How conceptual system architecture leads to business processes

Hernan Astudillo
Soluciones S.A.
hernan@acm.org
www.ime.usp.br/~ha/slides/oopsla2k.ppt
Context
Project context

- Allow payments from Financial Institutions (FI) to various Federal Agencies
- In actuality, these payments are
  - ...collected by the Treasury
  - ...on behalf of the Agencies
  - ...and reported to the Federal Reserve Banks (FRBs)
- FIs may be anywhere in the world
- Agencies are located mostly in D.C. area
- 37 FRBs located across the U.S.
Project goals: functional

- Deposit reporting
- Accounting classification for cash management purposes
- Reconciliation tools for client agencies
- Consolidated financial reporting
- Cash forecasting (deposits & disbursements)
Project goals: non-functional

- Consolidated, standardized information
  - all-electronic
  - open architecture
  - single on-line stream to agencies
- Simplified access to information
  - in format (standardized) and platform (web)
- Robust and adaptable
On discovering
Relevant business processes
Reasoning

- System allows:
  - FIs pay to Agencies
- Well, not really:
  - Treasury collects and keeps this payments
- Well, not really:
  - fund transfers happen inside the FRBs
- So we support an accounting fiction
- ...but this doesn’t help to understand reconciliation and forecasting
Key representation

2000-10-18 Astudillo / OOPSLA’2000
Key insight

- Paying and reflecting are architecturally different
  • OLTP, available, real-time, geographically replicated vs. datahouse, 9-to-5, batch, central location
- Payment processing ? post-facto analysis
- Hence, two system halves:
  • “payments mediator”
  • “analysis enabler”
- Business processes!!
  • (so sayd the analysts)
The two "lobes"
Key representation

- Image of an Agency's payments
- Image of Treasury's payments

Flows:
- Agency
- Treasury
- Federal Reserve Bank

2000-10-18  Astudillo / OOPSLA’2000
“Lobes”? 

- Lobe:
  - aggregate of business functions demanding similar mechanisms across several dimensions (e.g. availability, distribution, interactivity...)
- “This system is composed of two lobes”...
- ...or rather:
  - “this system’s overall functionality can be meaningfully aggregated into two large sets with meaningful architectural differences"
## Characteristics of the lobes

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Payment mediator</th>
<th>Analysis enabler</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Timeliness</strong></td>
<td>Near-time processing</td>
<td>Batch processing</td>
</tr>
<tr>
<td><strong>Typical transactions</strong></td>
<td>OLTP-like</td>
<td>DSS-like</td>
</tr>
<tr>
<td><strong>Synchronicity</strong></td>
<td>Event-driven</td>
<td>Data-driven</td>
</tr>
<tr>
<td><strong>Availability</strong></td>
<td>24x7</td>
<td>Daily</td>
</tr>
<tr>
<td><strong>Location</strong></td>
<td>Distributed, fail-over</td>
<td>Centralized (internal replication)</td>
</tr>
<tr>
<td><strong>Reliability</strong></td>
<td>Extremely high ($)</td>
<td>Plain old high</td>
</tr>
</tbody>
</table>
Lobes architecture

- **Payment mediator**
  - fully distributed, fully replicated, fail-over replicated, supporting only payment processing
  - all historical analysis (i.e. large volumes of data) must be left to analysis enabler

- **Analysis enabler**
  - centralized, internally replicated (for load-balancing and availability), data-centric architecture
  - all near-time payment processing must be left to payment mediator
On documenting
Architectural strategy

- **Strategy**: “multi-layered derivation”
- **Aspects**:
  - Refinement
    - different abstraction and completeness levels
  - Layered description
    - diagrams and entities for each layer
  - Traceable refinement
    - traceability matrices
- 1) Lobes
  • aggregate business functions that demand similar mechanisms

- 2) Components
  • large-grain software or data pieces
  • each component offers “services” to manipulate what is conceptually a single unit
  • typical component kinds: lifecycle, reporting, importing [details later]
The 5 layers (2/2)

- 3) Services and connectors
- 4) Modules
  - pieces that can be treated as a single entity by a programming environment
- 5) Technical infrastructure
  - network/node topologies
  - deployment strategies
Typical components

• Typical component kinds:
  - lifecycle
    • allow CRED of a system-tracked entity
    • ...which can be internal or image of an external entity
    • ...providing basic retrieval strategies
  - reporting
    • generate reports on a system-tracked entity
    • ...providing complex/multiple retrieval strategies
    • ...with a content- & format-description mechanism
  - importing
    • accepts and converts external data meaningfully
    • conversion description mechanism
Level 1: lobes

- FMS
- CA$H-LINK
- payment
- reconciliation
- public monies
- sync
- activity
- other sys
- FRB
- STAR

FRB
agency
FI
monies
reconciliation
other sys
sync

payment mediator
analysis enabler

2000-10-18
Astudillo / OOPSLA’2000
Level 2: components (deposit)
### Level 2: traceability matrix

<table>
<thead>
<tr>
<th>Event</th>
<th>Actors</th>
<th>Component(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>#93 (Fedwire reversal)</td>
<td>Agency</td>
<td>Payment lifecycle</td>
</tr>
<tr>
<td>#68 (funds transfer instructions)</td>
<td>FA</td>
<td>Payment lifecycle</td>
</tr>
<tr>
<td>#25 (Fedwire reversal instructions)</td>
<td>FA</td>
<td>Payment lifecycle</td>
</tr>
<tr>
<td>#1 (deposit report)</td>
<td>FI</td>
<td>Payment lifecycle</td>
</tr>
<tr>
<td>#22 (deposit adjustment)</td>
<td>FI</td>
<td>Payment lifecycle</td>
</tr>
<tr>
<td>#3 (deposit deletion/reversal)</td>
<td>FI</td>
<td>Payment lifecycle</td>
</tr>
<tr>
<td>#102 (identif. Unclassified deposit)</td>
<td>FI</td>
<td>Payment lifecycle</td>
</tr>
</tbody>
</table>
## Level 3: components refinement

<table>
<thead>
<tr>
<th>Element</th>
<th>Element kind</th>
<th>Decomposition (used components)</th>
<th>Additional properties</th>
<th>Platform or product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Payment lifecycle</td>
<td>Component</td>
<td>Lifecycle</td>
<td>Fault-tolerant</td>
<td>Ad-hoc</td>
</tr>
<tr>
<td>Interaction to FI/FA</td>
<td>Component</td>
<td>Collection</td>
<td>Fault-tolerant (duplicated to RHS)</td>
<td></td>
</tr>
<tr>
<td>Payment mediator</td>
<td>Component</td>
<td>Persistence-LHS</td>
<td></td>
<td>Java/Oracle</td>
</tr>
<tr>
<td>Depositing</td>
<td>Component</td>
<td>Collection</td>
<td></td>
<td>Legacy-based</td>
</tr>
</tbody>
</table>
Conclusion
Conclusion

- Architectural experience may lead to identify business processes
  - lobes correspond to “real” things
  - business processes
- Multi-layered derivation is a good strategy
  - facilitates reasoning
  - provides traceability
How conceptual system architecture leads to business processes

Hernan Astudillo
Soluciones S.A.
hernan@acm.org
www.ime.usp.br/~ha/slides/oopsla2k.ppt