MANAGING TECHNICAL DEBT IN SOFTWARE PROJECTS USING SCRUM: AN ACTION RESEARCH
Frederico Oliveira | Brazil
Master’s degree in Computer Engineering
Institute of Research Technology, São Paulo, Brazil
Software Development Manager, Scrum Master, Java Developer
Atlantic Solutions, São Paulo, Brazil

Alfredo Goldman, Ph.D. | Brazil
Institute of Mathematics and Statistics,
University of São Paulo, São Paulo, Brazil

Viviane Santos, Ph.D. | Brazil
Federal University of Pará,
Belém, Brazil
TECHNICAL DEBT is a metaphor developed by Ward Cunningham in 1992 to communicate problems due to “developing not in the right way” with non-technical stakeholders.

**met-a-phor** noun \( /m-e-t-a-, f-o-r ~ a-l-s-o ~ -f-a-r/ \)

A word or phrase for one thing that is used to refer to another thing in order to show or suggest that they are similar.
Technical Debt

Immature, incomplete, or inadequate artifacts in software development cycle that cause **higher costs** and **lower quality** in the long run.

(SEAMAN et al., 2011)
MOTIVATION

• Such as...
  • Not the right design choices;
  • Duplicated code;
  • Lack of tests;
  • “Workaround”.

“Don’t worry about the documentation for now!”
“The only one who can change this code is Carl!”
“Let’s finish the testing in the next release!”

“We don’t have time to reconcile these two databases before our deadline, so we’ll write some triggers to keep them synchronized for now”.

(SEAMAN, 2013)

(MCCONNEL, 2013)
"Technical debt is everything that makes your code harder to change."
(Tom Poppendieck - Lean Software Development)

“Most companies have to spend 80% of their software development budget maintaining code.”

Aaron Erickson, informit.com
• **Technical Debt x Agile Methods**

  - Developing and delivering very rapidly, with **no time for proper design (..)**, and a **lack of rigor or systematic testing (...)**, lead some agile projects into massive amounts of debt **very rapidly**.
    
    (KRUCHTEN et al., 2012)

  - It’s **difficult** to plan the work and define a **strategy to reduce technical debt** because it is **not a part of the regular development process** which focuses on implementing features.

    (BUCH, 2011)
• **Technical Debt x Scrum**

  • It’s not clear **who is responsible for the reduction of technical debt**: the Team, Product Owner, or the Scrum Master?

  • **Product Owner** often doesn’t understand the need of reducing technical debt and doesn’t allow it in backlog.

  • Technical debt **is not structured and documented** in the projects.

(BUCH, 2011)
This is how the source code looks like when we don't pay off technical debt.
HOW TO PAY THE DEBT?
Carolyn Seaman and Yuepu Guo proposed a Technical Debt management framework.

**TD (Technical Debt) list**: represents the tasks that may cause future problems if not completed.

(UMBC, 2015)
Technical Debt Identification and Measurement

<table>
<thead>
<tr>
<th>ID</th>
<th>Technical debt identification number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Technical debt identification date</td>
</tr>
<tr>
<td>Responsible</td>
<td>Person who identified the technical debt.</td>
</tr>
<tr>
<td>Type</td>
<td>Testing (missing test cases, not executed test cases, or missing test plans), Defect (known latent defects that have not been fixed), Documentation (missing, outdated, or incomplete documentation), Design (an imperfection of the software’s design or architecture negatively affecting future maintenance)</td>
</tr>
<tr>
<td>Location</td>
<td>Description of where the debt item is.</td>
</tr>
<tr>
<td>Description</td>
<td>Justification of why that item needs to be considered.</td>
</tr>
<tr>
<td>Estimated Principal</td>
<td>Work required to pay off the TD item.</td>
</tr>
<tr>
<td>Estimated Interest Amount</td>
<td>Extra effort needed in the future if the TD item is not paid off at the moment of its identification.</td>
</tr>
<tr>
<td>Estimated Interest Probability</td>
<td>Probability of extra work needed, if the TD item is not paid off in the future.</td>
</tr>
</tbody>
</table>
Should some debt be paid down in the next sprint?

1. **Extract all debts associated** with the Sprint goal.

2. **Re-evaluate high/medium/low** estimates for these items.

3. **Perform numeric estimates** for all items with **HIGH Interest Probability** and **HIGH Interest Amount**.

4. **Compare Cost (Principal)** with Benefit (Interest Probability multiplied by Interest Amount). Eliminate any items for which the benefit does not outweigh the cost.

5. **Add up the estimated principal** for all items left after step 4.
Evaluate the application of the technical debt management framework through an action research in the real context of software projects using Scrum.
Action Research:
- A combination of scientific and practical objectives;
- To resolve real world problems in collaboration between researchers and participants;
- Little research has been carried out about the framework yet.

(SUSMAN et al., 1983)


**Case Selection**

- Ongoing projects with **frequently changing requests**;
- Projects using **Scrum** for project management;
- Projects with **evidence of debts not managed**.

