



Andreas Lux

16.03.2010

ex|Xcellent
solutions

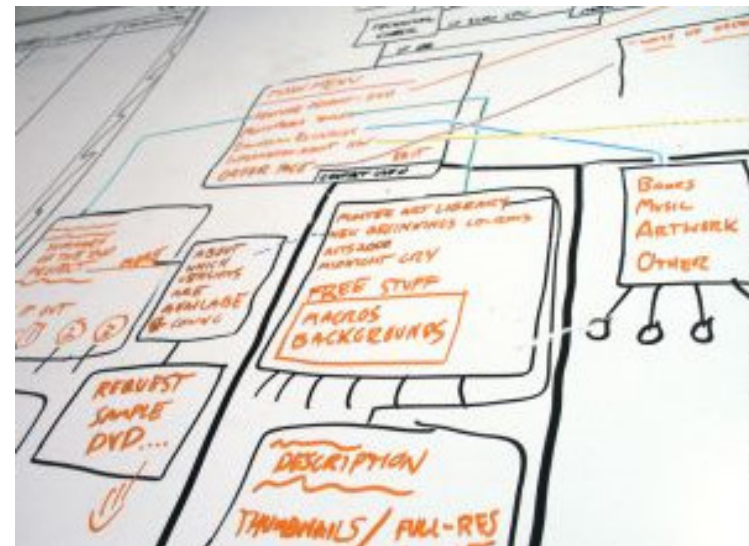
**Verknüpfung unterschiedlicher
Modellsprachen (BPMN, UML, DSL)
zur Anforderungsanalyse**



Nicht alle Probleme eignen sich, um mit Standardsprachen beschrieben zu werden

Gründe können sein:

- x| zu umfassend
- x| nicht konkret genug
- x| nicht sprechen genug
- x| ineffizient in der Modellierung
- x| zu aufwändig bei der Integration von Generatoren



-> Die Lösung!?: **DSL**
(**D**omain **S**pecific **L**anguage)

Grafische Modellierung

- x| Darstellung netzartiger Zusammenhänge (Flüsse, Strukturen)
- x| sinnvolle Darstellung von abstrakten Sichten



Textuelle Modellierung

- x| Sie aus wie Freitext, ist aber Modell!
- x| Klare Strukturen
- x| Einfache Sprache (wenige Regeln)
- x| Einfache Editierbarkeit

x| DSL ist in eclipse ein separates Plugin

x| **Definition** der (fachlichen) Struktur

x| **Generierung** des Tree-Editors über EMF

x| Generierung des **eclipse Plugin Projekts**

x| **Deployment** des Plugins über eine **Updatesite**



-> „prinzipiell“ Verwendbar über den generierten Tree-Editor

Nachteil: relativ umständlich in der Handhabung

x| Grafischer Editor

- | Kann über GMF ebenfalls aus DSL heraus generiert werden
- | Generierung des eclipse Plugin Projekts
- | Deployment des Plugins über eine Updatesite

⌘ **Muss noch um Features zum Handling erweitert werden**

x| Textuelle DSL (am Beispiel xText)

- | Definition einer eigenen Grammatik
- | Generieren des ecore Modells über xText Generator
- | Deployment des Plugins über eine Updatesite

-> **Sehr effizient in der Anwendung, kann aber u. U. sehr unübersichtlich sein**

Beispiele für unterschiedliche Sprachen:

- x| UML:** sehr allgemeine und umfangreiche Modellierungssprache (in unterschiedlichen Ausprägungen!)
- x| BPMN:** Modellierungssprache für Prozesse / Abläufe
- x| SysML:** Systembeschreibungssprache basiert auf UML

- x|** Von der OMG gibt es diverse weitere Entwürfe für Domänenspezifische Sprachen

