



Hochschule Reutlingen
Reutlingen University

Hacking or Engineering?

Towards an Extended Entrepreneurial Software Engineering Model

Marco Kuhrmann, Jürgen Münch
Jil Klünder

Reutlingen University, HHZ
Leibniz University Hannover

(ICSSP/ICGSE 2022)

Agenda

- Context: The Entrepreneurial Software Engineering Model
- The Study
- Results
 - Study Results
 - The **Extended** Entrepreneurial Software Engineering Model
- Conclusion

The Entrepreneurial Software Engineering Model

Some Context

Setting

- Single case study: one startup
- Young people with mixed skill profiles, only a few software engineers

Approach

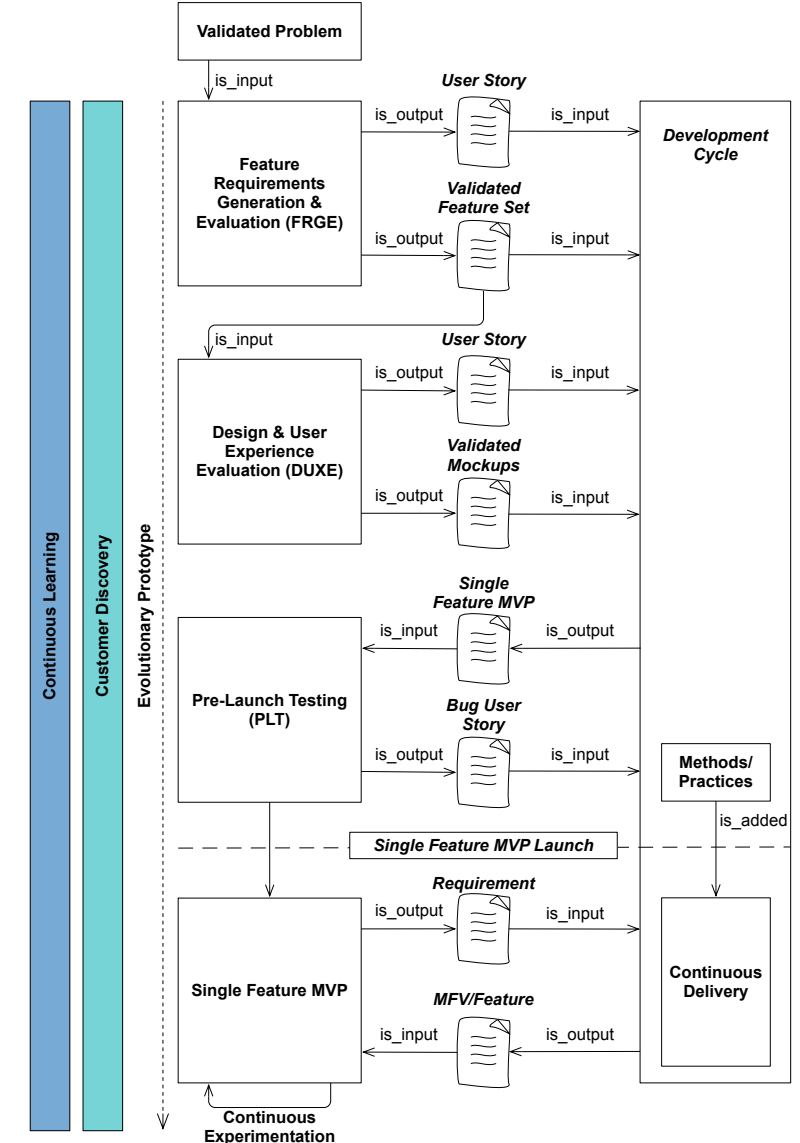
- Utilize research on hybrid development methods
- Propose a specific approach tailored for this startup
- Generalize and propose method
- Evaluate with experts

Result

- 3-staged process with main goal: support MVP development
- Basic idea: use FDD as a framework and provide stage-wise support
- Model: KISS principle – help non-SE folks to provide IT solutions

Problem

- One startup only – risk: one-case generalization
- What do other startups do?



