



# *Protecting and leveraging your agency's assets through legacy re-engineering*

Presentation for:

***Federation of Tax Administrators –  
Technology Conference***

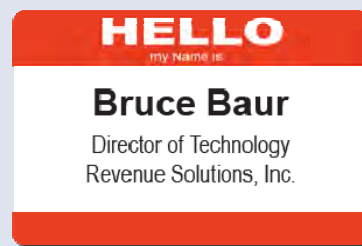
August 6, 2007

FTA Technology Conference 2007

## Introductions



- **Bruce Baur**
  - Director, Revenue Solutions, Inc.
  - Leads Technology Products and Services
  - Eighteen years as industry consultant
  - Both integrated tax system and compliance data warehouse experience



FTA Technology Conference 2007 2

# Legacy Re-engineering Defined



*“Reengineering [Legacy Modernization] is concerned with the **safe, risk-free, and, above all, rapid transition of a legacy system to an open platform, the preservation of the organization's assets wherever possible, and the elimination of technical risk to the organization by eliminating its dependence on proprietary or obsolete technologies. Reengineering efforts are rapid due to their reliance on tools to automate the process and to ensure the consistency of the resulting code.**”*

Researchers from the University of Edinburgh and Carnegie Mellon University

## Re-engineering Focus



- Focused on Technical Transformation
  - Hardware, Operating System, Programming Language
  - Transformation from one platform to another
  - Modern Target Environment
  
- Software and Methodology and Tools
  
- Open Standards


# Major Types of Re-engineering



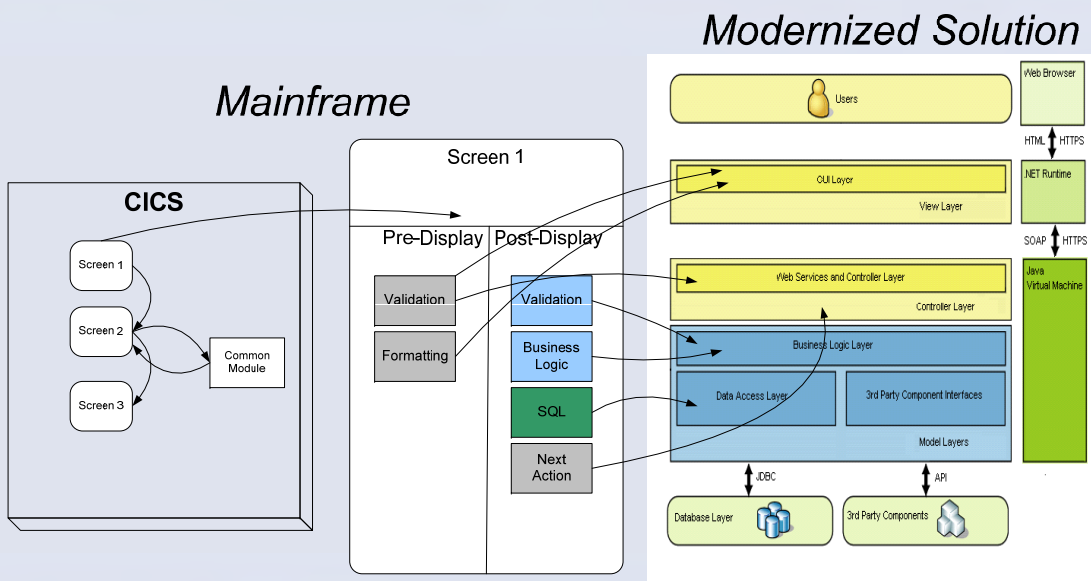
- **Software Translation**
  - Scan Source Code
  - Build Repository of Logic/Definition/Interaction
  - Generate New Language Version of each Program
- **Service Enablement**
  - Scan Source Code
  - Create Program Interface APIs
  - Provide Services Layer to Call
- **Business Rule Mining**
  - Scan Source Code
  - Build Repository of Key Logic / IF statements
  - Provide Ability to Generate Application Snippets

