INTERSYSTEMS®

Configuring an Enterprise Service Bus

Mike Moulckers
Ray Wright

Configuring an Enterprise Service Bus

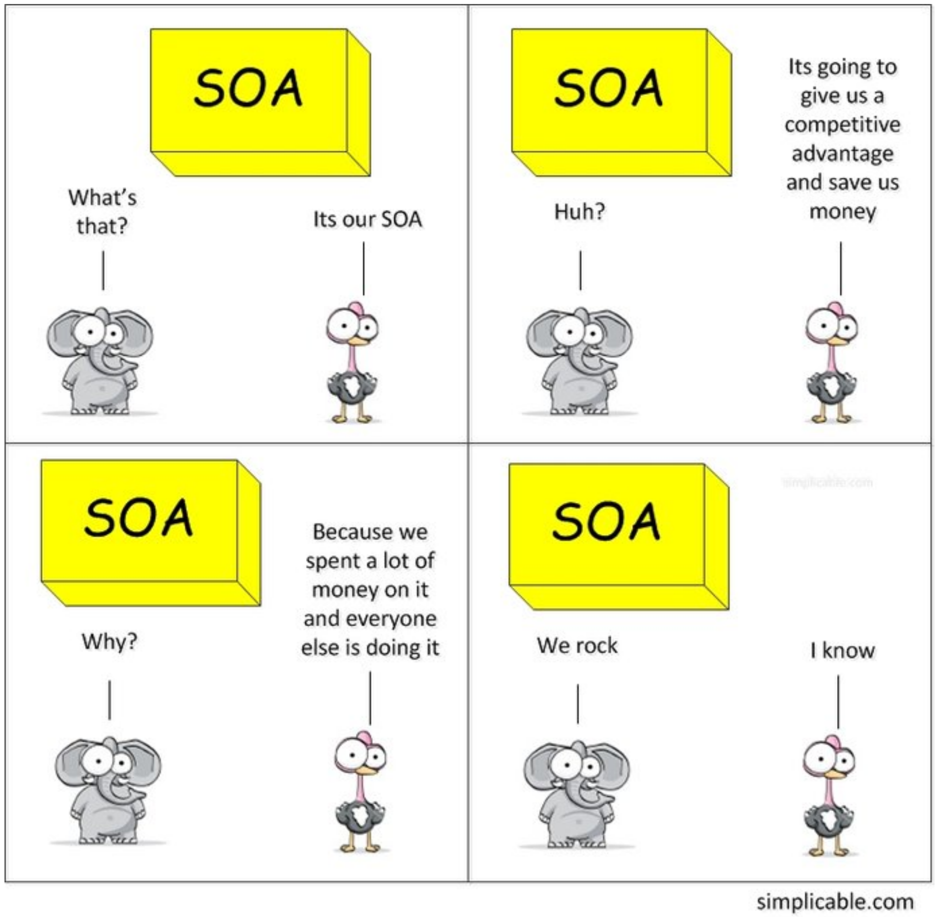
- Monday, 1:30 PM – 3:30 PM, Plaza Ballroom K
- Tuesday, 1:00 PM – 3:00 PM, Plaza Ballroom K
- Track: Data Platforms
- Technical level: Intermediate
- Presenters: Mike Moulckers, Ray Wright
- Establishing an ESB (Enterprise Service Bus) enables you to orchestrate calls to data and services from across your enterprise and expose them for easy consumption by a variety of user platforms including mobile devices. This hands-on academy will teach you how to use Ensemble as your ESB, with examples of RESTful and SOAP services. NOTE: the breakout session "Routing REST/SOAP Through Your ESB" complements this hands on academy.
- Prerequisite: Familiarity with Ensemble.

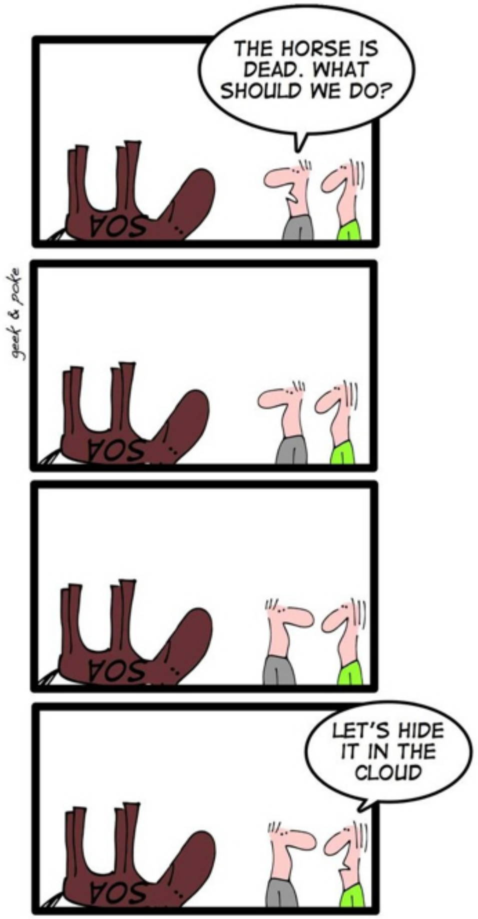
INTERSYSTEMS®

Agenda

- What is a Service Oriented Architecture (SOA)?
- What is an Enterprise Service Bus (ESB)?
- Ensemble as an ESB
- Ensemble Passthrough Services
- The Academy Scenario
- Exercises 1-7
- Q&A

What is a Service Oriented Architecture (SOA)?


SOA
What's that?
Its our SOA
SOA
Huh?
Its going to give us a competitive advantage and save us money
SOA
Why?
Because we spent a lot of money on it and everyone else is doing it
SOA
We rock
I know
simplicable.com


THE HORSE IS DEAD. WHAT SHOULD WE DO?
SOA
geek & poke
LET'S HIDE IT IN THE CLOUD

INTERSYSTEMS®

What is a Service Oriented Architecture (SOA)?

- Service-oriented architecture (SOA) is a design pattern based on distinct pieces of software providing application functionality as services to other applications via a protocol. This is known as service-orientation. It is independent of any vendor, product or technology.

  – *Wikipedia*

- A loosely-coupled architecture designed to meet the business needs of the organization.

  – *MSDN*

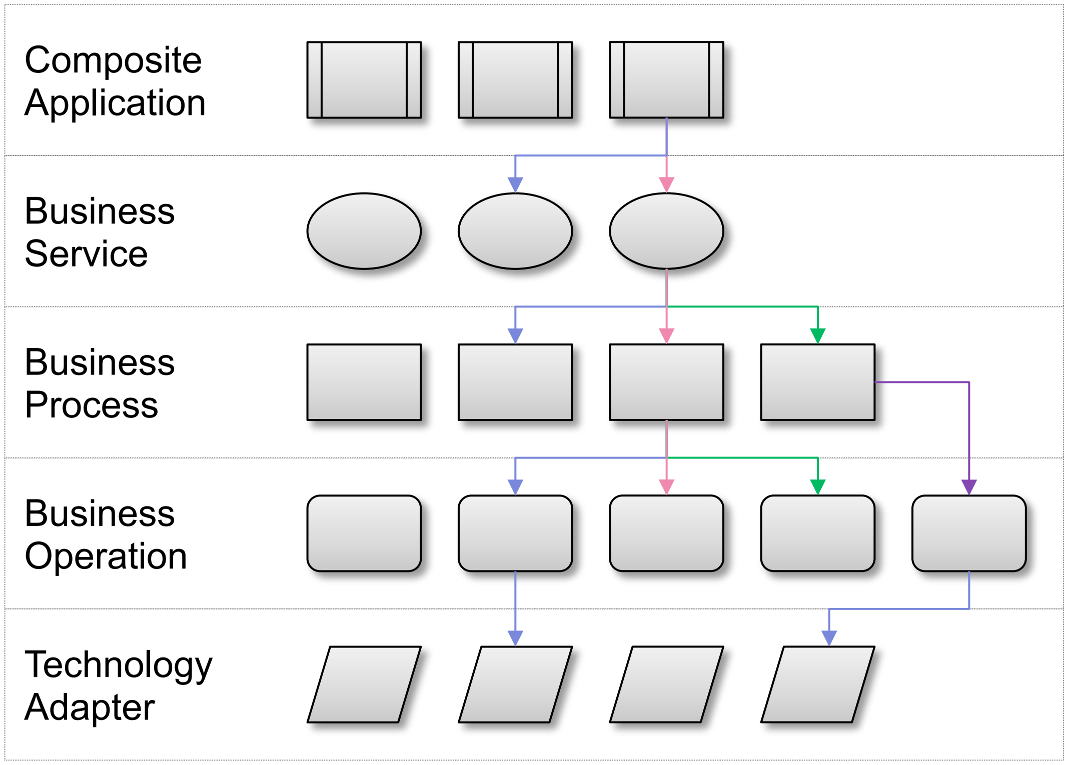
- SOAs are like snowflakes – no two are alike.