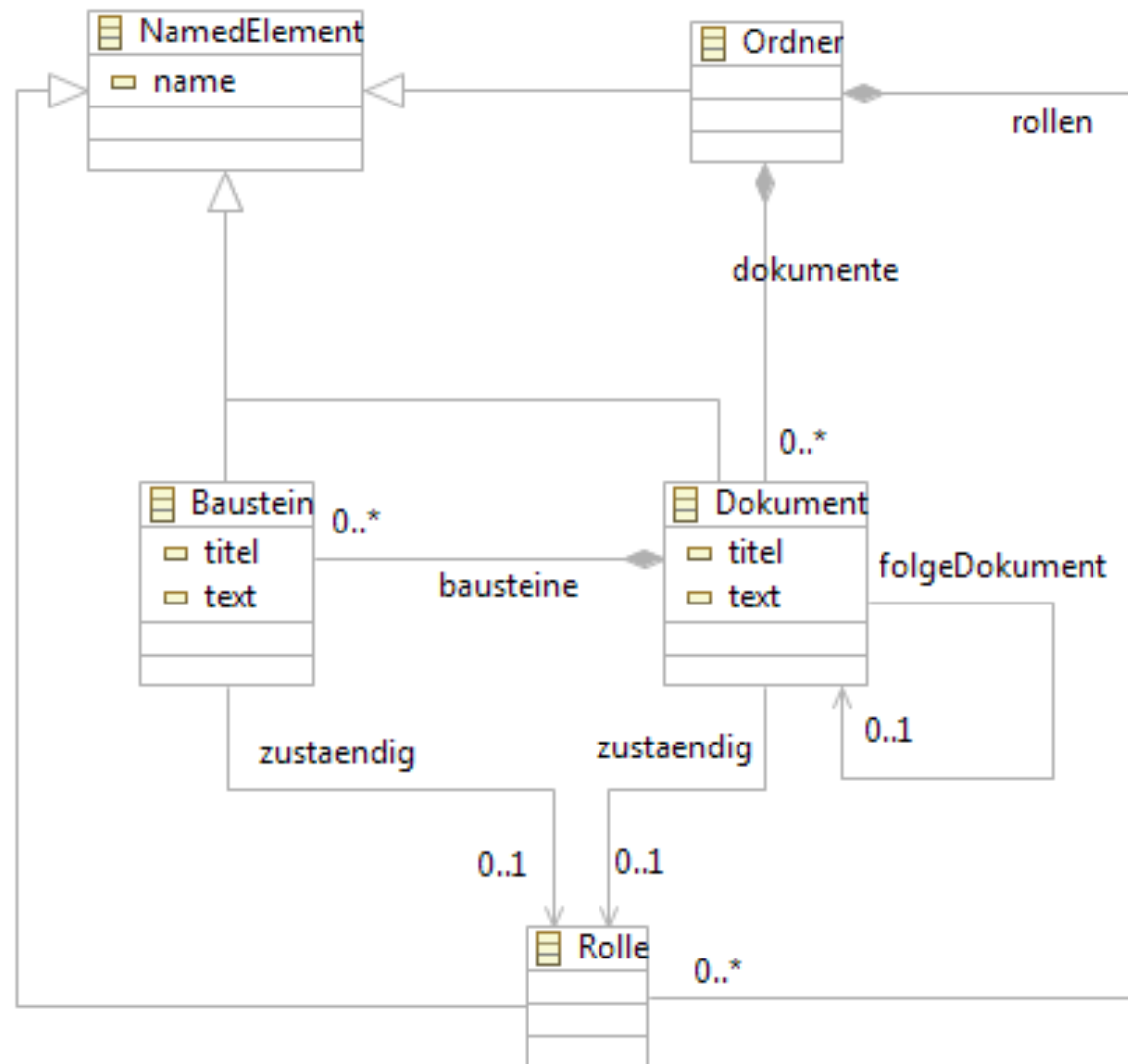
- x|** DSLs von eXXcellent solutions:
 - | orchideo|gui-prototype DSL
 - | orchideo|object DSL
 - | orchideo|aspect DSL
 - | orchideo|documents Templatesprache

