

Building real-world J2EE applications that fit into SOAs

Bert Ertman

IT Architect

Java Competence Center

Info Support

- **Joined Info Support in 1998.**
- **Involved in several Java/J2EE projects for large Dutch companies as a developer/designer/architect.**
- **Founder of Info Support Java Course Curriculum.**
- **Founder of Info Support Java Competence Center.**
- **Certified Microsoft Systems Engineer, IBM Websphere Specialist and Sun Java Enterprise Architect.**
- **Frequent speaker on Java/J2EE & Software Architecture related topics.**
- **Speaker at Microsoft Enterprise Seminars on J2EE and .NET interoperability.**
- **Author for Java Magazine and Software Release Magazine.**
- **Inventor of Wheel (often).**



- This is not a web services tutorial!
- I assume you have:
 - General understanding of the concepts of an SOA
 - General understanding of the Java/J2EE platform and preferably working experience building J2EE applications
 - General understanding of how a web service operates

- Introduction
- Setting the right perspective on SOA
- What makes and brakes an SOA
 - Anti-patterns
 - Best Practices
 - Granularity
 - Interoperability
 - Performance
- Summary
- Q&A

A word cloud of various IT and business acronyms is displayed against a background of rolling green hills under a blue sky with white clouds. The acronyms are scattered across the image, with some appearing larger than others. The acronyms include: HTTP, JAXP, B2B, QoS, WS-*, PTP, JAXM, SAAJ, RPC, EAI, XML, WS-I, UDDI, XSD, OASIS, MTOM, URL, SOAP, SSL, BPEL, JAXR, WSDL, EDI, OSS, W3C, POJO, JCP, IIOP, SOA, CICS, SEI, SMTP, WSCI, DIME, EJB, API, RMI, JBF, PKI, ebXML, JMS, RTFM, ws-security, J2EE, URI, BTP, ESB, JSR, MQ, JCA, FTP, JAX-RPC, SOAD, JAXB.

SOA = 10% IT, 40% Business, 50% Communication

- SOA is different things to different people:
 - Business Analyst: business alignment, on demand, etc.
 - Architect: principles, patterns...it's an architectural style!
 - Developer: programming model, tools, methodologies and technology. (not just plain vanilla web services!)
- Developers don't implement an SOA!
 - Architects and Business Analysts do that, so don't pretend you know more about the business than the business itself!
 - But, developers can really help out by realizing layering in their applications, choosing open standards, etc.

- J2EE is an architecture at the technical (application) level while SOA is an architecture at the logical level
 - SOA is not a replacement for J2EE
 - SOA is not a part of J2EE
 - J2EE should be made a part of your SOA!
- J2EE (1.4) is provisioned for building services-enabled applications
 - Web technology
 - Servlets, Java Server Pages, Java Server Faces
 - Component technology
 - Enterprise Java Beans
 - Web Services technology
 - JAXB, JAXM, JAXP, JAXR, JAX-RPC
 - J2EE Connector Architecture

- Are services always SOAP/HTTP?
 - No, a service can be anything described with an interface definition language such as WSDL.
- Different vendors = different opinions:
 - Microsoft is very HTTP focused concerning web services.
 - IBM: “If you can describe it in WSDL it’s a (web) service”.
 - However: very important to remain acronym compatible! ☺
- 60-70% of critical applications worldwide run on mainframes!
 - *IF* they are connected, it must be with something like MQ
 - MIPS costs for doing WS on a mainframe are much higher than proprietary implementations, think about it!