The Study

Objective and Research Questions

Objective: [How do software startups develop software and how to provide them with proper methodical support?](#)

- Startups: “get some software done”
- Is there any systematic approach - or just hacking...?
- Focus: core activities (RE, AR, PD)

Research Questions

RQ1: Which requirements engineering methods do startups use?

RQ2: Which software architecture means do startups use to develop their software?

RQ3: Which product development strategies do startups use?

RQ4: Does the software process change with the age of a startup?

Method and Instrument

- Instrument: online survey
- Structure:
 - Metadata (to know whom we’re talking to)
 - 3 extra parts: RE, AR, PD
 - Final part: importance of activities and own skills
- General Question Design:
 - No “explicit” questions – avoid “agile” bias
 - Questions are designed around attributes of interest
- Data Collection
 - Mixed approach: snowballing + convenience sampling
 - Incubators, social networks, personal networks
 - April-May 2020 (70 responses in total, 40 complete)
- Data Analysis
 - RQ1-3: descriptive statistics + qualitative analysis
 - RQ4: analytical statistics

Results: Study

These are the “key” results only.
Further details can be obtained
from the paper...

Requirements Engineering

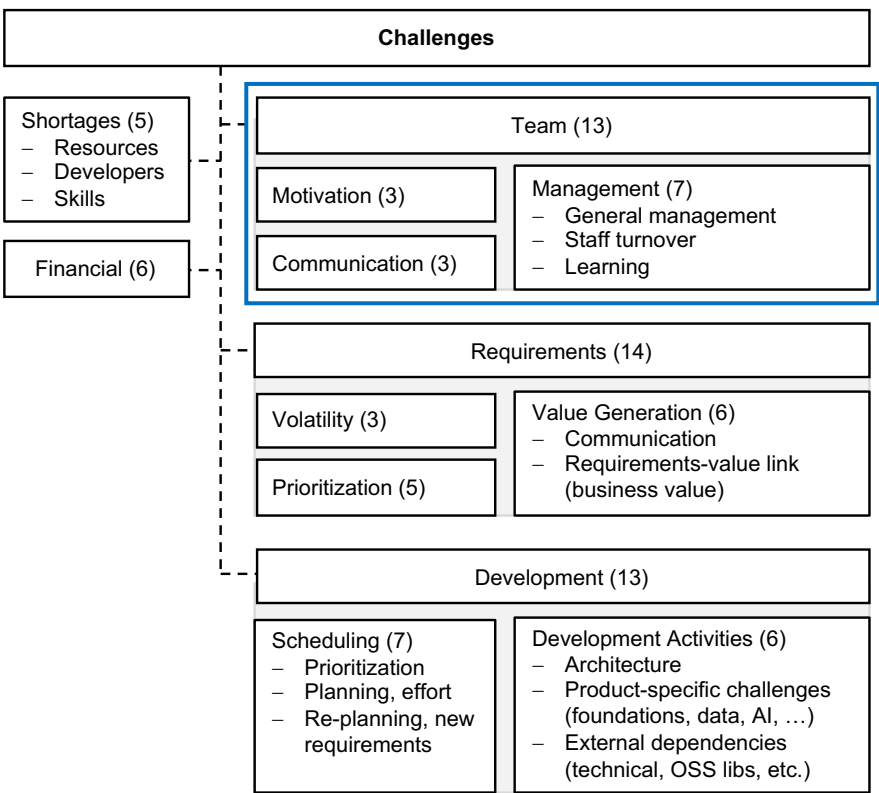
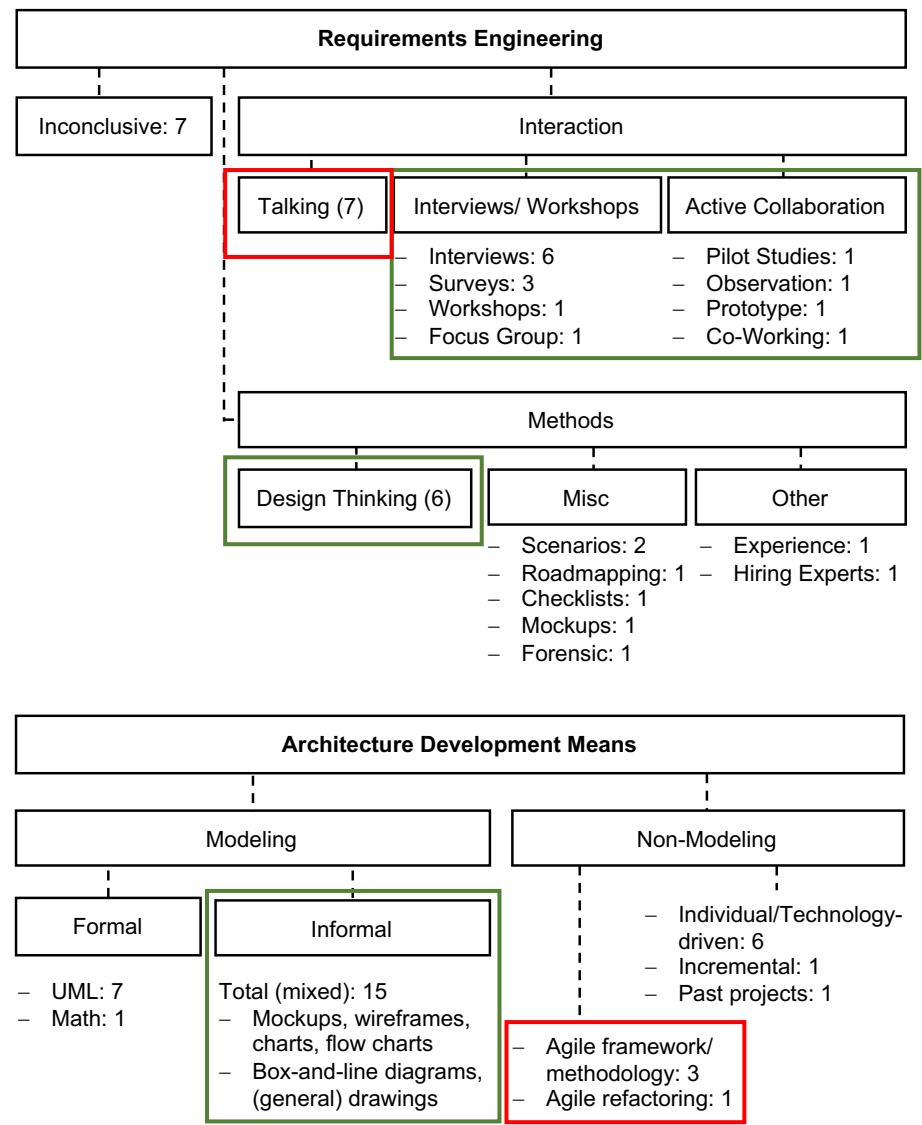
- Interaction and in-/semi-formal methods
- Talking?

Architecture

- Mostly informal
- WTF is “agile”?

Challenges

- Management (aka process)
- 2nd most hurting challenge: Requirements Engineering



