
SAP ABAP Programmierrichtlinien mit Fokus „Sicherheit“



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Inhaltsverzeichnis

Inhaltsverzeichnis	1
Geltungsbereich.....	2
Überblick SAP ABAP Programmierrichtlinien mit Fokus „Sicherheit“	2
Regelwerk - SAP ABAP Programmierrichtlinien mit Fokus „Sicherheit“	4

Geltungsbereich

Diese SAP ABAP Programmierichtlinien gelten ab Inkrafttreten für alle ABAP Entwicklungen auf allen SAP Systeme exklusive den SAP Systemen des CCHH (wie z.B. RD2 und QD2) die durch die TU-Darmstadt verantwortet werden. Dies beinhaltet unter anderem folgende SAP Technologien:

- SAP ERP
- SAP NetWeaver Application Server ABAP
- SAP Solution Manager
- SAP S/4HANA

Für die SAP-Systeme der TU Darmstadt gelten die Vorgaben des BSI-Grundschutzes. In Bezug auf Custom Code Schwachstellen innerhalb der SAP Anwendungen betrachtet dieses Dokument daher Ableitungen und Umsetzungshinweise in Bezug auf Baustein [[BSI GSHB APP.4.6](#) „SAP ABAP-Programmierung“] und [[Umsetzungshinweise zu APP.4.6](#)]. Zusätzlich gibt es in [[BSI GSHB APP 3.1](#) „Entwicklung von Webanwendungen“] allgemeine Web-Entwicklungs-Anforderungen die im Kontext von SAP ebenfalls relevant sind.

Es besteht der Wunsch die Programmierichtlinien für ABAP mit denen des CCHH möglichst weit zu harmonisieren.

Überblick SAP ABAP Programmierichtlinien mit Fokus „Sicherheit“

Für die SAP-Systeme der TU Darmstadt gilt entsprechend BSI die „[Standard-Absicherung](#)“. Sofern nicht anders angegeben gilt für die SAP-Systeme die Schutzbedarfskategorie NORMAL.

In Bezug auf die Gefährdungslagen aus [[BSI APP.4.6](#) Seite 2] sind 2.1 bis 2.4 relevant, diese sind:

- „2.1 Fehlende Berechtigungsprüfungen“ im Kontext von Programm-Code
- „2.2 Verlust von Vertraulichkeit oder Integrität von kritischen Daten“
- „2.3 Injection-Schwachstellen“
- „2.4 Umgehung von vorhandenen SAP-Sicherheitsmechanismen“

Zusätzlich gibt es in [[BSI GSHB APP 3.1](#) „Entwicklung von Webanwendungen“] allgemeine Anforderungen. Im Kontext von SAP wird bereits heute diverse Funktionalität im SAP Support-Portal bereitgestellt. Prinzipiell ist es möglich einen Großteil der klassischen ABAP Dynpro Anwendungen ebenfalls über die SAPGUI for HTML über die Webschnittstelle auszuliefern. Mit einer zukünftigen Umstellung auf SAP S/4HANA wird zusätzlich die Fiori Technologie schrittweise SAP GUI Anwendungen sukzessive ablösen. Daher sollten die Anforderungen für die Entwicklung von sicheren Webanwendungen zusätzlich allgemein für sämtliche ABAP Eigen-Entwicklungen gelten.

In Bezug auf die Gefährdungslagen aus [[BSI GSHB APP 3.1](#)] sind 2.1 bis 2.3 relevant, diese sind:

- „2.1. Unzureichende Protokollierung von sicherheitsrelevanten Ereignissen“
- „2.2. Offenlegung sicherheitsrelevanter Informationen bei Webanwendungen und Webservices“
- „2.3. Missbrauch einer Webanwendung durch automatisierte Nutzung“
- „2.4. Unzureichende Authentisierung“

