SUBMISSION: 4

TITLE: The Design Science Paradigm as a Frame for Empirical Software Engineering

REVIEW 1

SUBMISSION: 4

TITLE: The Design Science Paradigm as a Frame for Empirical Software Engineering

AUTHORS: Per Runeson, Emelie Engstrm and Margaret-Anne Storey

——— Overall evaluation ———

SCORE: 2 (accept)

—– TEXT: I very much appreciate this contribution given the increased relevance of Design Science research nowadays also in Software Engineering research. Below, I summarise some observations I made in the sincere hope to support the authors in their revision of the book chapter.

Main observations:

- Scope: From reading the abstract and the introduction, it was not really clear to me what exactly the goal and the scope of the chapter was. My perception at the end was that the authors specifically aim to (1) raise awareness of Design Science as a research philosophy and to (2) delineate from other philosophies given the many misconceptions we still have (comparable to the overuse of the term "Grounded Theory" in SE papers). I suggest the authors to make this goal explicit at the end of the introduction. *[Per says: Agree with interpretation. Clarify scope.][Emelie says: agree]*

- Target audience: Related to my previous remark, the authors should highlight at the end of the introduction what the primary audience is. My impression is that the target audience is rather advanced scholars in software engineering as many key terms in empirical software engineering are used but not explained (as we would do in a classic ABC chapters). *[Per says: The topic is quite advanced, still we want it to be understood by e.g. PhD students. Define key terms?][Emelie says: sounds like a good idea. Which are the key terms?]*

- Delineation: The authors should very briefly set Design Science into context of empirical software engineering to make explicit that while (in my view) DS is a collection of principles with a rather rigid form into which different research strategies may be fitted, empirical software engineering can be rather described as a (inter-)disciplinary movement with various methods following same or similar high-level goals (theory building and / or evaluation). *[Per says: Clarify paradigm – methodology – method?][Emelie says: It's always good to calrify things but I'm not sure the reviewers' distinction btw. DS and Empirical software engineering helps here. Maybe we should avoid using the term empirical software engineering instead as it is not very distinct. I think we should stick to DS as a paradigm and explain what we mean by that.]*

- Related to the scope: The introduction of visual abstracts as a means to communicate research should be motivated a bit better. My personal opinion is that visual abstracts are an excellent means to especially communicate DS-based contributions, but in principle, they can be used to communicate any empirical research contribution (my understanding). When reading the chapter, I didn't expect the subsections on visual abstracts. I therefore suggest motivating them already in the introduction and explaining their relevance to the overall book chapter. *[Per says: The VAs are a tools to communicate DS-based research. Clarify motivation][Emelie says: At least our template, but maybe not all VAs, I agree that we should clarify this.]*

Further (mostly minor) comments from here on:

- I found the abstract a bit too complicated to digest (wording + long sentences). Especially: I was wondering what would be new about this chapter when there is already so much literature on this topic. Later I realised well that existing literature comes indeed from information systems research and that it needs adoptions and different forms of introduction to fit the particularities of (E)SE research. I recommend, therefore, making this need already explicit in the abstract. *[Per says: Clarify in abstract]*

- Abstract (second sentence): "validation" –¿ "the validity"? (or "how to validate proposals") *[Per says: Prefer: how to validate proposals]*

- Introduction (second page, second paragraph): "With the advent of empirical software engineering [2] and evidence-based software engineering [22], the research focus shifted towards validation of solution proposals". I wouldn't exclude problem understanding / framing from the focus of empirical software engineering. Maybe rephrase to "With the advent of empirical software engineering [2] and evidence-based software engineering [22], the research focus shifted towards empirically-informed solution proposals" (or something like that). *[Per says: ok]*

- Introduction (page 2, bottom part): "in depth" –¿ "in-depth" *[Per says: ok]*

- Section 2 (first sentence): The term "paradigm" is heavily overused in our community, often for small increments in knowledge creation; at least when comparing the term with where the term "paradigm-shift" actually emerges from. In my understanding and as introduced by T. Kuhn, a paradigm entitles socially accepted belief systems and research procedures by which whole disciplinary communities may or may not governed (e.g. quantum mechanics). A paradigm therefore also includes a system of theories underlying whole research programmes and on basis of which scientific practices are generally built. In this context, I found the introduction as used by van Aken too simplistic and wouldn't consider DS as a paradigm but rather as a research philosophy (which may well co-exist with others in contrast to when it would be a paradigm which then is often seen as a –competing– alternative). I might be of course wrong here, but would ask the authors to at least reflect upon the use of this term. *[Per says: To discuss: research paradigm vs. research philosophy][Emelie says: Yes, let's discuss this again ... :-). Maybe our claims are a bit bold but I think it is a paradigm and as such a competing alternative. I don't want to change to philosphy as I think it is to weak.]*

