Guaranteed Value Projects in Legacy Modernization

Michael Oara
VP for R&D
Relativity Technologies, Inc.
Business Problem

**Mainframe Applications – Key Attributes**

- Millions of lines of code, poorly documented
- Typically 10 – 30+ years old
- Complex, brittle, difficult to enhance
- Maintained by antiquated tools
- Drive core business processes
- Issues with regulatory compliance
Global 1000 IT Environment

60% of enterprise applications will remain on high-end mainframes through 2010

- **Reality:** Mainframes worldwide contain >200 billion lines of COBOL Code (*IBM*)
- **Complexity:** limited understanding of applications
  - Outsourcing – increases transition time and costs
  - Risk – increased
  - Regulatory compliance – more difficult and costly
- 60-80% of IT budgets go to maintaining existing systems*
  - Time consuming to make changes
  - Risk of destabilizing system
  - Difficult to train new team members
- Complexity reduces flexibility and increases costs
Causes of complexity

**Causes:**
- Cumulative effect of years of changes
- Inadequate documentation
- Technical complexities stem from:
  - Inadequate tools
  - Non-standard code
  - Large and complex programs
Modernization Projects

- Modernization projects are:
  - Costly
  - Time consuming
  - Risky

- Companies hesitate to make large investments for risky projects

- Cost overruns may lead to project cancellation
Solution

- Create a modernization path with incremental steps
- Insure that every phase ...
  - delivers clear value
  - serves as its own justification, independent of the next steps
- Four major types of modernization projects...
  - Documentation
  - Clean-up
  - Restructuring
  - Migration
## ROI for Modernization Projects

<table>
<thead>
<tr>
<th>Activity</th>
<th>% Time consumed</th>
<th>% Effect of automation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis</td>
<td>15%</td>
<td>30%</td>
</tr>
<tr>
<td>Detailed design</td>
<td>10%</td>
<td>30%</td>
</tr>
<tr>
<td>Coding</td>
<td>15%</td>
<td>5%</td>
</tr>
<tr>
<td>Unit testing</td>
<td>20%</td>
<td>15%</td>
</tr>
<tr>
<td>System testing</td>
<td>25%</td>
<td>5%</td>
</tr>
<tr>
<td>Deployment</td>
<td>10%</td>
<td>5%</td>
</tr>
<tr>
<td>Documentation</td>
<td>5%</td>
<td>75%</td>
</tr>
</tbody>
</table>
Legacy Documentation

- Application statistics
- Code defects
- Application structure
- Business rules
# Documentation: Application statistics

## Statistics

- Volume measurements
  - By LOC
  - By cyclomatic complexity
- Volume reports
  - By application, module
  - By supporting technology
- By access to data
  - Interfaces
  - Databases

## Benefits

- Measure team productivity
- Measure quality
- Estimate change costs
- Remove obsolete technology
- Determine interdependencies
- Determine costs of modernizing database
## Application Analyzer: Cyclomatic Complexity Report

