Mobile Software Engineering

Volker Gruhn
University of Leipzig, Computer Science
Chair for Applied Telematics / e-Business
Endowed by Deutschen Telekom AG
Overview

• Mobile Software Engineering

• Mobile Process Landscaping

• Mobile Architectures: Modelling and Simulation

• Mobile Dialogue Specification

• State-of-the-Art Research: MobCo
Mobile Software Engineering
Motivation

• Enterprises are mobilizing their information systems (at least partially)

• Mobile projects in the past often failed due to
  • wrong distribution of code and data
  • service levels needed
  • telecommunication costs

• Current trend to service orientation emphasizes
  • explicit separation between interfaces and implementation and
  • loose coupling,
  allowing system architecture to change during runtime

• Mobile code and consequent application of loose coupling and runtime
  re-configuration of architecture can be regarded as similar

→ Application of methods for modeling and evaluating mobile architectures
Mobile Information Systems

- **Corporate externally**
  - **Mobile B2C**
    - Use of mobile technology for sales and service channels

- **Corporate internally**
  - **Mobile B2E**
    - Use of mobile technology for connecting mobile workers to the company

  - **Mobile M2M**
    - Use of mobile technology for communication between machines
Motivation

• “… mobility represents a total meltdown of all the stability assumptions … associated with distributed computing”
  (Roman, Picco, Murphy: Software Engineering for Mobility: A Roadmap, 2000)
  • Network structures which are no longer fixed
  • Communication failures
  • Restricted connectivity, low bandwidth

• Functional and non-functional properties influence system design
  • Functional properties express functionality of system
  • Non-functional properties express how functional aspects have to be covered

• Mobile Architecture Description: Functional and non-functional properties play important role
Goal: realistic decision at design time whether mobile system is fit for purpose

- Combination of software aspects with telecommunications yields complex emergent behavior

- Support needed for all kinds of mobility
  - Physical mobility (user / device mobility)
  - Logical mobility (mobile code, stateless or stateful)

- Challenges with regard to related work
  - Clear distinction between logical and physical components and connectors to allow realistic modeling of mobility influence
  - Inclusion of communication network model
  - Inclusion of user interactions into system evaluation
Mobile System Assessment

Deadlocks, Security Aspects, Hard Real-time Constraints etc.  
Mean Values, Probabilities

Formal Methods (Model Checking etc.)  
Stochastic Methods (Markov Chains etc.)

Mobile System Specification
Static: Components, Connectors  
Dynamic: Behavior

Simulation

Emergent Behavior / Properties: Response Times, Bottlenecks etc.

“behavior [ / properties] of complex adaptive system … which cannot be deduced from the components of the system.” [www.agtivity.com]
Mobile Software Engineering

- Mobile Computing
  - Mobility requirements determine business process models, software architectures and middleware functionality needed
  - Programming languages should allow to abstract from mobility issues (e.g. mobile code languages like Obliq and mobile agent languages like Agent Tcl)
  - Middleware in the focus of Mobile Computing
Mobile Software Engineering

- New requirements raised by logical mobility
  - Data replication
  - Explicit validity of replicated data
  - Synchronisation of data
  - Management of lost updates

- New requirements raised by physical mobility
  - Mapping of logical and physical mobility
  - Units of physical mobility
  - Space and location as explicit entities
  - Space-aware context management
First Conclusion

Software engineering for mobile systems is basically software engineering.

And it has some additional challenges:
- Performance
- Re-engineering of business processes
- Flexible relocation of code and data

Dealing with these challenges:
- Modelling of mobile business processes
- Designing mobile software architectures
- Infrastructure for mobile dialogues
- Infrastructure for mobile runtime support
Mobile Process Landscaping
Motivating Example: Mobile Processes

