Let it Flow: Building Mashups with Data Processing Pipelines

Biörn Biörnstad  
Department of Computer Science  
ETH Zurich, Switzerland  
bioernstad at inf.ethz.ch

Cesare Pautasso  
Faculty of Computer Science  
University of Lugano, Switzerland  
c.pautasso at ieee.org
Web Site Access Monitor Mashup


Geocoding
Limitations of Browser-based Integration

- Browser sandbox is not an integration platform
- Communication limited
  - HTTP only
    - Cannot access Web Log or Database
  - Same-origin JavaScript policy
    - Difficult to access remote hosts other than origin host
- How to program the mashup integration logic?
  - JavaScript
  - Data Flow Pipeline out of reusable components
  - Other languages and metaphors (e.g., spreadsheets)
Layered Mashup Architecture
Outline

- Web Site Access Monitor Mashup
- Layered Mashup Architecture
- Mashup Integration Challenges
- Developing the Mashup Data Flow Pipeline
- Demo
- The Mashup Engine and its Data Ports
- Conclusion
Mashup Integration Challenges

- Data Extraction
- Data Heterogeneity
- Data Integration
- Service Heterogeneity
- Data Quality
- Real-time update
- Maintainability
- Security
1. Integration Logic

Read Log → Log entries → Get Coords → Coords → Stream to Browser

Geocoding DB
2. Communication Mechanism

- Dealing with Service Heterogeneity

[Diagram showing the process flow from Read Log to Stream to Browser, with SSH, JDBC, and HTTP as interconnections.]
3. Data Mediation

- Dealing with data extraction & heterogeneity

- Read Log
  - Extract IP addr
  - Get Coords
  - Package JSON
  - Stream to Browser

- Log entry
  - IP address
  - Numbers
  - JSON

Glue Component
Download the Video at http://www.jopera.ethz.ch/docs/videos (Mashups’07 Demo Part 1)
Download the Video at http://www.jopera.ethz.ch/docs/videos (Mashups’07 Demo Part 2)
Download the Video at [http://www.jopera.ethz.ch/docs/videos](http://www.jopera.ethz.ch/docs/videos) (Mashups’07 Demo Part 3)
Mashup Engine

- Integration and Coordination of Services/Data sources
- Data is streamed through Mashup Data Flow Pipeline
- Engine runs multiple mashup pipelines at the same time
- Extensible with more Data Ports and Local Services (plug-ins)
Data Ports

- Interface to external systems and components
- Provide communication abstractions to deal with service heterogeneity
- Different Interaction Styles
  - Request-response ↔ ↔ HTTP, JDBC
  - One-way ← → E-mail, JMS
  - Streaming ← → RSS, SSH, HTTP

outbound    inbound
Layered Mashup Architecture

UI Integration Visualization
- Mashup UI (JavaScript)
- Browser
- Map widget

Data/Service Integration
- Mashup Integration Logic (Data Flow Pipeline)
- Mashup Engine
- Java
- RegEx
- String concat.

Service APIs
- Data sources
- Web server
- Remote WS
- Local DB
- DNS

Protocols
- HTTP
- SSH
- RSS
- JDBC
- DNS

Data/Service Integration Service APIs
- Map widget
- HostIP GeoCoding
- Log entries
- IP address
- Coordinates
- Address
- Name/Address

Visualization
- UI Integration
- Mashup Integration Logic (Data Flow Pipeline)
Conclusion

- Mashup design benefits from separation of UI from integration logic layers
- Data flow pipelines help end-users to quickly build the integration logic of mashups
- JOpera provides a visual composition environment for mashing up predefined and reusable components
- The JOpera Mashup Engine is extensible with data ports supporting rich interaction semantics (Request-Response, One-Way, and Streaming)
Let it Flow: Building Mashups with Data Processing Pipelines

Biörn Biörnstad  
Department of Computer Science  
ETH Zurich, Switzerland  
bioernstad at inf.ethz.ch

Cesare Pautasso  
Faculty of Computer Science  
University of Lugano, Switzerland  
c.pautasso at ieee.org