



Integration der domänenspezifische Sprache Movisa in den nutzerzentrierten Entwicklungsprozess der Ueware

Workshop: Modellbasierte Entwicklung von
Benutzungsschnittstellen

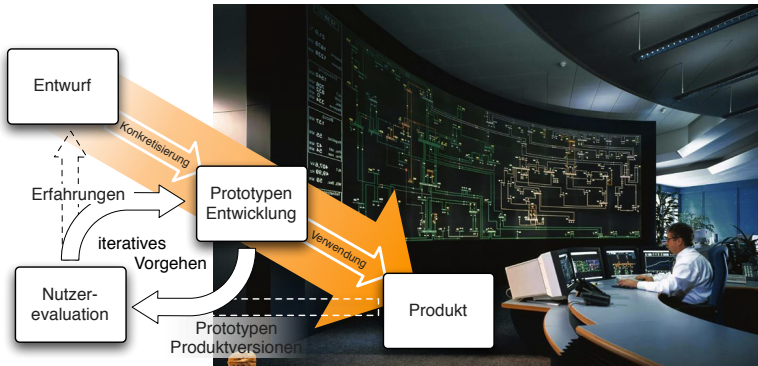
H. Hager, St. Hennig, M. Seißler, A. Braune

Dresden, 06.10.2011



Gebrauchstaugliche UIs

Nutzerzentrierte Entwicklung für die Automatisierungstechnik



Agenda

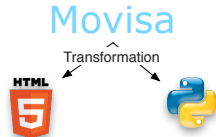
- ① **Einführung:** Movisa + Ueware-Entwicklungsprozess
- ② **Problemstellung:** Herausforderungen bei der Transformation
- ③ **Lösungsvorschlag:** Interaktive Transformation mit Petmap
- ④ **Auswertung**

01 Einführung

Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



01 Einführung

Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



Ueware-Entwicklungsprozess

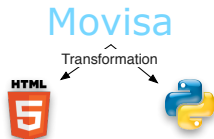


01 Einführung

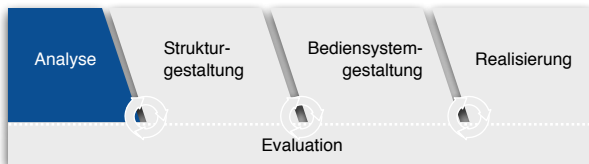
Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



Ueware-Entwicklungsprozess



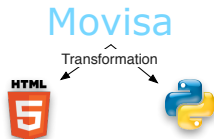
- Datenerhebung
- Datenaufbereitung
- (UseML)

01 Einführung

Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



Ueware-Entwicklungsprozess



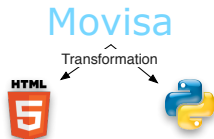
- Datenerhebung
- Datenaufbereitung
- (UseML)
- Aufgabenstruktur
- Benutzerinteraktionen
- (DISL)

01 Einführung

Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



Ueware-Entwicklungsprozess



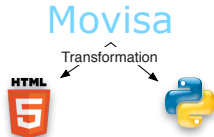
- | | | |
|---------------------|-------------------------|--------------------|
| • Datenerhebung | • Aufgabenstruktur | • Layout |
| • Datenaufbereitung | • Benutzerinteraktionen | • Modalitätenwahl |
| • (UseML) | • (DISL) | • (UIML, VoiceXML) |

01 Einführung

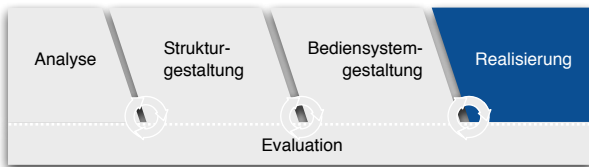
Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



Ueware-Entwicklungsprozess



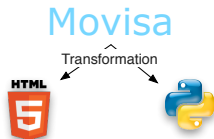
- Datenerhebung
- Datenaufbereitung
- (UseML)
- Aufgabenstruktur
- Benutzerinteraktionen
- (DISL)
- Layout
- Modalitätenwahl
- (UIML, VoiceXML)
- Feingestaltung
- (XAML)

