IBM Software Group

EGL Overview

P. Michael Virga Jr.
IBM EGL EcoSystem Team, IBM Software Group
mvirga@us.ibm.com

So what is Enterprise Generation Language (EGL)?

A modern programming language that leverages existing ***business developer*** *skills while hiding complex J2EE and other runtime technologies. EGL automates construction of state-of-the-art commercial applications with incredible productivity!*

More importantly, EGL is a complete application development environment that is part of IBM's software development platform designed for developers who need to solve business problems, not technology problems.

What is EGL ?

- **EGL** = **E**nterprise **G**eneration **L**anguage
  - High level programming specifications
  - Hides complexities of implementation technology
    - For the non-Java programmer
    - For the non-CICS programmer
- Special Parts + Scripting Language
- Interactive Development and Debugging
  - Environment independent language
  - Built-in debugger
  - Can be used for RAD Rapid Application Development
    - Prototyping

Net it out for me … EGL is a

Business Programming Language

- Robust, easy-to-learn, mature
  - Based on long history of business language expertise (CSP, VAGen, I4GL)
  - Over 25 years of R&D + production use
- **No need to understand Java/J2EE … No need to be *Object-Oriented***
- Aimed squarely at **business application developers**, who need to **solve business problems** as quickly and efficiently as possible

+

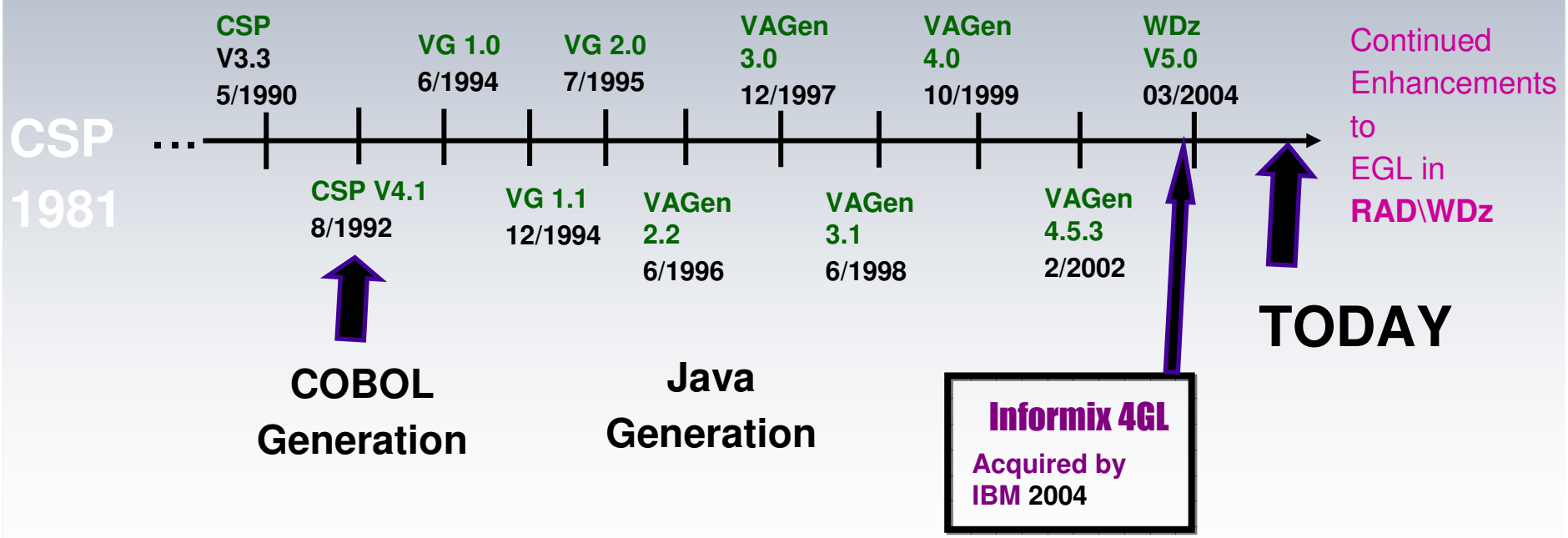
Development Environment

- Seamlessly integrated into Rational Software Development Platform (SDP)
  - Benefits from all SDP R&D and award-winning technologies
  - Based on Eclipse
- Based on industry-standard artifacts and development patterns (MVC)

History

History of preserving the value of customer investment

- Robust, easy-to-learn, mature
  - Based on long history of business language expertise (CSP, VAGen, I4GL
  - Over 25 years of R&D + production use



Value Proposition for 3GL & 4GL Developers

- Modern state-of-the-art application development environment
  - Easy to learn, highly productive without knowing a line of Java (or COBOL for that matter)
    - First-class Service support for SOA
    - Work side-by-side with Java developers and COBOL developers – coexistence - no more programming 'silos
    - Distributed development and unit testing (Windows & Linux) lowers MIPs usage related to development
- Deployment platform flexibility (e.g. Windows, System z, System i, Unix)
- COBOL generation for CICS, IMS and batch as required
  - Code generation makes IT staff more productive, reduces code errors and increases governance
- Access to WebSphere MQ, VSAM, DB2 and other data sources using the same I/O verb set
- Integrated with Rational Software Lifecycle portfolio (ReqPro, ClearCase, ClearQuest)'

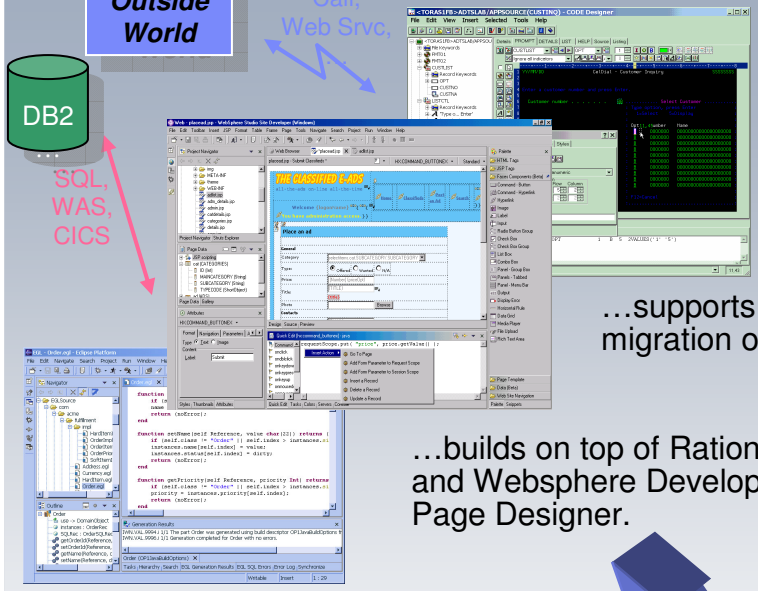
Features and Capabilities

The Outside World

MQ, Call, Web Srvc,

DB2

SQL, WAS, CICS



Productivity, High Quality, Leading Edge Technology

Enable broad class of business developers for leading-edge technical work

…supports Text-based UI's for migration of existing apps

…builds on top of Rational Developer and Webspheres Developer Tools, like Page Designer.