  – *David Linthicum, Consultant*

Goals of an SOA

- Agreement to…
  - Develop modular software elements
  - Define the meaning of elements and interfaces
  - Make the elements available – directory/registry, bus
- Leads to…
  - Reuse
    - Lower development costs
    - Reduced defects
    - More consistent processes
  - Agility
    - Assembly vs development
    - Increased ability to change (flexibility)
    - Service improvements

SOA Class Hierarchy



What is an Enterprise Service Bus (ESB)?

- A term originally coined by Gartner and Sonic Software

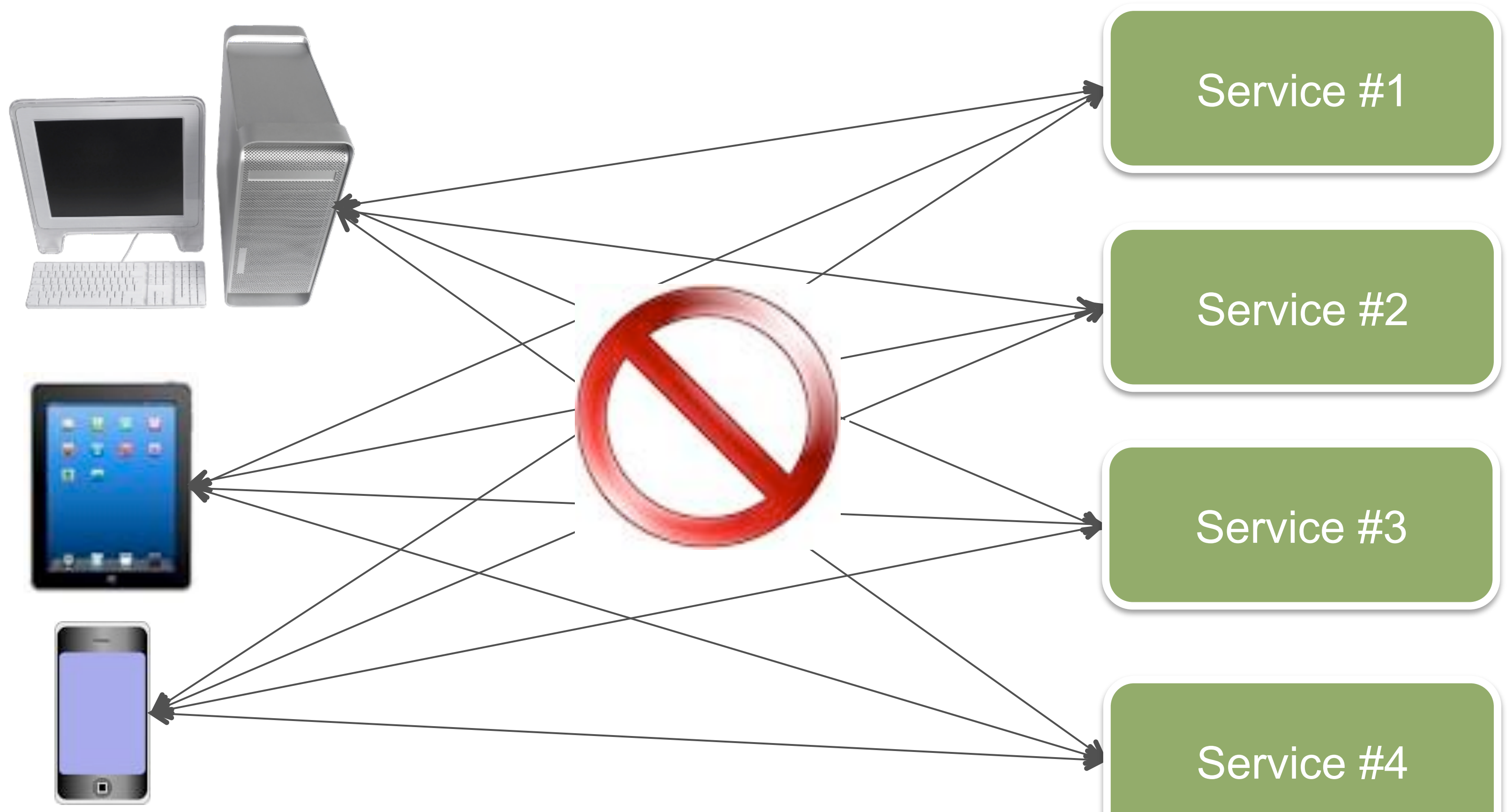
- An enterprise service bus (ESB) is a software architecture model used for designing and implementing communication between mutually interacting software applications in a service-oriented architecture (SOA).

  As a software architectural model for distributed computing it is a specialty variant of the more general client server model and promotes agility and flexibility with regard to communication between applications.

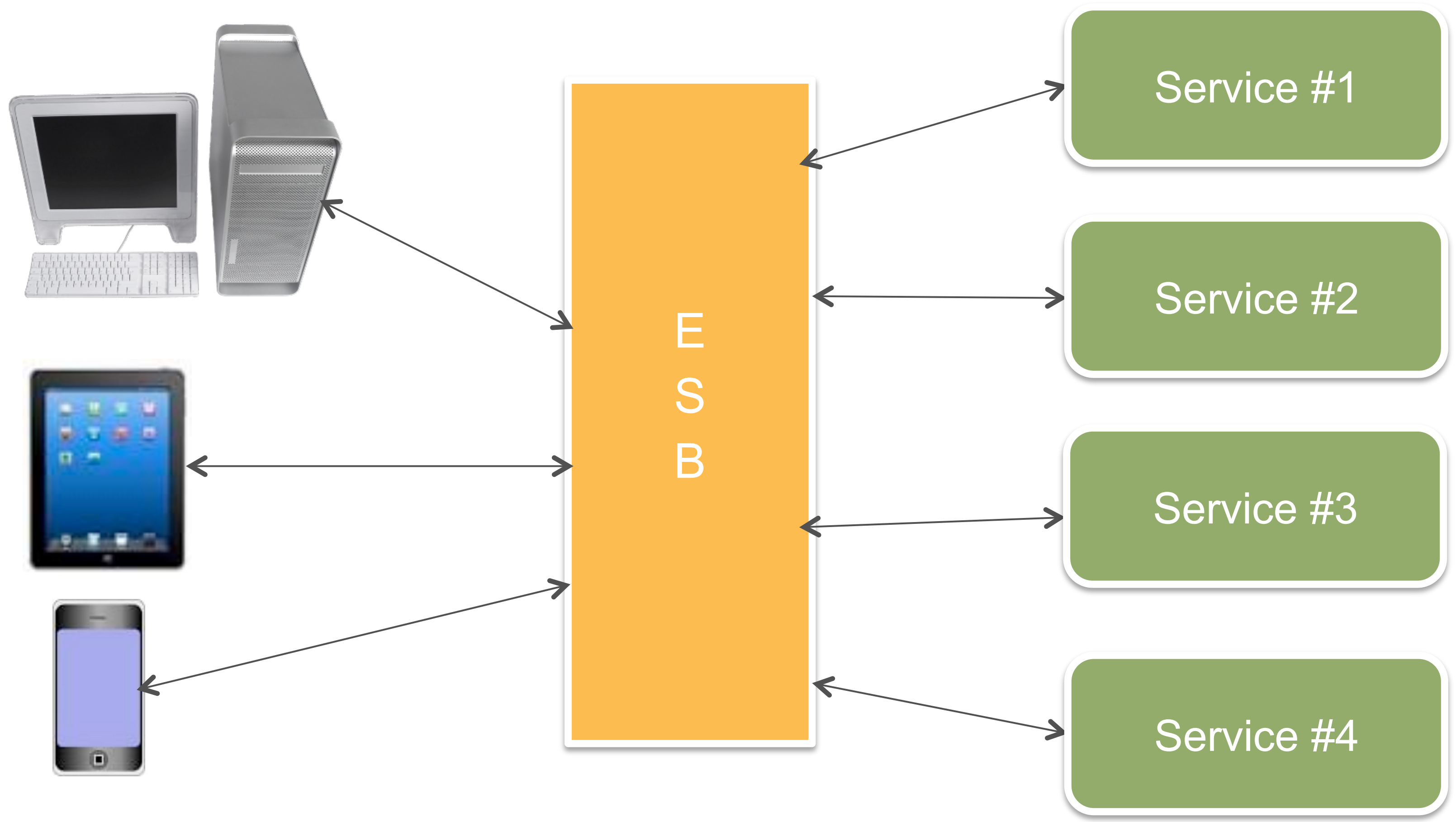
  Its primary use is in enterprise application integration (EAI) of heterogeneous and complex landscapes.