Customer

Order

Invoice, Gathered data

Craftsman

Database, Document Management, SAP

Customer

Craftsman

Invoice, Gathered data

Order
Motivating Example: Mobile Processes

Database, Document Management, SAP

Disposition, Tour schedule

Synchronisation

customer
Typical application landscape

- CRM
- DB
- ERP
- HR
- Groupware
- SCM

Legacy Systems

Internet

Interface

Mobile Interface

Mobile Interface

Purchase

Service

Supplier

Customers

Mobile Sales
Typical mobile processes

Relevant vertical markets

- Logistics
- Energy
- Insurance
- Health Care

Benefits of mobile solutions increase, if:

- Large number of mobile employees has to be supported
- Exchange of structured data between decentral and central location
- Data to be exchanged is location-based or time-related
Goals of introducing mobile solutions

Office in Your Pocket = Anywhere access to business critical applications

Source: IDC 2005
Mobile Business Processes

• All sales-oriented processes tend to become more and more mobile.

• All processes spreading over various sales channels tend to become mobile.

• All processes bringing services to customer locations tend to encompass mobile parts.

• Mobility of processes is still hindered by some main obstacles:
  • Telecommunication infrastructure (of decreasing importance)
  • Inflexible legacy systems, which are difficult to open
  • Organizational issues
  • Variety of mobile devices – no mainstream
Mobile Process Landscaping

Mobile Process Landscaping is a method for modelling and analysing mobile business processes.

The basic idea is to start from business processes to be supported.

<table>
<thead>
<tr>
<th>Process</th>
<th>MobileProcessbase</th>
<th>MobileAssessment</th>
<th>MobileROI</th>
<th>MobileAnalyser</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalogue of business processes</td>
<td>Check of mobile potential</td>
<td>Business case analysis</td>
<td>Simulation of mobile business process model properties (compare to Con Moto)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application</th>
<th>MobileArchitect</th>
<th>Architectural patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Architectural patterns</td>
<td></td>
</tr>
</tbody>
</table>
Catalogue of mobile business process models and their basic building blocks.

**Electricity, water, gas supply**
Installation, maintenance, repair of infrastructure

**Real estate management**
Repair, inspection, maintenance

**Insurance**
Sales (employed), sales (broker), claims management (composite)
## Mobile Process of Energy Providers

<table>
<thead>
<tr>
<th>mobile business processes</th>
<th>structuredness</th>
<th>recurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>electricity supply application fulfillment for L/V customers</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>electricity supply application fulfillment for M/V customers</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>inspection of simple installations</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>inspection of blocks of flats</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>meter disconnection</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>meter re-connection</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>substation maintenance</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>preventive maintenance on network elements</td>
<td>strong</td>
<td>high</td>
</tr>
<tr>
<td>installation modification</td>
<td>medium</td>
<td>high</td>
</tr>
<tr>
<td>failure restoration</td>
<td>medium</td>
<td>medium</td>
</tr>
<tr>
<td>installation dismantlement</td>
<td>medium</td>
<td>medium</td>
</tr>
<tr>
<td>repair of hazardous situations</td>
<td>less</td>
<td>low</td>
</tr>
<tr>
<td>repair of damages to network due to natural disaster</td>
<td>less</td>
<td>low</td>
</tr>
<tr>
<td>pruning trees that interfere with network</td>
<td>less</td>
<td>low</td>
</tr>
</tbody>
</table>

*Mobile Processbase (Example)*

Volker Gruhn • University of Leipzig
MobileProcessbase (Example)

Industry
Electricity Network Provider

Process
Electricity Supply Application
Fulfillment for L/V Customers

Structuredness
Strong

Recurrences
High
Mobile Assessment

Mobile processes are decomposed into activities. These are checked with respect to their mobility potential.

**Indicators for general improvement**
- value contribution to the overall business process
- number of executions
- Importance of customer satisfaction
- non digital data transfers

**Indicators for potential improvements (focus mobility)**
- involved persons meet at a certain location
- involved persons are – usually - at different sites
- activity is carried out in-motion
- data to be exchanged is structured
## MobileAssessment (Example)

### Assessment Example for Facility Service Processes

<table>
<thead>
<tr>
<th>weight</th>
<th>1,0</th>
<th>1,5</th>
<th>0,3</th>
<th>1,8</th>
<th>Overall Weighted Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>process data records</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>plan tour</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>inform service technician</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>provide information sheet</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>print list</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>drive to office</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>brief service technician</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>drive to property</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>meter reading</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>second recording</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>drive to office</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>drive to property</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>deliver list</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>enter data into software</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>error handling</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