…Highly productive and intuitive business language, simplifies complex runtimes & enhances quality by reducing amount of code needed

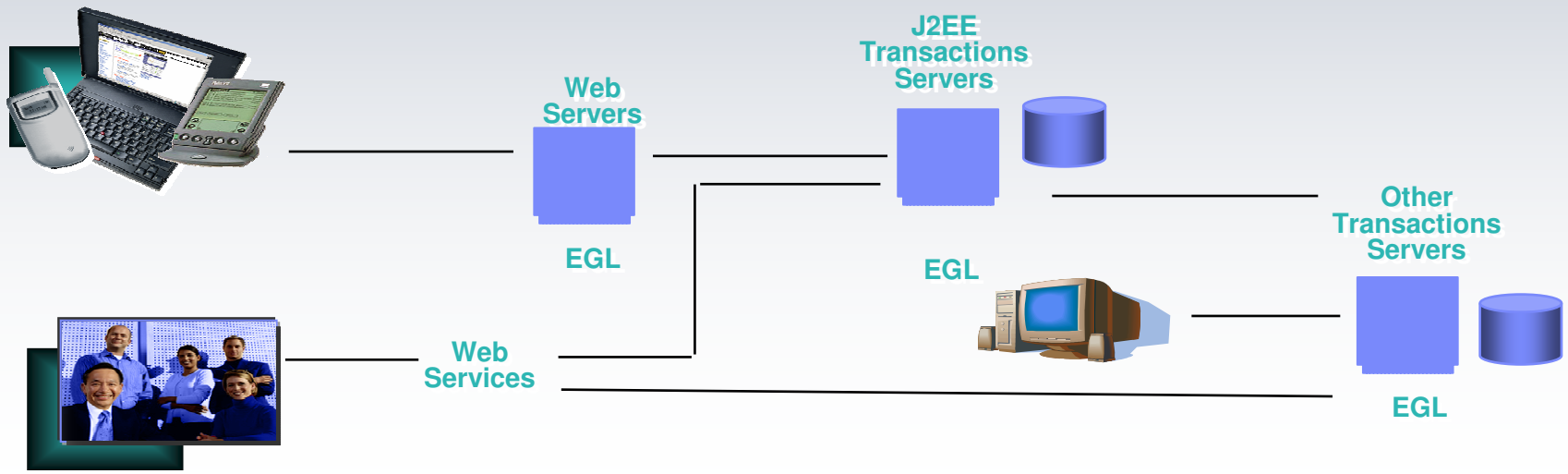
EGL

Deploy optimally to diverse platforms

For *developers* who need to *solve* *Business Problems* not *Technology* Problems

What Applications can be developed in EGL?

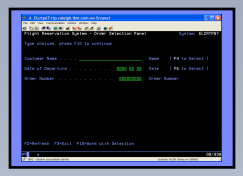
- Internet applications (Web, Rich Client)
- SOA - Web Services, EGL Services
- Database applications (CRUDs)
- Callable programs from traditional environments
- Standalone batch applications (e.g. COBOL)
- Standalone green-screen application
  - For iSeries, CICS (zOS), Linux, Unix, Windows