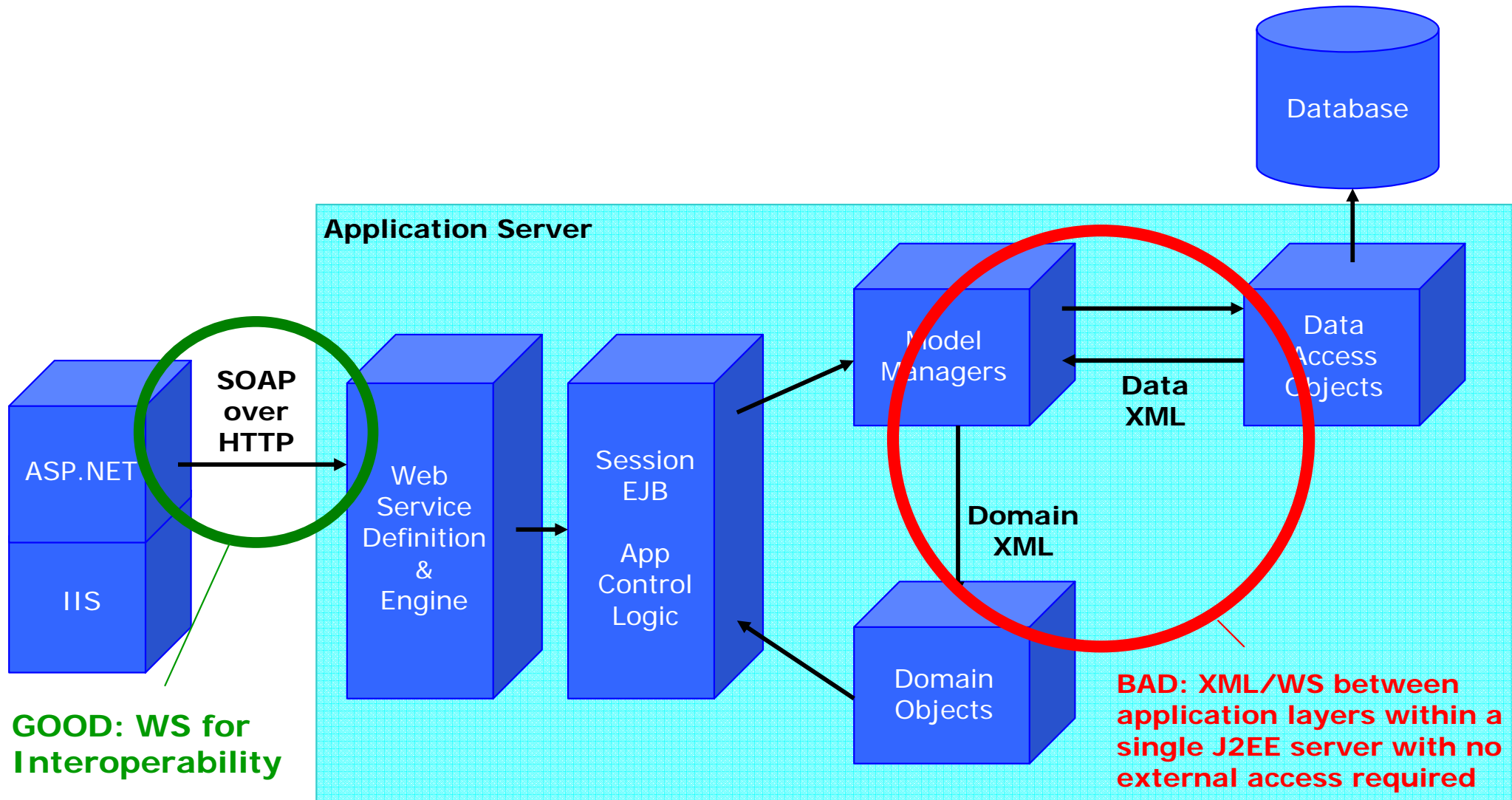
The background of the slide is a photograph of a vast, rolling green landscape under a bright blue sky filled with scattered white clouds. The hills are smooth and grassy, extending towards the horizon. The text is overlaid on the upper portion of the image.

There are NO greenfields in an enterprise!

- Over 80% of web services applications is built for integration purposes!
- Connecting mission critical backends using HTTP is not going to happen!

