Knowledge Extraction and Analysis of Software Systems for Modernization and Risk Management: A Standards Approach

Rama S. Moorthy, CEO
Mike Oara, CTO
Hatha Systems
Issues raised in Managing Operational Software Management

• Legacy and operational code continues to grow
• Staying current for functional, security and compliance reasons does not scale to address amount of existing code
• Various attempts in tools development (non-standard) seem to address a very narrow field of knowledge extraction (e.g. Business rules, Code quality weaknesses, code security weaknesses)
• System Integrators, despite high revenue generation, are frustrated with the lack of success in modernization
• CIOs (industry and government) continue to rank risk managing operational systems as one of their top 2 priorities two years running.
Industry and Standards progress for managing operational software

• In an attempt to bridge the knowledge gap, Industry through standards innovation invested in the development of a number of standards
  – Knowledge Discovery Metamodel (KDM)
  – Business Process Modeling Notation (BPMN)
  – Semantics of Business Vocabulary and Business Rules (SBVR)
  – Rules Interface Framework (RIF)
  – Resource Description Framework (RDF)

• Significant efforts were made toward interoperability as later standards were developed

• Focus on common data format enabling ease of movement from one analysis environment to another
Static Analysis
Helping to Manage Software Systems’ Risk

Knowledge Refinery (KDM)
Extraction of code characteristics | Detailed Design | Architecture | Business Terms & Rules

Business Process Simulation (BPMN)
System Function Simulation (KDM)
Code Generation for migration (KDM/UML)

Business Process Extraction and Analysis (BPMN)

Compliance (kPath/SBVR) (FISMA, CC, FIPS, CWE)

XML Schema Mapping

KDM Native Source Code Extraction (e.g. COBOL, Java, C, C++, C#, PL1, assembler, Fortran, ADA, Mumps, etc.)

KDM Native Binary Code Weakness Analysis Tools

Open Source Code Weakness Analysis Tools

Rules Engines (RIF)

Non-Standard Code Weakness Analysis Tools (Source and Binary)

Code Quality Analysis Tools

GrammaTech Fortify Coverity Veracode

© Hatha Systems™, LLC
The Standards
Knowledge Extraction and Analysis
Standard models: KDM

- **Knowledge Discovery Metamodel (KDM):** an ISO/OMG Standard providing ontology (a set of definitions) for system knowledge extraction and analysis. KDM provides a framework for the capture of code, platform and other software system characteristics. This further allows the extraction of data flows, control flows, architectures, business/operational rules, business/operational terms, and the derivation of business/operational process; the extraction can be delivered from source, binary, or byte code. Additionally the intermediate representation of the extraction is in executable models creating the possibility of simulation and code generation.
Standard models: SBVR

- **SBVR (Semantics of Business Vocabulary and Business Rules):** An ISO/OMG standard, this specification provides a structured process for formalizing, in natural language, the existing English language representation of compliance points. The standard enables the various compliance specifications (e.g. FISMA, HIPAA, SOX, FIPs, CWEs, etc) to be formalized reducing the room for interpretation from organization to organization when implementing the compliance and auditing requirements.
Standard models: BPMN

- **Business Process Modeling Notation (BPMN):** an OMG standard delivering a modeling notation used to capture business/operational processes in support of system and organizational process simulation and analysis. It is used today to capture both human and IT system processes for the purposes of simulating environments both ‘as is’ and ‘to be’ for software modernization. This notation is compatible with KDM so that system extraction can be represented in BPMN for gap analysis of the current state of the system vs. what is thought to be the current state of the system – critical for modernization and compliance.
• **Data/Metadata Storage Standards:** With the emergence of the standards noted above and the need for storing this information for analysis, a set of storage standards needed to be embraced. XMI, RDBMS, and RDF (Resource Description Framework) are the three formats that are compatible with these standards.

