

IS01

NASA Enterprise Applications Competency Center (NEACC)

SAP ABAP Development Standards

REVISION H

Approved for Release; Distribution is Unlimited			
<u>APPROVING AUTHORITY</u>			
Signature	Name, Title	Org	Date

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

**NASA Enterprise Applications Competency Center
IS01**

Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003	Revision: H
	Effective Date: 11/01/2013	Page 2 of 45

DOCUMENT HISTORY LOG

Status (Baseline/ Revision/ Canceled)	Document Revision	Effective Date	Description
Baseline		04/28/2004	
Revision	A	05/24/2004	
Revision	B	11/09/2004	
Revision	C	04/12/2006	
Revision	D	05/04/2006	
Revision	E	07/10/2010	
Revision	F	01/26/2011	
Revision	G	09/27/2012	
Revision	H	11/01/2013	

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 3 of 45

TABLE OF CONTENTS

1.0 INTRODUCTION.....	5
1.1 PURPOSE	5
1.1.1 <i>Policy Statement on SAP Standard Objects</i>	6
1.2 APPLICABILITY/SCOPE	6
1.3 APPLICABLE DOCUMENTS.....	6
1.4 REFERENCES	6
1.5 DEFINITIONS	6
1.6 ACRONYMS/ABBREVIATIONS.....	7
2.0 ROLES AND RESPONSIBILITIES.....	8
3.0 PROCEDURE	10
3.1 STANDARDS EFFECTIVITY DATES	10
3.2 STANDARDS QUICK REFERENCE	11
3.3 PROGRAM STRUCTURE.....	16
3.3.1 <i>External ABAP Program Elements</i>	16
3.3.2 <i>Program Name</i>	16
3.3.3 <i>Text Elements</i>	17
3.3.4 <i>Attributes</i>	17
3.4 INTERNAL ABAP PROGRAM ELEMENTS	18
3.4.1 <i>Header</i>	18
3.4.2 <i>Declarative Elements</i>	21
3.4.3 <i>Event Elements</i>	25
3.4.4 <i>Controlling Elements</i>	26
3.5 FUNCTION MODULES	28
3.6 PROGRAM COMMENTS	29
3.6.1 <i>Internal Programming Comments</i>	29
3.6.2 <i>External Programming Comments</i>	30
3.7 DATABASE TABLES.....	30
3.8 OBSOLETE ABAP STATEMENTS	31
3.9 AUTHORIZATION TECHNIQUES.....	31
3.9.1 <i>Authorization Group Checks</i>	31
3.9.2 <i>Transaction Code Checks</i>	31
3.9.3 <i>Additional Authorization Object Checks</i>	31
3.10 ABAP CODING TECHNIQUES.....	32
3.10.1 <i>Data Selection</i>	32
3.10.2 <i>Miscellaneous Guidelines</i>	33
3.11 ABAP OBJECT-ORIENTED PROGRAMMING.....	35
3.11.1 <i>Classes</i>	35
3.11.2 <i>Attributes</i>	35
3.11.3 <i>Methods</i>	36
3.11.4 <i>Constructors</i>	36
3.11.5 <i>Interfaces</i>	36

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003	Revision: H
	Effective Date: 11/01/2013	Page 4 of 45

3.12	SAPSCRIPT	36
3.12.1	<i>SAPscript Form Documentation</i>	36
3.12.2	<i>Other SAPscript Form Subobjects</i>	38
3.12.3	<i>SAPscript Text Elements</i>	38
3.13	SCREEN PAINTER STANDARDS	38
3.14	SAP PROGRAM CUSTOMIZATION	39
3.14.1	<i>OSS Notes Internal Programming Comments</i>	39
3.14.2	<i>OSS Note Assistant: SNOTE</i>	40
3.15	USER EXITS.....	40
3.16	REPORT PAINTER REPORTS	40
3.17	ARCHIVED STANDARDS FROM EARLIER VERSIONS	41
4.0	RECORDS	44
	APPENDIX A: POINTS OF CONTACT	45

LIST OF TABLES

Table 1	– Definitions	6
Table 2	– Programming Standards Definitions.....	7
Table 3	– Acronyms and Abbreviations	7
Table 4	– Roles and Responsibilities	8
Table 5	– Added or Deleted Standards	10
Table 6	– Standards Quick Reference Guide 1	11
Table 7	– Standards Quick Reference Guide 2	13
Table 8	– Standards Quick Reference Guide 3	14
Table 9	– Standards Quick Reference Guide 4	15
Table 10	– Archived Standards.....	41
Table 11	– Records Applicable To This Document.....	44
Table 12	– Points of Contact.....	45

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 5 of 45

1.0 INTRODUCTION

1.1 Purpose

The procedures in this Organizational Issuance (OI) provide information about programming standards for the National Aeronautics and Space Administration (NASA) Enterprise Applications Competency Center (NEACC) Advanced Business Application Programming (ABAP) personnel. The program standards contained within this document are living and incomplete, but continually enhanced with new suggestions, improvements, standards and technologies.

This document's procedure objectives:

- Ensure that programs are well structured.
- Improve program readability.
- Produce consistent and expectant results (program reliability).
- Allow programs to be easily maintained (improved productivity / customer service).
- Provide optional programming guidelines (strong preferences / recommendations).
- Provide a source document for periodical review and update.
- Provide programs that run efficiently.

The ABAP Application Developers use these standards to develop custom ABAP objects in the areas of:

- Custom Reporting
- Dialog Transactions
- SAPscript / Smart Forms / Adobe Portable Document Formats (PDFs)
- Function Modules / Business Application Programming Interfaces (BAPIs)
- Data Dictionary (DD)
- User Exits / Business Add-Ins (BADIs) / Enhancement Implementations
- *Note: IS01-NEACC-CF_ABAP-STD-SW-001, ABAP Development Naming Standards directly addresses the ABAP naming standards, although they are referenced as needed throughout this document. Refer to the ABAP naming standards information in Section 3.3.2 of this document for file name and path information.*

A copy of this document is located in the Standards folder of the ABAP project on the NEACC tab of bReady, file name: IS01-NEACC-CF_ABAP-STD-SW-003, SAP ABAP Development Standards.docx. The NEACC Information Assurance (IA) Team Documentation Workflow Processor stores the official copy of this document in the Documentum Repository. Users may request a copy from this department.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 6 of 45

Note: All references in this document to **bReady** refer to the bReady Enterprise Portal at the URL address of [REDACTED]. The ABAP Project on the bReady portal replaces the shared drive as the designated repository for ABAP Team documentation.

1.1.1 Policy Statement on SAP Standard Objects

([QREF4](#)) ABAP Application Developers (the ABAP Team) **shall not** make modifications to any SAP Standard Objects. The AppDev NASA counterpart shall review and approve any exceptions to this policy. The ABAP Team **shall** make every effort to find technically sound solutions that eliminate the need to justify any exception to this policy.

1.2 Applicability/Scope

The programming standards and guidelines contained in this document are written for ABAP programmers who are employed by, or under contract to, the NEACC.

1.3 Applicable Documents

- IS01-NEACC-CF_ABAP-STD-SW-001, ABAP Development Naming Standards
- NEACC-CF_ABAP-STD-SW-002, ABAP Code Signoff Sheet

1.4 References

- IS01-NEACC-CF_ABAP-STD-SW-001, ABAP Development Naming Standards

1.5 Definitions

Table 1 – Definitions

Term	Definition
ABAP Objects	Components of the ABAP programming language that allows object-oriented programming on the basis of classes and interfaces . ABAP objects were introduced by SAP as of SAP Version 4.6.
ALM Tool	Application Lifecycle Management (ALM) is an application formerly called the Quality Center (QC) tool that the ABAP Team uses to enter and store test steps and test results as well as track defects.
JIRA	Migration Tracking Software provided by Atlassian used by the ABAP team to track SAP ABAP transport migration requests from the development system to the production system.
Milestones	Tasks completed by the ABAP Team.
Packages	Development classes in newer versions of SAP, e.g. ECC 6.0.
SAP Standard Objects	Programming objects in the SAP system that are owned and maintained by SAP. Typically, no modification is possible for these objects without retrieving the object registration key from SAP.
T-code	Transaction code.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 7 of 45

Table 2 – Programming Standards Definitions

Term	Definition
Guidelines	Optional
Program	ABAP development object
Recommended	A strong preference. The adherence of the statement is preferred.
Shall	An imperative. The statement will apply in every case. No exceptions.
Standards	Mandatory

