Design and prototypical implementation of a language empowering business users to define Key Performance Indicators for Enterprise Architecture Management

Student: Thomas Reschenhofer
Advisor: Ivan Monahov
Supervisor: Prof. Florian Matthes

Date: 26.11.2012
Type: Guided Research
Term: SS 2012

Software Engineering for Business Information Systems (sebis)
Technische Universität München, Germany
wwwmatthes.in.tum.de
What is EA? What is EAM? What are KPIs?

• **EA:** Enterprise Architecture
  – “Fundamental organization of a system [enterprise] embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution”. [ISO Standard 42010]

• **EAM:** Enterprise Architecture Management
  – Maintaining the EA’s flexibility, efficiency, and transparency by
  – developing, implementing and **controlling** the EA’s components and relations

• **KPI:** Key Performance Indicator
  – Metric, which measures quality and performance aspects
  – Evaluation of goal achievement
Motivation

- **EAM KPI Catalog**
  - 52 well-documented KPIs
  - Supports enterprise architects
  - Requires a certain EA data model

- **EAM tools**
  - Methods for gathering the EA model’s data
  - EA modeling techniques
  - Visualization guidelines

Lack of implementation and tool integration

No support for formal definition and automated computation of EAM KPIs

Tricia eXpression Language (TxL)
Exemplary KPI of the catalog

• Application continuity plan availability

  “A measure of how completely IT continuity plans for business critical applications have been drawn & tested up for the IT's application portfolio”

  – Information model:

    [Diagram]

  – Calculation:

    The number of critical applications where tested IT continuity plan available divided by the total number of critical applications

  – Possible interpretation:

    100%  80%  60%  0%
    Good  Normal  Problematic
Requirements & research questions

• Identified requirements for EAM KPI Catalog tool integration:
  – **RQ1**: EAM tool has to provide a flexible data model to adapt it to the catalog’s needs
  – **RQ2**: The language has to provide a sufficient expressiveness to define all the catalog’s KPIs
    ➢ Language requirements were already examined in another research
  – **RQ3**: Business users must be able to define and visualize KPIs at runtime

• Emerging research questions:
  – Are these requirements met by existing EAM tools and wiki systems?
  – Which language properties are relevant for the design of a DSL to support the computation of the catalog’s KPIs?
  – What is a possible implementaiton of such a DSL?
Existing EAM tools and wiki systems

• 9 examined EAM tools:

• 4 examined wiki systems:

Neither the examined EAM tools, nor the wiki systems, fulfil the three requirements RQ1, RQ2, and RQ3

➤ Design of TxE and prototypical implementation
Design of TxL

• TxL is a domain-specific language (DSL)
  – Object-oriented | Sequence-oriented | Functional | Reflective | Strictly evaluated | Dynamically typed

• Data types in TxL
  – Simple types: Object | String | Number | Boolean | Date
  – Constructor types: Sequence | Map | Function | Entity

• Operators in TxL
  – Arithmetic operators: add, sub, mul, div, ...
  – Conditional and logical operators: equals, greaterThan, and, or, not, ...
  – Query operators: select, where, selectMany, orderby, ...
  – Aggregation operators: count, sum, max, min, first, ...
Formal definition of an exemplary KPI

• Reminder: The calculation of the KPI Application continuity plan availability is informally described as:

The number of critical applications where tested IT continuity plan available divided by the total number of critical applications

• A formal definition by TxL could be:

```scheme
/* Determine all critical business applications */
let criticalApplications = find("business application","is critical","yes") in

/* Determine all critical business applications with tested IT continuity plan */
let criticalApplicationsWithCoveringContinuityPlan =
    criticalApplications .where(?(ca)(ca["covering continuity plan"].isNotNull()
        ? ca["covering continuity plan"].first()["is tested"]
        : false)) in

/* Calculate proportion of critical business applications */
criticalApplicationsWithCoveringContinuityPlan.count()
    .div(criticalApplications.count())
```
Prototypical implementation of TxL

• Integration in our research tool *Tricia*

![Diagram showing integration between Tricia core, platform plugin, script plugin, derived attributes, and embedded script]

• Three logical components:
  – **Script core**
    TxL Interpreter | TxL execution environment | Extensibility mechanism | User interface enhancements
  – **Embedded script**
    Embedding of single TxL expressions into rich text content
  – **Derived attributes**
    Definition of an attribute, whose value is derived from other attribute values
Definition of a KPI as custom TxL function

Each KPI of the catalog can be defined as custom function:

<table>
<thead>
<tr>
<th>Type</th>
<th>getApplicationContinuityPlanAvailabilityKPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>getApplicationContinuityPlanAvailabilityKPI</td>
</tr>
<tr>
<td>Parameters</td>
<td>no parameters</td>
</tr>
<tr>
<td>Description</td>
<td>A measure of how completely IT continuity plans for business critical applications have been drawn &amp; tested up for the ITs application portfolio</td>
</tr>
</tbody>
</table>
| Method Stub   | // Determine all critical business applications  
                let criticalApplications =  
                    find("business application","is critical","yes") in  
                // Determine all critical business applications  
                // with tested IT continuity plan  
                let criticalApplicationsWithCoveringContinuityPlan =  
                    criticalApplications  
                    .where(?ca)(ca["covering continuity plan"].isNotNull()  
                        ? ca["covering continuity plan"].first()["is tested"]: false)) in  
                // Calculate proportion of critical business applications  
                criticalApplicationsWithCoveringContinuityPlan.count()  
                    .div(criticalApplications.count()) |

Invocation of KPI function by name:

getApplicationContinuityPlanAvailabilityKPI()
HTML-based visualization of KPIs

• Based on the custom function and the capability to embed TxL expressions into rich text, business users are able to define suitable KPI visualizations based on HTML-markup:

```javascript
/* Evaluate KPI value by custom function */
let kpivalue = getApplicationContinuityPlanAvailabilityKPI() in

/* Interpretation: Good if >80%, normal if 60% - 80%, problematic if <60 */
kpivalue.greaterThanOrEqualTo(0.8)
  ? "<img src='trafficlight-green.jpg'/>"
  : kpivalue.greaterThanOrEqualTo(0.6)
    ? "<img src='trafficlight-yellow.jpg'/>"
    : "<img src='trafficlight-red.jpg'/>"
```

Application continuity plan availability: 1

| Application continuity plan availability: 1 | or | Application continuity plan availability: 0.5 |
Conclusion & Outlook

• Conclusion
  – Existing EAM tools and wiki systems do not fulfil the identified requirements
  – Therefore, we designed TxL and implemented a prototype, which fulfils these requirements
  – All KPIs of the catalog were successfully implemented

• Outlook
  – Evaluation
  – Authorization concept
  – Time series
  – Visualizations
Thanks for your attention!

Questions?

Student: Thomas Reschenhofer
Advisor: Ivan Monahov
Supervisor: Prof. Florian Matthes

Date: 26.11.2012
Type: Guided Research
Term: SS 2012

Software Engineering for Business Information Systems (sebis)
Technische Universität München, Germany
wwwmatthes.in.tum.de