### Graphical Representation

- General Potential
- Mobile Potential

Volker Gruhn • University of Leipzig
Typical problems of mobile information systems:

1. **Telecommunication costs**
   - Identify data-intensive activities and their detailed costs
   - Reduce data volume (e.g. data replication, move of activities)

2. **Performance of application due to missing bandwidth**
   - Identify activities which may cause user waiting times
   - Prefetching of data
   - Reduce data volume (e.g. data replication, move of activities)

3. **Unclear net behavior at POA**
   - Identify dialogue specific requirements for bandwidth, transaction behavior, response time
   - Identify SLAs and requirements for seamless roaming
MPL Experience
Mobile Architectures: Modelling and Simulation
Types of Architectures

Four main types of architectures for mobile systems:

- web-based always-online architecture
- rich client always-online architecture
- rich client hybrid architecture
- fat client offline architecture
## Types of Architectures

<table>
<thead>
<tr>
<th>Layer</th>
<th>Type</th>
<th>offline</th>
<th>hyb-off</th>
<th>hyb-on</th>
<th>online</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User Interface</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Client</td>
<td></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Flashlight" /></td>
</tr>
<tr>
<td>• Server</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><img src="image" alt="Sun" /></td>
</tr>
<tr>
<td><strong>Business Logic</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Client</td>
<td></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Flashlight" /></td>
<td><img src="image" alt="Sun" /></td>
</tr>
<tr>
<td>• Server</td>
<td></td>
<td></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Sun" /></td>
<td><img src="image" alt="Sun" /></td>
</tr>
<tr>
<td><strong>Data</strong></td>
<td></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
</tr>
<tr>
<td>• Client</td>
<td></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
</tr>
<tr>
<td>• Server</td>
<td></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
<td><img src="image" alt="Database" /></td>
</tr>
</tbody>
</table>

**Legend:** Execution ![Sun](image)  Transport ![Flashlight](image)  Persistence ![Database](image)
Con Moto Approach

• Con Moto for specification and evaluation of mobile architectures
  • Mobile ADL for modeling structure and behavior of the mobile system
  • Model of the used communication networks
  • Model of usage- / interaction patterns
  • Instantiation model for initial configuration

• Simulator for model evaluation allows predicting compliance with non-functional requirements like transaction times, scalability etc.
Behavioral Model

- Mobile system as a set of communicating concurrent processes
- Communication as exchange of objects
- Based on $\pi$ calculus
  - Complete $\pi$ calculus not necessary cf. Pict [Pierce, Turner 2000]
  - Restriction still allows necessary expressive power, avoids non-determinism at the same time
- Con Moto $\pi$ calculus
  - Output performed synchronous (rendezvous) on a pin ($\pi$ calculus: name)
  - Different to Pict (there: asynchronous), but like in original $\pi$ calculus
    - Decoupling of senders and receivers realized by connector behavior
    - Replication restricted to input operations
    $\Rightarrow$ new process instances (threads) can be created using input operations or parallel operator
Structural Model

• Processes as model for mobile system are over-simplification

Structural model

• Managing complexity by abstraction (components, connectors etc.)
• Determines essential mobility aspects
  • Which entities are mobile at all?
  • What’s the smallest mobile entity?

• Common knowledge about ADLs
  • Components act as locus computandi
  • Connectors model communication relations among components
  • Configurations grasp changes in architecture over time

Volker Gruhn • University of Leipzig
Structural Model

• Connectors
  • Physical: Communication links between devices
    • Are not ideal (limited bandwidth, latency time > 0)
  • Logical: Communication links between software components
    • Are ideal (unlimited bandwidth, zero latency)
  • Logical connectors can be embedded in physical connectors (e.g. if multiple software components are deployed on one single device)

• Assumptions
  • Communication is handled in packages
  • Connectors are not software from the modeled systems perspective (hence, they cannot be specialized components like in other ADLs)
Various models of communication should be supported
  • Synchronous calls (RPC, service invocation)
  • Asynchronous communication (broadcast, events)
    → Possible in dynamic + structural model

Mapping “Problem”: Granularity
  • Dynamic model: processes communicate via pins
  • Structural model: Processes in components, which communicate via ports (pins) and connectors
    • In contrast: OO-model – method calls, provides and uses interfaces etc.

