnawi, A.; Enoiu, E.; Saadatmand, M.; Sundmark, D. On the relationship between similar requirements and similar software: A case study in the railway domain. *Requir. Eng.* **2022**. [CrossRef]
169. Martakis, A.; Daneva, M. Handling requirements dependencies in agile projects: A focus group with agile software development practitioners. In Proceedings of the 7th International Conference on Research Challenges in Information Science (RCIS'13), IEEE, Paris, France, 29–31 May 2013; pp. 1–11. [CrossRef]
170. De Angelis, G.; Ferrari, A.; Gnesi, S.; Polini, A. Requirements elicitation and refinement in collaborative research projects. *J. Softw. Evol. Process.* **2018**, *30*, e1990. [CrossRef]
171. Wohlin, C. Case Study Research in Software Engineering—It is a Case, and it is a Study, but is it a Case Study? *Inf. Softw. Technol.* **2021**, *133*, 106514. [CrossRef]
172. Staron, M. Action Research as Research Methodology in Software Engineering. In *Action Research in Software Engineering: Theory and Applications*; Staron, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2020; Chapter 2, pp. 15–36. [CrossRef]
173. Wieringa, R.J. *Design Science Methodology for Information Systems and Software Engineering*; Springer: Berlin/Heidelberg, Germany, 2014. [CrossRef]
174. Runeson, P.; Höst, M.; Rainer, A.; Regnell, B. *Case Study Research in Software Engineering: Guidelines and Examples*; Wiley: Hoboken, NJ, USA, 2012.
175. Wieringa, R.J.; Daneva, M. Six strategies for generalizing software engineering theories. *Sci. Comput. Program.* **2015**, *101*, 136–152. [CrossRef]
176. Galloway, A.J.; Cockram, T.J.; McDermid, J.A. Experiences with the Application of Discrete Formal Methods to the Development of Engine Control Software. *IFAC Proc. Vol.* **1998**, *31*, 49–56. [CrossRef]
177. Chudnov, A.; Collins, N.; Cook, B.; Dodds, J.; Huffman, B.; MacCárthaigh, C.; Magill, S.; Mertens, E.; Mullen, E.; Tasiran, S.; et al. Continuous Formal Verification of Amazon s2n. In Proceedings of the 30th International Conference on Computer Aided Verification (CAV'18), Oxford, UK, 14–17 July 2018; Chockler, H., Weissenbacher, G., Eds.; Springer: Berlin/Heidelberg, Germany, 2018; Volume 10982, pp. 430–446. [CrossRef]
178. Leuschel, M.; Falampin, J.; Fritz, F.; Plagge, D. Automated property verification for large scale B models with ProB. *Form. Asp. Comput.* **2011**, *23*, 683–709. [CrossRef]
179. Ferrari, A.; Fantechi, A.; Magnani, G.; Grasso, D.; Tempestini, M. The Metrô Rio case study. *Sci. Comput. Program.* **2013**, *78*, 828–842. [CrossRef]
180. Bosschaart, M.; Quaglietta, E.; Janssen, B.; Goverde, R.M.P. Efficient formalization of railway interlocking data in RailML. *Inf. Syst.* **2015**, *49*, 126–141. [CrossRef]
181. Hamid, B.; Pérez, J. Supporting pattern-based dependability engineering via model-driven development: Approach, tool-support and empirical validation. *J. Syst. Softw.* **2016**, *122*, 239–273. [CrossRef]
182. Comptier, M.; Leuschel, M.; Mejia, L.F.; Perez, J.M.; Mutz, M. Property-Based Modelling and Validation of a CBTC Zone Controller in Event-B. In Proceedings of the 3rd International Conference on Reliability, Safety, and Security of Railway Systems: Modelling, Analysis, Verification, and Certification (RSSRail'19), Lille, France, 4–6 June 2019; Collart-Dutilleul, S., Lecomte, T., Romanovsky, A.B., Eds.; Springer: Berlin/Heidelberg, Germany, 2019; Volume 11495, pp. 202–212. [CrossRef]
183. Behm, P.; Benoit, P.; Faivre, A.; Meynadier, J.M. Météor: A Successful Application of B in a Large Project. In Proceedings of the 1st World Congress on Formal Methods in the Development of Computing Systems (FM'99), Toulouse, France, 20–24 September 1999; Wing, J.M., Woodcock, J., Davies, J., Eds.; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1708, pp. 369–387. [CrossRef]
184. Johannesson, P.; Perjons, E. *An Introduction to Design Science*; Springer: Berlin/Heidelberg, Germany, 2014. [CrossRef]
