Research on Technical Debt
What is this?

- **Pitch** for doing Research @ *<insert company name here>*

- Imagine being a **developer** in a company

- And I am standing here, telling you about **technical debt**
Who am I?

- Victor Brekenfeld
- Research Associate @ Freie Universität Berlin
- Software Engineering
- Technical Debt
Structure

● *Definition*: Technical Debt

● Research Approach

● Points of interest

● Current Research Subjects
Section 1

Technical Debt
Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite… The danger occurs when the debt is not repaid.

Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation[...].
Technical Debt?

The first kind of technical debt is the kind that is incurred unintentionally. […]

The second kind of technical debt is the kind that is incurred intentionally. […]

Short-term debt is the debt that's taken on tactically and reactively. […]

Long term debt is the debt a company takes on strategically and proactively. […]

Steve McConnell
Technical Debt?

I. Debt incurred unintentionally due to low quality work

II. Debt incurred intentionally
   II.A. Short-term debt, usually incurred reactively, for tactical reasons
       II.A.1. Individually identifiable shortcuts (like a car loan)
       II.A.2. Numerous tiny shortcuts (like credit card debt)
   II.B. Long-term debt, usually incurred proactively, for strategic reasons
Technological Debt?

<table>
<thead>
<tr>
<th>Deliberate</th>
<th>Reckless</th>
<th>Prudent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inadvertent</td>
<td>“We don’t have time for design”</td>
<td>“We must ship now and deal with the consequences”</td>
</tr>
<tr>
<td></td>
<td>“What’s layering?”</td>
<td>“Now we know, how we should have done it”</td>
</tr>
</tbody>
</table>

Martin Fowler
Where does it end?

- design debt / architectural debt
- defect debt?
- infrastructure debt
- testing debt
- documentation debt
- people debt
- requirement debt
- ...

victor.brekenfeld@fu-berlin.de
Landscape

Tackling TD

Using methods such as:

● SQALE
● DR-NEP
● CAST Software AIP
Section 2

Research Approach
Software Engineering (in research) is first and foremost constructive

- How should we do something?
  - New Methods
  - New Notations
  - New Tools
- This approach has mostly failed

We focus on Empirical Research (at first)

- Typically under-research topics using qualitative research methods
TD in the Enterprise Context

- *Metaphor* was designed to help communication
- **Decisions** are often made by non-technical stakeholders
  - e.g. “An Enterprise Perspective on Technical Debt” by Tim Klinger et al.
- **Because** software engineering involves complex action from and interaction between humans, psychological and sociological insights are required
  - Even worse in enterprises
  - This data can only be collected **empirically** by observation
Section 3
Points of Interest
Hypothesis
Points of interest

● **What pains** do **real teams** in various companies **perceive** in these respects?

● What **terms and concepts** do teams **use** to think and **talk about** their **TD** situation?

● Where and how is their **perception of the TD situation incorrect**? If so: **Why**?

● How do teams **cope** with **simple, easy-to-repair TD**?

● How with **medium-scale TD** that takes hours to repair?

● How with **large-scale (e.g. architectural) TD**?

● **What mindset** is **behind** these **approaches**?

● ...
Section 4

Research Subjects
Work so far

● Start in April 2019
  ○ Deep literature research
  ○ Contact & Interviews with ~6 companies
  ○ Interview with Michael Feathers
  ○ Cooperation with consulting company on the matter
  ○ Cooperation with two companies
Research Pitch

- Free **consulting**
- External **perspective**
- Specialized **knowledge**
- **Analysis** you don’t have the time for
- **Conceptualize** your *unique* problems
- => Indirect **knowledge transfer**
Research Subjects

- *How do you deal with TD?*
- *How do you refactor?*

- Attend meetings
- Pair-programming sessions
- Interviews
- Code-Reviews
- ...
Thank you!
Feedback?