



IBM Software Group

Accelerate and simplify deployment of SOA Solutions with Rational Business Developer extension

Stefano Sergi

Product Line Manager

sergi@us.ibm.com

Rational software



Topics

- Why and what is SOA
- Services development challenges
- Developing services in EGL
- Using Services in EGL
- Generating Web Services
- Summary



Business flexibility depends on IT flexibility

“Today’s IT architectures are the biggest roadblocks most companies face when making strategic moves.”

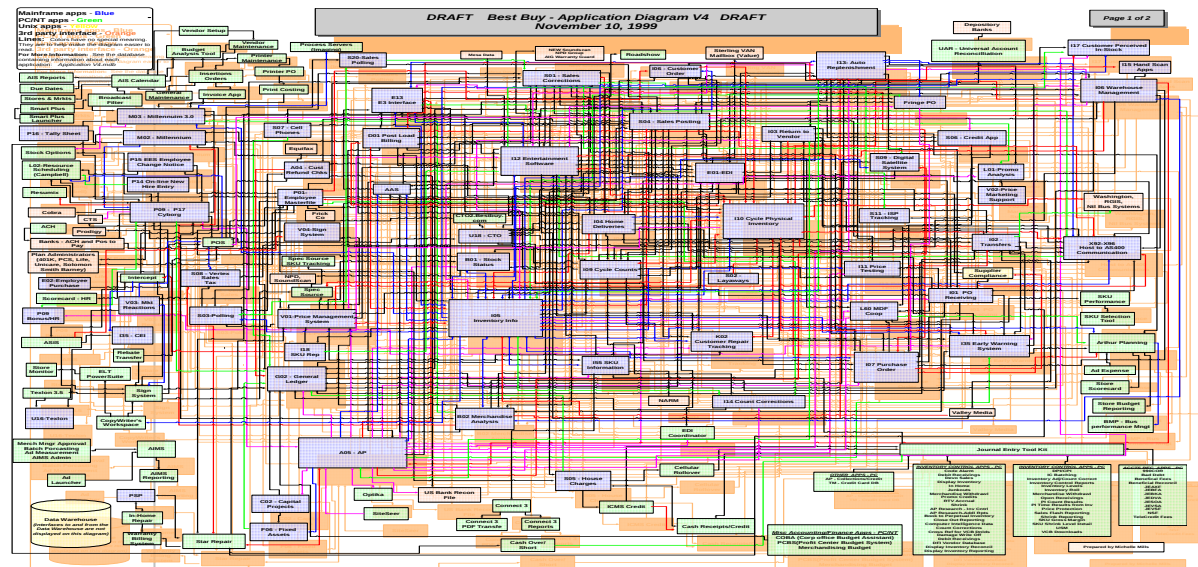
–McKinsey

“Flexible IT, Better Strategy”



Why are today’s architectures roadblocks?

- Complexity
- Monolithic and silo’d applications
- Hidden interfaces
- Custom coded connections
- Not designed for change



Actual application architecture for a consumer electronics company



What is?

... a service?

A **repeatable business task** – e.g.,
check customer credit;
open new account

... service orientation?

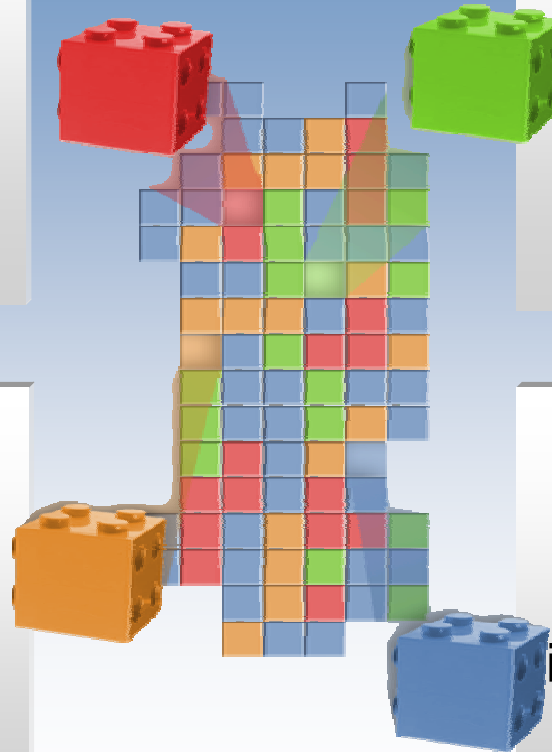
A way of integrating your
business as linked services
and the outcomes that
they bring

... service oriented architecture (SOA)?

An IT **architectural style** that supports
service orientation

... a composite application?

A set of **related & integrated** services that
support a business
process built on an SOA



What is difficult about developing Services?

