Theory Based Software Engineering with the SEMAT Kernel: Preliminary Investigation and Experiences

Pan-Wei Ng
My Background and Objective

- I am a coach – help teams and organization improve
- Making theory practical to practitioners
What is Theory? Why talk About Theory?

• Wikipedia: **Theory** is a group of ideas meant to explain a certain topic, such as a single or collection of fact(s), event(s), or phenomenon(s).
• My version: Theory is a set of statements that relates variables and cause and effect.
• Why talk about theory? It is all about improvement, becoming better at what we are doing and to help others become better too
  – Explain (the theory) how we become better
  – Adapt the theory to another but similar context

Einstein: “Theory defines what we observe (behave)”
Light hearted analogy: World Cup Fever (1/3)

- Soccer: Goal is to get the ball between the posts?
- How should I kick? How much force? Which angle?

The theory

I followed your theory, it did not work in practice.
Light hearted analogy: World Cup Fever (2/3)

• Soccer: Goal is to get the ball between the posts?
• How should I kick? How much force? Which angle?

The reality (considering drag)

I followed your theory, it did not work in practice
Light hearted analogy: World Cup Fever (3/3)

- Soccer: Goal is to get the ball between the posts?
- How should I kick? How much force? Which angle?

The reality (don’t forget human factors)

I see, but what should I do now?

Human factors are separate and distinct from physics, but still affects outcome.
Software development (engineering) is complex with many factors.

“Theory”

Practice (Reality)

Different stages
Different problems

Kurt Levin: “There is nothing as practical as good theory”
What is Essence? Two Main Ideas

1. Alphas and states – aspects of progress and health

   - Stakeholders: Recognized, Represented, Involved, In Agreement, Satisfied for Deployment, Satisfied in Use
   - Opportunity: Identified, Solution Needed, Value Established, Viable, Addressed, Benefit Accrued
   - Requirements: Conceived, Bounded, Coherent, Acceptable, Addressed, Fulfilled
   - Software System: Architecture, Selected, Demonstrable, Usable, Ready, Operational, Retired
   - Team: Seeded, Formed, Collaborating, Performing, Adjourned
   - Work: Initiated, Prepared, Started, Under Control, Concluded, Closed

2. Separation of concerns – Methods are a composition of practices on top of the kernel

   Team’s Method = Kernel + Requirements Elicitation Practice + Acceptance Testing Practice + Practices from various sources (e.g. industry)

Separation of concerns apply to theories too
Underlying foundations of process improvement theories

- Van Hilst and Fernandez’s Pattern System of Underlying Theories of Software Process Improvement (2010)

- States as value stream flow
- Organize factors Through alphas
- Agile & iterative approach
- Precise Practices
- Alpha states as a plan/strategy
Theory Based Software Engineering

- **Architecture Views from perspective of alphas**
- **Context Description**
  - describes the context of
  - gives context to
- **Objectives and Factors**
- **Software Engineering Endeavor**
  - impacts
  - relates and explains
  - validates and tunes
  - recommends
- **Recommended Practices**
  - change structure and behaviors
  - and improves maturity of
  - impacts
  - relates and explains
- **Specific Theories**
  - General Theories
- **Separation of concerns between theories**

- **IVAR JACOBSON INTERNATIONAL**

Creating winning teams.
Steps to TBSE

1. Identify which aspect to improve
   - Alphas for selecting area(s) of improvement
2. Describe context – architecture views
   - Structural versus dynamic view
   - Gives factors context
3. Theorize the relationship between factors and outcomes
   - Specific theory and general (background) theory
4. Make recommendations
   - Recommendations affect factors
5. Act and observe behaviors
   - They may work according to or against recommendations
6. Validate/Tune the theory
Case Study: Knowledge Management System

• **Area to improve: Stakeholder and Opportunity Management**
  – Symptoms: too many requirements, implemented requirements not being used by end-user community, development overload

• **Architecture Views: Structural and Behavioral**

  ![Diagram showing stakeholder and opportunity management structures and processes]

  - Structural views constructed based on instance of alphas
  - Behaviors defined through views of alpha states

Creating winning teams.
Identify success factors and recommendations

- Identified factors grow as more observations are made

<table>
<thead>
<tr>
<th>Customer (Stakeholders)</th>
<th>Solution (Requirements)</th>
<th>Endeavor (Team)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Stakeholder Engagement</td>
<td>Team Engagement</td>
</tr>
<tr>
<td></td>
<td>Stakeholder Participation</td>
<td>Required Effort by Team</td>
</tr>
<tr>
<td>Communication</td>
<td>Development Responsiveness</td>
<td></td>
</tr>
<tr>
<td>timeliness</td>
<td></td>
<td></td>
</tr>
<tr>
<td>honesty and completeness of information</td>
<td></td>
<td></td>
</tr>
<tr>
<td>empathy and equity of treatment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>power</td>
<td></td>
<td></td>
</tr>
<tr>
<td>urgency of request</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Regular prioritization meetings
Open ROI prioritization rules

Powerful stakeholders do not like to be prioritized (negative behavior)
Reduce development lead time from three to one month
Simplify effort estimation

IVAR JACOBSON INTERNATIONAL
Creating winning teams.
Monitor and observe

- In the ideal world, everything works out as planned or recommended
- Negative behaviors often highlight missing factors or oversight in some aspects

+ Innovating
  - Adapting
    - Embracing
    - Rejecting
  - Refine prioritization rules
  - Simplify effort estimation
  - Regular prioritization meetings
  - Open ROI prioritization rules
  - Powerful stakeholders do not like to be prioritized (negative behavior)

- Creating winning teams.
Lessons learnt (1/2)

• Essence is an attractive candidate for organizing context descriptions and factors
  – Check assumptions from different aspects / perspectives
• Objective is to gradually and systematically engage practitioners to “theorize” their approach to development and process improvement
  – Theory defines what you observe
  – Specific theory versus general/background theory (assumptions)
  – Context is important
    • Organize and describe context
Lessons learnt (2/2)

- TBSE is very much like Systems Thinking
  - Actually, it is built on the underlying pattern system of process improvement (Van Hilst and Fernandez)
  - Differentiators:
    - having an agreed domain model (Essence) to begin with
    - architecture descriptions to give further context
    - Specific versus background theories (separation of concerns applied to theory)
- Training practitioners to “theorize” is challenging
  - It is not something they do naturally
  - They want answers fast (“you tell me” syndrome)
- Still very much work in progress
• Thank you