  – *Wikipedia*

Service Oriented Architecture without an ESB



Service Oriented Architecture with an ESB



The ESB as a Business Transformation Engine – Why ?

- Put the communication bus in place and enable the applications to talk to the bus – not point-to-point integration
- Decouple applications from each other, allowing them to communicate without dependency on or knowledge of other systems on the bus
- Building applications for todays market may require a change of business model :
  - Unified solutions instead of application portfolios
  - Delivery on the device of choice, from desktop to mobile
  - Solutions focused on user experience
- Achieving this will require distinct development organizations to provide compatible software components which work together to provide powerful new solutions

The ESB as a Business Transformation Engine – How ?

- An ESB is key to making this work, and becomes an engine for business transformation when it can :
  - Unify a portfolio of applications into a cohesive, mobile-enabled suite
  - Empower users with information, analysis and insight at the point of action, and enable them to drive business processes based on that insight
  - Reduce the customers deployment time and costs by rapidly delivering applications that are interoperable, easy to configure, and scalable

Key Concepts

- Smart decisions
- Back end process orchestration
- Mobile application support
- Operational insights
- Service configuration
- Interoperability
- Governance
- Reliability
- Scalability
- Future development

Key Concepts – Smart Decisions

- Message passing is not enough – ESB needs to add intelligence, based on rule-based decision making and action based on message content and metadata
- Intuitive rules engine that can be extended to running applications without additional coding
- Event processing and alerting for immediate reaction to significant changes
- Real time analysis of structured and unstructured data to feed the rules engine and business processes, and to inform users in audited workflows

Key Concepts – Back End Process Orchestration

- Rich graphical editor for diagramming process and information flows
- Automatic generation of code from business process diagrams, and of diagrams from code
- Support for long running business processes, including human workflow
- Workflow engine to distribute and move tasks among users while incorporating their decisions automatically into the business process

Key Concepts – Mobile Application Support

- Presentation layer the enables a simplified end-user experience of complex back-end services
- API led approach to identifying what need to be exposed to the ESB, and using the ESB to orchestrate calls to the enterprise functions required to drive the mobile user interface
- Mobile technology tightly integrated with the ESB :
  - Rapid development capability
  - Utilize the ESB's business process management features to serve as a bridge between mobile devices and enterprise applications, data and analytics
  - The ability to access, transform and combine information from back-end applications to provide composite services for consumption by all types of device, from desktop to mobile

Key Concepts – Operational Insights

- Ability to derive insights from information and act quickly on them is a competitive advantage
- Real-time analytics that can be embedded in application workflows to help users make better business decisions
- Analytics technologies that work with both structured and unstructured data
- Business activity monitoring based on user-specified thresholds and events that tracks system performance and raises alerts

Key Concepts – Service Configuration

- Intuitive configuration allied to the ability to quickly create business logic to fill the gaps between available services and expected functionality
- Single, consistent development environment encompassing service creation, business process orchestration, business rule definition, data transformation, workflow, event processing and dashboard creation
- Service enablement of existing applications without writing code
- Service exposure and consumption without writing code

Key Concepts – Interoperability

- Easily integrate into existing systems for bi-directional information sharing
- Support for a broad range of standards. REST and SOAP are essential but to fully leverage capabilities and information in legacy sources the ESB should also support TCP/IP, SSL, FTP, SFTP, email, xDBC, CSV and custom interfaces
- Integration of applications and services built with different technologies, such as Java and .Net
- Embedded high-performance database to store all messages, protect against data loss and ensure the integrity of (long running) business processes
- Robust adapter framework that manages errors, retries, timeouts and recovery

Key Concepts – Governance

- Governance policy and process is key to the smooth running of an ESB and the services deployed through it
- Data security, service level agreements, version control, etc., supported by the ESB technology
- Robust access control built on role based authentication and authorization
- Audit and activity reporting
- Statistics gathering for managing SLAs
- Version management that supports moving between service versions without disrupting applications
- Message repository for proactive monitoring, message replay, exception identification, analysis and response

Key Concepts – Reliability

- Proven performance – demonstrable widespread use in simple and complex integration environments
- Guaranteed message delivery
- Minimal downtime

Key Concepts – Scalability

- Deliver the same rich functionality wherever the solution is deployed – on-site, hosted or in the cloud
- Provide the framework to build a robust, multi-tenanted solution
- Centralized management capability for all instances of the solution

Key Concepts – Future Development

- Meet todays needs – and tomorrows
- Platform supplier committed to product innovation and backward compatibility

Goals of an ESB

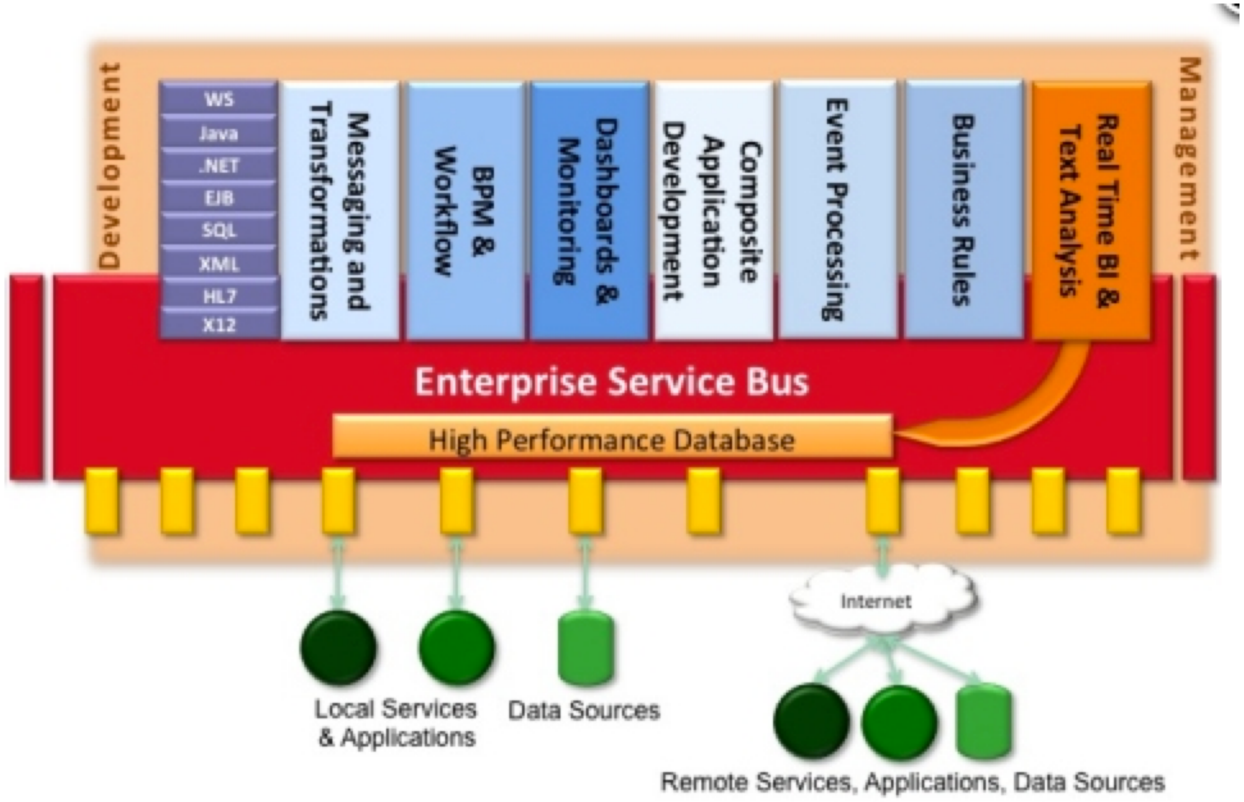
- Less design, more configuration
- Less message processing – more of a switchboard
- Configurable message patterns
- Lighter-weight SOAP and REST processing