- Realization and construction of services requires developers to deal with additional concerns
 - ▶ Understanding and mastering Web Services standards (e.g. XML, WSDL, SOAP, Services invocation frameworks, etc)
 - ▶ Creating service facades to existing programs and data: dealing with adapters or integration APIs
 - ▶ SOA crosses platform boundaries, logical services are often realized across different machines, systems, IT resources. Creating new services requires specialized skills for each of these



Why Enterprise Generation Language for SOA?

- Developing Services can be slow, repetitive and error prone
 - ▶ Complex low level coding bogs down programmers
- Many developers skills are “business oriented”
 - ▶ Know the business...been building business applications for years
 - ▶ RPG, COBOL, PL/I, 4GL, Visual Basic
 - ▶ ... but Services typically require knowledge of Java and the Web Services standards
- Re-training may not be an option
 - ▶ High costs
 - ▶ Business pressure may not afford time
 - ▶ Results may be sub-optimal
- Many “legacy developers” retiring
 - ▶ .. who is going to create services for CICS, IMS, iSeries?



EGL for SOA Design Points

- Develop Service independently from where they will execute (Platform Independent Specification)
- Immediately useable by developers of any background
- Hide the technical complexity
- Support SOA standards
- Guarantee optimal (native) deployment to any platform
 - ▶ New and traditional
- Inter-operability for easy “Service-enablement” of legacy



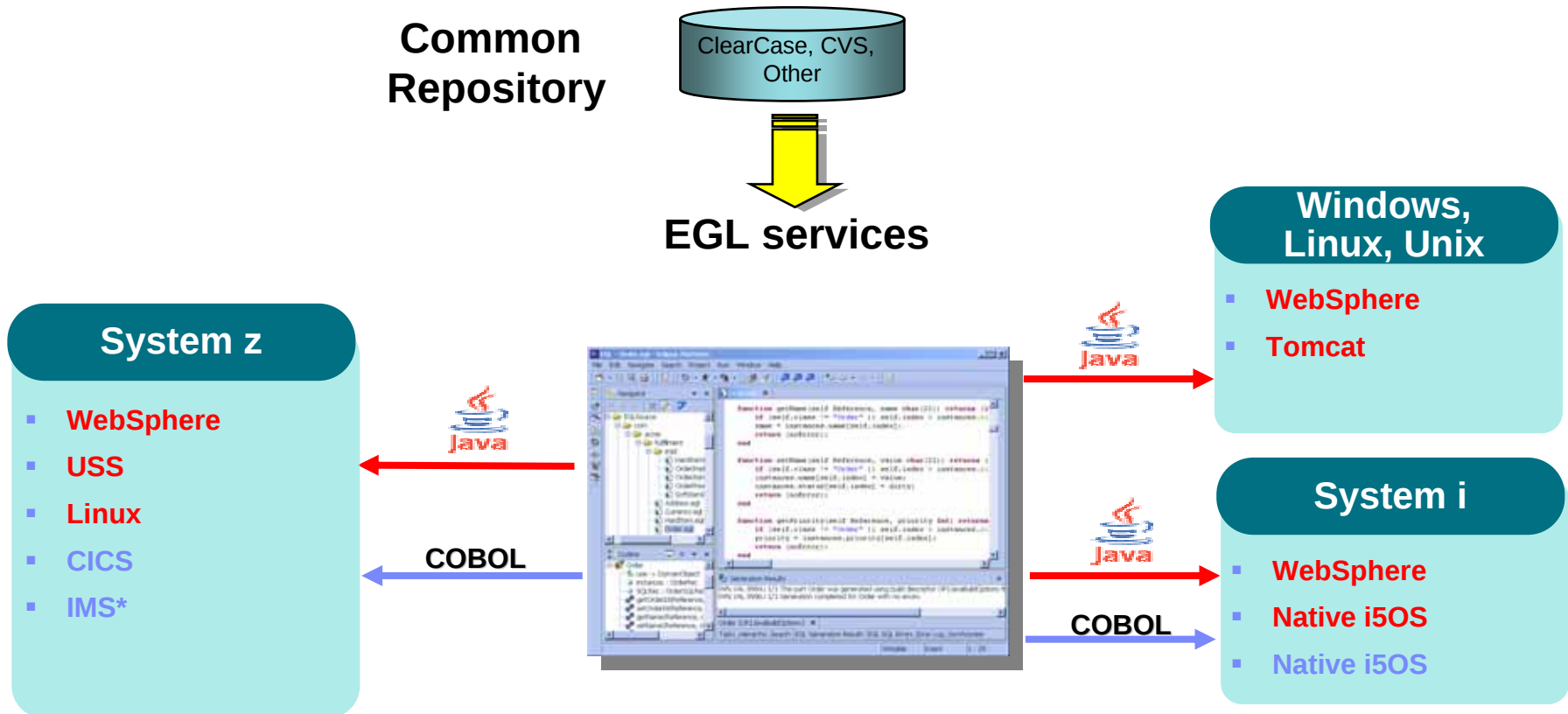
Topics

- Why and what is SOA
- Services development challenges
- **Developing services in EGL**
- Using Services in EGL
- Generating Web Services
- Summary



