FHDW HANNOVER

How to Mitigate Technical Debt in Startups

Extending the Entrepreneurial Software Engineering Model

Jil Klünder, Florian Heintz, Jakob Droste , Marion Wiese und Marco Kuhrmann

21.11.2025
PVM Tagung

Über mich



- Werdegang:
  - seit 2025: Prof. für Software Engineering und Data Science (FHDW Hannover)
  - 2015 bis 2025: Wiss. MA im Software Engineering (Leibniz Universität Hannover)
  - 2011 bis 2015: Mathestudium (Leibniz Universität Hannover)

- Themenschwerpunkte:
  - Sozio-technische Aspekte des Software Engineering
    - Fokus auf psychologischen Aspekten wie Zusammenarbeit, Kommunikation, Interaktion
  - Software Process Engineering
  - Entrepreneurial Software Engineering
  - AI4SE

Startups in Zahlen

- In Deutschland wurden im ersten Halbjahr 2023 insgesamt 239 neue Software-Startups gegründet. Das sind 23 Prozent mehr als im zweiten Halbjahr 2022[1] und mehr als in allen anderen Branchen
- Insgesamt befassten sich 2023 31.9% aller Startups in Deutschland mit Software[2]
- Ihr Ziel: Marktlücken zu füllen, indem sie innovative Produkte entwickeln, die beispielsweise Privatpersonen in ihrem Alltag oder Unternehmen in ihrem Betrieb unterstützen.
- Dabei oft Verwendung von „neuen“ Technologien wie künstliche Intelligenz und Cloud Computing[2]

[1] S. Álvarez, „In diesen Bereichen gibt es die meisten Neugründungen,“ Jul 2023. https://www.wiwo.de/politik/deutschland/start-up-standort-deutschland-in-diesen-bereichen-gibt-es-die-meisten-neugruendungen/29252560.html.
[2] A. Hirschfeld, J. Gilde und V. Walk, „Der Deutsche Startup Monitor 2023 ist veröffentlicht,“ PwC Deutschland, Sept 2023. https://deutscherstartupmonitor.de/.

Startups in Zahlen

- Zum jetzigen Zeitpunkt schaffen es mehr als zwei Drittel aller Startups nicht, ihren Investoren eine positive Rendite zu liefern[3]
- Im Durchschnitt scheitern 60% aller gegründeten Startups innerhalb der fünf ersten Jahre[4]
- Das bedeutet, dass zwar die Idee eines Startups vielversprechend sein kann und jährlich eine hohe Anzahl an solchen Unternehmen entstehen, es aber nur die wenigsten schaffen, sich durchzusetzen.

Auch wenn die Idee eines Startups vielversprechend ist und jährlich eine hohe Anzahl an solchen Unternehmen entsteht, schaffen es nur die wenigsten, sich durchzusetzen.

[3] T. Eisenmann, Why Startups Fail: A New Roadmap for Entrepreneurial Success, P. R. H. LLC, Hrsg., Currency, 2021.
[4] C. Giardino, M. Unterkalmsteiner, N. Paternoster, T. Gorschek und P. Abrahamsson, „What Do We Know about Software Development in Startups?," IEEE Software, pp. 28-32, 2014.

Herausforderungen von Startups

- Zeitdruck: Schneller sein als die Konkurrenz
  - Möglichst schnell ein MVP auf den Markt bringen
  - Lauffähige Software ist wichtiger als qualitativ hochwertige
- Finanzen: Um Einnahmen zu generieren, müssen Startups in Vorleistung gehen
  - Unternehmen zahlen für Software erst, wenn sie eine gewisse Reife hat
  - Zahlende Kunden werden benötigt, um die Software weiterentwickeln zu können

Dabei werden oft Unzulänglichkeiten in Kauf genommen...

Warum Entrepreneurial Software Engineering?