Über die Onapsis Plattform hat die TU-Darmstadt eine Drittanbieter-Lizenz mit der mittels statischer Code-Analyse nach ABAP-Code-Schwachstellen im Produktivsystem gesucht werden kann. Diese wird für das Vulnerability-Management und das kontinuierliche Monitoring von Schwachstellen in Produktion genutzt. In der jetzigen Version ist es nur möglich externe PDF Berichte für die Coding-Schwachstellen zu produzieren, eine eingeschränkte Sicht ist ebenfalls über das Web-Interface der Onapsis Plattform vorhanden. Eine Anzeige der Details (genauer: Link auf die betroffene Coding-Zeile) von den zugehörigen Schwachstellen innerhalb der Entwicklungsumgebung oder eine direkte Integration in das Transportwesen erfordert weitere signifikante Lizenzkosten und ist mit der jetzigen Lizenz nicht abgedeckt. Für die Schutzbedarfskategorie NORMAL ist dies ausreichend, siehe BSI GSHB APP 4.6 - „APP.4.6.A22 Einsatz von ABAP-Codeanalyse Werkzeugen (H)“.

Entwickler können unabhängig das SAP ABAP Testcockpit und den SAP Code-Inspector für Eigenprüfungen innerhalb der SAP Systeme ohne weitere Lizenzkosten nutzen.

Regelwerk - SAP ABAP Programmierrichtlinien mit Fokus „Sicherheit“

Grundsätzlich erwartet die TU Darmstadt, neben den allgemein gültigen Empfehlungen zur Softwareentwicklung (z.B. stabil, verständlich, wiederverwendbar, transportierbar, wartbar, sicher, performant, funktional korrekt, benutzerfreundlich) die Verwendung der offiziellen SAP ABAP Programmierrichtlinien.

https://help.sap.com/doc/abapdocu_750_index_htm/7.50/de-DE/index.htm → ABAP – Schlüsselwortdokumentation → ABAP – Programmierrichtlinien

Für jede Regel gelten Verbindlichkeiten bei den Anforderungen. In Großbuchstaben gesetzte Verben zeigen dabei die Verbindlichkeit einer Anforderung. Die folgende Tabelle gibt hierzu eine Übersicht (siehe auch [BSI Onlinelehre](#)):

Ausdruck	Bedeutung
MUSS, DARF NUR	So gekennzeichnete Anforderungen müssen unbedingt erfüllt werden.
DARF NICHT, DARF KEIN,	Etwas darf in keinem Fall getan werden.
SOLLTE	Dieser Ausdruck bedeutet, dass eine Anforderung zwar normalerweise erfüllt werden muss, bei stichhaltigen Gründen aber auch davon abgesehen werden kann.
SOLLTE NICHT, SOLLTE KEIN	Dieser Ausdruck bedeutet, dass etwas normalerweise nicht getan werden darf, bei stichhaltigen Gründen aber trotzdem erfolgen kann.

Die Prozessdokumentation befindet sich im [TU Signavio Process Collaboration Hub](#).

Regelwerk Allgemeine ABAP Entwicklung:

- [ABAP-GEN-1] Objekte MÜSSEN zukunftsicher entwickelt werden (Modularisierung, einheitliche Strukturierung, Kapselung)
- [ABAP-GEN-2] Es SOLLTE kein SAP-Standard kopiert werden. Es ist bevorzugt SAP Standardfunktionalität zu nutzen („Wiederverwendung“). In Ausnahmefällen kann Code adaptiert werden um Modifikationen zu vermeiden. Diese sind schriftlich abzustimmen.
- [ABAP-GEN-3] Pro Codezeile SOLLTE nur eine Anweisung stehen
- [ABAP-GEN-4] Deklarationen MÜSSEN in dem Kontext deklariert werden, in dem sie für alle potenziellen Verwender sichtbar sind, aber nicht darüber hinaus.
- [ABAP-GEN-5] Zur Formatierung MUSS der Pretty Printer in der Werkseinstellung eingesetzt werden (z.B. Schlüsselwörter in Großbuchstaben, Bezeichner in Kleinbuchstaben, Anweisungen einrücken)
- [ABAP-GEN-6] Programminterne Namen MÜSSEN so verwendet werden, das keine Verwechslung entstehen: Beispiele: g_ globale Datenobjekte, i_ Importing Parameter
- [ABAP-GEN-7] Die Präsentations- und Anwendungslogik MUSS getrennt werden, als auch die Kommunikation mit externen Systemen

