Leitfaden

Einsatz des ABAP Test Cockpit (ATC)

Inhaltsverzeichnis

1 Einleitung

Die professionelle Software-Entwicklung verfolgt das Ziel, das erstellte Produkt in hoher Qualität zu liefern. Dies ist insbesondere im Bereich der kundenindividuellen Erweiterungen oder Individualentwicklungen auf Basis von SAP ein essenzieller Aspekt, um die Geschäftsprozesse im SAP-System nicht zu gefährden. Ein Baustein dafür stellt die statische Code-Analyse dar. Das Mittel der Wahl im Bereich der ABAP-Entwicklung ist das ABAP Test Cockpit (ATC), das eine statische Überprüfung des Codes nach unterschiedlichen Gesichtspunkten erlaubt.

Neben der „klassischen“ Qualitätssicherung, wie die statische Code-Prüfung auf eine robuste und performante Programmierung, hat sich in den letzten Jahren ein weiterer „Qualitätsaspekt“ für die Entwicklung von ABAP Code ergeben: Die Readiness des kundenindividuellen Codes für die Konvertierung nach SAP S/4HANA oder für die Überführung in das SAP-Cloud-Platform-ABAP-Environment. Auch hier setzt SAP auf das ATC, was die zentrale Bedeutung des Tools unterstreicht. Hier ist zu bedenken, dass die Readiness keine einmalige Aktion ist, sondern die fortschreitende Simplifizierung in S/4HANA eine kontinuierliche Überprüfung des Codes mit jedem neuen Release bedingt.

Das Ziel des Dokuments ist es daher, durch beschriebene Best-Practices die Einführung und den Einsatz des ATC so effizient und effektiv wie möglich für Ihr Unternehmen zu gestalten.

1.1 Motivation

Die Arbeit der Deutschsprachigen SAP-Anwendergruppe e. V. (DSAG) fußt auf drei Säulen: Wissensvorsprung, Einflussnahme und Netzwerk. Das vorliegende Dokument wurde von Mitgliedern des DSAG-Arbeitskreises Development initiiert und adressiert die erste Säule, den Wissensvorsprung für Anwender und Partner.

Als Autoren-Team ist es unser Anliegen, das in den Unternehmen gesammelte Wissen zum ATC in einem kompakten Dokument zur Verfügung zu stellen. Allerdings wird das ATC kontinuierlich weiterentwickelt, und auch die Erfahrungen im Umgang mit dem Tool werden bei den DSAG-Mitgliedern zunehmen. Vor diesem Hintergrund ist es unser Ziel, dass auch dieses Dokument „lebt“ und sich durch Ihren Erfahrungsschatz kontinuierlich verbessert.

Sie haben zwei Optionen, uns Feedback zukommen zu lassen. Zum einen können Sie im DSAGNet im Bereich des Arbeitskreises Development einen Beitrag veröffentlichen. Zum anderen stellen wir mit dem Leitfaden Code-Beispiele im GitHub-Account der DSAG zur Verfügung (https://github.com/1DSAG/ATC-Best-Practice-Guide). Dort können Sie den GitHub-Mechanismus von Issues für Ihre Anmerkungen nutzen. Wir freuen uns auf Ihre Rückmeldung!

1.2 Positionierung

Von SAP und auch von einigen Fachverlagen existieren bereits sehr gute Publikationen zum ATC und dessen Einsatz im Bereich der Qualitätssicherung und der Konversion nach S/4HANA. Im Verlauf des Leitfadens verweisen wir auf aus unserer Sicht lesenswerte Literatur aus diesen Quellen.

Der Mehrwert dieses Dokuments liegt daher in der Zusammenfassung bewährter Vorgehensweisen, Praxistipps und Dos und Don’ts aus den Anwenderunternehmen. Der Leitfaden soll Ihnen als Anwender, Entwickler, Entwicklungs-, Projekt- oder IT-Leiter Anregungen und Hilfestellungen geben, um auf den Erfahrungen anderer aufbauen zu können und von den daraus destillierten Best-Practices zu profitieren. Die in diesem Dokument vorgestellten Empfehlungen erheben nicht den Anspruch auf Vollständigkeit oder absolute Gültigkeit, sondern repräsentieren eine Auswahl der Praxistipps des Autoren-Teams.

Wir haben das Ziel verfolgt, die richtige Mischung aus Überblickswissen und Detailtiefe zum Thema ATC zu Papier zu bringen. Aus diesem Grund verweisen wir an den entsprechenden Stellen auf weiterführende Quellen, um das Dokument nicht zu überfrachten und der Leserschaft trotzdem die Möglichkeit zu bieten, bei Bedarf detaillierte und weiterführende Informationen zu finden.

1.3 Gender-Erklärung

Aus Gründen der besseren Lesbarkeit wird in diesem Leitfaden die Sprachform des generischen Maskulinums angewandt. Es wird an dieser Stelle darauf hingewiesen, dass die ausschließliche Verwendung der männlichen Form geschlechtsunabhängig verstanden werden soll.

2 Motivation

Bei der Erstellung von Software treten in der Regel Fehler auf, die sich schon während der Entwicklung vermeiden lassen. Häufige Fehler sind zum Beispiel:

- keine Überprüfung der übergebenen Tabelle bei einem SELECT … FOR ALL ENTRIES
- Prüfung auf SY-SUBRC nach Aufruf von Funktionsbausteinen zur Behandlung von Fehlern
- Typkonflikte, weil z. B. bei Unterprogrammaufrufen die Typen der Parameter fehlen
- MESSAGE-Befehl mit falscher Anzahl Parametern

Werden solche Fehler in eine Produktivumgebung eingespielt, kommt es in Konsequenz immer wieder zu Programmabbrüchen, Korruption von Daten und/oder Performance-Problemen. Für den Kunden selbst bedeuten solche Fehler im schlimmsten Fall einen Produktionsstillstand, der mit beträchtlichen Umsatzeinbußen und sogar Image-Verlusten einhergehen kann. Der Software-Dienstleister auf der anderen Seite sieht sich mit der gleichen Art von Problemen konfrontiert. Eskalationen, wenn „das Band stillsteht“, sind für keinen Beteiligten ein Vergnügen!

Neben diesen potenziellen „harten“ Fehlern gibt es auch „weiche“ Qualitätsprobleme, die zwar nur mit geringer Wahrscheinlichkeit zu direkten Fehlern führen, aber die Wartungsrisiken und damit -kosten der Programme erhöhen:

- Verwendung obsoleter Sprachelemente
- Prozeduren, die implementiert bzw. deklariert sind, aber gar nicht benutzt werden
- Parameter oder Variablen, die implementiert bzw. deklariert sind, aber gar nicht benutzt oder nur gelesen, aber nie befüllt werden
- überflüssige Programmabschnitte, z. B. IF ENDIF ohne Anweisungen dazwischen, welche auch nicht verwendet werden

Mit Hilfe des in die ABAP Workbench und die ABAP Development Tools integrierten Frameworks ABAP Test Cockpit (ATC) können Sie derartige Probleme ohne größeren Aufwand erkennen und im Entwicklungsprozess die notwendigen Korrekturen vornehmen. Dabei enthält das ATC schon in der Standardkonfiguration eine Vielzahl an statischen Code-Tests. Falls Sie ABAP Unit Tests einsetzen, können die Tests automatisch im Rahmen einer Prüfung ausgeführt werden. Zusätzlich gibt es z. B. in GitHub etliche außerhalb von SAP programmierte, frei verfügbare Prüfklassen, und bei Bedarf können Sie auch eigene implementieren.

Zusätzlich kann das ATC auch dafür genutzt werden, bestimmte Programmierstile zu etablieren. Hierzu gehören Themen wie Definition von Modularisierungsobjekten und

deren Größe (Methoden, Funktionsbausteine, Unterprogramme usw.), Verhältnis der Anzahl von Code- und Kommentarzeilen, Schachtelungstiefe in Modularisierungseinheiten, leicht lesbare Kontrollstrukturen (Schachtelung von CASE- und/oder IF-Konstrukten) und auch Namenskonventionen.

2.1 Abgrenzung: Code Inspector (SCI) vs. CHECKMAN

Das ATC kommt mit zwei „Flavors“, welche sich je nach Einstellung des Systemtyps (Systemparameter transport/systemtype in Transaktion RZ11) unterschiedlich verhalten:

- CUSTOMER: Dies ist vermutlich der gebräuchlichste Fall; ATC verwendet hier den SCI für die Definition der Prüfvarianten
- SAP: In der standardnahen Entwicklung durch SAP-Partner ist diese Einstellung durchaus gebräuchlich, da durch Auflagen im Rahmen einer Zertifizierung die Einstellung erforderlich ist. Intern verwendet das ATC in diesem Fall den CHECKMAN.

2.2 Remote- vs. Non-Remote-ATC

Das ATC lässt sich in zwei Varianten aufsetzen: Remote- oder Non-Remote-ATC. Die Varianten unterscheiden sich in erster Linie dadurch, wo die Code-Prüfungen erfolgen. Je nachdem ergeben sich unterschiedliche Möglichkeiten und Einschränkungen.

Die **Non-Remote-**ATC-Prüfungen gibt es dabei schon länger, während die Remote-Prüfungen noch relativ neu sind.

Die Non-Remote-Prüfungen finden meist in den Entwicklungs- und Qualitätssicherungssystemen einer Systemlandschaft statt. Als Qualitäts-Manager können Sie so regelmäßige Massentests auf dem Qualitätssicherungssystem durchführen und die Ergebnisse in ein Entwicklungssystem replizieren. Es ist auch einstellbar, dass die (befundfreien) Prüfungen Voraussetzung für jede Transportfreigabe sind. Als Entwickler können bzw. müssen Sie diese Ergebnisse dann zur Verbesserung des Codes nutzen, wobei Sie für Befunde auch Befreiungen erstellen können. Das Qualitätssicherungssystem wird in diesem Zusammenhang in der SAP-Hilfe auch als ATC-Master-System bezeichnet, was jedoch nicht zu verwechseln ist mit dem zentralen ATC (Master)-Prüfsystem, welches nun erläutert wird.

Beim **Remote-ATC** haben Sie ein zentrales ATC-System, das die Entwicklungsobjekte auf verschiedenen sogenannten Satellitensystemen per RFC-Verbindung prüft.

Der große Vorteil ist, dass Sie ein zu prüfendes System in Ihrer Landschaft nicht erst auf einen „aktuellen“ Release-Stand bringen müssen, um „aktuelle“ Prüfungen auf den dortigen Entwicklungsobjekten ausführen zu können.

SAP hat die wesentlichen ATC-Prüfungen RFC-fähig gemacht. Folgende Prüfungen sind bisher noch nicht verfügbar[1]:

- aus dem Bereich Performance: Verschachtelte Zugriffe auf interne Tabellen, Kopieren der aktuellen Tabellenzeile bei LOOP AT, Tabelleneigenschaften
- mehrere Prüfungen aus dem Bereich Syntaxprüfung/Generierung, z. B. Paketprüfungen
- alle Metriken
- Unit Tests
- Prüfungen von Oberflächen (Dynpro und Web Dynpro)

Ansonsten ist die Funktionsweise wie beim Non-Remote-ATC.

Sie können Befreiungen beantragen, welche auf dem zentralen System bearbeitet werden müssen.

Sie können zentrale Prüfläufe durchführen, wobei die Ergebnisse immer auf dem zentralen Prüf-Server bleiben. Eine Replikation der Ergebnisse ist nicht vorgesehen und daher technisch nicht möglich.

Auch bei dieser Variante kann das System so konfiguriert werden, dass bei Task und/oder Transportfreigabe auf den Entwicklungssystemen via RFC eine Prüfung mit einer Prüfvariante auf dem zentralen ATC-Server durchgeführt wird. Hier wird in der zentralen Prüfvariante definiert, welche Befunde mit welchen Prioritäten belegt werden. Somit können Sie eine zentrale Variante definieren, um auf dem zentralen System Prüfungen der Entwicklungssysteme durchzuführen. Die Entwicklungssysteme können dabei auch unterschiedliche Release-Stände haben.

Bei beiden Konzepten können die Entwickler ATC-Prüfungen im Entwicklungssystem per SE80 oder in den ABAP Development Tools (ADT) nach Bedarf (ad hoc) selbst durchführen. Je nach Variante erfolgt die Prüfung auf dem lokalen System oder per Remote-Prüfvariante auf dem zentralen ATC-Prüfserver.

Folgende Faktoren können die Entscheidung „Remote“ oder „Non-Remote“ beeinflussen:

- Es sind mehrere Entwicklungssysteme vorhanden. Durch ein zentrales System müssen Einstellungen nur noch einmal vorgenommen werden.
- Die Entwicklungssysteme sind auf einem niedrigen Release-Stand (ab 7.02). Es sollen aber umfangreichere Prüfungen, z. B. in Vorbereitung auf eine HANA-DB- oder S/4HANA-Umstellung, durchgeführt werden.
- Funktionalitäten wie die Baseline sollen genutzt werden können, um die Anzahl der Meldungen in einem vertretbaren Rahmen zu halten.

[1] Stand: Release 7.52 SP4

Nehmen Sie Alt-Befunde in die Baseline auf, so können Sie sich bei weiteren ATC-Läufen tatsächlich auf Befunde zu neuem oder geändertem Coding fokussieren. Weitere Infos zur Baseline finden Sie im Blog-Artikel von Olga Dolinskaja:
https://blogs.sap.com/2016/12/13/remote-code-analysis-in-atc-working-with-baseline-to-suppress-findings-in-old-legacy-code/ (abgerufen am 10.05.2019)

2.3 ATC- und S/4HANA-Konvertierung

Mit der Einführung von S/4HANA werden nicht nur neue Funktionalitäten bereitgestellt, sondern auch viele bestehende Komponenten unter dem Stichwort „Simplification“ modernisiert.

Ihre Eigenentwicklungen, User-Exits, Enhancements oder Modifikationen sind u. U. auf die in S/4HANA neuen Datenmodelle und Zugriffslogiken angewiesen und müssen deshalb ggf. angepasst werden.

Falls Sie eine S/4HANA Konvertierung planen, sollten Sie sich am besten schon frühzeitig damit auseinandersetzen, welche Anpassungen in Ihren Systemen notwendig sind.

Dafür können Sie ein zentrales ATC-System nutzen, das den Code Ihres zu konvertierenden Systems schon vor der Migration auf „S/4HANA-Readiness“ überprüft.

SAP stellt dafür eine Simplification Database bereit, die für alle relevanten Fälle Prüfungen und Hinweise zur Aktualisierung enthält. Details zur Aktualisierung der Simplification Database finden Sie im Hinweis 2241080.

Wie bereits in der Einleitung erwähnt, stellt diese Überprüfung keine einmalige Aktion dar. Sie sollten regelmäßig, u.a. auch bei S/4HANA-Upgrades, die Simplifizierungen enthalten, ihre Code-Basis mit einer aktualisierten Version der Simplification Database überprüfen.