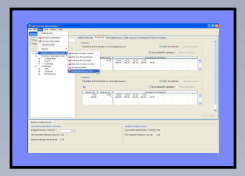


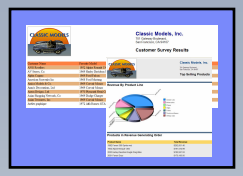
EGL Application Design : Flexibility

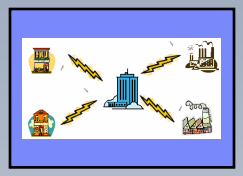
Presentation

Batch



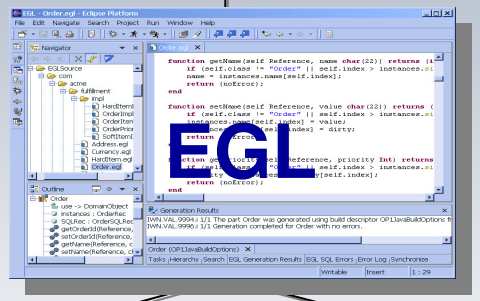






Batch | 3270/5250 | Web | Rich Client* | Reports | Services

Business Logic & Control



Data Access

MQ Series

DB2
Informix

SQL Server
Oracle

MySQL
Derby Informix
Cloudscape

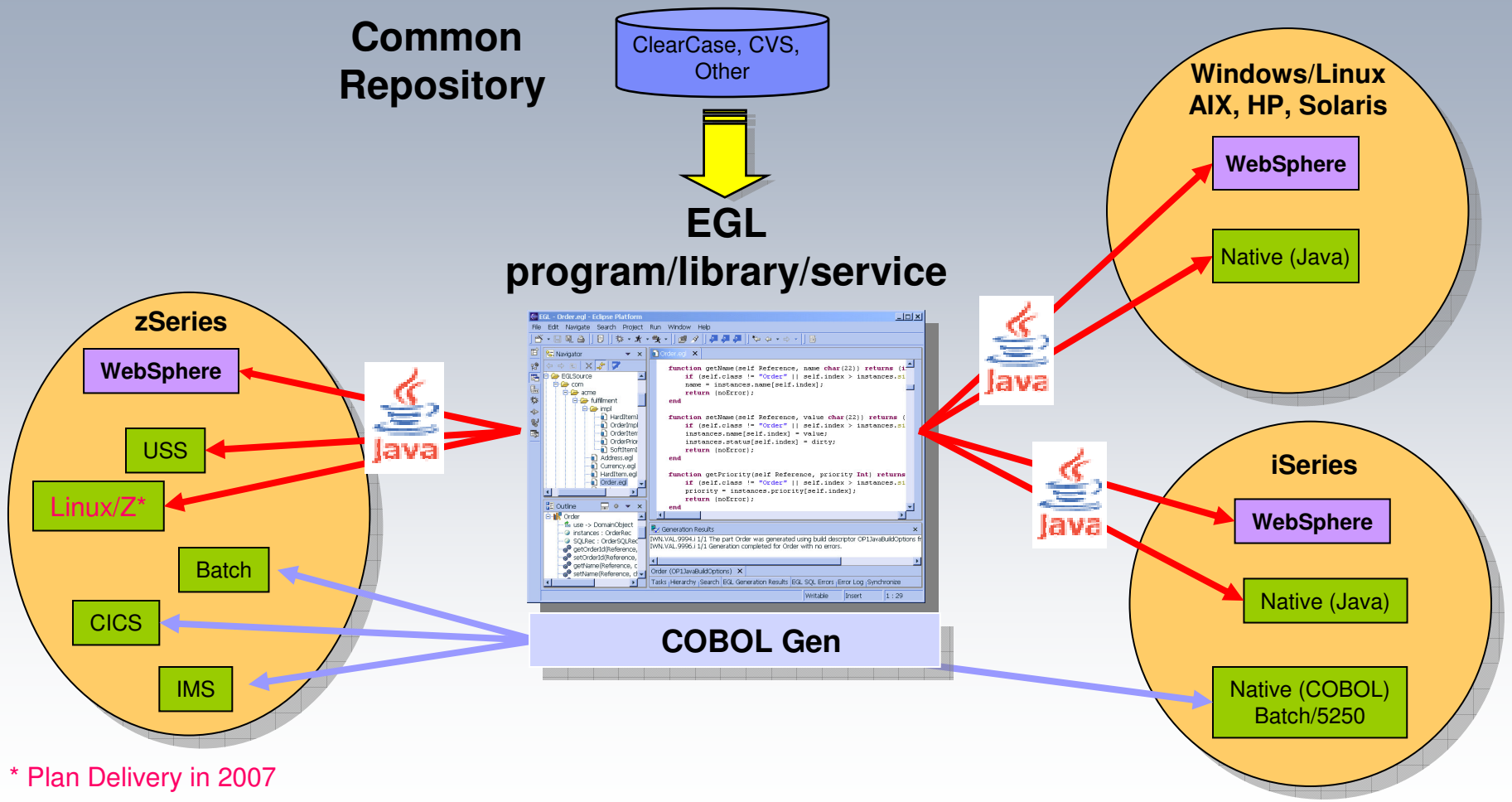
Serial, Relative,
Indexed (VSAM)
Other…

IMS DB | IMS DC | COBOL RPG | Java J2EE | C / C++ PLI | CICS

** Deliver 1H07*

Integration with and Generation of Existing Applications

EGL Platform Flexibility – Code once, deploy anywhere

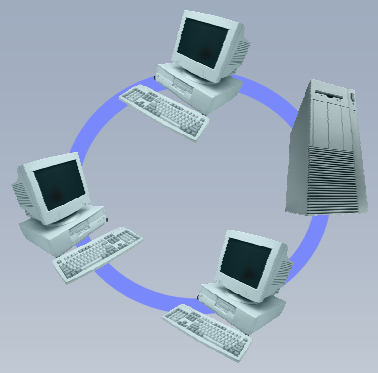


* Plan Delivery in 2007

EGL Development Overview

Develop

- High level abstraction specification
- Target platform neutral
- Shield complexity of target system
- Interactive test of logical specification
- Promotes Iterative development
- Strong Team support

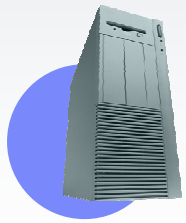


Generate

- Transform EGL logical specification into Java or COBOL (zOS)
- Create build script for zOS build server
- Create class files ready to export to JAR for deployment

Deploy and Run

- zOS: CICS or Batch (IMS in follow on release)
- iSeries
- Windows
- Linux, AIX , Solaris and HP



EGL Business Logic Example – Server program

```
package pot.programs;
import pot.records.*;
// basic called program
program TE01A type basicProgram (awork TE01W01)  // Received record
 arecord TE01R01; // SQL Record Data declaration
 function main()
  move awork.id to arecord.id; // Move from work to re
  TE01_READ(); // read DB2 table
  if  ( arecord is noRecordFound)    /* if not found*/
        awork.MESSAGE = "Record  does not exist";
      else
        awork.MESSAGE = " ";
        move arecord to awork;     /* move from DB2 record to work*/
      end
  end

 function TE01_READ(); // function to read DB2 table
  try
   get arecord usingKeys arecord.ID ;
  end
 end
end
```

*TE01W01.egl

```
package pot.records;
record TE01W01 type basicRecord
    03 ID smallInt;
    03 NAME char(9);
    03 SALARY decimal(7,2);
    03 COMM decimal(7,2);
    03 MESSAGE char (50);
end
```

*TE01R01.egl

```
package pot.records;
record TE01R01 type sqlRecord
    {tableNames=(STAFF T1)}
    ID smallInt            {column="ID", isRead
    NAME char(9)           {column="NAME", isRe
    DEPT smallInt          {column="DEPT", isRe
    JOB char(5)            {column="JOB", isRea
    YEARS smallInt         {column="YEARS", isR
    SALARY decimal(7,2)    {column="SALARY", is
    COMM decimal(7,2)      {column="COMM", isRe
end
```

View SQL Statement

```
select
    ID, NAME, DEPT, JOB, YEARS, SALARY, COMM
into :arecord.ID, :arecord.NAME, :arecord.DEPT, :arecord.JOB,
    :arecord.YEARS, :arecord.SALARY, :arecord.COMM
from STAFF T1
where
    ID = :arecord.ID
```

Simplicity – Basic Premise of EGL

Simple Data Access

- Records provide access to SQL, Indexed, Relative, Serial, DL/I, MQ, Service data
- Common Verbs identified for data access (**Get, Add, Replace, Delete**)
- Complete access to SQL statement available, if needed
- Common Error Handling provided

```
Function allLoans()
        loans LoanRec[];
        get loans;
End
Function loansInFlorida()
        loans LoanRec[];
        get loans with #sql{
                select *
                from LOAN
                where state = "FL"};
End
```

DDL

Simple Validation/Editing Rules

- Specified via properties in "Data Items"
- Define formatting & validation rules once in common place
- Reuse data items for Records, screens, reports

```
DataItem Password char(10) {
        validationFunction = "passwordValidation",
        displayUse = secret,
        displayName = "Enter your password",
        inputRequired = yes}
End
```

Simple Transaction Control

- Manage UOW independent of transaction manager (CICS, IMS, DB2, WAS or combinations)

```
Function loansInFlorida()
        …
        commit();
        …
        rollback();
End
```

Simple Invocation

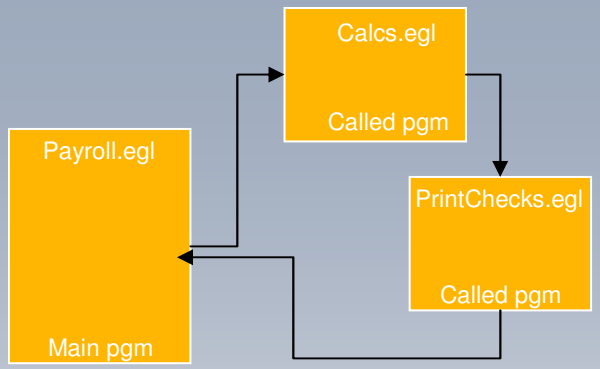
- Call COBOL, RPG, C, Java
- Linkage information separated from code… simplifies development

```
Function callHelloWorld()
        salutation char(30);
        call helloworld salutation;
End
```

Properties of selected callLink elements:

Property	
pgmName	
type	
alias	
conversionTable	
ctgKeyStore	
ctgKeyStorePassword	(no value set)
ctgLocation	(no value set)
ctgPort	(no value set)
javaWrapper	(no value set)
library	helloworldLibrary
location	rtpas64.raleigh.ibm.com
luwControl	(no value set)
package	(no value set)
refreshScreen	(no value set)
remoteBind	(no value set)
remoteComType	JAVA400
remotePgmType	EXTERNALLYDEFINED
serverID	(no value set)

EGL Batch applications



- Easy access to data sources
  - EGL record concept
- Inline coding of SQL statements
  - execute #sql{ ... }
- Command line arguments are supported
  - SysLib.getCmdLineArg()
  - SysLib.getCmdLineArgCount( )
- Character based output and input (line mode)
  - displayLineMode()
  - result = promptLineMode()

RAD\WDz IDE

EGL - HelloWorld.egl - IBM Rational Software Development Platform

File Edit Navigate Search Project Run Window Help

Project Explorer

- EGL Projects
  - HelloWorld
    - EGLSource
      - (default package)
        - HelloWorld.egl
      - HelloWorld.eglbld
    - JavaSource
      - (default package)
        - Ezecustomer.ja
        - HelloWorld.jav
      - runtunit.properties
    - EGL_PLUGINDIR/runtim
    - EGL_PLUGINDIR/runtim
    - EGL_PLUGINDIR/runtim
    - EGL_PLUGINDIR/runtim
    - EGL_PLUGINDIR/runtim
    - EGL_PLUGINDIR/runtim
    - EGL_PLUGINDIR/runtim
    - EGL_GENERATORS_PLI
    - EGL_GENERATORS_PLI
    - ifxjdbc.jar - C:\ifmxjdb
    - JRE System Library [ec
    - bin

