“SOFTWARE PROCESS IMPROVEMENT THROUGH THE LEAN MEASUREMENT (SPI-LEAM) METHOD”

ARTICLE AUTHORS: KAI PETERSON, CLAES WOHLIN

DESCRIPTION AUTHOR: JOOST OREMUS, GROUP 3 (KARL)

INTRODUCTION

The SPI-LEAM Method developed by Petersen (PostDoc, Software Engineering Research Lab (SERL), Blekinge Institute of Technology, Sweden) and Wohlin (Professor of Software Engineering, Blekinge Institute of Technology, Sweden) is designed to continuously improve the performance of software processes by combining the quality improvement paradigm (QIP) (Basili & Green, 1994) with lean software development practices. The goal of the method is to “(1) enable continuous software process improvement leading to a lean software process; and (2) avoid problems related to resistance of change by improving in a continuous manner.” (Petersen & Wohlin, 2010).

Lean principles focus on the elimination of waste during production processes. When applying lean principles to software development some examples of eliminating waste are: reduction of the time it takes to produce a product, reducing problems with the quality of the product and the reduction of change requests. QIP focuses on improving quality and consists of 6 steps (fig 1). Petersen & Wohlin (2010) apply lean measurement to steps 2 and 5 of QIP (fig 1).

Thus the main activities of the SPI-LEAM method (fig 2) are setting quantifiable goals and measurements and the analysis of the situation with the use of lean measurement. Step 2 involves measuring individual inventories and then combining them with each other and with quality measurements. To find the goals and measurement it is recommended to use the goal-question-metric approach (Basili, 1985) (fig 3). Step 5 consists of the analysis of the current situation and how to improve the current situation. The use of systems thinking as a method is advised to conduct complex analyses. The results of the SPI-LEAM method are the abilities to assess the performance of the development process and the ability to take continues action to make sure the process is more lean over time. The main deliverables are: inventory descriptions and levels (either high or low, see Appendix), process state diagrams and a situation analysis including improvement scenarios.

Figure 1 SPI-LEAM: Integration of QIP and lean principles (Petersen & Wohlin, 2010)
EXAMPLE

To illustrate how the SPI-LEAM method works this section contains an example of the method in practice. This example shows the steps you take to describe the (sub)inventory levels, combine them with a quality measurement and analyze the data to suggest improvements. The example only describes QIP Step 2 & 5 because this is where the SPI-LEAM method differs from QIP.

QIP PHASE 2: SET QUANTIFIABLE GOALS AND MEASUREMENTS

The first step is to describe the individual inventory levels. For analyzing purposes only five individual inventories are used with a minimum of one inventory focusing on quality dimensions. The individual inventories can be derived from several sub inventories. This is done by using the Goal-questions-metrics approach (Basili, 1985) (table 1).

The next step is measuring individual inventories. Because in software development you cannot just count the inventory in the way you would do in a factory, the inventory is counted by measuring the effort to implement for example a change request. To count the effort to implement a change request an impact analysis can be used (Lindvall & Sandahl, 1998).

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Specification</th>
</tr>
</thead>
</table>
| Goals     | • Enable continuous software process improvement leading to a lean software process.  
            • Avoid problems related to resistance of change by improving in a continuous manner. |
| Questions | • Q1: What is the performance of the development process in terms of inventories?  
          • Q2: What is the cause of performance problems? |
| Metrics   | • Requirements (individual inv.)  
            – High level req. (sub-inv.)  
            – Detailed req. (sub-inv.)  
            – Req. in design and impl. (sub-inv.)  
            – Req. in test (sub-inv.)  
            • Test cases (individual inv.)  
            – Unit test (sub-inv.)  
            – Function test (sub-inv.)  
            – Integration test (sub-inv.)  
            – System test (sub-inv.)  
            – Acceptance test (sub-inv.)  
            • Change requests (individual inv.)  
            – CR under review (sub-inv.)  
            – Approved CRs (sub-inv.)  
            – CRs ready for impact analysis (sub-inv.)  
            – CRs in test (sub-inv.)  
            • Faults and failures (individual inv.)  
            – Internal faults and failures (test)  
            – External faults and failures (customer)  
            • Faults-slip-through (quality)  
            – Req. review slippage  
            – Unit test slippage  
            – Function test slippage |