Mit der Aktivierung des ABAP Call Monitors (SCMON) im Produktivsystem erhalten Sie eine detaillierte Übersicht über die tatsächlich genutzten Programme und Funktionen. Zudem sollten Sie die Aggregation der so gesammelten Daten per SUSG aktivieren, damit diese nicht nach sieben Tagen verlorengehen. Im Vergleich zum Vorgänger Usage Procedure Logging (UPL) wird nicht nur das aufgerufene Objekt, sondern werden auch Daten zum Aufrufer gespeichert. Somit können Sie das UPL durch SCMON und SUSG ersetzen.

Damit können Sie Ihre Aktivitäten priorisieren und herausfinden, welcher Code nicht mehr relevant für Ihre Geschäftsprozesse ist. Übernehmen Sie nur Custom-Code, den Sie wirklich benötigen; vermeiden Sie „toten“ Code.

Nach der Konvertierung Ihres Systems nach S/4HANA werden Sie bei den Code-Anpassungen durch Eclipse ADT mit Quick-Fixes unterstützt, die für ausgewählte Fälle die technische Anpassung Ihres Codes automatisieren. Weiterhin stellt SAP die Fiori-App Custom Code Migration zur Verfügung, die Sie bei der Abarbeitung unterstützen kann. Diese App setzt ein zentrales ATC-System ab S/4HANA 1809 voraus.

Weitere Details und Tipps zur Einbindung der weiteren Tools, wie SQLM, SUSG und auch der Custom Code Migration, finden Sie in Kapitel 8.

Link zur SAP-Dokumentation „Migration der Eigenentwicklungen“:

https://help.sap.com/viewer/7bfe8cdcfbb040dcb6702dada8c3e2f0/7.5.4/de-DE/0bb83ef76fbf46c89fd9fa5f3af8c0c6.html (abgerufen am 10.05.2019)

2.4 ATC und ABAP in der Cloud

Mit der neuen Prüfvariante SAP_CP_READINESS_REMOTE können Sie Ihren eigenen Code auf Ausführungsbereitschaft in der ABAP-Umgebung der SAP Cloud Platform (auch bekannt als Steampunk) prüfen. Die Cloud-Readiness-Prüfungen erkennen die folgenden Inkompatibilitäten:

- Verwendung von nicht unterstützten Entwicklungsobjekten (z. B. Dynpros, Reports etc.)
- Verwendung von Entwicklungsobjekten, die nicht per Whitelist freigegeben sind
- Verwendung von Sprachelementen, die nicht im eingeschränkten ABAP-Sprachumfang für SAP Cloud Platform liegen

Zur Verwendung dieser Prüfungen wird die Komponente SAP_BASIS mindestens in der Version 7.52 vorausgesetzt.

Weitere Informationen dazu finden Sie im Blog:
https://blogs.sap.com/2018/10/02/how-to-check-your-custom-abap-code-for-sap-cloud-platform-abap-environment/ (abgerufen am 10.05.2019)

2.5 Verfügbarkeit

Das ATC steht ab folgenden SAP-NetWeaver-Releases zur Verfügung:

- SAP NetWeaver 7.0 EHP2 Support Package 12
- SAP NetWeaver 7.3 EHP1 Support Package 5
- SAP NetWeaver 7.4 Erst-Release

(Quelle: help.sap.com; „ABAP Test- und Analysewerkzeuge“/Qualitätsprüfung mit dem ABAP Test Cockpit (ATC)).

Die Verwendung von Remote-Prüfungen setzen einen ABAP-Server mit SAP NetWeaver 7.51 voraus, von dem aus die Zielsysteme per RFC geprüft werden können.

3 Deployment-Optionen und technisches Set-up

3.1 Allgemeines

Das ABAP Test Cockpit wird mit den folgenden Release-Ständen eingeführt:

- SAP NetWeaver 7.0, EhP2, Support Package 12
- SAP NetWeaver 7.0, EhP3, Support Package 05
- SAP NetWeaver 7.3, EhP1, Support Package 05
- SAP NetWeaver 7.4, Support Package 02

Das ATC ist vollständig in den Object Navigator und den Transport Organizer integriert und steht für entwicklungsbegleitende Tests zur Verfügung. Das ATC unterstützt sowohl Entwickler als auch Qualitäts-Manager. Entwickler können Entwicklungsobjekte im Entwicklungssystem testen. Qualitäts-Manager führen Überprüfungen im Rahmen eines Massenlaufs mit der gleichen ATC-Konfiguration im Qualitätssicherungssystem durch.

Die Funktionalität der Remote-ATC-Prüfungen ist ab SAP NetWeaver AS ABAP 7.51 Innovation Package verfügbar.

Weiterhin können Sie das ATC als Quality-Gate bei der Transportfreigabe oder Freigabe einer Aufgabe[2] konfigurieren und den Transport von Objekten so lange verhindern, bis alle entsprechenden ATC-Befunde korrigiert oder befreit worden sind.

Das ATC ersetzt ab SAP Solution Manager 7.1 SP12 den Code Inspector (SCI) als Werkzeug zur Sicherstellung der Code-Qualität und ist in den ChaRM- und CCLM-Prozess integriert. Der SCI dient aber weiterhin als Basis für die Definition von Prüfvarianten, die vom ATC verwendet werden.

Das ATC lässt sich auf unterschiedliche Arten in Ihre Systemlandschaft integrieren:

- Option 1 – Non-Remote-ATC mit einem System

  In diesem Szenario wird ein System sowohl für die Durchführung und die Administration des ATC als auch für die Behebung der Befunde aus dem ATC verwendet. Der zu prüfende Code liegt auf dem gleichen System wie das ATC. Dies ist typischerweise das Entwicklungssystem.

- Option 2 – Non-Remote-ATC mit einem Master-System

  In diesem Szenario erfolgen die ATC-Prüfungen zentral in einem ATC-Master-System. Der zu prüfende Code liegt physikalisch in diesem System. Typischerweise ist dies das Qualitätssicherungssystem. Die Befunde werden aus dem ATC-Master-System in ein Satellitensystem repliziert und stehen dort

[2] Siehe OSS Note: 2495410.

für die weitere Bearbeitung zur Verfügung.
Das Satellitensystem ist typischerweise das Entwicklungssystem. Die Behebung der Befunde muss in diesem Szenario also erst in das Master-System transportiert werden, um die Prüfergebnisse zu aktualisieren. Des Weiteren finden die Genehmigung von Ausnahmen im ATC-Master-System statt.

- Option 3 – Remote-ATC mit einem zentralen Prüfsystem
  In diesem Szenario ist das ATC-System ein zentrales, dediziertes System, das als zentrales Prüfsystem bezeichnet wird. Im Unterschied zu den Optionen 1 und 2 liegt der zu prüfende Code nicht in diesem System. Die zu prüfenden Systeme sind remote an das zentrale Prüfsystem angebunden und werden daher als Satellitensysteme bezeichnet.

Die folgenden Abschnitte zeigen Ihnen das Set-up der Deployment-Optionen und liefern Ihnen weiterführende Informationen, wie zum Beispiel Nutzung des ATC in Verbindung mit ChaRM Solution Manager.

3.2 Set-up Non-Remote-ATC

Die Optionen 1 und 2 kommen nur dann in Betracht, wenn Sie nicht auf einen zentralen ATC-Server umstellen können. Sie haben einige technische Nachteile und führen zu einem erhöhten administrativen Aufwand im Daily-Business, da Sie z. B. auf jedem System eigene Prüfvarianten definieren müssen. Außerdem können Sie auf älteren Systemen bestimmte Prüfungen nicht implementieren.

Sollten Sie jedoch zwingend eine der Optionen nutzen müssen, dann möchten wir hier auf die Blogs von Olga Dolinskaja verweisen, in denen detailliert der technische Aufbau und die Administration der Szenarien beschrieben werden.

ATC-Introduction:

https://blogs.sap.com/2012/09/19/abap-test-cockpit-an-introduction-to-sap-s-new-abap-quality-assurance-tool/ (abgerufen am 28.10.2019)

ATC-Administration:

https://blogs.sap.com/2012/10/19/getting-started-with-the-abap-test-cockpit-for-qms-and-admins/ (abgerufen am 28.10.2019)

ATC für den Developer:

https://blogs.sap.com/2012/10/18/getting-started-with-the-abap-test-cockpit-for-developers/ (abgerufen am 28.10.2019)

Falls Sie in diesem Set-up eine Migration auf die HANA-DB durchführen wollen, ist die Möglichkeit einer statischen Remote-Prüfung ggf. von Bedeutung. Sie finden weitere Informationen dazu auf **help.sap.com** unter: https://help.sap.com/viewer/7bfe8cdcfbb040dcb6702dada8c3e2f0/7.5.4/de-DE/550a16acb2de445a85e96b5406cf6ee7.html (abgerufen am 29.10.2019)

3.3 Set-up Remote-ATC mit zentralem Prüfsystem

Führen Sie folgende Schritte aus, um das Remote-ATC-Szenario mit einem zentralen Prüfsystem zu konfigurieren.

1. Definition der RFC-Verbindung vom zu prüfenden Satellitensystem zum zentralen Prüfsystem und umgekehrt (Transaktion SM59).
   Sie sollten die RFC-Verbindung immer in Englisch aufbauen. Folgende Punkte sollten Sie beachten:

   - das Feld „Current User“ markieren
   - Trust Relationship auf „Yes“ stellen Voraussetzung: Die User, die vom Satellitensystem auf das zentrale Prüfsystem zugreifen (zum Beispiel bei der Auftragsfreigabe), müssen im zentralen Prüfsystem existieren.

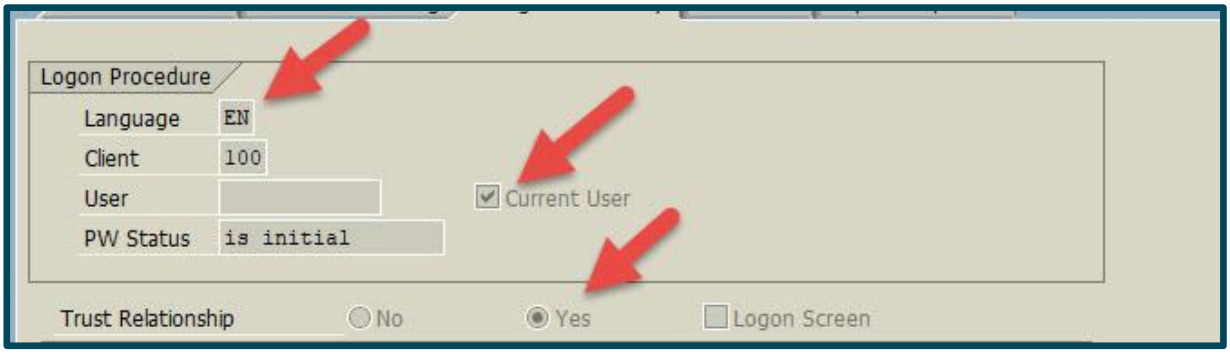


Abbildung 1 RFC Verbindung – Logon

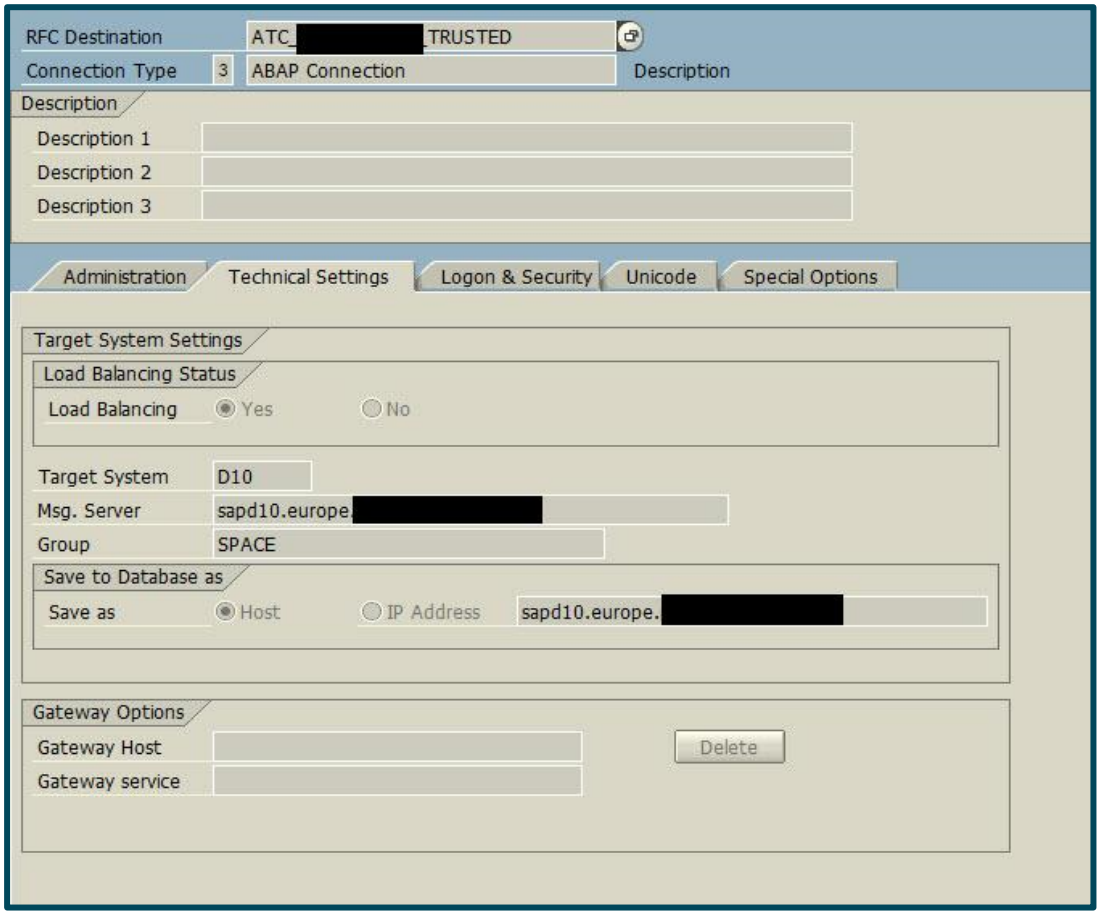


Abbildung 2 RFC Verbindung – Technical Settings

Weitere Informationen dazu finden Sie unter:
https://help.sap.com/saphelp_snc700_ehp01/helpdata/de/9e/641866d4d74e44823c6c1a49626d9b/content.htm?no_cache=true (abgerufen am 04.10.2019)

2. Im Global Customizing des Transport Organizer im Satellitensystem (Transaktion SE03), muss eingestellt werden, wie sich das Transport-Management bei Transportaufgabe-/Transportauftrag-Freigabe verhalten soll. In unserem Beispiel haben wir eingestellt, dass grundsätzlich bei Freigabe einer Transportaufgabe geprüft wird. Dies ist auch der Best-Practice-Ansatz, da unterschiedliche Entwickler an einem Auftrag arbeiten können und die Anzahl der zu prüfenden Objekte damit klein gehalten wird.

