On Whose Shoulders We Stand: Theory testing or theory building in requirements engineering research

Keynote REFSQ Doctoral Consortium
Sjaak Brinkkemper
Utrecht University
Outline

1. Your contribution a theory?
2. Theory development
3. Theory in Design Science
4. Example of Theory building research
5. Conclusion and discussion

For this presentation I stand on the shoulders of many colleagues.
A PhD: Your contribution to the sciences

- A PhD work builds upon all the work done in a particular domain.
- In Computer Science we design innovative technology, and we forget our theoretical contributions.
- Poor habit, so our colleagues in other sciences are looking down on us.

If I have seen further than others, it is by standing upon the shoulders of giants.

(Isaac Newton)
Your hypothesis

Conceptual model

Examples:
- Executing systems development in sprints improves stakeholder satisfaction
- Establishing linguistic relationships simplify requirements engineering in high volume requirements management
What is a theory?

- A theory is a set of propositions about an object of study.
- A proposition consists of concepts and specifications of relations between concepts.
- The relations are assumed to be true for the object of study.
- The set of instances to which the relations apply is called the domain, i.e. the field in which the proposition can be generalized.

Four essential characteristics of a theory:

1. Object of study
2. Concepts
3. Propositions
4. Domain
Discussion: what is your theory?

Name the four essential theory characteristics of your project:
1. Object of study
2. Concepts
3. Propositions
4. Domain

What are the Dependent and Independent Concepts?
Outline

1. Your contribution a theory?
2. Theory development
3. Theory in Design Science
4. Example of Theory building research
5. Conclusion and discussion
Theory Development

Empirical cycle

- Theory building research is research with the objective of formulating new propositions based on the empirical evidence drawn from observation of instances of the object of study.
- Theory testing research is research with the objective of testing propositions.
Research strategies for Theory building
Types of causal propositions

1. Sufficient: If there is A, then there will be B
2. Necessary: B exists only if A is present
3. Deterministic: If A is higher, then B is higher
4. Probabilistic: If A is higher, then it is likely that B is higher

Discussion: what are good examples of these causalities in our field?

What applies for RE research?
Contributions with new techniques and tools are usually focusing on Deterministic and Probabilistic causality of the efficiency or effectiveness of RE work.
# Research strategies for theory testing

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Preferred</th>
<th>Case study</th>
<th>Survey</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sufficient condition</td>
<td>Experiment</td>
<td>Single-case study</td>
<td>Third-best</td>
</tr>
<tr>
<td>Necessary condition</td>
<td>Experiment</td>
<td>Single-case study</td>
<td>Third-best</td>
</tr>
<tr>
<td>Deterministic relation</td>
<td>Experiment</td>
<td>Longitudinal single-case study or</td>
<td>Third-best</td>
</tr>
<tr>
<td></td>
<td></td>
<td>comparative case study</td>
<td></td>
</tr>
<tr>
<td>Probabilistic relation</td>
<td>Experiment</td>
<td>Third-best: comparative case study</td>
<td>Second-best</td>
</tr>
</tbody>
</table>
Outline

1. Your contribution a theory?
2. Theory development
3. Theory in Design Science
4. Example of Theory building research
5. Conclusion and discussion
Theory in design science?

- Theories of the usage and performance of newly designed artifacts to understand, explain or enhance aspects of information technology in general

- **Usage**: changes for the users, e.g. their way of working
- **Performance**: change of properties, e.g. speed, integration
- **Artifact**: Interface, method, tool, specification language, ...

- Measure improvements, gain insight in the way work is performed
- Application in medical and technological sciences, including information and computing science
Design cycle

Operational principles and design theories

Theory contribution

Awareness of problem

Suggestion

Development

Evaluation

Conclusion
Design and its context

External context

Internal context

interface

solution

Example: Wifi Streetlight

[Faculty of Science, Information and Computing Sciences]
Approach

1. analysis
2. design
3. implement
4. evaluate
Design Research Cycle

Abstraction domain

Analysis → Design

Problem domain

Problem ↔ Solution
IST and SOLL

Abstraction domain

Analysis -> Design

IST -> SOLL

Problem domain

Problem <-> Solution
Design levels

Abstraction domain

Analysis → Design

Problem domain

Problem ← Solution

conceptual level

fysical level
Outline

1. Your contribution a theory?
2. Theory development
3. Theory in Design Science
4. Example of Theory building research
5. Conclusion and discussion
Theory building: Speeding up Requirements Management in a Product Software Company