→ Abstraction mechanism allows structural similarities e.g. with OO-systems to ease modeling
Configurations

• Dynamic changes of the architecture possible by communication of components and component instances

• Restrictions can be used to guard evolving configurations at runtime

• Graphical representation: extended deployment diagram:
  • Dependencies (uses-relationships) between components
  • Deployment possibilities of the components (constraints)
  • Delivers first sketch of communication relations (however, no full network model)
Simulator, Network Model

- Prototypical realization of simulator as proof-of-concept

- Derivation of local and global properties
  - Elapsed Time between probe points
  - Data volumes at probe points

- Used ad-hoc model for simulation
  - Nodes with node type and count
  - Automatic connection of nodes at creation time
  - Size-dependent delay of data packets in the nodes
  - Lookup-possibilities (node type)
Interaction Model, Instantiation Model

- Models how user interact with devices and software deployed on these

- Activities like
  - Switching devices on and off
  - Changes of location
  - Invoking services offered by components deployed on devices

- Challenge: description of such activities and mixing with probabilistic processes (similar to random traffic in telecommunication systems)

- Instantiation model reflects initial setup of the system
Example System

- **Hardware components (physical)**
  - SERVER as host
  - MOBILE as mobile devices

- **Software components (logical)**
  - UI always on MOBILE
  - BUSINESS either on MOBILE or on SERVER
  - DATA on SERVER

- **Invocation relationships**
  - Service in UI calls Service from BUSINESS
  - Service in Business invokes Service from DATA

![Diagram of Example System](Image)
<system>
  <connection>
    <port-role name="in" />
    <port-role name="out" />
  </connection>
  <port-role name="methodsInvoker" extends-role="out">
    <ports name="methodInvoker" />
  </port-role>
  
  <network-config>
    <passive-node name="backbone" />
    
    <active-node name="UMTS">
      <multiplicity>0.5</multiplicity>
      <auto-link>
        <node>backbone</node>
        <bandwidth>10.0</bandwidth>
      </auto-link>
    </active-node>
    
    <component name="BUSINESS">
      <size>200</size>
      
      <start-process>
        <pi>
          PhysComp remoteHW = lookupPhysComp("SERVER");
          LogComp remoteSW = remoteHW.lookupLogComp("DATA");
          connect(this.getData, remoteSW.getData);
        </pi>
      </start-process>
      
      <port type="methodInvoker" name="getData" />
      <port type="methodProvider" name="getInfo" >
        <extend-process>
          object res;
          object par;
          res = getData(par);
          response.size = 5000;
          useCpu(500);
        </pi>
      </extend-process>
    </component>
    
    ...
  </network-config>
</system>
Simulation Results

- Mobility of BUSINESS – Variants
  - Not mobile (always on SERVER)
  - Mobile (on UI), deployment for all channels
  - Mobile (on UI), deployment only for UMTS
Mobile Dialogue Specification
Mobile Business Processes

- Software applications have to be modified to fit different mobile devices. With respect to:
  - Dialog size
  - Dialog interface
  - Transactional logic
Motivation: Usability vs. Technology

- “suitability for the task”, “conformity with user expectations”
  (ISO 9241-10 dialog principles)
  - user interface should be guided by how users handle a problem, but not be determined by characteristics of underlying technology

- in web apps, most user interaction occurs as navigation
  - design of navigation structures heavily influences usability

- web infrastructure and devices impose technical limitations
  - implementation of complex navigation structures requires high effort

- particular challenges
  - hierarchical dialog structures (multiple windows in traditional apps)
  - device-specific dialog sequences (more steps on smaller screens)
Efficient Dialog Modeling and Control