Responsibilities of an ESB

- Messaging
- Buffering
- Transformation and Routing
- Mediation
- Orchestration and Composite Services
- Caching
- Service Discovery/Publication
- Lifecycle Management
- Governance
- Protocol Conversion
- Adapter Library
- Security
- High Availability
- Scalability
- …

Ensemble as an ESB

- Ensemble provides many ESB capabilities out of the box
- "Build your own" ESB production
  - Many partners have done/are doing this
  - We continue to add capabilities to make this easier


Development
WS
Java
.NET
EJB
SQL
XML
HL7
X12
Messaging and Transformations
BPM & Workflow
Dashboards & Monitoring
Composite Application Development
Event Processing
Business Rules
Real Time BI & Text Analysis
Management
Enterprise Service Bus
High Performance Database
Internet
Local Services & Applications
Data Sources
Remote Services, Applications, Data Sources

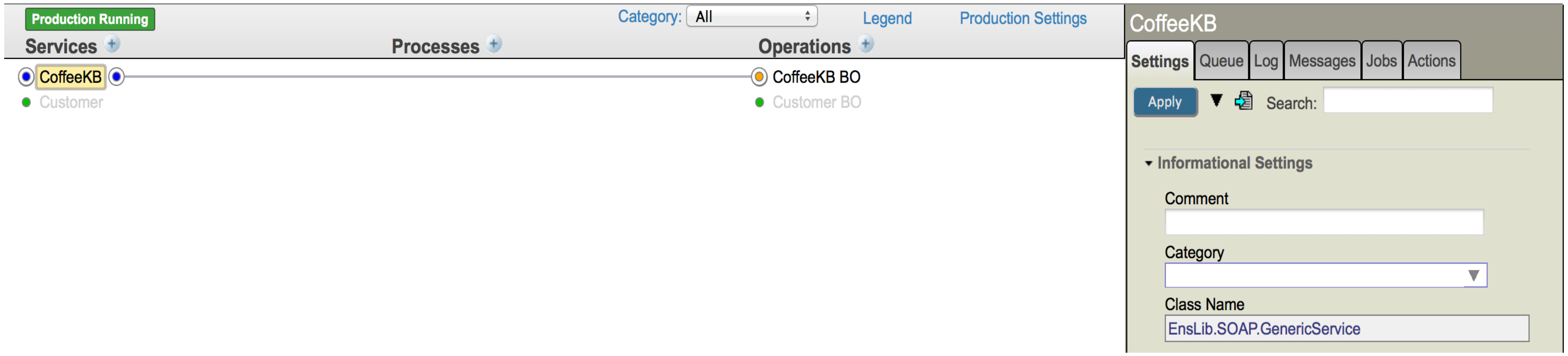
Ensemble Passthrough Services

- A service where Ensemble does not act as the client or the provider
- The File Passthrough service has been available for awhile
- The REST, SOAP and HTTP Passthrough services were introduced in 2014.1
  - Essential where the services are a combination of Ensemble and external third party services
  - Ensemble provides routing, transformation, auditing and other capabilities

Ensemble Passthrough Services

- EnsLib.REST.GenericService
- EnsLib.SOAP.GenericService
- EnsLib.HTTP.GenericService
- EnsLib.REST.SAMLGenericService
- EnsLib.SOAP.SAMLGenericService

- The first three Services listed have corollary Operations



Comment

Category

Class Name: EnsLib.SOAP.GenericService

Ensemble Service Registry

- Official release in 2015.2
- Provided an "experimental" implementation in 2014.1

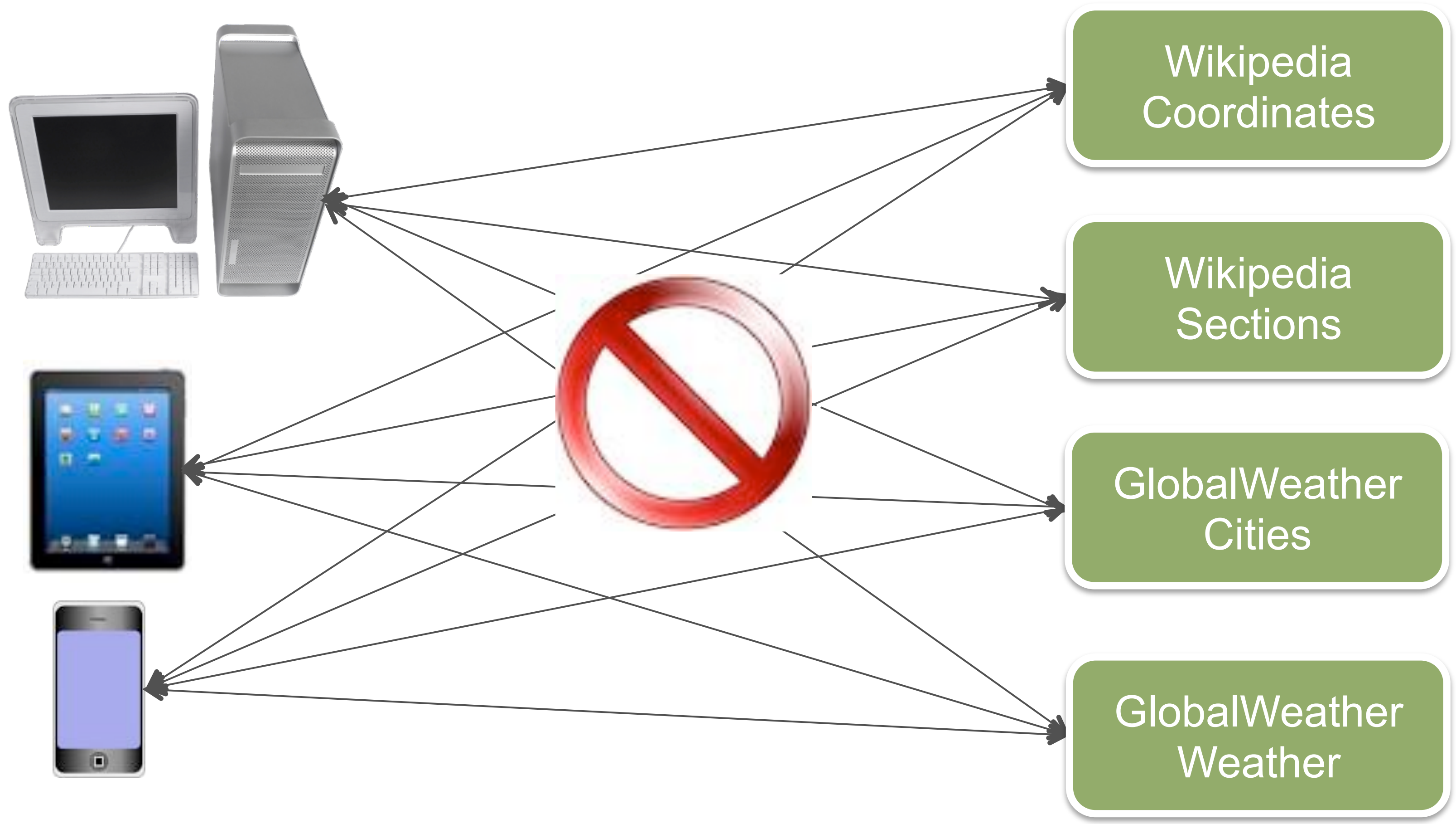
- Internal Registry: makes configuration easier
- Public Registry: makes services discoverable