1.6 Acronyms/Abbreviations

Table 3 – Acronyms and Abbreviations

Acronym	Description
ABAP	Advanced Business Application Programming
AIS	Application Integration Support
ALM	Application Lifecycle Management
ALV	ABAP List Viewer
APCMS	Application Point Capacity Management System
BADI	Business Add-Ins
BAPI	Business Application Programming Interface
BDC	Batch Data Conversion
BDT	Business Data Toolset
BTE	Business Transaction Events
CMOD	Customer Modifications
DD	Data Dictionary
ECC	ERP Central Component
EDI	Electronic Data Interchange
ERP	Enterprise Resource Planning
ID	Identification
IPO	Integration Project Office
NASA	National Aeronautics and Space Administration
NEACC	NASA Enterprise Applications Competency Center
NERF	NEACC Enhancement Requirements Form
OI	Organizational Issuance
OO	Object-Oriented
OSS	Online SAP Support
PAI	Process After Input
PDF	Portable Document Format
PII	Personally Identifiable Information

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 8 of 45

Acronym	Description
QA	Quality Assurance
QC	Quality Center
RCA	Root Cause Analysis
RICEF	Reports, Interfaces, Conversions, Enhancements, Forms
SAP	System Application and Product
SBU	Sensitive But Unclassified
SQL	Structured Query Language
SR	Service Request
SRS	Software Requirement Specification

2.0 ROLES AND RESPONSIBILITIES

Table 4 – Roles and Responsibilities

Role	Responsibilities
ABAP Application Developer	<p>The ABAP Application Developer shall:</p> <ul style="list-style-type: none"> • Assign new custom objects to the correct packages • Implement complete knowledge of program comment details and the workings of the program • Implement program standards to develop objects • Write programs and code • Make use of the authorization objects • Work an assigned SR by following the process detailed below. All steps are followed with the exception that all updates for defects are tracked in ALM on the defect instead of on the Application Point Capacity Management System (APCMS) milestone associated with the SR: <ul style="list-style-type: none"> - Review the NEACC Enhancement Requirements Form (NERF), or the milestone in APCMS to gather requirements - Track “in process” task hours in APCMS - Create a new Software Requirement Specification (SRS) document or “check-out” an existing SRS document from bReady, update the SRS, and then request the SRS to be reviewed and approved. An ABAP team member reviews and approves the updated SRS, attaches it to the APCMS milestone, and requests the developer to “check-in” the approved SRS

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

**NASA Enterprise Applications Competency Center
IS01**

Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 9 of 45
--	---	---

Role	Responsibilities
	<ul style="list-style-type: none"> - “Check-in” the approved SRS in bReady - Perform ABAP coding in the development environment upon assignment of the SR, unless special cases require development in another environment - Perform unit tests and log results in ALM - Create a code signoff sheet detailing all modifications as specified in the template - Assign all custom ABAP objects to a Reports, Interfaces, Conversions, Enhancements, Forms (RICEF) Identification (ID). Additional categories of identification include Mobile Apps, Report Painter, Workflow and Others - Create and manage a JIRA ticket for each transport - Host a code review with a teammate, to determine if all standards were met and no obvious issues were found - In APCMS, mark tasks complete, update milestones, and attach the code signoff sheet - Release code and request a transport migration to a test environment for functional testing
Functional Process Owner	<p>The Functional Process Owner shall:</p> <ul style="list-style-type: none"> • Review the NERF form or APCMS milestone with the developer • Identify which user group is allowed access to which data during an authorization security check, depending on what area the context indicates
NEACC Functional Support Team	<p>The NEACC Functional Support Team shall:</p> <ul style="list-style-type: none"> • Complete the NERF form or APCMS milestone • Collaborate with the ABAP Team and the Developer to ensure understanding of the Service Request (SR) requirements • Ensure that other process, configuration, or development changes since the requirements for this SR were first defined will not change the initial requirements defined for this SR
NEACC Functional Team Analyst	<p>The NEACC Functional Team Analyst shall:</p> <ul style="list-style-type: none"> • Review and approve any external documentation that will be viewable to the user community • Consider the context appropriate to the area in which the

—CHECK THE MASTER LIST—

VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 10 of 45

Role	Responsibilities
	ABAP object is supporting, and grant authorization accordingly <ul style="list-style-type: none"> Ensure that the object and context are in agreement with the program or code that the ABAP Team has written
NEACC Quality Assurance (QA) Lead	The NEACC QA Lead shall: <ul style="list-style-type: none"> Review and approve any changes made to the System Application and Product (SAP) ABAP Development Standards document.
NEACC Security Team	The NEACC Security Team shall: <ul style="list-style-type: none"> Identify the proper authorization objects to satisfy the additional security checks
R/3 Tools Developer Team	The R/3 Tools Developer Team shall: <ul style="list-style-type: none"> Complete portions of the NERF Development Process, Application Integration Support (AIS) Evaluation, and Technical Design and Development processes

3.0 PROCEDURE

A quick reference guide aids the procedure details in this document. Hyperlinks to the guide are available throughout the document and the table below.

3.1 Standards Effectivity Dates

For audit purposes, the ABAP programming standards in this document are effective from the approval date for the version in which the standard was defined. Version 2.0 of this document established a new baseline for all ABAP programming standards; therefore, **the initial effectivity date for all standards is 5/24/04.**

Standards added or deleted (**changes not permitted**) in subsequent versions are entered in the following table to ensure that the effectivity date for each standard modification is properly identified:

Table 5 – Added or Deleted Standards

Document Version	Action	Effectivity Date	Standards Affected (Referenced by Number)	Hyperlinks
2.0	Added	05/24/04	Standards 1 to 59	QREF1 , QREF2 , QREF3
3.0	Added	11/30/04	Standard 7.1	QREF1

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 11 of 45

Document Version	Action	Effectivity Date	Standards Affected (Referenced by Number)	Hyperlinks
3.0	Added	11/30/04	Standards 35.1 to 35.2	QREF2
3.0	Deleted	11/30/04	Standard 59	Archived
3.0	Added	11/30/04	Standards 60 to 73	QREF3 , QREF4
4.0	Deleted	04/18/06	Standards 10, 11, 12, 15, 17, 18 and 19	Archived
4.0	Added	04/18/06	Standards 6.1, 10.1, 11.1, 12.1, 15.1, 17.1, 18.1 and 19.1	QREF1
4.0	Added	04/18/06	Standards 66.1, 70.1, 70.2, 74	QREF4
4.1	Deleted	05/09/06	Standards 35.1, 35.2, 42, 43 and 44	Archived
4.1	Added	05/09/06	Standards 35.3 to 35.4	QREF2
4.2	Deleted	07/12/10	Standard 70	QREF4
F	Added	01/26/11	Standards 65.1, 73.1, and 75	QREF3 , QREF4
G	Deleted	09/27/12	Standards 10.1, 11.1, 12.1, 15.1, 17.1, 18.1, 19.1, 28, 57, 61, 62, 63, 64 and 73	QREF1 , QREF3
G	Added	09/27/12	Standards 6.2, 10.2, 11.2, 12.2, 15.2, 17.2, 18.2, 19.2, 28.1 and 57.1	QREF4
H	Added	11/01/13	Standards 73.2 and 73.3	QREF4

3.2 Standards Quick Reference

Table 6 – Standards Quick Reference Guide 1

No.	Standard	Hyperlink
	External ABAP Program Elements	
1	Program identification shall be in compliance with the ABAP naming standards.	Program Name
2	All text literals shall be declared as "Text Elements".	Text Elements
3	The application attribute shall be defined.	Application
	Internal ABAP Program Elements: Header	
4	The header shall be declared just before or just after the introductory statement.	Header
5	The header shall take the form of that contained in the template program ZABAPTEMPLATE.	
6	The phrase “ take the form of ” means that the header shall contain all six items from the template.	
6.1	For user exit objects, the header shall take the form of that contained in the template program ZABAPTEMPLATE_EXITS.	

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 12 of 45

No.	Standard	Hyperlink
6.2	For enhancement implementation objects, the header shall take the form of that contained in the template program ZABAPTEMPLATE ENHANCEMENTS	
7	All new modification log entries shall include the transport and SR numbers.	Modification Log
7.1	Modification log entries for include programs shall also be added to the main program.	
	Internal ABAP Program Elements: Declarative Elements	
8	The introductory statement shall start in column one on a line immediately before or after the header.	Intro Stmts
9	All messages shall be defined in a message class via transaction SE91.	
10.2	The TABLES element shall start in a column consistent with Pretty Printer formatting. Each table declaration shall start on a new line.	Tables
11.2	The TYPES element shall start in a column consistent with Pretty Printer formatting. Each type declaration shall start on a new line.	Types
12.2	The DATA element shall start in a column consistent with Pretty Printer formatting. Each data field declaration shall start on a new line.	Data
13.1	The data field declaration type shall be aligned except for structure definitions utilizing the “INCLUDE STRUCTURE” format.	
14	Data fields with the same name shall not be declared as both global and local.	
15.2	The CONSTANTS element shall start in a column consistent with Pretty Printer formatting. Each constant declaration shall start on a new line.	Constants
16	The constant declaration type and value definitions shall be aligned.	
17.2	The FIELD-SYMBOLS element shall start in a column consistent with Pretty Printer formatting. Each field symbol shall start on a new line.	Field-Symbols
18.2	The SELECT-OPTIONS element shall start in a column consistent with Pretty Printer formatting. Each selection option shall start on a new line.	Select-Options
19.2	The PARAMETERS element shall start in a column consistent with Pretty Printer formatting. Each parameter shall start on a new line.	Parameters
	Internal ABAP Program Elements: Event Elements	
20	If needed, the INITIALIZATION event shall be defined before the START-OF-SELECTION event.	Initialization
21	The START-OF-SELECTION event shall always be present in	Start-of-

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 13 of 45

No.	Standard	Hyperlink
	all ABAP reporting programs.	Selection
22	The END-OF-SELECTION event shall always be present in all ABAP reporting programs.	

