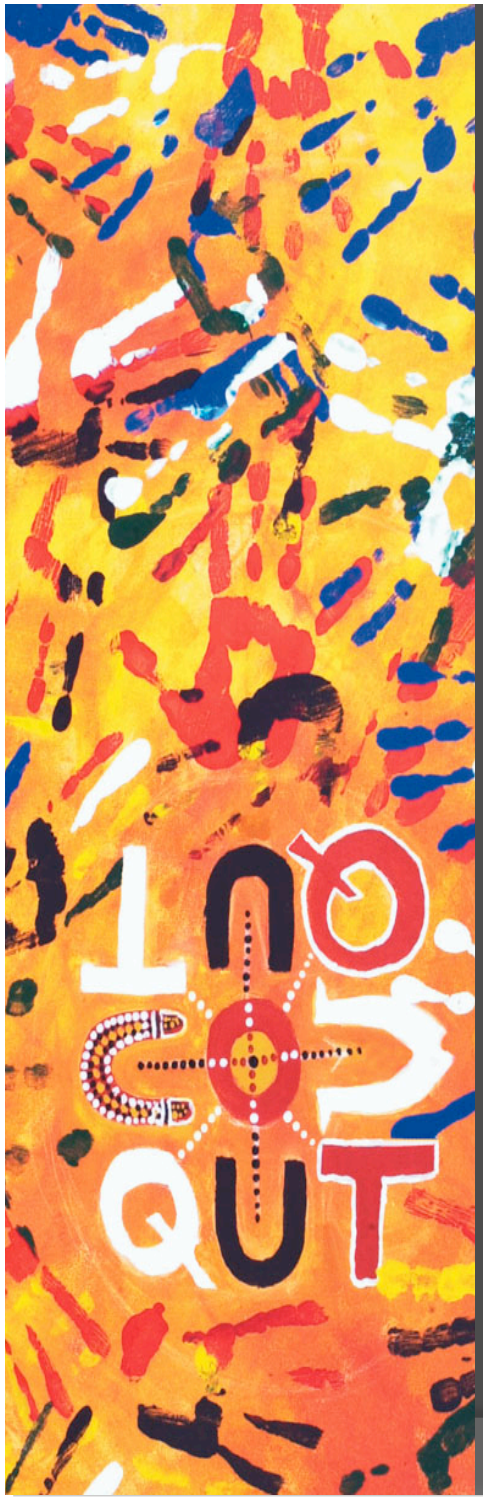
WHAT WOULD SMART SERVICES LOOK LIKE?

And how can we build them
on dumb infrastructure?

Keith Duddy (keith.duddy@qut.edu.au)
QUT/Smart Services CRC, WESOA Workshop, 1 Dec 2008