- Supported Service Types:
  - Managed
    - Service requests are validated and routed through the bus
  - Independent
    - Service requests may be invoked through the bus or directly
  - Core
    - Service requests are provided by the bus
  - Federated
    - Passthrough proxy services exposed by Ensemble and primarily routed to an external service implementation

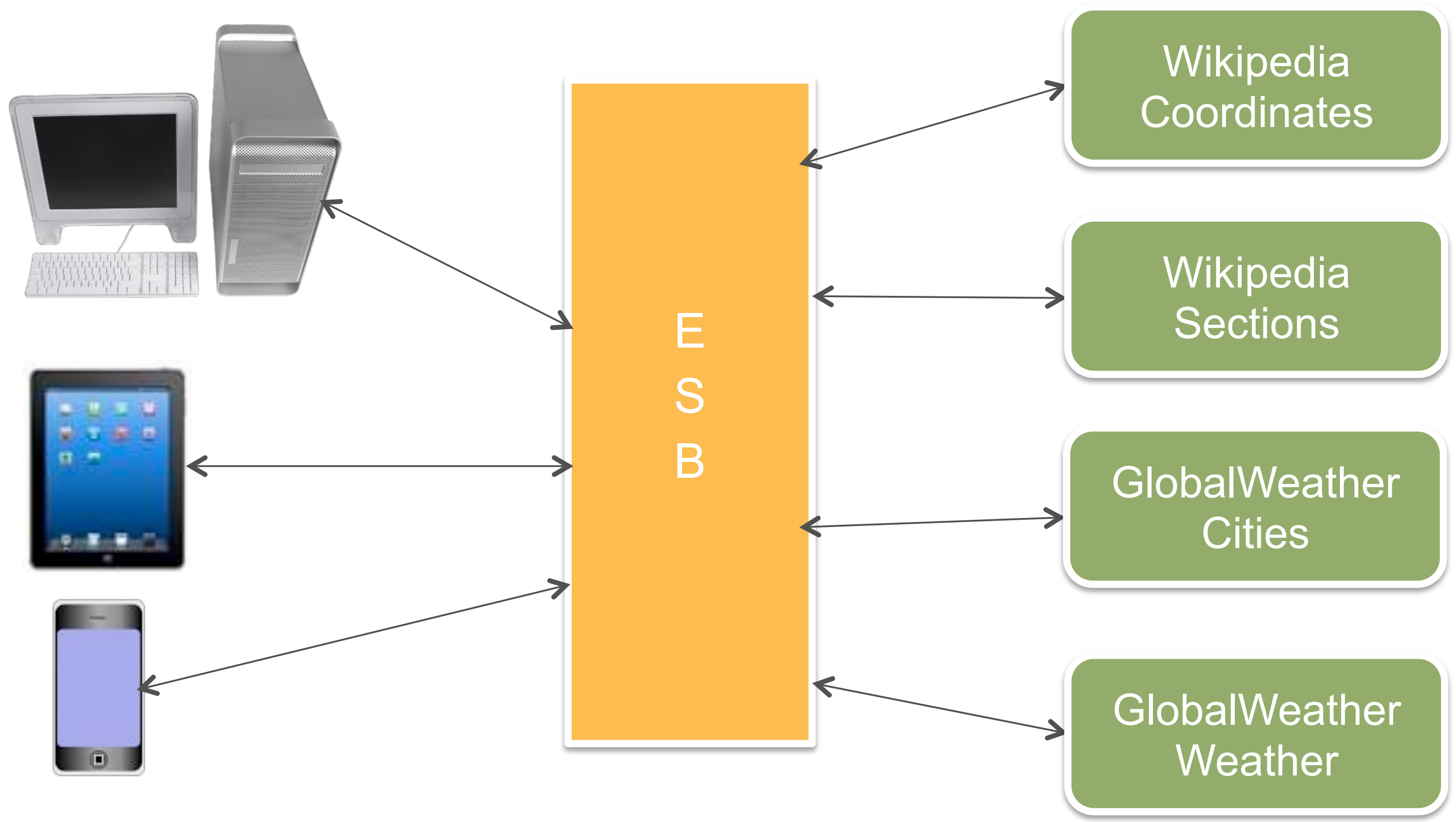
Academy Scenario

- We will be creating an ESB production that manages external REST and SOAP web services.
- We will be access these external web services via an external web service client (SoapUI), both directly and via the ESB production.
- We will examine what happens to custom HTTP and SOAP headers as they pass through the ESB "Generic" services and operations.
- We will validate an incoming SAML token, halting any requests that don't pass this validation.