EGL Platform Flexibility

Code once, deploy anywhere



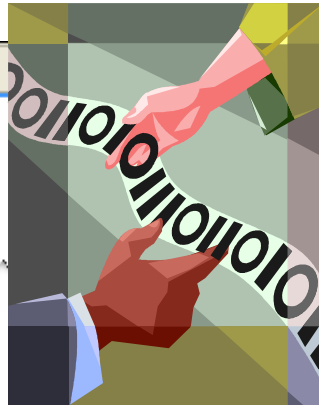
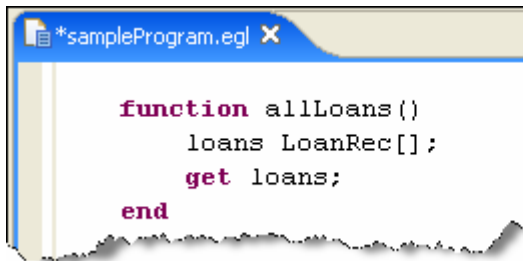
* Planned



The power of abstractions

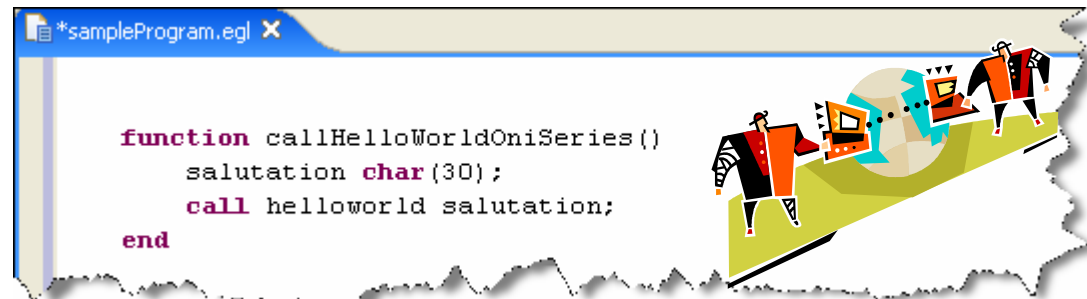
■ Data access:

- ▶ “Records” provide access to:
 - SQL, Indexed, Relative, Serial, DL/I, MQ, Service data
- ▶ Common Verbs for data access (**Get, Add, Replace, Delete**)
- ▶ Allows complete access to SQL statement if needed
- ▶ Common Error Handling



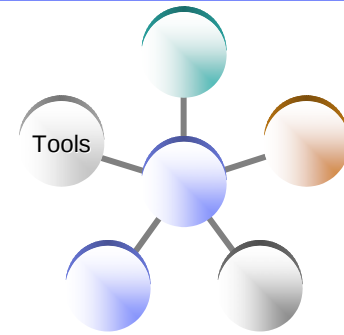
■ Remote Invocation

- ▶ Call COBOL, RPG, C, Java
- ▶ Linkage information separated from code
- ▶ Data mapping, protocol invocation all resolved at runtime, NO code necessary!



The power of tools: Model Transformations

A new generation of Architected Rapid Application Development



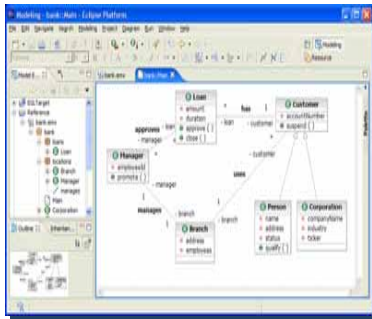
- ▶ Transform UML models to EGL
- ▶ Best way to go...

from **SOA models**

to **construction of services**

to **services**

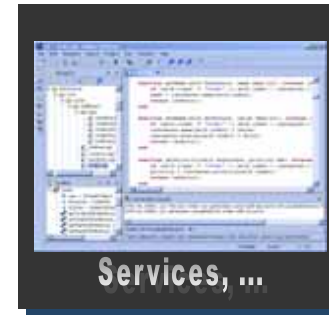
deployed on System z, System i, or anywhere else