- Section 2: I believe that it would be useful to (at least consider) visually adding the cycle as introduced by Wieringa (book chapter 3). This would make the elaboration more accessible to the reader. *[Per says: Wieringa's adapted cycle may be too specific, but a more general one is fine with me.]*

- On page 5, when introducing the figure, I was wondering about the notion of "artifacts" as introduced by Wieringa (and others in DS). Later on page 8, the authors relate to artifacts as introduced by Wieringa in light of information systems research to delineate this view from software engineering research (i.e. the focus on solutions). I recommend briefly introducing what Wieringa (and related scholars) sees as artifacts on page 5 with a few examples and then to reference back to it from page 8. *[Per says: Apparently, we have to address artifacts]*

- Section 3.3, page 8 upper paragraph: The authors introduce some of the particularities in software engineering research referring to its typical socio-technical context. I suggest the authors to broaden this discussion a bit more. I myself go even so far to say that software engineering research should be seen as an inherently interdisciplinary area (see our work "Empirical Software Engineering: From Discipline to Interdiscipline", JSS, 2018 - https://arxiv.org/abs/1805.08302 - discussing our (inter)discipline in light of scientific movements and visualising the empirical lifecycle for theory building and theory evaluation; maybe this article is useful to the authors of this chapter). In any case, I see this - often human-centric - characteristic of software engineering research as one strong argument for DS research. In this very section, the authors could give some examples for the importance of DS to tackle the emergent challenges in SE research (e.g. in context of academia-industry collaborations). *[Per says: I will try to read his paper – has been on my wish list for long]*

- Section 3.4, page 9: The authors refer to the work of Stol et al (reference 31). Please note that these authors are contributing also a chapter to this book. Maybe explicitly refer to this chapter. *[Per says: Agree]*

- Section 3.5, second sentence: I would remove the term "realism" and leave it with "natural settings". Even simulations may rely on a realism world view when what the authors refer to (in my understanding) is natural / realistic environments. *[Per says: Ok]*

- Section 4.3.1. (and others): "machine-learning" –¿ "machine learning" (in this instance: "machine learning-based"?) *[Per says: Ok]*

- Section 4.5, second paragraph: The sentences "We are currently in the process of getting feedback from industry partners. For example, see http://dsse.org for a collection of visual abstracts we authored based on the distinguished papers from ICSE from 2014 to 2018." seem a bit disconnected to the rest. To what exactly do the authors refer when writing that they were getting feedback? On the abstracts? If so, then maybe write "At the time of writing this chapter, we are in the process of evaluating our visual abstracts"? *[Per says: Ok]*

Best regards, Daniel Mendez

—————— REVIEW 2 ——————

SUBMISSION: 4

TITLE: The Design Science Paradigm as a Frame for Empirical Software Engineering

AUTHORS: Per Runeson, Emelie Engstrm and Margaret-Anne Storey

——— Overall evaluation ———

SCORE: 3 (strong accept)

—– TEXT:

This is very nice work and I am completely on board with the view that SE and Design Science can enjoy a very fruitful interaction, especially with a view to greater theorizing in SE.

One issue I would encourage the authors to tone down, or at least acknowledge, relates to the use of technological rule. I know this comes from Bunge who sees it as underpinned by a scientific law which goes beyond craft and intuition. However, I believe the authors themselves acknowledge the need for a holistic socio-technical view. An over-emphasis on technology has implications. For example, a critique of Hevners model of Design Science is that it has led to too much emphasis on building and evaluating a technological artefact, thereby downplaying the socio-technical elements. *[Per says: I think we should keep the material on the technological rule, but clarify that there typical social aspects of it. Alternatively calling it "socio-technical rule" in line with the discussion above on the scope of SE research?][Emelie says: or use "design propositions" instead. What would you say is the social aspect of a technological rule? The scope in terms of a context and effect or the open format: "To achieve" that leaves the actual decision to a person in a situation? People tend to stumble on the term "technological rule so it may be a good idea to use another term]*

A minor quibble would be the authors claim (p.5) that technological rules equate with Gregor and Hevners design theory. I think it the latter are fairly clear that technological rules equate with empirical generalizations. To be elevated to design theory would require an explanation of why the rule works as it does, specific conditions under which it holds, confidence levels as well as adequate testing. Certainly, the claim (p.6) that technological rules are examples of middle range theories is too much in my opinion. I personally believe that an increased emphasis on theory in SE is beneficial, but I would be concerned if technological rules such as To achieve ¡Effect¿ in ¡Situation¿, apply ¡Intervention¿ were proposed as middle range theory. I believe the authors qualify this somewhat, e.g., (p.17) technological rules can at best be considered as theory fragments, but readers will not necessarily note the qualification. *[Per says: Add the qualitifaction of theory* fragments *to the first instance]*

Overall, very nice work which will be widely read and used by SE researchers.