01 Einführung

Integration von Movisa in den Ueware-Entwicklungsprozess

Movisa

- Domänenspezifische Sprache für die AT
- Plattformunabhängige Beschreibung



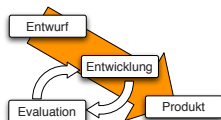
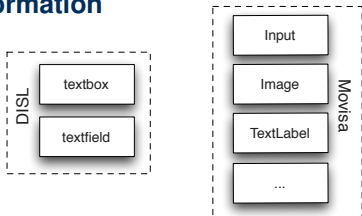
Ueware-Entwicklungsprozess



- | | | | |
|---------------------|-------------------------|----------------|------------------|
| • Datenerhebung | • Aufgabenstruktur | • Layout | • Feingestaltung |
| • Datenaufbereitung | • Benutzerinteraktionen | • AT Spezifika | • (HTML5,Python) |
| • (UseML) | • (DISL) | • (Movisa) | |

02 Problemstellung

Herausforderungen bei der Transformation

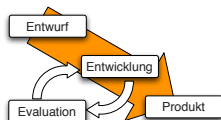
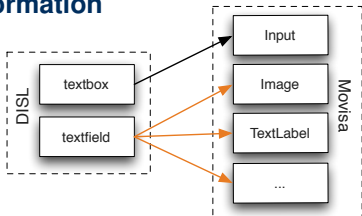


02 Problemstellung

Herausforderungen bei der Transformation

❶ Mehrdeutige Elementzuordnungen

- 6 DSL Interaktionsobjekte
- 12 Movisa Interaktionsobjekte

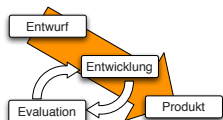
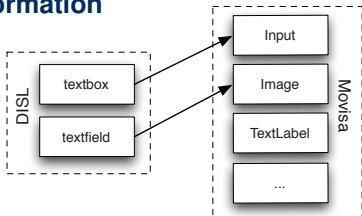


02 Problemstellung

Herausforderungen bei der Transformation

❶ Mehrdeutige Elementzuordnungen

- 6 DSL Interaktionsobjekte
- 12 Movisa Interaktionsobjekte
- ✗ Standardzuordnungen

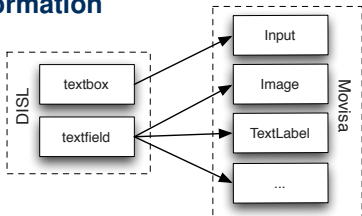


02 Problemstellung

Herausforderungen bei der Transformation

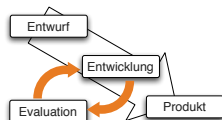
❶ **Mehrdeutige Elementzuordnungen**

- 6 DSL Interaktionsobjekte
- 12 Movisa Interaktionsobjekte
- ✗ Standardzuordnungen



❷ **Iterative Entwicklung** der Userware

- Veränderung/Anpassung aller Modelle zu jeder Zeit
- Konsistenz der Modelle bzw. Transformation beachten

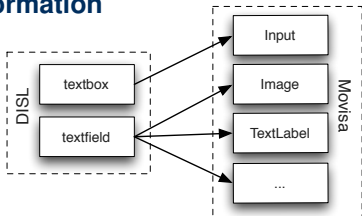


02 Problemstellung

Herausforderungen bei der Transformation

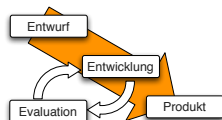
❶ Mehrdeutige Elementzuordnungen

- 6 DSL Interaktionsobjekte
- 12 Movisa Interaktionsobjekte
- ✗ Standardzuordnungen
- **Interaktive Transformation**



❷ Iterative Entwicklung der Useware

- Veränderung/Anpassung aller Modelle zu jeder Zeit
- Konsistenz der Modelle bzw. Transformation beachten



02 Problemstellung

Herausforderungen bei der Transformation

❶ Mehrdeutige Elementzuordnungen

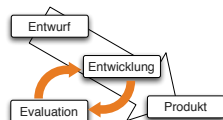
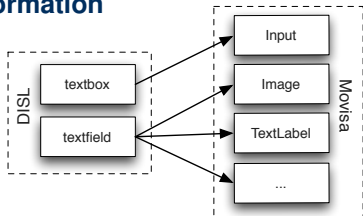
- 6 DSL Interaktionsobjekte
- 12 Movisa Interaktionsobjekte
- ✗ Standardzuordnungen

