MAKING THE MOVE TO SOA IN TEXAS GOVERNMENT

Rick Goldgar, Ph.D.
Chief Technology Officer and Deputy CIO
Texas Education Agency

David E. Butler
Project Manager
Texas Education Agency
August 12, 2008
WHO IS THIS RICK GOLDGAR?

- 20+ years in commercial software product development, management, marketing:
  - Systems Simulation / Code generation
    - 1982-1984 – screen based automated code generation multiple languages
    - 1990 – 1995 – systems simulation and OO analysis/design/code generation
    - Worked with key industry OO methodologists (Shlaer/Mellor)
  - State Government
    - 2002 – Current Texas Education Agency
  - Aerospace
    - 1984-1990 – Aerospace, embedded systems (Tracor, now British Aerospace)
  - Systems Administration / Monitoring tools
    - 1995-2000 DB / Application management (BMC Software)
    - 2000 – 2001 Application monitoring / management
WHO IS DAVID BUTLER

- 18 years in commercial/public utility/government software development/management:
  - Commercial Systems Development
    - 1988-1993 – Software Developer for AT&T Bell Laboratories (now Alcatel/Lucent)
    - 1993 – 1998 – Project Manager for IBM Global Services & Advantis
  - Public Utility
    - 1998 – 2002 – I.T. Program Manager for Entergy Services
      • Nuclear Generation and Distribution Business Units Software
  - State Government
    - 2005 – Current I.T. Program Manager Texas Education Agency
Texas Education Agency - IT Statistics

- IT development staff of about 150 FTEs and contractors
- 80+ legacy and new applications, supporting:
  - Texas K-12 Public Schools Funding, Grants, Entitlements (> $20B /yr in funds management / distribution)
  - Textbook management (> $150M / yr)
  - Data gathering / analysis / reporting / compliance (>4M schoolchildren, > 1200 school districts)
  - Teacher certification tracking (> 400K active teachers)
- Constant changes due to Federal and State Legislation
- Infrastructure under consolidation
  - Over three hundred servers (Windows and AIX)
  - Very little mainframe legacy
- New development is 80% C#, 20% Java
- Virtually all web based applications
- > 90% of our projects are successful.
  (The 2006 Standish report states the industry average is now 35%).
WHY ARE WE HERE TODAY?

We all want to do more with less:

- More functionality
- More flexibility
- More quality
- Less resources
- Less time
- Less money
MAKING SOFTWARE DEVELOPMENT MORE EFFICIENT

Software development has some of the qualities of factory work, except that we in the software world always seem to have to build the car factory and the cars.

To build an efficient, flexible factory takes careful, creative thought. It is an investment that will take more than one project to pay off and mistakes can be very costly.

Building our “software” cars (applications) correctly and efficiently using that factory requires good, integrated processes, good tools, and discipline.
WHY DO SOA?

Someday (someday) we want to lower the cost and shorten the cycle of application development, through:

- Reusability (wherever reasonable)
- Shared components (wherever reasonable)
- Standards (e.g. XML)
- A direct relationship between the business process and the resulting application
- Feedback loops and monitoring to improve business and application processes.

SOA claims to help do these things.
HOW WILL SOA DO IT?

The Grail:
Business **Process** Models
that *drive* Systems Models
that both *represent* Workflows
that *automatically translate* to executable code
that uses predefined shared services
in a scalable and fault tolerant architecture.

This would ideally be coupled with
the simulation of the business models
integrated with the monitoring of production systems
to allow validation and performance analysis and
explore opportunities for both business and systems improvement.
A MODEL OF MODEL DRIVEN DEVELOPMENT

**Business Model**
Developed by Business Analysts and Customers
Containing Workflow, People and Automated Tasks, References to Artifacts
Validated in Walkthroughs and Through Simulation

**Implementation Model**
Developed by System Integrator / Lead Developer
Containing Workflow, Graphical Specifications for Implementation,
Validated in Walkthroughs and Through Simulation

**Code**
(e.g. BPEL, Java web services, other code and SQL)
Automatically Generated from the Implementation Model

**Operational Integration**
Integration of code into SOA architecture
Support for Workflow Execution, Service Calls, and Task Management
THE VALUE OF THIS MODEL DRIVEN DEVELOPMENT MODEL

- Graphical model of the business that customers can understand, validate and see simulate
- Mappable Constructs from Business level to implementation level (e.g. artifacts > data structures, business level workflow > system workflow, etc.)
- Automatically generated from Implementation Model, exact representation of actual workflow, directly deployable, easily modified and validated.
- A standard mechanism for integrating workflow and services
- Reusable Components available to other applications
HOW DO YOU GET THERE?