Web Services anti-patterns

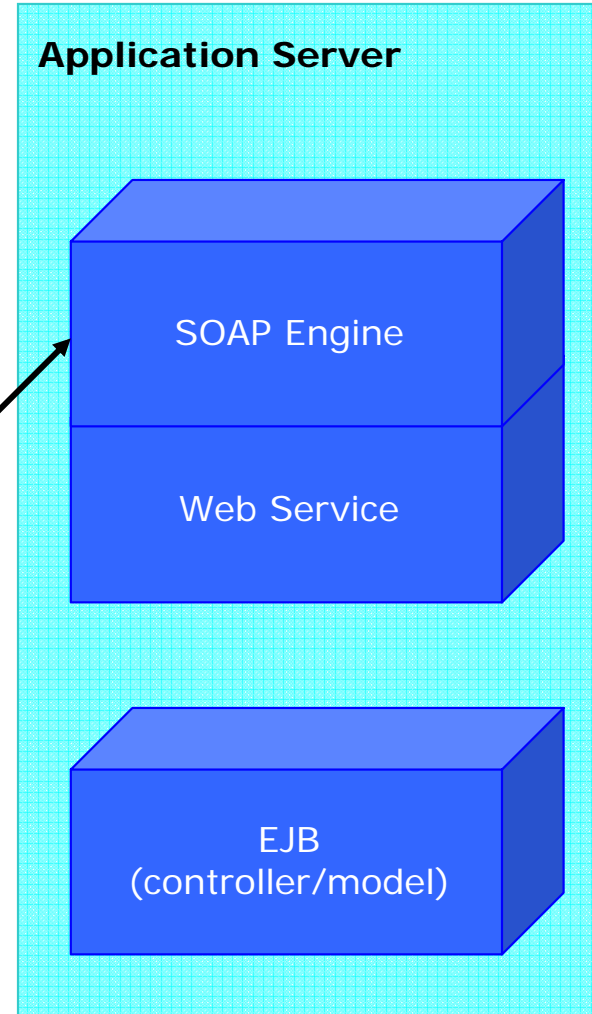
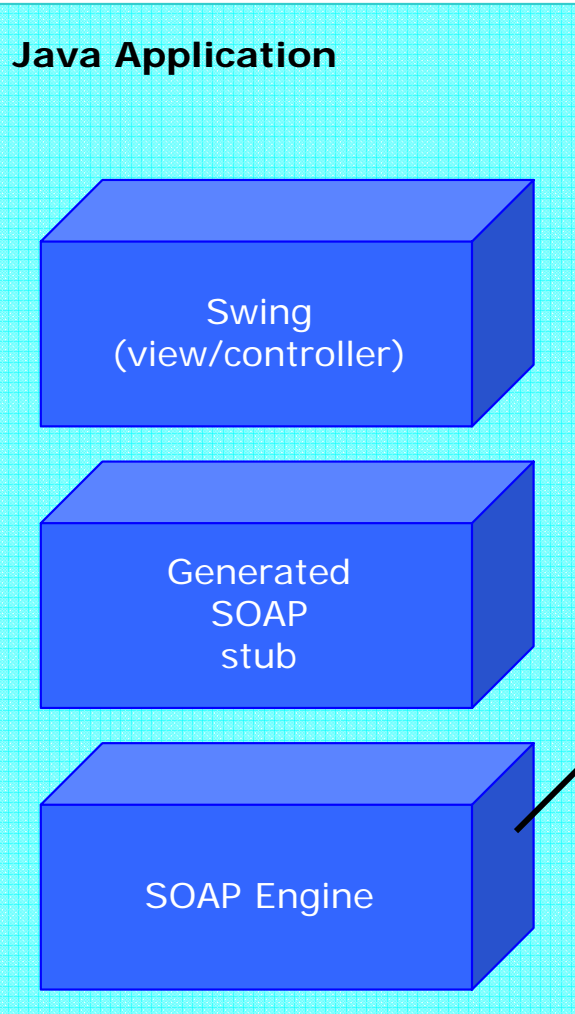
- Do not use SOAP/HTTP between layers of an application
 - 20:1 performance degradation
 - “EJB Mentality”
 - Don’t do Web Services for value objects
 - Use RMI-IIOP instead!
- Balance J2EE protocols based on requirements
 - Java – Java? (application) Use local calls or RMI-IIOP
 - Websphere – Weblogic? Use SOAP/HTTP
 - Can also be done by ORB-ORB (who wants to do that?) ☺
 - Websphere – Websphere? Use RMI-IIOP for best performance
 - Except if there is a firewall in between which does not allow RMI-IIOP...
- Good: Use Web Services for interoperability!



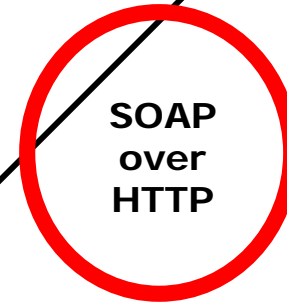
- Do not use fine-grained calls across distributed systems
 - Not specific to Web Services
 - EJB and CORBA analogies
- Good: Use Façade Pattern

Presentation Layer

Business Layer



Example calls:
 getFirstName
 getLastName
 getOfficePhone
 ...