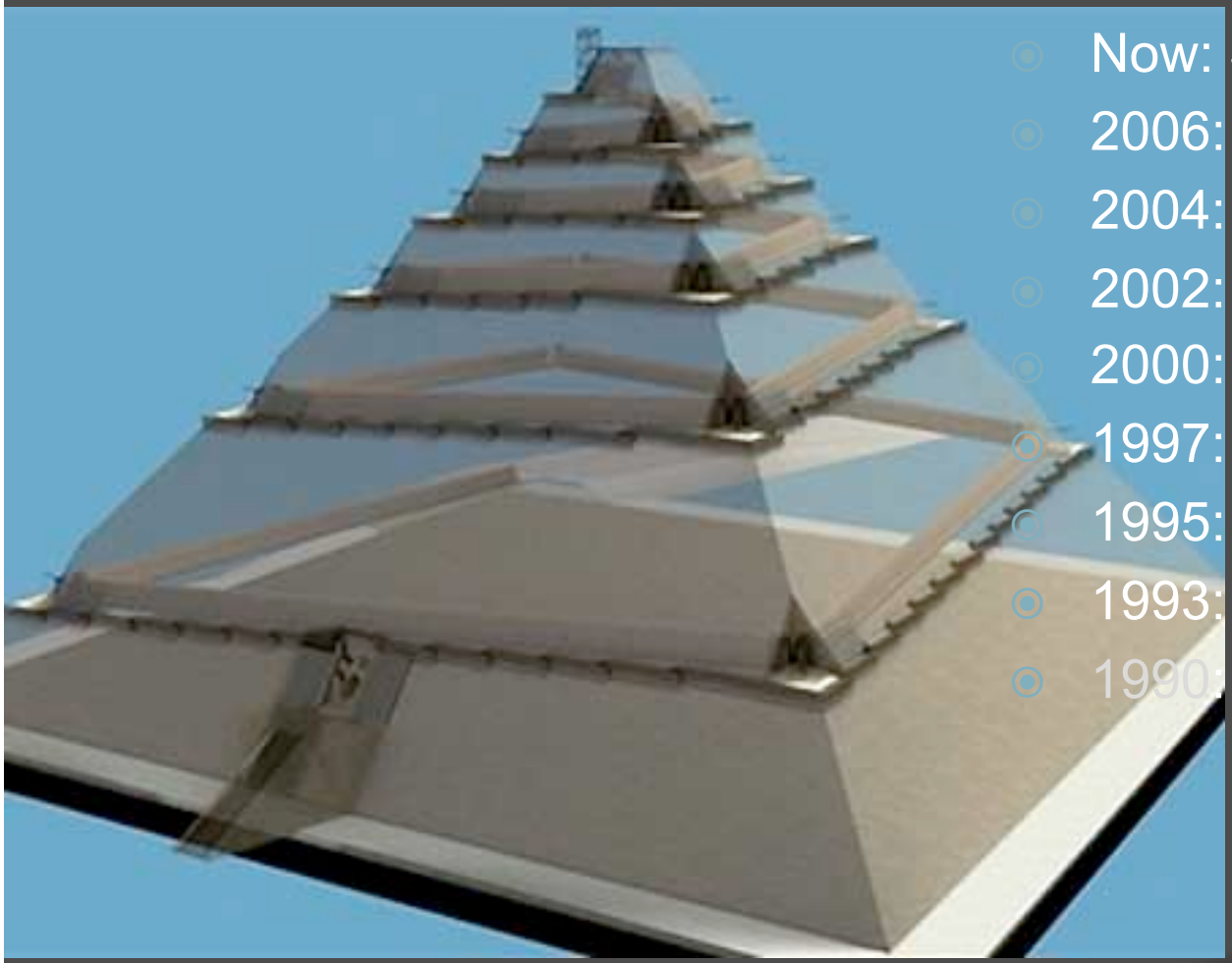


Reconciliation

In keeping with the spirit of Reconciliation, I acknowledge the Cadigal people who are the traditional owners of the land on which we are meeting today, and acknowledge the important role Indigenous people continue to play within the QUT community.