185. Britto, R.; Smite, D.; Damm, L.O.; Börstler, J. Evaluating and strategizing the onboarding of software developers in large-scale globally distributed projects. *J. Syst. Softw.* **2020**, *169*, 110699. [CrossRef]
186. Lwakatare, L.E.; Kilamo, T.; Karvonen, T.; Sauvola, T.; Heikkilä, V.; Itkonen, J.; Kuvaja, P.; Mikkonen, T.; Oivo, M.; Lassenius, C. DevOps in practice: A multiple case study of five companies. *Inf. Softw. Technol.* **2019**, *114*, 217–230. [CrossRef]
187. Tómasdóttir, K.F.; Aniche, M.F.; van Deursen, A. The Adoption of JavaScript Linters in Practice: A Case Study on ESLint. *IEEE Trans. Softw. Eng.* **2020**, *46*, 863–891. [CrossRef]
188. Ferrari, A.; Gori, G.; Rosadini, B.; Trotta, I.; Bacherini, S.; Fantechi, A.; Gnesi, S. Detecting requirements defects with NLP patterns: An industrial experience in the railway domain. *Empir. Softw. Eng.* **2018**, *23*, 3684–3733. [CrossRef]

189. Ochodek, M.; Hebig, R.; Meding, W.; Frost, G.; Staron, M. Recognizing lines of code violating company-specific coding guidelines using machine learning. *Empir. Softw. Eng.* **2020**, *25*, 220–265. [CrossRef]
190. Manzano, M.; Ayala, C.P.; Gómez, C.; Abherve, A.; Franch, X.; Mendes, E. A Method to Estimate Software Strategic Indicators in Software Development: An Industrial Application. *Inf. Softw. Technol.* **2021**, *129*, 106433. [CrossRef]
191. Brereton, P.; Kitchenham, B.A.; Budgen, D.; Turner, M.; Khalil, M. Lessons from applying the systematic literature review process within the software engineering domain. *J. Syst. Softw.* **2007**, *80*, 571–583. [CrossRef]
192. Kitchenham, B.A.; Brereton, O.P.; Budgen, D.; Turner, M.; Bailey, J.; Linkman, S.G. Systematic literature reviews in software engineering - A systematic literature review. *Inf. Softw. Technol.* **2009**, *51*, 7–15. [CrossRef]
193. Garousi, V.; Felderer, M.; Mäntylä, M.V. The need for multivocal literature reviews in software engineering: complementing systematic literature reviews with grey literature. In Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering (EASE'16), ACM, Limerick, Ireland, 1–3 June 2016; pp. 26:1–26:6. [CrossRef]
194. Garousi, V.; Felderer, M.; Mäntylä, M.V. Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Inf. Softw. Technol.* **2019**, *106*, 101–121. [CrossRef]
195. Ferrari, A.; ter Beek, M.H. Formal Methods in Railways: A Systematic Mapping Study. *ACM Comput. Surv.* **2022**. [CrossRef]
196. Filho, R.M.P.T.; Oliveira, L.P.; Carneiro, L.N. Security, Power Consumption and Simulations in IoT Device Networks: A Systematic Review. In Proceedings of the 36th International Conference on Advanced Information Networking and Applications (AINA'22), Sydney, Australia, 13–15 April 2022; Barolli, L., Hussain, F., Enokido, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; Volume 451, pp. 370–379. [CrossRef]
197. Armando, A.; Basin, D.A.; Boichut, Y.; Chevalier, Y.; Compagna, L.; Cuéllar, J.; Drielsma, P.H.; Héam, P.C.; Kouchnarenko, O.; Mantovani, J.; et al. The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In Proceedings of the 17th International Conference on Computer Aided Verification (CAV'05), Scotland, UK, 6–10 June 2005; Etessami, K., Rajamani, S.K., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3576, pp. 281–285. [CrossRef]
198. Mishra, A.D.; Mustafa, K. A review on security requirements specification by formal methods. *Concurr. Comput. Pract. Exp.* **2022**, *34*, 6702. [CrossRef]
199. Zahid, F.; Tanveer, A.; Kuo, M.M.Y.; Sinha, R. A systematic mapping of semi-formal and formal methods in requirements engineering of industrial Cyber-Physical systems. *J. Intell. Manuf.* **2022**, *33*, 1603–1638. [CrossRef]
200. Durelli, R.S.; Durelli, V.H.S. A systematic review on mining techniques for crosscutting concerns. In Proceedings of the IX Experimental Software Engineering Latin American Workshop (ESELAW'12), Brasília, Brazil, 12–14 December 2012; pp. 1–9.