Service Oriented Architecture without an ESB

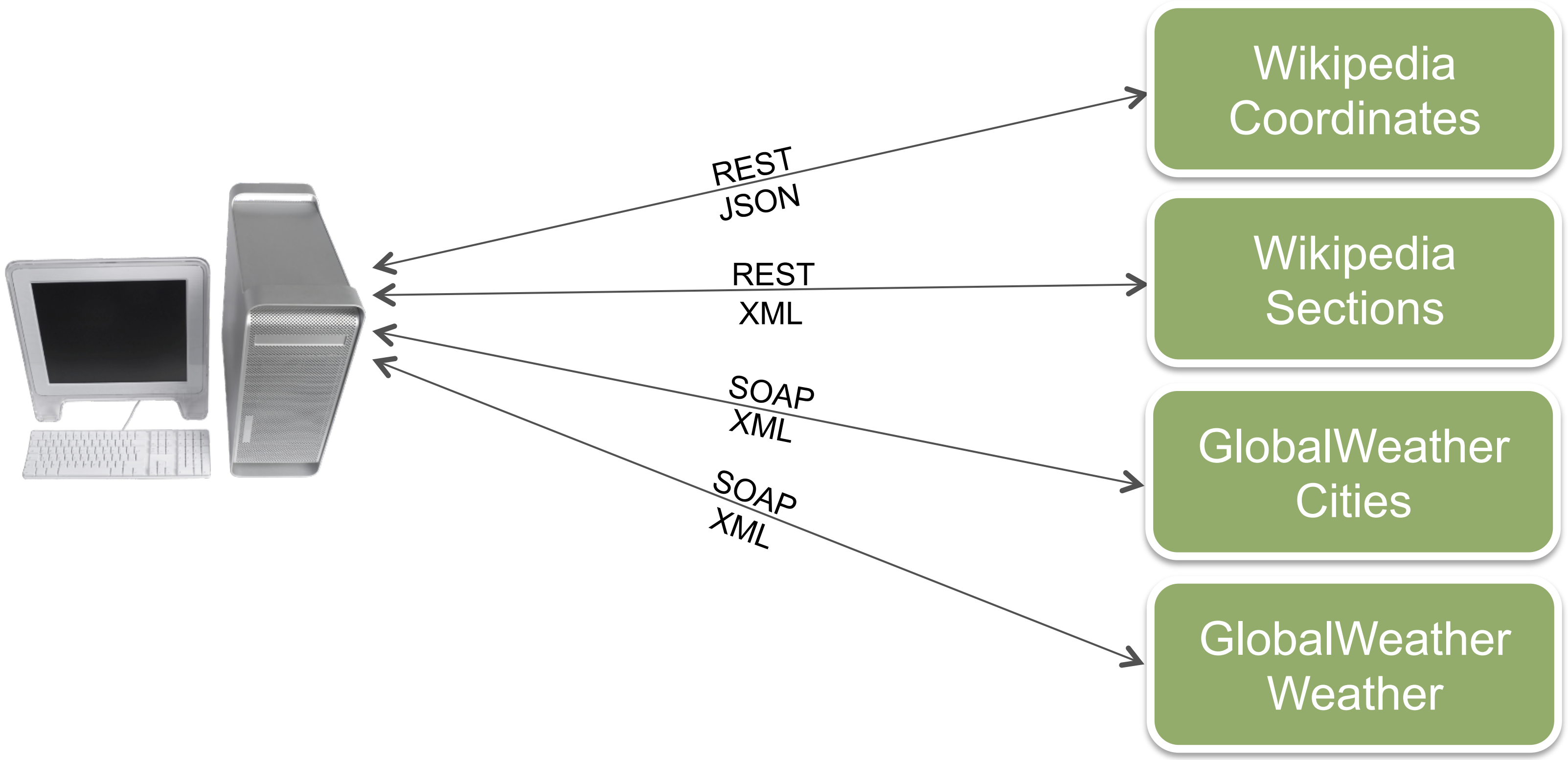


Service Oriented Architecture with an ESB



Exercise #1

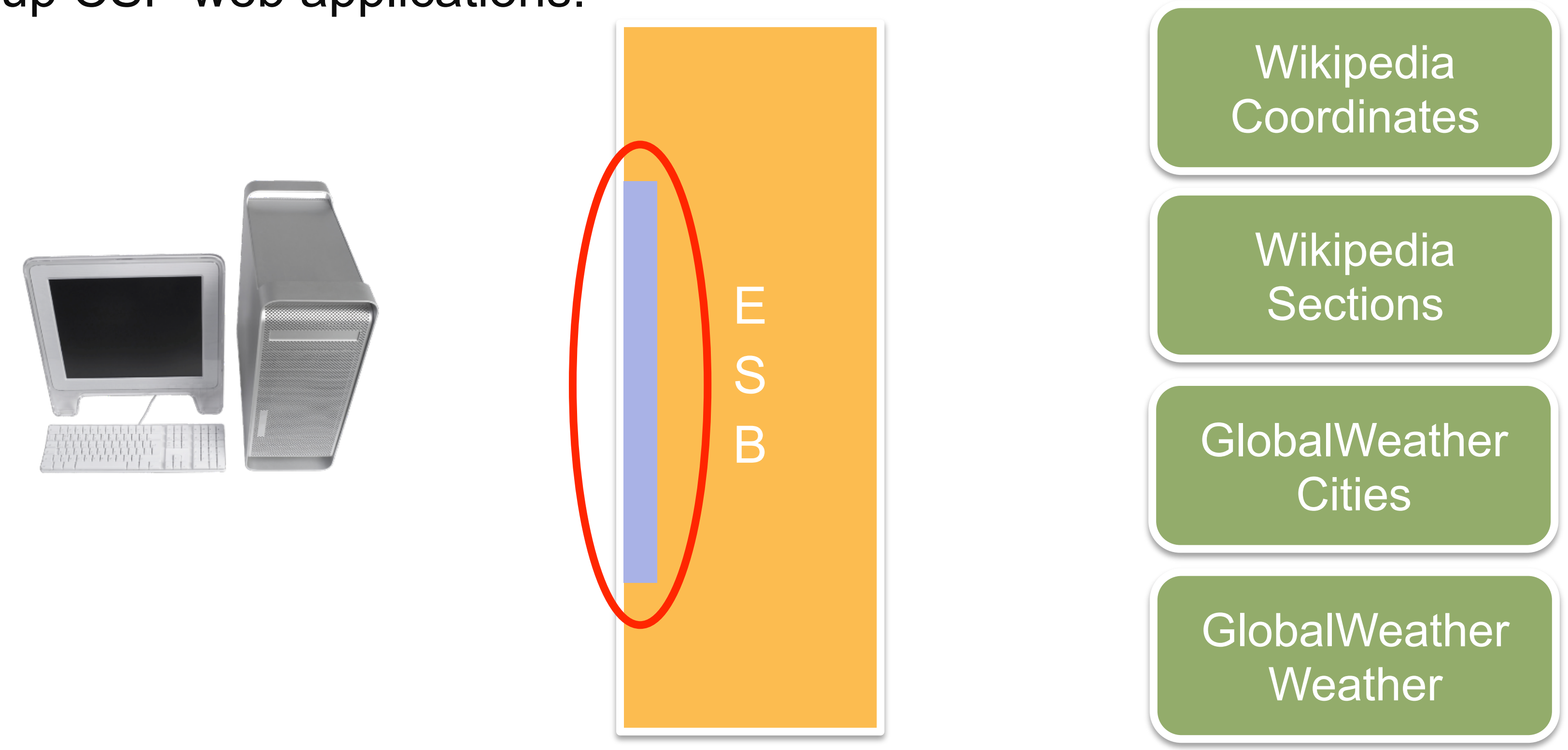
- Call External Web Services with External Web Service Client
  Call external REST and SOAP web services with SoapUI.



Exercise #2

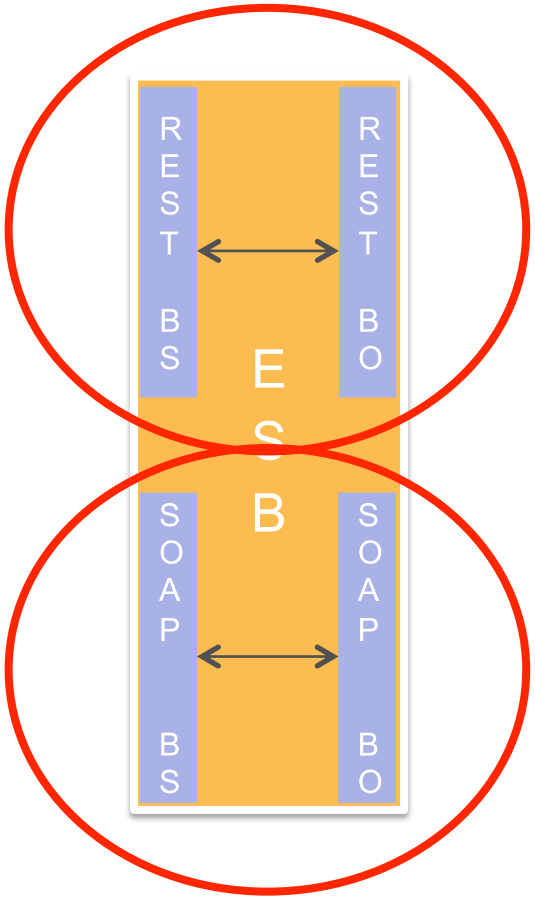
- Configure Ensemble Web Applications for Managing External Web Services

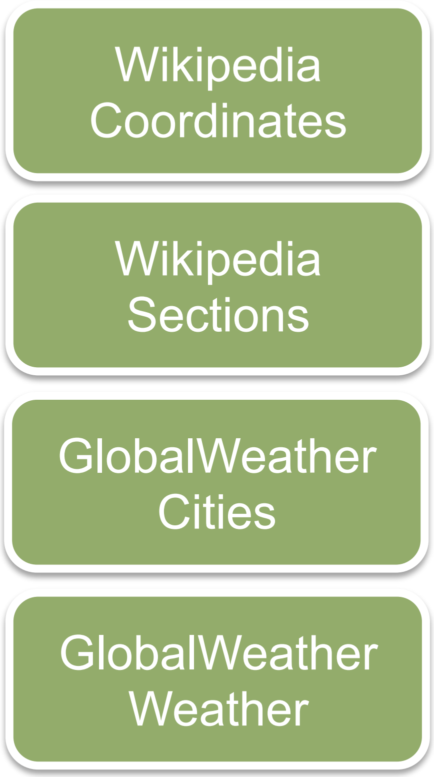
  Prepare Ensemble to handle incoming service requests by setting up CSP web applications.



Exercise #3

- Create and Configure an Ensemble ESB Production
  Create and configure REST and SOAP "passthrough" services.

