Automated ABAP Optimization for Upgrades and Unicode Conversions

Chris Hanshew
- Upgrades and Unicode Conversions require you to address custom developed ABAP issues
- Manual approaches focus on addressing errors only
- Utilizing a combination of automated analysis and code correction can:
  - Assist with planning your activities and resources more effectively
  - Reduce manual efforts associated with custom ABAP corrections
  - Potential to address additional quality initiatives never before possible
smartShift Technologies is a Cloud Transformation Company. Our automated tools based on patented technology empower businesses to make use of the latest advances such as the Cloud, Big Data and SAP HANA.

- Proven success in more than 400+ of SAP transformation implementations and 2 decades experience
- Founded in 1992, 500+ employees globally
- CMMI Level 5, ISO 27001, ISO 9001
- SAP Solutions:
  - Upgrade, Unicode and EHP / SPS
  - Code compliance & QA
  - Transformation to SAP HANA and the cloud
  - Big Data
  - SLO
Chris Hanshew  
North America Delivery Lead  
smartShift

Mr. Hanshew has 15 years of experience working with SAP technology including ABAP Development, leading SAP Technical teams and project management. Over the years, he has worked for both large Systems Integrators and SAP customers supporting a wide variety of industries including retail, consumer products and medical device. Today, Chris is responsible for delivery of smartShift projects in North America.

Chris has completed 6 full lifecycle SAP Upgrade projects over his career including multiple upgrade projects utilizing the smartShift suite of tools.

In the past, Chris has been a volunteer with the ASUG organization supporting the Development Technologies Special Interest Group and has spoken at the ASUG spring conference and SCN Community Days at SAP TechEd. He holds a degree in Industrial Management and Management Information systems from Purdue University.
Current Landscape Considerations

- Upgrade to current releases to avoid additional maintenance fees
- System Landscape Optimizations – Consolidations, re-implementations, M&A, carve-outs, etc.
- Leverage new functionalities within the Business Suite
- Preparation for future technologies, e.g., Hana

*IT departments must continue innovation in order to bring value to your business users. Upgrades and Landscape Optimizations are often costly and delay the ability to deliver innovation to their business customers.*
Project Considerations

- Hardware refreshes/migrations
- OS/DB upgrades and migrations
- Unicode Conversions
- Downtime Planning
- Technical Upgrade? Additional Functionality?
- Testing – Full testing? Regression Testing?
- Custom Development remediation
- Interfacing Systems
Custom Development Pain Points

Modification and Custom Code Adjustment (ABAP)

Test Organization and Execution

End-User-Training

Project Management

HW Upgrade

Source: AMR Study Q2/2002
What are those pain points?

- Large inventories of custom developed objects
  - Difficult to scope
  - Limited view into used vs unused objects
- Standard Analysis solutions are not unified, do not provide visibility into all upgrade related issues
- SAP continues to change standard objects and functionality
  - Data Dictionary
  - Function Module
  - Classes
  - Business Transactions
- Use of cloned SAP objects, often times not adapted during prior upgrades
- Identification and upgrade of third party bolt-ons
- Continuous evolution of ABAP syntax
Traditional approaches require some level of trial and error – not all issues can be identified
- Must use a team of developers to correct issues
  - In-house teams
  - System Integrators
  - Off Shore/Near Shore
- Manual corrections can be error prone or inconsistent from one developer to the next
- Manual Corrections and larger teams require more project overhead to track and manage
- Key development resources might be pulled off other business initiatives to focus on upgrade related remediation
- Development freezes and dual maintenance activities
Continuous Evolution of ABAP

- Introduced over 3 decades ago as proprietary language for SAP systems
- Continuous enhancement and expansion to embrace new technologies and language features
- With every SAP upgrade, language enhanced but kept downward compatible
- ABAP has grown to over 350 constructs
- While standard functionalities are upgraded, custom ABAP syntax is hardly ever upgraded because it can be too expensive, disruptive and time consuming
ABAP Source Code Issues over Time

Source Code
- Warnings
- Obsolete Constructs
- Compliance Issues

Follow @ASUG365 and #ASUG2013 on Twitter
What is your ABAP wish list?