- x| Es wird ein modellgetriebener Ansatz im Projekt verfolgt
- x| Ausdrucksmöglichkeiten bestehender Sprachen entsprechen nicht dem Modellierungsproblem weil diese:
 - | zu viele Freiheitsgrade (= Fehler) zulässt
 - | es für ein Problem mehrere Modellierungswege gibt
 - | zu umständlich zu modellieren ist
 - | zu wenig exakt ist
 - | nicht verständlich genug ist (sich nicht an der Problem-Domäne orientiert)

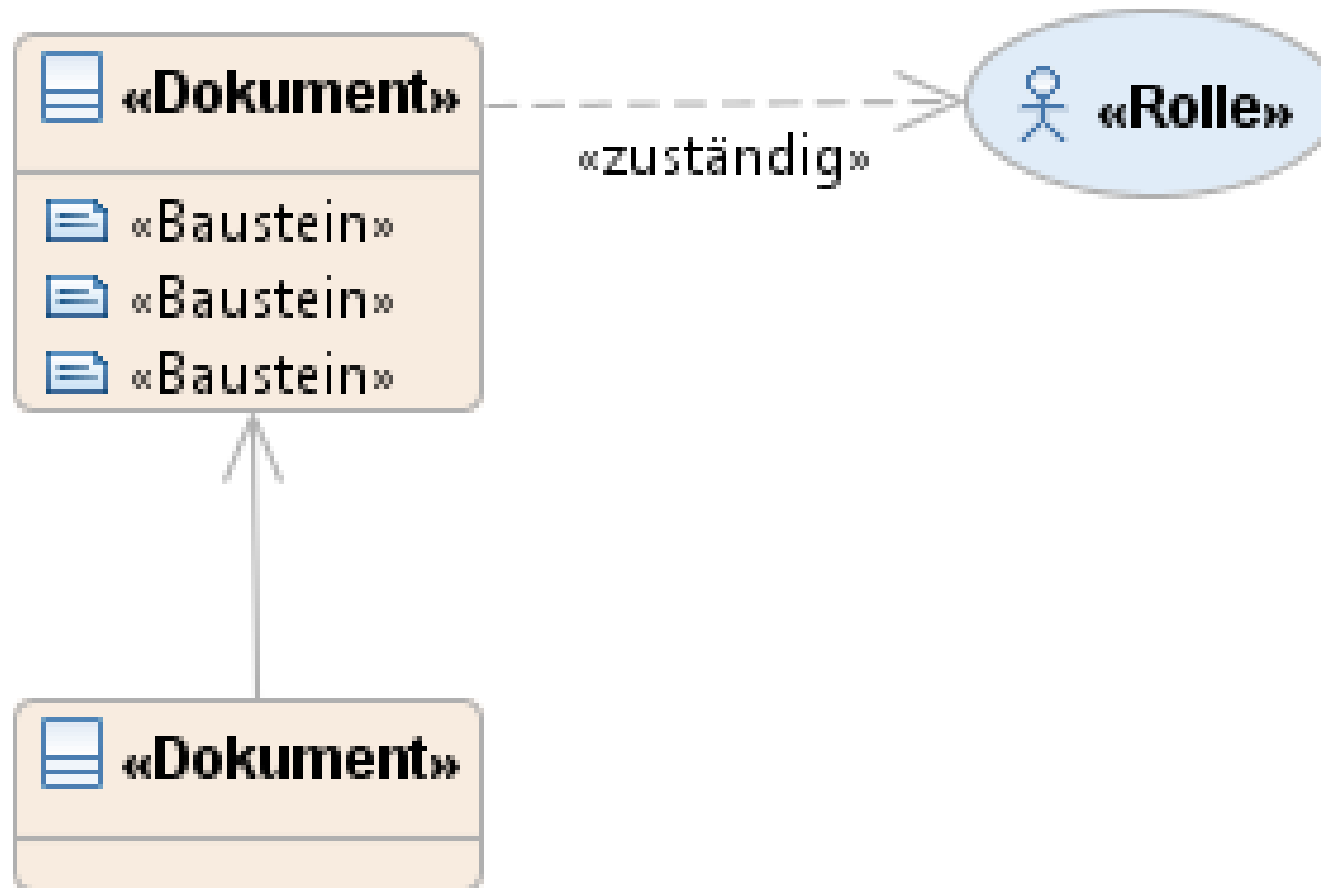


- x| Verwendung der Dokumentdefinition für eine Generator (nicht Teil der Präsentation)
- x| Es gibt unterschiedliche Dokumenttypen, die in unterschiedlichen Versionen vorliegen
- x| Dokumente bestehen aus Bausteinen
- x| Es gibt Zuständigkeiten pro Dokumenttyp und optional pro Baustein
- x| Die Versionshierarchie muss erkennbar sein
- x| Die Dokumenttypen soll grafisch modelliert
- x| Struktur und Inhalt der Dokumenttypen soll dokumentiert werden können