→ Interaktive Transformation

❷ Iterative Entwicklung der Userware

- Veränderung/Anpassung aller Modelle zu jeder Zeit
- Konsistenz der Modelle bzw. Transformation beachten

→ Persistant Transformation Mapping



03 Lösungsvorschlag

Interaktive Transformation mit Petmap

Interaktive Transformation

- Der Entwickler wählt das konkrete Element aus (Instanz-Ebene)
- Vorauswahl der Optionen

DISL



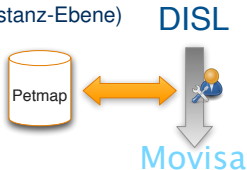
Movisa

03 Lösungsvorschlag

Interaktive Transformation mit Petmap

Interaktive Transformation

- Der Entwickler wählt das konkrete Element aus (Instanz-Ebene)
- Vorauswahl der Optionen



Persistent Transformation Mapping (Petmap)

- Speicherung der ausgewählten Zuordnungen (Trace-Modell)
- Wiederverwendung bei erneuter Transformation (Mapping-Modell)



03 Lösungsvorschlag

Anwendung der Transformation

DISL Modell

```
<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->
```

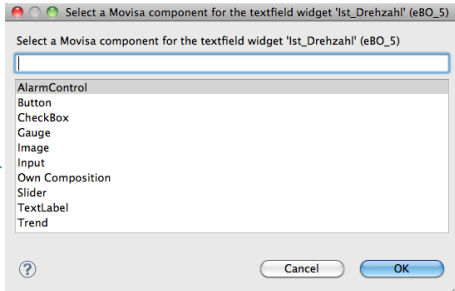
03 Lösungsvorschlag

Anwendung der Transformation

DISL Modell

```
<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->
```

- Drei Interaktionen notwendig



03 Lösungsvorschlag

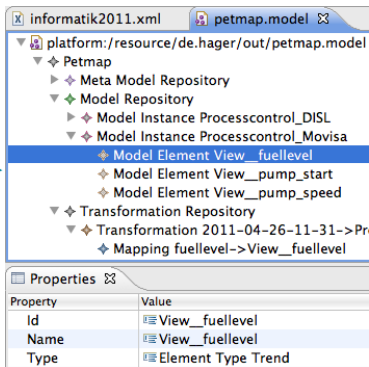
Anwendung der Transformation

DISL Modell

```

<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->
  
```

- Drei Interaktionen notwendig



The screenshot shows a software development environment with two tabs: 'informatik2011.xml' and 'petmap.model'. The 'petmap.model' tab is active, displaying a tree view of the model structure. The tree is expanded to show the 'Model Element View_fuellevel' node, which is highlighted in blue. Below the tree view is a 'Properties' panel with a table of properties for the selected element.

Property	Value
Id	View_fuellevel
Name	View_fuellevel
Type	Element Type Trend

03 Lösungsvorschlag

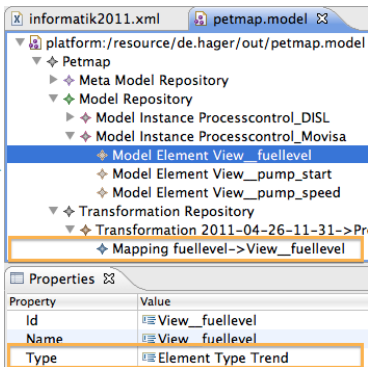
Anwendung der Transformation

DISL Modell

```

<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->
  
```

- Drei Interaktionen notwendig



The screenshot shows a software development environment with two tabs: 'informatik2011.xml' and 'petmap.model'. The main area displays a tree view of a model structure:

- platform:/resource/de.hager/out/petmap.model
 - Petmap
 - Meta Model Repository
 - Model Repository
 - Model Instance Processcontrol_DISL
 - Model Instance Processcontrol_Movisa
 - Model Element View_fuellevel (highlighted)
 - Model Element View_pump_start
 - Model Element View_pump_speed
 - Transformation Repository
 - Transformation 2011-04-26-11-31->Pr
 - Mapping fuellevel->View_fuellevel (highlighted)