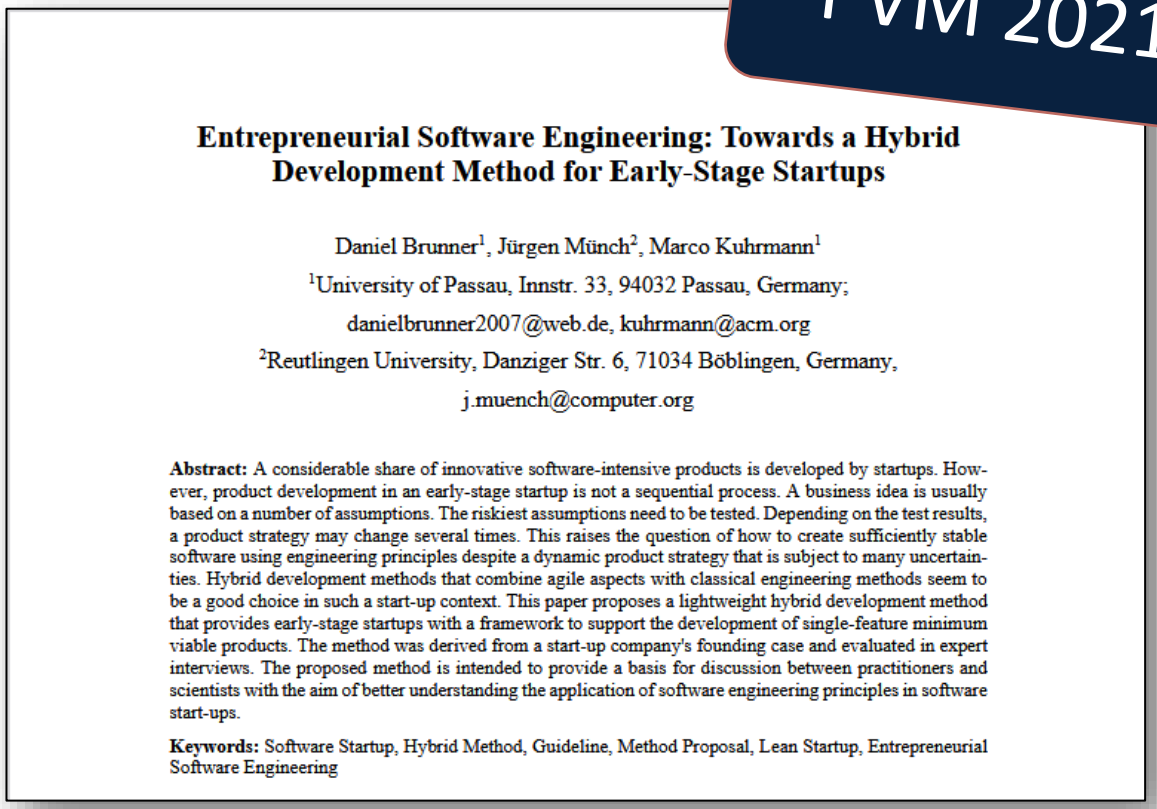
Wie können wir (Forschung) Startups helfen,
die richtige Balance zwischen Qualität und Geschwindigkeit zu finden?
Können wir Best Practices für die Softwareentwicklung in Startups herleiten?

Wie alles begann...

PVM 2021

Ansatz

- Einzelfallstudie mit einem Startup
- Berücksichtigung von Forschung zu hybriden Softwareentwicklungsmethoden
- Herleitung eines spezifischen Ansatzes für dieses Startup
- Generalisierung und Vorschlag einer Entwicklungsmethode bzw. eines Modells
- Evaluation mit Experten



Entrepreneurial Software Engineering: Towards a Hybrid Development Method for Early-Stage Startups

Daniel Brunner[1], Jürgen Münch[2], Marco Kuhrmann[1]
[1]University of Passau, Innstr. 33, 94032 Passau, Germany;
danielbrunner2007@web.de, kuhrmann@acm.org
[2]Reutlingen University, Danziger Str. 6, 71034 Böblingen, Germany,
j.muench@computer.org

Abstract: A considerable share of innovative software-intensive products is developed by startups. However, product development in an early-stage startup is not a sequential process. A business idea is usually based on a number of assumptions. The riskiest assumptions need to be tested. Depending on the test results, a product strategy may change several times. This raises the question of how to create sufficiently stable software using engineering principles despite a dynamic product strategy that is subject to many uncertainties. Hybrid development methods that combine agile aspects with classical engineering methods seem to be a good choice in such a start-up context. This paper proposes a lightweight hybrid development method that provides early-stage startups with a framework to support the development of single-feature minimum viable products. The method was derived from a start-up company's founding case and evaluated in expert interviews. The proposed method is intended to provide a basis for discussion between practitioners and scientists with the aim of better understanding the application of software engineering principles in software start-ups.

Keywords: Software Startup, Hybrid Method, Guideline, Method Proposal, Lean Startup, Entrepreneurial Software Engineering