1. Model



2. Define Transformation Parameters



3. Transform to EGL code



4. Deploy to platform (z, i, or distributed)

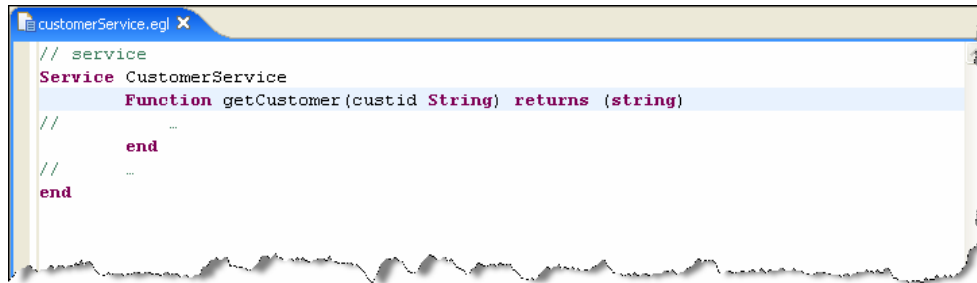
- Traceability from requirements to code
- Create your own transformations
- Transformations enriched by Transformation Parameters
- Easily build / deploy Services on host



The power of Services

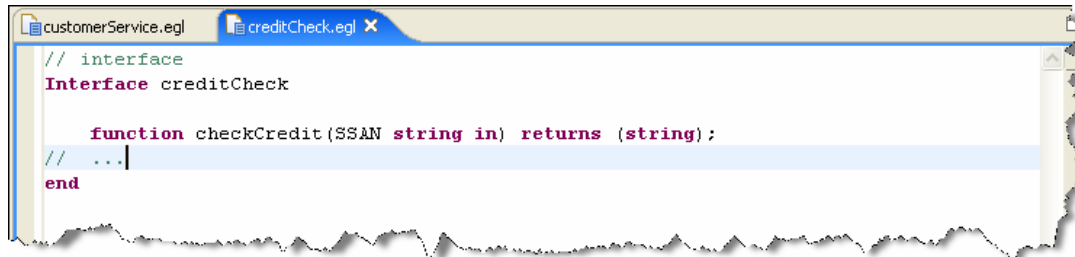
Built into the language

- Service part
 - ▶ a generatable part containing code that will be accessed
 - from EGL code by way of a local or TCP/IP connection (*EGL Service*)
 - from any code by way of an HTTP connection (*EGL Web service*).



```
// service
Service CustomerService
  Function getCustomer(custid String) returns (string)
  ...
end
end
```

- Interface part
 - ▶ Used to access external services as EGL services or simply to provide separation of concern



```
// interface
Interface creditCheck
  function checkCredit(SSAN string in) returns (string);
  ...
end
```



The power of Services

EGL: cross platform language for business oriented services development

At development time...

- Focus on the business logic
- Implement SOA design elements: services and interfaces
- Leverage existing business developers for new SOA development
- Ignore deployment targets/technology while coding/testing

Leverage external web services...

- EGL Interfaces
 - represent external web services
 - Are created via import from WSDL
 - Allow the EGL developer to stay within the context of the EGL programming model

Deploy EGL services...

To any platform

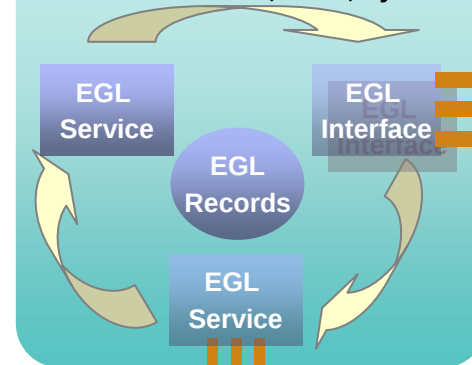
- Java to WAS/Tomcat/etc.
- COBOL to CICS, iSeries (1Q 2007)
- COBOL to IMS (2H 2007)

As...

- A Web service (uses SOAP)
- A private service (uses CICS ECI or TCP)
- Other SOA runtimes when they reach critical mass