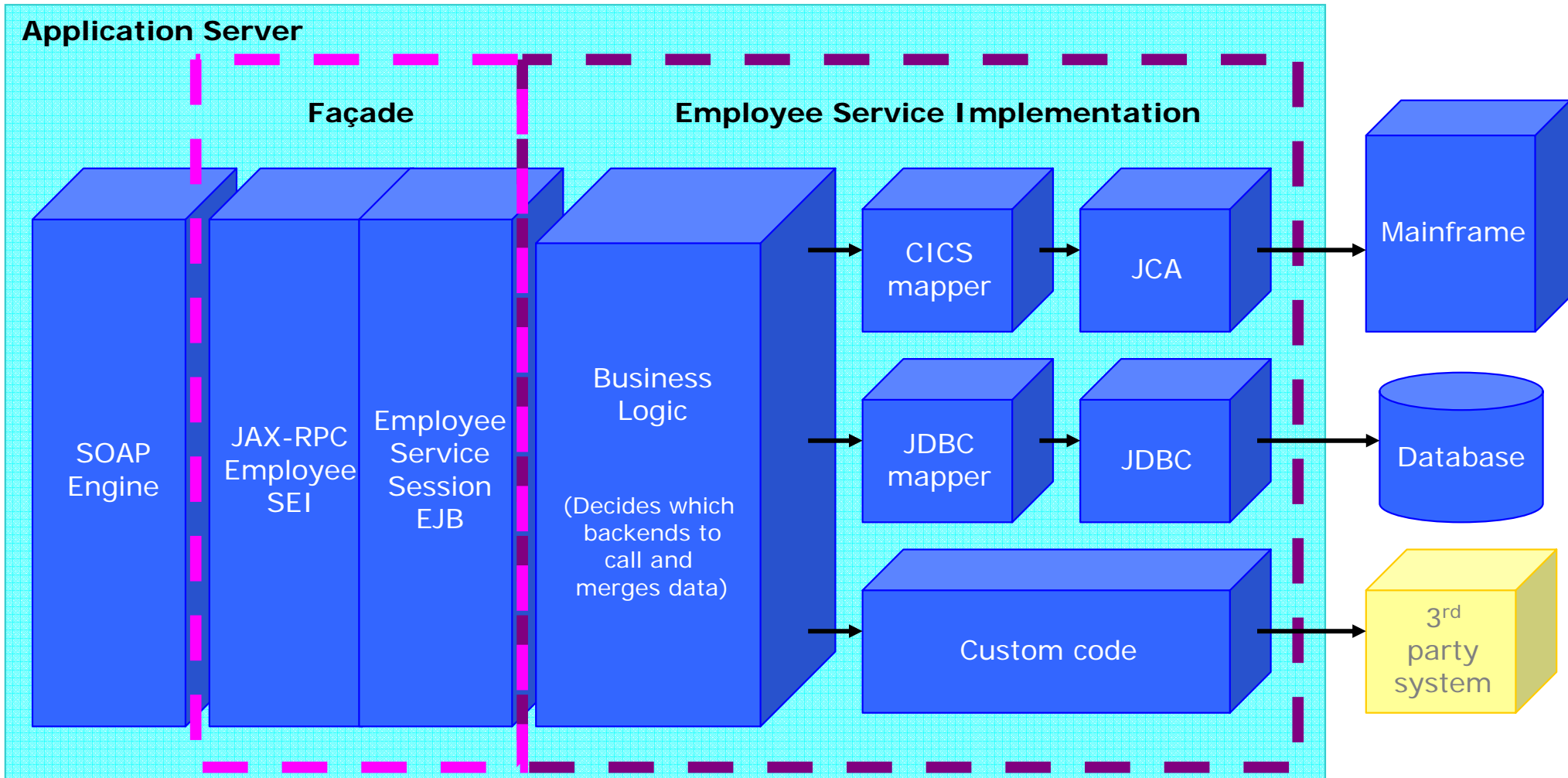


BAD: Using WS between Application Layers/Servers.