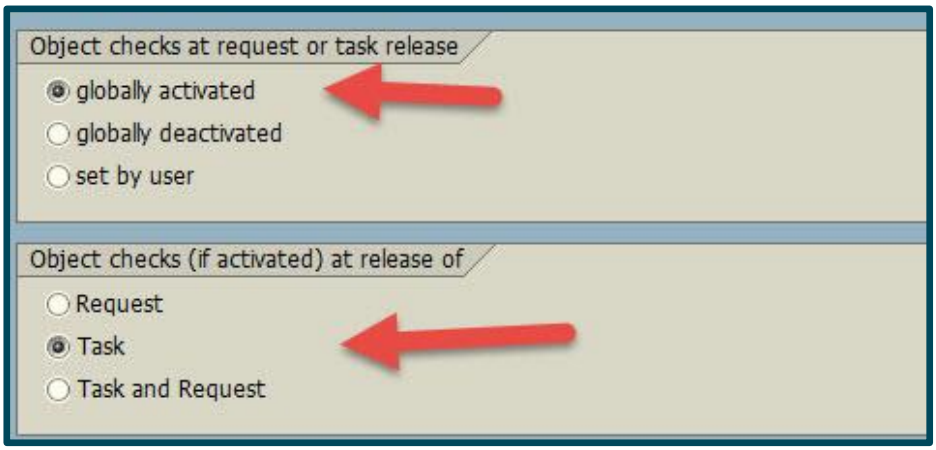


Abbildung 3 Transport Organizer

3. Stellen Sie in der ATC-Administration (Transaktion ATC) des zentralen Prüfsystems die System-Role auf den Wert „ATC Checks by Object Providers“. Damit legen Sie fest, dass in diesem System nur Prüfungen gegen andere Entwicklungssysteme (Satellitensysteme) durchgeführt werden.

Abbildung 4 System Rolle

4. Danach müssen Sie die „Basic Settings“ hinsichtlich der Behandlung von Ausnahmen (Exemptions) setzen. Sie konfigurieren so, ob Sie grundsätzlich mit Ausnahmen arbeiten wollen und für welche Art von Befunden Sie Ausnahmen genehmigen können.

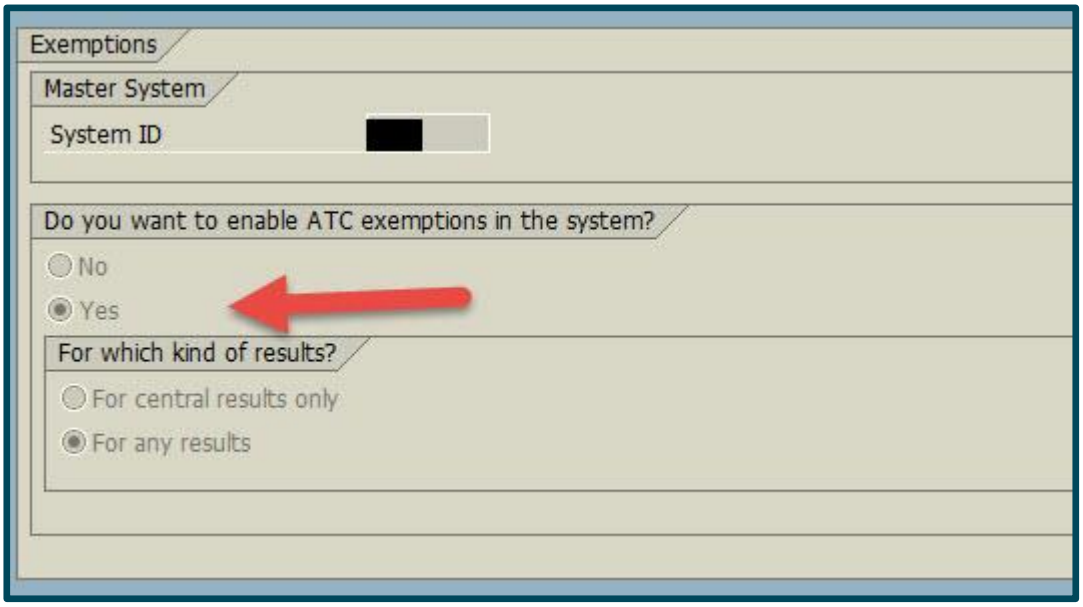


Abbildung 5 Basic Settings

5. Zusätzlich können Sie über den Menüpunkt „Admissable Exemption Requests“ definieren, für welche Befunde auf welcher Ebene eine Befreiung beantragt werden kann (nur der einzelne Befund, das Objekt bzw. Subobjekt oder auf Paketebene). Sie können die von SAP vorgegebenen Einstellungen übernehmen oder entsprechend der Anforderungen in Ihrem Unternehmen nachjustieren.

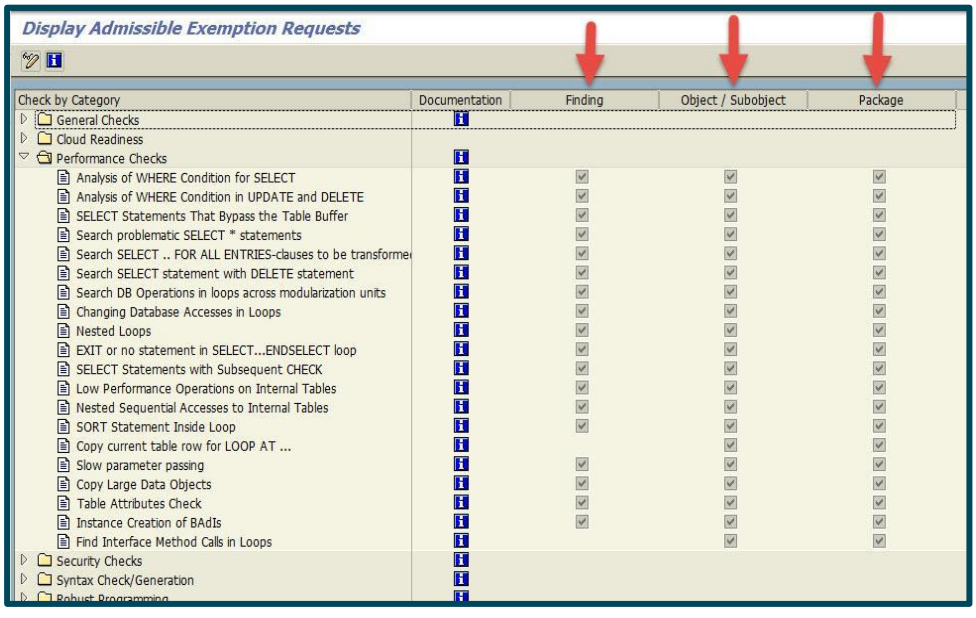


Abbildung 6 Admissable Exemption Requests

6. Machen Sie über den Menüpunkt „RFC Object Providers“ die Satellitensysteme mit dem zentralen Prüfsystem bekannt. Voraussetzung dafür ist, dass die Punkte 1 bis 3 abgearbeitet wurden.

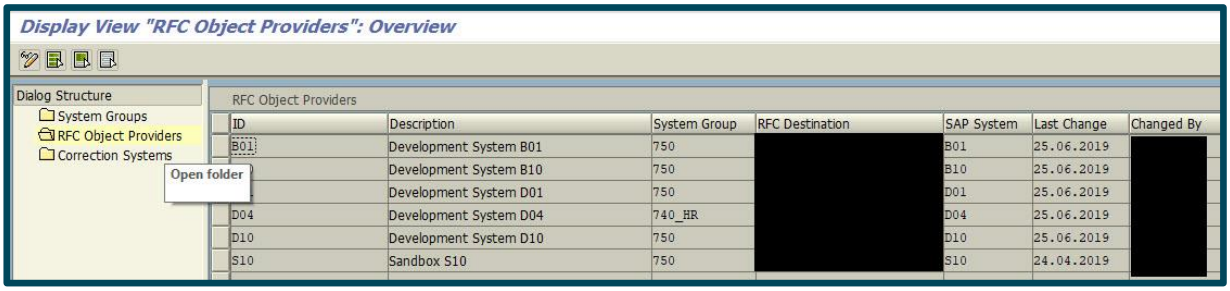


Abbildung 7 RFC Object Providers

7. Sie haben die Möglichkeit, zusätzliche Funktionalitäten auf dem zentralen Prüfsystem zu aktivieren bzw. zu konfigurieren. Sie können zum Beispiel E-Mail-Jobs konfigurieren und festlegen, ob E-Mails für neu angeforderte Befreiungen versendet werden sollen (1). Sie können so auch festlegen, ob eine E-Mail zur Genehmigung der Ausnahme sofort oder in einem anderen Zyklus versendet werden soll (2).

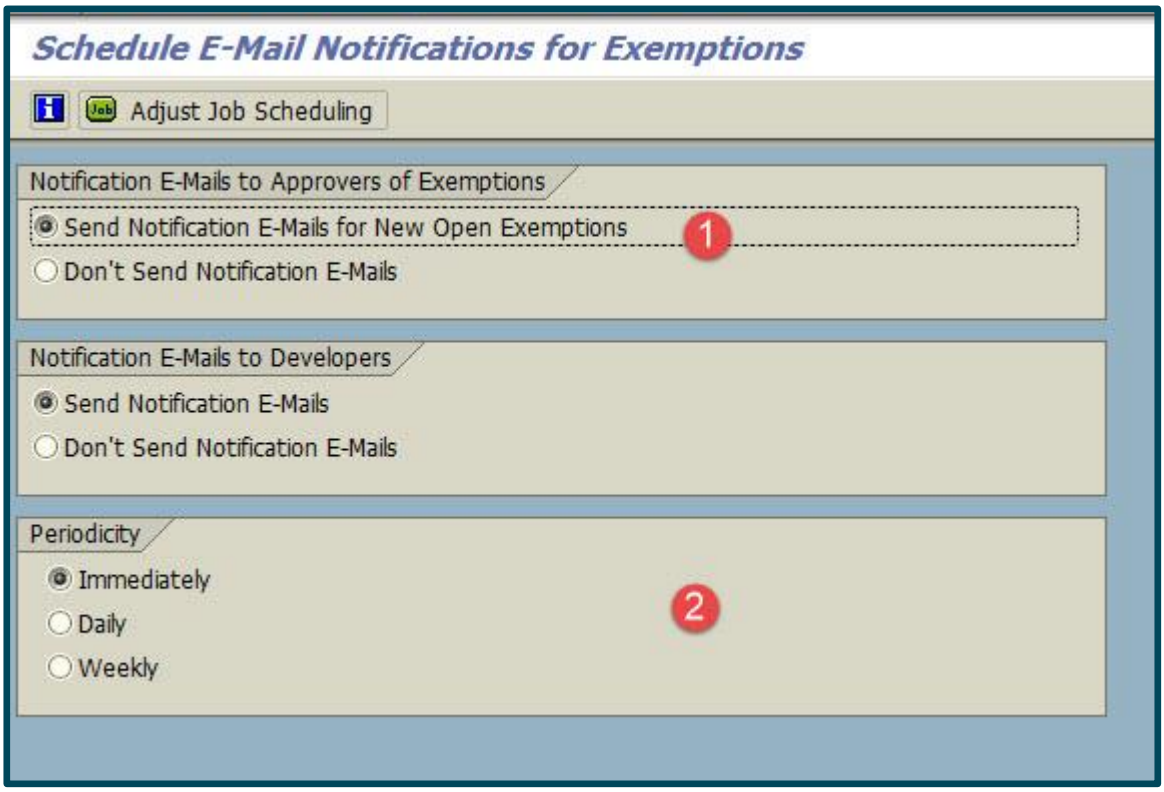


Abbildung 8 E-Mail Notification

Ebenso haben Sie die Möglichkeit, an dieser Stelle die Qualitäts-Manager für die Genehmigung der Ausnahme zu pflegen.

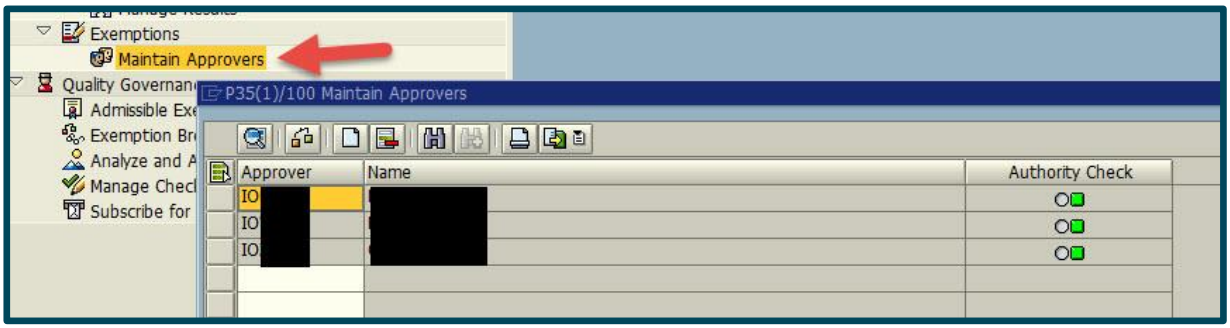


Abbildung 9 Maintain Approvers

8. Erstellen Sie im Code Inspector (Transaktion SCI oder aus ATC heraus) auf dem zentralen Prüfsystem eine globale Prüfvariante, die ausschließlich RFC-fähige Prüfungen beinhaltet. Ansonsten kann die Variante nicht in den Satellitensystemen als Remote-Prüfvariante ausgewählt werden. Prüfungen mit einer markierten Checkbox „RFC-based“ erfüllen dieses Kriterium.