Note: See section 3.1 for standards change history ([Standards Effectivity Dates](#)).

Table 7 – Standards Quick Reference Guide 2

No.	Standard	Hyperlink
	Internal ABAP Program Elements: Controlling Elements	
23	When a variable is repeatedly evaluated for discrete values, the IF...ELSE...ENDIF statement shall be replaced with a CASE...WHEN...ENDCASE statements.	If Statement
24	CASE...WHEN...ENDCASE statements shall not exceed three levels of nesting; instead, the coding block will be restructured or redesigned.	Case Statement
25	The ‘WHEN OTHERS’ processing block shall always be coded in a CASE statement.	
26	DO...ENDDO loops shall always be coded with a controlled exit point.	Do Statement
27	WHILE...ENDWHILE loops shall always be coded with a controlled exit point.	
28.1	FORM...ENDFORM statements shall start in a column consistent with Pretty Printer formatting, with subroutine statements indented between them.	Form Statement
29	Each subroutine shall be prefixed with a FORM header comment block.	
30	The subroutine comment block shall include the subroutine name.	
31	All declarative elements defined in a function module shall conform to the standards in this document.	Function Modules
	Internal Programming Comments	
32	Program comments shall appear in the header and before each FORM routine.	Internal Comments
33	For program changes, the transport number shall be included in a full comment line pairing for code blocks or in an inline comment for each single line change.	
34	Obsolete lines of code (code that is no longer needed as a result of an SR change) shall initially be retained (commented out), not deleted.	
35	When lines of code are commented out, the transport number shall be included in a full comment line pairing.	
	Data Dictionary Objects	
35.3	When creating custom Z-tables, the table name shall begin with a	Data Base

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 14 of 45

No.	Standard	Hyperlink
	“Z” followed by a meaningful description.	Tables
35.4	Non-SAP field names appended to existing SAP tables shall begin with a “ZZ” as a naming convention.	
	Authorization Techniques	
36	All programs shall have an authorization check either at the transaction level, the calling program level, or within the program itself.	Authorization Techniques
	SAPscript	
37	Documentation shall be entered in the DOCUMENTATION subobject of a SAPscript form.	SAPscript Form
38	The SAPscript form header shall be defined in the FORM section of the Documentation subobject.	Form Section
39	The SAPscript form header shall take the form of that contained in the template program ZABAPTEMPLATE.	
40	When deleting obsolete SAPscript code, the manual version management process described in section 3.12.2 of this document shall be followed.	Form Version Management
41	For SAPscript text element changes, the transport number shall be included in a full comment line pairing for code blocks or a full comment line for each single line change.	Text Elements

Note: See section 3.1 for standards change history ([Standards Effectivity Dates](#)).

Table 8 – Standards Quick Reference Guide 3

No.	Standard	Hyperlink
	Screen Painter	
45	The processing logic command ON CHAIN INPUT and ON CHAIN REQUEST shall be used to reduce redundant processing.	Screen Painter
46	When dialog messages are issued on a Screen Painter Dynpro that relate to an individual field, the cursor shall be dynamically positioned on the field.	
47	The EXIT FROM STEP LOOP statement shall be used where premature termination of Dynpro loop processing is required.	
48	Transactions that change the SAP databases (DBs) shall check whether the desired record is free to be changed.	
	SAP Program Customization	
50	Approval shall be received from NASA before proceeding with the copy.	Pgm Customization
51	The header standards as defined in the Header section shall be followed.	
52	The name of the original program that was copied shall be included in the header.	

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 15 of 45

No.	Standard	Hyperlink
53	The reason why a copy was needed shall be included in the header.	
54	For Online SAP Support (OSS) Note changes, the OSS Note number shall be included in the Modification Log of the header.	
55	All custom code changes shall adhere to all the standards in this document.	
56	The original code and OSS Note code changes shall not be subject to the standards of this document.	
57.1	An entry shall be added and maintained in the spreadsheet found in the ABAP RICEF and Documentation folder of the ABAP project on the NEACC tab of bReady.	
	OSS Notes	
58	The OSS Note number, the transport number, and the SR number shall be included in a full comment line pairing for code blocks.	Comments
60	For single line changes, a full comment line pairing shall be used to delimit the change instead of an inline comment.	
	User Exits	
65	User exits defined by SMOD shall be maintained using transaction Customer Modifications (CMODs).	User Exits
65.1	All new CMOD project names shall start with a 'Z' in compliance with the ABAP naming standards.	

Note: See section 3.1 for standards change history ([Standards Effectivity Dates](#)).

Table 9 – Standards Quick Reference Guide 4

No.	Standard	Hyperlink
	ABAP Object Oriented Programming	
66	Local classes shall be defined in an ABAP program with the prefix 'lcl'.	Classes
66.1	Global classes shall maintain a header in the class relevant local definitions section.	
67	Data objects shall be defined with the DATA statement using the TYPE addition.	Attributes
68	The LIKE reference shall only be used for data objects that reference typed objects.	
69	Work areas and internal tables shall be used (instead of internal tables with header lines).	
70.1	Each method shall be prefixed with a METHOD header comment block.	Methods
70.2	The comment block shall include the method name.	
71	The constructor shall be defined and implemented in the PUBLIC SECTION of the class.	Constructors

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 16 of 45

No.	Standard	Hyperlink
72	Interfaces shall be implemented publicly in the PUBLIC section of the CLASS DEFINITION.	Interfaces
	ABAP Coding Techniques	
73	When updating more than one record in a DB table, programs shall use an internal table to change the records.	
73.1	SAP tables shall only be modified by SAP standard functions, BAPIs, or Batch Data Conversions (BDCs).	Coding Techniques
	Anticipating collisions for Major Releases	
73.2	For existing function groups, new includes shall be used for new development whenever possible.	
73.3	The name of all new includes shall be consistent with the names of existing includes within the function group.	
	Report Painter	
74	When a “ZNASA” report painter report is modified (via transaction FMEM); the BW team shall be notified to ensure that their reports remain in sync with the report painter reports.	Report Painter
	Policy Statement on SAP Standard Objects	
75	ABAP Application Developers shall not make modifications to any SAP Standard Objects. The ABAP Team shall make every effort to find technically sound solutions that eliminate the need to justify any exception to this policy.	SAP Std Objects

Note: See section 3.1 for standards change history ([Standards Effectivity Dates](#)).

3.3 Program Structure

An important aspect of any programming standard is the need to define the standard layout of the program. This requirement is critical in producing commonly structured programs. All custom programs will adhere to these standards except for those programs that SAP automatically generates.

3.3.1 External ABAP Program Elements

External program elements refer to those program characteristics external to the actual program source code.

3.3.2 Program Name

([QREF1](#)) Program identification **shall** be in compliance with the naming standards. The official copy is stored in the Documentum Repository and is available from the NEACC IA Team. In addition, a *copy* of the latest document is located in the Standards folder of the ABAP project on the NEACC tab of bReady, file name: IS01-NEACC-CF_ABAP-STD-SW-001 ABAP Development Naming Standards.docx.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 17 of 45

3.3.3 Text Elements

([QREF1](#)) All text literals **shall** be "Text Elements" within an ABAP program. Please note that character data declared as a constant is not a text literal (the ABAP syntax checker does not allow a constant value to be assigned to a text element).

Examples:

WRITE: TEXT-001 (Preferred).
WRITE: 'Material number' (001).....

3.3.4 Attributes

Attributes include type, application, title, and package.

3.3.4.1 Type

SAP requires the type attribute. Use the drop down selection list to choose the proper value for the program.

Examples:

Executable program
INCLUDE program
Module pool
Function group

3.3.4.2 Application

([QREF1](#)) The application attribute **shall** be defined. Use the drop down selection list to choose the value that best fits the application area for the program.

Examples:

Financial accounting
General ledger
Materials management

3.3.4.3 Title

SAP requires the title attribute. It should be a meaningful description for the program. For custom reports (executable programs), keep in mind that the title is saved in the system variable SY-TITLE and is displayed on the selection screen and the generated list; however, the title can be customized, if necessary, by changing the content of SY-TITLE.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 18 of 45

3.3.4.4 Package

Note: In newer versions of SAP, e.g. Enterprise Resource Planning (ERP) Central Component (ECC) 6.0, development classes are now called packages.