HelloWorld.egl

```
Record customer type SQLRecord { tableNames = [["customer"]], keyitems = ["custor
    customer_num int;
    fname char(15);
    lname char(15);
end

program HelloWorld
    customers customer[];

    function main()
        i, max int;
        connect("jdbc:informix-sqli://akoerner:1527/stores:INFORMIXSERVER=swu2005
                    "informix","informix");
        get customers;
        max = size(customers);
        i = 1;
        while (i <= max)
            displaylinemode("customer_num:" + customers[i].customer_num + ", lnar
            i = i + 1;
        end
    end
end
```

Outline

- customer : sqlRecord
  - customer_num : int
  - fname : char(15)
  - lname : char(15)
- HelloWorld : basicProgram
  - customers : customer[]
  - main()

Problems | Console | EGL SQL Errors | EGL Generation Results

<terminated> HelloWorld [Java Application] C:\RAD60\eclipse\jre\bin\javaw.exe (Apr 19, 2005 10:25:41 AM)

```
customer_num:101, lname:Pauli          , fname:Ludwig
customer_num:102, lname:Sadler         , fname:Carole
customer_num:103, lname:Currie         , fname:Philip
customer_num:104, lname:Higgins        , fname:Anthony
customer_num:105, lname:Vector         , fname:Raymond
customer_num:106, lname:Watson         , fname:George
customer_num:107, lname:Ream           , fname:Charles
customer_num:108, lname:Quinn          , fname:Donald
```

Writable | Smart Insert | 23 : 1

EGL - Which MQ access can you code faster?

```
/* Connect to message queue manager
MQCONN(MQSAMPLE_STATE, MQUIR.MQMANAGER) ;
MQWEB_MQCHECK();
/* Put messages to queue ;
IF MQSAMPLE_STATE.COMPCODE LE MQCC_WARNING;
/* Open message queue
   MQOD_INIT(MQOD);
   MOVE MQUIR.MQQUEUE TO MQOD.OBJECTNAME;
   MOVE MQUIR.MQMANAGER TO MQOD.OBJECTQMGRNAME;
   MQSAMPLE_STATE.OPTIONS = MQOO_OUTPUT;
   MQOPEN(MQSAMPLE_STATE, MQOD);
   MQWEB_MQCHECK();
/* Put messages to queue
   IF MQSAMPLE_STATE.COMPCODE LE MQCC_WARNING;
     MQPMO_INIT(MQPMO);
     MQMD_INIT(MQMD);
     MQSAMPLE_STATE.BUFFERLENGTH =
EZEBYTES(MQUIR.PUT_MESSAGE) ;
     MQPUT(MQSAMPLE_STATE, MQMD, MQPMO,
MQUIR.PUT_MESSAGE) ;
     MQWEB_MQCHECK();
     IF MQSAMPLE_STATE.COMPCODE EQ 0;
       MOVE "Message written to queue" TO
MQUIR.ERROR_MESSAGE;
     END;
/* Close queue
     MQSAMPLE_STATE.OPTIONS = MQCO_NONE;
     MQCLOSE(MQSAMPLE_STATE);
     MQWEB_MQCHECK();
   END;
  END;
/* Disconnect from message queue manager
  MQDISC(MQSAMPLE_STATE);
  MQWEB_MQCHECK();
END
```

This?

Or this?

```
/* write to the Queue
  ADD MQREC;
IF MQREC NOT ERR;
MESSAGE = "Message written to
queue";
```

And how many times do you do it in a complex application?

EGL TUI applications

- EGL supports text oriented (TUI) applications
  - CUI: Windows, Linux, Unix
  - TUI: zOS, iSeries

```
MVIRGA   SF050P            ANYCITY PUBLIC SCHOOLS                    SF050M
Oct 12, 2006          SCHOOL ACCOUNTING SYSTEM - MAIN MENU         02:22:31 PM


                                  SCHOOL: 509 FINANCIAL SERVICES


          SELECTION                 DESCRIPTION
          ---------  ------------------------------------------
             ET      ENTER TRANSACTIONS
             QA      QUERY ACCOUNTS
             FD      FINANCE DEPARTMENT REPORTS
             FM      FILE MAINTENANCE
             PR      PRODUCE REPORTS
             CC      CLOSING CYCLE
              ?      HELP
              .      END SESSION
          ---------  ------------------------------------------

          SELECTION:


ENTER-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
ENTER             QUIT
```

EGL - HelloWorldCUI.egl - IBM Rational Software Development Platform

File Edit Navigate Search Project Run Window Help

Project Explo... | EGL Parts Ref...

- EGL Projects
  - HelloWorld
  - HelloWorldCUI
    - EGLSource
      - (default package)
        - HelloWorldCUI.egl
        - HelloWorldCUI.eglbld
    - JavaSource
      - (default package)
        - EzecustomerForm.java
        - HelloWorldCUI.java
        - runtunit.properties
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/itext-1.
    - EGL_PLUGINDIR/runtime/jasperre
    - ifxjdbc.jar - C:\ifmxjdbc_221JC6\
    - EGL_GENERATORS_PLUGINDIR/r
    - EGL_GENERATORS_PLUGINDIR/r

Outline

- customerForm : consoleForm
  - * : ConsoleField
  - * : ConsoleField
  - * : ConsoleField
  - * : ConsoleField
  - * : ConsoleField
  - * : ConsoleField
  - firstName : ConsoleField
  - lastName : ConsoleField
  - address : ConsoleField
  - city : ConsoleField
  - state : ConsoleField
  - zipCode : ConsoleField
- HelloWorldCUI : basicProgram
  - firstName : char(20)
  - lastName : char(20)
  - address : char(30)
  - city : char(30)
  - state : char(2)
  - zipCode : char(5)
  - main()
  - simple_form()
  - write_to_console()

HelloWorldCUI.egl

```
record customerForm type ConsoleForm { formSize = [25, 80], showBrackets = yes}

    *ConsoleField { position = [5,9], value = "First Name:" };
    *ConsoleField { position = [6,9], value = "Last Name :" };
    *ConsoleField { position = [7,9], value = "Address   :" };
    *ConsoleField { position = [8,9], value = "City      :" };
    *ConsoleField { position = [9,9], value = "State     :" };
    *ConsoleField { position = [10,9], value = "ZIP       :" };

    firstName ConsoleField { position = [5, 22], fieldlen = 20, datatype = "char" };
    lastName  ConsoleField { position = [6, 22], fieldlen = 20, datatype = "char", caseFormat = upper };
    address   ConsoleField { position = [7, 22], fieldlen = 30, datatype = "char" };
    city      ConsoleField { position = [8, 22], fieldlen = 30, datatype = "char" };
    state     ConsoleField { position = [9, 22], fieldlen = 2, datatype = "char" , autonext = yes};
    zipCode   ConsoleField { position = [10, 22], fieldlen = 5, datatype = "char", color = RED };

end

program HelloWorldCUI
    firstName char(20);
    lastName char(20);
    address char(30);
    city char(30);
    state char(2);
    zipCode char(5);

    function main()

        OpenUI new Menu( labelText= "Simple EGL CUI App",
        menuItems=[
            new MenuItem(Name= "SimpleForm",LabelText= "SimpleForm", C
            new MenuItem(Name= "WriteToConsole",LabelText= "WriteToCon
            new MenuItem(Name= "Exit",LabelText= "Exit", Comment= "Exi
                ] }
        onEvent(MENU_ACTION:"SimpleForm")
            simple_form();

        onEvent(MENU_ACTION:"WriteToConsole")
            write_to_console();

        onEvent(MENU_ACTION:"Exit")
            ClearWindow(ConsoleLib.screen);
            EXIT PROGRAM;
        end /* OpenUI menu */ ;
    end
```

