Challenges in Using a Mobile Component Framework to Develop Adaptive Groupware Applications

Radu Litiu and Atul Prakash
University of Michigan, EECS
Outline

■ Problems and Design Goals
■ System Architecture
■ Building Adaptive Groupware Applications
■ Lessons Learned and Open Issues
■ Summary
Problems

- Variability and heterogeneity
- User and application demands
- Hardware and network variability
- Flexibility and adaptability
- Application and user mobility
- Intermittent connectivity
- Support for offline users
DACIA* Features

- Component-based framework
- Dynamic application reconfiguration
- Component mobility
- Persistent connectivity between components
- Application parking

*Dynamic Adjustment of Component InterActions
DACIA Architecture

**Engine** (mechanism)
- Communicate between hosts
- Manage connections between components
- Relocate components
- Reconfigure the application

**Monitor** (policy)
- Monitor performance
- Make reconfiguration decisions
- Implement application-specific reconfiguration policies

**PROCs** (Processing and Routing Component)
- Communication through ports
- Component mobility – state transfer
Component Mobility (I)

Connection closed: Socket[addr=saturn.eecs.umich.edu/141.213.10.11, port=6301]
Can't re-establish connection to host saturn

Connections:
Local PROC: Engine > print
Remote PROC:
Engine > connect saturn 6301
Connection established: Socket[addr=saturn/141.213.10.100, port=6301]

Connections:
Local PROC:
Engine > connect saturn 6301
Remote PROC:
Engine > print

Connections:
Local PROC:
Engine > connect saturn 6301
Remote PROC:
Engine > print

Connections:
Local PROC:
Engine > connect saturn 6301
Remote PROC:
Engine > print

Connection established: Socket[addr=saturn/141.213.10.100, port=6301]

Connections:
Local PROC:
Remote PROC:

Remote PROC:
Engine > connect saturn 6301
Connection established: Socket[addr=saturn/141.213.10.100, port=6301]

Connections:
Local PROC:
Remote PROC:

Remote PROC:
Engine > connect saturn 6301
Connection established: Socket[addr=saturn/141.213.10.100, port=6301]

Connections:
Component Mobility (II)
Dynamic Application Reconfiguration

- Change connections between components
- Change components’ location
- Load new components

Mechanisms
- Specialized monitors
- Programming API
- Command-line interface
- Graphical interface

An adaptive application: multi-party communication
Structuring Groupware Applications

- Types of components
- Separate *User Interface* and *User Agent* code
- Horizontal/vertical decomposition
Division of Functionality and Coordination

- Separate mechanisms from adaptive policies
- Measure application performance
- Distributed coordination
  - Multiple administrative domains
  - Cooperation vs. competition
Support for Multiple Devices

- Context-aware applications
- Move components between different types of devices
- Hardware-dependent components
  - All-in-one
  - Multiple versions for one component
- State transfer
Security

- Host security
  - Malicious components
  - Programming errors

- Component security
  - Execution integrity
  - Data integrity
  - Secrecy
Summary

- Component-based framework for building adaptive groupware applications
- Dynamic reconfiguration
- Application and user mobility

Open issues:
- Application decomposition and coordination
- Support or multiple devices
- Security