Table 1 Goal-questions-metrics for SPI-LEAM (Petersen & Wohlin, 2010)
Step three is to describe the individual inventory levels, which is done at the sub inventory level of the inventory. In this example the individual inventory is Change Request and the sub inventory is for example Change Requests in Test. To assess if the inventory is high or low the effort for each sub inventory is compared with the capacity of the company. A radar diagram is an easy way to show if inventory levels are high or low (Fig 3).

![Radar Chart Change Requests](image)

**Figure 3 Radar Chart Change Requests**

The next step is simulating overloads to determine the thresholds for critical overload situations. This step is followed by setting the inventory level to high or low (High=above capacity, Low=below capacity). If most sub inventories are labeled high, the inventory level is set to high. In Figure 3 only one of four sub inventories is high (i.e. CR under review) thus the inventory level can be set to low. However inventories are not independent so this has to be taken into account when setting the level to low or high.

Finally the state of the process is determined by combining the individual inventories states (High or Low) with the quality measurement. Quality effort can be determined by measuring the effort of fixing faults. The result of all these steps are a description of the values of the inventory levels rated as high or low and a radar chart showing the average efforts and capacities of each individual inventory.

**QIP PHASE 5: ANALYZE DATA AND RECOMMEND IMPROVEMENTS**

This part of the method is about understanding the current situation and finding the best ways to improve the situation. If for instance all individual inventory states are labeled high this part of the method is focused on finding the easiest way to improve the process to where most inventories are labeled low. With the support of systems thinking as a method analysis can be made to improve the situation. Fig 4 shows a simplified example of state changes to improve the inventory situation. In this example only four states are shown. The example follows the example of the Change requests inventory: (CR’s under review = High, Approved CR’s = High, CR’s ready for impact analysis = High, CR’s in tests = Low). As you can see decision one is the best way to improve the current situation. After using decision one to improve the inventory situation the action and the lessons learned should be described to be used as data for different improvement scenarios.
The Process-Deliverable Diagram (PDD) is a modeling technique to provide a clear view of the SPI-LEAM method. The creation of the PDD is described by Weerd & Brinkemper (2008). Within the PDD the Roles, Activities and Deliverables are modeled in diagrams and described in tables.

The model (figure 5) shows the two phases of the SPI-LEAM method and their activities: 1) Set Quantifiable Goals and Measurements by measuring the individual inventories and 2) Analyze Data and Recommend Improvements by analyzing the process states (depicted by the gray boxes on the left side). The right side of the model shows the concepts associated with the activities which are further explained in the tables.
Figure 5 Process-Deliverable Diagram
### The Activity Table

<table>
<thead>
<tr>
<th>Activity</th>
<th>Sub Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measure individual inventories</td>
<td>Describe inventories</td>
<td>The project manager needs to select and describe a maximum of five INVENTORIES relevant to their software process. Each INVENTORY can consist out of multiple SUB INVENTORIES.</td>
</tr>
<tr>
<td>Measure inventories</td>
<td></td>
<td>In this step each SUB INVENTORY is measured individually considering the effort required for each item in the INVENTORY.</td>
</tr>
<tr>
<td>Describe inventory levels</td>
<td></td>
<td>Each SUB INVENTORY effort is compared to the capacity to determine whether the inventory level is either in high or low.</td>
</tr>
<tr>
<td>Simulate overload</td>
<td></td>
<td>A simulation has to be performed to determine the thresholds for critical, high and overload situations of SUB INVENTORY.</td>
</tr>
<tr>
<td>Set inventory level</td>
<td></td>
<td>With the knowledge of when there are overload situations the INVENTORY STATE can be determined. If the majority of STATES are high, the INVENTORY STATE is set to high.</td>
</tr>
<tr>
<td>Determine process state</td>
<td></td>
<td>PROCESS STATE is determined by combining states with QUALITY measurements.</td>
</tr>
<tr>
<td>Analysis of process states</td>
<td>Analyse current situation</td>
<td>The consultant performs a SITUATION ANALYSIS by analysing the current PROCESS STATE and the INVENTORY STATES.</td>
</tr>
<tr>
<td></td>
<td>Improve current situation</td>
<td>The consultant provides different IMPROVEMENT SCENARIOS that can be used to improve the current situation. These scenarios are part of the SITUATION ANALYSIS.</td>
</tr>
</tbody>
</table>