- Repository clean-up
  - Clean up of unused objects
  - Reduce number of similar objects
  - Revert clones back to standard

- Upgrade issue development remediation
  - Code changes due to standard SAP object changes
  - ABAP error clean up

- Additional ABAP modernization and optimization
  - Enforcement of naming conventions
  - Replace obsolete constructs
  - Security clean up
  - Performance optimization
Repository Cleanup

Challenge
- Large Custom Development Repositories
- Copies of Custom Objects created over time
- Limited view into used objects from SAP Usage Statistics
- Dead/Commented Code due to changes over time

Analysis Opportunities
- Usage Analysis combined with dependent Objects
- Custom Code Similarity Analysis

Transformation Potential
- Deactivation of objects without risk of deletion
- Automatic deletion of dead/commented code
Upgrade Remediation

Challenge
• Large Custom Development Repositories
• Syntax changes between releases
• SAP Standard object changes between releases

Analysis Opportunities
• Modification Analysis
• Clone Analysis
• BDC Analysis
• Change Impact Analysis

Transformation Potential
• Automatic Source Code Remediation for release, unicode and runtime errors
• Replacement of Obsolete Function Modules
ABAP Modernization and Optimization

Challenge
- Large Custom Development Repositories
- Continuous evolution of ABAP language
- Many developers over time, different knowledge levels and styles
- Source code not optimal for supporting latest technologies

Analysis Opportunities
- Custom Code Rules based Analysis
- Performance Analysis
- Security Analysis
- Hard Code Detection and Analysis

Transformation Potential
- ABAP rules based optimization based on current best practices from SAP
- Enforcement of customer naming conventions and standards
- Hard Coding Replacement
Tying it all together

Blueprint

- Plan Infrastructure
- Assess Technical Objects
- Analyze Upgrade Impacts
- Upgrade Sandbox environment
- Preliminary Remediation
- Plan Testing
- Plan Training

Realization

- Upgrade Development
- SAP Customizing Adjustment
- ABAP Remediation
- Security Remediation
- Unit Testing
- Issue/Defect Resolution
- Upgrade of Quality Environment

Testing

- Integration Testing
- User Acceptance Testing
- Go-Live Planning
- Mock Cutover/Dress Rehearsal
- Issue/Defect Resolution

Sandbox

- Custom Object/Repository Analysis

Development

- Automated Code Transformation

Quality Assurance

Follow @ASUG365 and #ASUG2013 on Twitter
Tying it all together – cont.

Analysis

- Repository Cleanup Analysis
  - Usage/Dependency
  - Code Similarity
- Upgrade Analysis
  - Clone Analysis
  - BDC analysis
  - Modification Analysis
  - Rules based upgrade/unicode source code Analysis
- Optimization
  - Rules based source code analysis
  - Performance
  - Security
  - Hard Coding

Automatic Transformation

- Deactivation of unused Objects
- Source Code based error correction and optimization

Manual Remediation

- SPAU/SPDD
- Clone Adjustment
- BDC Remediation
Upgrade / UC

Pot. Runtime Errors

Stability

Maintainability

Architecture

Coding-Standards

Performance

Security

Custom

Upgrade / Unicode

SAP recommended Coding Standards and best Practice

Customer Coding Standards

Code Performance

Code Security

Customer Specific Coding Issues
smartShift Approach

Extract

Discovery

Transform

Upload

Parser & Meta Model

Rules Engine

SAP R/3 3.x, 4.x, 5.x, ECC 6.0

Extractor

Detect Code Errors

Correct Code

Targets System
SAP R/3 ECC 6.0

Uploader

ABAP experts with extensive transformation experience

Follow @ASUG365 and #ASUG2013 on Twitter
Custom Development Pain Points

Source: AMR Study Q2/2002

- Technical Upgrade & Preparation
- Modification and Custom Code Adjustment (ABAP)
- Test Organization and Execution
- End-User-Training
- Project Management
- HW Upgrade
Benefits of Automated Approach

Source: AMR Study Q2/2002

- Modification and Custom Code Adjustment (ABAP)
- Test Organization and Execution
- End-User-Training
- Project Management
- HW Upgrade

Automated Approach

Save up to 70% Time and Budget

Follow @ASUG365 and #ASUG2013 on Twitter
Impact on your custom ABAP development

smartUpgrade + Optimization Project

Source Code Warnings

R/3 ECC EHP Updates

4.6 4.7 5.0 6.00 6.01 6.02 6.03 6.04 6.05 6.06

Follow @ASUG365 and #ASUG2013 on Twitter
Case Study

Project:
- SAP ECC R/3 4.6c to ECC 6 EhP4
- Unicode Conversion
- Code Optimization

smartTool:
- smartUpgrade + optimization

Automation Rate: 99,6%
425,558 issues handled

Project Time Line:
- 2 weeks transformation

Results:
- Automated replacement of 230 function calls
- Zero-error conversions
- Lowering total cost of ownership
- Minimize dual maintenance
- Reduce testing effort
- Creation of Shell-specific transformation rules to standardize custom code
- Deleted 222,038 lines of obsolete comments