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Executable State</th>
<th>Operators</th>
<th>Operands</th>
<th>Vocabulary</th>
<th>Program Vo.</th>
<th>Complex.</th>
<th>Development Time</th>
<th>Cyclomatic Complexity</th>
<th>Est.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DE421</td>
<td>Program</td>
<td>3747</td>
<td>5229</td>
<td>14341</td>
<td>8128</td>
<td>287117.30</td>
<td>0.06</td>
<td>243380.60</td>
<td>117.66</td>
<td>2540</td>
</tr>
<tr>
<td>EX4001</td>
<td>Program</td>
<td>4599</td>
<td>5774</td>
<td>15462</td>
<td>8008</td>
<td>288329.30</td>
<td>0.05</td>
<td>264529.00</td>
<td>1429</td>
<td>2359</td>
</tr>
<tr>
<td>DE422</td>
<td>Program</td>
<td>3227</td>
<td>5147</td>
<td>12050</td>
<td>7188</td>
<td>220317.20</td>
<td>0.06</td>
<td>195446.50</td>
<td>1254</td>
<td>2249</td>
</tr>
<tr>
<td>DE4241</td>
<td>Program</td>
<td>2320</td>
<td>6442</td>
<td>9474</td>
<td>5049</td>
<td>173652.00</td>
<td>0.07</td>
<td>145279.30</td>
<td>1172</td>
<td>1865</td>
</tr>
<tr>
<td>W8P5W8</td>
<td>Program</td>
<td>1826</td>
<td>3871</td>
<td>8810</td>
<td>4149</td>
<td>152407.20</td>
<td>0.09</td>
<td>90112.17</td>
<td>1154</td>
<td>1520</td>
</tr>
<tr>
<td>BL0412</td>
<td>Program</td>
<td>1620</td>
<td>4059</td>
<td>17316</td>
<td>11954</td>
<td>237662.40</td>
<td>0.10</td>
<td>173956.90</td>
<td>1029</td>
<td>2053</td>
</tr>
<tr>
<td>W8F3W8</td>
<td>Program</td>
<td>1537</td>
<td>3405</td>
<td>6150</td>
<td>3697</td>
<td>137877.30</td>
<td>0.10</td>
<td>80033.39</td>
<td>961</td>
<td>1632</td>
</tr>
<tr>
<td>DE4208</td>
<td>Program</td>
<td>1877</td>
<td>3255</td>
<td>8803</td>
<td>5913</td>
<td>151082.00</td>
<td>0.07</td>
<td>113093.10</td>
<td>862</td>
<td>1466</td>
</tr>
<tr>
<td>DE4203</td>
<td>Program</td>
<td>1710</td>
<td>2879</td>
<td>6375</td>
<td>3462</td>
<td>108802.90</td>
<td>0.07</td>
<td>93843.24</td>
<td>786</td>
<td>1597</td>
</tr>
<tr>
<td>W8B330</td>
<td>Program</td>
<td>4261</td>
<td>3735</td>
<td>13860</td>
<td>5980</td>
<td>248966.90</td>
<td>0.06</td>
<td>244557.70</td>
<td>681</td>
<td>939</td>
</tr>
<tr>
<td>W8BT3L</td>
<td>Program</td>
<td>1904</td>
<td>2744</td>
<td>5776</td>
<td>2726</td>
<td>97225.32</td>
<td>0.07</td>
<td>74796.62</td>
<td>660</td>
<td>1230</td>
</tr>
<tr>
<td>DE4201</td>
<td>Program</td>
<td>2127</td>
<td>3323</td>
<td>6030</td>
<td>3664</td>
<td>111442.40</td>
<td>0.07</td>
<td>82095.72</td>
<td>650</td>
<td>1022</td>
</tr>
<tr>
<td>DE4000</td>
<td>Program</td>
<td>2289</td>
<td>3316</td>
<td>7118</td>
<td>3003</td>
<td>124000.00</td>
<td>0.07</td>
<td>103662.00</td>
<td>640</td>
<td>1016</td>
</tr>
<tr>
<td>BL0381</td>
<td>Program</td>
<td>308</td>
<td>1932</td>
<td>9078</td>
<td>5430</td>
<td>135902.90</td>
<td>0.12</td>
<td>95997.68</td>
<td>536</td>
<td>1188</td>
</tr>
<tr>
<td>DE4205</td>
<td>Program</td>
<td>1223</td>
<td>3007</td>
<td>4695</td>
<td>2548</td>
<td>74569.47</td>
<td>0.07</td>
<td>90306.30</td>
<td>516</td>
<td>1114</td>
</tr>
<tr>
<td>BL0493</td>
<td>Program</td>
<td>936</td>
<td>1840</td>
<td>7712</td>
<td>5028</td>
<td>118705.60</td>
<td>0.10</td>
<td>64592.20</td>
<td>475</td>
<td>768</td>
</tr>
<tr>
<td>DE4204</td>
<td>Program</td>
<td>1102</td>
<td>1937</td>
<td>3748</td>
<td>2122</td>
<td>62494.59</td>
<td>0.06</td>
<td>95662.95</td>
<td>469</td>
<td>693</td>
</tr>
<tr>
<td>DE4002</td>
<td>Program</td>
<td>1920</td>
<td>2640</td>
<td>5446</td>
<td>2726</td>
<td>92252.99</td>
<td>0.07</td>
<td>77182.31</td>
<td>463</td>
<td>569</td>
</tr>
<tr>
<td>W8W020</td>
<td>Program</td>
<td>3185</td>
<td>4101</td>
<td>9038</td>
<td>4656</td>
<td>159727.10</td>
<td>0.06</td>
<td>141012.30</td>
<td>462</td>
<td>611</td>
</tr>
<tr>
<td>DE4223</td>
<td>Program</td>
<td>1230</td>
<td>2102</td>
<td>4629</td>
<td>2619</td>
<td>77253.38</td>
<td>0.07</td>
<td>90896.96</td>
<td>457</td>
<td>670</td>
</tr>
</tbody>
</table>

*Note: The table details include various metrics such as executable state, operators, operands, vocabulary, program variable, complexity, development time, and cyclomatic complexity.*
# Documentation: Application structure