The ABAP Team creates packages for use when creating programs. These packages follow the naming standards listed in IS01-NEACC-CF_ABAP-STD-SW-001, ABAP Development Naming Standards. See the section on packages for more details. Refer to the ABAP naming standards information in Section 3.3.2 of this document for file path information.

Use transaction SE80 to review all custom packages that are available. Developers should be careful to assign new custom objects to the correct package. The ABAP Team will create additional packages when necessary.

3.4 Internal ABAP Program Elements

Internal program elements refer to those program characteristics that directly relate to the ABAP program source code.

3.4.1 Header

([QREF1](#)) The header **shall** be declared just before or just after the introductory statement. Use the header to describe the characteristics of a development object at a summary level.

The header **shall** take the form of that contained in the template program ZABAPTEMPLATE. This template program is located in package Z_LOCAL on the development system and a backup copy is located in the Templates folder of the ABAP project on the NEACC tab of bReady, file name: zabaptemplate.txt.

Copy the template into the program and then modify, filling in the details for the specific program. Each item contained within the standard header is detailed below:

```

*****
* NASA                ZABAPTEMPLATE: V1.2, 04/11/06
*****
* PROGRAM:           <name of ABAP object>
*
* CREATION DATE:     <date created, e.g. May, 2004>
*
* AUTHOR (CREATED BY): <programmer name>
*
* DESCRIPTION:       Header template for all ABAP objects. The
*                   header will be declared just before or just
*                   after the introductory statement. The header
*                   is used to describe the characteristics of a
*                   program at a summary level.
*
*****

```

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 19 of 45

```

*                               Modification Log
*****
*   Date   Developer Transport   Description
*-----
*mm/dd/yy userid   DEVK9nnnnn SR-nnnnn Initial Program Development.
*****

```

The phrase “**take the form of**” means that the header **shall** contain all six items from the template program ZABAPTEMPLATE. These items are listed below:

- NASA line
- PROGRAM line
- CREATION DATE line
- AUTHOR (CREATED BY) line
- DESCRIPTION line or lines
- Modification Log

For user exit objects, the header **shall** take the form of that contained in the template program ZABAPTEMPLATE_EXITS. This template program is located in package Z_LOCAL on the development system and a backup copy is located in the Templates folder of the ABAP project on the NEACC tab of bReady, file name: zabaptemplate_exits.txt. This template has additional information that is applicable only to user exits:

- CMOD PROJECT line
- SAP ENHANCEMENT line
- MODULE POOL line
- FUNCTION EXIT line
- SCREEN EXIT line

```

*****
* NASA                               ZABAPTEMPLATE_EXITS: V1.3, 08/25/08
*****
* PROGRAM:                            <name of ABAP object>
*
* CMOD PROJECT:                        <name, e.g. ZFIFM_FC>
*
* SAP ENHANCEMENT:                    <name, e.g. FMMD0009>
*
* MODULE POOL:                        <name, e.g. SAPLXM06>
*
* FUNCTION EXIT:                       <name, e.g. EXIT_SAPMM06E_016>
*
* SCREEN EXIT:                         <screen number, e.g. SAPLXM06-0101>
*                                     <screen number, e.g. SAPLXM06-0111>
*
* CREATION DATE:                       <date created, e.g. May, 2004>
*
* AUTHOR (CREATED BY): <programmer name>
*
* DESCRIPTION:                         Header template for all ABAP objects. The
*                                     header will be declared just before or just

```

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 20 of 45

```

*           after the introductory statement. The header
*           is used to describe the characteristics of a
*           program at a summary level.
*
*****
*
*           Modification Log
*****
*   Date   Developer Transport   Description
*-----
*mm/dd/yy userid   DEVK9nnnnn SR-nnnnn Initial Program Development.
*****

```

For enhancement implementation objects, the header **shall** take the form of that contained in the template program ZABAPTEMPLATE_ENHANCEMENTS. This template program is located in package Z_LOCAL on the development system and a backup copy is located in the Templates folder of the ABAP project on the NEACC tab of bReady, file name: zabaptemplate_enhancements.txt. This template has additional information that is applicable only to enhancement implementations:

- COMPOSITE ENH line (optional)
- ENHANCEMENT-POINT or ENHANCEMENT SPOT line
- IMPLEMENTATION line
- VERIFICATION STMT line

```

*****
* NASA           ZABAPTEMPLATE_ENHANCEMENTS: V1.0, 08/25/08
*****
* PROGRAM:      <name of ABAP object>
*
* COMPOSITE ENH.: <name, e.g. ZEM_EQUI_CEI>
*
* ENHANCEMENT-POINT: <name, e.g. Form EXIT_COMMAND_F00, Start>
* SPOT:          <name, e.g. ES_SAPLIT0B_BAPI_EQ>
*
* IMPLEMENTATION: <name, e.g. Z_EXIT_COMMAND_F00>
*
* VERIFICATION STMT: The following verification statement was
* confirmed on mm/dd/yy:
*
* *****
* *** This implementation does NOT supercede an SAP implementation. ***
* *****
*
* *****
* *** This implementation supercedes SAP implementation ***
* *** <name of implementation>. The SAP logic has ***
* *** been reviewed and is NOT applicable to NASA. ***
* *****
*
* *****
* *** This implementation supercedes SAP implementation ***
* *** <name of implementation>. The SAP logic has ***
* *** been reviewed and is applicable to NASA. The logic ***
* *** has been incorporated into this implementation. ***
* *****

```

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 21 of 45

```

*
* MODULE POOL:          <name, e.g. SAPLXM06>
*
* FUNCTION EXIT:       <name, e.g. EXIT_SAPMM06E_016>
*
* SCREEN EXIT:         <screen number, e.g. SAPLXM06-0101>
*                       <screen number, e.g. SAPLXM06-0111>
*
* CREATION DATE:       <date created, e.g. May, 2004>
*
* AUTHOR (CREATED BY): <programmer name>
*
* DESCRIPTION:         Header template for all ABAP objects. The
*                       header will be declared just before or just
*                       after the introductory statement. The header
*                       is used to describe the characteristics of a
*                       program at a summary level.
*
*****
*                       Modification Log
*****
*   Date   Developer Transport   Description
*   -----
*mm/dd/yy userid   DEVK9nnnnn SR-nnnnn Initial Program Development.
*****

```

3.4.1.1 Modification Log

(QREF1) All new modification log entries **shall** include the transport and SR numbers. Existing modification log entries should include the transport and SR numbers. Modification log entries for include programs **shall** also be added to the main program. This ensures that the main program has a complete history of all modifications for each include referenced by the main program.

3.4.2 Declarative Elements

"Declarative Elements" is the term used to describe available ABAP elements necessary to define the internal environment under which a program will run. Each new program will include only those declarative elements required to satisfy the specifications for the approved work item.

Throughout all the declarative elements defined for a program, an exact level of indentation is not required; however, maintain a **“level of consistency”**. A “level of consistency” means use indentation to improve the readability of the source code. Separate each new declarative element by at least one blank line.

3.4.2.1 Introductory Statements

(QREF1) Each ABAP program type has a statement that introduces programs of that type.

Examples:

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003	Revision: H
	Effective Date: 11/01/2013	Page 22 of 45

Program Type	Type Description	Introductory Statement
1	Executable program	REPORT
M	Module pool	PROGRAM
F	Function group	FUNCTION-POOL
S	Subroutine pool	PROGRAM
I	INCLUDE program	n/a

The introductory statement **shall** start in column one on a line immediately before or after the header. Use a minimal number of blank lines to separate the introductory statement from the header.

Recommendations:

1. Include the "NO STANDARD PAGE HEADING" parameter for those programs that are of Type 1 and produce one or more reports as output.
2. Ensure the report heading formats satisfy the specifications of the client, if provided (a common heading routine for all NASA reports does not exist at this time).

Example:

```
REPORT ZRFI_AP_HHS_REPORT NO STANDARD PAGE HEADING
      LINE-SIZE 132
      LINE-COUNT 65
      MESSAGE-ID ZFI.
```

When using the MESSAGE statement, define all messages in a message class via transaction SE91.

3.4.2.2 TABLES Element

([QREF1](#)) The TABLES element **shall** start in a column consistent with Pretty Printer formatting (e.g. starts at Level 1 indentation within a code block). Each table declaration **shall** start on a new line. Descriptive inline comments are recommended.

Examples:

```
TABLES:      BKPF,           " accounting document header
              LFA1,           " vendor master
              T001.          " company codes

TABLES:      BKPF,           " accounting document header
              LFA1,           " vendor master
              T001.          " company codes
```

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 23 of 45

Warning: Creating table work areas with the TABLES statement is forbidden in ABAP Objects.