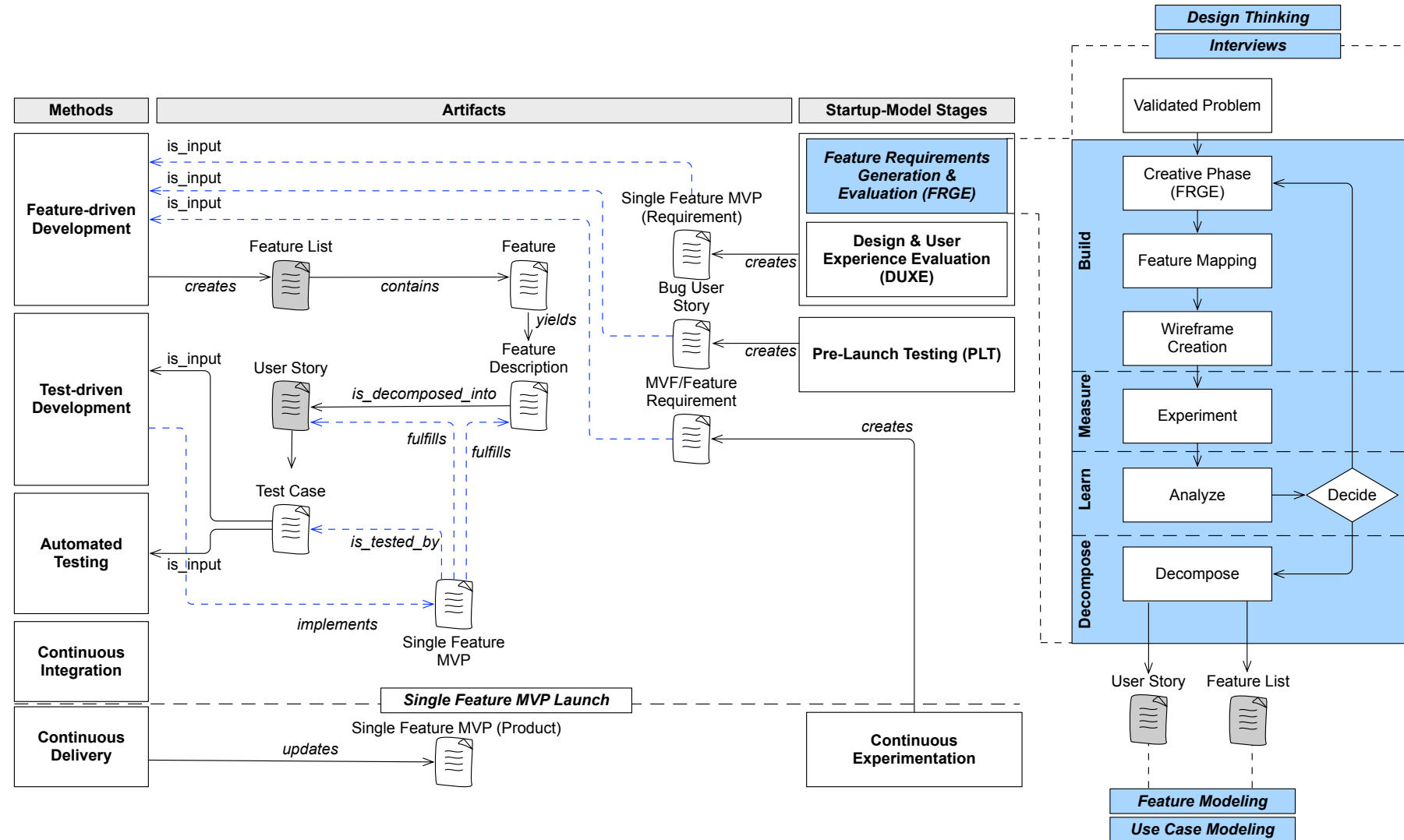
Results: The Extended Entrepreneurial Software Engineering Model

Updating the model...

- Emphasis on 1st phase
- Make Design Thinking explicit
- Formalize review process
- Put feature and use case modeling into the spotlight

Why?

- Design Thinking is used anyway → make it a first-class citizen
- RE puzzles startups – use FDD and supporting methods to help keep the focus
- Management? Well, this was already in the model...



Conclusion

Formal

- We had a model (method proposal)
- We conducted a survey to strengthen evidence
- We developed an extension

Findings Summarized

- Startups suffer from Requirements Engineering
- Startups suffer from (general) management
- Startups seem to have little to no idea about architecture...
- The model: original model fairly good fit with few updates

Next Steps

- Put the model into practice and validate it
- Refine and evolve
- Rethink education (methods and tool)

Missed Something? Check the paper...

