Putting industrialization at work: developing Normalized Systems in practice

Martin Op 't Land & Arco Oost
Summary

• Several **case studies** in developing Normalized Systems for different domains show remarkable **results** in terms of time-to-market, code quality and especially flexibility, providing indications of the relevance and feasibility of using more theory-based approaches in IT systems development.

• Also experiences with using **organizational building blocks** according to DEMO in the domains of grants, patents and HRM, as a pre-cursor to Normalized Systems development, show promising ways of achieving new levels of business-IT “loose coupling” in an industrial way.
Who are we?

Martin Op 't Land
- Certified Enterprise Architect, Innovation Alliance Developer, Capgemini
- Professor Enterprise Engineering, Antwerp Management School
- Practitioner and researcher on splitting and allying of (extended) enterprises

Arco Oost
- Senior Technical Consultant, Capgemini
- Expert programmer on Normalized Systems Expanders, NSX
- Specialized in refactoring & redesign
- Practitioner in advanced programming concepts (a/o DSL’s, frameworks, ...)

We like to advance enterprise agility by connecting organization granularity with ICT granularity

May 8th, 2012
Putting industrialization at work: developing Normalized Systems in practice
Content

• what is Normalized Systems (NS)?
• my hesitations in October 2009 ...
• case Defense
• case Telesur
• case NS Hispeed
• prototype DEMO-NS
• considerations on an NS adoption process
• benefits / conclusions
• future developments
What is Normalized Systems (NS)?

Content

- **Normalized Systems** (NS) = systems that can be changed without *combinatorial effects*
  - one type of anticipated change will only propagate in one module
  - a given maintenance action on an NS-system costs (even formally) proven exactly as much impact now, or two years later (when the NS-system is, say, five times as large and complex)

- NS theory and tools developed by University of Antwerp (UA)

Benefits

- productivity improvement in development
  - factor 2-3 has been proven realistic
- maintenance improvement even better, because NS = top quality software

“Our insight in what *software quality* comprises is changed to such an extent that *we can no longer imagine to develop good software without Normalized Systems.*” (5 well-seasoned developers of Capgemini, 2011)
Background: NS under the hood – 4 Principles, 5 Elements, 8-10 cross-cutting concerns

4 Principles

- SoC: Separation of Concerns
- DVT: Data Version Transparency
- AVT: Action Version Transparency
- SoS: Separation of State

5 Elements

- Data
- Action
- Workflow
- Connector
- Trigger

X-cutting concerns

With NS, an Application = n instantiations of Elements, fully separated from each other ⇒ no Combinatorial Effects (CE), ex ante proven*

My hesitations in October 2009 ...

• Is Normalized Systems somewhat like the Holy Grail, *too good to be true*?
  - will it perform?
  - granularity as key for agility is OK; where is the trade-off?
• so we started a controlled adoption process ...

<table>
<thead>
<tr>
<th>adoption activity</th>
<th>period</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>awareness session 1</td>
<td>2009-October</td>
<td>10 specialists not able to shoot NS</td>
</tr>
<tr>
<td>content orientation</td>
<td>2010</td>
<td>4 developers reading, 1 MSc-student (thesis)</td>
</tr>
<tr>
<td>awareness session 2</td>
<td>2010 July</td>
<td>10 specialists tested again; mgt-choice: go!</td>
</tr>
<tr>
<td>educating developers</td>
<td>2010 December</td>
<td>5 developers educated in NS; NS assessment</td>
</tr>
<tr>
<td>business case</td>
<td>2011-H1</td>
<td>ROI, development method, adoption road map</td>
</tr>
<tr>
<td>tool industrialization</td>
<td>from 2011–H2</td>
<td>refactoring, new features</td>
</tr>
<tr>
<td>applying the tools</td>
<td>from 2011-H1</td>
<td>MoDefense, Telesur, NS HiSpeed, ...</td>
</tr>
<tr>
<td>building the NS Alliance</td>
<td>2011 May 2012-H1</td>
<td>Memorandum of Understanding with NSA parties Contract with NSX</td>
</tr>
<tr>
<td>developing methodology</td>
<td>2011 May - ...</td>
<td>EEWC-2011, PRET-2011, EEWC-2012 ...</td>
</tr>
</tbody>
</table>
Proof of Concept NS at Dutch Ministry of Defense

Edward van Dipten, Chief Operational Architecture

• Question: find out hands-on the potentials of Normalized Systems
  - agility, productivity, time-to-market, quality

• Way of Working (WoW)
  - Scrum method during 4 sprints of one week
  - Information Product Management as subject