- **External Context**: Baan with 8000 customers and 15,000 product installations
- **Internal Context**: 60 Product managers responsible for the requirements management processes
- **Problem**: linking of customer wishes to product components is too cumbersome
- **Idea**: Use linguistic engineering techniques to link customer wishes to product requirements

- **Joint work of**:
  - Björn Regnell and Johan Natt och Dag, Lund Institute of Technology, Sweden
  - Vincenzo Gervasi, Pisa University, Italy
  - Sjaak Brinkkemper, Utrecht University, The Netherlands
Approach: what is the problem?
Context: organisation “Baan”

- Founded in 1978 in the Netherlands
  - 15,000+ customer sites worldwide
  - 1000+ employees
  - 1,000,000+ users
- Markets
  - Manufacturing industry
  - Engineering industry
- Headquartered in Barneveld, The Netherlands
  - Offices and partners worldwide
  - 400 employees Development
  - 200 employees Customer Service & Support
- Now part of Infor
Complexity of large scale product software

- Several thousands of wishes, needs, and requirements stored in a database
- Market- and technology-driven software development
- Large, complex software systems
- Several different product lines
- Distributed development
- Uncertainty, frequent change, and time pressure
- Requirements written in plain text
A customer wish related to current or future markets, defined using the terminology and context of the customer.

A generic product specification to be covered by Baan solutions described in Baan’s terminology and context.
Example requirements

<table>
<thead>
<tr>
<th>Field</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>MR10739</td>
<td></td>
</tr>
<tr>
<td>Example</td>
<td>[Request raiser’s company]</td>
<td></td>
</tr>
<tr>
<td>Request Person</td>
<td>[Request raiser]</td>
<td></td>
</tr>
<tr>
<td>Date</td>
<td>1996-05-29</td>
<td></td>
</tr>
<tr>
<td>Label</td>
<td>Pricing and Containerization</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Specifically what I am interested in is containerization and pricing. For a prospect I am working with (pretty much a distributor of electronic components) I need pricing by type of package by customer type (wholesale or retail). I think pricing by container solves this problem, but I understand to use this feature the item must be a process item and I don’t know if this is good or bad. If I must use process what do I gain or lose, like do I have to run a separate MRP etc. Do I have to have one process company and one non-process company. They have mainly an assembly operation with no process involved. If process would be to cumbersome how difficult a mod would it be to disconnect containerization from process.</td>
<td></td>
</tr>
<tr>
<td>Keywords</td>
<td>Pricing, order planning</td>
<td></td>
</tr>
<tr>
<td>Priority</td>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Functionality</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>Closed/Completed</td>
<td></td>
</tr>
<tr>
<td>User name</td>
<td>[Requirement submitter]</td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>020699: functionality is available in BaanERP in the Pricing module</td>
<td></td>
</tr>
<tr>
<td>Agreement</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>1. Container (end item) in statistics Purchase and sales statistics used to be maintained only at main item level. But now it has also become possible to build statistics at container level. There are two aspects: printing statistics in the number of containers for a main item selecting and/or printing statistics at container level 2. Displays in statistics Displays are compositions of end items (for example, an attractive display of different types of cake). The statistics will be updated at both the levels of display item and container (which is part of the display). Prevention of duplicate counting, and correct pricing must be arranged in a procedural manner.</td>
<td></td>
</tr>
<tr>
<td>Keywords</td>
<td>Process industries</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Usability</td>
<td></td>
</tr>
<tr>
<td>Status</td>
<td>Assigned</td>
<td></td>
</tr>
<tr>
<td>User name</td>
<td>[Requirement submitter]</td>
<td></td>
</tr>
<tr>
<td>Comments</td>
<td>Warehousing only</td>
<td></td>
</tr>
</tbody>
</table>
Approach – step 1: analysis