Table 2 Activity Table
## THE CONCEPT TABLE

<table>
<thead>
<tr>
<th>Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVENTORY</td>
<td>Partially done work that is not (yet) implemented in software engineering is called INVENTORY (Poppendieck &amp; Poppendieck, 2003)</td>
</tr>
<tr>
<td>SUB INVENTORY</td>
<td>The SUB INVENTORY is a detailed part of the INVENTORY. For example: SUB INVENTORIES 'detailed requirements' and 'requirements in test' are detailed parts of the INVENTORY 'requirements'. (Petersen &amp; Wohlin, 2010)</td>
</tr>
<tr>
<td>PROCESS STATE</td>
<td>Combination of the INVENTORY state and QUALITY to model the state of the entire process</td>
</tr>
<tr>
<td>QUALITY</td>
<td>QUALITY is determined by the Fault-Slip-Through measurement (are the right faults found in the right place) (Damm et al., 2006) and the number of INVENTORY faults.</td>
</tr>
<tr>
<td>SITUATION ANALYSIS</td>
<td>SITUATION ANALYSIS models the reasons for the current situation and describes the best improvement alternatives to arrive at an improved state (Petersen &amp; Wohlin, 2010)</td>
</tr>
<tr>
<td>IMPROVEMENT SCENARIOS</td>
<td>Documents and models with scenarios that can be used to improve the current situation</td>
</tr>
</tbody>
</table>

Table 3 Concept Table
**RELATED LITERATURE**

Peterson & Wohlin based their SPI-LEAM method mostly on the quality improvement paradigm (QIP) (Basili & Green, 1994) in combination with lean development (translated to software engineering by Poppendieck & Poppendieck (2003)). QIP is a process improvement framework like the Capability Maturity Model Integration (CMMI) (Chrissis, Konrad, Shrum, 2003). Lean principles are derived from efficiency efforts dating back to the days of Henry Ford in the early 20th century when Ford tried to build cars more efficiently and the just in time principles introduced by Toyota (Shingo & Dillon, 1989). The field of lean processes focuses on the elimination of waste during production processes. When applying lean principles to software development some examples of eliminating waste are: reduction of the time it takes to produce a product, reducing problems with the quality of the product and the reduction of change requests. In an article by Middleton (2001) the importance of reducing waste due to partially done work (called inventory) is explained.

Lean software development has shared principles with agile practices especially practices like SCRUM and are easily combined with each other, but there is a clear difference. Where agile focuses on customer involvement and rapid delivery of software, lean software development mainly focuses on waste elimination. (Hibbs, Jewett & Sullivan, 2009) The SPI-LEAM method gives companies the freedom to choose their own software development method like Scrum but can also be used with more rigid software development methods like waterfall (Petersen & Wohlin, 2010).

So far only the book Modern Software engineering concepts and practices (Dogru & Bicar, 2011) mention the SPI-LEAM method as an example of value stream maps and inventory management but they do not further investigate the method. More research has to be done to further develop the method into a practical solution.

**REFERENCES**


APPENDIX
TEMPLATE of Deliverable sub-inventory state:

<table>
<thead>
<tr>
<th>Sub Inventory?</th>
<th>State High (+) or LOW (-)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub-INV0</td>
<td></td>
</tr>
<tr>
<td>Sub-INV1</td>
<td></td>
</tr>
<tr>
<td>Sub-INV2</td>
<td></td>
</tr>
<tr>
<td>Sub-INV3</td>
<td></td>
</tr>
<tr>
<td>Sub-INV4</td>
<td></td>
</tr>
<tr>
<td>Sub-INV5</td>
<td></td>
</tr>
</tbody>
</table>