• Results
  - system size: 256 Function Points (NESMA FPA, double-blind)
  - measured productivity improvement: factor 2 – 2.5
  - flexible, both for functional and technical changes, until the last moment
  - evolved vision on WoW for NS in combination with DEMO
    • enables agile working by production quality results after every sprint
Proof of Concept NS at Dutch MoD: Conclusions

Edward van Dipten, Chief Operational Architecture

C2SC

- NS proved to have the basic elements for quickly changeable and mission critical applications
- NS can support a way of working where the business process and object model evolve during the realization of the application. Possibility for direct reflection from users
- For military use it is important to have graphical and geographical oriented NS expanders
- DEMO helps to get a good insight about involved actors, transactions, responsibilities and information necessary for a successful execution of a business process

Capgemini

- First project in which Capgemini applied NS in practice as productivity accelerator
- Discovered the potential of carefully splitting up business analysis work from building customizations ⇒ enabler for Rightshore®

University of Antwerp

- “Further evidence of NS approach
- Increasingly successful knowledge-transfer NS → industry
- industry feed-back ⇒ future enhancements of NS expanders”
Proof of Concept NS at Dutch MoD: Lessons learned

Edward van Dipten, Chief Operational Architecture

- fast feed-back modeling $\leftrightarrow$ implementation works excellent
  - creates a sort of *Extreme Data Modeling*
- the one day NS course (UA) helps the customer organization
  - understand NS principles, understand results of NS audit
- insert a small design workshop in the beginning of the project
  - clarifying models, creating common language etc.
- increase sprint period (1 week $\Rightarrow$ 2 weeks)
  - less overhead, more time for finishing functionality
- maintain at least 1 physical contact moment customer – developer per week
- planned improvements on NS expanders would add lots of value
  - testability, productizing, clearer separation of customizations, data validation

For more details, see paper

*Exploring Normalized Systems Potential for Dutch MoD’s Agility: A Proof of Concept on Flexibility, Time-to-market, Productivity and Quality* (PRET-2011)
Case Telesur

- prototype system for Job Application
- used DEMO models as a foundation
- needed to augment an OpenERP system for the Suriname situation
  - different requirements in salary calculations, not available in OpenERP
- extras were developed in Normalized Systems
  - features don’t have to be ported when newer version of OpenERP arrive
- OpenERP was coupled through views in the Normalized System
  - loose coupling (SoC)
  - coupling confined to a single component
- simple reporting demonstrated with Jasper Reports
- coordination
Case Telesur – some metrics

- NS part
  - application with 80 screens, 160 Java files
  - 2 components, 10 data elements
- NS part built in 10 man days
  - 4 days for the base application
  - 6 man days for customization, e.g. screens & reporting
    (lessons already incorporated in latest version of tooling)
- 2 hours for bridging between databases underlying OpenERP & NS
Case NS Hispeed: NS as integration technology (back-end only)

- data on delays and train mutations are provided through a CORBA server
- CORBA interaction had to be written by hand
- Normalized Systems used to develop the part that manages the database
- metrics
  - 4 days to connect to the CORBA server (all kinds of problems)
  - 1 day to set up the Normalized Systems part
- during acceptance testing
  - lots of changes to the front end
  - no changes to the Normalized System/Corba part
Connecting DEMO to NS: bridging organizational with ICT system granularity

- Normalized Systems can be built, taking common system requirements as a starting point (e.g., data model, mockup screens, use cases)
- this already creates $\Delta$(ICT system requirements)$\equiv\Delta$(NS system-change)
  - the 8 anticipated changes of NS are in terms of $\Delta$(ICT system requirements)
- how to create $\Delta$(organization change)$\equiv\Delta$(ICT system requirements)?
  - start with DEMO as a language for organizational essence at a moment
  - map DEMO systematically and coherently to
    - organizational implementation (assigning organizations/functionary types, separation of functions etc.)
    - ICT system requirements (mockup screens, screenflows, data model, etc.)
  - clarify $\Delta$(DEMO model) $\rightarrow$ $\Delta$(ICT system requirements) $\rightarrow$ $\Delta$(NS system-change)
- to create such an industry strength mapping DEMO-NS, ongoing research is performed by UA, TU Delft and Capgemini
  - earlier work (e.g., patents, grants) by Huysmans, Krouwel, Op ‘t Land
  - today a sneak preview on work in progress ... a prototype DEMO-NS
Putting industrialization at work: developing Normalized Systems in practice

Organization Construction Diagram

Transaction Result Table

<table>
<thead>
<tr>
<th>Transaction kind</th>
<th>Transaction result</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-T01 rental start</td>
<td>B-R01 [rental] has been started</td>
</tr>
<tr>
<td>B-T02 rental end</td>
<td>B-R02 [rental] has been ended</td>
</tr>
</tbody>
</table>
## DEMO-NS prototype EU-Rent