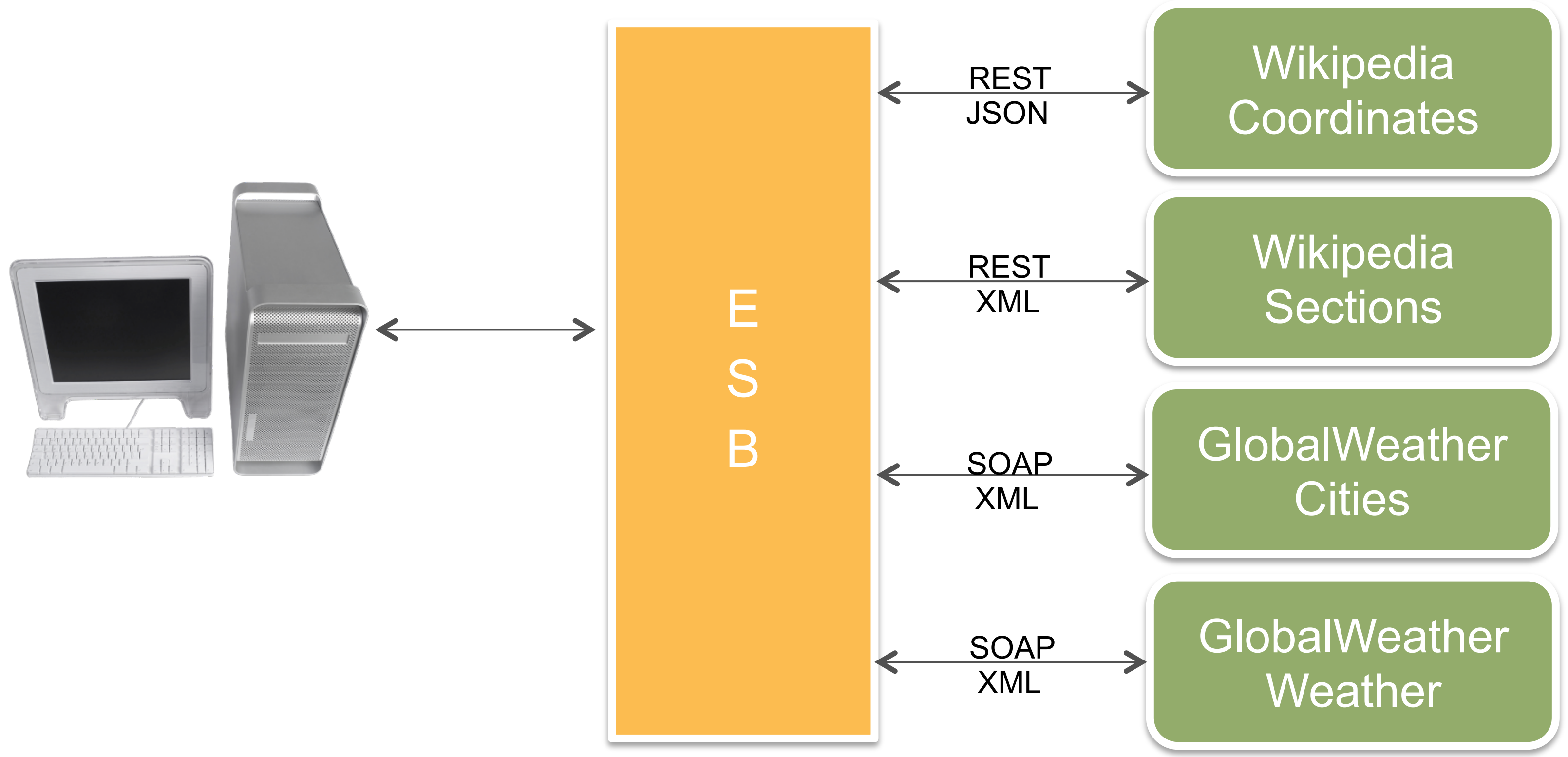




Exercise #4

- Call Managed Web Services with External Web Service Client

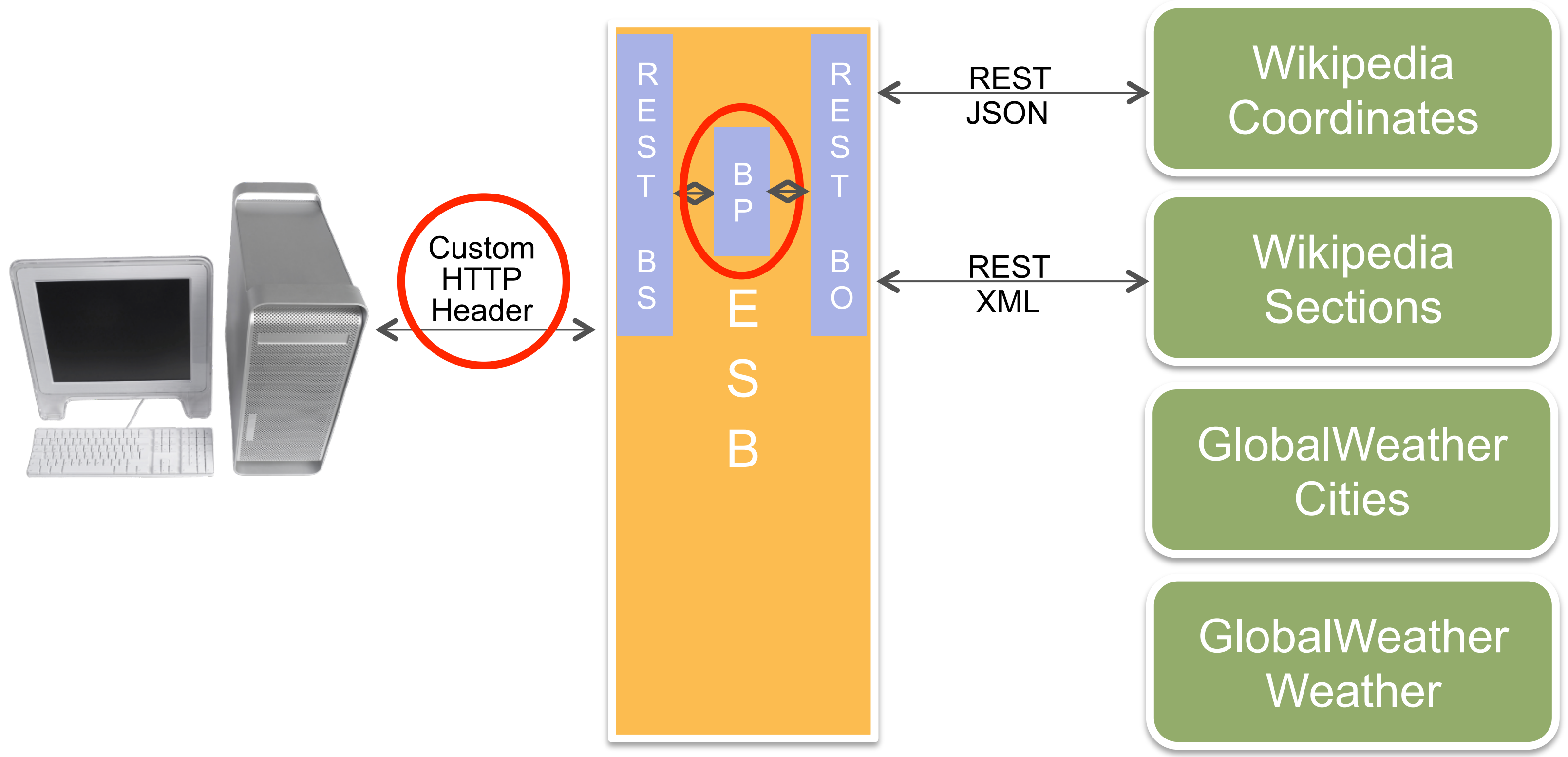
  Test our managed REST- and SOAP-based services with SoapUI.