3.4.2.3 TYPES Element

([QREF1](#)) The TYPES element **shall** start in a column consistent with Pretty Printer formatting. Each type declaration **shall** start on a new line.

Recommendations:

1. Use references to DD fields and structures wherever possible.
2. Use meaningful assignment of names and descriptive inline comments.

Type definitions include: simple, structured and internal table.

Example of a structured type:

```
TYPES:
  BEGIN OF T_VENDOR_DATA,
    GSBER TYPE GSBER, " business area
    EKORG TYPE EKORG, " purchasing org
    STCD3 TYPE STCD3, " EIN number
    LIFNR TYPE LIFNR, " vendor number
  END OF T_VENDOR_DATA.
```

3.4.2.4 DATA Element

([QREF1](#)) The DATA element **shall** start in a column consistent with Pretty Printer formatting. Each data field declaration **shall** start on a new line. The data field declaration type **shall** be aligned except for structure definitions utilizing the "INCLUDE STRUCTURE" format.

Recommendations:

1. Align data field value definitions with minimal exceptions.
2. Reference DD fields and structures via the addition TYPE (if possible, avoid using LIKE).
3. Use meaningful assignment of names and descriptive inline comments.

Data fields with the same name **shall not** be declared as both global and local.

Data definitions include: simple, structured and internal table.

Example of simple field definitions:

```
DATA:
  V_GSBER TYPE GSBER VALUE '62', " MSFC business area
  V_EKORG TYPE EKORG VALUE 'MSFC', " purchasing org
  V_STCD3 TYPE STCD3, " EIN number
```

—CHECK THE MASTER LIST—

VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 24 of 45

```
V_LIFNR TYPE LIFNR,           " vendor number
V_FLAG(1) TYPE C              VALUE 'X'.      " 'X'=yes, ' '=no
```

Example of internal table and work area definitions:

```
DATA:
  I_VENDOR_DATA TYPE STANDARD TABLE OF T_VENDOR_DATA,
  WA_VENDOR_DATA TYPE T_VENDOR_DATA.
```

3.4.2.5 CONSTANTS Element

[\(QREF1\)](#) The CONSTANTS element **shall** start in a column consistent with Pretty Printer formatting. Each constant declaration **shall** start on a new line. The constant declaration type and value definitions **shall** be aligned.

Recommendations:

1. Use references to DD fields and structures wherever possible via the additions of LIKE or TYPE (preferred).
2. Use meaningful assignment of names and descriptive inline comments.
3. Use constants wherever possible, instead of hard-coded literals.

Example:

```
CONSTANTS:

* PO document types for HHS (EKK0-BSART)
C_YF      TYPE BSART  VALUE 'YF',  " space act agreement
C_YG      TYPE BSART  VALUE 'YG',  " grant
C_YH      TYPE BSART  VALUE 'YH',  " cooperative agreement
C_YS      TYPE BSART  VALUE 'YS',  " contact letter of credit
```

3.4.2.6 FIELD-SYMBOLS Element

[\(QREF1\)](#) The FIELD-SYMBOLS element **shall** start in a column consistent with Pretty Printer formatting. Each field symbol **shall** start on a new line.

Recommendations:

1. Use references to DD fields and structures wherever possible.
2. Use meaningful assignment of names and descriptive inline comments.

Example:

```
FIELD-SYMBOLS:
  <FS_GSBER> TYPE GSBER,  " business area
  <FS_ANY>   TYPE ANY.   " generic field symbol
```

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 25 of 45

Warning: Untyped field symbols are forbidden in ABAP Objects (see ABAP help for more details).

3.4.2.7 SELECT-OPTIONS Element

([QREF1](#)) The SELECT-OPTIONS element **shall** start in a column consistent with Pretty Printer formatting. Each selection option **shall** start on a new line.

Recommendations:

1. Use references to DD fields and structures wherever possible.
2. Use meaningful assignment of names and descriptive inline comments.

Example:

```
SELECT-OPTIONS:
  USERID FOR SY-UNAME, " SAP user logon name
  DOC_DATE FOR SY-DATUM.    " current date
```

Note: SELECT-OPTIONS variable names are restricted to a maximum length of 8 characters.

3.4.2.8 PARAMETERS element

([QREF1](#)) The PARAMETERS element **shall** start in a column consistent with Pretty Printer formatting. Each parameter **shall** start on a new line.

Recommendations:

1. Use references to DD fields and structures.
2. Use meaningful assignment of names and descriptive inline comments.

Example:

```
PARAMETERS:
  P_GSBER TYPE TGSB-GSBER,          " business area
  P_EKORG   TYPE T024E-EKORG MANDATORY, " purchasing org
  P_UP2DTE  TYPE SYDATS.           " "Up to" date
```

Note: Parameter names may be up to 8 characters long.

3.4.3 Event Elements

3.4.3.1 Initialization

([QREF1](#)) If needed, define the INITIALIZATION event before the START-OF-SELECTION event.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 26 of 45

3.4.3.2 Start-of-Selection / End-of-Selection

([QREF1](#)) The START-OF-SELECTION event **shall** always be present in all ABAP reporting programs. The END-OF-SELECTION event **shall** always be present in all ABAP reporting programs.

3.4.3.3 Top-of-Page / Top-of-Page During Line-Selection

Recommendations:

1. Include the TOP-OF-PAGE event for Type 1 programs that produce a basic list as output.
2. Define the TOP-OF-PAGE DURING LINE-SELECTION event in all executable programs that produce a secondary list as output.

3.4.4 Controlling Elements

Controlling elements are ABAP statements that control the flow of the program within processing blocks. Throughout all the controlling elements defined for a program, an exact level of indentation is not required; however, maintain a “**level of consistency**”.

Note: Use Pretty Printer formatting to indent all controlling elements.

3.4.4.1 IF...ELSE...ENDIF

([QREF2](#)) When a variable is repeatedly evaluated for discrete values, the IF...ELSE...ENDIF statement **shall** be replaced with a CASE...WHEN...ENDCASE statement.

Recommendations:

1. The IF...ELSE...ENDIF statements not exceed three levels of nesting; instead, use the CASE...WHEN...ENDCASE statement, or redesign the coding block.
2. Avoid using IF...ELSEIF statements.
3. Do not use IF ... ENDIF single line statements.

Example of a typical IF... ELSE ... ENDIF statement block:

```
IF BKPF_BELNR EQ INVOICE_NUMBER.
  PERFORM PROCESS_INVOICE.
ELSE.
  PERFORM CHECK_DOCUMENT_NUMBER.
ENDIF.
```

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003	Revision: H
	Effective Date: 11/01/2013	Page 27 of 45

3.4.4.2 CASE...WHEN...ENDCASE

([QREF2](#)) CASE...WHEN...ENDCASE statements **shall not** exceed three levels of nesting; instead, the coding block **shall** be restructured or redesigned. The ‘WHEN OTHERS’ processing block **shall** always be coded in a CASE statement.

The most likely 'WHEN' condition is coded first to improve performance.

Example:

```

CASE LFB1-EIKT0.
  WHEN '00014000'.           " most likely condition
    <...CODE...>.
  WHEN '00014500'.
    <...CODE...>.
  WHEN '00014500'.
    <...CODE...>.
  WHEN OTHERS.
    <...CODE...>.
ENDCASE.

```

3.4.4.3 DO...ENDDO / WHILE...ENDWHILE

([QREF2](#)) DO...ENDDO loops **shall** always be coded with a controlled exit point. WHILE...ENDWHILE loops **shall** always be coded with a controlled exit point.

Examples:

```

DO 9 TIMES.                 " controlled exit
  <...CODE...>.
ENDDO.

DO.
  <...CODE...>.
  IF <...CONDITION...>.
    EXIT.                   " controlled exit
  ENDIF.
ENDDO.

```

3.4.4.4 LOOP TERMINATION

For readability and maintainability purposes, control loop termination via the CHECK and EXIT statements.

Examples:

```

(Using CHECK)
LOOP AT ITAB INTO WA_TAB.
  <...CODE...>.
  CHECK <...CONDITION...>. " controlled exit
ENDLOOP.

```

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 28 of 45

```

(Using EXIT)
LOOP AT ITAB INTO WA_TAB.
  <...CODE...>.
  IF <...CONDITION...>.
    EXIT.           " controlled exit
  ENDIF.
ENDLOOP.

```

3.4.4.5 Termination of Processing

Warning: Do not use REJECT and STOP statements to terminate coding blocks. Both statements are forbidden in ABAP Objects (see ABAP help for more details).

3.4.4.6 FORM...ENDFORM (SUBROUTINES)

([QREF2](#)) FORM...ENDFORM statements **shall** start in a column consistent with Pretty Printer formatting, with subroutine statements indented between them. Each subroutine **shall** be prefixed with a FORM header comment block. The subroutine comment block **shall** include the subroutine name.