Mockup screen for B-T01 rental start

### Request Car Rental Start

**Renter**

- **Name:** Mr. T.H.E. Renter
- **Address:** Transaction path 1
- **ZIP-code:** 4444TR
- **Residence:** DEMO Town

### Requested Car

<table>
<thead>
<tr>
<th>Car Group</th>
<th>Car Brand</th>
<th>Car Type</th>
<th>Rental rate per day</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>Citroen</td>
<td>C1</td>
<td>€ 31</td>
</tr>
<tr>
<td>Small</td>
<td>Peugeot</td>
<td>107</td>
<td>€ 32</td>
</tr>
<tr>
<td>Medium</td>
<td>Opel</td>
<td>Astra</td>
<td>€ 45</td>
</tr>
<tr>
<td>Medium</td>
<td>Ford</td>
<td>Focus</td>
<td>€ 46</td>
</tr>
<tr>
<td>Medium</td>
<td>Volkswagen</td>
<td>Golf</td>
<td>€ 49</td>
</tr>
<tr>
<td>Medium</td>
<td>Peugeot</td>
<td>306</td>
<td>€ 47</td>
</tr>
<tr>
<td>Medium</td>
<td>Citroen</td>
<td>C1</td>
<td>€ 48</td>
</tr>
</tbody>
</table>

- **Start Date:** 2012/04/02
- **End Date:** 2012/06/01
- **Pick-up Branch:** Amsterdam
- **Intended Drop-off Branch:** Madrid

---

©2012 Op ’t Land & Oost
DEMOS-NS prototype EU-Rent
Mockup screen for B-T01 rental start
Typical research questions

• what happens when actor roles are combined/split across functionary types?
  – e.g., request is typed by renter on webpage himself – can internal screen be reused?
  – process flow, mock-up screens, screenflow

• what happens when order of working in workflow is changed?

• how can the universal transaction pattern (rq, pm/dc, st, ac/rj, all with potential revokes, etc.) be supported best?

For more details, we can send you published and recently submitted articles on this subject
Considerations on an NS adoption process

- crucial: complete transparency, exact understanding, right expectations
- two types of parties consider adopting Normalized Systems:
  - **WhiteBox** party: open to understand NS, have own developers educated in NS, have a development factory based on NS and the NS Expanders
  - **BlackBox** party: wants to know the principles and added values (e.g., costs, speed, ensured maintainability & agility, openness) to make informed choices in their RFP processes for development and/or lifecycle solutions
- **WB/BB** parties: different ambition level, different stakeholders in adoption

### Typical adoption step

<table>
<thead>
<tr>
<th>Step</th>
<th>BB</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. one hour explanation + Q&amp;A</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>b. one day NS awareness day + one hour management explanation</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>c. 4-6 week Proof of Concept project</td>
<td>?</td>
<td>x</td>
</tr>
<tr>
<td>• goal: assess added value of NS for own organization ⇒ business case</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• develop (production quality) software ⇒ solve well-scoped &amp; real problem</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d. educating own developers; specialized course, MEITA (AMS/DelftTopTech)</td>
<td>-</td>
<td>x</td>
</tr>
<tr>
<td>e. implement NS development factory</td>
<td>-</td>
<td>x</td>
</tr>
<tr>
<td>f. apply NS quality criteria as black-box testable criteria, to be used in RFP</td>
<td>x</td>
<td>-</td>
</tr>
</tbody>
</table>
Conclusions

- Future proof
  - Isolation from underlying frameworks
  - Algorithms and data are separated out
  - Beneficial for maintenance and continuity

- Development cycle
  - Granularity allows for easier customization
  - Rapid expansion of elements allows quick feedback
  - Faster development, not just in green-field environments

- Consolidation
  - Expert knowledge is captured in the expanders
  - When new insights are gained, existing code can benefit as well (SoC)

- Theory gives insights in what choices to make
  - E.g. Separation of Concerns, gives insight into (hidden) couplings

- Normalized Systems gives applications a very stable foundation
Future developments

- Easier modeling tools, e.g. Prime Radiant (Q2012-Q3)
- New platforms, e.g., .Net (2013)
- Integration of Business Rules
  - Probably using external Business Rule Engines
- New descriptors
  - User interface elements
  - Descriptors for interconnection
    - Web services
    - Richer client-side models (e.g. Microsoft’s MVVM)
- Further improvements on
  - Speed of development
  - Lowering costs of maintenance
- Lots more ;-)
Any questions?

Thanks for your attention!

Martin.OptLand@capgemini.com & Arco.Oost@capgemini.com