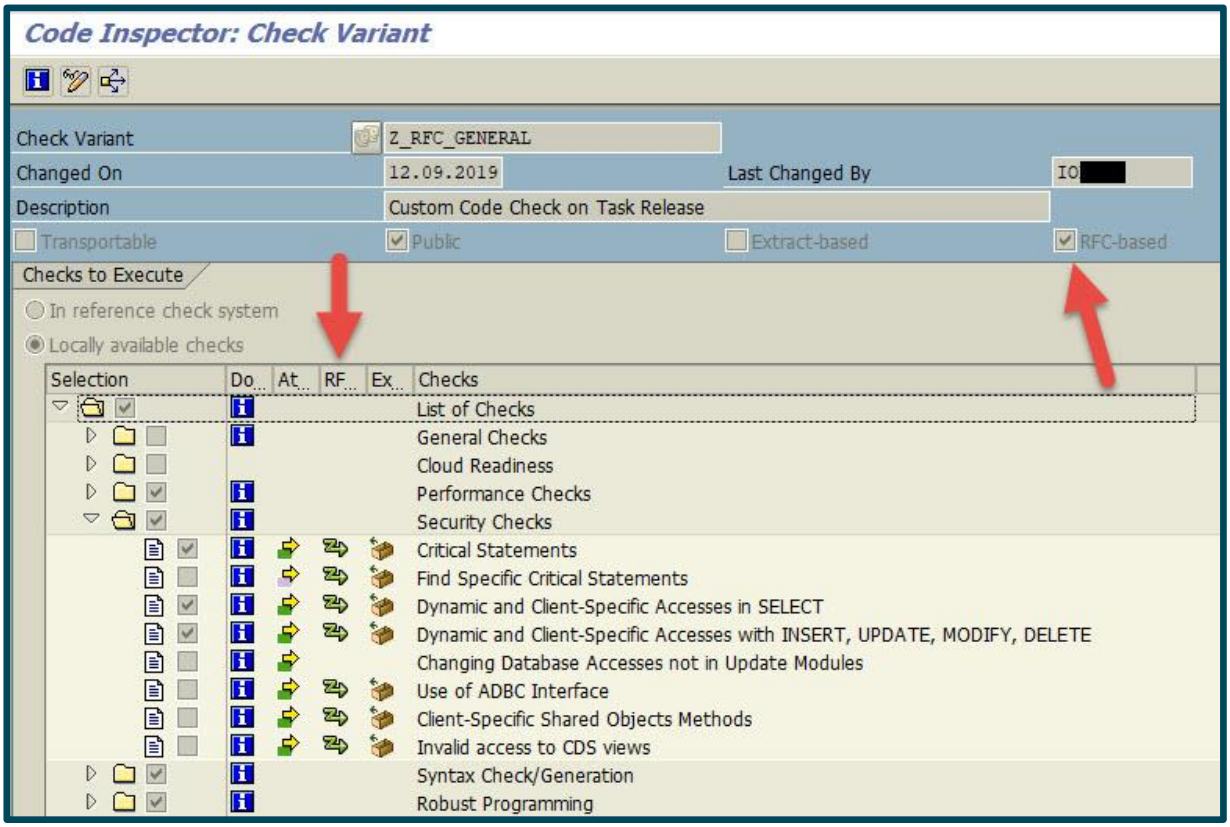


Abbildung 10 Check Variant

9. Hinterlegen Sie auf dem Satellitensystem im Code Inspector (SCI) noch das „Referenzsystem zum Prüfen“.

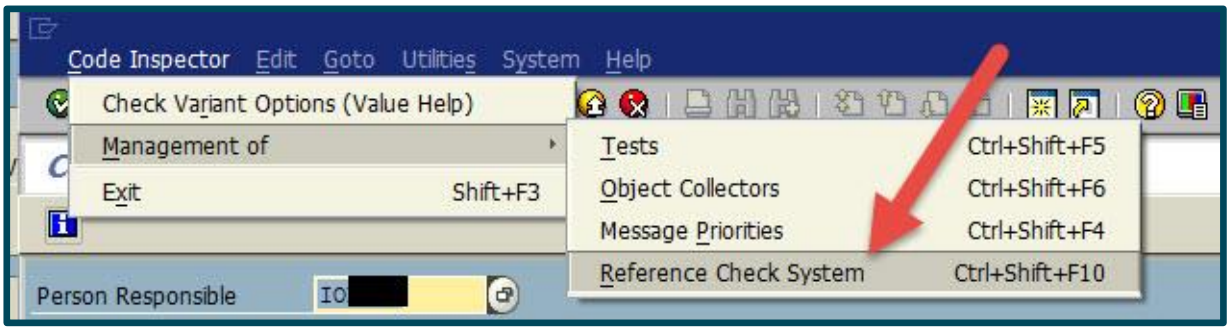


Abbildung 11 Reference Check System

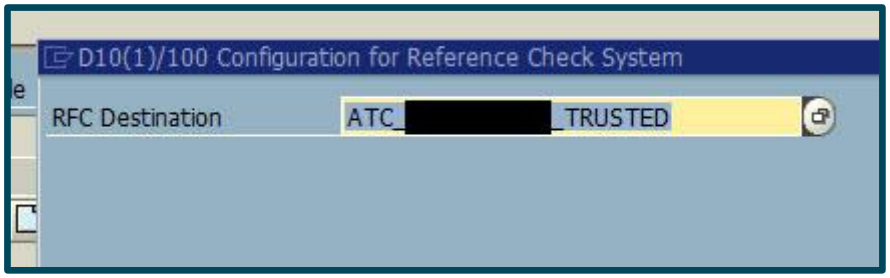


Abbildung 12 Reference Check System – RFC Destination

10. Erstellen Sie auf dem Satellitensystem eine Prüfvariante, die auf die im zentralen Prüfsystem erstellte Prüfvariante verweist. Das System prüft, ob eine RFC-Verbindung zum zentralen Prüfsystem vorhanden ist und der User die notwendigen Berechtigungen hat.

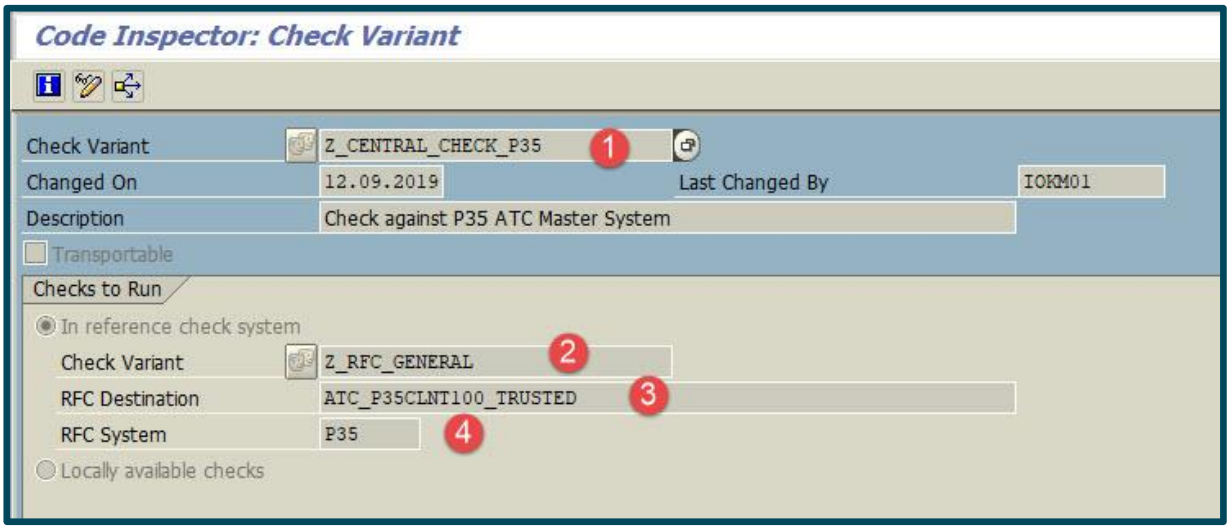


Abbildung 13 Check Variant

Folgende Eingaben sind möglich:

(1) Frei zu vergebender Name der Prüfvariante. Es sollte aus dem Namen hervorgehen, dass es sich um eine Remote-aufgerufene Variante handelt.
(2) Name der im zentralen Prüfsystem erzeugten RFC-fähigen Prüfvariante
(3) RFC-Destination (des zentralen Prüfsystems)
(4) RFC-System (des zentralen Prüfsystems)

11. Bei der Definition eines Object-Sets im SCI haben sie die Möglichkeit, Code, der auf dem Satellitensystem generiert wurde, auszuschließen. Dies ist aber nur relevant, wenn Sie eine Batch-Analyse starten, bei der auch ein Object-Set angegeben werden kann. Alternativ können Sie direkt im ATC bei der Definition der Run-Series eine Query mitgeben. Sie können so z. B. die relevanten oder die auszuschließenden Pakete angeben.

Hinweis:
Bei der Definition von „Scheduled Runs“ (Laufserien) im ATC können Sie entweder direkt Selektionskriterien eingeben (Pakete, Transport Layer etc.) oder ein vordefiniertes Object-Set aus dem SCI angeben.
Leider verhalten sich diese Selektionen unterschiedlich. Um z. B. generierte Tabellenpflegedialoge und deren Coding bei Events analysieren zu können, müssen Sie die direkte Abfrage im ATC wählen. Wenn Sie eine SCI-Prüfvariante nutzen, haben Sie keine Möglichkeit, diese Art der Unterprogramme zu inkludieren.
Dieses ist der aktuelle Software-Stand des ATC zum 19.11.2019. Gegebenenfalls wird diese Design-Lücke in einem der zukünftigen Releases geschlossen oder über eine SAP-Note.

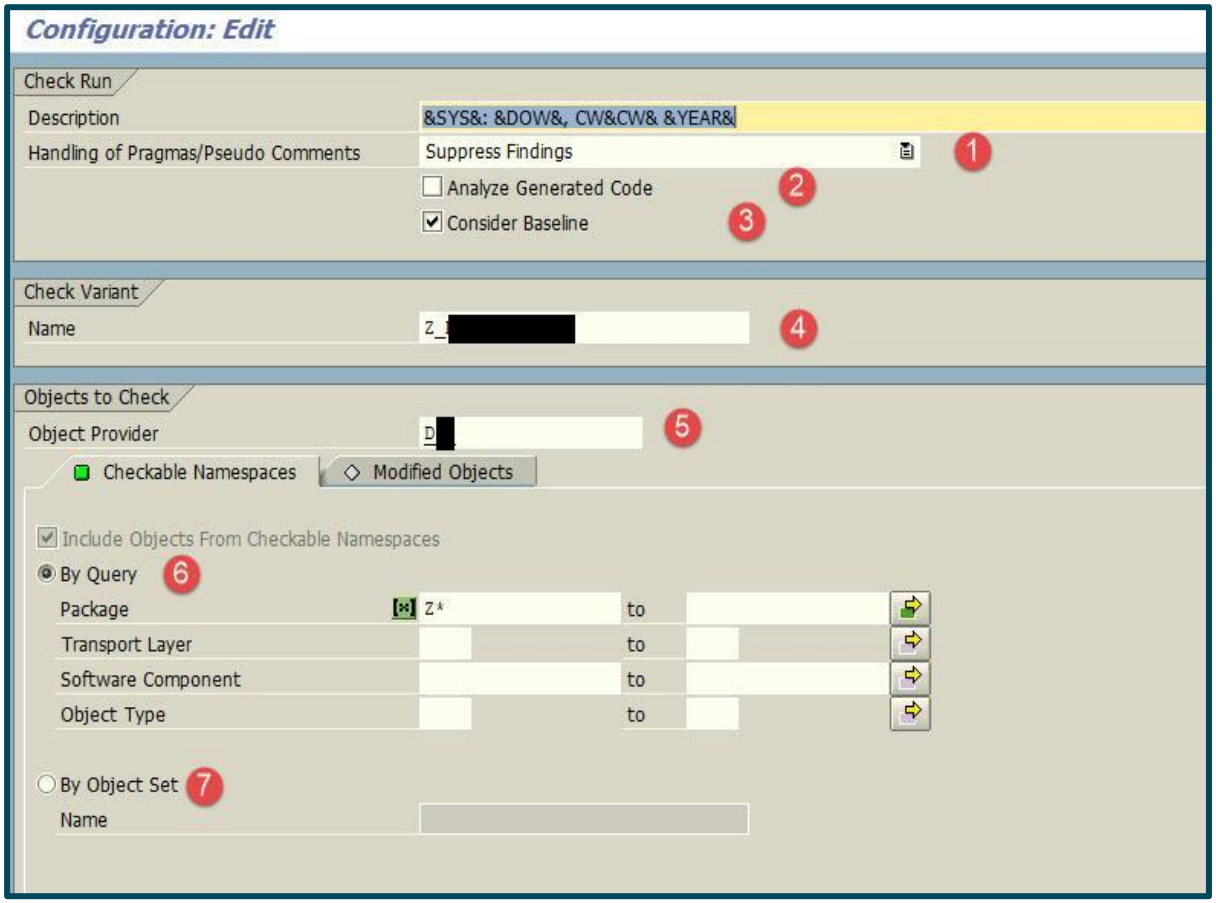


Abbildung 14 Check Run Configuration

Folgende Konfigurationsmöglichkeiten existieren:

(1) Wie sollen Pseudokommentare und Pragmas behandelt werden?
(2) Soll generiertes Coding berücksichtigt werden?
(3) Soll die Baseline berücksichtigt werden?
(4) Name der Prüfvariante auf dem zentralen ATC-System
(5) Object-Provider (Bezeichnung des Satellitensystems, wie in der ATC-Administration auf dem Central-Master-System hinterlegt)
(6) Definition der Pakete und auszuschließenden Pakete für die Analyse
(7) Angabe des auf dem Satellitensystem definierten Object-Sets

3.3.1 Vor- und Nachteile des zentralen Prüfsystems

Vorteile:

- Sie müssen die Prüfvariante nur einmal zentral pflegen.
- Die Prüfvariante wird durch die Remote-Prüfung für alle Satellitensysteme genutzt und ist für alle Satellitensysteme identisch.
- Das Upgrade eines zentralen Prüfsystems, das nicht gleichzeitig noch ERP-System ist, ist mit einem relativ geringen Aufwand und Risiko verbunden. Dadurch können Sie das System einfach aktuell halten und profitieren so immer von den neuesten Prüfungen.

Nachteile:

- SAP stellt derzeit nur eine Teilmenge aller Prüfungen als Remote-fähige Prüfungen bereit.
- Es entstehen zusätzliche Kosten für das zentrale Prüfsystem. Ist die Transportfreigabe an den ATC gekoppelt, muss die Verfügbarkeit des Systems hoch sein, da bei Ausfällen keine Freigaben im Entwicklungssystem möglich sind.

3.3.2 Zusätzliche Hinweise

- Der Hinweis 2364916 enthält die empfohlenen SAP-Hinweise für die Verwendung des ATC zur Durchführung von Remote-Analyse. Allerdings wird der Hinweis aus unserer Erfahrung nicht immer zeitnah aktualisiert. Daher empfehlen wir auf jeden Fall, in der SAP Knowledge Base nach dem Begriff „ATC Remote“ zu suchen und auf die Komponenten „BC-DWB*“ und „BC-ABA-LA*“ einzuschränken. Hier finden sich immer wieder sehr aktuelle Hinweise, die Sie auf jeden Fall implementieren sollten.
- Hinweise 2527903 und Hinweis 2270689 müssen auf jeden Fall in der neusten Version implementiert sein. Hierzu kann es nötig sein, via RZ11 den Profilparameter rdisp/scheduler/prio_high/max_runtime auf 60 Minuten zu setzen, da der Download der Hinweise extrem lange dauert.
- Interne Tests und interne Performance-Tests dürfen nicht über den Remote-Check ausgeführt werden, auch wenn die Option dazu besteht (hierzu Hinweis 2701747), ansonsten treten Laufzeitfehler im zentralen System während des Prüflaufs auf.
- Der Report RS_ABAP_INIT_ANALYSIS muss auf dem zu prüfenden Systemen gestartet worden sein (er erzeugt vier Tabellen in $TMP Paket).