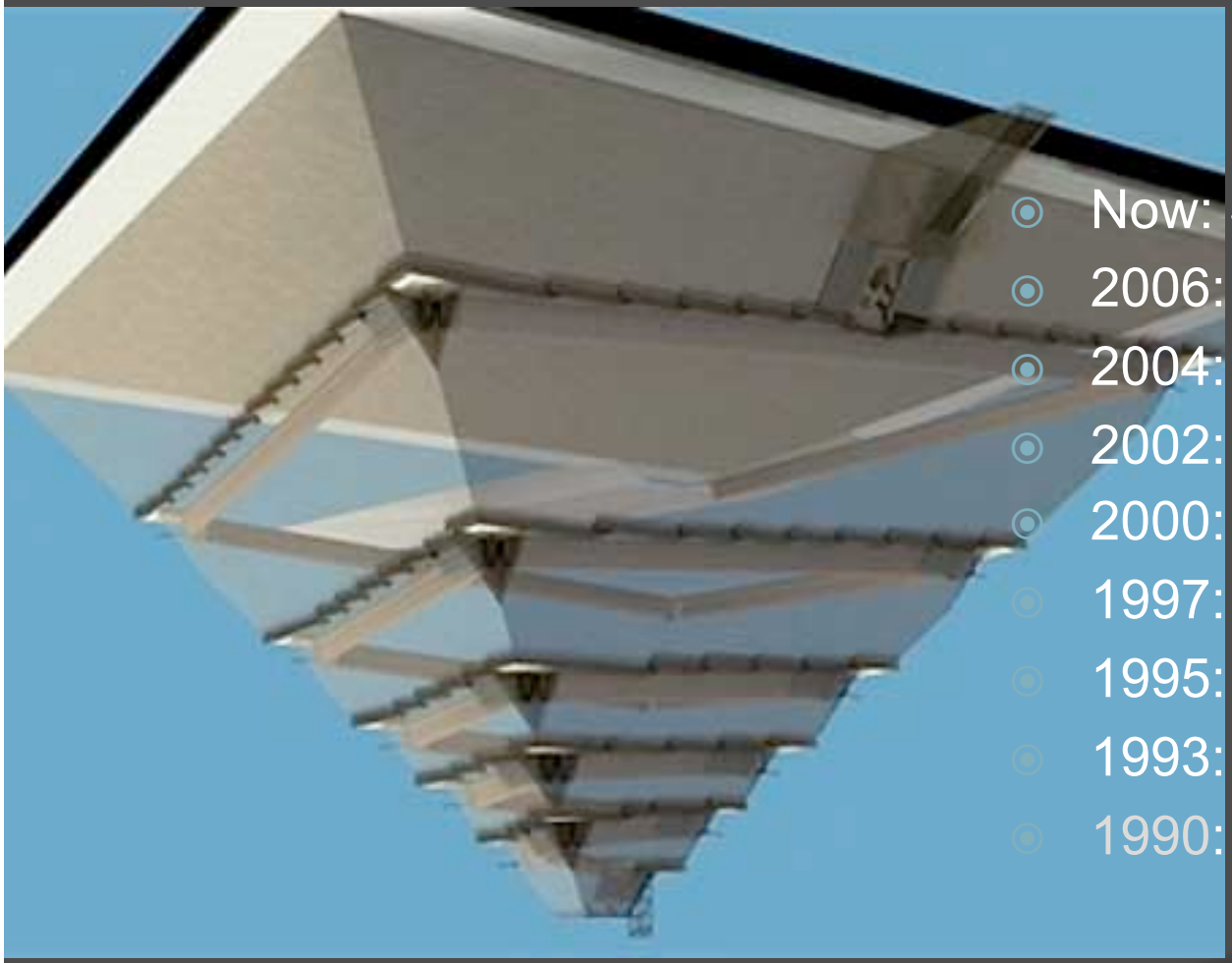
www.reconciliation.qut.edu.au

A little about me



- Now: Smart Services CRC
- 2006: NEHTA
- 2004: Model Transformation
- 2002: Models of Enterprise Systems
- 2000: Metamodels & Repositories
- 1997: CORBA Components
- 1995: CORBA Trader & Notifications
- 1993: Network Protocol Drivers
- 1990: Operating Systems Drivers

Or should it be...



- Now: Smart Services CRC
- 2006: NEHTA
- 2004: Model Transformation
- 2002: Models of Enterprise Systems
- 2000: Metamodels & Repositories
- 1997: CORBA Components
- 1995: CORBA Trader & Notifications
- 1993: Network Protocol Drivers
- 1990: Operating Systems Drivers