- [ABAP-GEN-8] Verlagerung von datenintensiven Operationen SOLLTE auf die Datenbank („Code2Data“) erfolgen
- [ABAP-GEN-9] Paketabhängigkeiten SOLLTEN vermieden werden
- [ABAP-GEN-10] Coding MUSS jederzeit manuell untersuchbar sein, d.h. es DARF KEINE Techniken zur Obfuskation in Eigenentwicklungen genutzt werden. Ein Debugging und Auditierung sowie Fehleranalysen MÜSSEN möglich sein.
- [ABAP-GEN-11] Coding MUSS im Hinblick auf die HANA-Kompatibilität entwickelt werden („HANA Readiness“)
- [ABAP-GEN-12] Es SOLLTEN KEINE harten Codierungen genutzt werden (z.B. für Benutzernamen, Pfade, Magic Numbers, Texte)
- [ABAP-GEN-13] Im Gesamtkontext der Anwendung/Prozesse SOLLTE das Coding performant gestaltet werden, Abweichungen müssen dokumentiert werden und sind z.B. für unkritische Einmalumsetzungen denkbar unter dem Aspekt der Wirtschaftlichkeit
- [ABAP-GEN-14] Werden größere Datenbereiche verarbeitet, MÜSSEN entsprechend Laufzeitanalysen mit ausreichend großen Testdaten durchgeführt werden.
- [ABAP-GEN-15] Datenbankzugriffe SOLLTEN performant gestaltet werden (z.B. Treffermenge/zu übertragende Datenmenge, Zahl der Datenbankzugriff und Suchabfragen sind klein zu halten)
- [ABAP-GEN-16] Eigenentwicklungen MÜSSEN möglichst robust und Fehlertolerant zu entwickeln (z.B. Ausnahmen abfangen, SY(ST)-SUBRC). Zugehörige ggf. kritische Events MÜSSEN die Standardlogmechanismen von SAP nutzen
- [ABAP-GEN-17] Zur Qualitätssicherung ist das Coding spätestens vor Freigabe des Transportauftrages über den SAP Code Inspector, ABAP Testcockpit erfolgreich zu prüfen.

Regelwerk ABAP Sicherheit:

- [ABAP-SEC-1] Es MÜSSEN die Anforderungen des BSI GSHB APP 4.6 Bausteins eingehalten werden:
https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-GS-Kompendium_Einzel_PDFs_2023/06_APP_Anwendungen/APP_4_6_SAP_ABAP_Programmierung_Edition_2023.pdf?__blob=publicationFile&v=4#download=1 – die zugehörigen Umsetzungshinweise unter [Umsetzungshinweise zu APP.4.6](#) MÜSSEN beachtet werden
- [ABAP-SEC-2] Abweichungen von den BSI GSHB APP 4.6 SOLLTE Vorgaben müssen schriftlich im Source-Code als Kommentar dokumentiert werden.
- [ABAP-SEC-3] Bei Web-Entwicklungen SOLLTEN zusätzlich die aktuellen Vorgaben der OWASP Top Ten berücksichtigt werden, siehe <https://owasp.org/www-project-top-ten/> - Abweichungen müssen dokumentiert werden
- [ABAP-SEC-4] Eine Eingabevalidierung MUSS von sämtlichen direkten und indirekten Benutzereingaben erfolgen. Eine indirekte Benutzereingabe ist beispielsweise eine Benutzereingabe in einer Tabelle die von einem anderen Programm als direkte Benutzereingabe erstellt wurde.
 Durch eine konsequente Benutzereingabe werden viele typische Schwachstellen (siehe https://help.sap.com/doc/abapdocu_751_index_htm/7.51/de-de/abendynamic_programming_scrty.htm) bereits eingeschränkt.
 Berücksichtigung der SAP Notes 1520356, 887168, 944279, 822881, 1497003:
 Insbesondere die Klasse CL_ABAP_DYN_PRG sollte zur Mitigation genutzt werden.
- [ABAP-SEC-5] Es MUSS Output-Encoding entsprechend dem Zielkontext durchgeführt werden, d.h. für Cross-Site-Scripting muss beispielsweise im zugehörigen Kontext encodiert werden (HTML, URL, CSS, Javascript, bzw. ggf. eine zusätzliche Validierung je nach Applikationslogik).