1. analysis

Problem domain

Problem

Abstraction domain

Analysis
# The Baan RDB

<table>
<thead>
<tr>
<th>Year</th>
<th># Business Requirements</th>
<th># Linked</th>
<th># Market Requirements</th>
<th># Linked</th>
</tr>
</thead>
<tbody>
<tr>
<td>1996</td>
<td>0</td>
<td>0</td>
<td>183</td>
<td>113</td>
</tr>
<tr>
<td>1997</td>
<td>5</td>
<td>4</td>
<td>683</td>
<td>262</td>
</tr>
<tr>
<td>1998</td>
<td>275</td>
<td>169</td>
<td>1,579</td>
<td>388</td>
</tr>
<tr>
<td>1999</td>
<td>709</td>
<td>261</td>
<td>2,028</td>
<td>502</td>
</tr>
<tr>
<td>2000</td>
<td>669</td>
<td>167</td>
<td>1,270</td>
<td>397</td>
</tr>
<tr>
<td>2001</td>
<td>1,000</td>
<td>153</td>
<td>864</td>
<td>224</td>
</tr>
<tr>
<td>2002</td>
<td>1,121</td>
<td>340</td>
<td>1,695</td>
<td>514</td>
</tr>
<tr>
<td>Total</td>
<td>3,779</td>
<td>1,094</td>
<td>8,302</td>
<td>2,400</td>
</tr>
</tbody>
</table>
Linking statistics

3,779 Business Requirements
1,094 linked

8,302 Market Requirements
2,400 linked
Approach – step 2: design

Abstraction domain

Analysis

2. design

Design

Problem domain

Problem
Tactics: possible solutions

- Add more resources
  - Too costly!

- Reduce the amount of incoming requirements
  - Not a good idea for customer relationship!

- Let customers do the linking
  - Plans of new releases may not be shown externally!

- ...

Idea: use linguistic techniques to find similar requirements

Stop word removal
Stemming
Tokenization
Flattening

Similarity calculation

Stop word removal
Stemming
Tokenization
Flattening

Suggestions

Market Requirements
Business Requirements

Requirements Database
ongoing, continuous
Linguistic Engineering approach

<table>
<thead>
<tr>
<th>Flattening</th>
<th>Tokenization</th>
<th>Lemmatisation</th>
<th>Stop word removal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storing multiple charts in one file.</td>
<td>storing multiple charts in one file</td>
<td>store multiple chart in one file</td>
<td>store multiple chart file</td>
</tr>
</tbody>
</table>

The user should be able to store charts in a single (.SOV) file

\[ \sigma(r_m, r_b) = \frac{\sum_i [1 + \log_2 r_m(i)] \cdot [1 + \log_2 r_b(i)]}{\sqrt{\sum_i 1 + \log_2 r_m(i)} \cdot \sqrt{\sum_i 1 + \log_2 r_b(i)}} \]
# Suggestion list

<table>
<thead>
<tr>
<th>Pos</th>
<th>Requirement</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BR10012</td>
<td>0.45</td>
</tr>
<tr>
<td>2</td>
<td>BR10156</td>
<td>0.43</td>
</tr>
<tr>
<td>3</td>
<td>BR10006</td>
<td>0.42</td>
</tr>
<tr>
<td>4</td>
<td>BR10536</td>
<td>0.38</td>
</tr>
<tr>
<td>5</td>
<td>BR10987</td>
<td>0.36</td>
</tr>
<tr>
<td>6</td>
<td>BR10273</td>
<td>0.36</td>
</tr>
<tr>
<td>7</td>
<td>BR10740</td>
<td>0.34</td>
</tr>
<tr>
<td>8</td>
<td>BR10419</td>
<td>0.33</td>
</tr>
<tr>
<td>9</td>
<td>BR10622</td>
<td>0.24</td>
</tr>
<tr>
<td>10</td>
<td>BR10082</td>
<td></td>
</tr>
</tbody>
</table>
Design issue: What is the optimal top list size?

