Using Interpreted Runtime Models for Devising Adaptive User Interfaces of Enterprise Applications

Pierre A. Akiki, Arosha K. Bandara, and Yijun Yu

14th International Conference on Enterprise Information Systems - ICEIS 2012
Outline

Introduction
- Enterprise Application Usability
- MDE and Adaptive UIs

Adaptive Scenarios
- Simplifying Individual UIs
- Composing New Functionality

Methodology
- Architecture (CEDAR)
- Support Tool

Case Study
- Using Role Based Access Control
- Performance Analysis

Conclusion
Survey Conducted in the Nordic Countries (Lykkegaard & Elbak 2011)

40% of the Participants Found Enterprise Applications Somewhat, Very, or Extremely Hard to Use

Maximize Usability
MDE Approach to UIs

**Static Models**
- NOT Available at Runtime
- Depend on Code Artefacts

**Generative Runtime Models**
- Available at Runtime
- Depend on Code Artefacts

**Interpreted Runtime Models**
- Available at Runtime
- Do NOT Depend on Code Artefacts
Adaptive UI Scenarios

**Scenario 1: Simplifying Individual User Interfaces**
- **UI Widget Substitution**
- **UI Widget Elimination**
- **UI Widget Realignment**
  - Eliminate Unused Features
  - Adapt to Computer Literacy
    - Novice, Intermediate, Expert

**Scenario 2: Composing New Functionality from Existing User Interfaces**
- **Selecting UI Subsets**
- **Merging UIs at Runtime**
- **End User Behaviour**
- **Scattered User Interfaces**
  - Example: Microsoft Dynamics GP Item Maintenance
Proposed Architecture

Adaptive & UI Models

Adaptation Components

Decision Components

Client Components

Three Layer Architecture Model for Self-Management (Kramer and Magee) | CAMELEON Reference Framework (Calvary et al.) | MVC | Three Tiers
---|---|---|---
Domain Model | Task Model | Abstract User Interface (AUI) | Concrete User Interface (CUI)

Adaptive & UI Models Database

Adaptive Model(s)

Cognition | Culture | Resolution | Mobility

Goal Management

Change Management

Component Control

Final User Interface (FUI)

Client Components

C1 | C2

View | Client

Report Context Status | Caching Engine | Context Monitor | UI Renderer

Controller | Application Server

Request Adapted UI | Caching Engine | Context Evaluator | Modify UI / Propose new UI

Adapt UI

S1 | S2 | S3 | S4 | S5

Store Proposed Adapted UI | Adaptive Engine(s) | Trade Off Manager | UIDL Converter

S1

ICEIS 2012
Using Interpreted Runtime Models for Devising Adaptive User Interfaces of Enterprise Applications
Akiki, Bandara, & Yu
Tool Support

Ribbon

Designer Canvas

Toolbox

Properties & Events
Case Study

Role Based Access Control (RBAC)

Open Dental – Claims Form
- 87 Widgets of 9 Different Types
- Reverse Engineered to Relational Data

Users (Enterprise Employees)

Roles (Manager, Cashier, etc.)

Resources (User Interfaces)

User Interface Performance

Time in Milliseconds

UI Loaded & Closed (1000 Times)

- Compiled Code Based UI
- Interpreted Runtime Model Based UI
Conclusion

- Interpreted runtime models provide more flexibility for advanced adaptive scenarios.
- An adaptive UI architecture with middleware and tool support could provide a generic solution for multiple types of adaptive factors.
- Applying appropriate caching techniques prevents any degradation in performance.
Thank You!

International Conference on Enterprise Information Systems - ICEIS 2012

http://adaptiveui.pierreakiki.com