Wie alles begann…

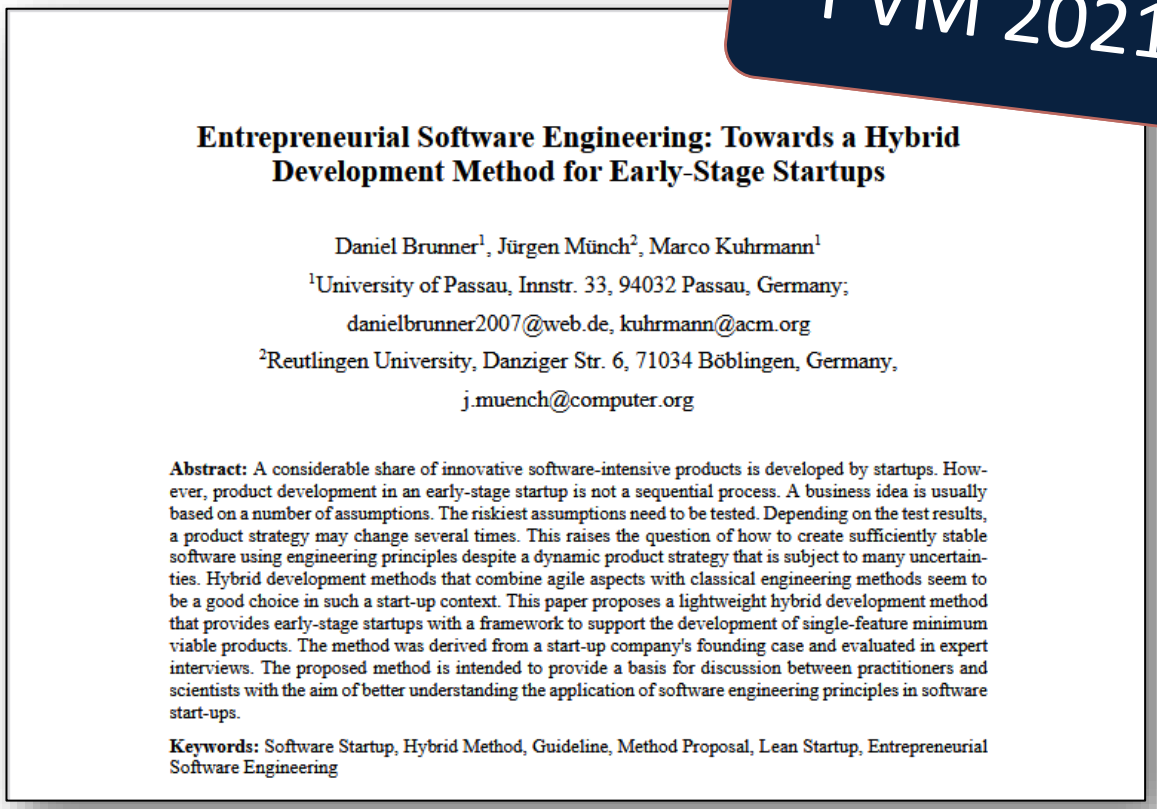
PVM 2021

Ergebnis

- 3-stufiger Prozess mit einem Hauptziel: Unterstützung der MVP-Entwicklung
- Kernidee: Verwende FDD als Framework und biete Unterstützung für jede Phase
- Modell: KISS-Prinzip – Modell für Nicht-SEler

Problem

- Nur ein Startup: Wir können nichts generalisieren
- Was machen andere Startups?



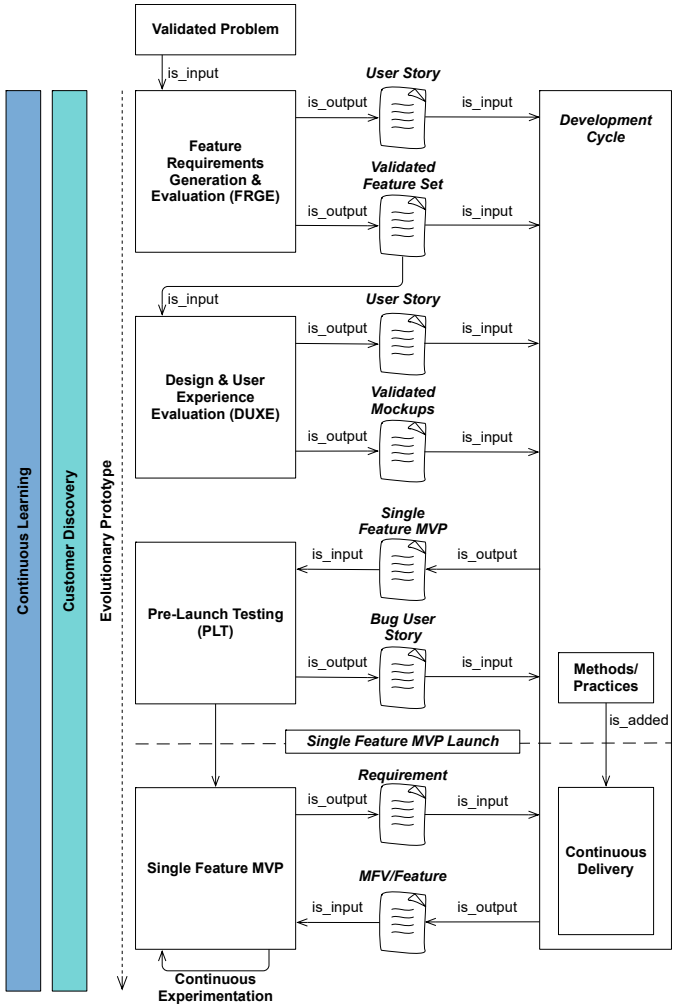
Entrepreneurial Software Engineering: Towards a Hybrid Development Method for Early-Stage Startups