3.4 ATC in ChaRM Solution Manager

Um die Prüfung durch das ATC bei der Freigabe einer Aufgabe oder eines Transports im ChaRM durchzuführen, müssen Sie die Prüfung in der Transaktion SE03 aktivieren.

(Transaktion SE03: Global Customizing (Transport Organizer) -> „Objektprüfungen bei Auftrags- bzw. Aufgabenfreigabe“).

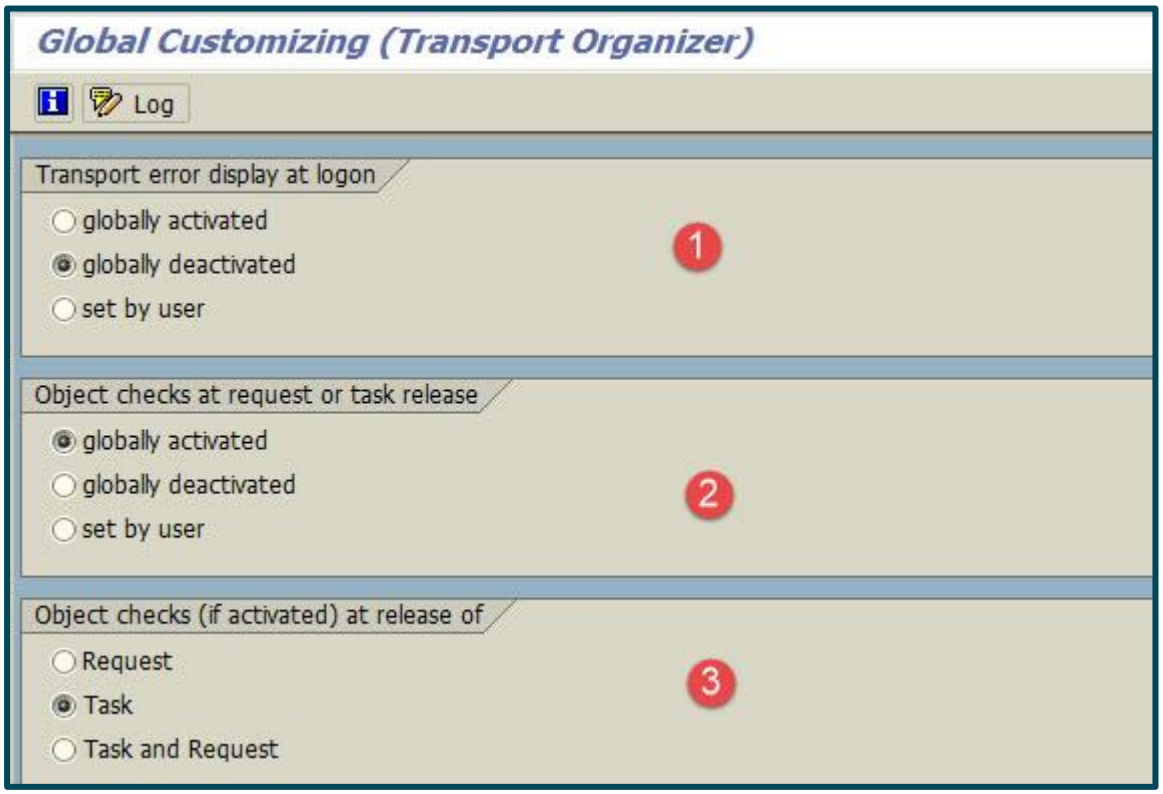


Abbildung 15 Transport Organizer – Global Customizing

In der ATC-Konfiguration können Sie wählen, ob bei Freigabe eines Transports oder einer Aufgabe eine Information ausgegeben werden soll oder die Freigabe abgebrochen wird.

In ChaRM wird der Transport bei Auftreten eines ATC-Fehlers nicht freigegeben, und die aktuelle Aktion wird nicht ausgeführt. Im Gegensatz dazu erzeugt der Code Inspector eine allgemeine Fehlermeldung. Der Transport wird allerdings trotz des Fehlers freigegeben.

Die Wahl zwischen Code Inspector und dem ATC treffen Sie direkt in der Transaktion ATC: „ATC-Administration/Setup/Basiseinstellungen“ im Bereich „Transportwerkzeugintegration“. Um das ATC zu aktivieren, müssen Sie den Code Inspector auf die Auswahl „Code Inspector als Testtreiber deaktivieren“ stellen. Diese Einstellung muss auf dem Satellitensystem vorgenommen werden.

Wichtig: Mit der SAP-Note 2688510 wird die Funktionalität ausgeliefert, dass aus dem ChaRM heraus das globale Transport-Customizing des Satellitensystems überhaupt berücksichtigt wird.

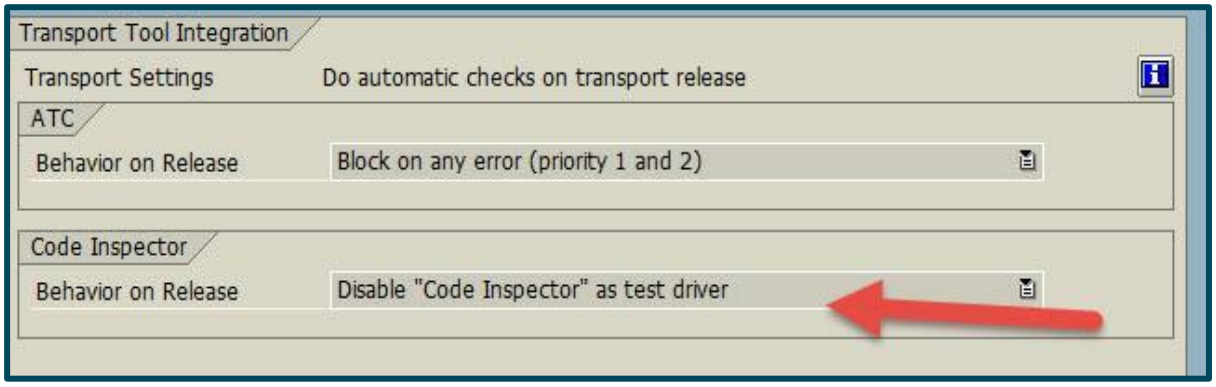


Abbildung 16 Transport Tool Integration

Für das ATC können Sie im Bereich „Verhalten bei Transportfreigabe“ festlegen, ob ein Transport überhaupt geprüft werden soll oder bei Auftreten eines Fehlers der Priorität 1, 1 und 2 oder allen Prioritäten nicht freigegeben werden kann (welche Prioritäten geblockt werden können, hängt vom Release-Stand ab). Für die gleichen Kombinationen an Fehlerprioritäten können Sie alternativ auch nur eine Information bei Freigabe des Transports oder der Aufgabe anzeigen lassen.

Basiseinstellung in einem S/4HANA-1809-System

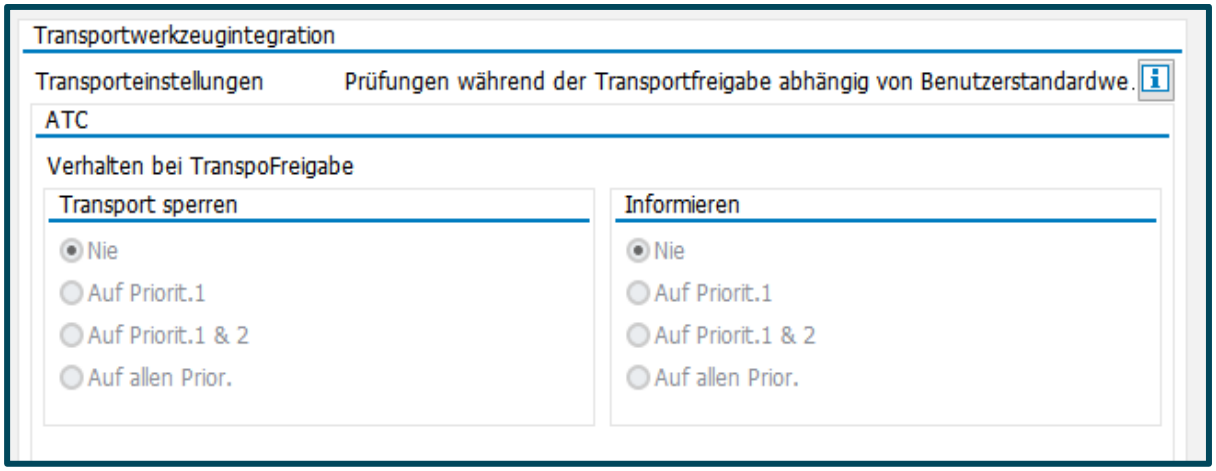


Abbildung 17 Transport Tool Integration (S/4HANA 1809)

Basiseinstellung in einem **Nicht-**S/4HANA-System

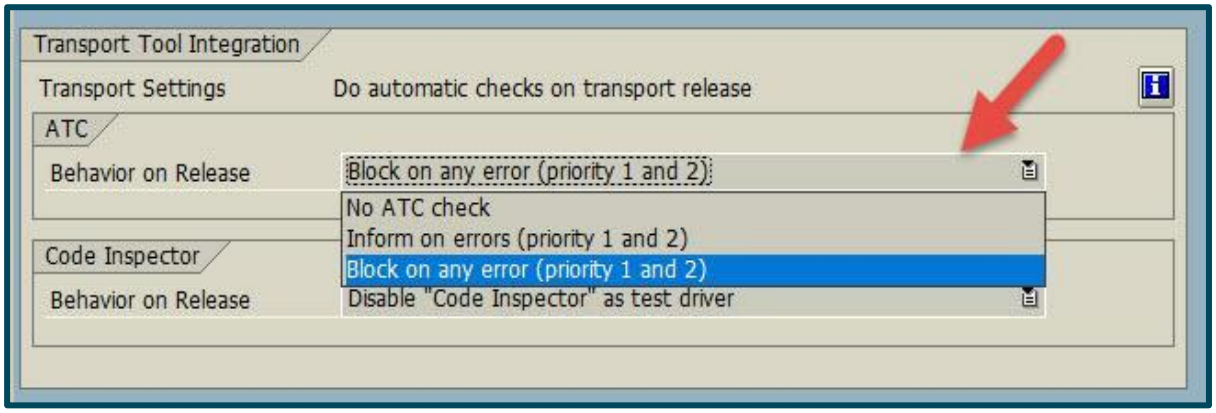


Abbildung 18 Transport Tool Integration (Nicht-S/4HANA-System)

4 Inhaltliches und organisatorisches Set-up

Vor dem konkreten Einsatz des ATC müssen Sie für Ihr Unternehmen zwei grundsätzliche Fragen beantworten:

- Wie verbindlich soll die Durchführung der automatischen Prüfungen und die Behebung der gefundenen Probleme sein?
  - als optionale Hilfestellung für die Entwickler
  - oder ganz verbindlich: Ein Transport von Entwicklungen ins Testsystem ist nur nach Behebung aller gefundenen ATC-Probleme möglich.
  - Oder man wählt einen Weg zwischen diesen beiden Extremen, z. B. Behebung von Problemen erst vor Produktivsetzung, oder zu bestimmten Zeitpunkten innerhalb einer mehrmonatigen Release-Entwicklung.

- Wie umfangreich soll geprüft werden?
  - nur hochkritische Sicherheitsthemen (zum Beispiel)
  - alle von SAP im Standard angebotenen Prüfungen, zuzüglich eventuell weitere selbst oder von anderen SAP-Kunden programmierte
  - oder eine mehr oder weniger große Auswahl von Prüfungen, die im jeweiligen Unternehmen und Kontext am sinnvollsten erscheinen

Durch die Anlage verschiedener Prüfvarianten sind auch Kombinationen möglich: Eine umfangreiche Prüfvariante als optional nutzbare Hilfestellung und eine kleinere für obligatorisch zu behebende Probleme.

Im Folgenden sollen zunächst diese beiden Dimensionen (Verbindlichkeit und Umfang) vertieft werden. Danach folgt ein Abschnitt mit allgemeinen organisatorischen Hinweisen, und schließlich werden einige Einzelthemen dargestellt.

4.1 Verbindlichkeit

Setzt man das Cockpit nur als optionales Hilfsmittel für Entwickler ein, so wird es nach allgemeiner Erfahrung nur von einer Minderheit der Entwickler genutzt werden. Daher sollten Sie zum einen den Einsatz des ATC in den Entwicklungsrichtlinien vorschreiben. Zum anderen ist es sinnvoll, die Behebung der wichtigsten ATC-Probleme (normalerweise Priorität 1 und 2 einer bestimmten Prüfvariante) verpflichtend zu bestimmten Zeitpunkten einzustellen.

Bei SAP-Kunden, bei denen laufend kleinere (und teilweise größere) Entwicklungsarbeiten stattfinden, empfiehlt es sich erfahrungsgemäß, die Prüfung und Problembehebung bereits bei Freigabe der Transportaufgabe als obligatorisch ins Qualitäts-/Testsystem (im Folgenden: Q-System) einzustellen[3].

Aus der Sicht von Entwicklern kann dies allerdings teilweise stark negativ erlebt werden, vor allem in der Einführungsphase. Denn für den Entwickler ergeben sich folgende Einschränkungen:

- Der Entwickler kann nicht mehr so einfach und schnell etwas im Q-System ausprobieren, sondern muss sich sofort mit den durch die automatische Prüfung gefundenen Problemen auseinandersetzen. Diese können in vielen Fällen als „unnötig formal“ erscheinen und den tatsächlichen Arbeitsfortschritt bremsen sowie evtl. auch den eigentlichen Problemlösungs-„Flow“ unterbrechen.
- Es kommt durch die im Rahmen der Transportfreigabe durchgeführten ATC-Prüfungen zu einer Wartezeit, die bei umfangreichen Transportobjektlisten unter Umständen im Minutenbereich liegen kann.
- In Einzelfällen kann ein Befreiungsantrag notwendig werden, wenn das vom ATC gefundene Problem nicht sinnvoll behoben werden kann und kein Pragma oder Pseudokommentar möglich ist. Dadurch kommt es dann zu einer weiteren Wartezeit, bis einer der zur Genehmigung der Befreiungsanträge hinterlegten Kollegen den Antrag anschauen und genehmigen kann.

