Composition, EUP, Reuse: Synergies

Boualem Benatallah

Professor, UNSW, Sydney, Australia
Chaire d’Excellence, Region d’Auvergne, LIMOS, Clermont-Ferrand, France
Agenda

- Development, Composition and EUP
- Observations
- Synergies
Gartner Quote on Composition

- Through 2014, the act of composition will be a stronger opportunity to deliver value from software than the act of development.
- By 2015, people will spend more than 80% of their time working collaboratively — but not necessarily face to face.

(Composition and BPM Will Change the Game for Business System Design, Daryl Plummer and al., Gartner report, 2009)
Composition vs Development

- Bottom up / top down
- Explicit / implicit interactions
- Coarse / fine grain
- Executable - non executable artifacts / code
- Iterative and incremental / waterfall
- More pronounced evolution/collaboration

Focus on programming productivity of composition
Effective Composition (dev. base)

Drivers

• Digital natives

• Extended role of IT (developers, end users, programmers)

• Web democracy (resources, applications, data, people, remix culture)

• Enhanced accessibility (code snippets, UI widgets, mashups, semantic tagging)

Implications

• Increased productivity, cost reduction, global visibility

• Extend the reach of IT-driven innovation

• Simplify collaboration

• Higher flexibility
Example 1: Services composition

- **Check Local Stock**: inStock=false
  - Check with supplier: shippingAvail=false
    - Cancel Order
  - Check with supplier: shippingAvail=true
    - Confirm Order
- **Check Local Stock**: inStock=true
Example 2: Scientific processes

Data is acquired from mass spectrometer. The format is native to the instrument used.

The native binary data is converted to a standard XML-based intermediate format.

The data is converted to peak list format. Instrument-specific algorithms are used.

Remove low-quality spectra from peak list or determine the charge state of the precursor ion.

Analysis of peak list by a database search engine such as Mascot.

Analyse the peptide or protein identifications. Statistically analyze for proteomics results.

Biological sample analysis by mass spectrometry

Convert binary data to mzXML format using RedAW*.

Convert mzXML to PeakList using mzXML to Other*.

PeakList filter

PeakLists

Processed PeakLists

Database search using Mascot*.

Statistical analysis using ProValst.

Database search results

Final results

Biological information
Why composition?

- Integration: EAI and B2B
- Productivity: Repetitive tasks (e.g., scientific processes, personal / long-tail processes), via script reuse and customization
- Sharing (feedback, community)
- Various types of mashups: feeds aggregation, mobility, maps, visualization, and many other of course
Observations (1)

- Current composition techniques are mainly for programmers
- To some extent, mashups and scripting are also used by non-professional programmers (users who can write/customize some programs)
- But, the reach is fast changing
- More focus on making composition easy (e.g., API developers, screen scrapers like dapper)
Observations (2): challenges

- Concept / abstraction mismatches (also, users may use different labeling for same action)
- Programming constructs are not meant to be intuitive (variables, control flow, symbols) although programming evolved over the years to the high level end
- Heterogeneity (data integration)
- Versioning and tracing is beyond the capabilities of average programmers, yet not systematically and intuitively supported in most composition environments
- Understand all components behavior, interfaces to build composites
- Quality of components is hard to characterize
How about EUP?

- Objective: allow anyone (not afraid of playing with computers) to program with very little learning efforts or at least not learning non intuitive programming constructs
- Programming is not a core skill
- Scripting or programming by demonstration
- Examples of end-user programming paradigms
  - Scripting languages (e.g., configuration of scientific experiments, Python, Flash, Perl, PHP, etc)
  - Spreadsheets (e.g., compute student marks)
  - Web macros, e.g., search hotel, pay bills

CoScripter

- CoScripter (IBM, available since 2007)
  - Create Web macros (command line like scripting language, go to, select, click, etc)
  - A repository of scripts
  - Keyword based search for scripts
  - Recommendation of scripts based on download counters, ratings, reviews, etc
  - More accessible, integration/concept diversity still not handeled effectively
EUP vs Composition

• EUP
  – Write a new program
  – Reuse (copy, changes/customization)
  – Reuse (copy, parameter configuration)
  – Macros (sequence of programs)
  – Generalize from examples

• Composition
  – Calls to APIs (services)
  – Explicit data/control flow

• Code vs Service/flow sharing

• Synergies: SOA for EUP (services scale in EUP), EUP for composition (increasing accessibility to services)
Spreadsheets for composition

• Drivers
  – Flexible and simple data model (unstructured, string/numbers, no complex types and variables)
  – Formulas: spatial relationships among cells (no explicit control and parallel vs sequential reasoning)
  – Incremental approach for building fairly complex applications with immediate feedback,
  – Analysis and manipulation functions spanning various application domains, e.g., financial, statistics