Outline

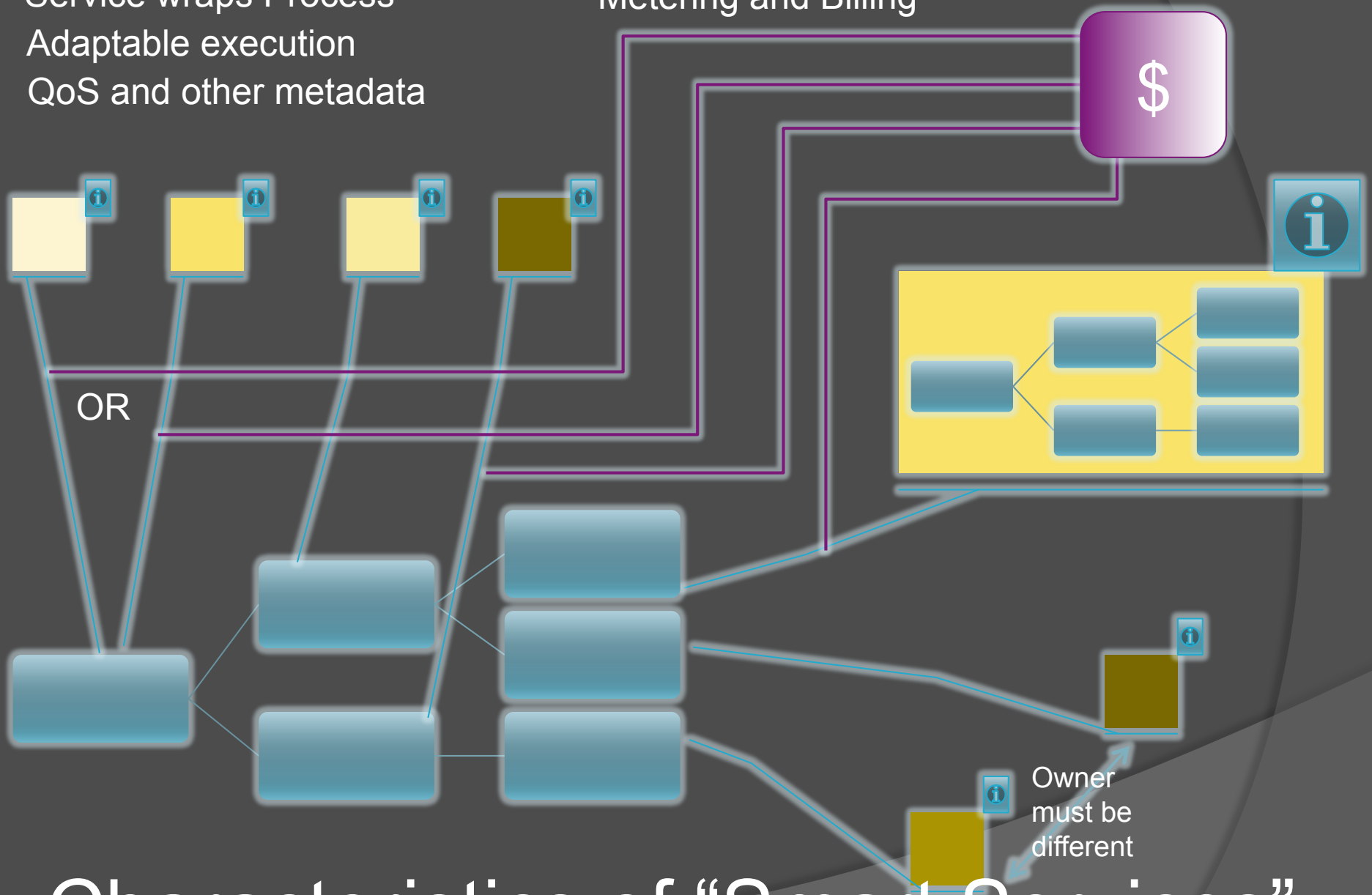
- What is a “Smart” Service?
- Overview of relevant Smart Services CRC projects
- How does this relate to Web Services (WS*)
- What’s wrong with the WS* & BP* Platforms?
- What other challenges do we face?
- Some practical initiatives to overcome challenges

What is a “Smart” Service?

- Better to think of infrastructure to enable “smarts” of regular web services
  - Selection of “best fit” services
    - based on QoS, context or policy
    - dynamic and static selection
  - Support for service meta-data to allow fitness to be discovered
  - Aggregation of services within Business Processes
    - Allowing the BP to be the context for selection
    - Exposure of aggregations as first class services
  - Integration with payment, logging & auditing
  - Replacement of services with “similar” services
    - Using ontologies, AI, other approaches

Process Tasks invoke Services
Service wraps Process
Adaptable execution
QoS and other metadata