Deploy Services as Web Services

EGL SOA for WAS, CICS, System i



Consume external services

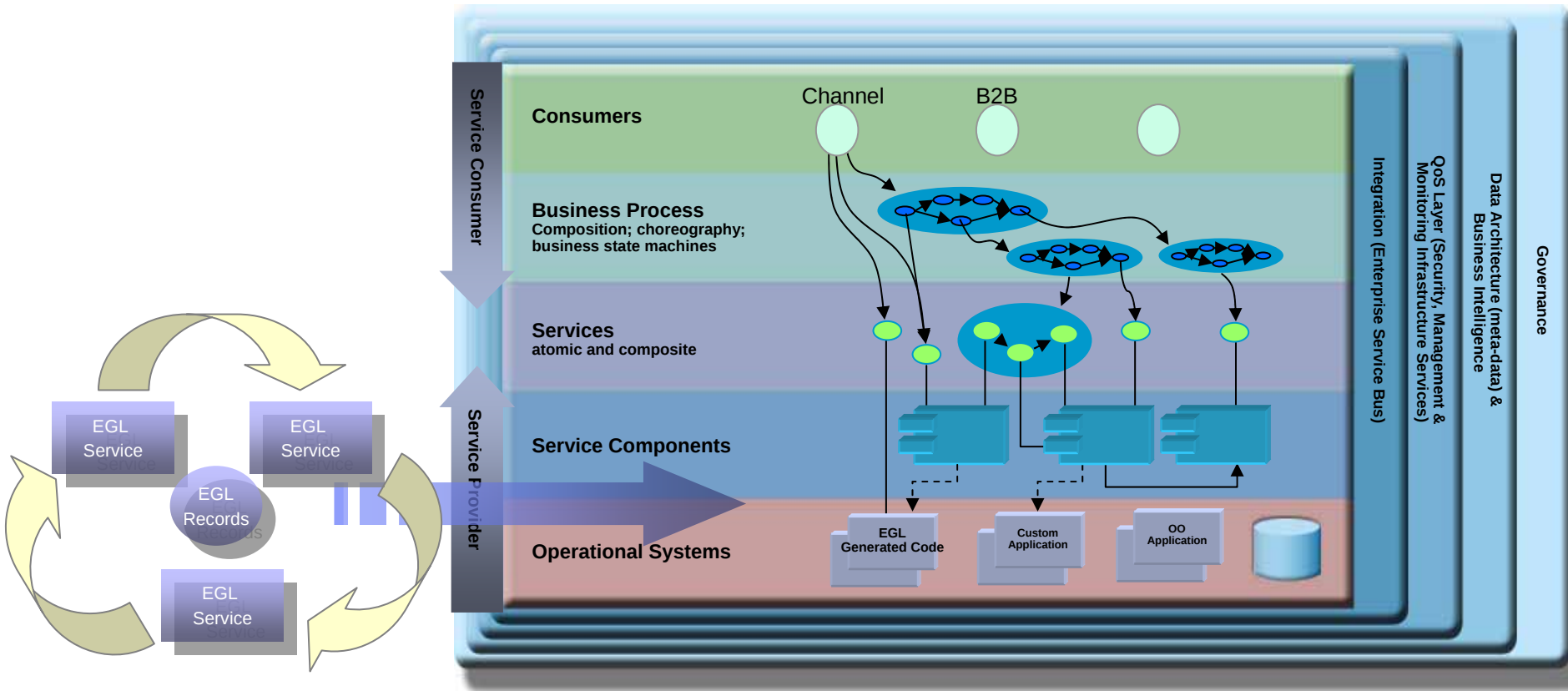


The power of Services

Seamless integration with SOA stack

EGL Services can be generated into deployable artifacts that are accessible as Web Services