Problems | Console | EGL SQL Errors | EGL Generation Results | Properties

HelloWorldCUI [Java Application] C:\RAD60\eclipse\jre\bin\javaw.exe (Apr 19, 2005 3:11:45 PM)

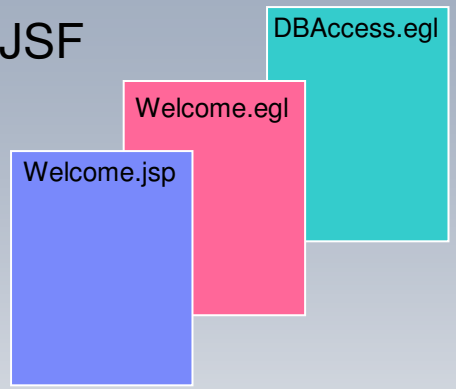
HelloWorldCUI.java - HelloWorldCUI/JavaSource

EGL Console Window

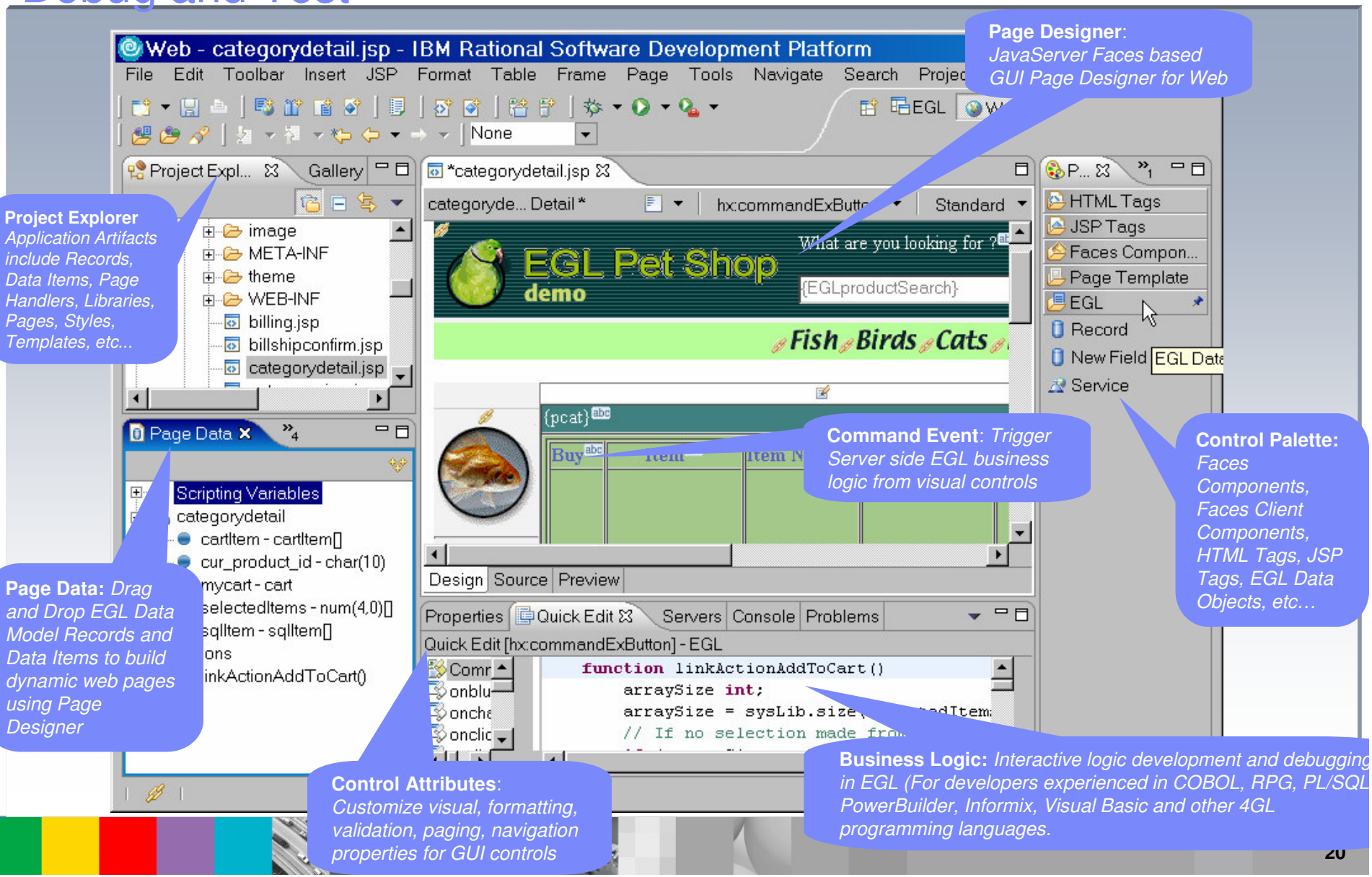
```
Simple EGL CUI App:  SimpleForm  WriteToConsole  Exit
Display a simple CUI Form

        First Name: [                    ]
        Last Name : [                    ]
        Address   : [                              ]
        City      : [                              ]
        State     : [  ]
        ZIP       : [     ]
```

EGL Web applications

- Seamles integration with JSP (Java Server pages) and JSF (Java Server Faces)
- Pagehandler code: 100% EGL
  - Contains functions and data related to a .jsp page
  - “onPageLoad()” function
  - Declare local data structures
  - Functions bound to command buttons
- Deployment support for WebSphere application server and Apache‘s Tomcat
- Model-View-Controller Architecture



Web Application Development w/ EGL – Design, Deploy, Debug and Test



Page Designer: *JavaServer Faces based GUI Page Designer for Web*

Project Explorer *Application Artifacts include Records, Data Items, Page Handlers, Libraries, Pages, Styles, Templates, etc...*

Command Event: *Trigger Server side EGL business logic from visual controls*

Control Palette: *Faces Components, Faces Client Components, HTML Tags, JSP Tags, EGL Data Objects, etc…*

Page Data: *Drag and Drop EGL Data Model Records and Data Items to build dynamic web pages using Page Designer*

Control Attributes: *Customize visual, formatting, validation, paging, navigation properties for GUI controls*

Business Logic: *Interactive logic development and debugging in EGL (For developers experienced in COBOL, RPG, PL/SQL PowerBuilder, Informix, Visual Basic and other 4GL programming languages.*

EGL Reporting (JasperReports)

- EGL has a builtin reporting API, based on JasperReports
- JasperReports: powerful OpenSource, Java based reporting engine
  - Multiple Output formats: Text (only in combination with EGL), PDF, HTML, XML, CSV
  - Callback function support
  - Complex sub-report and grouping functionality
  - http://jasperreports.sourceforge.net
- The JasperReports libraries are bundled with Rational SDP
- Jasper report design editors available
  - iReport – Standalone, OpenSource
  - JasperAssistant – Eclipse plugin, Commercial

Jasper Reporting Capabilities

EGL - allcustomers.jrxml - IBM Rational Software Development Platform

All Customers

Customer Number	Last Name	First Name	Company
$F{customer_num}	$F{lname}	$F{fname}	$F{company}

$V

Jasper Report Example

All Customers

Customer Number	Last Name	First Name	Company
101	Pauli	Ludwig	All Sports
102	Sadler	Carole	Sports Spot
103	Currie	Philip	Phil's Sports
104	Higgins	Anthony	Play Ball!
105	Vector	Raymond	Los Altos Sports
106	Watson	George	Watson & Son
107	Ream	Charles	Athletic Supplies
108	Quinn	Donald	Quinn's Sports
109	Miller	Jane	Sport Stuff
110	Jaeger	Roy	AA Athletics
111	Keyes	Frances	Sports Center
112	Lawson	Margaret	Runners &
113	Beatty	Lana	Sportstown
114	Albertson	Frank	Sporting Place
115	Grant	Alfred	Gold Medal

1

EGL Messaging / File Access

- EGL supports easy access to message queues and external files
  - Based on the EGL record concept
- EGL Messaging support based on WebSphere MQ
  - Allows easy integration with heterogeneous applications
- EGL file access
  - serialRecord (all operating systems)
  - indexedRecord, relativeRecord (VSAM access only – AIX and zOS)
- EGL relational database table export / import functions
  - sysLib.unloadTable()
  - sysLib.loadTable()

EGL - HelloWorldMQ.egl - IBM Rational Software Development Platform

File Edit Navigate Search Project Run Window Help

Project Explo... | EGL Parts Ref...