Constraint enforcement
Metering and Billing

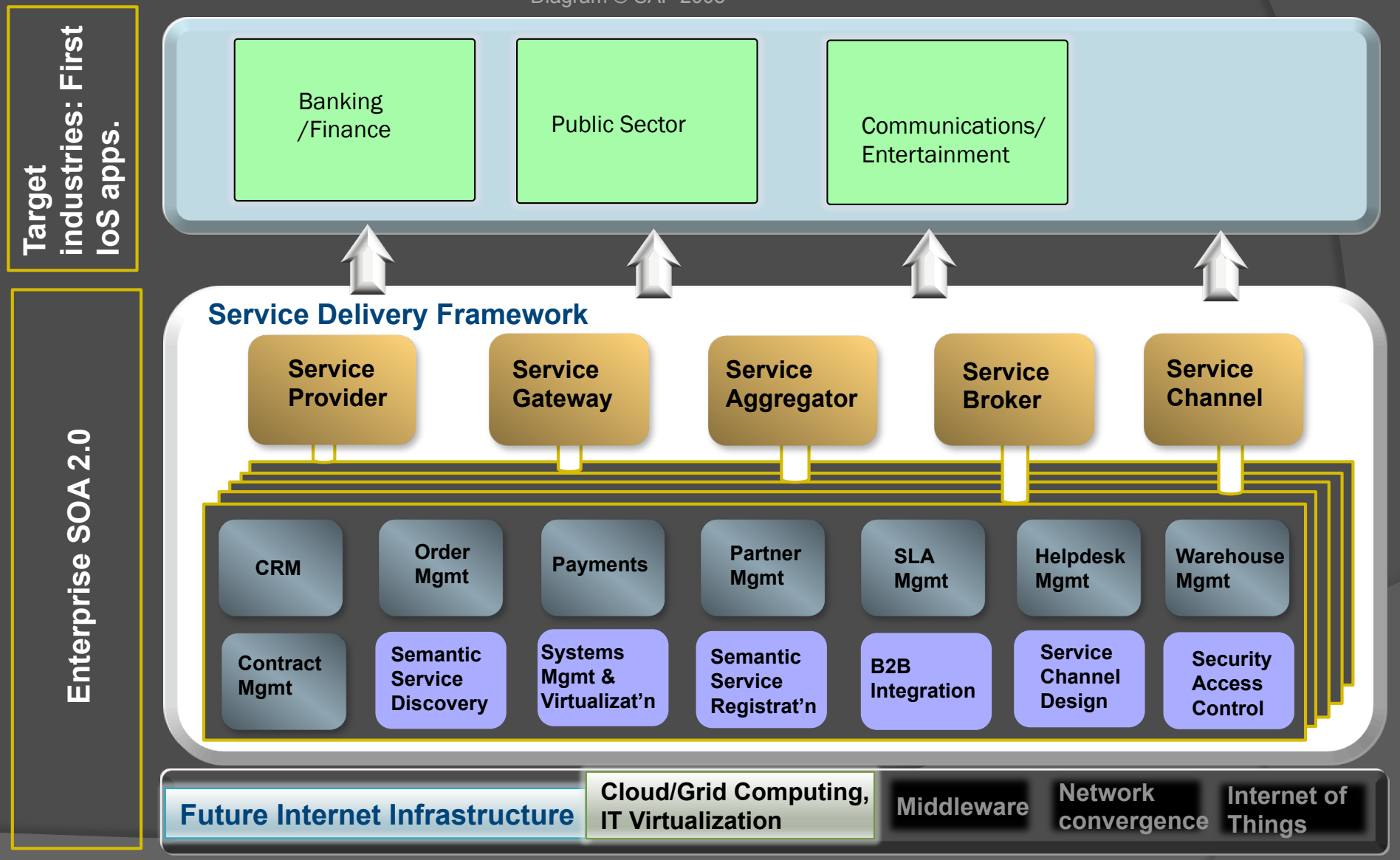


Characteristics of “Smart Services”

Overview of relevant Smart Services CRC projects (Year 1)