(no firewall issues, fine grained interaction)

GOOD: Do not expose (legacy) interfaces/transactions

Provide new abstraction (Façade Pattern)



Web Services best practices

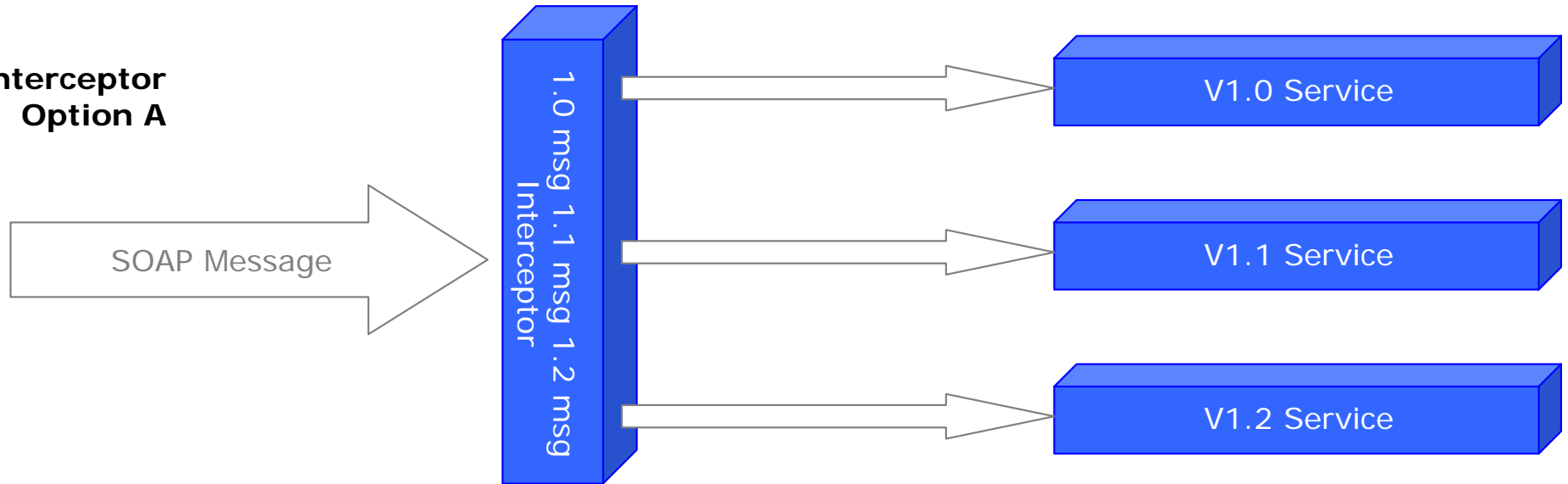
- Stateless Session Beans are great for implementing Web Services
 - Use EJB 2.1 Service Endpoint Interface (SEI)
 - Runtime will do everything for you
 - EJBs run within transactional and secure boundaries
 - Leverage existing logic with an SEI
 - Tool support available

- WS-I, whassup?
 - Basic interop is doing fine using BP 1.1
 - More advanced features like choices in XSD can still give you a hard time
 - Security is on the move
 - Only SSL in Basic Profile
 - Security Profile based on ws-security
 - QoS ws-* is currently only alphabet soup ☺
 - Soap with Attachments not interoperable (see also: next slide)
 - Use WS-I compliancy tools to test interoperability
- J2EE, whassup?
 - J2EE 1.4 is the first enterprise platform to be fully compatible with the WS-I Basic Profile.
 - Basic technology is there, but still no support for the alphabet soup!
 - Vendor-added technology might come in handy, but might also work against you! (Write once...run maybe?)