- EGL Projects
  - HelloWorld
  - HelloWorldCUI
  - HelloWorldMQ
    - EGLSource
      - (default package)
        - HelloWorldMQ.egl
      - HelloWorldMQ.eglbld
    - JavaSource
      - (default package)
        - EzemyMQRecord.java
        - HelloWorldMQ.java
      - rununit.properties
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/commor
    - EGL_PLUGINDIR/runtime/itext-1.
    - EGL_PLUGINDIR/runtime/jasperre
    - JRE System Library [eclipse]
    - bin

Outline

- myMQRecord : mqRecord
- HelloWorldMQ : basicProgram

HelloWorldCUI.egl | *HelloWorldMQ.egl

```
// basic program
//

Record myMQRecord type mqRecord {queueName="myqueue"}
    customer_num int;
    fname char(15);
    lname char(15);
end

program HelloWorldMQ

    function main()

        myMQ myMQRecord;

        myMQ.customer_num = 1000;
        myMQ.fname = "Kevin";
        myMQ.lname = "Koerner";

        add myMQ;

        close myMQ;

    end
end
```

Problems | Console | EGL SQL Errors | EGL Generation Results | Properties

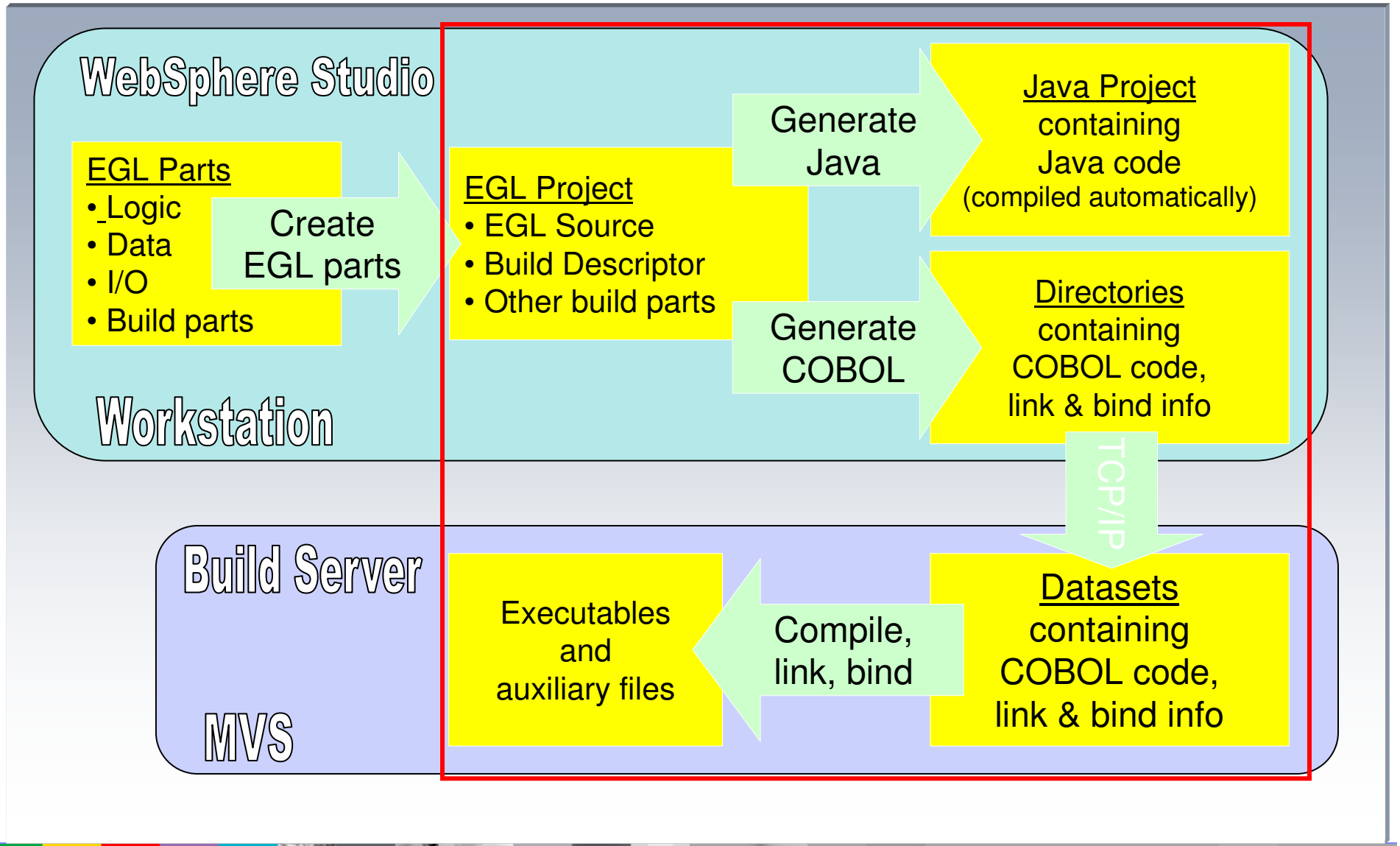
Generation Results

IWN.VAL.9994.i 1/1 The part HelloWorldMQ was generated using build descriptor HelloWorldMQJavaBuildOptions from file HelloWorldMQ/EGLSource/h
IWN.VAL.9996.i 1/1 Generation completed for HelloWorldMQ with no errors.

HelloWorld (HelloWorldJavaBuildOptions) | HelloWorldCUI (HelloWorldCUIJavaBuildOpt... | HelloWorldMQ (HelloWorldMQJavaBuildOpti...

HelloWorldMQ

EGL Generation Process Overview



EGL SOA and Web Services

- Interoperable, Simple, Composable
  - Hides technology issues of communication between disparate systems
  - Masks application complexity
  - Services can rely on other services or be composed into Modules (SCA)
- Just concentrate on building business logic
  - Simple and Intuitive Programming Model
  - Wizard Driven Development
  - Easy Testing of Generated WSDL Through Web Service Explorer
- Deployment information is separated from implementation; deploy as an EGL Service, Web service, or COBOL service