  – RDF - perhaps the least known of them - is a W3C standard that is compatible with KDM and BPMN. There is a specific approach in the standard called RDF triple store which is currently being used in semantic web applications. The value of RDF is that it can manage large amounts of data and metadata which is critical for doing comprehensive static analysis.
What knowledge does Integrating the Standards deliver?
Enterprise Software Systems Static Analysis Standards Mapping

Legend:
- KDM: Knowledge Discovery Metamodel ISO standard
- RDF: Resource Description Framework – W3C standard
- BPMN: Business Process Modeling Notation – OMG standard
- SBVR: Semantics of Business Vocabulary and Rules - ISO standard
Static Analysis & Standards
A Scenario

- Parsing
  - Cobol/CICS/DB2 (Tool A)
  - C/C++ (Tool B)
  - Java (Tool C)
  - C# (Tool C)

- KDM Repository
- Tool D
  - Architecture
- Tool E
  - Common Code Weaknesses
- Tool F
  - Business rules (Business Rules Eng)
- Tool G
  - Processes (Simulation)
## Static Analysis

<table>
<thead>
<tr>
<th>Methods</th>
<th>Benefits</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture diagrams</td>
<td>System understanding in breadth and depth</td>
<td>No performance data</td>
</tr>
<tr>
<td>Data flows</td>
<td>Direct discovery of non-compliance</td>
<td>Some run-time conditions may not be captured</td>
</tr>
<tr>
<td>Control flows</td>
<td>Pinpointing errors in the code, <strong>architecture, process</strong></td>
<td></td>
</tr>
<tr>
<td>Investigative queries</td>
<td>Proof of compliance</td>
<td>Source code is required</td>
</tr>
<tr>
<td>Software measurements</td>
<td>No “guesswork” testing</td>
<td></td>
</tr>
<tr>
<td>Pattern discovery</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simulation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Benefits**

- System understanding in breadth and depth
- Direct discovery of non-compliance
- Pinpointing errors in the code, **architecture, process**
- Proof of compliance
- No “guesswork” testing

**Limitations**

- No performance data
- Some run-time conditions may not be captured
- Source code is required

### Source code to Repository model

- Parsing
- Analysis

**Comprehensive System Diagrams**

- User Definable Queries
- Both comprehensive and Custom/specialized analysis tools
Integrated Analysis Environment

- Business Process Simulation (BPMN)
- Functional Simulation (KDM)
- System Documentation (KDM)
- Simulation (BPMN/KDM)
- Security Analysis: Weakness & Criticality (KDM/kPath)
- Compliance (kPath/SBVR) (FISMA, CC, FIPS, HIPPA, SOX) (SBVR/kPath)
- Rules extraction for Analysis or Rules engine use (RIF)
- Generation (Code, D/B schema, WSDL, Service Definitions) (KDM/UML)

Legend:
- Standards based Analysis Platform
- Standards based analyses and migration solutions
- Software Systems & Applications (COTS, GOTS, Custom)

Software System Static Analysis Platform(s)
- Extraction of Data Flows| Control Flows| Architecture| Business Rules|
- Code/Architecture/Process Weaknesses| Business Processes| System Compliance|
- KDM based platform – Querying environment (kPath or SparQL)
- Storage Standard (XMI, RDMBS, RDF)

Extraction
Automated Analysis
Parsing

Legacy Applications (COBOL/CICS/ICL/IMS/DB2/PL1/VB6, etc)
COBOL/C Program
Java Application
Java/C++ Program
C Program
C#/VB Program
Others (ADA, Fortran, etc)
Software System Knowledge using Standards-Based Tools

Examples of Extractable Static Knowledge Retrievable in Java and COBOL Code
Static Analysis: Data Flows

<table>
<thead>
<tr>
<th>Definition</th>
<th>Compliance applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shows dependencies between variables</td>
<td>What data appears in a user interface? (HIPAA)</td>
</tr>
<tr>
<td></td>
<td>Was data encrypted before being stored or transmitted?</td>
</tr>
</tbody>
</table>
Data Flow

Data flow variant (in a Cobol program)
Static Analysis: Control Flow

**Definition**
Shows flow dependencies between statements

**Compliance applications**
Is data validated before being processed? What conditions lead to a particular action? (example: approval of a request)
Call Maps
Definition

A diagram of relationships between runtime elements of the application (programs, transactions, tables, files, queues, etc.)