Recommendations:

1. Use subroutines to ensure that programs are well structured.
2. Include a brief description of what the subroutine does.
3. Include an ENDFORM statement inline comment with the subroutine name.
4. Call subroutines the same from one program to another.
5. Incorporate subroutines that are common to two or more programs into an INCLUDE program or a FUNCTION MODULE.
6. Do not duplicate common code across more than one program.

Example:

```

*&-----*
*&      Form <SUBROUTINE NAME>
*&-----*
* <Free form descriptive text.>
*-----*
FORM READ_VENDOR_DATA.

  <...CODE...>.

ENDFORM.           " <SUBROUTINE NAME>

```

3.5 Function Modules

([QREF2](#)) All declarative elements defined in a function module **shall** conform to the standards in this document.

Recommendations:

1. Define function modules in the same logically related function group.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 29 of 45

- Limit function groups to 10-15 functions to avoid performance degradation and memory constraints during execution.

3.6 Program Comments

Ensure that program comments contain sufficient detail so a developer can understand the intended program workings. Program documentation includes both internal and external documentation.

3.6.1 Internal Programming Comments

([QREF2](#)) Program comments **shall** appear in the header and before each FORM routine. See [Header section](#) and [FORM...ENDFORM section](#) for details.

For program changes, the transport number **shall** be included in a full comment line pairing for code blocks or in an inline comment for each single line change.

Recommendations:

- Use comment blocks, line comments, and in-line comments throughout the program.
- Ensure all program comments are clear and concise, emphasizing the WHAT and WHY aspects of the program.
- In the interest of program changes, include the SR and APCMS milestone numbers whenever possible.

Examples:

```
(CODE BLOCK)
* START OF CODE BLOCK DEVK916258/SR-226905/S30277
  <...CODE...>.
  <...CODE...>.
* END OF CODE BLOCK DEVK916258/SR-226905/S30277

(SINGLE LINE)
<...CODE...>.                "DEVK916258/SR-226905/S30277
```

Obsolete lines of code (code that is no longer needed because of an SR) **shall** initially be retained (commented out), not deleted. When lines of code are commented out, the transport number **shall** be included in a full comment line pairing for code blocks or in an inline comment for each single line change.

Recommendations:

- Include the SR and APCMS milestone numbers are included whenever possible for obsolete lines of code.
- Remove some of the oldest obsolete code to maintain good readability (SAP version management provides an adequate history of all program changes) within the program.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 30 of 45

Example:

```
(CODE BLOCK)
* START OF COMMENT BLOCK DEVK916258/SR-226905/S30277
* <...OBSOLETE CODE...>.
* <...OBSOLETE CODE...>.
* END OF COMMENT BLOCK DEVK916258/SR-226905/S30277

(SINGLE LINE)
* <...OBSOLETE CODE...>. "DEVK916258/SR-226905/S30277
```

3.6.2 External Programming Comments

External comments supplement the internal documentation instead of replacing it. **Note:** Any documentation for custom reports will be viewable by the user community via the information button on the application toolbar.

The appropriate NEACC Functional Team analyst shall review and approve external documentation that will be viewable to the user community.

A detailed procedure on how to display and create external documentation is located in the Procedures or Tutorials folder of the ABAP project on the NEACC tab of bReady, file name: ABAP External Documentation Procedure.docx. Also, a complete list of existing external documentation can be viewed by executing transaction SE61 and specifying document class “RE” and programs that begin with a “Z” (specify “Z*”).

3.7 Database Tables

([QREF2](#)) When creating custom Z-tables, the table name **shall** begin with a “Z” followed by a meaningful description. Custom Z-table names shall comply with the ABAP naming standards from the section entitled “Custom Data Base Tables”. Refer to the ABAP naming standards information in Section 3.3.2 of this document for file path information.

Examples:

ZPS_PROJ_TYP	Project System custom table
ZFI_PVA_COSTPOOL	Financial custom table

Non-SAP field names appended to existing SAP tables **shall** begin with a “ZZ” as a naming convention. Non-SAP fields are appended to SAP tables via an include or append structure.

Example:

EKKO-ZZOBETC	Obligated Estimated Cost
--------------	--------------------------

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 31 of 45

3.7.1.1 Technical Settings

When defining the Data Class, be careful to select the correct setting. Most of the time, the setting will be “USER” for customer data class.

3.8 Obsolete ABAP Statements

The upgraded version of SAP (ECC 6.0) has a new category on the extended program check called “Obsolete Statements.” Whenever creating or changing an ABAP object, it is recommended that every effort is made to avoid using obsolete ABAP statements.

3.9 Authorization Techniques

[\(QREF2\)](#) All programs **shall** have an authorization check either at the transaction level, the calling program level, or within the program itself to ensure that only the appropriate user groups will execute a given program for the appropriate business area (i.e., agency vs. specific center). The programming team will work with the security administration team to ensure that the proper authorization checks have been setup for each program.

There are three authorization check techniques that may be used. The purposes of each and their related procedures are located in the Procedures or Tutorials folder of the ABAP project on the NEACC tab of bReady, file name: ABAP Authorization Check Procedures.docx.

3.9.1 Authorization Group Checks

An authorization group security check is performed when a program is executed. When a program is set up to use an authorization group check, the field ‘SECU’ on the program attributes screen should be populated. If a user’s profile contains the same value that is stored in the program attributes field ‘SECU’, then the user has the necessary authority to execute the given program.

3.9.2 Transaction Code Checks

The transaction code (T-code) authorization check, using the authorization object S_TCODE, is one of the most important and often used security checks used. Users are assigned different S_TCODE objects in their security profile. Users can only execute transactions that are authorized in their profile. This security method offers a great deal of flexibility and control. To use this method, programmers must assign their programs to a t-code in SE80 or SE93.

3.9.3 Additional Authorization Object Checks

Additional authorization objects are those constructed within programs which provide further security restrictions, or limit access to objects that cannot be restricted by the use of a t-code (e.g. Business Application Programming Interface -- BAPIs). For example, accountants for a

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 32 of 45

given NASA center should be able to view only that center's data. Hence, an authorization check at the center level needs to be performed.

There are three steps needed to accomplish these additional security checks:

- The functional process owner must identify which user group(s) is allowed access to which data.
- The SAP security administrators must identify the proper authorization objects to satisfy those security checks.
- The ABAP programmers must check these objects in the appropriate programs using the 'AUTHORITY-CHECK' statement.

3.10 ABAP CODING TECHNIQUES

This section is set aside for miscellaneous programming techniques. Most techniques are guidelines, not standards; however, some techniques are declared to be standards and will also be included in the quick reference table (see [Section 3.2 – Standards Quick Reference](#)).

3.10.1 Data Selection

- When a program is executed, it reads the selected data from the relevant DBs based on evaluations according to the report's specifications. Strongly consider the controlled restriction of unwanted DB accesses. The use of precise specifications within all programs will significantly enhance program performance.
- Use internal tables as work areas for DB tables when practical. Reference tables used repeatedly in a program are candidates for loading internally into the program. If all the fields are necessary, use the STRUCTURE syntax; otherwise, name each field in the internal table the same as the DB field and use the TYPE addition to define the attributes. Use the MOVE-CORRESPONDING statement to load the table unless the entire table is necessary. In that case, use the SELECT ...INTO. The structure of the internal table must be identical to the DB so use the STRUCTURE syntax. Be as restrictive as possible with the WHERE clause to reduce the answer set. Make the estimate of the size of internal tables as accurate as possible (OCCURS). Overestimation will cause excessive memory to be allocated. Underestimation will cause the table size to be incremented four times, doubling each time and then cause the remainder of the table to be paged to disk if the estimate is exceeded. The former is wasteful and the latter causes performance degradation. If the size of the internal table is unknown, use OCCURS 0.
- Use Select Single when selecting records using the key fields, instead of using Native Structured Query Language (SQL) calls. Only select the fields of a table required by the program. This will increase performance and decrease the amount of data sent between the DB server and the application server. Whenever possible, read data into an internal table with one select statement, and then loop through the table rather than using a Select...Endselect loop.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 33 of 45

- When using the key word SORT to sort an internal table, use the BY parameter as often as possible to increase performance. Use Binary Search when reading large internal tables (> 100 rows). Sorting the internal table is much faster than using the ORDER BY in the select statement.
- All select statements will either make use of the table key or an activated table index. **Note:** The DB system does not activate all indexes defined in the DD. Use the ABAP runtime analysis (t-code SE30) and SQL trace request (t-code ST05) programs to ensure optimization of the select statement.
- Recommendations:
 - Select statements with a “FOR ALL ENTRIES” condition in the “where” clause are checked for an empty table condition before selecting any data.
 - Skip select statements if the “FOR ALL ENTRIES” table is empty.
 - Add comments to the code if an empty table condition is valid, to explain why the select statement is not skipped.