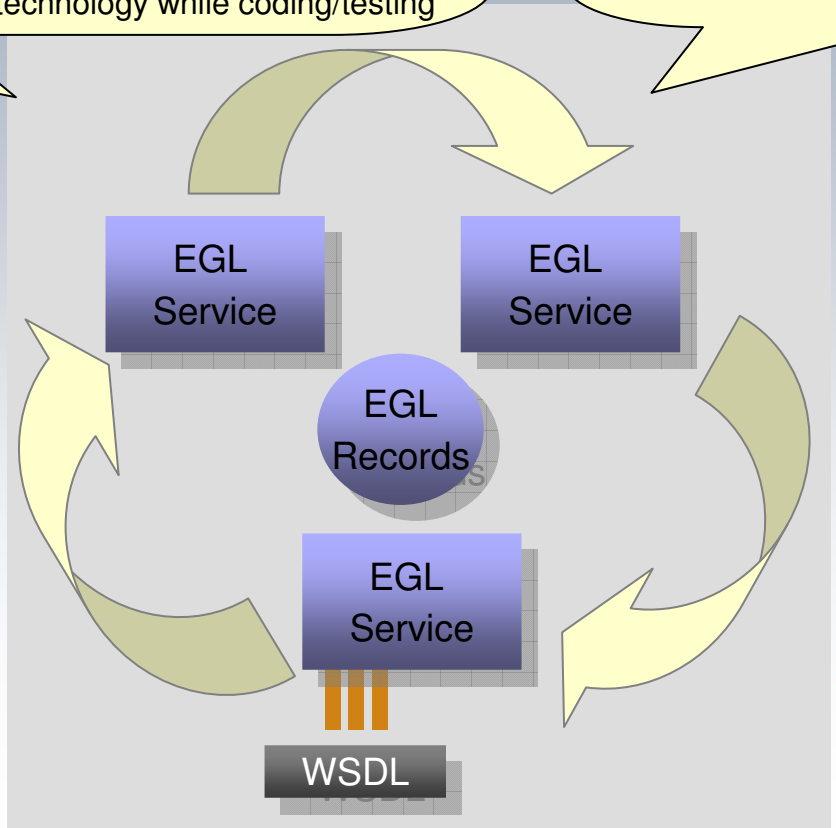
SOA and EGL: Building New Services with EGL

At development time…

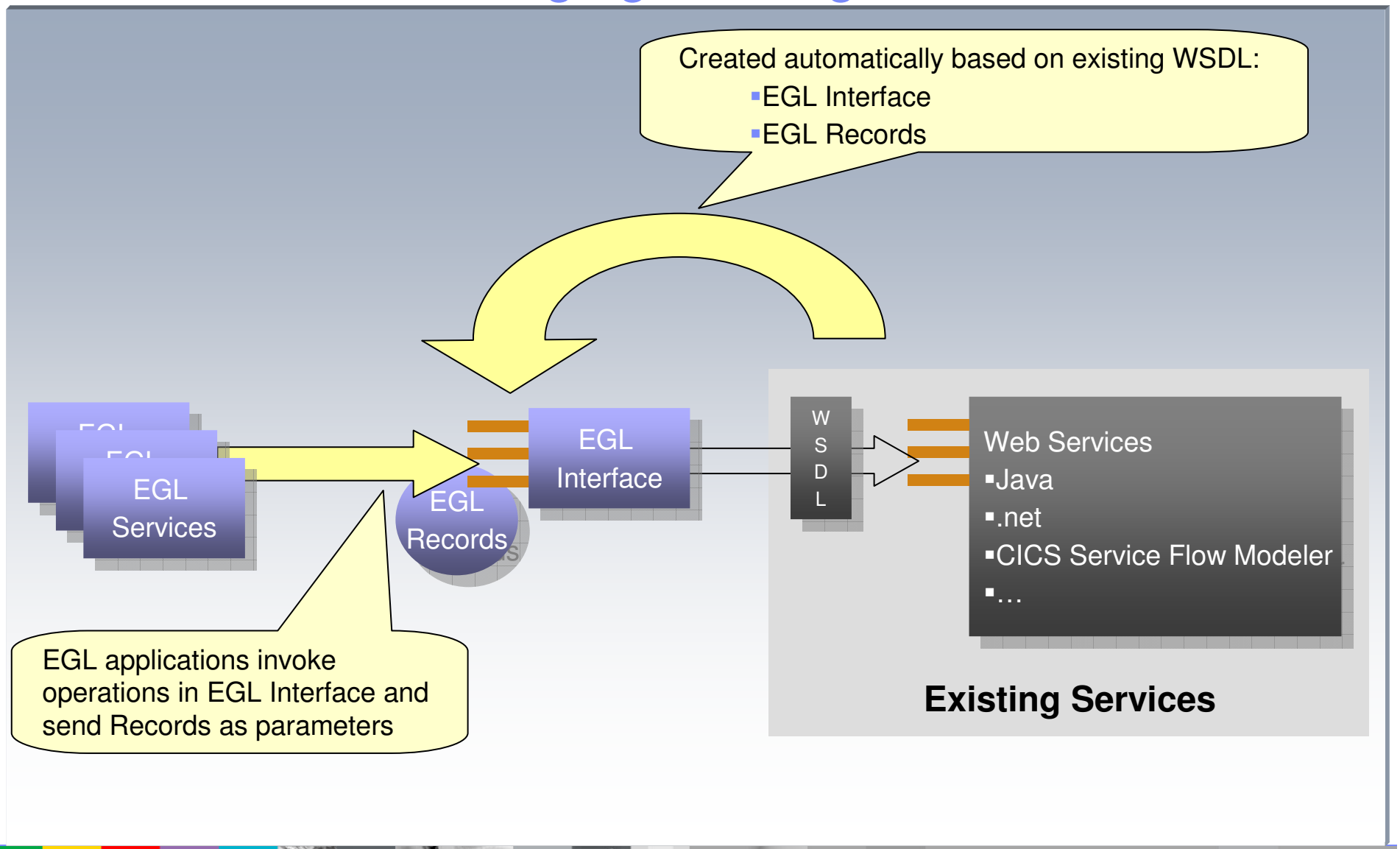
- Focus on the business logic
- Implement SOA design elements: Services & Interfaces
- Leverage existing COBOL or RPG developers for new SOA development
- Ignore deployment targets/technology while coding/testing

Deploy EGL Services…

- In Java to Java to WAS, Tomcat, Win, Linux, HP-UX, Solaris, iSeries
- In COBOL to CICS, IMS, iSeries (coming in 2H06)
- Expose as a Web Service or not…your choice