Compliance applications

Is the application properly layered?
Are all data access methods done through an API layer?
Which elements of the application work with confidential or secret information?
## Business Rules Report

<table>
<thead>
<tr>
<th>#</th>
<th>Rule Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Admin - Delete CO</td>
<td>Validation</td>
</tr>
</tbody>
</table>

**Description:**
Cannot delete a CO with active solicitations

**WorkUnits:**
- System: Admin App, Function: Generic, Sub-Function: Generic

**Terms:**
(\(\text{none}\))

**Code:**
```java
    if ((person != null) && person.hasActiveSolicitations()) {
        FabricatorFactory.getActivityLogFabricator(application).save(person, "Cannot delete a CO with active solicitations.", NOT_REJECTED, NOT_REJECTED);
        colDto.setSoftDeleteMessage("Cannot delete a CO with active solicitations.");
    } else {
        fabricator.deleteDomainObject(colDto);
    }
```

<table>
<thead>
<tr>
<th>#</th>
<th>Rule Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Admin - Delete Resource Category</td>
<td>Validation</td>
</tr>
</tbody>
</table>

**Description:**
Cannot delete a Resource Category associated with existing solicitations.

**WorkUnits:**
- System: VAPR App, Function: Resources > Structure > Resource Families, Sub-Function: Delete resource category type

**Terms:**
(\(\text{none}\))
Database Schema – Table References
User Navigation

Example: JSP application
Architecture Extraction

Architecture Colorized by Layers – client facing, data facing, mixed
Source Code Version Differential Analysis (Metadata)

Comparing two versions of the same source using a differential analysis technique to show architecture, business layer, design layer differences. What changed? What is impacted by the change?
Rules Extraction Coverage Report

Table: Rule Coverage for Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin App :: Generic</td>
<td>16</td>
</tr>
<tr>
<td>CO App :: About</td>
<td>0</td>
</tr>
<tr>
<td>CO App :: Agreements</td>
<td>9</td>
</tr>
<tr>
<td>CO App :: Amendments</td>
<td>72</td>
</tr>
<tr>
<td>CO App :: Assistants</td>
<td>0</td>
</tr>
<tr>
<td>CO App :: DPLs/Resource Lists</td>
<td>37</td>
</tr>
<tr>
<td>CO App :: Help Homepage</td>
<td>0</td>
</tr>
<tr>
<td>CO App :: Help With</td>
<td>0</td>
</tr>
<tr>
<td>CO App :: Inspections</td>
<td>6</td>
</tr>
<tr>
<td>CO App :: Interested party</td>
<td>6</td>
</tr>
<tr>
<td>CO App :: Login</td>
<td>4</td>
</tr>
<tr>
<td>CO App :: Logout</td>
<td>0</td>
</tr>
<tr>
<td>CO App :: Modifications</td>
<td>106</td>
</tr>
<tr>
<td>CO App :: Overview</td>
<td>0</td>
</tr>
<tr>
<td>CO App :: Reports</td>
<td>1</td>
</tr>
<tr>
<td>CO App :: Resource category</td>
<td>1</td>
</tr>
<tr>
<td>CO App :: Responses</td>
<td>8</td>
</tr>
<tr>
<td>CO App :: Solicitation</td>
<td>88</td>
</tr>
<tr>
<td>VAPR App :: Maintenance</td>
<td>9</td>
</tr>
<tr>
<td>VAPR App :: Resources &gt; Forms</td>
<td>7</td>
</tr>
<tr>
<td>VAPR App :: Resources &gt; NAICS</td>
<td>4</td>
</tr>
<tr>
<td>VAPR App :: Resources &gt; Structure &gt; Resource Families</td>
<td>18</td>
</tr>
<tr>
<td>VAPR App :: Templates &gt; Associations &gt; Manage</td>
<td>0</td>
</tr>
</tbody>
</table>

Shows the number of rules for each function in the application
Weaknesses

Deep nested ‘ifs’ increasing the cost of maintenance
Not to mention quality and security issues
Program Access to Files or Tables

<table>
<thead>
<tr>
<th>GSS3</th>
<th>GSS2</th>
<th>GSSB0001</th>
<th>GSS6</th>
<th>GSSB0003</th>
<th>GSS4</th>
<th>GSSB0002</th>
<th>GSS1</th>
<th>GSS5</th>
<th>GSS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Diagram with various labels and data points.
Example Commercial Solution: Hatha Systems Knowledge Refinery Platform
Knowledge Refinery Platform: Main Window

An Example of a Standards based Commercial Platform
Demonstration
Thank You
Hatha Systems, LLC
202-756-2974
www.hathasystems.com

Rama S. Moorthy
Rama.Moorthy@hathasystems.com

Mike Oara
Mike.oara@hathasystems.com