3.10.2 Miscellaneous Guidelines

3.10.2.1 Coding

- ([QREF4](#)) SAP tables **shall** only be modified by SAP standard functions, BAPIs, or Batch Data Conversions (BDCs). ***This standard is not applicable to*** custom ABAP tables and SAP configuration tables with a Delivery Class setting of C (i.e. customer table, data is maintained by the customer only).
- Anticipating collisions for major releases:
 - For existing function groups, new includes **shall** be used for new development whenever possible (e.g. new subroutines, modules, macros, data declarations, etc...). The name of all new includes **shall** be consistent with the names of existing includes within the function group:
 - New data declaration includes typically end with “Tnn”.
 - New subroutines includes typically end with “Fnn”.
 - New PBO modules includes typically end with “Onn”.
 - New PAI modules includes typically end with “Inn”.

Example:

Existing function group ZREBU_EXT has the following includes:

```

LZREBU_EXTF01  Bldg Extension: Subroutines for custom screens
LZREBU_EXTI01  Bldg Extension: PAI modules for custom screens
LZREBU_EXT001  Bldg Extension: PBO modules for custom screens
LZREBU_EXTTOP  Bldg Extension: Global data for custom screens
LZREBU_EXTUXX

```

—CHECK THE MASTER LIST—

VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003	Revision: H
	Effective Date: 11/01/2013	Page 34 of 45

For a major release, the function group may look as follows:

LZREBU_EXTF01	Bldg Extension: Subroutines for custom screens
LZREBU_EXTF02	Bldg Extension: New Subroutines for major release
LZREBU_EXTI01	Bldg Extension: PAI modules for custom screens
LZREBU_EXTI02	Bldg Extension: New PAI modules for major release
LZREBU_EXT001	Bldg Extension: PBO modules for custom screens
LZREBU_EXT001	Bldg Extension: PBO modules for major release
LZREBU_EXTTOP	Bldg Extension: Global data for custom screens
LZREBU_EXTT01	Bldg Extension: New Global data for major release
LZREBU_EXTUXX	

- Use **CLEAR** <...> for variable initialization or reset.
- Use **REFRESH** <...> for internal table initialization or reset.
- Use **FREE** <...> for internal tables to release memory whenever large data volumes are encountered.
- Global variables:
 - If variable data is used by only one routine, then define the data locally.
 - If variable data is used by multiple routines, then define the data globally or define formal parameters for the subroutine (programmer's discretion).
- Access sequential files using only two reads: one to prime the process, and one for sequential processing to the end of the file.
- Only use the **ASSIGN** keyword when there are no other possibilities. This keyword generates a heavy load on the system and adversely affects performance.
- Utilize a DB with minimal levels (even if one will be created) when using a logical DB.
- Use the ABAP List Viewer (ALV) grid (SAP List Viewer in release 4.7), whenever possible, in custom reports that produce standard lists.
- Use standard table control logic, whenever possible, in dialog programs.
- Use the same data type for variables that are compared in logical expressions.
- When updating more than one record in a DB table, it is recommended that the program use an internal table to change the records (e.g. **UPDATE dbtab FROM TABLE itab**).

3.10.2.2 Formatting

- Use structured programming or object oriented techniques and a top down methodology where applicable.
- Use the Pretty Printer option of the ABAP editor to help align elements.
- The ABAP programming language permits the specification of multiple statements on one physical line of code. To increase program readability, code each ABAP statement on an individual line.
- The practice of chaining identical ABAP statements can improve program readability and eliminate redundant statements.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 35 of 45

3.10.2.3 Tips

- The programming standards support the ready-made key word structures available in the ABAP language. In the SAP ABAP editor (t-code SE38) a template can be inserted for each function call by first placing the cursor on the line where the code will go, then by pushing the Insert Statement button. Next select the function.

3.11 ABAP Object-Oriented Programming

ABAP Objects is the Object-Oriented (OO) extension to the ABAP language. This section will address the basics of an ABAP Objects program.

3.11.1 Classes

([QREF4](#)) Local classes **shall** be defined in an ABAP program with the prefix 'lcl'. A local class is visible only inside the program in which it is defined. If the class is executed more than once, it is recommended that the class is defined globally in the class library using the Class Builder (transaction SE24). Global classes **shall** maintain a header in the class relevant local definitions section (menu path: Goto -> Local Definitions/Implementations -> Class relevant local Definitions).

Example:

```
CLASS lcl_classname
```

3.11.2 Attributes

([QREF4](#)) Attributes describe the data that can be stored in the objects in a class. Data objects **shall** be defined with the DATA statement using the TYPE addition. The LIKE reference **shall** only be used for data objects that reference typed objects. It is recommended that all attributes are sorted.

Examples:

```
DATA fieldname TYPE type_name.
```

```
DATA: variable1 TYPE type_name.  
      variable2 LIKE variable1.
```

Work areas and internal tables **shall** be used (instead of internal tables with header lines).

Examples:

```
TYPES: table_type TYPE STANDARD TABLE OF table.
```

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 36 of 45

DATA: work_area LIKE LINE OF table_type.

Global data consists of types and constants.

3.11.3 Methods

([QREF4](#)) Each method **shall** be prefixed with a METHOD header comment block. The comment block **shall** include the method name. *This standard is not applicable to* methods for BADI implementations if all the code was inherited from SAP.

It is recommended that:

1. All methods are sorted.
2. A brief description of what the method does is included.
3. ENDMETHOD statements include an inline comment with the method name.

Example:

```

*&-----*
*&      Method <METHOD NAME>
*&-----*
* <Free form descriptive text.>
*-----*
METHOD READ_VENDOR_DATA.

    <...CODE...>.

ENDMETHOD.                                " <METHOD NAME>

```

3.11.4 Constructors

([QREF4](#)) The constructor **shall** be defined and implemented in the PUBLIC SECTION of the class. *This standard is not applicable to* classes for BADI implementations.

3.11.5 Interfaces

([QREF4](#)) Interfaces **shall** be implemented publicly in the PUBLIC section of the CLASS DEFINITION. Like classes, if the interface is executed more than once, it is recommended that the interface is defined globally in the class library using the Class Builder.

3.12 SAPscript

SAPscript is the component of SAP used to maintain custom forms for output processing.

3.12.1 SAPscript Form Documentation

([QREF2](#)) Documentation **shall** be entered in the DOCUMENTATION subobject of a SAPscript form.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 37 of 45

3.12.1.1 FORM Section

(QREF2) The SAPscript form header **shall** be defined in the FORM section of the Documentation subobject. The SAPscript form header is used to describe the characteristics of the form at a summary level. The SAPscript form header **shall** take the form of that contained in the template program ZABAPTEMPLATE (see [Header section](#) for details).

3.12.1.2 WINDOW Sections

These recommendations are for SAPscript form changes:

1. The transport number is included in the window sections that were updated.
2. Clear and concise comments are added to each window section that was changed, emphasizing the WHAT and WHY aspects of the change.

Example:

```
WINDOW : INF02
* ELEMENT : DATES
* START D01K9XXXXX/SR-XXXXX
*   Added new element, DATES, with fields:
*   Transaction Date:
*   Document number:
* END D01K9XXXXX/SR-XXXXX
```

3.12.1.3 SAPscript Form Version Management

(QREF2) SAP does not use version management for SAPscript forms; however, SAP does provide utility program RSTXSCRIP to export and import SAPscript forms. The utility program can be executed using SAP transaction SE38.

As a result, *a manual SAPscript version management process* has been defined as follows:

- Before changing the form, export the current version to the SAPscript folder of the ABAP project on the NEACC tab of bReady.
- Use the following naming convention:

<Form Name>_<Transport#>.FOR

Example:

```
ZFI_AR_1080_DEVK919767.FOR
```

When deleting obsolete SAPscript code, the manual version management process described in this section of the document **shall** be followed. When adding or changing SAPscript code, it is recommended that the manual version management process is followed.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 38 of 45

3.12.2 Other SAPscript Form Subobjects

The manual version management process is recommended to be followed when SAPscript form changes occur in the following subobjects:

- Header
- Paragraph Formats
- Character Formats

3.12.3 SAPscript Text Elements

([QREF2](#)) For SAPscript text element changes, the transport number **shall** be included in a full comment line pairing for code blocks, or a full comment line for each single line change. Include the SR and APCMS milestone numbers whenever possible.

Examples:

```
(CODE BLOCK)
/* START DEVK912345/SR-123456/S12345
/: <...SAPscript CODE...>
/: <...SAPscript CODE...>
/* END DEVK912345/SR-123456/S12345
```

```
(SINGLE LINE)
/* DEVK912345/SR-123456/S12345
/: <...SAPscript CODE...>
```

3.13 Screen Painter Standards

([QREF3](#)) The Screen Painter standards apply to those customized developments involving the creation of tables and transactions.

