Our perspective on Service Oriented Architecture (SOA)

- **SOA is:**
  - *An enterprise and business architecture approach* – a way to understand and integrate the enterprise in the context of its community and as a network of business services. “A SOA” at the business level is part of the enterprise architecture showing how this network of services delivers business value.
  - *A system of systems solution architecture* – a way to understand and integrate enterprise systems internally and externally as a network of technology services. “A SOA” at the systems of systems level is the solutions architecture showing how this network of systems works together to delivers business value.
  - *A system integration approach* – a way to expose existing capabilities to integrate applications and create new composite solutions.
SOA Hype and Reality

• This is easy – no planning required
  – Hype: “Just start exposing capabilities as services – use these to make new services and “mash up” applications”
  – Reality: Service anarchy is a road to disaster – architect for longevity and loose coupling

• Network of Services
  – Hype: Services are simple and stand-alone
  – Reality: Services can be complex and interdependent

• Suitability, process and trust
  – Hype: Dynamically find and use services from across the internet
  – Reality: Mission critical use of services requires trusted and reliable services from known parties
Spaghetti Enterprise Example

Offers a Service
Uses one Of the Services
Offers a New service
Builds an Application
Using other services
Services Use Services
Anti-Agile Service Dependencies

Core problem – services “bottom up” – not architected or standard

Changed or Discontinued
SOA Simplification

- Services are limited to a single interaction between provider and consumer
- There is just a “request message” and a result
Reality – Multiple services work together

The Asset service collaboration shows the service interactions supporting the management of asset records and the posting of general ledger as a result of asset transactions.
Services can be complex and long lived interactions

Call for fire is a “protocol” of action and information exchange between parties
Trust and Validation

Clear, Trusted and Validated Service Effects

Test

Service Contract

FIRE

Under All Conditions
Message – Don’t forget the “A” in SOA!

• A Service Oriented Architecture Should:
  – Treat the enterprise as “service oriented” at the business level - people and organizations provide and use services
  – Define technology services to augment and enable business services
  – Define “service contracts” so all parties know what to expect
  – Abstract “enterprise services” from solution specific capabilities
  – Provide for reuse and longevity
  – Define how services collaborate to provide business value
  – Separate service contracts from implementing technologies
  – Provide an integrated view of processes, collaborations, information and services – without coupling these together
Business Focus Using Model Driven Services Architecture

MDA Terms

Computation Independent Model

Platform Independent Model

Platform Specific Model

Business Concerns

SOA Business Model
Business Services (b-SOA)
Roles, Collaborations & Interactions
Process & Information

Logical Systems of Systems Model
Technology Services (t-SOA), Components
Interfaces, Messages & Data

Technology Specification
Web Services
WSDL, BPEL, XML Schema

Refinement & Automation

Line-Of-Sight

Computation Independent Model

Platform Independent Model

Platform Specific Model

18/06/07

Copyright © 2007 Model Driven Solutions
Core of services architecture

• **Services contract**
  – What is the service provided?
  – Under what terms and guarantees?
  – What are the interactions – information and assets exchanged?
  – What is the data?
  – How is the services choreographed?

• **Service Collaboration**
  – Why do parties work together – what is the business goal?
  – What roles do they play in this collaboration?
  – What services does each party provide and use?

• **Process**
  – Externally, what process or processes does the collaboration serve?
  – Internally, what activities take place to enact a service?
  – What resources are consumed or produced?

• **Information**
  – What is the core information (ontology) of the domain
  – What information is exchanged to enact services
  – What information is retained and shared
Automate from Architecture model to technology

Model Driven Architecture Standards

Enterprise Architecture Model (CIM)

Infrastructure Mapping (E.G. J2EE-WS)

Enterprise Service Components

Framework & Infrastructure (E.G. -J2EE-WS)

Tools Produce & Integrate

Mapping is tuned to the infrastructure

Minimize and structure manual implementation

Technical Architecture
Model Driven Solutions

- Business Enablement
  - Business Transformation
  - Enterprise Agility
  - Enterprise Integration
  - Service Implementation

- Architecture
  - Enterprise Architecture
  - Business Architecture
  - Service Oriented Architecture
  - Model Driven Architecture
  - Business Process Architecture

- Open Source
  - www.ModelDriven.org – Open community for MDA, SOA and the Semantic web

- Opportunity to solution – architected, fast, strategic
Backup Example
Example Call for Fire - MLRS

Call for fire is a “protocol” of action and information exchange between parties.
Call for Fire - MLRS

Bn-FSE -> Bde - FSE -> Bde - Artillery

Co-FIST

Forward Observer

FA Bn - FSE

MLRS Battery

MLRS
Model Information Flows

Call For Fire
Fire Ready
Rounds Complete
Fire Shot
Fire Splash
Mission Fire Rpt
End Of Mission

* Not technology details!
Choreography – Understanding When
Generated Web Services Definition

```xml
<wsdl:portType name="CustomerOrderEstablishment.CustomerOrderEstablishment">
  <wsdl:operation name="CustomerOrderEstablishment">
    <wsdl:input message="tns:CustomerOrderEstablishmentPanopticInheritanceCluster"
               name="CustomerOrderEstablishment"/>
  </wsdl:operation>
</wsdl:portType>

The primary port type has operations corresponding to the request flows in the protocol.

<wsdl:portType name="CustomerOrderEstablishment.CustomerOrderEstablishmentCallback">
  <wsdl:operation name="CustomerOrderEstablished">
    <wsdl:input message="tns:CustomerOrderEstablishedPanopticInheritanceCluster"
               name="CustomerOrderEstablished"/>
  </wsdl:operation>
  <wsdl:operation name="CustomerOrderEstablishmentRejected">
    <wsdl:input message="tns:CustomerOrderEstablishmentRejectedInheritance"
               name="CustomerOrderEstablishmentRejected"/>
  </wsdl:operation>
</wsdl:portType>

The callback port type has operations corresponding to the response flows in the protocol.
```
Example Transaction Message XML Document

```xml
<CustomerOrderEstablishment>
  <customerOrderEstablishment>
    <newOrder>
      <customerOrder>
        <customerOrderID> ... </customerOrderID>
        <customerOrderAmount> ... </customerOrderAmount>
        <orderingCustomer>
          <customer>
            <customerID> ... </customerID>
          </customer>
          <party>
            <name> ... </name>
          </party>
        </orderingCustomer>
        <controllingSalesInstrument>
          <salesInstrumentID> ... </salesInstrumentID>
        </controllingSalesInstrument>
        ... 
        <lineItems>
          ... 
        </lineItems>
      </customerOrder>
    </newOrder>
  </customerOrderEstablishment>
  <businessDomainTransaction>
    <transactionID> ... </transactionID>
  </businessDomainTransaction>
</CustomerOrderEstablishment>
```