## Documents
- Call maps
- Data access
- Data flows
- Online flows
- Application interfaces

## Benefits
Help to determine:
- Program change impact
- Data change impact
- Batch outage consequences
- Use case scenarios
- Testing scenarios
- Side effects of application replacement
Application Profiler: Call Map Diagrammer
Application Analyzer:
Global Data Flow Analysis
Application Profiler: Reference Reports

**Reference Reports**

### Table 1: Unreferred Report

<table>
<thead>
<tr>
<th>Object Name</th>
<th>Object Type</th>
<th>Legacy Object</th>
<th>Source</th>
<th>Object Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORDSET1.CPY</td>
<td>Copybook File</td>
<td>ORDSET1.CPY</td>
<td>Source/Copybook/ORDSET1.CPY</td>
<td></td>
</tr>
<tr>
<td>DECISIONS</td>
<td>Program</td>
<td>DECISIONS.CBL</td>
<td>Source/Cebel DECISIONS.CBL</td>
<td></td>
</tr>
<tr>
<td>PRODU0000</td>
<td>Program</td>
<td>PRODU0000.cop</td>
<td>Source/Cebel PRODU0000.cop</td>
<td></td>
</tr>
<tr>
<td>FAAPLIPI</td>
<td>Program Entry Point</td>
<td>FAAPLIPI.cop</td>
<td>Source/Cebel FAAPLIPI.cop</td>
<td></td>
</tr>
<tr>
<td>DECISIONS</td>
<td>Program Entry Point</td>
<td>DECISIONS.CBL</td>
<td>Source/Cebel DECISIONS.CBL</td>
<td></td>
</tr>
<tr>
<td>FAAPLIPI</td>
<td>Program Entry Point</td>
<td>FAAPLIPI.cop</td>
<td>Source/Cebel FAAPLIPI.cop</td>
<td></td>
</tr>
<tr>
<td>PRODFILL</td>
<td>Program Entry Point</td>
<td>PRODFILL.cop</td>
<td>Source/Cebel PRODFILL.cop</td>
<td></td>
</tr>
<tr>
<td>PRODU0000</td>
<td>Program Entry Point</td>
<td>PRODU0000.cop</td>
<td>Source/Cebel PRODU0000.cop</td>
<td></td>
</tr>
<tr>
<td>ORDERERR1</td>
<td>Program Entry Point</td>
<td>ORDERERR1.cop</td>
<td>Source/Cebel ORDERERR1.cop</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2: Unresolved Reference Report

<table>
<thead>
<tr>
<th>Object Name</th>
<th>Object Type</th>
<th>Referred by</th>
<th>Referring Object Type</th>
<th>Relationship</th>
<th>Object Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAAAO504</td>
<td>Program Entry Point</td>
<td>FAAPLIPI</td>
<td>Program</td>
<td>Xref</td>
<td></td>
</tr>
<tr>
<td>DATEFMT</td>
<td>Program Entry Point</td>
<td>AR100</td>
<td>Program</td>
<td>Cols</td>
<td></td>
</tr>
<tr>
<td>DATEFMT</td>
<td>Program Entry Point</td>
<td>AR200</td>
<td>Program</td>
<td>Cols</td>
<td></td>
</tr>
<tr>
<td>DATEFMT</td>
<td>Program Entry Point</td>
<td>AR299</td>
<td>Program</td>
<td>Cols</td>
<td></td>
</tr>
<tr>
<td>INV3000</td>
<td>Program Entry Point</td>
<td>DE0001RN</td>
<td>Job</td>
<td>Runs</td>
<td></td>
</tr>
<tr>
<td>INV2000</td>
<td>Program Entry Point</td>
<td>DE0001RN</td>
<td>Job</td>
<td>Runs</td>
<td></td>
</tr>
<tr>
<td>TOTALRPT</td>
<td>Program Entry Point</td>
<td>PRODU0000</td>
<td>Program</td>
<td>Cols</td>
<td></td>
</tr>
<tr>
<td>RPTA</td>
<td>Program Entry Point</td>
<td>PRODU0000</td>
<td>Program</td>
<td>Cols</td>
<td></td>
</tr>
<tr>
<td>RPTI</td>
<td>Program Entry Point</td>
<td>PRODU0000</td>
<td>Program</td>
<td>Cols</td>
<td></td>
</tr>
<tr>
<td>PS_ID_C</td>
<td>Table</td>
<td>INTEDIT</td>
<td>Program</td>
<td>ReadTable</td>
<td></td>
</tr>
</tbody>
</table>
Application Analyzer:
Batch Application Viewer
Documentation: Code defects

Code defect: A departure from internal standards and best architecture and code practices – not to be confused with “bugs”

- Code defects result in…
  - Frequent outages
  - High maintenance costs
  - Difficulties in applying enhancements and changes
Definition
A statement expressing a policy or condition that governs business actions and establishes data integrity guidelines. A business rule statement must always be true.