- Screen fields are associated with DD objects whenever possible.
- The SET SCREEN statement is coded for each possible logical path that can be taken in the Process After Input (PAI) processing logic. This will make the transaction easier to read.
- Use the CHAIN command to group logical fields together before calling a module, which will aid the end user in error corrections. In newer versions of SAP, e.g. ECC 6.0, some custom subscreens, when implemented via a BADI or the Business Data Toolset (BDT), will not need the CHAIN command, especially when exploiting message-logging techniques.
- The processing logic command ON CHAIN INPUT and ON CHAIN REQUEST **shall** be used to reduce redundant processing.

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 39 of 45

- When dialog messages are issued on a Screen Painter Dynpro that relate to an individual field, the cursor **shall** be dynamically positioned on the field.

Example:

```
SET CURSOR FIELD 'LFA1-LIFNR' .           (Standard)
SET CURSOR 6 10 .                         (Non Standard)
```

- The EXIT FROM STEP LOOP statement **shall** be used where premature termination of Dynpro loop processing is required.
- Transactions that change the SAP DBs **shall** check whether the desired record is free to be changed. This check will ensure data integrity is maintained.
- The ENQUEUE / DEQUEUE statements **shall** be used to individually lock table records.
- For the purposes of this standard, a DB update is any modification, deletion or insertion of a DB record or external table.

3.14 SAP Program Customization

([QREF3](#)) Some custom programs are copies of SAP programs. The following standards apply only to these types of custom programs:

- Approval **shall** be received from NASA before proceeding with the copy.
- The header standards as defined in the [Header section](#) **shall** be followed.
- The name of the original program that was copied **shall** be included in the header.
- The reason why a copy was needed **shall** be included in the header.
- For OSS Note changes, the OSS Note number **shall** be included in the Modification Log of the header.
- All custom code changes **shall** adhere to all the standards in this document.
- The original code and OSS Note code changes **shall not** be subject to the standards of this document.
- An entry **shall** be added and maintained in the spreadsheet found in the ABAP RICEF and Documentation folder of the ABAP project on the NEACC tab of bReady, file name: Customization of SAP Programs.xls

3.14.1 OSS Notes Internal Programming Comments

([QREF3](#)) For all OSS Notes applied to custom programs which are copies of SAP programs, the OSS Note number, the transport number, and the SR number **shall** be included in a full comment line pairing for code blocks. For single line changes, a full comment line pairing **shall** be used to delimit the change instead of an inline comment. It is recommended that the APCMS milestone number is included whenever possible.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 40 of 45

Examples:

```
(CODE BLOCK AND SINGLE LINE)
* START OF INSERTION OSS NOTE 1234567 DEVK912345/SR-123456/S12345
  <...OSS NOTE CODE...>.
  <...OSS NOTE CODE...>.
* END OF INSERTION OSS NOTE 1234567 DEVK912345/SR-123456/S12345
```

```
(CODE BLOCK AND SINGLE LINE)
* START OF DELETION OSS NOTE 1234567 DEVK912345/SR-123456/S12345
* <...OBSOLETE CODE...>.
* <...OBSOLETE CODE...>.
* END OF DELETION OSS NOTE 1234567 DEVK912345/SR-123456/S12345
```

3.14.2 OSS Note Assistant: SNOTE

The responsibility to apply OSS Notes has been transferred to the Basis team.

3.15 User Exits

(QREF3) See [Appendix C](#) for details on implementing SAP user exits from SMOD:

- User exits defined by SMOD **shall** be maintained using transaction CMOD.
- All new CMOD project names **shall** start with a 'Z' in compliance with the ABAP naming standards. Refer to the ABAP naming standards information in Section 3.3.2 of this document for file path information. Some project names match the name of the SAP enhancement from SMOD. For existing projects, CMOD provides a rename function which may be used to change a project name, if necessary; but this would require that the project be deactivated, renamed, and then reactivated which is not recommended.
- BADIs are maintained using transaction SE19.
- User exits defined as Business Transaction Events (BTEs) are function modules that are maintained in the following configuration tables:
 - TPS34 - Process BTE: Customer Enhancements
 - TBE34 - Publish & Subscribe BTE: Customer Enhancements
 - TBE24 - Customer Products (activation status)

3.16 Report Painter Reports

Recommendations:

- All custom report painter libraries (use transaction GRR3) begin with a Z.

Examples:

Library: ZPS

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 41 of 45

- All custom report painter report names associated with a library begin with at least the first two characters of the library name.

Examples:

Library: ZPS Report Name: ZPSARPT
Library: ZPS Report Name: ZP123456

- All custom report painter report names associated with a report group (use transaction GR53) begin with at least the first two characters of the report group name or begin with the whole report group name (preferred).

Examples:

Report Group: ZABC Report Name: ZABC-001 (preferred)
Report Group: ZABC Report Name: ZA12-123

[\(QREF4\)](#) When a “ZNASA” report painter report is modified (via transaction FMEM); the BW team **shall** be notified to ensure that their reports remain in sync with the report painter reports.

3.17 Archived Standards from Earlier Versions

The following is a table listing archived standards. (See Section 3.1 for [Standards Effectivity Dates](#)). Each standard has a validity period. Once a standard is no longer used, as determined by ABAP Team, the standard is moved from the active list to the archived list. Because this table is only for historical purposes, the wording represents the wording at the time it was archived, and has not been updated to NASA standards regarding the use of “will” being a statement of fact, versus “shall” being a direct command.

Table 10 – Archived Standards

No.	Standard
10	The TABLES element will start in column one and each table declaration will start on a new line.
10.1	The TABLES element shall start in column 1, 2 or 3. Each table declaration shall start on a new line.
11	The TYPES element will start in column one and each type declaration will start on a new line.
11.1	The TYPES element shall start in column 1, 2 or 3. Each type declaration shall start on a new line.
12	The DATA element will start in column one and each data field declaration will start on a new line.
12.1	The DATA element shall start in column 1, 2 or 3. Each data field declaration shall start on a new line.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 42 of 45

No.	Standard
13	The data field declaration type and value definitions will be aligned except for structure definitions utilizing the “INCLUDE STRUCTURE” format.
15	The CONSTANTS element will start in column one and each constant declaration will start on a new line.
15.1	The CONSTANTS element shall start in column 1, 2 or 3. Each constant declaration shall start on a new line.
17	The FIELD-SYMBOLS element will start in column one and each field symbol will start on a new line.
17.1	The FIELD-SYMBOLS element shall start in column 1, 2 or 3. Each field symbol shall start on a new line.
18	The SELECT-OPTIONS element will start in column one and each selection option will start on a new line.
18.1	The SELECT-OPTIONS element shall start in column 1, 2 or 3. Each selection option shall start on a new line.
19	The PARAMETERS element will start in column one and each parameter will start on a new line.
19.1	The PARAMETERS element shall start in column 1, 2 or 3. Each parameter shall start on a new line.
28	FORM...ENDFORM statements shall start in column one, with subroutine statements indented between them.
35.1	When creating custom Z-tables, the table name will be in compliance with the ABAP naming standards from the section entitled “Custom Data Base Tables.”
35.2	Non-SAP field names added to new Z-tables or appended to existing SAP tables will begin with a “ZZ” as a naming convention.
42	Dynpro sequencing will be controlled dynamically from within the module pool.
43	The SET SCREEN statement will be coded for each possible logical path that can be taken in the PAI processing logic.
44	The CHAIN command will be used to group logical fields together before calling a module.
57	An entry shall be added and maintained in the spreadsheet found on the IEMP Competency Center shared drive.
59	For single line changes, an inline comment with the transport number will be included.
61	For OSS Notes that require new Z-programs, the status attribute shall be set to “SAP standard production program”.
62	The OSS Note log entry shall contain the SR number that required the note to be applied.
63	When an OSS Note has been implemented, the OSS Note processing status shall be set to “Completed”.
64	The transport description field for OSS Notes shall contain the OSS Note number, the date, and the SR number.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 43 of 45

No.	Standard
70	The syntax for calling methods will be as follows: CALL METHOD instance->instance_method().
73	When updating more than one record in a DB table, programs shall use an internal table to change the records.

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center OWI		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003	Revision: H
	Effective Date: 11/01/2013	Page 44 of 45

4.0 RECORDS

Table 11 – Records Applicable To This Document

Name of Record	Storage Location	SBU/PII*	Retention Schedule	Responsible Party	E-mail	Phone
N/A						

*SBU = Sensitive But Unclassified / PII = Personally Identifiable Information

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE

NASA Enterprise Applications Competency Center IS01		
Title: SAP ABAP Development Standards	Document No.: IS01-NEACC-CF_ABAP-STD-SW-003 Effective Date: 11/01/2013	Revision: H Page 45 of 45

APPENDIX A: POINTS OF CONTACT

Table 12 – Points of Contact

Name	Position	E-mail	Phone
	ABAP Manager, Document Owner		
	NEACC QA Lead		
	ABAP Team Lead		
	ABAP Team Lead		

—CHECK THE MASTER LIST—
VERIFY THAT THIS IS THE CORRECT VERSION BEFORE USE