EGL data appears as XML payload with no need for transformation

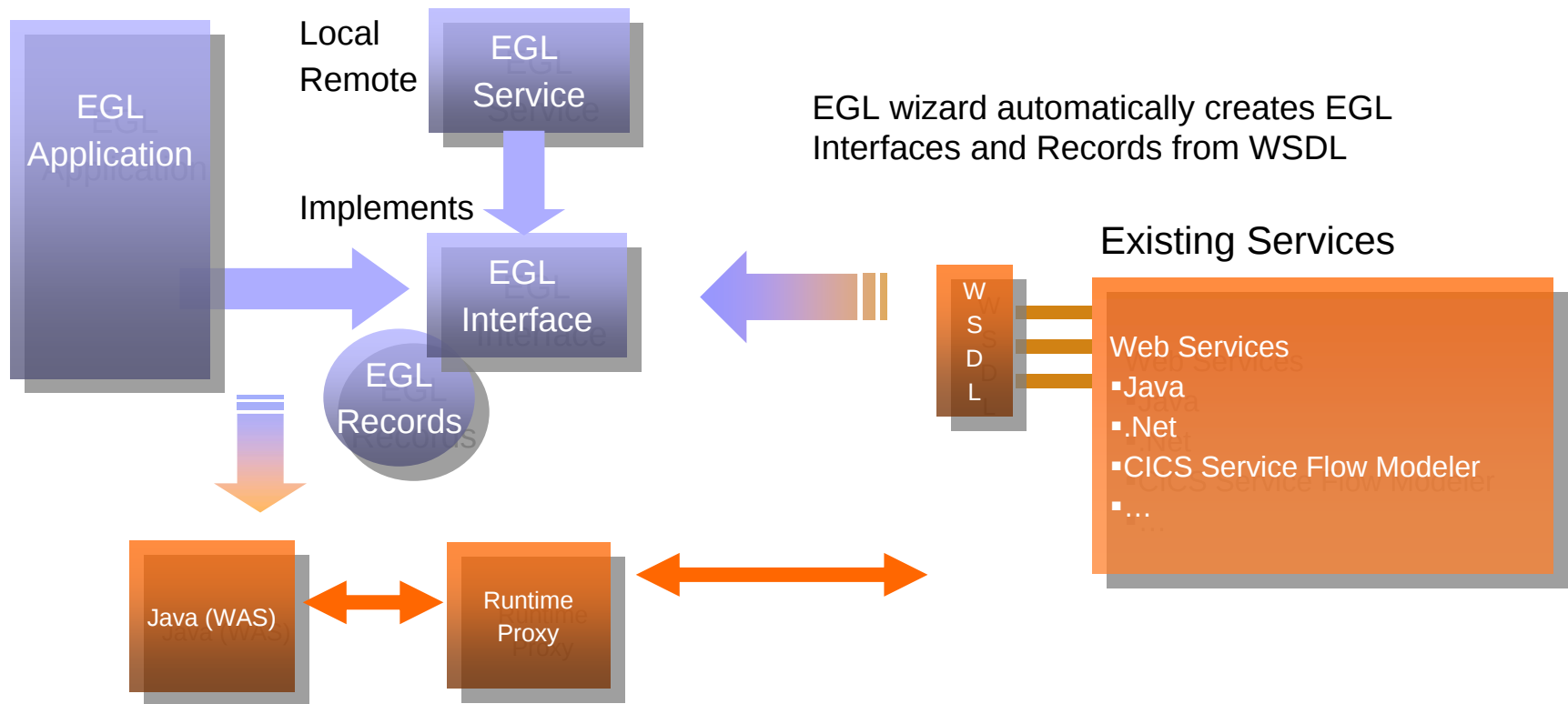


Topics

- Why and what is SOA
- Services development challenges
- Developing services in EGL
- **Using Services in EGL**
- Generating Web Services
- Summary



Leveraging Existing Services from EGL



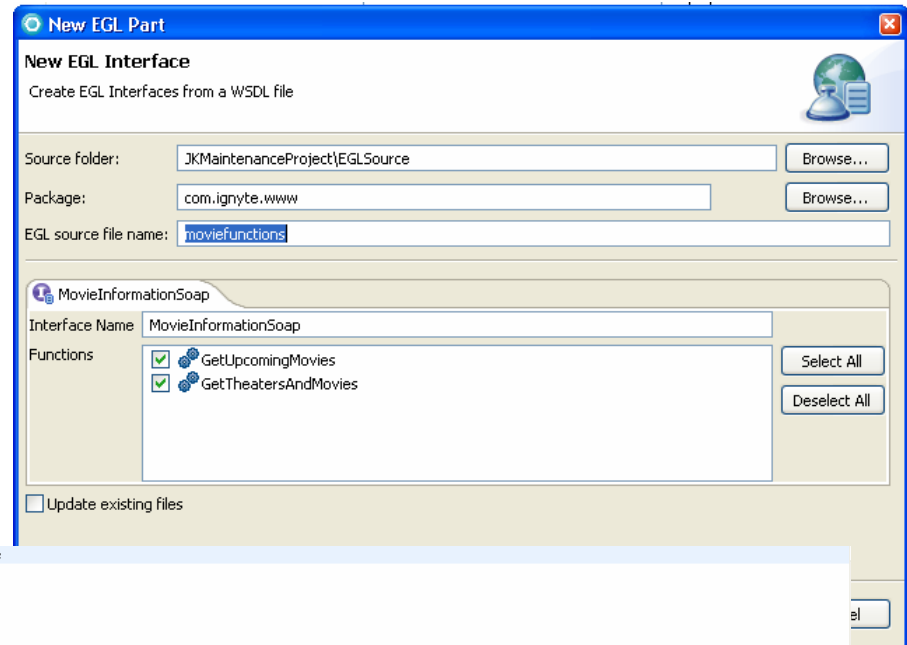
EGL applications invoke operations in EGL Interface and send Records as parameters

EGL generator generates Runtime proxy that handles web service invocation and XML conversion



Consuming Services...EGL Artifacts From WSDL Files

- External service consumption is wizard driven and simple
- Intuitive interface to allow easy EGL artifact creation
- WSDL or ASMX files are parsed and consumed
- All necessary Interfaces and data structures are created in EGL



```
package com.ignite.www;  
  
record Theater()  
  Name string?;  
  Address string?;  
  Movies Movie?[];  
end  
  
record Movie()  
  Rating string?;  
  Name string?;  
  RunningTime string?;  
  ShowTimes string?;  
end  
  
record UpcomingMovie()  
  MovieName string?;  
end  
  
interface MovieInformationSoap(@xml (name="MovieInformationSoap", namespace="http://www.ignite.com/whatsshowing"))  
  function GetUpcomingMovies(month int in, year int in) returns(UpcomingMovie?[]){@xml (name="GetUpcomingMovies");}  
  function GetTheatersAndMovies(zipCode string? in, radius int in) returns(Theater?[]){@xml (name="GetTheatersAndMovies");}  
end
```