Hacking or Engineering? Towards an Extended Entrepreneurial Software Engineering Model

Marco Kuhrmann
Reutlingen University
Germany
kuhrmann@acm.org

Jürgen Münch
Reutlingen University
Germany
j.muench@computer.org

Jil Klünder
Leibniz University Hannover
Germany
jil.kluender@inf.uni-hannover.de

ABSTRACT

Startups play a key role in software-based innovation. They make an important contribution to an economy's ability to compete and innovate, and their importance will continue to grow due to increasing digitalization. However, the success of a startup depends primarily on market needs and the ability to develop a solution that is attractive enough for customers to choose. A sophisticated technical solution is usually not critical, especially in the early stages of a startup. It is not necessary to be an experienced software engineer to start a software startup. However, this can become problematic as the solution matures and software complexity increases. Based on a proposed solution for systematic software development for early-stage startups, in this paper, we present the key findings of a survey study to identify the methodological and technical priorities

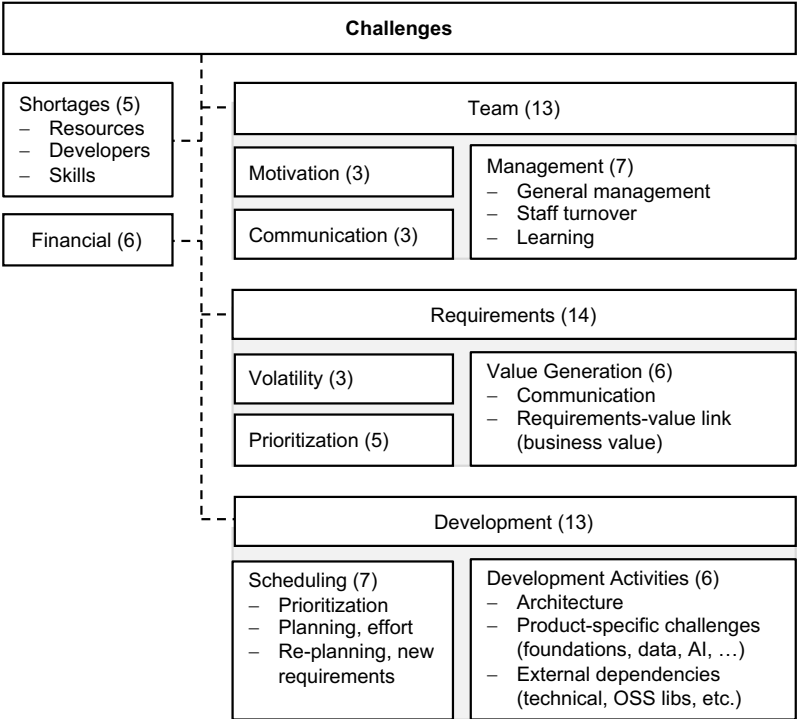
1 INTRODUCTION

Startups are fresh companies, which usually grow and mature fast [12, 32] and which are founded by one or more *entrepreneurs* to develop a unique product or service. Startups understand themselves as highly innovative, which has a few implications: Quite often, startups work with little or no "history" [13]. Accordingly, Blank [3] describes a startup as a *temporary organization that creates high-end technology and has no operation record*, and Bach et al. [1] define a startup as a *team of committed and high-energy people who are working without well defined development processes*. That is, startups cannot rely on settled structures and existing customers. Therefore, startups often lack clear requirements [4, 6] and competencies regarding methods and tools to develop an innovative software product. That makes software startups risky businesses.

Conclusion – Something’s wrong here...

But wait...

Look at the figure again:



What do the people think of themselves...

	Please evaluate your own skill levels of the following activities:					Q
	Excellent skills	Good skills	Moderate skills	No skills	don't know	
Management of the software development	13	15	8	2	1	28.1
Elicitation and management of requirements	11	13	13	0	2	28.2
Design of the system/software architecture	19	9	4	6	1	28.3
Quality assurance and testing	8	16	11	3	1	28.4
Continuous development, integration, and delivery	10	16	10	2	1	28.5

Self-perception, challenges and survey results do not match!