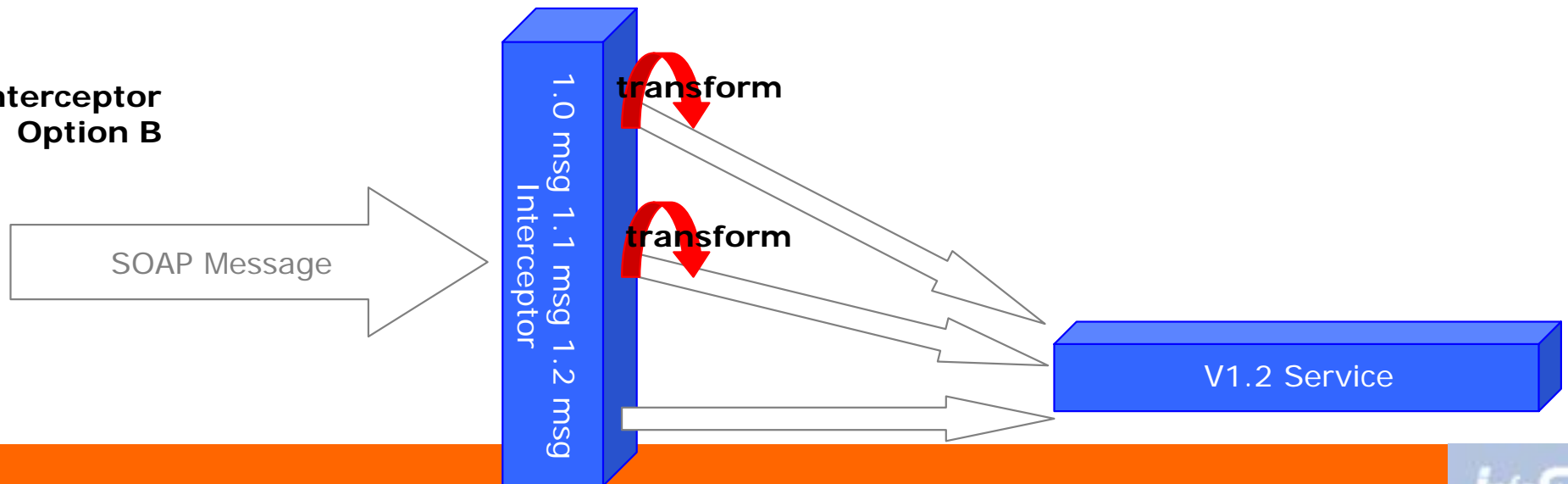
- Soap with Attachments (Attachments Profile 1.0)
 - Microsoft does not support Sw/A
 - They have proposed a new spec to W3C (MTOM/XOM)
 - WS-I will likely create an MTOM profile in the future
 - There is no obvious good choice today, it depends...
 - Solution: Don't use Sw/A (oh btw: DIME is DEAD!)
- UTF-16
 - .NET's WSDL tool is not able to handle UTF-16 encoded WSDL
 - Solution: Don't use UTF-16 encoded WSDL
- Null primitives
 - `<element name="x" type="xsd:int" nillable="true" />`
 - .NET maps this to a simple int, so it can't be null
 - JAX-RPC maps this to `java.lang.Integer`
 - Nillable complex types are fine, since they become C# objects which can be null
 - Solution: avoid nillable primitive types

- Short answer: There isn't any ☹
 - WSDL 1.1 specification doesn't address versioning
 - The only "version" you can safely create is a new namespace for a new web service
- But there are still some games you can play
 - Backwards compatibility
 - Try not to break anything
 - `xsd:anyType` for input/output message
 - This removes all type info!
 - Bigger burden on application
 - Interceptor (see next slide)