<table>
<thead>
<tr>
<th>Pos</th>
<th>Requirement</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BR10012</td>
<td>0.45</td>
</tr>
<tr>
<td>2</td>
<td>BR10156</td>
<td>0.43</td>
</tr>
<tr>
<td>3</td>
<td>BR10006</td>
<td>0.42</td>
</tr>
<tr>
<td>9</td>
<td>BR10622</td>
<td>0.24</td>
</tr>
<tr>
<td>10</td>
<td>BR10982</td>
<td>0.21</td>
</tr>
</tbody>
</table>

Suppose we restrict the suggestions to the top list:
- How many candidate requirements are in the top list?
- How many candidate requirements are missed?
- How long is an optimal top list?
Theory testing in text retrieval: Found and correct? – Confusion matrix

TP: True Positives
FP: False Positives
FN: False Negatives
TN: True Negatives

Recall = Found correct / All Correct = \( \frac{TP}{TP + FN} \)
## Manual related to automated

<table>
<thead>
<tr>
<th>Pos</th>
<th>Requirement</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BR10012</td>
<td>0.45</td>
</tr>
<tr>
<td>2</td>
<td>BR10156</td>
<td>0.43</td>
</tr>
<tr>
<td>3</td>
<td>BR10006</td>
<td>0.42</td>
</tr>
<tr>
<td>4</td>
<td>BR10536</td>
<td>0.38</td>
</tr>
<tr>
<td>5</td>
<td>BR10987</td>
<td>0.36</td>
</tr>
<tr>
<td>6</td>
<td>BR10273</td>
<td>0.36</td>
</tr>
<tr>
<td>7</td>
<td>BR10740</td>
<td>0.34</td>
</tr>
<tr>
<td>8</td>
<td>BR10419</td>
<td>0.33</td>
</tr>
<tr>
<td>9</td>
<td>BR10622</td>
<td>0.24</td>
</tr>
<tr>
<td>10</td>
<td>BR10882</td>
<td></td>
</tr>
</tbody>
</table>

Manually linked and presumed correct
## Recall related to top list size

<table>
<thead>
<tr>
<th>Pos</th>
<th>Requirement</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BR10012</td>
<td>0.45</td>
</tr>
<tr>
<td>2</td>
<td>BR10156</td>
<td>0.43</td>
</tr>
<tr>
<td>3</td>
<td>BR10006</td>
<td>0.42</td>
</tr>
<tr>
<td>4</td>
<td>BR10536</td>
<td>0.38</td>
</tr>
<tr>
<td>5</td>
<td>BR10987</td>
<td>0.36</td>
</tr>
<tr>
<td>6</td>
<td>BR10273</td>
<td>0.36</td>
</tr>
<tr>
<td>7</td>
<td>BR10740</td>
<td>0.34</td>
</tr>
<tr>
<td>8</td>
<td>BR10419</td>
<td>0.33</td>
</tr>
<tr>
<td>9</td>
<td>BR10622</td>
<td>0.24</td>
</tr>
<tr>
<td>10</td>
<td>BR10082</td>
<td></td>
</tr>
</tbody>
</table>

Recall(7) = \( \frac{2}{3} = 67\% \)
Approach – step 3: implement
Implementation

- Linguistic functionality was coded and inserted into the Baan RDB
- A pilot was run first to check whether extension was according quality standards
- Product managers were trained to use the functionality
Approach – step 4: evaluate
A comparative cost-benefit evaluation

<table>
<thead>
<tr>
<th>Subset providing 100% recall using a top-10 list</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BRs</td>
<td>690</td>
</tr>
<tr>
<td>The MRs linked by product managers</td>
<td>1,249</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Manual search</th>
<th>~30 hits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assume 1 search term is enough</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Automatically supported</th>
<th>10 hits</th>
</tr>
</thead>
</table>

Savings: ~66% or ~115 hours
Potential next steps

- Aggregate similarity measures using other techniques
- Reuse information in already linked requirements
- Incorporate semantics from names of software modules
- Expert validation in concept similarities
Outline

1. Your contribution a theory?
2. Theory development
3. Theory in Design Science
4. Example of Theory building research
5. Conclusion and discussion
Conclusion

- Theory building or theory testing is an overlooked issue in PhD project design.
- Most RE research work is theory building, where the design is the theory.
- Design science research requires a variety of research methods for validating the design.

Discussion