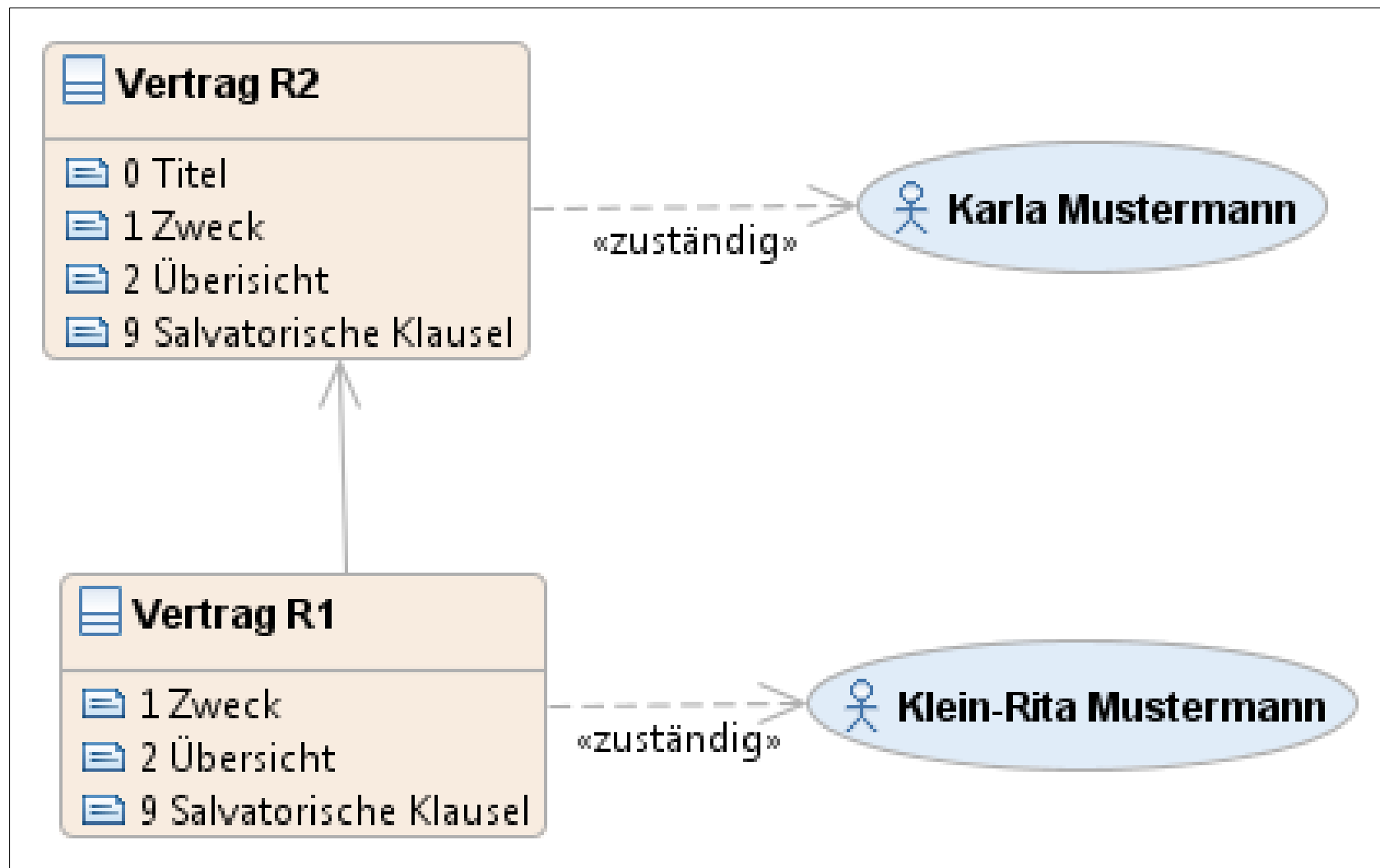


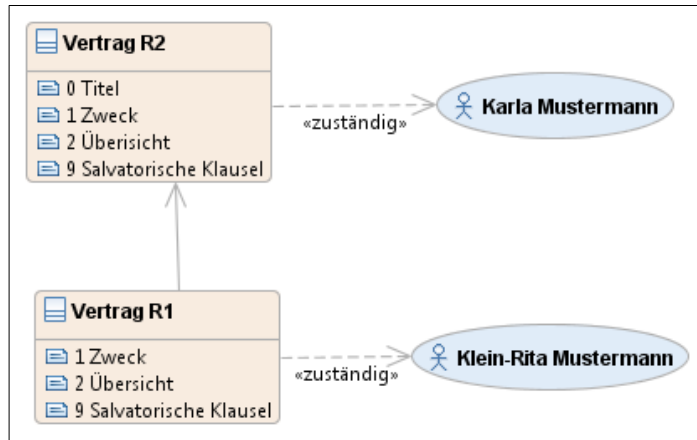
Wie sieht so was grafisch aus...



-> Beispiel 1 live in der orchideo|suite

Und am konkreten Beispiel





definition for

```
"http://www.excellent.de/orchideo/  
Dokuments/1.0/DocManager.ecore"
```

```
new Dokument "Vertrag R2"
```

```
new Baustein "0 Titel"
```