Interceptor Option A



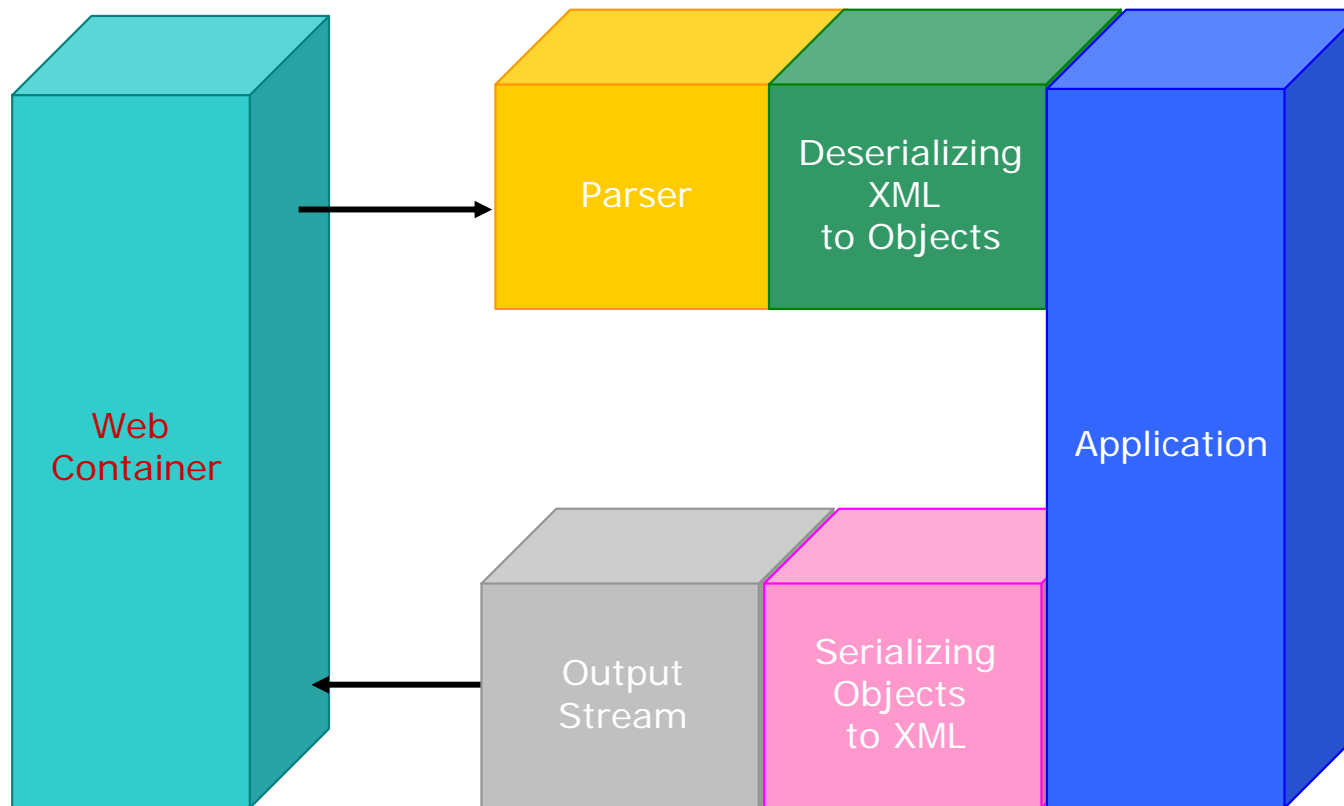
Interceptor Option B



- WS interfaces are “remote interfaces” by definition!
- Use TCP-level monitoring tools when testing web services
- Payloads, pay attention!

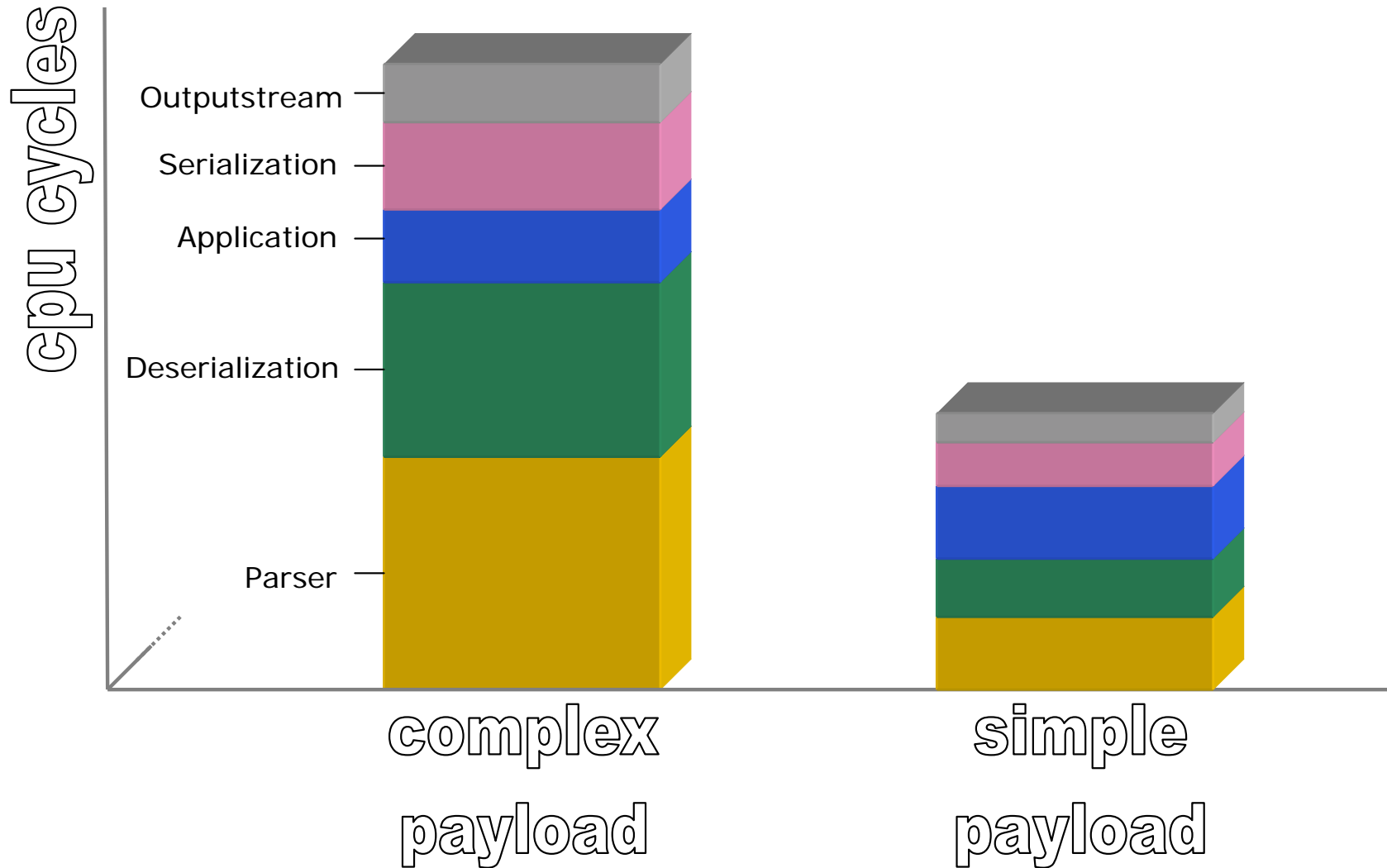
- Complex payload
 - Deeply nested structures
 - Complex XML structures like choices, etc.
 - Maximum stress on parser and object (de)serialization
- Simple payload
 - Almost no nesting of elements
 - Maximum stress on implementation behind web service
- Small payloads are usually faster
 - But: 1x 10kb vs. 10x 1kb ???
 - Beware of parsing and (de)serialization overhead!