Daniel Brunner[1], Jürgen Münch[2], Marco Kuhrmann[1]
[1]University of Passau, Innstr. 33, 94032 Passau, Germany;
danielbrunner2007@web.de, kuhrmann@acm.org
[2]Reutlingen University, Danziger Str. 6, 71034 Böblingen, Germany,
j.muench@computer.org

Abstract: A considerable share of innovative software-intensive products is developed by startups. However, product development in an early-stage startup is not a sequential process. A business idea is usually based on a number of assumptions. The riskiest assumptions need to be tested. Depending on the test results, a product strategy may change several times. This raises the question of how to create sufficiently stable software using engineering principles despite a dynamic product strategy that is subject to many uncertainties. Hybrid development methods that combine agile aspects with classical engineering methods seem to be a good choice in such a start-up context. This paper proposes a lightweight hybrid development method that provides early-stage startups with a framework to support the development of single-feature minimum viable products. The method was derived from a start-up company's founding case and evaluated in expert interviews. The proposed method is intended to provide a basis for discussion between practitioners and scientists with the aim of better understanding the application of software engineering principles in software start-ups.

Keywords: Software Startup, Hybrid Method, Guideline, Method Proposal, Lean Startup, Entrepreneurial Software Engineering

Das erste Zwischenergebnis



PVM 2021

Entrepreneurial Software Engineering: Towards a Hybrid Development Method for Early-Stage Startups

Daniel Brunner[1], Jürgen Münch[2], Marco Kuhrmann[1]
[1]University of Passau, Innstr. 33, 94032 Passau, Germany;
danielbrunner2007@web.de, kuhrmann@acm.org
[2]Reutlingen University, Danziger Str. 6, 71034 Böblingen, Germany,
j.muench@computer.org

Abstract: A considerable share of innovative software-intensive products is developed by startups. However, product development in an early-stage startup is not a sequential process. A business idea is usually based on a number of assumptions. The riskiest assumptions need to be tested. Depending on the test results, a product strategy may change several times. This raises the question of how to create sufficiently stable software using engineering principles despite a dynamic product strategy that is subject to many uncertainties. Hybrid development methods that combine agile aspects with classical engineering methods seem to be a good choice in such a start-up context. This paper proposes a lightweight hybrid development method that provides early-stage startups with a framework to support the development of single-feature minimum viable products. The method was derived from a start-up company's founding case and evaluated in expert interviews. The proposed method is intended to provide a basis for discussion between practitioners and scientists with the aim of better understanding the application of software engineering principles in software start-ups.

Keywords: Software Startup, Hybrid Method, Guideline, Method Proposal, Lean Startup, Entrepreneurial Software Engineering

Nächster Schritt...

ICSSP 2022

Ziel

- Ein bisschen mehr Generalisierbarkeit: Gilt das Modell auch für andere Startups?
- Konkretisierungen für einzelne Phasen

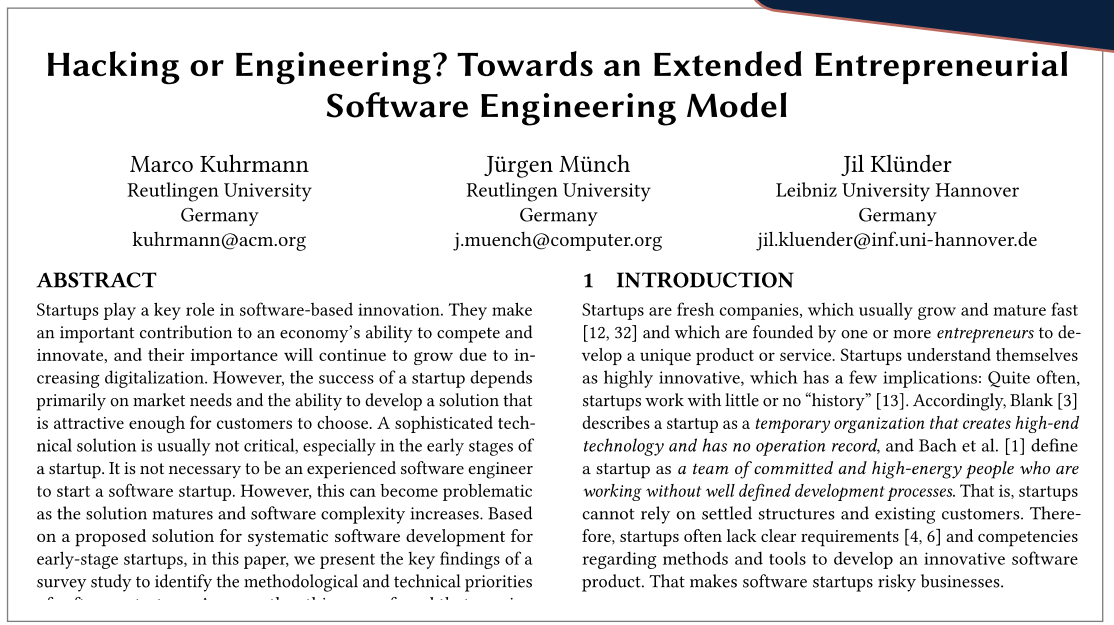
Forschungsfragen

RQ1: Welche RE-Methoden werden verwendet?