Below the tree view is a 'Properties' panel with the following table:

Property	Value
Id	View_fuellevel
Name	View_fuellevel
Type	Element Type Trend

03 Lösungsvorschlag

Anwendung der Transformation

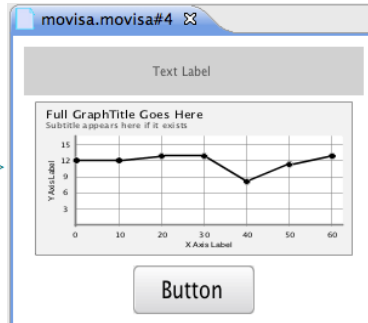
DISL Modell

```

<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->

```

- Drei Interaktionen notwendig
- Beschriftungen, Interaktionseigenschaften generiert



03 Lösungsvorschlag

Anwendung der Transformation

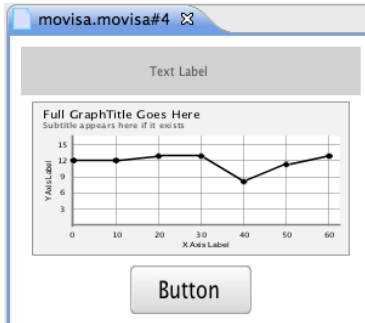
DISL Modell

```

<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="command" id="pump_stop"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->

```

- Drei Interaktionen notwendig
- Beschriftungen, Interaktionseigenschaften generiert
- Anordnung der Elemente erfolgt manuell



03 Lösungsvorschlag

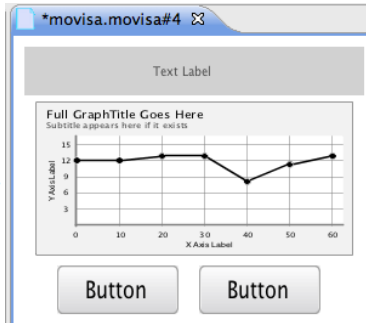
Anwendung der Transformation

DISL Modell

```

<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="command" id="pump_stop"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->
  
```

- Drei Interaktionen notwendig
- Beschriftungen, Interaktionseigenschaften generiert
- Anordnung der Elemente erfolgt manuell
- Erneute Transformation: Benutzerinteraktion nur für neue Elemente



03 Lösungsvorschlag

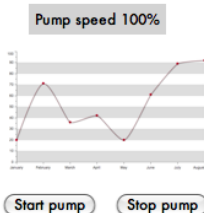
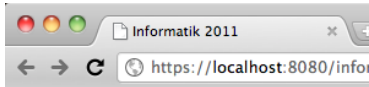
Anwendung der Transformation

DISL Modell

```

<!-- -->
<interface id="View" state="start">
  <structure>
    <widget generic-widget="textfield" id="fuellevel"/>
    <widget generic-widget="command" id="pump_start"/>
    <widget generic-widget="command" id="pump_stop"/>
    <widget generic-widget="textfield" id="pump_speed"/>
  </structure>
  <style> <!-- ... --> </style>
  <behavior> <!-- ... --> </behavior>
</interface>
<!-- -->

```



- Drei Interaktionen notwendig
- Beschriftungen, Interaktionseigenschaften generiert
- Anordnung der Elemente erfolgt manuell
- Erneute Transformation: Benutzerinteraktion nur für neue Elemente



04 Auswertung

Integration der DSL Movisa in den Ueware-Entwicklungsprozess

Ergebnisse

- Transformation mit Entwicklerinteraktion
- Kombination von Trace- und Mapping-Modell (Petmap)
- Wiederverwendung der Mappings

04 Auswertung

Integration der DSL Movisa in den Ueware-Entwicklungsprozess

Ergebnisse

- Transformation mit Entwicklerinteraktion
- Kombination von Trace- und Mapping-Modell (Petmap)
- Wiederverwendung der Mappings

Forschungspotential

- Roundtrip-Engineering: Veränderungen im Movisa-Modell gehen verloren
- Synchronisierungsmechanismen