- dialog flow should be driving aspect of development process
- needs to be modeled intuitively and processed efficiently
  - sufficiently expressive to describe complex nested dialogs
  - sufficiently flexible to be adapted to different devices
  - suitable for automatic transformation into executable format
  - keeping specification redundancies and overhead to a minimum

Solution: Dialog Flow Notation (DFN)

- graphical notation for nestable dialog modules on various devices
- can be transformed into machine-readable XML format as input for Dialog Control Framework (DCF) that handles dialog flow at run-time
- need formal semantics of notation elements to ensure that the framework will control the dialog flow at run-time as intended by the application developer at design-time
Related Work

- RMM development process, Fraternali, 1999
- OOHDM, Schwabe, Rosse, 1995
- HDM-lite, Paolini, 2000
- DoDL, Doberkat, 1996
- WebML, Ceri, 2000

Focused on data-intensive hypertext solutions, little support of dynamic features.
Multiple Presentation Channels

I/O Capabilities

Mobility
Enterprise Applications on Thin Clients

- devices may have strict energy/memory/interface limitations
  - individual implementation for every kind of device too costly

- thin client architecture most economic solution
  - typically implemented as web-based application

- however, different devices have different UI capabilities
  - web applications must cope with different usage patterns
Device-Specific Interaction Patterns

Example: Online Shop Checkout Process

checkout [desktop]

A,S,B Mask

address eval

shipping eval

billing eval

checkout [mobile]

Address Mask

Address eval

Shipping Mask

Shipping eval

Billing Mask

Billing eval

Example:

Online Shop Checkout Process

- **checkout [desktop]**
  - A,S,B Mask
  - Address eval
  - Shipping eval
  - Billing eval

- **checkout [mobile]**
  - Address Mask
  - Address eval
  - Shipping Mask
  - Shipping eval
  - Billing Mask
  - Billing eval
Architectural Responses

- separate presentation and application logic
  - as suggested by MVC pattern, implemented in window-based apps
  - additional logic required to govern sequence of pages/computations
  - dialog control logic often mixed with application logic, rendering it device-dependent
  - employ distinct dialog controller to decouple app/presentation logic
- link device-specific UI to device-independent app logic
  - presentation logic must be device-specific to cater to individual device capabilities
  - application logic should be device-independent to allow re-use across all presentation channels
  - link both layers flexibly through dialog flow specification interpreted at run-time, instead of hard-wired implementation
- need a means for specifying similarities and peculiarities of different devices‘ dialog flows in an efficient way
Generic vs. Specific Aspects of Dialogs

• W3C device independence principles, ISO dialog principles call for dialog flows that are as generic as possible:
  • provide the same functionality across all devices
  • structure features similarly on all devices to help users build a consistent conceptual model
  • employ similar navigation/interaction patterns on all devices

• However, some challenges call for specific solutions:
  • dialogs may require different pagination depending on screen size
  • selected features may be reduced/missing on some devices due to interface limitations or business process considerations
  • some features may be exclusively available on certain devices (e.g. location-based services on certain mobile phones)

➤ generic parts should only have to be modeled once, while specific parts should be modeled explicitly for certain device
Generic and Specific Module Definition

- using *dialog modules* of Dialog Flow Notation (DFN):
  - generic dialog module definition is valid for all channels (unless overridden by channel-specific definition)
  - specific dialog module definition inherits generic definition and extends/overrides it with channel-specific dialog flow fragments
  - reduced specification redundancy, re-use of application logic
Create Account Module Flow Spec

```xml
<dfs-flows>
  <in-module name="create account">
    <channel>
      <on-init>
        <call-mask>address, prefs, passwd form</call-mask>
      </on-init>
      ...
      <ex-action name="passwd plausib check">
        <on-event name="invalid">
          <call-mask>address, prefs, passwd form</call-mask>
        </on-event>
        <on-event name="valid">
          <call-action>create account</call-action>
        </on-event>
      </ex-action>
      <ex-mask name="create account">
        <on-event name="ok">
          <term-event>done</term-event>
        </on-event>
      </ex-mask>
    </channel>
    <channel name="mobile">
      <on-init>
        <call-mask>address form</call-mask>
      </on-init>
      ...
      <ex-action name="passwd plausib check">
        <on-event name="invalid">
          <call-mask>passwd form</call-mask>
        </on-event>
      </ex-action>
    </channel>
  </in-module>
</dfs-flows>
```
Dialog Module Definition and Use