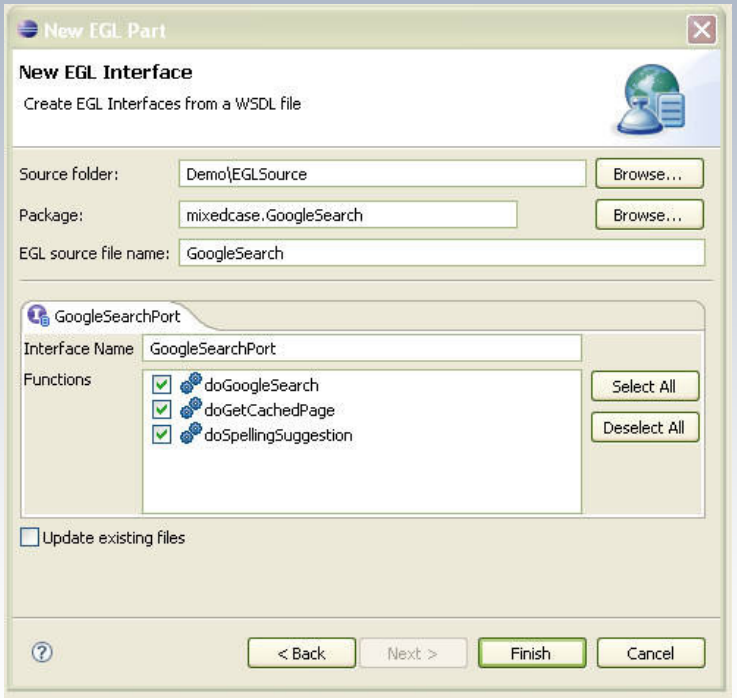


SOA and EGL: Leveraging Existing Services from EGL



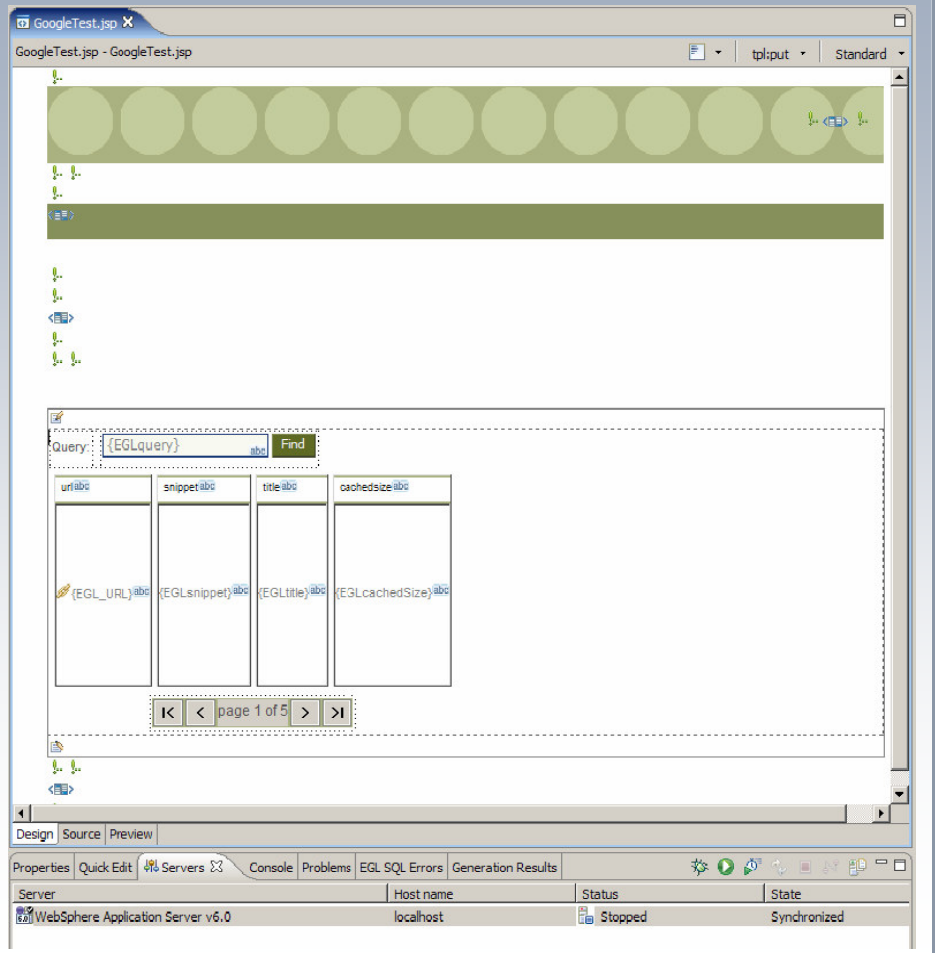
EGL and Web Services

- Ability to Consume Web Services from WSDL:
  - All Necessary Interfaces Generated From the WSDL File
  - Complex Types Automatically Created From WSDL File Content
  - Both EGL and External Web Service Consumption Possible



Consuming External Web Services

- Ability to Consume External WSDL Files to Create EGL Artifacts
- All Necessary Interfaces Generated From the WSDL File
- Communication Protocols Automatically Created In Service Binding Library (From WSDL)
- Complex Types Automatically Created From WSDL File Content
- Both EGL and External Web Service Creation Possible



Where to go from here

- Assuming that you are still interested in EGL, how can you find out more?
  - There is a three phased education road map for EGL:
    - First, do the QuickStart Tutorials (download from www.jsayles.com/ibm)
    - Second, do a Proof-Of-Concept for a customer or for yourself
    - Third, do a real application for your customer
  - Note: Let the EGL Ecosystem Team help you. We are here to make you successful through out each of these phases.
- And what about when the customer says, this sounds all well and good, but…?
  - Objection handling:

Objection Handling

- EGL related
  - Don't want to be tied to a proprietary language again
  - Want open, standard tools
  - Don't have the staff &/or skill set
- Migration/Conversion related
  - Don't have resources for migration
  - Too expensive
  - Takes too long
  - Code is unreliable

Objection Handling

- EGL related
  - Don’t want to be tied to anything like a 4GL again

    EGL is the continuation of more than 25 years of IBM R&D investment in rapid development technologies. EGL is IBM's business developer language. With the emergence of new computing models, such as web and SOA, IBM views a simplified programming approach as critical to customers success and has therefore invested significantly in this strategic solution and is fully behind it. The code EGL generates is fully standards compliant and open. EGL is the migration path for IBM Visual Age Generator and Informix 4GL customers and is also a migration choice for fourth generation languages (e.g. Natural, CA Ideal®, CA Cool:Gen, CA Cool: Enterprise®).

Objection Handling

- EGL related
  - Want open, standard tools

    Rational is in the process of creating an EGL Standards spec to be presented to the OMG Standards body **http://www.omg.org/** It is also the intent to Open Source parts of the EGL language to allow for easy integration and extension of EGL.

    In addition, all of the JAVA and COBOL generated by EGL is completely standards compliant.

Objection Handling

- EGL related
  - Don't have the staff &/or skill set
  - EGL has a very short learning curve
  - According to the Gartner Study, "A Model for Calculating the ROI of Crossing Over to Service-Oriented Development of Applications", it is fairly expensive to effectively train a business-oriented developer into a Java developer with a low success rate. There is an extensive network of consultants and IBM business partners that have a lot of expertise in this technology and can help you get your projects off the ground quickly. EGL is far easier to learn and master than Java. Your in-house developers can ramp up the required skills in a matter of weeks, eliminating the need to compete for hiring scarce Java skilled resources. Productivity with EGL is high as it is much easier for business-oriented developers to learn than Java.
  - Resources
    - EGL Zone on developerWorks
    - Enterprise Application Transformation: "At-a-Glance" Selling Guide
    - Enterprise Application Transformation XL Page
    - Sales Resources specific to customers using COBOL with EGL
    - EGL Family Page
    - EGL EcoSystems Team
      - Web-based training classes
      - On-site training/Consulting PoCs
    - RedBooks
    - IGS/AMS & Business Partners

Objection Handling

- Migration/Conversion related
  - Don't have resources for migration

IBM Services and IBM Business Partners are prepared to help you succeed. The same resources who have traditionally maintained these system can now be retrained in EGL and their domain expertise leveraged. This environment allows for easy maintenance of the migrated systems by the very developers who originally implemented them and gives these same developers the flexibility to use a modern, state-of-the-art IDE for business application development and SOA deployment.

Objection Handling

- Migration/Conversion related
  - Too expensive
  - Takes too long

Repeated analyst studies have indicated the viability and cost-effectiveness of Enterprise Application Transformation which leverages the huge investments already made in proven, production-ready code. A phased transformation project using automated tools takes weeks/months not years. It is the most cost-effective way to reach system equivalency.

Objection Handling

- Migration/Conversion related
  - Code is unreliable

    EGL's generated code is production-ready, standard-compliant code. EGL potentially generates higher quality Java and COBOL than hand-coded code as it generates much of the middle-ware layer of code...this is where many, many manual coding errors occur.

    Over the years, we've optimized the code that EGL generates, so that it is highly performant... many customers have built core-business applications with EGL.