RQ2: Wie steht es um SWA?

RQ3: Wie sieht die Produktentwicklung aus?

RQ4: Ändert sich der Prozess mit dem Alter eines Startups?



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.

Nächster Schritt...

ICSSP 2022

Methodik

- Onlineumfrage
  - Metadaten (Demographie)
  - 3 Teile: RE, SWA, PD
  - Schluss: Bewertung der Aktitivitäten und der eigenen Fähigkeiten
- Wichtig: Wir haben keine expliziten Fragen gestellt, um Bias zu vermeiden

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.

Datenerhebung

- Recruiting über Snowballing und Convenience Sampling
- Inkubatoren, soziale Netzwerke, persönliche Netzwerke
- April bis Mai 2020 (70 Antworten, davon 40 vollständig)

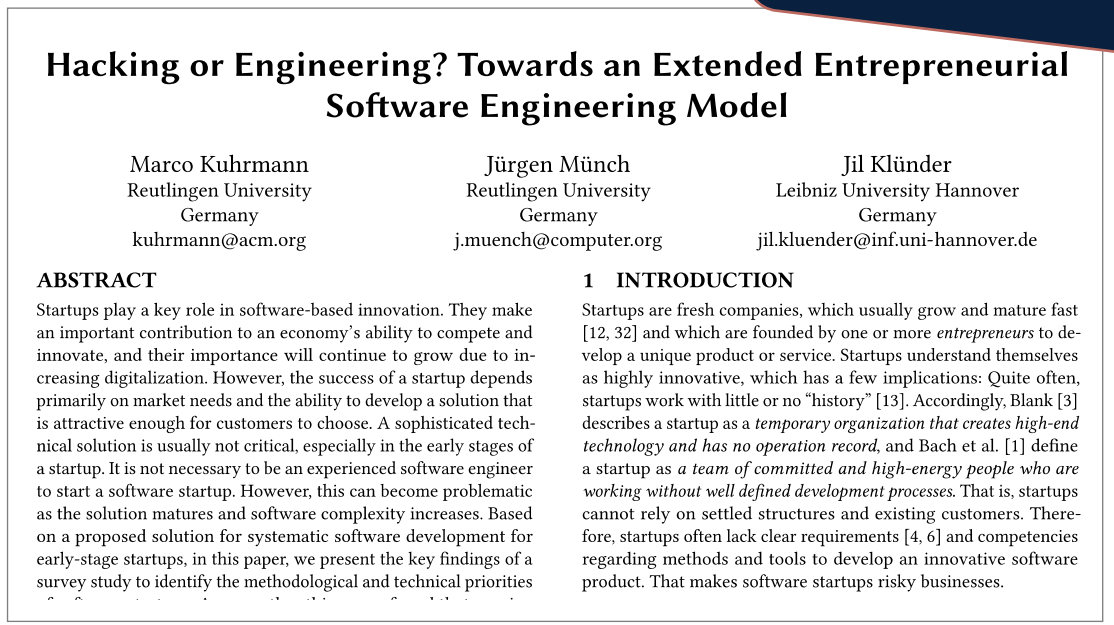
Erkenntnisse

ICSSP 2022

- Startups haben keinen systematischen RE-Prozess
- Startups haben Schwierigkeiten mit dem allgemeinen Projektmanagement
- Startups scheinen nicht viel Wissen über SWA zu haben…
- Das initiale Modell war schon recht generisch und passend. Einige Ergänzungen wurden dennoch vorgenommen.

Problem:

- Keine praktische Anwendung
- Keine Validierung in der Praxis



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.

Weiteres Problem – weiterer Ansatz

- Startups nehmen provisorische Lösungen in Kauf
  - „Softwarearchitektur? Machen wir später…“
  - „Softwarequalität? Erstmal eine lauffähige Software haben…“

- Dabei sind sie sich der Tragweite technischer Schulden nicht bewusst

Welche technischen Schulden können auftreten?
Welche davon sind besonders wichtig?
Wie können wir Startups dabei helfen, sich den technischen Schulden bewusst zu werden und sie zu vermeiden (oder sie bewusst in Kauf zu nehmen)?

Exkurs: Definition „Technische Schulden“

Technical debt is a “*a collection of design or implementation constructs that are expedient in the short term, but set up a technical context that can make future changes more costly or impossible*”[6]

[6] Avgeriou, Paris and Kruchten, Philippe and Ozkaya, Ipek and Seaman, Carolyn: Managing technical debt in software engineering (dagstuhl seminar 16162). Dagstuhl reports 6 (4), pp. 110–138, 2016.