- In the words of the band, REM:
  “Stand in the place where you live
  Now face North
  Think about direction
  Wonder why you haven't before.”

- More specifically:
  - Determine your level of SOA maturity
  - Start small
  - Get organized
  - Do initiatives that work for your maturity level
  - Evolve and build on successes
  - Expect to do things twice (or more) before you get there
  - Don’t be afraid to ask for help.
## SOA MATURITY

**“STAND IN THE PLACE THAT YOU LIVE”**

<table>
<thead>
<tr>
<th>Coders R Us</th>
<th>Pretty Pictures</th>
<th>Analysts R Us</th>
<th>Totally Business Driven Dudes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buried in Legacy / Not much OO</td>
<td>We do OO</td>
<td>Business / systems requirements</td>
<td>Model driven development</td>
</tr>
<tr>
<td>No Requirements documents</td>
<td>UML diagrams / Do UML and SOA go together?</td>
<td>Traceability of requirements to implementation</td>
<td>Business Model drives design / development</td>
</tr>
<tr>
<td>No standard processes</td>
<td>Some processes</td>
<td>Wrote and use some services</td>
<td>Integrated models / services</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wrap some legacy</td>
<td>Find and convert some code to services</td>
<td>Start doing business process modeling</td>
<td>Define business to systems to code translation</td>
</tr>
<tr>
<td>Start business / requirements analysis</td>
<td>Investigate mapping business processes to services</td>
<td>Investigate mapping business models to systems models</td>
<td>Exploit reusability at all levels.</td>
</tr>
<tr>
<td></td>
<td>Start understanding business process modeling</td>
<td></td>
<td>Investigate simulation and monitoring</td>
</tr>
</tbody>
</table>

Copyright © 2008 Texas Education Agency.
CHOOSING A PROJECT
DEPENDS ON YOUR MATURITY LEVEL

- **For Coders R Us and Pretty Pictures – Start small**
  - Look for candidate legacy code to wrap or repackage as services and reuse
  - Try some new small services that are clearly things you can reuse (e.g. login, DB lookup)
  - Do a project that can afford to fund the infrastructure required for true SOA
  - Work on the fringes of a well designed OO application to refactor useful pieces into services

- **For Requirements are our Business and Totally Business Driven Dudes – Get Focused and Abstract**
  - Focus on business processes and how to map them to systems
  - Work on rules for automatically translating models to code
  - Do a project that can afford to fund the infrastructure required for true SOA and that can provide clearly reusable service for other future or existing applications.
STAFFING RECOMMENDATIONS

- Systems Architecture support
- Business analysts / requirements analysts
- Systems Integrator(s) / Team Lead
- Service Developers
- Someone(s) with strong Websphere production knowledge
- Build Person
WHAT SKILLS YOUR TEAM NEEDS

- Business process modeling
- Systems modeling
- Service development coding (preferably Java, not too hard to find or teach)
- Abstraction (hard to find, harder to teach)
- BPEL knowledge if possible
- Websphere knowledge or help from outside
COMMON ISSUES

- **STAFFING ISSUES** – The transition to SOA includes a transition in staff roles. New roles may include:
  - Business / Requirements Analysts
  - Systems Integrators
  - Services Developers

- **BUSINESS LIFECYCLE ISSUES** – The greatest value to a business may be in *business process re-engineering* and this should occur prior to or at least early in the lifecycle, not halfway into the design. Rick’s rule #432 – Automating a bad business process just makes it run bad faster.

- **OLD THINK ISSUES** – Just because someone can make a piece of C# or Java into a “service” does not make it a useful, well defined or reusable component for your business. Be very aware of your staff and their interests or you may just get more regular OO code disguised as services, rather than services that are directly tied to business process needs and invoked in flexible workflows.