```
new Baustein "1 Zweck"
```

```
new Baustein "2 Übersicht"
```

```
new Baustein "9 Salvatorische Klausel"
```

```
end Dokument
```

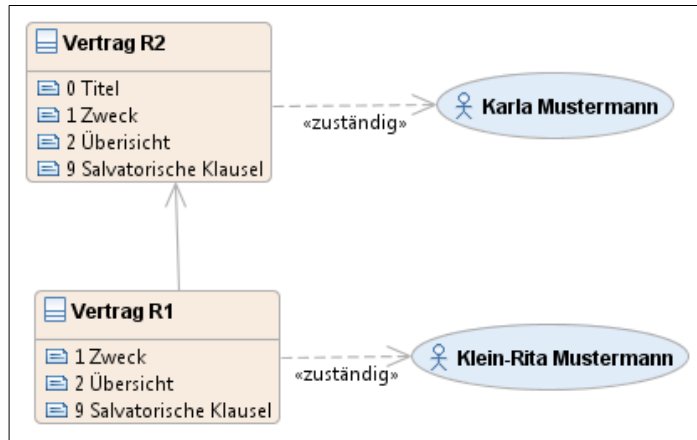
```
new Dokument "Vertrag R1"
```

```
new Baustein "1 Zweck"
```

```
new Baustein "2 Übersicht"
```

```
new Baustein "9 Salvatorische Klausel"
```

```
end Dokument
```