Ansatz unserer Studie und Vorgehen

Ziel:

- Linderung von technischen Schulden in Startups in den unterschiedlichen Phasen des Entwicklungsprozesses

Ablauf:

1. Identifiziere die relevantesten TD-Typen (zunächst allgemein, dann für Startups)
2. Analysiere die Gründe und Konsequenzen der unterschiedlichen TD-Typen
   - Wie können Startups sie vermeiden?
   - Welche Risiken gehen damit einher?
3. Bilde die TD-Typen auf die Phasen des Entwicklungsprozesses ab
4. Synthetisiere die Ergebnisse in einer Erweiterung des Entrepreneurial Software Engineering Modells

Schwierigkeiten mit technischen Schulden für Startups

Wichtige Fragestellungen:

- Welche technischen Schulden sind wann relevant? D.h., wann sollten sie angegangen werden?
- Welche technischen Schulden dürfen (für einen gewissen Zeitraum oder generell) akzeptiert werden?

Welche technischen Schulden sollten behoben oder vermieden werden
und welche dürfen hingenommen werden?

Technical Debt Management

- TD Management umfasst die Entscheidungen um den Umgang mit technischen Schulden
  - Identifikation der technischen Schulden
  - Überwachung der technischen Schulden
  - „Bezahlung“ der technischen Schulden

Worauf sollten Startups beim TD Management achten?

- Startups brauchen ein TD Management!
- Jemand muss sich verantwortlich fühlen
- TD Management braucht Zeit und Geld…
- Das ganze Team muss sich der technischen Schulden bewusst sein
  - Da liegt oft das Problem
  - In Startups noch wichtiger als in größeren Firmen
- Priorisierung der technischen Schulden oft schwierig

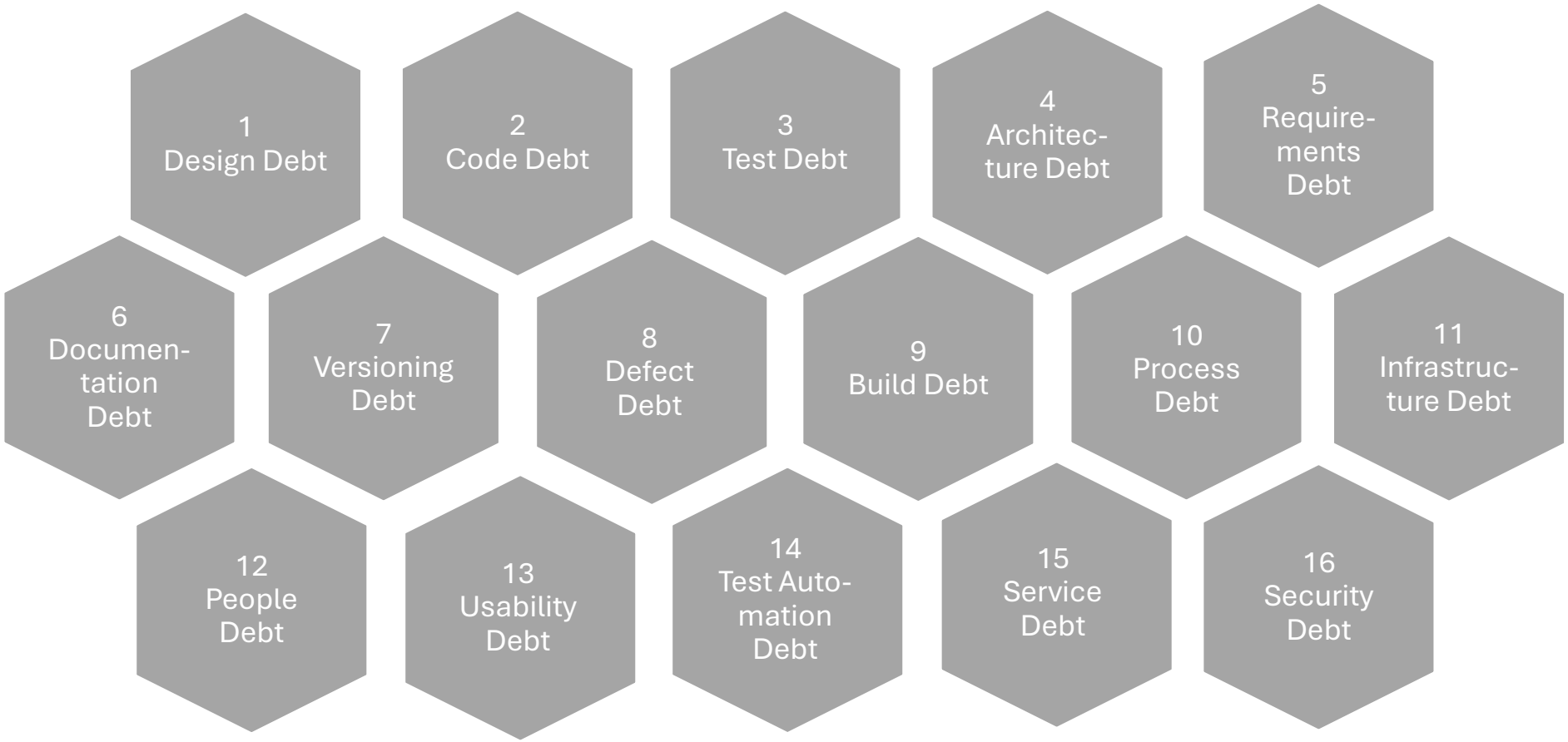
Das gilt gleichermaßen für große Firmen, aber die können damit anders umgehen…