Caution: A correct definition is needed for the purposes of any project using business rules
Not to be confused with flow, process or implementation
Business rules are usually hidden deep in the code, hence the need to discover them
Application Profiler:
Business Rules
Business Rule Manager: Business Rule Management

[Image of a computer screen showing a code editor with code snippets and a dialog box]
Documentation: Business Rules (2)

- Classification (many classifications are possible)
  - Validation (constraints on inputs)
  - Computation (of outputs)
  - Triggers
- Automatic detection is possible to a certain degree (30% - 95%)
Business Rule Manager:
Business Rule Discovery
Attributes:

- Business description
- Classification (validation, trigger, …)
- Functional area (customer, product, …)
- Pointer to implementation code
- Inputs and outputs
- Status (valid, approved, obsolete, …)
Documentation: Business Rules (4)

- Documenting the application
- Change management
  
  🔸 User or Business Analyst has partial access
- Gap analysis
- Specification for application migration to a new technology implementation
Business Rule Manager: Open for the business community

![Business Rule Manager Diagram](image-url)
Code cleanup and restructuring

- Cleanup:
  - Source elimination
  - Dead code elimination
  - Superfluous code elimination
  - Obsolete code elimination

- Restructuring:
  - Modularization
  - Application partitioning
Code cleanup: Source elimination

Eliminate:
- Unrefered programs
- Orphans
- Obsolete reports

Methodology:
- Review user requirements
- Create cross-references
- Eliminate unrefered objects in multiple iterations
Code cleanup: Dead code elimination

Dead code: Program code, in an otherwise functional program, which can never be reached because of logical conditions.

Why dead code?
- Programmer mistakes
- Application maintenance
- Package customization
- Debugging

Troubles:
- Complicates maintenance
- Distorts program measurements

Can be automated

Typically we find between 3% and 6% dead code in old legacy applications
Code cleanup: Superfluous code elimination

Superfluous code: Program code, in an otherwise functional program, which does not produce any results in program output.

- Why superfluous code?
  - Incomplete maintenance
  - Reduction in program output
- Can be automated
Code cleanup: Obsolete code elimination

Dead code: Program code, in an otherwise functional program, which can never be reached because of nature of data.

- Why obsolete code?
  - Business conditions change
  - Technology change (response conditions)
- Can be automated
Modularization

Definition: Break large programs into smaller programs, which together deliver the same functionality

Benefits:
- Increase code reuse
- Prepare for user interface change
- Web enablement

Can be automated
Application partitioning

- Definition: Break a large application into sub-applications which together deliver the same
- It is mainly for management purposes
- Applications:
  - Establish migration or change progression
  - Distribute the application between teams
Migration: Data

- Most common projects: non-RDBMS to RDBMS migration

Caveats:
- Clean up data before migration
- Do not allow radical changes in data structures
- Document data (example: what does FILLER1 mean?)
Migration: Interface

- Most common: Web enablement
- Simple solution: screen scrapping
- Advanced solution…
  - Break programs into reusable components
  - Reconstruct the user interface
  - Build communication infrastructure
  - Reassemble mainframe components into callable programs
Migration: Code

- Basic solution
  - Completely manual conversion (documentation is useful!!!)
  - Business rules based
  - Partial automation (extract smaller components which may be automatically translated)
  - Massive automation – works only in special cases.
The Modernization Workbench™

The Modernization Workbench™ drives down the cost and accelerates the modernization, management, and maintenance of business-critical enterprise applications.

- **Step 1**: Application source code is added into the Modernization Workbench™
- **Step 2**: The source code is parsed at a deep level
- **Step 3**: An application knowledgebase is created showing all programmatic interrelationships
- **Step 4**: Visually-rich HTML-based documentation, flow charts and diagrams are generated
- **Step 5**: Previously hidden information about the application, like “dead code”, is revealed
- **Step 6**: The knowledgebase is refreshed to keep all documentation and information current for use with the modernization suite
The Modernization Workbench™ drives down the cost and accelerates the modernization, management, and maintenance of business-critical enterprise applications.
Application Knowledgebase and Repository

Multi Environment Repository

- Multiple environments can be simultaneously accessed by users
- 3rd party parsers are efficiently integrated
  - Assembler
  - OpenROAD
  - RPG
  - Others

Natural Parser
Cobol Parser
PL/I Parser
3rd Party Parsers
Modernization Workbench: Enterprise Architecture