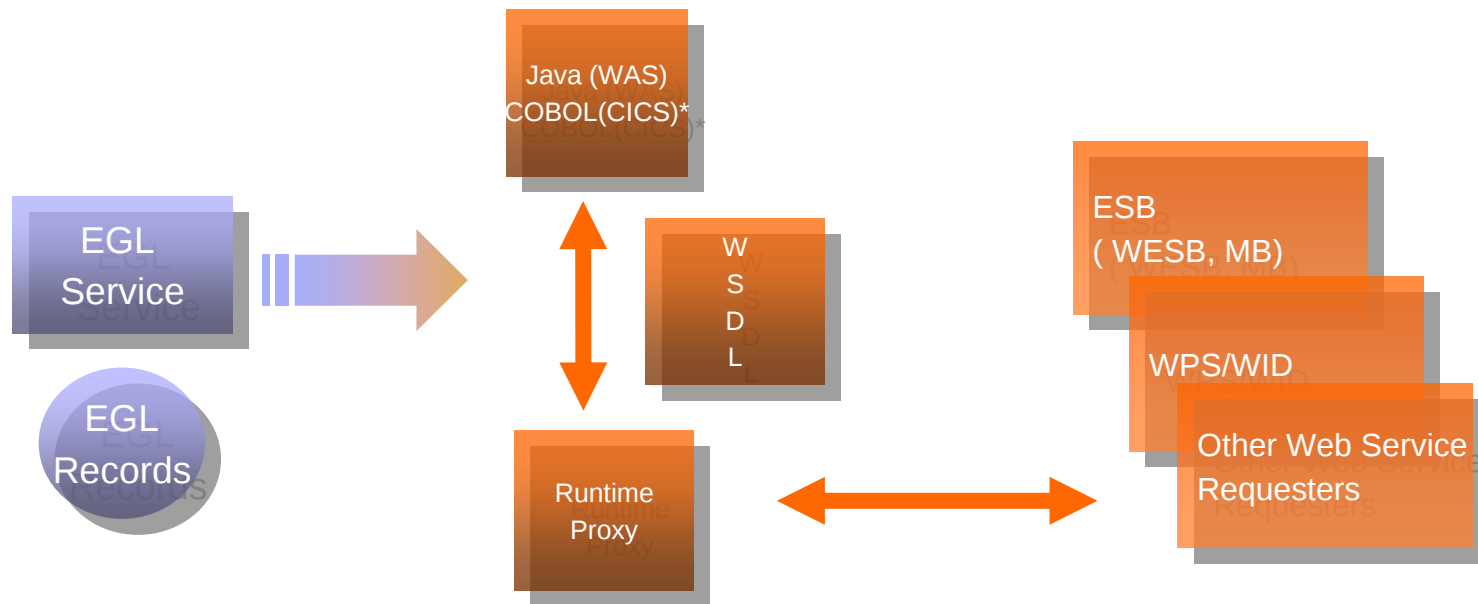


Topics

- Why and what is SOA
- Services development challenges
- Developing services in EGL
- Using Services in EGL
- **Generating Web Services**
- Summary



Generating Web Services consumable in a SOA



EGL Services can be generated into deployable artifacts that are accessible as Web Services

EGL data appears as XML payload with no need for transformation

Choice of native EGL services or EGL generated Web Services (see speaker notes)