• Accessing services from spreadsheets
  – Spreadsheet extensions (e.g., MS Excel, StriKeIron, AMICO:Calc)
  – Object/service cells (e.g., A1 / Java, HUSKY/control not transparent)
  – APIs to access spreadsheets (e.g., Zoho, google spreadsheets, EditGrid)
Spreadator - Overview (1)
(WWW 09, EDBT 08, CAISE 08)

1. How to access heterogeneous data sources?

2. How to support structured data in spreadsheets?

3. How to provide a reuse-driven approach for Web data mashups development?

4. How to synchronize spreadsheet data and Web data?
Spreadator - Overview (3, import mapping)

Data Sources:
- Stock
- News
- Company
- Contacts
- Orders
- Suppliers

Data model:
- Grid

Data access API (REST):
- JDBC
- SOAP
- JSON
- Custom

Service Client on Spreadsheet Application:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Stock[001]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Symbol</td>
<td>B1.Symbol</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>% Volume Change</td>
<td>B1.Volume</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>% Price Change</td>
<td>B1.Price</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data-to-Tabular Mappings:
- B1 = Stock[001]
- B2 = "Symbol"
- B3 = "% Volume Change"
- B4 = "% Price Change"
- C2 = Stock[001].Symbol
- C3 = Stock[001].Volume
- C4 = Stock[001].Price
### Spreadator - Overview (4, patterns)

**Side-by-side comparison of stock quotes (Table pattern)**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>StockTable</td>
</tr>
<tr>
<td>2</td>
<td>Symbol</td>
</tr>
<tr>
<td>3</td>
<td>RATE</td>
</tr>
<tr>
<td>4</td>
<td>BEAT</td>
</tr>
<tr>
<td>5</td>
<td>THOR</td>
</tr>
<tr>
<td>6</td>
<td>LHCG</td>
</tr>
<tr>
<td>7</td>
<td>NTLS</td>
</tr>
</tbody>
</table>

**Summary of stock quotes**

**Detail of a particular stock in another worksheet**

**Fast access to detailed information of an item of interest (index pattern)**

**A list of news related to a particular stock in another worksheet (Relationship Index pattern)**

**A list of stocks and their related contact information organized in a nested structure (Hierarchical pattern)**

**Stock information with user-defined labels (Content pattern)**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>StockDetails</td>
</tr>
<tr>
<td>2</td>
<td>Symbol</td>
</tr>
<tr>
<td>3</td>
<td>Change in Volumn</td>
</tr>
<tr>
<td>4</td>
<td>Change in Price</td>
</tr>
<tr>
<td>5</td>
<td>News</td>
</tr>
<tr>
<td>6</td>
<td>StockList</td>
</tr>
<tr>
<td>7</td>
<td>Symbol</td>
</tr>
<tr>
<td>8</td>
<td>Volumn%Change</td>
</tr>
<tr>
<td>9</td>
<td>Price%Change</td>
</tr>
<tr>
<td>10</td>
<td>Symbol</td>
</tr>
<tr>
<td>11</td>
<td>Volumn%Change</td>
</tr>
<tr>
<td>12</td>
<td>Price%Change</td>
</tr>
</tbody>
</table>

**Stock details shown in a list (Repeater pattern)**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Contact</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Prospect</td>
<td>ID</td>
<td>Name</td>
<td>ACC_Owner</td>
</tr>
<tr>
<td>3</td>
<td>RATE</td>
<td>3088299</td>
<td>56388</td>
<td>Elizabeth Alex…</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>Cell C</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>BEAT</td>
<td>1111111</td>
<td>44455</td>
<td>SK</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>2222222</td>
<td>55544</td>
<td>SK</td>
</tr>
</tbody>
</table>
MashArt - Guiding principles (ER 09)

- **Universality**: UI, application logic, and data
- **Synchronization and orchestration**: bring together UI and application/service logic
- **Lightweight** composition model: no transactions, no exceptions/compensations
- Complexity **inside** components
- **Standard** layout and graphics tools
Incremental Reuse (1)

- Offset complexity with incremental knowledge capture and reuse

- Composition knowledge - CK (APIs, data transformation rules, composition fragments, versions, ...)

- Modularity and automated support for linking services

- Patterns (e.g, data transformation, data presentation)
Incremental Reuse (2)

• Increased accessibility (e.g., tagging, controlled NL processing, effective indexing of CK - bridge concept/abstraction devide, terms diversity)

• Search/QA (social networks)

• Characterization of reusable components

• Incremental and intuitive capture of CK (common knowledge, + behavior rules, immediate feedback)
Summary

• EUP/Mashups/CK capture and reuse/ universal composition to unlock services, for both IT and domain experts contribute added value services.

• Some key challenges:
  • How to make composition models/tools that are simple enough and useful enough?
  • How to build reusable components? What are the characteristic of a “good” reusable component?

• Composition knowledge/recommender systems
• Declarative/natural composition