# Examples



- COBOL
  - Mainframe
  - CICS
  - Visual Basic
  - IMS
  - IDMS
  - DB2
  - ADABAS/Natural
  - DEC/VAX
  - Unisys
  - Bull
- 
- Sun Solaris
  - AIX
  - Windows
  - Intel
  - Java/J2EE
  - ASP.NET, C#
  - SOA Web Services
  - XML
  - Oracle
  - MS SQL Server
  - UDB

# Example Migration



# Example COBOL to Java



```

BEFORE.txt - Notepad
File Edit Format View Help
IF LOAN-LENGTH > 4 THEN
  COMPUTE START-YEAR = LOAN-LENGTH - 4
ELSE
  COMPUTE START-YEAR = 0
END-IF.

IF INTEREST-RATE > 0.4 THEN
  COMPUTE START-INTEREST = INTEREST-RATE - 0.4
ELSE
  COMPUTE START-INTEREST = 0
END-IF.

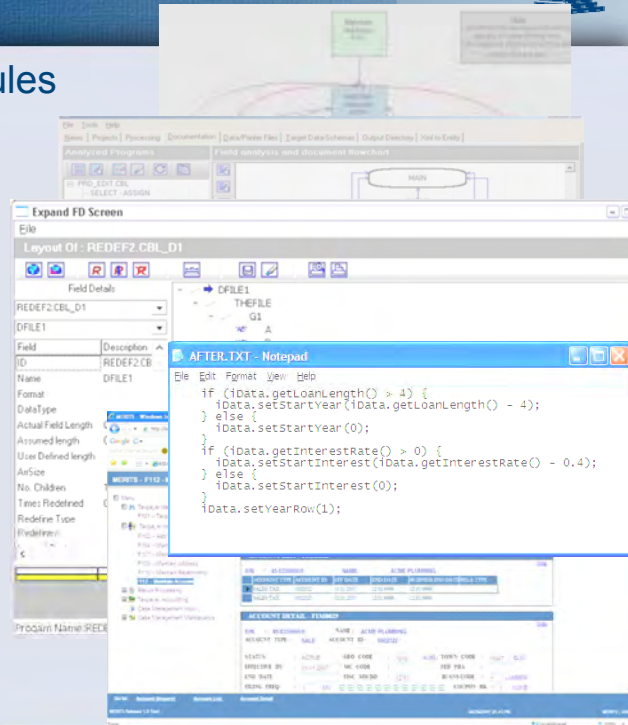
MOVE 1 TO YEAR-R

AFTER.TXT - Notepad
File Edit Format View Help
if (iData.getLoanLength() > 4) {
  iData.setStartYear(iData.getLoanLength() - 4);
} else {
  iData.setstartyear(0);
}
if (iData.getInterestRate() > 0) {
  iData.setStartInterest(iData.getInterestRate() - 0.4);
} else {
  iData.setStartInterest(0);
}
iData.setYearRow(1);
    
```

## Possible Extracted Artifacts



- Inventory of Business Rules
- Program Flow Charts
- Data Dictionary
- Documentation from Comments
- Executable Program Code
- Migrated Database, Data Access Layer
- Graphical User Interface
- Message-based APIs
- Dead Code Inventory



## Why Legacy Reengineering?



- IT industry is increasingly embracing the benefits of the legacy reengineering approach.
- Low-risk, cost-effective and rapid transition of a legacy system to a modernized open platform.
- Rather than engaging in the additional cost, risk and organizational disruption associated with the implementation of a new system.
- Reengineering can provide agencies the opportunity to modernize and extend the life of business applications without the added burden of replacing all aspects of their business application logic.