201. Weyns, D.; Iftikhar, M.U.; de la Iglesia, D.G.; Ahmad, T. A survey of formal methods in self-adaptive systems. In Proceedings of the 5th ACM International C* Conference on Computer Science & Software Engineering (C3S2E'12), Montreal, QC, Canada, 27–29 June 2012; pp. 67–79. [CrossRef]
202. Bonfanti, S.; Gargantini, A.; Mashkoor, A. A systematic literature review of the use of formal methods in medical software systems. *J. Softw. Evol. Process.* **2018**, *30*, e1943. [CrossRef]
203. Mashkoor, A.; Kossak, F.; Egyed, A. Evaluating the suitability of state-based formal methods for industrial deployment. *Softw. Pract. Exp.* **2018**, *48*, 2350–2379. [CrossRef]
204. Rajabli, N.; Flammini, F.; Nardone, R.; Vittorini, V. Software Verification and Validation of Safe Autonomous Cars: A Systematic Literature Review. *IEEE Access* **2021**, *9*, 4797–4819. [CrossRef]
205. Zhumagambetov, R. Teaching Formal Methods in Academia: A Systematic Literature Review. In Proceedings of the 1st International Workshop on Formal Methods – Fun for Everybody (FMFun'19), Bergen, Norway, 2–3 December 2019; Cerone, A., Roggenbach, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; Volume 1301, pp. 218–226. [CrossRef]
206. Kitchenham, B. *Procedures for Performing Systematic Reviews*; Technical Report TR/SE-0401; Keele University: Keele, UK, 2004.
207. Petersen, K.; Feldt, R.; Mujtaba, S.; Mattsson, M. Systematic Mapping Studies in Software Engineering. In Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE'08), Bari, Italy, 26–27 June 2008; Visaggio, G., Baldassarre, M.T., Linkman, S.G., Turner, M., Eds.; BCS: Swindon, UK, 2008; *Workshops in Computing*. [CrossRef]
208. Petersen, K.; Vakkalanka, S.; Kuzniarz, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Inf. Softw. Technol.* **2015**, *64*, 1–18. [CrossRef]
209. Zhang, H.; Babar, M.A.; Tell, P. Identifying relevant studies in software engineering. *Inf. Softw. Technol.* **2011**, *53*, 625–637. [CrossRef]
210. Wohlin, C. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. In Proceedings of the 18th ACM International Conference on Evaluation and Assessment in Software Engineering (EASE'14), London, UK, 13–14 May 2014; Shepperd, M.J., Hall, T., Myrtveit, I., Eds.; 2014; pp. 38:1–38:10. [CrossRef]
211. Ampatzoglou, A.; Bibi, S.; Avgeriou, P.; Verbeek, M.; Chatzigeorgiou, A. Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. *Inf. Softw. Technol.* **2019**, *106*, 201–230. [CrossRef]
212. Wohlin, C.; Mendes, E.; Felizardo, K.R.; Kalinowski, M. Guidelines for the search strategy to update systematic literature reviews in software engineering. *Inf. Softw. Technol.* **2020**, *127*, 106366. [CrossRef]
213. Dąbrowski, J.; Letier, E.; Perini, A.; Susi, A. Analysing app reviews for software engineering: A systematic literature review. *Empir. Softw. Eng.* **2022**, *27*, pp. 43:1–43:63 [CrossRef]
214. Bano, M.; Zowghi, D. A systematic review on the relationship between user involvement and system success. *Inf. Softw. Technol.* **2015**, *58*, 148–169. [CrossRef]

215. Martínez-Fernández, S.; Bogner, J.; Franch, X.; Oriol, M.; Siebert, J.; Trendowicz, A.; Vollmer, A.M.; Wagner, S. Software Engineering for AI-Based Systems: A Survey. *ACM Trans. Softw. Eng. Methodol.* **2022**, *31*, 37e:1-37e:59. [CrossRef]
216. Horkoff, J.; Aydemir, F.B.; Cardoso, E.; Li, T.; Maté, A.; Paja, E.; Salnitri, M.; Piras, L.; Mylopoulos, J.; Giorgini, P. Goal-oriented requirements engineering: An extended systematic mapping study. *Requir. Eng.* **2019**, *24*, 133–160. [CrossRef]
217. Garousi, V.; Felderer, M.; Hacaloğlu, T. Software test maturity assessment and test process improvement: A multivocal literature review. *Inf. Softw. Technol.* **2017**, *85*, 16–42. [CrossRef]
218. Scheuner, J.; Leitner, P. Function-as-a-Service performance evaluation: A multivocal literature review. *J. Syst. Softw.* **2020**, *170*, 110708. [CrossRef]