Deep coverage of the numerous enterprise environments

Residing off the mainframe minimizes security, cost, and risk issues

Individual users can execute tasks based on the latest source code

Remote users (e.g. other offices, or outsourcers) can access current information via the Web

Users focused on specific projects may operate in an ‘occasionally connected’ environment
# Modernization Workbench Modules Usage

<table>
<thead>
<tr>
<th>Module</th>
<th>Developer</th>
<th>Systems Analyst</th>
<th>Business Analyst</th>
<th>Support</th>
<th>QA</th>
<th>Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Analyzer</td>
<td>●</td>
<td>●</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Application Profiler</td>
<td>○</td>
<td>○</td>
<td>●</td>
<td>●</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Business Rule Manager</td>
<td>●</td>
<td>○</td>
<td>●</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Application Architect</td>
<td>●</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Transformation Assistant</td>
<td>●</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

### Key

- ○ Limited or no likely use
- ○ Partial use likely
- ● Heavy use likely
Application Analyzer™: Features and Benefits

- **Application Portfolio Management and Assessments**
  - Inventory reports give a total snapshot of your portfolio
  - Complexity analysis and estimation allow managers to understand where to direct resources
  - Impact analysis identifies the effect of changes
  - Powerful querying tools deliver deep insight

- **Visual, Interactive documentation and deep insight**
  - Multiple, interactive views illustrate programmatic and information flows
  - Users always have access to the most current information about your application portfolio

- **Team-centric and customizable**
  - Reports and diagrams can easily be shared
  - Application visualization is highly customizable
  - Analysis tools can be tailored to suit requirements
Application Analyzer: Call Map Diagrammer
Application Profiler™: Features and Benefits

- **Accessibility and usability**
  - Information **bottlenecks are removed**
  - **No additional software** simplifies management
  - **Intuitive interface** accelerates user productivity
  - **Dynamic access** delivers up-to-date information

- **Efficient application portfolio assessments**
  - System documentation is **easy to retrieve**
  - Powerful diagrams illustrate **inter- and intra-program relationships**
  - **Interactive and synchronized views** allow for efficient portfolio assessments

- **Streamlined application management**
  - Analysts can **audit and control** business logic
  - Managers gain a complete **overview** of their systems
  - Complexity analysis helps to **focus resources**
  - **Impact analysis** allows analysts to control the effect of changes to applications
Application Profiler:
Screen Shots

Complexity report

Reference reports

Business rule management

Call map diagrammer
Business Rule Manager™: Features and Benefits

- **Accelerated business rule identification**
  - Sophisticated tools help to quickly identify rules
  - Powerful interrogation simplify manual searches
  - Creation of rules directly from a search screen accelerates the collection process

- **Powerful business rule management**
  - Persistence ensures that rules are not lost as programs change
  - Rules Filter helps to focus business rule searches
  - Analysts can categorize and describe their portfolio of business rules, simplifying ongoing usage
  - Customizable interface allows analysts to efficiently manage their business rules portfolio

- **Integrated approach magnifies benefits**
  - Browser-based access allows users to remotely search, audit, and modify the latest business logic
  - Rich diagramming and search functions help users to focus searches
  - Convenient reports help analysts to plan, manage, and share business rules
Business Rule Manager: Screen Shots

Business rule discovery

Business rule management

Integration with Application Profiler
Application Architect™: Features and Benefits

- **More efficient operations through componentization**
  - Multiple componentization tools enable the creation of more reusable and maintainable programs
  - Examines all dependencies to ensure that the extraction is a functionally complete component
  - Coverage Report identifies additional opportunities and ensures completeness

- **Reduced complexity with application renovation**
  - Reduces complexity by partitioning business logic, data access, and user interfaces
  - Ensures compliance with corporate standards by propagating naming conventions
  - Eliminates dead, redundant, and duplicate code to reduce complexity and enhance maintainability

- **Extend applications through Web Services**
  - Accelerates the web-enablement of components within your applications
  - Enables external monitoring of processes
Refactoring of application to increase maintainability
Transformation Assistant™: Features and Benefits

- **Component-based modernization**
  - Migrate selected components to take advantage of the Java environment
  - Tight integration with Application Architect speeds modernization projects

- **Modernization to a fully functional application**
  - Highly assisted path to modernization reduces costs and risks associated with the migration
  - Create database schemas to accelerate the migration
  - Generate Internet accessible screens that transition from your enterprise applications

- **Modernization Workbench integration speeds your initiative**
  - Management of business logic ensures functional equivalency of modernized application
  - Reports illustrate programmatic relationships, simplifying the transition to a new environment