Exercise #5

- Handle a Custom HTTP Header in Managed REST Service

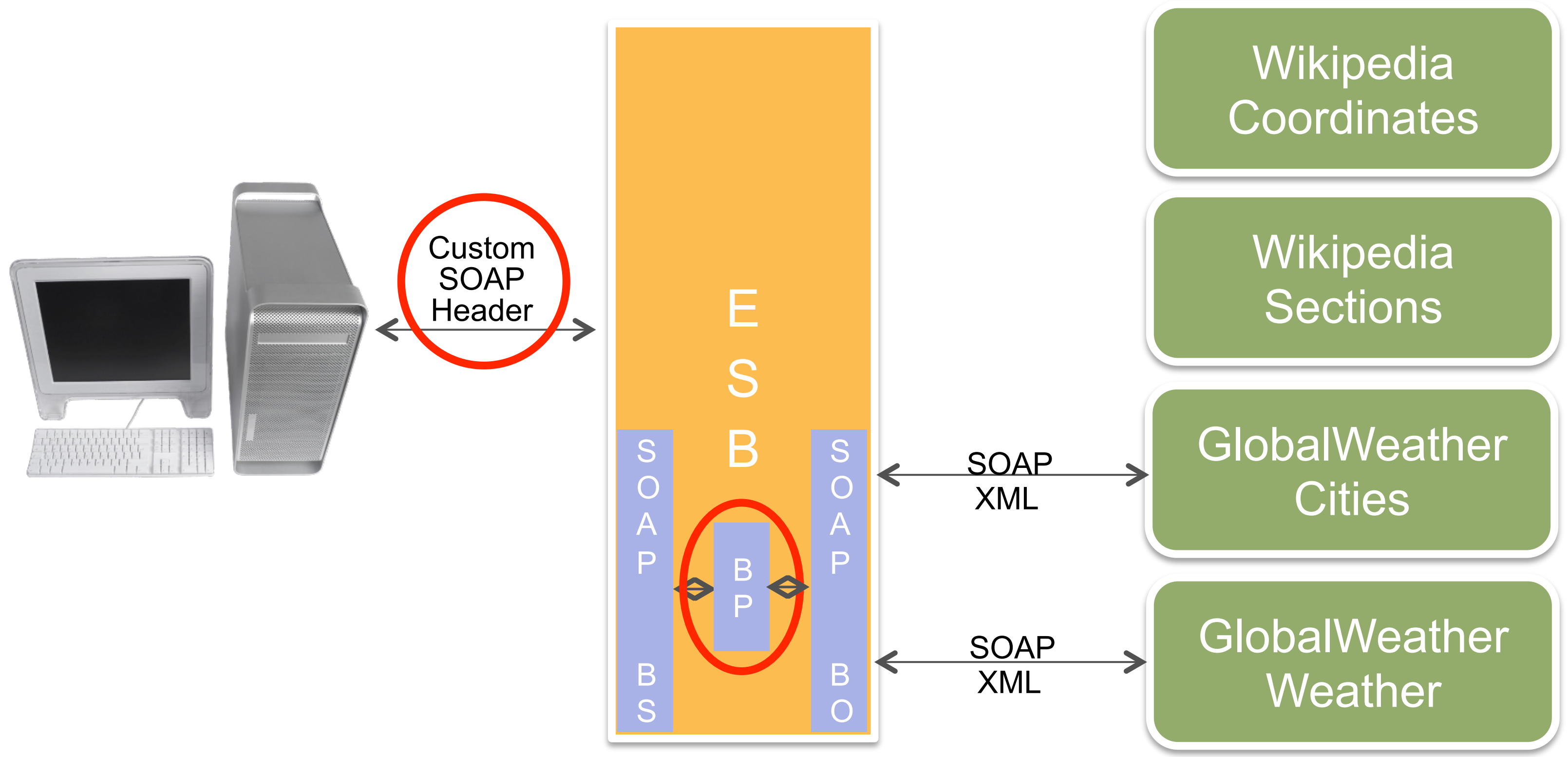
  Insert a simple Business Process to count requests by customer



Exercise #6

- Handle a Custom SOAP Header in Managed SOAP Service

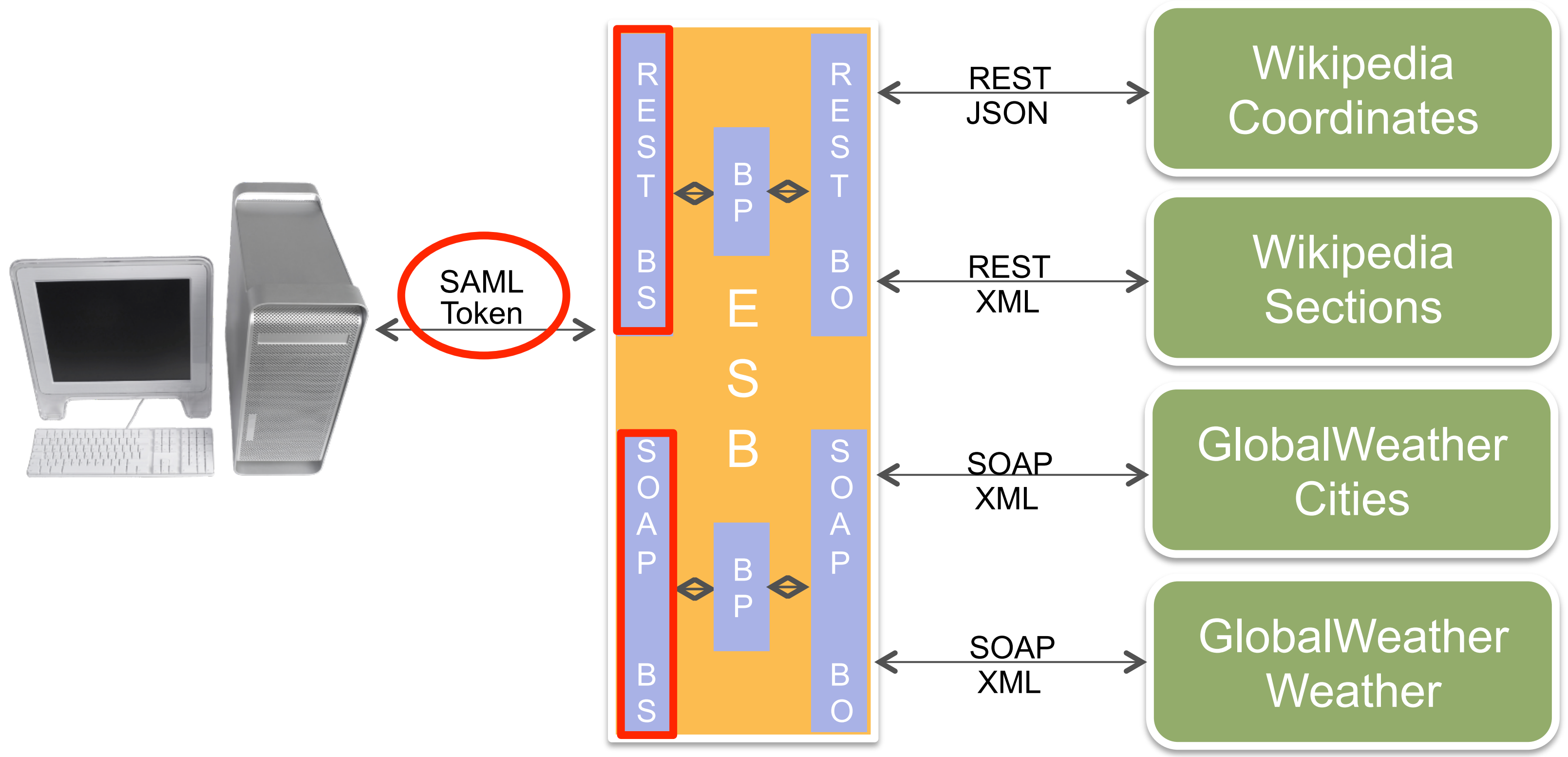
  Insert a simple Business Process to count requests by customer.



Exercise #7

- Validate Inbound SAML Tokens in Managed Web Services

  Require that incoming requests for our managed web services contain a valid SAML token.



ESB Plans

- New Service Registry
- Persistence-Free Implementation for Optimal Performance
- Continued development/enhancement around partners' needs
  - Please contribute!

Related Sessions and Academies

- InterSystems' New Service Registry (Session)
  - Tuesday, 9:00am – 9:30am
  - Wednesday, 11:00am – 11:30am
  - Regency Ballroom O
- Routing REST/SOAP Through Your ESB (Session)
  - Tuesday, 4:50pm – 5:20pm
  - Wednesday, 11:40am – 12:10pm
  - Regency Ballroom O
- Implementing RESTful Applications (Academy)
  - Tuesday, 1:00pm – 3:00pm
  - Wednesday, 8:30am – 10:30am
  - Orlando Ballroom L

Q & A

You'll make breakthroughs

INTERSYSTEMS®

Configuring an Enterprise Service Bus

Mike Moulckers
Ray Wright