SOAP Engine physics



Payload (1/3)

Size matters! Complexity matters as well!



- Architect your objects for transfer!
 - Remember: WS are a remoting mechanism!
 - Avoid deep nesting
 - No hashtables of hashtables
 - Try to avoid excessive object creation
 - Both a performance and memory (GC) hit!

- Use Document/Literal instead of RPC/Encoding
 - Results in smaller payload
 - RPC/Encoding == payload bloat!
 - Use simple, yet descriptive XML element names
 - 200+ characters might not be such a good idea ☺
 - It all adds up to your total payload size
- Use custom (de)serializers (wrappers)
 - You know what objects you are dealing with
 - Default serializer uses reflection which is significantly slower!

- Determining the right level of granularity is hard!
 - Deliver “units of business value”
 - Service interface vs. API Interface
- Expose common legacy interfaces
 - Provide new protocol access and/or interface to legacy systems
 - Reuse existing CICS or ACMS transaction granularity
 - Transition scenario towards replacement?
- Think forward! Think Choreography!
 - Hardcode workflow today and BPEL tomorrow?
 - Vendors have tools for this (mostly proprietary)

- By post

1. Client requests application form
2. Provider sends it
3. Client fills it in and returns it
4. Provider says “yes” or “no”

Service-like – the interaction is not dependent on the “identity” of the service provider. The completed application form can be returned to a different branch office than the one that provided it.

API-like – the interaction is dependent on the specific representative of the estate agent i.e. the “identity” of the service provider.

- By phone

1. Client calls provider
2. Provider says “hello, how can I help?”
3. “I’d like a mortgage please”
4. “What’s your name?”
5. “John Doe”
6. “What’s the property address?”
7. “333 O’Farrell Street...”
8. Etc.
9. Etc.
10. Etc.
11. Etc.
12. Etc.
13. “...Okay, your mortgage agreement number is 12345, I’ll post the rest of the details”

- It is impossible to do everything right the first time, but there is a huge difference between refactoring and rebuilding
 - There is no magic button!
 - Granularity and modularity (layering) choices are a critical success factor!
- Think interoperability
- Think performance
- Think!

- Perspectives on Web Services, Zimmermann et al.
 - ISBN: 3-540-00914-0
- Implementing an SOA using an Enterprise Service Bus
 - IBM Redbook [sg246346]
- Turn EJB components into Web Services
 - <http://www.javaworld.com/javaworld/jw-08-2004/jw-0802-ejbws.html>

- My Blog
 - <http://blogs.infosupport.com/berte>
- My E-mail
 - berte@infosupport.com
- Info Support
 - <http://www.infosupport.com>

- Visit our booth, and you might win this...