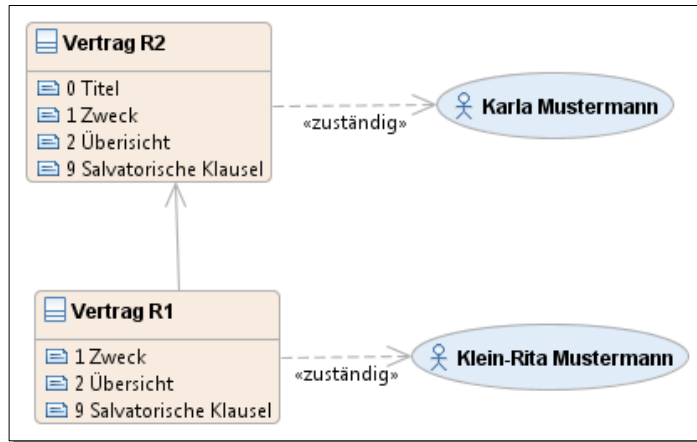
```
new Rolle "Karla Mustermann"
end Rolle
```

```
new Rolle "Klein-Rita Mustermann"
end Rolle
```

```
new Dokument "Vertrag R2"
  Zuständigkeit "Karla Mustermann"
  new Baustein "0 Titel"
  new Baustein "1 Zweck"
  new Baustein "2 Übersicht"
  new Baustein "9 Salvatorische Klausel"
end Dokument
```

```
new Dokument "Vertrag R1"
  FolgeDokument "Vertrag R2"
  Zuständigkeit "Karla Mustermann"
  new Baustein "1 Zweck"
  new Baustein "2 Übersicht"
  new Baustein "9 Salvatorische Klausel"
end Dokument
```

Textuelle Repräsentation 3



Eigenschaften		
Dokument		
Core	Eigenschaft	Wert
Appearance	Eigenschaften	
	Name	Vertrag V1
	Text	Dies ist ein längerer Text zum Vertrag
	Titel	Titel der Vertrags
Referenzen	Folge Dokument	Dokument Vertrag V2
	Zuständig	Rolle Klein-Rita Mustermann

```
new Rolle "Karla Mustermann"  
end Rolle
```

```
new Rolle "Klein-Rita Mustermann"  
end Rolle
```

```
new Dokument "Vertrag R2"  
  Zuständigkeit "Karla Mustermann"  
  new Baustein "3 Rahmenbedingungen"  
  new Baustein "1 Zweck"  
  new Baustein "2 Übersicht"  
  new Baustein "3 Rahmenbedingungen"  
  new Baustein "9 Salvatorische Klausel"  
end Dokument
```

```
new Dokument "Vertrag R1"  
  Titel: „Dies ist ein Titel zum aktuellen Vertrag“  
  FolgeDokument "Vertrag R2"  
  Zuständigkeit "Karla Mustermann"  
  new Baustein "1 Zweck"  
    Titel: „Dies ist ein Titel zum aktuellen Baustein“  
    Text: „Dies ist ein längerer Text Titel zum aktuellen Baustein.“
```

```
  new Baustein "2 Übersicht"  
    Titel: „Dies ist ein Titel zum aktuellen Baustein“  
    Text: „Dies ist ein längerer Text Titel zum aktuellen Baustein. D  
      Titel: „Dies ist ein Titel zum aktuellen Baustein“  
    Zuständigkeit "Karla Mustermann"
```

```
  new Baustein "9 Salvatorische Klausel"  
    Titel: „Dies ist ein Titel zum aktuellen Baustein“  
    Text: „Dies ist ein längerer Text Titel zum aktuellen Baustein.  
Dies ist ein längerer Text Titel zum aktuellen Baustein. Dies ist ein  
längerer Text Titel zum aktuellen Baustein. Dies ist ein längerer Text Titel  
zum aktuellen Baustein. Dies ist ein längerer Text Titel zum aktuellen  
Baustein. Dies ist ein längerer Text Titel zum aktuellen Baustein.“  
    Titel: „Dies ist ein Titel zum aktuellen Baustein“  
    Zuständigkeit "Karla Mustermann"
```

```
end Dokument
```



Werden spezielle DSLs im Projekt eingesetzt, sollten folgende Rahmenbedingungen eingehalten werden können:

- x|** Verwendung unterschiedlicher DSL innerhalb einer Umgebung
- x|** Erweitern von „fremden“ DSLs
- x|** Verknüpfen Modellinhalten aus unterschiedlichen DSL
- x|** Generieren von Dokumenten



Verwendung unterschiedlicher DSL innerhalb einer Umgebung

Warum ist das wichtig?