Follow @ASUG365 and #ASUG2013 on Twitter
Automation Tool-based: 2 weeks

Manual: 5+ Months with 45+ Resources
Demo
## Tool demo

![Image of a tool demo interface with various analysis results and charts]

### Analysis Results

<table>
<thead>
<tr>
<th>Category</th>
<th>Number of Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maintainability</td>
<td>258272</td>
</tr>
<tr>
<td>Stability</td>
<td>188110</td>
</tr>
<tr>
<td>Coding Standard</td>
<td>115987</td>
</tr>
<tr>
<td>Architecture</td>
<td>40422</td>
</tr>
<tr>
<td>Error Only</td>
<td>21798</td>
</tr>
<tr>
<td>#105 Resolution of untyped FORM part</td>
<td>13783</td>
</tr>
<tr>
<td>#102 Obsolete character usage (dash, space)</td>
<td>3108</td>
</tr>
<tr>
<td>#168 Convert binary use of linefeed</td>
<td>773</td>
</tr>
</tbody>
</table>

**Total number of parser issues:** 430659

### Parser Actions

<table>
<thead>
<tr>
<th>Action</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic issue resolution</td>
<td>404722 (99.5%)</td>
</tr>
<tr>
<td>Marker inserted in the code</td>
<td>0 (0%)</td>
</tr>
</tbody>
</table>

**Total number of parser actions:** 430659 (406934 without report-only)

### List by issue:

- **Issues selected:** ALL
- **Actions selected:** 5

<table>
<thead>
<tr>
<th>Program</th>
<th>Number of Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZPHRFFI009_DOE_REPORT_SC60</td>
<td>1809</td>
</tr>
<tr>
<td>ZPHRFFI002_DOE_REPORT1</td>
<td>1791</td>
</tr>
<tr>
<td>ZMYRWP5002_OVERALL_PROG_RPT</td>
<td>1619</td>
</tr>
</tbody>
</table>

**Total issues:** 430659 in 5445 objects.

---

Follow @ASUG365 and #ASUG2013 on Twitter
168: Use of Binary Types as Control Characters

<table>
<thead>
<tr>
<th>Code</th>
<th>Classification</th>
<th>Severity</th>
<th>Parser Action</th>
<th>Action</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>168</td>
<td>Unicode</td>
<td>UC Error</td>
<td>Automatic</td>
<td>Replace</td>
<td>Error (UC/Rel) Maintainability</td>
</tr>
</tbody>
</table>

Description

smartTool detected the use of binary types as control characters, which is not Unicode-compliant. smartTool will automatically convert binary control characters as follows:

**smartTool replaces... with...**

<table>
<thead>
<tr>
<th>use of linefeed</th>
<th>object-oriented ABAP fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>binary types</td>
<td>object-oriented ABAP fields</td>
</tr>
<tr>
<td>control string</td>
<td>a basis object</td>
</tr>
</tbody>
</table>

smartTool replaces the following values within DATA .. VALUE or CONSTANTS .. VALUE statements to conform to Unicode standards, as follows:

**smartTool replaces... with...**

<table>
<thead>
<tr>
<th>'00'</th>
<th>CL_ABAP_CHAR_UTILITIES=&gt;MINCHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>'FF'</td>
<td>CL_ABAP_CHAR_UTILITIES=&gt;MAXCHAR</td>
</tr>
<tr>
<td>'09'</td>
<td>CL_ABAP_CHAR_UTILITIES=&gt;HORIZONTAL_TAB</td>
</tr>
</tbody>
</table>
Custom developed ABAP code still remains a key activity in your upgrade project.

Analysis solutions can assist in providing more detailed scoping and planning for your upgrade project.

Automatic transformation of ABAP code can:
- Reduce time to deliver corrected code
- Reduce manual efforts associated with code remediation
- Allow you to address other code initiatives often times bypassed due to time and effort required
- Enhance quality and standardization
Contact

Chris Hanshew
North America Delivery Lead

317-399-4504
chanshew@smartShiftTech.com
@smartShiftTech
http://www.smartshifttech.com/

Follow @ASUG365 and #ASUG2013 on Twitter
Follow @ASUG365 and ASUG CEO Bridgette Chambers @BChambersASUG on Twitter to keep up to date with everything at ASUG.

Follow the ASUGNews team of Tom Wailgum: @twailgum and Courtney Bjorlin: @cbjorlin for all things SAP.
THANK YOU FOR PARTICIPATING

Please provide feedback on this session by completing a short survey via the event mobile application.

SESSION CODE: 0713

For ongoing education on this area of focus, visit www.ASUG.com