Typos

p.7. Para 2: be influence by should be influenced

p.9. Para 5: pragmatists viewpoint would be better as pragmatist viewpoint

p.18. Para 5: Two instances of Johannesson should be Johannesson and Perjons

REVIEW 3

SUBMISSION: 4

TITLE: The Design Science Paradigm as a Frame for Empirical Software Engineering

AUTHORS: Per Runeson, Emelie Engstrm and Margaret-Anne Storey

——— Overall evaluation ———

SCORE: 2 (accept)

— TEXT: Summary. The chapter describes the Design Science Paradigm as a frame for EMSE *[Per says: This reviewer comments on DS as a method, not paradigm]*

Pro + The topic is very interesting and brings in a nice contribution to the planned book.

Con + The chapter is well written. However, I would recommend adding some overview figures and some examples (maybe an ongoing example) to support readers who are unfamiliar with design science. *[Per says: The risk of adding process figures is that DS is seen as methodology rather than paradigm. Adding an example is quite much work – deadline August 22][Emelie says: Agree, We already have an example, could it be introduced earlier in the chapter?]*

Comments to the authors

Abstract

+ I think it would be great having a structured abstract in the book chapter (resp. in all book chapters). I think this need to be decided by the Michael and Guilherme. *[Per says: For the editors]*

Intro (1)

+ The paragraph The phrase software engineering .. looks somehow strange in the text. Consider re-formatting.

+ Consider adding illustrative examples where the design science approach can contribute. *[Per says: We may choose some from the ICSE papers]*

Design Science An Overview (2)

+ Consider providing some overview figure on the design science approach (maybe in terms of some process workflow or similar). I think it would be good having such a high-level overview in the chapter. This could be used later on for the examples as well to link individual steps to this overview. *[Per says: Again, methodology vs paradigm...]*

A Model of Design Science Research (3)

+ Fig 1 and textual description. Consider using some numbered tags to better link the elements of the figure to the textual description (i.e., the bullet-point list). *[Per says: Good point – implement]*

+ In general, I would suggest having examples (or maybe one example throughout the chapter) to explain theoretical concepts complemented by practical examples. *[Per says: Extensive work to present it through a coherent example – maybe we may use different snippets of examples?]*

+ Wording. Try to be consistent with the wording in Fig 1 and related subsections to avoid misunderstandings. *[Per says: Check and update]*

+ Citations. It needs to be clarified, whether a reference to the overall publication is sufficient of if a citation including a specific page is needed. For example on page 9 you cite [28, p.30]. *[Per says: We should be more specific for book citations]*

+ 3.5. Not sure whether this sub-section fits into this section as there is also some transfer from academia to industry (the other section focus on approaches around Fig 1). It seems to be an add one without clear contribution to the section (except the title of the subsection). Maybe it would be better to have a new section 4 dealing with these type of topics? *[Per says: We might add more examples of methods/research processes. Could that help communicating DS as a paradigm?]*

Using the Design Science Frame in Software Engineering (4)

+ I would cite [32] in the last paragraph of the introduction to section 3 as well (visual abstract template). *[Per says: Ok]*

+ 4.2. It seems that the approach follows more Fig 3 with some aspects of design science. Consider providing an overview of contributions from Fig 3 (technology transfer model) and fig 1 (design science) to better see the contributions. In the text this has been discussed to some extent, an illustration would be helpful to fully understand the approach

+ 4.3. Please provide citations related to individual characteristics of contribution assessments. Where do they come from?*[Per says: Anchor better in the literature]*

Recommended Further Reading (5)

+ Nice section of recommended readings.

Conclusion (6)

+ The chapter introduced Design Science in general and by example in software engineering. However, I think there is the need for bringing up some limitations of the approach (e.g., also the template that has been introduced earlier) or lessons learned. *[Per says: Acknowledge limitations]*

Summary. The chapter is interesting but partly difficult to follow because of a set of introduced aspects in various contexts. From my perspective, it would help to add some more examples / illustrations to enable readers in better understanding and following the approaches. Especially Fig 1 and 3 could go into an overview figure that handles Design Science in Software Engineering followed by the example. *[Per says: Possibly adding examples to illustrate abstract concepts. Definitely clarifying that we consider DS a paradigm.]*