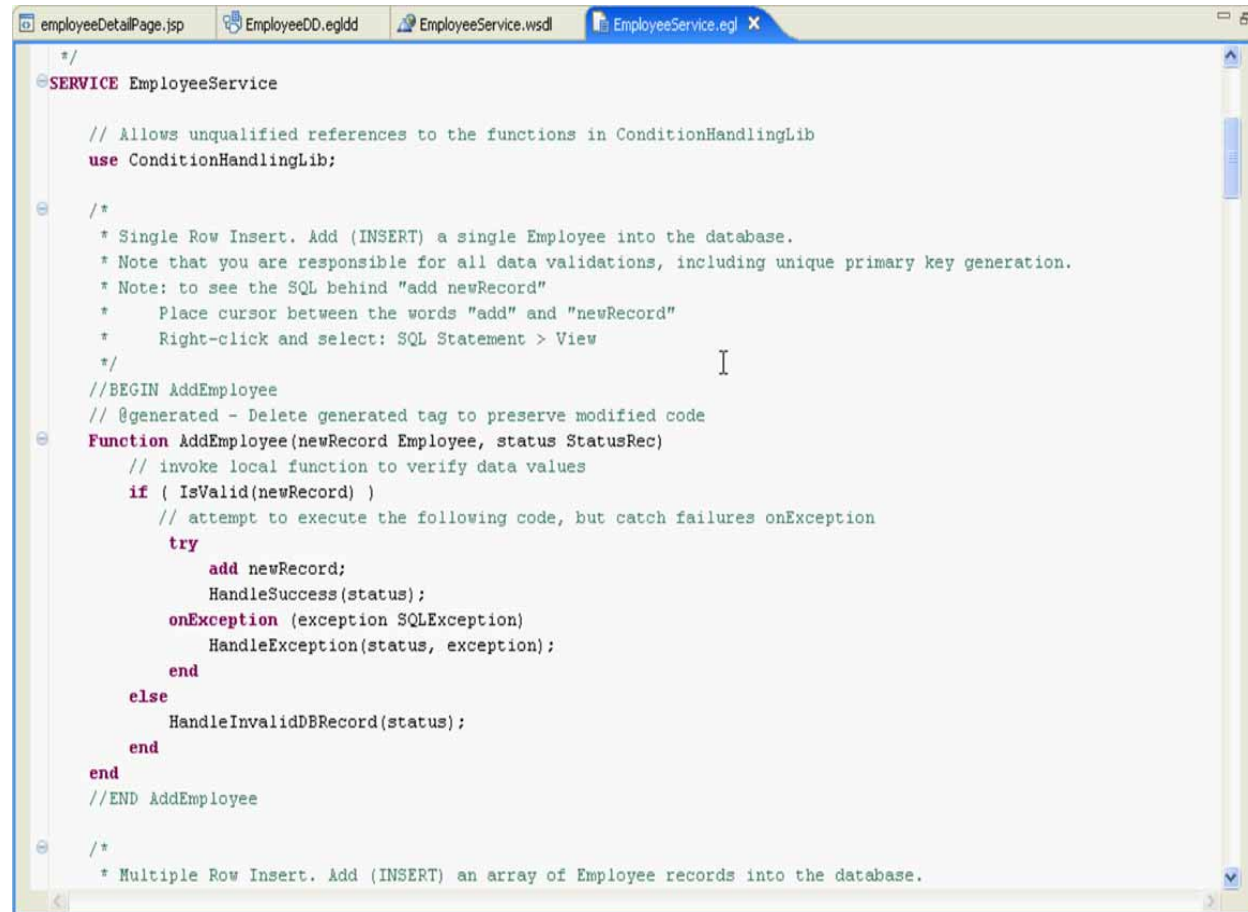
Native EGL services are efficient and faster than traditional web services

Remote or Local native EGL service available for generation



Creating Services...Typical Service Example

- Coding and deployment of services is easy and user-friendly
- No necessity to understand plumbing
- Procedural and reusable
- Full range of EGL functionality available to an EGL generated service
- **Simple CALL statement to allow remote programs to be exposed through EGL services**



```
employeeDetailPage.jsp EmployeeDD.egldd EmployeeService.wsdl EmployeeService.egl x
/*
SERVICE EmployeeService

// Allows unqualified references to the functions in ConditionHandlingLib
use ConditionHandlingLib;

/*
 * Single Row Insert. Add (INSERT) a single Employee into the database.
 * Note that you are responsible for all data validations, including unique primary key generation.
 * Note: to see the SQL behind "add newRecord"
 *   Place cursor between the words "add" and "newRecord"
 *   Right-click and select: SQL Statement > View
 */
//BEGIN AddEmployee
// @generated - Delete generated tag to preserve modified code
Function AddEmployee(newRecord Employee, status StatusRec)
// invoke local function to verify data values
if ( IsValid(newRecord) )
// attempt to execute the following code, but catch failures onException
try
add newRecord;
HandleSuccess(status);
onException (exception SQLException)
HandleException(status, exception);
end
else
HandleInvalidDBRecord(status);
end
end
//END AddEmployee

/*
 * Multiple Row Insert. Add (INSERT) an array of Employee records into the database.
```



Topics

- Services development challenges
- Developing services in EGL
- Using Services in EGL
- Generating Web Services
- **Summary**



Summary

- Lower the SOA adoption barriers
 - ▶ EGL is easy for business developers
 - ▶ EGL generated services deploy natively to all platforms
 - (CICS, WAS, IMS, System i)
- Accelerate SOA Solutions deployment
 - ▶ EGL is more productive
 - ▶ Requires minimum training and can be used by developers of any background
- For more information
 - External RBDe Web Site (www.ibm.com).
<http://www-306.ibm.com/software/awdtools/busdev/ext/>
 - External RBDe Web Site (developerWorks – RBD Extension zone).
<http://www-128.ibm.com/developerworks/rational/products/rbde>



Questions