-
- [ABAP-SEC-6] SAP Mandantentrennung SOLLTE eingehalten entsprechend BSI GSHB APP 4.6 eingehalten werden, Ausnahmen MÜSSEN dokumentiert werden und sind beispielsweise für technische Umsetzungen wie Upgrades denkbar
 - [ABAP-SEC-7] Beim Zugriff, Export oder der Anzeige und ggf. Übertragung von sensiblen und vertraulichen Daten MUSS eine zusätzliche Berechtigungsprüfung im Coding durchgeführt werden
 - [ABAP-SEC-8] Das Datenbank Logging SOLLTE durch Eigenentwicklungen nicht deaktiviert werden
 - [ABAP-SEC-9] Eigenentwicklungen SOLLTEN keine Änderungen an Business Daten ohne Änderungsbelege ermöglichen
 - [ABAP-SEC-10] Verfügbare Sicherheitsprüfungen der ABAP Workbench MÜSSEN genutzt und die Empfehlungen in Bezug auf Sicherheit SOLLTEN umgesetzt werden (SLIN und Code Inspector). Abweichungen von den Sicherheits-Empfehlungen von SLIN und SAP Code-Inspector MÜSSEN per Kommentar im Source-Code hinterlegt werden
 - [ABAP-SEC-11] Für jede Applikation MUSS ein Berechtigungskonzept existieren
 - [ABAP-SEC-12] Externe Inhalte die über Upload eingebunden werden MÜSSEN eingeschränkt und validiert werden, siehe APP.3.1.A4
 - [ABAP-SEC-13] Die folgenden Programmier-Prinzipien MÜSSEN bei jeder ABAP Entwicklung berücksichtigt werden (siehe APP.3.1.A9)
 - sichere Eingabevalidierung und Ausgabekodierung: dies bedeutet dass sämtliche Benutzereingabe validiert werden müsse,
 - sicheres Session-Management: es soll das SAP Standard Session Management genutzt werden,
 - sichere kryptografische Verfahren: die kryptographischen Vorgaben der TU-Darmstadt müssen berücksichtigt werden,
 - sichere Authentisierungsverfahren: die Authentifizierung ist per Web für normale Benutzer nur über SAML erlaubt, für technische Benutzer gilt aktuell Benutzername und Passwort,
 - sichere Verfahren zum serverseitigen Speichern von Zugangsdaten: Es müssen die SAP Standardmechanismen für das Benutzermanagement genutzt werden, es darf kein eigenes vom SAP Standard abweichendes Benutzermanagement benutzt werden, siehe APP.3.1.A14,
 - geeignetes Berechtigungsmanagement: jede Anwendung benötigt ein Berechtigungskonzept,
 - ausreichende Protokollierungsmöglichkeiten: Fehlerzustände müssen entsprechend der Kritikalität protokolliert werden,
 - regelmäßige Sicherheitsupdates durch den Entwickelnden der Software MUSS durch den Auftraggeber sichergestellt werden,
 - Schutzmechanismen vor verbreiteten Angriffen auf Webanwendungen und Webservices: siehe die Programmierrichtlinie und die OWASP Top 10
 - Schutzmechanismen vor dem Zugriff auf den Quelltext der Webanwendung oder des Webservices, sofern die entwickelnde Anwendung Source-Code bereitstellt, MUSS dieser über die zugehörigen Berechtigungsobjekte wie z.B. S_DEVELOP geschützt werden.