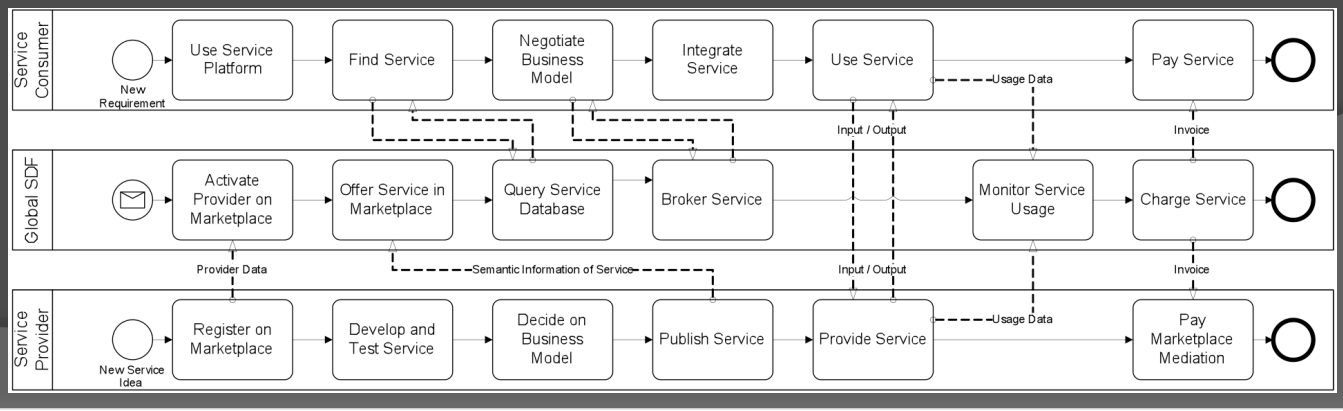
- Service Delivery Framework
  - Architecture & Vision
  - Service Delivery Use Cases from industry/govt
  - Service Broker
- Service Aggregation
  - Architecture & Vision
  - Service Aggregation Use Cases
  - BPM-based Service Aggregation Engine(s)
    - QoS and Constraint aware
    - Adaptive to changing environment
    - Range of enactment optimisations
  - Lightweight (Web UI-based) Aggregations

Enterprise SOA Big Picture



Service Broker

- Mediates Access to Web Services
  - Deployment
  - Metering/Payment
  - Discovery
  - Security/Trust
- Acts as a Marketplace of Services in some domain



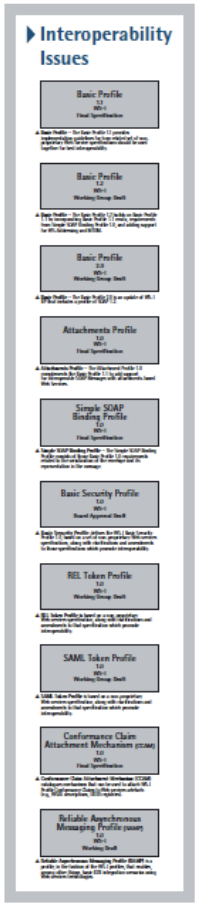
Déjà Vu

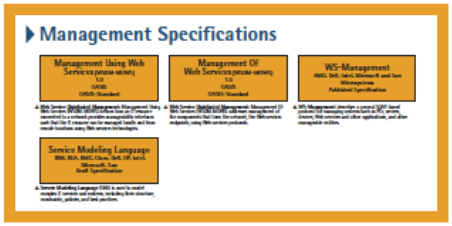
- Open Services Marketplaces have been promised since the early 90s
  - CORBA/ODP Trader (1995)
    - Stored standard metadata on services
    - Allowed selection based on query language
    - Federated model
  - UDDI (2001)
    - Stored standard metadata on services
    - Allowed selection based on criteria
    - Federated model
    - Global deployment by SAP, IBM & Microsoft until plug pulled in 2005
- Has anything changed?
  - Web Services have have gained critical mass?

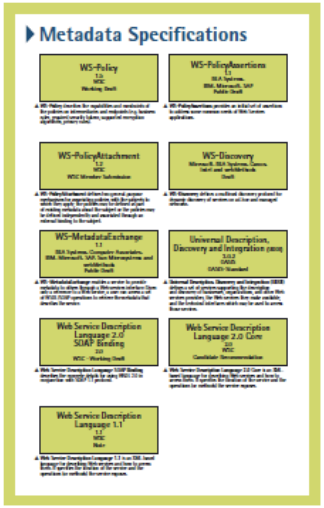
How does this relate to WS* ?