Demgegenüber stehen aber die Nachteile, wenn man die Behebung der Prüfmeldungen erst zu einem späteren Zeitpunkt (aber natürlich immer vor Produktivsetzung) verlangt:

- Die Software wird durch die ATC-Problembehebungen nach Beginn der Testphase im Q-System nochmals verändert. Streng genommen müssten also die Tests wiederholt werden.
- Fehler, die bei Beachtung der ATC-Meldungen schon im Entwicklungssystem bemerkt und mit geringerem Gesamtaufwand hätten korrigiert werden können, werden erst im Q-System gefunden.
- In der Gesamtkonstellation mit allen Stakeholdern kommt es oft später zu einer gemeinsamen negativen Meinung von Entwickler(n), Anwender(n) und evtl. Management: „Jetzt ist doch alles fertig und getestet, und wir wollen nun die neue Funktion endlich produktiv nutzen. Warum muss jetzt noch einmal zusätzlicher Aufwand in die Behebung von ‚formalen‘ Problemen gesteckt werden, deren Bedeutung uns unklar ist (und damit gering bewertet wird)?“

[3] Je nach Release und Anforderung per Konfiguration oder mit dem BAdI CTS_REQUEST_CHECK Die folgenden Ausführungen gelten für Systemlandschaften mit mindestens drei Systemen.

Wichtig ist in diesem Zusammenhang insbesondere:

- die Auswahl von Prüfungen mit gutem Kosten-Nutzen-Verhältnis für den Entwickler (siehe folgenden Abschnitt und folgendes Kapitel)
- die klare Kommunikation: Es geht nicht um Software-Qualität um ihrer selbst willen, sondern um Produktivitätssteigerung und Kostenreduzierung im gesamten Lebenszyklus der Software[4]

Best-Practice: Wir empfehlen aus der Gesamtsicht die obligatorische Prüfung bei Freigabe der Transportaufgabe, auch wenn diese für die Entwickler etwas unbequemer ist.

Bei Add-on-Herstellern, die umfangreichere Releases und längere Zeitspannen bis zur Finalisierung einer Entwicklung haben, ist die obligatorische Prüfung bei Transportfreigabe etwas weniger wichtig und wird manchmal nicht erzwungen. Dafür kann aber eine Prüfung zum Ende des Entwicklungszyklus dort wesentlich leichter durchgesetzt werden. Teilweise gibt es hierzu auch Regularien von SAP.

4.2 Umfang der Prüfungen

Der Umfang der Prüfungen wird optimalerweise direkt durch das Kosten-Nutzen-Verhältnis bestimmt. Folgende Kriterien sind hier auf Ebene der einzelnen Prüfung wichtig:

- Wie viele echte Fehler werden durch die Prüfung gefunden?
  - Wie viel Aufwand ist es, sie zu beheben?
  - Wie negativ hätten sie sich (wenn unbemerkt) ausgewirkt? Welcher Schaden und wieviel Aufwand wäre dann entstanden?
- Wie viele False-Positives werden von der Prüfung gemeldet?
  - Wie viel Aufwand ist es, die False-Positives zu bearbeiten?
    - Erkennen (seitens des Entwicklers), dass es sich um ein False-Positive handelt
    - gegebenenfalls Erstellung eines Befreiungsantrags für den ATC-Befund
    - ggf. Überzeugungsarbeit, bis der Genehmiger dies ebenso sieht
- Wie verständlich und einleuchtend ist der ATC-Befund dieser Prüfung für den Entwickler? Aufgrund der Historie haben die ATC-Prüfungen leider keinen durchgängigen Qualitätsstandard, und im Text fehlen teilweise wichtige Informationen.

[4] Siehe hierzu z. B. die Artikel von Martin Fowler: https://martinfowler.com/articles/is-quality-worth-cost.html sowie https://martinfowler.com/bliki/TradableQualityHypothesis.html (beide abgerufen am 05.12.2019)

- Wie performant ist die Prüfung? Dies gilt insbesondere für die nicht für den ATC optimierten Prüfungen der erweiterten Programmprüfung.
- Gegebenenfalls: welche Fehler müssen behoben werden, z. B. wegen der Migration auf die HANA-Datenbank?

Zusätzlich gibt es übergreifende Aspekte. Die ATC-Einführung in einem Unternehmen kann zunächst oft Widerstand und Schmerzen verursachen. Das sollte aber so dosiert werden, dass es nicht zu einer überwiegenden Ablehnung bei den Entwicklern kommt. Hierzu können folgende Vorgehensweisen sinnvoll sein:

- Den Prüfumfang schrittweise ausweiten.
- Zunächst nur eine Pilotgruppe von Entwicklern teilnehmen lassen.
- ggf. Anpassung bereits ausgerollter Prüfungen (siehe Kapitel 7)
- Meldungen mit Priorität 3, die (im Standard) den Transport nicht verhindern, komplett ausblenden, um den Blick auf das Wesentliche zu lenken. Siehe hierzu auch den folgenden Abschnitt und den Abschnitt „Anpassung/Filterung der Prüfergebnisse per Code“.

In diesem Zusammenhang möchten wir darauf hinweisen, dass Modifikationen und Enhancements erst ab Release 7.52 durch den ATC geprüft werden.

4.3 Allgemeine organisatorische Empfehlungen

Für eine erfolgreiche Einführung von ATC-Prüfungen im Unternehmen ist es wichtig, alle am Entwicklungsprozess beteiligten Personengruppen in den Prozess einzubinden. Zu den beteiligten Gruppen gehören:

- Entwicklung/Programmierung
- Basis/Betrieb - vor allem, wenn ein zentrales ATC-System installiert werden soll, aber auch für das Aufsetzen der notwendigen Berechtigungen und dem Einspielen diverser OSS-Notes
- Qualitäts-Manager/Qualitätsverantwortliche, falls diese Rolle vorhanden ist
- Prozess-Teams/fachliche IT-Experten - informatorisch

Der Zeitrahmen für die Einführung sollte nicht zu knapp bemessen werden, damit ausreichend Zeit für

- Infoveranstaltungen
- Justierung der Einstellungen im System
- Einspielen von OSS-Hinweisen

und andere Aktivitäten zur Verfügung steht.

Wir empfehlen, vor allem die Programmierer/Entwickler frühzeitig über die Einführung von ATC-Prüfungen zu informieren. Dies kann z. B. durch intern verteilte Blog-Artikel

passieren sowie durch Infoveranstaltungen. Gibt es im Unternehmen bereits online verfügbare Entwicklungsrichtlinien, können Sie diese Plattform als Dokumentations- und Informationsmedium nutzen.

Allgemein ist in der Regel der unten beschriebe Baseline-Mechanismus sinnvoll, und Sie sollten einen robusten Genehmigungsprozess für Ausnahmen etablieren, damit es zu keinen unnötigen Verzögerungen bei der Freigabe von Transporten kommt. Siehe dazu die Ausführungen unten in Abschnitt 4.5.

Wurden bisher die Code-Inspector (SCI)-Prüfergebnisse bei der Transportfreigabe angezeigt, bietet es sich an, frühzeitig und u. U. zusätzlich die ATC-Prüfungen basierend auf der gleichen Prüfvariante ausführen und anzeigen zu lassen. Dadurch können sich die Entwickler bereits zu einem frühen Zeitpunkt mit der geänderten Darstellung der ansonsten gleichen Ergebnisse vertraut machen.

Wenn Sie schrittweise in die ATC-Verwendung einsteigen möchten, empfehlen wir folgendes Vorgehen:

- Phase 1: Gewöhnungsphase. ATC-Meldungen werden bei der Transportfreigabe angezeigt, verhindern aber den Transport nicht.
- Phase 2: Durch entsprechendes Einstellen der Meldungsprioritäten oder der Prüfvariante werden zunächst nur einige besonders kritische Prüfungen aktiviert.
- Phase 3: Gegebenenfalls werden sukzessive weitere Prüfungen hinzugenommen bzw. auf Priorität 1 oder 2 hochgestuft, sodass sie den Transport verhindern.

Siehe zu diesen Themen auch den ausführlichen Erfahrungsbericht:
https://blogs.sap.com/2018/10/22/setting-up-a-central-atc-system-the-long-long-winding-road-edition/ (abgerufen am 05.12.2019)

4.4 Prioritäten

Die SAP-Standard-Prioritäten der Prüfungen können von der Einstiegsseite der Transaktion SCI Code Inspector aus über das Menü kundenspezifisch angepasst werden.

Hiermit kann der Umfang der transportverhindernden Prüfungen angepasst werden, wenn es eine obligatorische Meldungsbehebung bei Transportfreigabe gibt, die sich im Standard auf Prio 1- und -2-Meldungen beschränkt.

Hinweis: Allerdings ist es (wie allgemein bei der Verwendung von Software) ratsam, nicht unnötig stark vom Standard abzuweichen. Wenn viele Prioritäten angepasst werden sollen, sollte das Vorgehen hinterfragt und diskutiert werden.

4.5 Befreiungen (Ausnahmen), Prozessintegration

Wenn ein ATC-Befund nach Meinung des Entwicklers nicht sinnvoll behebbar ist und auch kein Pragma oder Pseudokommentar zur Verfügung steht (z. B. Namenskonventionsprüfungen bei generiertem Coding), kann der Entwickler aus der Anzeige des ATC-Befundes heraus einen Befreiungsantrag stellen.

Details zum Vorgehen (z. B. Für welche Objekte ist es möglich, Befreiungen zu beantragen?) bietet folgender Blog:

https://blogs.sap.com/2017/02/27/remote-code-analysis-in-atc-working-with-exemptions/ (abgerufen am 22.11.2019)

Der aktuelle Standard sieht vor, dass der Entwickler einen Genehmiger aus einer Liste von eingerichteten Genehmigern auswählt, und dann vom System eine automatische Mail an diesen versendet wird. Diese kann vom System dank Hinweis 2619387 mittlerweile nicht nur täglich, sondern auch unmittelbar oder wöchentlich verschickt werden. Normalerweise wird man die Option „sofort versenden“ wählen, da unnötige Wartezeit vermieden werden soll.

Hinderlich ist oft, dass im Standard der Genehmiger explizit ausgesucht werden muss. Der Beantragende weiß häufig nicht, welcher Genehmiger gerade verfügbar ist. Dies lässt sich am einfachsten über einen technischen User, hinter dem eine Postfach-Mailadresse oder Verteilerliste steht, lösen.

Ein anderes Problem kann darin bestehen, dass für die Entwicklungs- und Qualitätssicherungssysteme in manchen Firmen kein Anschluss an das Mail-System eingerichtet ist.

Hinweis: Man kann den Funktionsbaustein SATC_CI_EXEMPTION_REQUEST mit einer impliziten Erweiterung anpassen, um bei Befreiungsanträgen den Antragsteller darauf hinzuweisen, dass der Genehmiger bitte direkt (E-Mail, Telefon etc.) kontaktiert werden soll. Damit entschärft sich das Problem mit der Auswahl eines Genehmigers (der evtl. im Urlaub ist). Eine andere - generelle - Option ist, dass E-Mails aus Entwicklungs- und Qualitätssicherungssystemen nur an Adressen in der firmeneigenen Domäne verschickt werden.

4.6 Eigene Namensräume für ATC-Prüfungen registrieren

Das ATC kann neben den Standardnamensräumen „Y*“ und „Z*“ auch Objekte in eigenen Namensräumen prüfen. Die eigenen Namensräume müssen dazu registriert werden. Diese Namensraumreservierung wird ausschließlich vom ATC verwendet und hat keine Auswirkungen auf die Namensräume, die bei der SAP lizenziert wurden.

Vorgehensweise:

- Rufen Sie im Entwicklungssystem über die Transaktion SE38/SA38 den Report SATC_AC_INIT_NAMESPACE_REG auf
- Das System zeigt alle Namensräume an, denen die Rolle „Producer“ zugeordnet ist und die nicht SAP gehören.
- Sie können die entsprechenden Namensräume markieren und über den Button „Registrieren“ oder „Deregistrieren“ hinzufügen bzw. löschen.

4.7 Baseline

Systemvoraussetzung: SAP NetWeaver AS ABAP 7.51 Innovation Package

Für die oben angesprochene Kosten-Nutzen-Betrachtung und die psychologische Komponente einer ATC-Einführung ist der Baseline-Mechanismus sehr bedeutend.

Wenn es bisher keine automatischen Qualitätsprüfungen der Software gegeben hat, gibt es ohne definierte Baseline bei älteren, „historisch gewachsenen“ Kundensystemen normalerweise eine unüberschaubare Menge an ATC-Befunden.

Ein typischer Fall:

Ein Entwickler soll eine kleine Änderung an einem älteren Programm vornehmen. Wenn das Programm schon lange Zeit ohne auffällige Fehler produktiv genutzt wurde und er es evtl. auch nicht selbst geschrieben hat, wird er wenig Motivation aufbringen, eine große Anzahl „alter Probleme“ (ATC-Befunde) in dem Programm zu korrigieren. Auch der jeweilige Auftraggeber hat oft kein Verständnis für den Zusatzaufwand bei der Entwicklung und beim Testen.

Mit einer Baseline können Sie eine Menge von bekannten Befunden im Legacy-Code für weitere ATC-Läufe so markieren, dass sie nicht mehr in den Resultaten auftauchen.

Diese Optionen zur Markierung stehen dabei zur Verfügung:

- Befund unterdrücken
- Befund befreien
- niedrige Priorität zuordnen

Die Definition der Baseline können Sie auch nachträglich noch anpassen und verfeinern.

Mit der Baseline haben Sie also die Möglichkeit, einen Nullpunkt zu definieren, ab dem Ihre aktuelle Prüfvariante gültig sein soll. Optional können Sie auch bestimmte, besonders wichtige Prüfungen von der Baseline-Definition ausnehmen, so dass die entsprechenden Befunde auch in Alt-Coding behoben werden müssen. Eine weitere Möglichkeit ist, in größerem Umfang bestimmte False-Positive-Befunde zu unterdrücken. Mehr Informationen dazu finden Sie in folgendem Blog:

https://blogs.sap.com/2016/12/13/remote-code-analysis-in-atc-working-with-baseline-to-suppress-findings-in-old-legacy-code/ (abgerufen am 10.05.2019)

4.8 Anpassung/Filterung der Prüfergebnisse per Code

Aus verschiedenen Gründen kann es sinnvoll sein, die vom ATC im Standard gefundenen Befunde anschließend mit einer impliziten Erweiterung[5] anzupassen bzw. zu filtern:

- Änderung der Priorität, wenn dies mit dem Standardmechanismus nicht geht (z. B. bei der generischen Suchfunktion, die für kritische Sprachelemente wie Makros verwendet wird)
- Falls dies – je nach NetWeaver-Release – noch nicht im Standard gewährleistet wird: Löschung von Befunden für eingebundene SAP-Includes
- Löschung von Befunden für mehrfach-verwendete „Legacy“-Includes gemäß einer „Whitelist“ in einer Z-Tabelle, wenn man diese aus Qualitätssicht akzeptiert, aber nicht erneut genehmigen möchte. (Ansonsten werden diese Befunde - trotz Baseline-Mechanismus - bei jeder Verwendung in einem neuen Rahmenprogramm erneut bemängelt.)
- Löschung von Namenskonventions-Befunden für generiertes Coding oder Framework-Coding, wo die Parameternamen nicht frei gewählt werden können

Hierfür kann (im Fall NetWeaver 7.50, dezentraler ATC) eine implizite Erweiterung am Ende der Methode IF_SATC_CI_ADAPTER~ANALYZE_OBJECTS() der lokalen Klasse ADAPTER_2_CODE_INSPECTOR in der globalen Klasse CL_SATC_CI_ADAPTER verwendet werden.