Welche technischen Schulden sollten Startups priorisieren?

- Ranking der technischen Schulden basierend auf verschiedenen Kriterien
  - Wie oft ist der TD Typ relevant für Unternehmen?[7, 8]
  - Wie wichtig wird der TD Typ vom Management der Unternehmen erachtet?[9]
  - Wie oft wurde der TD-Typ in der Literatur betrachtet?[10, 11]
  - Wie viele Möglichkeiten gibt es, den TD-Typ zu händeln?[12]
  - Wie kritisch ist der TD-Typ, wenn er auftritt?[13]
  - Welche Folgen und negative Effekte gehen von dem TD-Typ aus?[14]

[7] Bogner, J.; Verdecchia, R.; Gerostathopoulos, I.: Characterizing technical debt and antipatterns in AI-based systems: A systematic mapping study. In: 2021 IEEE/ACM International Conference on Technical Debt (TechDebt). IEEE, pp. 64–73, 2021.
[8] Mandić,V.; Taušan,N.; Ramač, R.: The prevalence of the technical debt concept in Serbian it industry: Results of a national-wide survey. In: Proceedings of the 3rd international conference on technical debt. Pp. 77–86, 2020.
[9] Gomes, F. et al.: Investigating the point of view of project management practitioners on technical debt-a study on stack exchange. Journal of Software Engineering Research and Development 11 (1), pp. 12–1, 2023.
[10] Alves, N. S. et al.: Identification and management of technical debt: A systematic mapping study. Information and Software Technology 70, pp. 100–121, 2016.
[11] Rios, N.; de Mendonça Neto, M. G.; Spínola, R. O.: A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. Information and Software Technology 102, pp. 117–145, 2018.
[12] Li, Z.; Avgeriou, P.; Liang, P.: A systematic mapping study on technical debt and its management. Journal of Systems and Software 101, pp. 193–220, 2015.
[13] Kozanidis, N.; Verdecchia, R.; Guzmán, E.: Asking about technical debt: Characteristics and automatic identification of technical debt questions on stack overflow. In: Proceedings of the 16th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. Pp. 45–56, 2022.
[14] Rios, N. et al.: The practitioners' point of view on the concept of technical debt and its causes and consequences: a design for a global family of industrial surveys and its first results from Brazil. Empirical Software Engineering 25, pp. 3216–3287, 2020.

Ranking der 16 TD-Typen



Wo sollten technische Schulden im Prozess berücksichtigt werden? – Vorbeugende Maßnahmen