- **ABSTRACTION ISSUES** – Do not underestimate how important it is to get the right level of abstraction for your models and services. This is key to their reusability and finding folks who understand how to do this is hard.
HOW TEA IS DOING IT

- Starting with a specific project, with solid funding and a strong need
- Hiring the best and brightest we can find / afford (including the best and brightest contractors)
- Getting infrastructure in place in advance of the project
- Focus on model driven development and making that *work*, instead of making excuses why it won’t work.
- Getting help from IBM and others
<table>
<thead>
<tr>
<th>IBM Products</th>
<th>Likely Users</th>
<th>TEA Owns</th>
<th>TEA Uses</th>
<th>Someday May Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>W Business Modeler</td>
<td>Analysts / Dudes</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W Integration Dev</td>
<td>All</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W Rational App Dev</td>
<td>All</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W Process Server</td>
<td>All</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W Application Server</td>
<td>All</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W BM Publisher</td>
<td>Analysts / Dudes</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W Service Registry &amp; Repository</td>
<td>Analysts / Dudes</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>W Monitor</td>
<td>Post Dudes</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Adaptors</td>
<td>Coders R Us</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
FOCUS ON MODEL DRIVEN DEVELOPMENT

- If you start with a clear vision/picture, then you have a better chance of realizing it.
- Caveat: You need pictures that each of your stakeholder participants understand:
  - Business models for Customers
  - Implementation models for System Integrator
  - Operational models or specifications for Programmers
WHY MULTIPLE PICTURES?

- **Customer/Business User**
  - Sell vouchers
  - Collect vouchers
  - Store luggage
  - Transport up to 40 passengers
  - Refuel transport

- **Process Orchestrator**
  - 40 passenger bus
  - Transmission system domain
  - Seating config
  - Heat/AC system domain
  - Breaking system domain

- **Service Developer**
  - Tire
  - Wheel
  - Disc Break Assembly
WHAT SORT OF PICTURES WILL YOU NEED?

- Business Model
  - Describes the end-users processes/activities (User View)
- System Implementation Model
  - Describes the system needed by user to support above process (System Integrator view)
- Operational Model
  - Describes the atomic services that will be developed to implement system (Service Developer View)
  - Business Process Exec. Language (BPEL)
- Programming Code
BUSINESS MODEL

- Reflects the overall business process
- Includes all relevant enterprise and external processes
- Independent of project scope or implementation
- Business model workflow may be simulated
- Developed in Websphere Business Modeler
IMPLEMENTATION MODEL

- Includes just the workflow in scope of the system
- Use of existing services identified; new services defined
- Allocation of tasks to the technology of implementation
- Addition of implementation details
- Developed in Websphere Business Modeler
OPERATIONAL MODEL

- Generated from Implementation Model
- Workflow, transaction, and fault tolerance details added
- Developed in Websphere Integration Developer
- Generated BPEL (Business Process Execution Language) orchestrates workflow using Websphere Process Server
THINGS TEA MIGHT HAVE DONE BETTER

- Websphere Systems Administration
- More aggressive search for model driven development experts (Early Abstraction is KEY)
- Picked a project that did not get caught in the middle of mandated legislated changes
CONCLUSIONS

- Business and Requirements analysis are important to truly successful SOA

- The Grail is model *driven* development and it is pretty hot stuff. It takes talent and work. Still, you can certainly start the SOA path by just creating and using some reusable services

- It will typically take a few years to realize the bigger gains of SOA, even if you do it right.

- If you do not keep your team focused on the goal and don’t foster good abstraction skills, SOA is no more or less useful or valuable than traditional OO or other types of development.
QUESTIONS
&
CONVERSATIONS
MAKING THE MOVE TO SOA IN TEXAS GOVERNMENT

Rick Goldgar, Ph.D.
Rick.Goldgar@tea.state.tx.us
Chief Technology Officer and Deputy CIO
Texas Education Agency

David E. Butler
David.Butler@tea.state.tx.us
Project Manager
Texas Education Agency
August 12, 2008