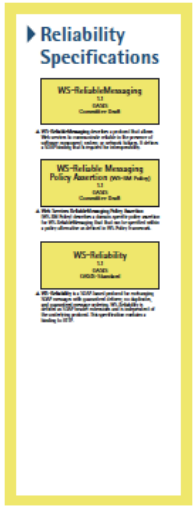
- SOAP and WSDL are assumed
  - But what style? RPC, Literal, Wrapped, Document? Any XML?
- WS Addressing is well supported by vendors
- WS Security is implemented, but divergent
- WS Reliable Messaging ??
- WS Policy ??
- WSFL, WS-Coordination, WS-Transactions, …

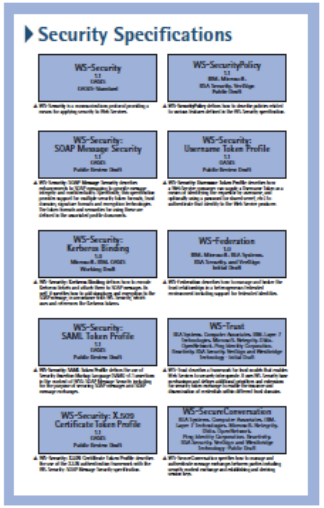
Web Services Standards Overview

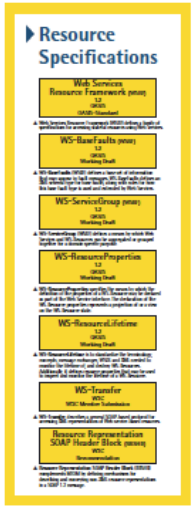


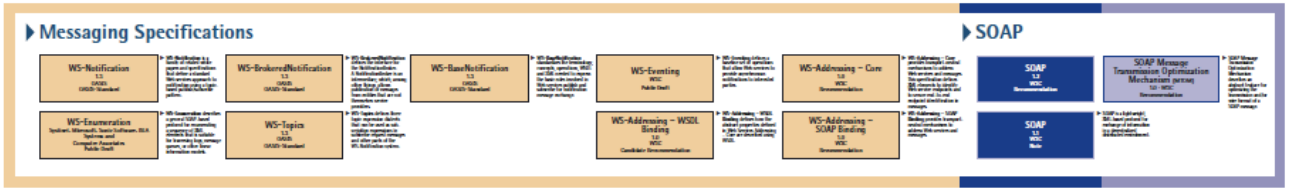


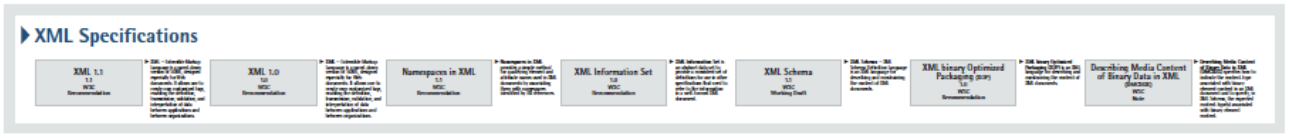


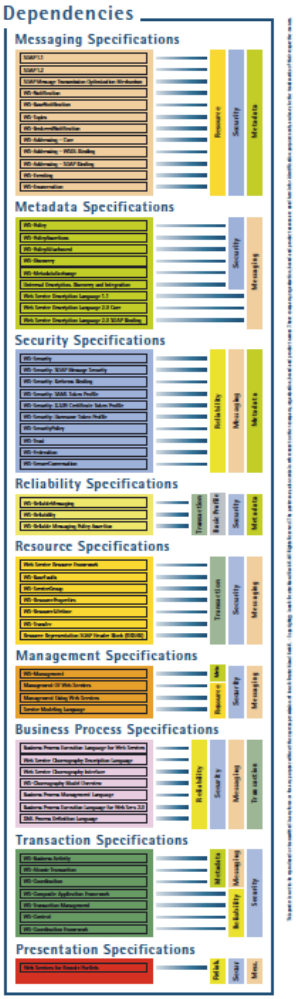












Version 3.0 - February 2007

innoQ Deutschland GmbH
Halskestraße 17
D-40880 Ratingen
Phone +49 2102 77162-100
info@innoq.com - www.innoq.com

innoQ Schweiz GmbH
Gewerbestrasse 11
CH-6330 Cham
Phone +41 41 743 0111

What's wrong with WS-* ?