- choice of required module specification/mask implementation at run-time
Dialog Control Framework Architecture

Dialog Control Framework

Module Stack

Push, pop, top

OO Dialog Flow Model

Request

Channel Servlet

Event

Dialog Controller

Browser (Client)

Response

Action

Update

Extract

Model

XML Dialog Flow Specification

Import

Web Application

Event

Event

push, pop, top

look up
From Specification to Implementation

create account [desktop]

create account [mobile]

...&lt;in-module name="create account">
 &lt;channel name="mobile">
   &lt;on-init>
     &lt;call-mask>Address Mask
   &lt;/on-init>
   &lt;ex-mask name="Address Mask">
     &lt;on-event name="next">
       &lt;call-action>address eval</call-action>
     &lt;/on-event>
   &lt;/ex-mask>
   &lt;ex-action name="address eval">
     &lt;on-event name="ok">
       &lt;call-mask>Prefs Mask</call-mask>
     &lt;/on-event>
   &lt;/ex-action>
...
ARGuS Project Experience

Specification and Design Phase

• application and user interface design
• definition of dialog flows for desktop and mobile channel
• 2 months

Implementation Phase

• desktop channel
  • implementation of app logic, 29 actions, 24 rich HTML masks
  • 2 months

• mobile phone channel
  • implementation of 22 WML masks
  • 2 days

• PDA channel (added late in project)
  • dialog flow spec extension, implementation of 24 light HTML masks
  • 1 day
Sample: Desktop, PDA, Mobile Channel

ARGuS
Anywhere Reachable Guiding System

Sights

Name: Kirche
Street: 
City District: 

Nikolaikirche | Zentrum, Zentrum-West | Nikolaikirchhof 3 | more...
Projekt Paulinuskirche | Zentrum, Zentrum-West | Augustusplatz | more...
Russische Gedächtniskirche | n/a | Philipp-Rosenthal-Straße | more...
Thomaskirche zu Leipzig | Zentrum, Zentrum-West | Thomaskirchhof 18 | more...

Home | City Map | Profile | Locations | Travelplan | Timetables | Sights | Restaurants | Hotels | Events | Timetables

Saved

Desktop channel

Mobile phone channel

PDA channel
Mobile Code Middleware: MobCo
Market
Market of mobile applications shows big future potential

Challenge
Development of mobile applications is a complex and conflictive task

Our Solution
MobCo brings Agility and Specialty together to build the mobile applications of the future
MobCo facts

Consultancy and Development

- Unique consulting approach
- Tool for knowledge externalization and knowledge repository
- Common methodology for mobile application development
- Reusability of software components

Operations

- Optimization during runtime to customers usage behavior
- Active SLA management
- Adoption of solutions to changing requirements, when
  - technical environment changes
  - underlying business processes change

MobCo Toolset and Middleware

Key deliverables

- Evaluated architecture blueprints for mobile architectures
- Simulator for mobile architectures
- Middleware for applications with self-mobilizing code
- Demonstrator application for one vertical market

Key facts

- Duration: 24 months
- Begin: 1/2007
- Involved SBUs: T-Systems, T-Mobile
- Contractors / partners: University of Leipzig, T-Systems
Mobile Software Engineering is essentially Software Engineering!

The software process deserves particular support for

- mobile business processes
- mobile architectures
- mobile interfaces
- mobility supporting infrastructure

In practice: development of mobile solutions is still error-prone, risky and expensive.
Volker Gruhn
University of Leipzig
Chair of Applied Telematics / e-Business
Klostergasse 3
04109 Leipzig, Germany

koehler@ebus.informatik.uni-leipzig.de

www.lpz-ebusiness.de