## Why Legacy Reengineering? (cont'd)

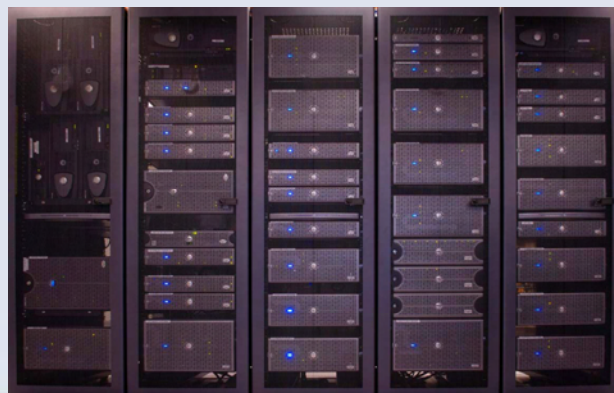


- Legacy Re-engineering provides agencies both short term and long term modernization options:
  - Short term - can be used to quickly eliminate dependence on proprietary systems
  - Long term - beginning of a trend of continuous improvement - away from wholesale replacement every 10 - 15 years
- Allows an agency to focus on moving forward as opposed to spending significant time/energy/cost on ensuring that you don't lose current capability
- Enables an agency to integrate best of breed products

## Modernization Approaches



- Custom
- COTS
- Re-engineering



# Approach Comparison

## Custom – Advantages/Disadvantages



### Custom System Approach

#### Advantages

- ✓ Likely Allows Agency to Own/Maintain Application (i.e., Source Code)
- ✓ Apply Lessons Learned from past implementations into New System
- ✓ Greater Agency control over resulting system
- ✓ Can Reduce functionality gaps

#### Disadvantages

- ✓ Completely New System - Training and May Require Organizational Change
- ✓ Requires Many User and IT Resources for Multiple Years
- ✓ Data Usually Converted at Summary Level *or* Risky Conversion Effort Required
- ✓ Highest Cost, Longest Schedule and Greatest Risk Option

# Approach Comparison

## COTS – Advantages/Disadvantages



### COTS Approach

#### Advantages

- ✓ Usually Based on Proven Methodology
- ✓ Generally Medium Risk
- ✓ System Upgrades and Support Included with Maintenance Contract
- ✓ Implementation Schedule Faster Than Custom Solution

#### Disadvantages

- ✓ Requires Organization to Adapt to COTS Model or Pay for Customization
- ✓ Reliance on Vendor to Support and Enhance – No Source
- ✓ Minimal offerings in the market
- ✓ Data Usually Converted at Summary Level *or* Risky Data Conversion Effort
- ✓ Higher Cost and Greater Risk than Legacy Modernization

# Approach Comparison

## Legacy Modernization – Advantages/Disadvantages



### Legacy Modernization Approach

#### Advantages

- ✓ Leverages and Extends Investment in Legacy System, Data and Customizations
- ✓ Lower Risk – Based on Proven Methodology and Tools
- ✓ Minimizes Impact on Organization
- ✓ Lowest Cost Modernization Approach
- ✓ Side by Side Testing Ability
- ✓ Shortest Implementation Timeframe

#### Disadvantages

- ✓ Modernized Application similar in structure to Legacy Application – i.e. not pure object oriented
- ✓ Minimal Increased Functionality

# Re-engineering Approach Criteria

## Legacy Re-engineering - Making the Business Case

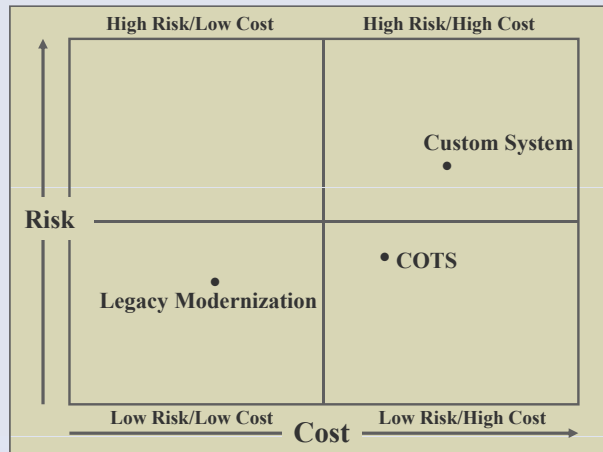


Existing System Supported in House and Generally Supports the Agency's Business Functions (i.e., Small % of "Future State" Requirements are Not Met by Current System)	
System Replacement is Primarily Driven by Technology Constraints/Limitations (e.g. COBOL/IDMS), Cost & Long-Term Maintenance Issues	
Looking to Reduce Risk and Organizational Impact of the Modernization Effort	
Agency maintenance strategy is to support system in-house	