- x| DSL dienen zur Beschreiben **eines konkreten Problemtyps**
 - x| Viele andere Problemtypen können mit bestehenden Standard-DSLs (z.B. UML, BPMN, ...) bereits ausreichend gut abgebildet werden
 - x| Man will die Welt nicht neu „erfinden“
- > Die neue DSLs müssen sich strukturell mit bestehenden DSLs im Modellierungstool mischen und referenzieren lassen

-> Beispiel: UseCases und UI-Diagramme



Erweitern von „fremden“ DSLs

Warum ist das wichtig?

- x| Erweitern bestehender DSL um projektspezifische Informationen (z.B.: „Description“-Feld bei UML2Tools UseCases, ...)
- x| Erweitern von (technischen) DSL um fachliche Aspekte (z.B.: mitführen einer UseCase ID, Tracker ID, Freigabeinformationen, ...)
- x| Erweitern von DSL um organisatorische Aspekte (z.B.: Referenzen auf weitere Inhalte)

-> **Beispiel: UML2 Use Cases**



Verknüpfen Modellinhalten aus unterschiedlichen DSL

Warum ist das wichtig?

- x| Systembeschreibungen sind sehr oft hierarchisch aufgebaut
- x| Sprachelemente verfeinern sich oft in weiteren Sprachelementen (z.B. UseCase Activity Sequence, UseCase UI Diagramm)
- x| Sprachelemente referenzieren oft bestehende Definitionen (z.B.: Akteur Klasse, UseCase Service-Interface, „Dokument“ Klasse oder Klassendiagramm, ...)

-> **Beispiel: Use Case Overview**



Generieren von Dokumenten aus gemischten Modelle für die Dokumentation / Kommunikation

Warum ist das wichtig?

- | Es gibt sehr oft Beteiligte (Kunde?), die keinen Zugriff auf das Modellierungswerkzeug haben
- | Erzeugung von Snapshots, falls das Modellierungswerkzeug keine „echte“ Versionierung beherrscht
- | Reduktion der Komplexität durch spezialisierte Dokumente (ausblenden der für den Leser unwichtigen / nicht relevanten Informationen)
- | Reuse von Modellinformationen für generierte Dokumente zur Vollständigen Dokumentation von Einzelteilen

-> Beispiel: Dokument „Use Cases“

Was sind die Vorteile bei Einsatz von spezialisierten DSLs?

x| Allgemein:

- | Erhöhung der Effizienz
- | Wiederverwertbare Fachdefinitionen (auswertbar, generierbar, ...)
- | Exakte Beschreibung der Fachlichkeit

x| Grafische Modellierung:

- | Guter Überblick
- | Gute Abstraktion

x| Textuelle Modellierung:

- | sehr effizienter Editor



Was können die Nachteile sein?

- x| Zu hoher initialer Aufwand durch die Entwicklung der DSL und der zugehörigen Plugins
- x| Verlust der Allgemeingültigkeit der verwendeten Sprache (Kommunikation nach außen!)
- x| Zu hohe Aufwände bei der Wartung
- x| Vorgehen (noch) stark abhängig von der eingesetzten Toollandschaft



- x| Vorgehen ist dann sehr hilfreich, wenn die Modelle zur (Code-/Config) Generierung verwendet werden
- x| Die Einarbeitungsaufwände, um grafische Editoren „handhabbar“ zu machen, können sehr groß werden
- x| Einsatz von eigenen DSL für uns nur dann sinnvoll, wenn
 - | das Problem überhaupt abstrahiert werden kann
 - | Die Modelle (mehrfach) wiederverwendet werden können (z.B. Code **und** Doku)
 - | die Modellinformationen extern kommuniziert werden können (via Tool oder Doku)



Bleibt nur noch ein...

ex|Xcellent
solutions

Danke!

... und herzlich willkommen an unserem Stand

<http://www.exxcellent.de/>