---

<table>
<thead>
<tr>
<th>Project</th>
<th>Company</th>
<th>Software created in</th>
<th>Project begun in</th>
<th>Number of people in the project</th>
<th>Number of participants in the research</th>
<th>Project uses Scrum since</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle Quotations Management Software</td>
<td>SoftTwo</td>
<td>2013</td>
<td>2013</td>
<td>10</td>
<td>3</td>
<td>2013</td>
</tr>
</tbody>
</table>
Main Results

<table>
<thead>
<tr>
<th>ID</th>
<th>Date</th>
<th>Responsible</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>All (50%)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>All (67%)</td>
<td></td>
</tr>
</tbody>
</table>

First cycle: Identify Technical Debts and choose a method for visualization by the team.

<table>
<thead>
<tr>
<th>Technical Debt were identified</th>
<th>Types were created</th>
<th>Main Type found</th>
<th>Average time spent to complete the first 6 rows</th>
<th>Scrum role responsible for identifying debt in the project (questionnaire)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoftOne</td>
<td>46</td>
<td>Design (56%)</td>
<td>3 minutes</td>
<td>All (50%)</td>
</tr>
<tr>
<td>SoftTwo</td>
<td>16</td>
<td>Design (81%)</td>
<td>-</td>
<td>All (67%)</td>
</tr>
</tbody>
</table>
First cycle: Identify Technical Debts and choose a method for visualization by the team

- **SoftOne**: *Vtiger* and *Trello* tools.

- **SoftTwo**: *Jira* tool.
## Main Results

<table>
<thead>
<tr>
<th>SoftOne</th>
<th>Technical Debt were identified</th>
<th>Technical Debt were measured</th>
<th>Average time spent to complete the 3 rows about measurement</th>
<th>High difficulty in filling up the fields</th>
<th>Scrum role responsible for debt measurement in the project (questionnaire)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>46</td>
<td>12</td>
<td>10 minutes</td>
<td>Estimated Interest Amount (27%); Estimated Principal (17%); Estimated Interest Probability (17%)</td>
<td>Team (43%)</td>
</tr>
<tr>
<td>SoftTwo</td>
<td>16</td>
<td>5</td>
<td>8 minutes</td>
<td>Estimated Interest Amount (40%); Estimated Principal (20%); Estimated Interest Probability (40%)</td>
<td>Team (67%)</td>
</tr>
</tbody>
</table>

“I believe that the greatest difficulty has been to stipulate the estimated interest amount. It's hard to abstract what would be the extra effort especially for cases that have not yet occurred.”
Monitoring is difficult because it is manual, so it requires some time. As there are numerous activities in the project, having something to automate it would help a lot.

**Main Results**

<table>
<thead>
<tr>
<th>Technical Debt were measured</th>
<th>Technical debt that the Benefit exceeded the Principal</th>
<th>Items prioritized for the next Sprint</th>
<th>Justification</th>
<th>Scrum role responsible for technical debt monitoring (questionnaire)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoftOne</td>
<td>12</td>
<td>1</td>
<td>0</td>
<td>The measured items were not linked to features in the Sprint</td>
</tr>
<tr>
<td>SoftTwo</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>The team chose to continue paying interest for lack of time during the next Sprint</td>
</tr>
</tbody>
</table>
MAIN RESEARCH FINDINGS

Two difficulties:

1. To estimate the three metrics
   - The participants had more difficulty for filling up.
   - Main reason: probabilistic fields.

2. Lack of tools to make the integration between the technical debt measurements and their tracking chart.

We acted in the first difficulty.
Technical Debt Measurement

- 2 metrics: **Principal** and **Current Amount of Interest**.

- For every debt found during the development cycle, the **team** must register it and estimate its **Principal**.

- The team also registers the **value of accumulated interest to date**.

- Each time the debt happens, the **Current Amount of Interest** must be updated.
Technical Debt Monitoring

Should some debt be paid off in the next sprint?

1. Extract all debts associated with the Sprint goal.

2. Re-evaluate Principal numeric estimates for these items based on current plans for the upcoming release.

3. Compare Cost (Principal) with Benefit (Current Amount of Interest) and eliminate any items for which the benefit does not outweigh the cost.

4. Decide if this cost can be absorbed into the next release.
We asked SoftOne research participants to identify new debts and fill up the 7 rows about them.

**TD Identification and Measurement:**
- For **2 weeks**, **5 debts** were identified and measured.
- Average time to fill up was **6 minutes**.
- **46%** of the time spent before (13 minutes).

**Technical Debt Monitoring:**
- Both companies plan to adopt these steps (**4 steps**) in their next sprints.
In a Scrum project, most technical debt management activities should be accomplished by the team. However, in identification, all can contribute during the project.

About the framework, the difficulty concerned technical debt measurement. Thus, we used only two metrics that correspond to the actual values (not the probability) of debt in the project.

Another barrier is about the tools. It is important to use tools to make TD measuring and monitoring easier.

The Seaman and Guo’s framework is an important first step in the TD management, but with the changes proposed in the TD measurement, the framework tends to have greater acceptance, as it considers actual values.
YOUR FEEDBACK IS IMPORTANT!
Please register and leave a comment if you liked this session!
http://sched.co/3mZK

GET THE FULL PAPER FOR MORE DETAILS!

E-mail: frederico.fredy84@gmail.com
MANAGING TECHNICAL DEBT IN SOFTWARE PROJECTS USING SCRUM: AN ACTION RESEARCH