4.9 Missbrauch von Pragmas/Pseudokommentaren

Es besteht immer die Gefahr, dass die Möglichkeit von Pragmas bzw. Pseudokommentaren missbraucht wird, dass also – ohne nachzudenken – ein ATC-Befund immer durch Eintragen des Pragmas „gelöst“ wird. Mit solch einem Vorgehen wird das eigentliche Ziel der ATC-Einführung wissentlich oder unwissentlich unterlaufen.

[5] Anmerkung für Nicht-Entwickler: Implizite Erweiterungen sind eine von SAP angebotene Erweiterungsmöglichkeit, bei der ohne Modifikationsschlüssel zusätzliche Programmzeilen am Anfang oder Ende einer Modularisierungseinheit eingefügt werden können.

Hierzu gibt es verschiedene Lösungsansätze:

- Ausschalten der Möglichkeit von Pragmas/Pseudokommentaren für bestimmte Prüfungen:
  Für einige Standardprüfungen kann durch Implementierung einer schlanken, abgeleiteten Prüfklasse die Möglichkeit von Pseudokommentaren und Pragmas ausgeschaltet werden.
  Siehe hierzu diesen Blog: https://blogs.sap.com/2019/01/08/atc-how-to-disable-usage-of-pragmaspseudocomments-for-standard-tests/
  (abgerufen am 05.12.2019)

- Alternativ können Sie eine eigene Prüfung implementieren, in der „nicht erlaubte“ Pragmas und Pseudokommentare per Z-Tabelle konfiguriert werden können, für die bei Benutzung ein entsprechender ATC-Befund „Pragma XYZ nicht erlaubt“ erzeugt wird. Hier bleibt dann immer noch der sinnvolle, kontrollierte Weg über Ausnahmeanträge.
- Sie können die Verwendung von Pragmas manuell oder per Programm überwachen und dann den jeweiligen Entwickler ansprechen.
- Denkbar ist auch die Implementierung einer eigenen ATC-Prüfung, die z. B. die Anzahl Pragmas je 1000 Code-Zeilen zählt und bei einem Grenzwert einen Prio 1-ATC-Befund auslöst.

4.10 Rollen/Berechtigungen

Folgende Tabelle stellt die im SAP-Standard gelieferten Rollen dar.

Function	Needed in Master System	Needed in Satellite System
Admin	SAP_SATC_ADMIN S_RFCACL (with ref. to Satellite System)	S_RFCACL (with ref. to Master System) S_ICF
Quality Expert	SAP_SATC_QE S_RFCACL (with ref. to Satellite System)	S_RFCACL (with ref. to Master System) S_ICF
Developer	SAP_SATC_XMPT_APPLICANT S_RFCACL (with ref. to Satellite System)	S_RFCACL (with ref. to Master System) S_ICF
Notes	We use trusted RFC connections Authorization Object S_RFCACL is therefore always needed	S_ICF with ICF_Field = Dest ICF_Value = ATCXMPT

Abbildung 19 Notwendige Berechtigungen

(Quelle: SAP-Community-Blog unter https://blogs.sap.com/2018/09/10/setting-up-a-central-atc-system-part-4-the-final-stretch/
(abgerufen am 05.12.2019)

5 Definition einer eigenen Standard-Prüfvariante

Die mit dem ATC bereitgestellten Prüfungen bieten insbesondere für Administratoren des ATC und Qualitäts-Manager einen umfangreichen Werkzeugkasten, aus dem Sie die für Ihre Szenarien optimal geeigneten Prüfumfänge zusammenstellen können.

Die Prüfvarianten können als lokale oder globale (für alle Benutzer sichtbare) Varianten hinterlegt werden. Die folgende Darstellung liefert einen Überblick über die wesentlichen Aspekte des ATC.

Abbildung 20 Aspekte des ATC

Definieren einer eigenen Prüfvariante

Unter dem Menüpfad „ATC-Administration/Setup/Basiseinstellungen“ können Sie im Bereich „Code Inspector“ eine globale Prüfvariante angeben. Diese globale Prüfvariante wird sowohl bei der Transportfreigabe als auch bei der Prüfung in den Entwicklungswerkzeugen als Standard-Prüfvariante verwendet, sofern nicht explizit eine andere Variante ausgewählt wird.

Hinweis: Die globale Prüfvariante kann im Satellitensystem so konfiguriert werden, dass sie auf eine im zentralen System definierte Variante verweist.

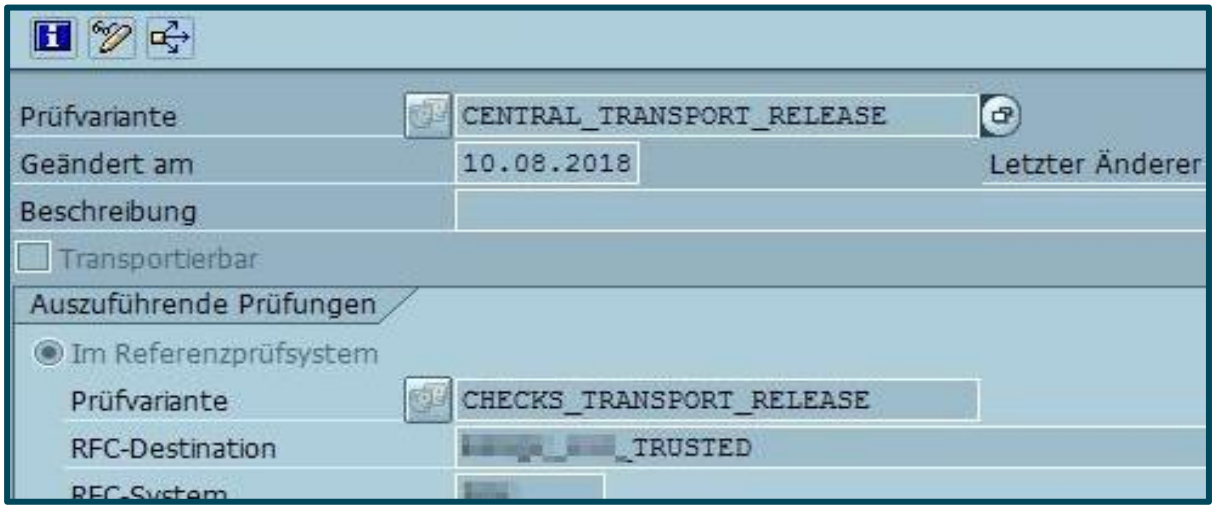


Abbildung 21 Beispiel für die Konfiguration einer zentralen Prüfvariante im Non-Remote-System

Die Prüfvarianten werden im Code Inspector (Transaktion SCI) definiert. Alternativ können Sie die Code-Inspector-Varianten auch direkt unter dem Menüpfad „ATC-Administration/Qualitäts-Governance/Prüfvarianten verwalten“ bearbeiten.

Eine Standard-Prüfvariante muss als

- „global“ (für alle Benutzer sichtbar) und
- „transportierbar“ (mit SAP-Transportaufträgen)

definiert sein und sollte im Rahmen des Remote-Prüfszenarios auf allen Entwicklungssystemen zur Verfügung stehen, damit Ihre Entwickler die gleichen Prüfungen für im Satellitensystem ausgeführte Tests (lokal) verwenden, die auch bei den zentralen Qualitätsprüfungen verwendet werden. Hierbei müssen Sie beachten, dass die ausgewählten Prüfungen Remote-fähig sind, was Sie am „Blitz-Symbol“ der Prüfungen erkennen.

Eine lokal definierte Kopie der globalen Prüfvariante kann hilfreich sein, wenn z. B. bei einem technischen Ausfall des zentralen Systems die Prüfungen trotzdem durchlaufen werden sollen.

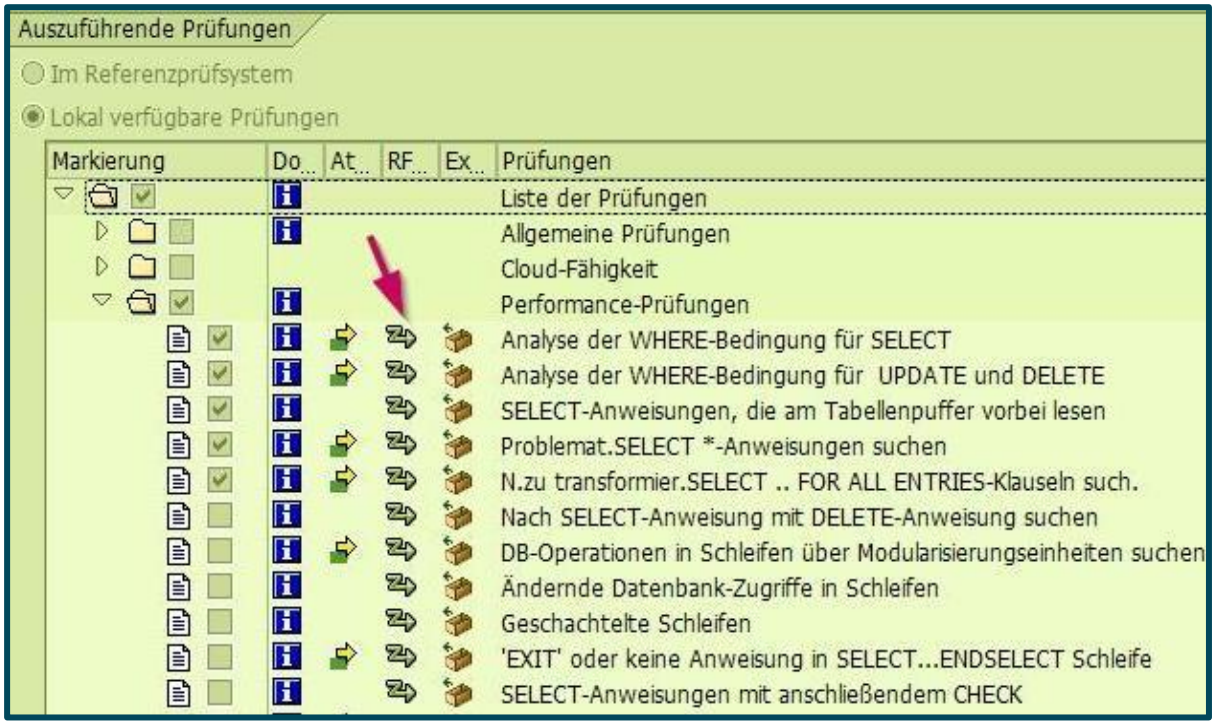


Abbildung 22 Remote-fähige Prüfvarianten

Die Definition des Prüfumfangs muss sich nach dem Zeitpunkt der Durchführung der Prüfung richten. Sowohl die lokale Online-Prüfung der Entwickler als auch die Prüfung bei der Freigabe eines Transports sollte möglichst schnell durchgeführt werden können und nur in Ausnahmefällen länger als einige Minuten dauern (z. B. bei Objekten mit vielen Includes). Deshalb sollten Sie in der Standard-Prüfvariante nur Prüfungen hoher Priorität aufnehmen, welche beim Auftreten eines Fehlers einen Transport ins nachgelagerte System verhindern sollen.

Die Einstellung, die Freigabe eines Transportes aufgrund von ATC-Fehlern der Priorität 1 und 2 abzubrechen, können Sie unter dem Menüpfad „ATC-Administration/Setup/Basiseinstellungen“ im Bereich „Transportwerkzeugintegration“ vornehmen.

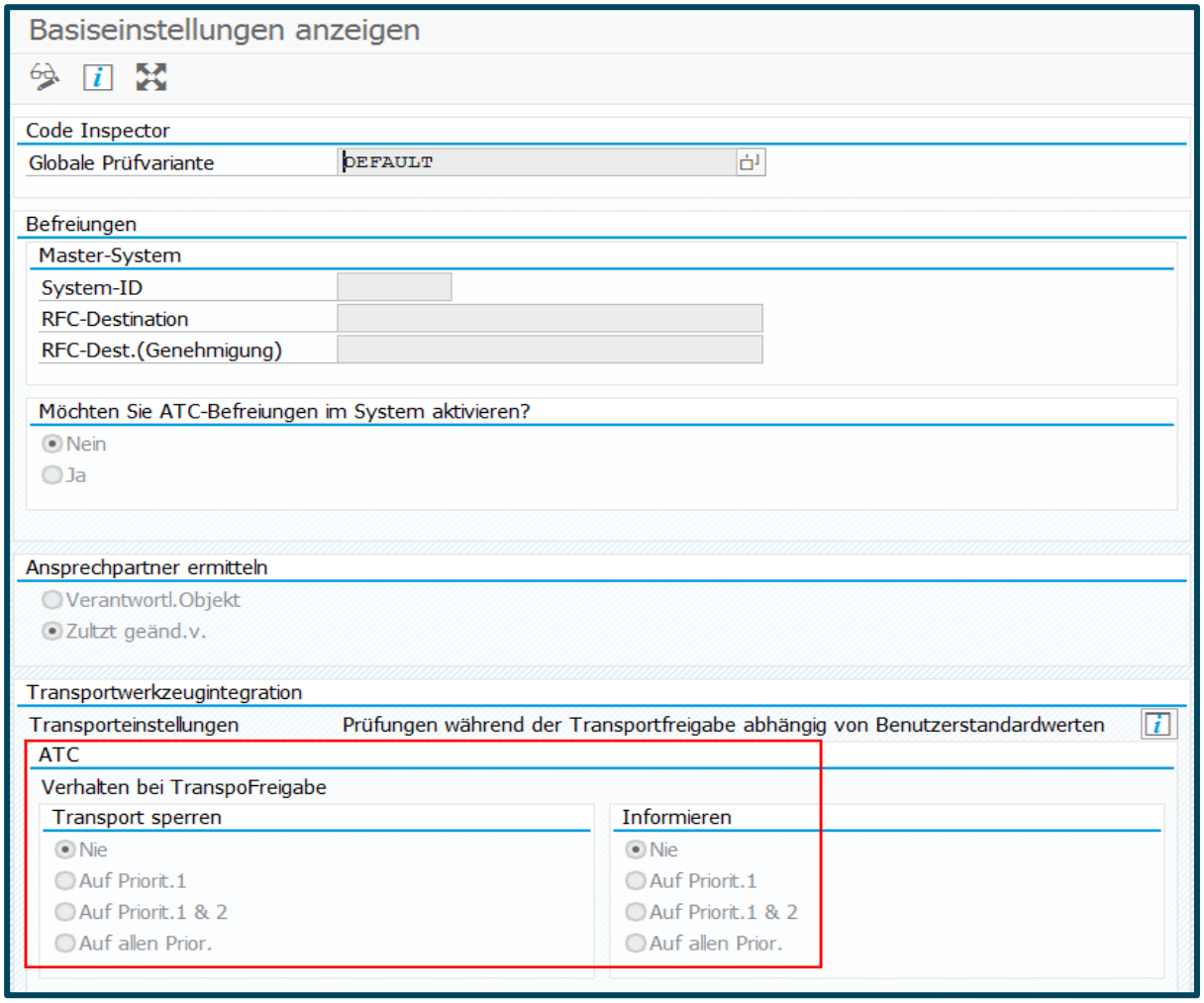


Abbildung 23 Verhalten Transportfreigabe ab S/4HANA 1809

Prüfungen der Priorität „Warnung“ oder „Information“ sollten Sie in Prüfvarianten aktivieren, die vom Entwickler explizit gewählt oder als Hintergrundjob eingeplant werden.