## Risk vs. Cost



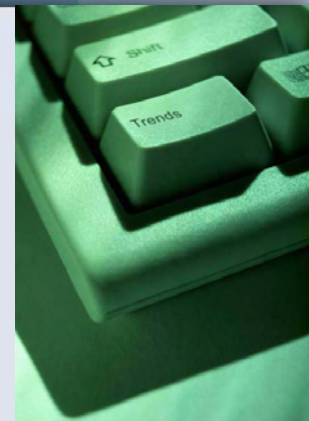
- Risk vs. Cost Comparison of Approaches to Modernization



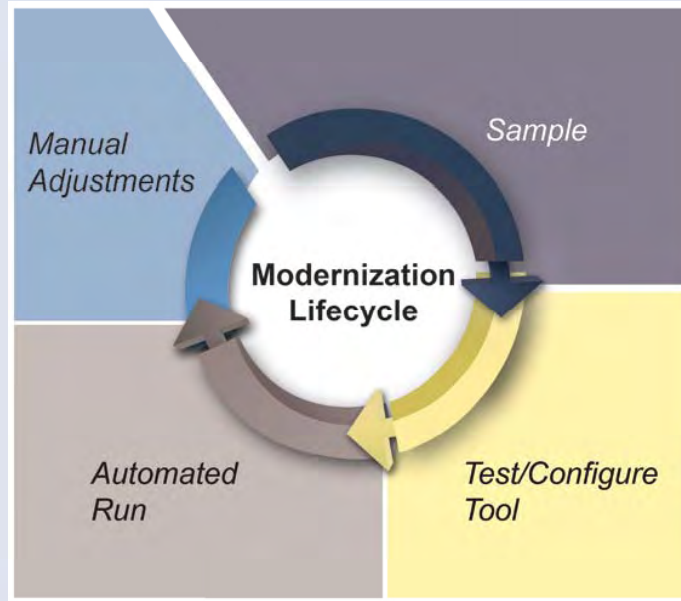
## Industry Observations



- Seeing more and more
  - Webinar, forums, industry groups
- Private Sector
  - Financial Services, Others
- Governments embracing
  - US Government, UK, Chile, Far East
- RSI currently performing re-engineering
  - Mainframe CICS COBOL DB2 – Web Services Java Oracle
  - State of Maine – Revenue Services



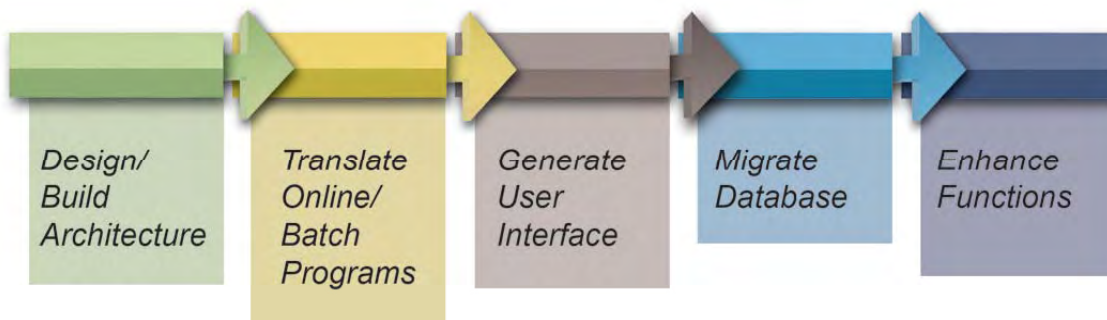
# Translation Process



# Modernization Progression



## Modernization Progression



## Conclusion



- Legacy Re-engineering offers a lower cost – lower risk alternative to COTS or Custom solutions
- Modern toolkits allow for highly automated translation of legacy application code to modern languages
- Technology skills such as COBOL/IDMS are becoming scarce and are no longer being taught in colleges
- Increased pressure to open up web services to external stakeholders for information interchange
- Agencies looking to replace existing systems should consider Legacy Re-engineering as a viable modernization option

## Questions and Answers



## Contact Information