Regelwerk ABAP Namenskonventionen:

- [ABAP-NC-1] Folgende Namensräume sind für jegliche Objekte/Pakete nicht zu verwenden: \$*, T*, \$TMP.
- [ABAP-NC-2] Die Softwarekomponente HOME, LOCAL ist nicht zu verwenden.
- [ABAP-NC-3] Folgende Konventionen MÜSSEN berücksichtigt werden:
 - Paket: ZXXX_ <Modul>_ <Freitext>
 - Im Paket enthaltene Objekte: ZXXX_ <Modul>_ <Freitext>
 - Transaktionen: ZXXX_ <Modul>_ <Freitext>
 - Berechtigungsobjekte: ZXXX_ <Modul>_ <Freitext>

Es SOLLEN bevorzugt eigene SAP Berechtigungsobjekte genutzt werden. In Fällen, in denen eine direkte Zuordnung zu einem SAP Berechtigungsobjekt möglich ist KANN dieses genutzt werden, z.B. S_TABU_DIS/S_TABU_NAM für Tabellenzugriffe.

XXX = Abkürzung der jeweiligen Hochschule, z.B. für die Technische Universität Darmstadt TUD, oder RM für Referenzmodell bei Objekten des CCHH
<MODUL> = Abkürzung für das zugehörige Modul, z.B. „FM“.
Freitext = beliebiger, aussagekräftiger Freitext
Beispiel: ZRM_FM_KONTOAUSZUG

ACHTUNG: Neben den hier verwendeten Namensräumen existieren aktuell auch genauso viele Objekte aus historischen Gründen, welche mit Z_RM* anfangen. Auch dieser Namensraum ist folglich für das Referenzmodell geblockt, SOLLTE aber für Neuimplementierungen nicht mehr verwendet werden!

Regelwerk ABAP Anpassungen SAP Standard, Erweiterungen, Modifikationen:

- [ABAP-ERW-1] Es SOLLEN keine Modifikationen durchgeführt werden (Clean Core Ansatz). Modifikationen MÜSSEN durch den Systemverantwortlichen und die zugehörige verantwortliche SAP Basis Leitung freigegeben werden.
- [ABAP-ERW-2] Anpassungen und Erweiterungen DARF NUR über die von SAP bereitgestellten Erweiterungsmöglichkeiten erfolgen (z.B. User-Exit, Customer-Exit, Enhancement Framework) und MÜSSEN dokumentiert werden.
- [ABAP-ERW-3] Auf Kopien aus dem SAP Standard in den Kundennamensraum SOLLTE verzichtet werden
- [ABAP-ERW-4] Eine bereits vorhandene Anpassung, Erweiterung oder Eigenentwicklung DARF NICHT ohne Zustimmung verändert oder erweitert werden.

Regelwerk ABAP Dokumentation:

- [ABAP-DOC-1] Kommentare im Quellcode sollen anderen Entwicklern das Verstehen des Quellcodes erleichtern. Im Programmkopf MUSS als Kommentar das Datum/Änderungsdatum, Autor, Ticket (Auftrag/ Projekt/ Anwendung/ Prozess -> „Was“ und „Wofür“) zu hinterlegen. Die Abnahme der Dokumentation ist wesentlicher Bestandteil vor der Produktivsetzung.
- [ABAP-DOC-2-DRAFT] (Zukünftige geplante Regel) Für jeden Transport in den Systemen des CCHH mit Eigenentwicklungen MUSS die folgende Transportdokumentation vom Entwickler gepflegt werden:



Vorlage_Transportdokumentation.docx

- [ABAP-DOC-3] Für umfangreiche Eigenentwicklungen SOLL ein Fachfeinkonzept erstellt werden, welches sich an der Vorlage orientiert.



Vorlage_Entwicklung.docx

- [ABAP-DOC-4] Für die Ausnahmefälle von Modifikationen MUSS ein Fachfeinkonzept erstellt werden, siehe [ABAP-DOC-3] und [ABAP-ERW-1].