- "We don't support WSDL-first development – you need to design your operations in C# first and export"

  Microsoft .Net Web Services Toolkit Support

  - Visual Studio/.Net currently doesn't support various WSDL Fault types or XML payloads
  - Jax-WS has problems with Canonicalization, which is required for WS-Security
  - And that's just the big 2
- WS-I "Basic" profile has 3 variants which are supported inconsistently by toolkits
  - The other WSI profiles are only implemented by a handful of toolkits
- In short – Interoperability is a farce
- So how can the pundits claim WS is the only viable distributed solution?

Whose Fault is the WS Mess?

- The vendors?
  - They do tend to propose overlapping WS standards to do the same thing
  - They implement convenient subsets of the standards
    - to deliberately thwart interop?
- The standards bodies?
  - W3C is notorious for slow process that leaves standards at “recommendation” status for too long
  - OASIS has little editorial or architectural quality control
  - WS-I has failed to fix the ambiguities that W3C and OASIS have allowed to pass
- In the end the contributors (mostly vendors) write, submit and vote on the standards
- We are seeing “Browser Wars” develop into “Service Wars”
  - At least HTTP & TLS work ☺

WS Success Stories

- Reardon Commerce (Axiom Travel Service)
  - SOA platform with user profiles & WS integration
- RightNow (Enterprise CRM)
  - WS in the background
- Salesforce.com (Smaller Business CRM)
  - WS in the background
- Amazon.com
  - Partner integration using WS
- Not Google
  - They use HTTP & proprietary messages and platform toolkits
  - Indicator that WS is not ready for prime time

What do the “successes” have in common?

- “Walled Garden”
  - Control over who uses what and how
- Minimal use of WS*
  - WSDL only (maybe WS Addressing)
  - Transport level security rather than WS-Security
- Controlled interface types
  - Simple payloads - no complex XML
  - Restricted subset of WSDL
- Integration behind a slick user interface
- Hosted platform in bespoke environment
  - User Profiles
  - Events/Notifications
  - Charging and Payment

What's wrong with BPM?

- The execution semantics of the popular BPM languages is incompatible:
  - BPMN is a pretty diagram format with no formal semantics
    - XPDL can represent BPMN, but with no more formality
  - UML Activity Graphs have a novel token passing semantics (with "semantic variations")
  - BPEL is based on Pi-Calculus, but with no formal mapping and no graphical syntax
  - YAWL is formally based on Petri Nets, but only has open source implementation
- Therefore, no semantic mappings are possible between the languages

What about QoS?

- Do we mean QoS of the service provision OR QoS of the service application?
  - Execution time – is this end-to-end, or just at the server? Average, or Max? Measured by whom?
  - Price – is this access price (search the books), or price of service (buy a book). Is Price even a QoS?
  - Reputation & Trust – who rates this? Who stores it? Is the number of reviews, etc., revealed?
- Who stores the QoS properties, and in what format?

Some practical initiatives to overcome challenges

- Service Description metamodels
  - Covers many of the questions asked about QoS
  - http://service-description.com/
- SDLs to raise the level of abstraction
  - Maturity of MDA, MDE by discarding the hype
  - Allows Code generation OR Runtimes from Software Factories
- KISS (Knowledge Industry Survival Strategy) Initiative
  - Modelling Tool Interoperability Manifesto & Projects
  - Workshop Series at major conferences
  - http://www.industrialized-software.org/kiss-initiative

To Wrap Up

- Smart Services will combine
  - Web Services (where viable)
  - BPM (several variants)
  - QoS and other service properties
  - Metering and Billing
  - Service Discovery, Substitution and Variability
- Standards have hit an all time quality low
  - Interoperability is compromised
  - We will need lots of duct tape & ticky-tacky
  - Model interoperability gives us a chance to abstract away from the ugly realities
  - Contributions to Open Source allow us to provide reference implementations and bottom-up interop