- Kontinuierliches TD-Management

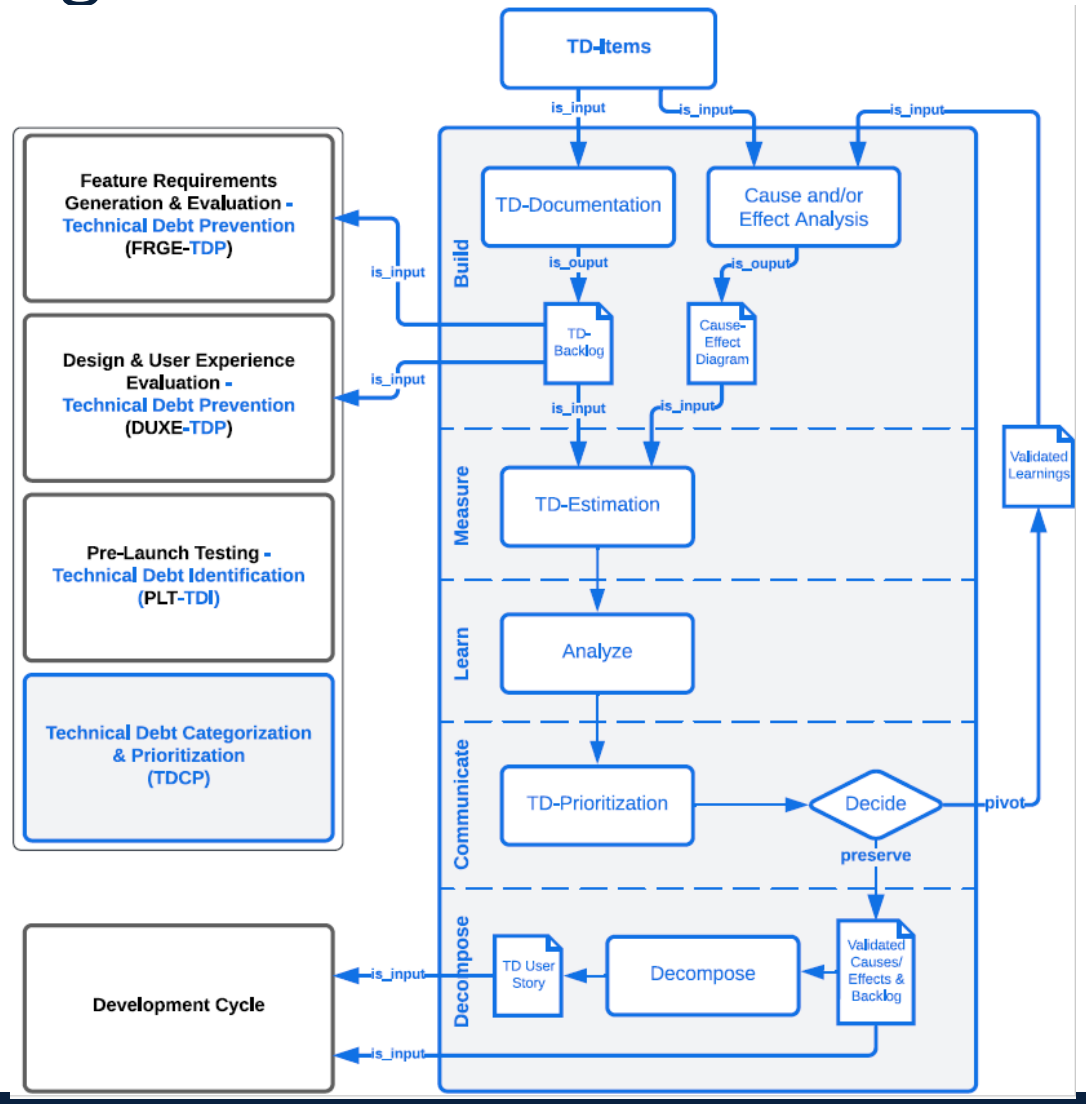
- Vorbeugung während der Feature Requirements Generation & Evaluation
  - Vermeidung von falsch verstandenen oder vergessenen Anforderungen

- Vorbeugung während der Design & User Experience Evaluation
  - Regelmäßiges Feedback von den Nutzern

Wo sollten technische Schulden im Prozess berücksichtigt werden? – Identifikation von technischen Schulden

- Identifikation der technischen Schulden beim Pre-Launch Testing
  - Keine Vorbeugung mehr möglich, nur noch Identifikation
  - Erzeugung von TD-Items (= identifizierte technische Schulden)
    - Input für die TD-Kategorisierung und –Priorisierung
    - Zur Identifikation nicht nur “gängige” Testarten (Security Testing, Installation Testing, Performance Testing), sondern auch Architecture Testing, Infrastructure Testing und Usability Testing

Wo sollten technische Schulden im Prozess berücksichtigt werden? – Umgang mit technischen Schulden



Wo sollten technische Schulden im Prozess berücksichtigt werden? – Umgang mit technischen Schulden

- Technical Debt Categorization & Prioritization
  - Dokumentation von TD und Analyse der Ursachen und Auswirkungen
  - Einschätzung der Relevanz von einzelnen TD-Items
  - Analyse und Priorisierung der TD-Items
  - Entscheidung, ob das TD-Item bearbeitet wird oder nicht

Fazit

- Technische Schulden sind nicht nur für große Unternehmen ein Problem
- Startups geraten aufgrund von fehlenden Erfahrungen, Druck vom Markt, finanziellen Abhängigkeiten und Zeitdruck schnell in Situationen, in denen sie unbeabsichtigt technische Schulden in Kauf nehmen
  - Diese später zu korrigieren, kostet noch mehr Zeit und Geld
  - Im schlimmsten Fall ist eine Neuentwicklung erforderlich
- Der Umgang mit technischen Schulden gestaltet sich jedoch als schwierig
  - Keine Standardlösung möglich
  - Bewusstsein muss geschaffen werden

Die meisten Entrepreneurs wissen nicht um die technischen Schulden, die sie eingehen, geschweige denn von deren Spätfolgen

Startups fehlt Wissen aus dem Software Engineering, obwohl Software DAS zu vermarktende Produkt ist

Vielen Dank für die Aufmerksamkeit!

Prof. Dr. Jil Klünder

Professor for Software Engineering and Data Science

Fachhochschule für die Wirtschaft Hannover (FHDW)

Freundallee 15 | 30173 Hannover

Fon +49 511 28483-116 | Mobil +49 174 917 6465

www.fhdw-hannover.de | jil.kluender@fhdw.de