Eine Übersicht der Fehlerpriorität bekommen Sie in der Transaktion SCI oder dem ATC-Menüpfad „ATC-Administration/Qualitäts-Governance/Prüfvarianten verwalten“ unter dem Menü „Code Inspector/Verwaltung von/Meldungsprioritäten“. Hier können Sie bei Bedarf die Meldungspriorität anpassen.

Hinweis: Meldungsprioritäten können nur global definiert werden und gelten deshalb für alle Prüfvarianten!

Für Hintergrundjobs können Sie die zu untersuchenden Objekte in der Transaktion SCI oder unter dem Menüpfad „ATC-Administration/Qualitäts-Governance/Prüfvarianten verwalten“ bestimmen. Dies kann eine Objektmenge, ein bestimmter Transportauftrag oder auch ein bestimmtes Entwicklungsobjekt sein.

Bei der Freigabe eines Transports von Kopien wird bis zum aktuellen Release 7.52 SP 0002 grundsätzlich keine ATC-Prüfung durchgeführt[6].

Bei der Verwendung von Prüfvarianten kann exemplarisch zwischen folgenden Use-Cases unterschieden werden:

- **Standardvariante (Transportfreigabe + Entwickler):** Wie oben ausgeführt, wird diese Variante insbesondere bei der Transportfreigabe und bei der Objektprüfung (sofern keine spezifische Variante gewählt wird) verwendet.
- **Variante für zentralen Prüflauf:** Sehr zeitintensive und geringer priorisierte Prüfungen sollten im Rahmen eines täglichen Hintergrundjobs durchgeführt werden. Der für das Objekt verantwortliche Entwickler kann sich mit Hilfe des ATC Result Browsers über die zu bearbeitenden Punkte informieren.
- **Variante für Baseline:** Diese Variante kommt in der Regel einmalig zum Einsatz. Sie muss die Prüfungen enthalten, die die Transportfreigaben für historisches Coding („Altlasten“) nicht blockieren sollen.

5.1 Baseline – konkretes Vorgehen

Grundlegende Informationen zur Baseline wurden bereits im Kapitel 4 „Inhaltliches und organisatorisches Set-up“ beschrieben, weshalb wir uns hier auf konkrete Hinweise in Bezug auf die zu verwendenden Prüfvarianten beschränken.

In der Regel wird die Baseline für alle kundeneigenen Programme einmal erstellt und die zu verwendende Prüfvariante orientiert sich daran, was zukünftig bei der Transportfreigabe geprüft werden soll. Beim Erstellen der Baseline können Sie festlegen, ob die Befunde komplett unterdrückt, als „ausgenommen“ oder mit Priorität 3 angezeigt werden sollen.

Praxisbeispiel:

Im Unternehmen wird festgelegt, dass direkte Updates auf SAP-Tabellen eine der Prio-1-Prüfungen bei der Transportfreigabe sein sollen. Diese Updates sollen wenn möglich entfernt werden, weshalb die Meldung eine Transportfreigabe verhindern soll, selbst wenn das betroffene Coding durch die aktuelle Änderung nicht angefasst wurde. Der Programmierer kann dann gegebenenfalls eine Befreiung beantragen und eine Begründung liefern, weshalb es aktuell keine Alternative zum harten Update gibt.

6 https://answers.sap.com/questions/309739/atc-transport-checks-not-working.html (abgerufen am 14.04.2019)

Da die Befreiungen in einer Tabelle gespeichert werden, steht die Information später noch zur Verfügung und kann bei der Dokumentation und der Festlegung weiterer Maßnahmen helfen.

Für die Prüfvarianten bedeutet dies, dass die Prüfung auf direkte SAP-Tabellen-Updates in der Prüfvariante für die Transportfreigaben enthalten sein muss, in der Prüfvariante für die Erstellung der Baseline jedoch nicht enthalten sein darf.

Hinweis: Damit bei der Prüfung auf harte Tabellen-Updates nur SAP-Tabellen berücksichtigt werden, können Sie in den Einstellungen die Auswahl entsprechend einschränken, z. B. durch das Ausschließen aller Tabellen im Kundennamensraum und der TVARVC:

Abbildung 24 Ausschluss von Tabellen

5.2 Prüfungen des Code Vulnerability Analyser (CVA)

Der Code Vulnerability Analyser ist ein Add-on zum ATC, um statische Sicherheitsprüfungen auszuführen. Im SAP-Standard werden bereits statische Security-Prüfungen angeboten. Der Vorteil des CVA ist der umfangreichere Prüfrahmen der Code-Stelle. Als Beispiel sei die SQL-Injection-Prüfung genannt: Der CVA erkennt mit einer Datenflussanalyse eine SQL-Injection dann, wenn der dynamische Wert an einer anderen Stelle des Programms durch Benutzereingabe befüllt wird. Dies führt zu einem genaueren Resultat.

Prüfungen umfassen Injection Erkennung, Directory Traversals und Backdoor Detection. Die Prüfungen lassen sich nur als Gesamtes in der Prüfvariante aktivieren. Wir empfehlen, die genauen Einstellungen über die Meldungsprioritäten festzulegen.

Dies ist in sehr detailliertem Ausmaß (je Meldung) konfigurierbar, oder man deaktiviert einzelne Meldungen, indem diese nicht angezeigt werden sollen.

Eine detaillierte Übersicht zu den angebotenen Prüfungen findet sich in folgendem SAP-Blog: https://blogs.sap.com/2017/01/19/code-vulnerability-analyzer-checks/ (abgerufen am 22.11.2019)

Sie müssen den CVA-On-Premise separat lizenzieren und mit dem Report RSLIN_SEC_LICENSE_SETUP aktivieren. Genauere Informationen finden Sie im Hinweis 1855773. Für ABAP in der Cloud ist die Nutzung des CVA lizenzkostenfrei.

5.3 abapOpenChecks und Code Pal for ABAP

Neben den im SAP-Standard verfügbaren Prüfungen können auch eigene Prüfklassen entwickelt werden. Die abapOpenChecks (siehe Anhang B) und der Code Pal for ABAP (siehe Anhang C) sind eine Sammlung von Prüfungen, die durch eine Open-Source-Community entwickelt und zur Verfügung gestellt werden. Sie können in das eigene SAP-System geladen und kostenlos genutzt werden.

Hinweis: Die abapOpenChecks und der Code Pal for ABAP sind Community-Projekte und somit kein Teil des SAP-Supports. Bitte öffnen Sie bei SAP keine Meldungen zu diesen Themen.

5.4 Übersicht empfohlener Prüfvarianten

Üblicherweise wird im ATC eine Prüfvariante als Default definiert, die dann standardmäßig sowohl bei manuell angestoßenen Prüfungen als auch bei der Task-/Transportfreigabe verwendet wird, in der Regel durch den Baseline-Mechanismus nur für neue oder geänderte Codestücke. Es kann aber Gründe geben - z. B. die Performance bestimmter Prüfungen -, diese nicht im Zuge der Transportfreigabe auszuführen. In diesem Fall empfehlen wir die Definition einer weiteren Prüfvariante, die diese zusätzlichen Prüfungen enthält und den Entwicklern für Ad-hoc-Prüfungen zur Verfügung gestellt werden kann. Diese Variante muss dann in der SE80 (bzw. ADT) manuell im Kontextmenü über „Prüfen – ABAP Test Cockpit (ATC) mit…“ ausgewählt werden (anschließend im Reiter Optionen die Code-Inspector-Prüfvariante wählen).

Die folgende Tabelle soll als Anregung dienen, welche Prüfungen aus unserer Sicht als Default-Prüfvariante Sinn machen. Welche davon einsetzbar sind, ist unter anderem vom Release-Stand abhängig.

Anmerkung zu den Performance-Prüfungen:

Beim Thema Performance wird manchmal außer Acht gelassen, dass es in der Regel nur extrem wenige „Hotspots“ gibt. Gemeint sind die Code-Stellen, die die Gesamt-Performance wirklich in relevantem Maße verschlechtern, also einige Prozent der Gesamtlaufzeit von Performance-kritischen Programmen ausmachen.

Insbesondere für Alt-Coding sollten Sie diese nicht flächendeckend „verbessern“, sondern mit den Werkzeugen SQL-Monitor (Transaktion SQLM) und Performance-Trace (ST05) zunächst die im tatsächlichen Betrieb kritischen SQL-Zugriffe ermitteln.

Auch für Neu-Coding sollten Sie beachten, dass es einen Zielkonflikt zwischen Performance und Einfachheit/Wartbarkeit geben kann. Wenn also das Coding durch eine von der ATC-Performance-Prüfung geforderte „Verbesserung“ komplexer (schwerer lesbar, schlechter wartbar) wird, sollte man ein Pragma/Pseudokommentar nutzen, solange der jeweilige SQL-Befehl noch nicht als Hotspot identifiziert ist.[7]

Empfohlene Prüfungen für Default-Variante	Gewichtung	Anmerkung
Allgemeine Prüfungen		
CDS-Modellierungsprüfung (SADL/BOPF)	+	
Annotationsprüfung für Datendefinitionen	+	
Referenzierte Objekte in CDS	+	
Prüfung von Datenbankprozedur-Proxies	+	
Performance-Prüfungen		
Analyse der WHERE-Bedingung für SELECT	+++	
Analyse der WHERE-Bedingung für UPDATE und DELETE	+++	
Nach zu transformierenden SELECT . FOR ALL ENTRIES-Klauseln suchen	+++	

[7] Siehe hierzu auch http://wiki.c2.com/?PrematureOptimization und http://wiki.c2.com/?ProfileBeforeOptimizing (abgerufen am 05.12.2019)

Empfohlene Prüfungen für Default-Variante	Gewichtung	Anmerkung
Nach SELECT-Anweisung mit DELETE-Anweisung suchen	+++	
Problematische SELECT*-Anweisungen suchen	+++	
DB-Operationen in Schleifen über Modularisierungseinheiten suchen	+++	
EXIT oder keine Anweisung in SELECT...ENDSELECT-Schleife	+++	
Ändernde Datenbankzugriffe in Schleifen	+++	
SELECT-Anweisungen mit anschließendem CHECK	+++	
Inperformante Operationen auf internen Tabellen	++	
SORT-Anweisung in Schleife	+++	
Prüfung der Tabelleneigenschaften	+++	
Sicherheitsprüfungen		**u. U. von Baseline ausschließen, um Altlasten zu entdecken**
Sicherheitsprüfungen für ABAP (CVA)*	++	Wenn CVA lizenziert wurde, empfiehlt es sich, die Prüfungen einzuschalten.
Sicherheitsprüfungen für BSP (CVA)*	++	
Kritische Anweisungen	+++	
Dynamic and Client-Specific Accesses with INSERT, UPDATE, MODIFY, DELETE → Changing Access (...) to Table ...	+++	

Empfohlene Prüfungen für Default-Variante	Gewichtung	Anmerkung
Verwendung der ADBC-Schnittstelle	+++	
Syntaxprüfung/Generierung		
Syntaxprüfung für DCL-Quellen	++	
Statusprüfung für DCL-Quellen	++	
Erweiterte Programmprüfung (SLIN)	+++	
Syntax erweiterter Programme prüfen	+	
Robuste Programmierung		
Hinzufügen von Einträgen zu SORTED TABLE an bestimmter Position	+	
DB-Operationen in Pool/Cluster-Tabellen suchen	++	
Komplexe WHERE-Bedingung in SELECT-Anweisung	+	
Prüfung der SY-SUBRC-Behandlung	++	
Problematische Anweisungen für Ergebnis von SELECT/OPEN CURSOR ohne ORDER BY suchen	+++	
Unsichere Verwendung von FOR ALL ENTRIES	+++	
Dynamische Tests		
ABAP Unit (nur Non-Remote ausführbar)	+++	Wenn Unit-Tests verwendet werden
Suchfunktionen		
Suche von ABAP-Anweisungsmustern	++	u. U. in zentralen Prüflauf

Empfohlene Prüfungen für Default-Variante	Gewichtung	Anmerkung
Anweisungsmuster: CALL FUNCTION 'DB_EXISTS_INDEX' * CALL FUNCTION 'DD_INDEX_NAME' * Regex zur Mailadressenerkennung		aufnehmen
Suche nach unerwünschten Sprachelementen: WAIT BREAK	+	u. U. in zentralen Prüflauf aufnehmen

Wir haben bewusst die Namenskonventionen weggelassen, da die Erfahrung zeigt, dass diese sehr unterschiedlich gelebt werden. Sie können Ihre individuellen Namenskonventionen in einer eigenen Variante unter „Programmierkonventionen/ Namenskonventionen“ bzw. „Erweiterte Namenskonventionen für Programme definieren“ hinterlegen. Letztere Prüfung erlaubt, wie der Name der Prüfung bereits ausdrückt, feinere Einstellungen.

5.5 abapOpenChecks und Code Pal for ABAP

Nutzen Sie bereits die abapOpenChecks, steht Ihnen hier auch eine Prüfroutine „NamingConventions“ zur Verfügung. Hier haben Sie u. a. die Möglichkeit, z. B. generierte BW-Extraktor-Bausteine von der Prüfung auszunehmen (in der Regel entsprechen die generierten Signaturen nicht den eigenen Namenskonventionen).

Siehe auch: https://docs.abapopenchecks.org/checks/ (abgerufen am 05.12.2019)

Prüfungen abapOpenChecks	Gewichtung	Anmerkung
EXIT or CHECK outside of loop	+	
Wrong use of TRY-CATCH	+++	
Functional writing style for CALL METHOD	+	
Obsolete statement	++	
Use icon_ constants	+	

Prüfungen abapOpenChecks	Gewichtung	Anmerkung
Max one statement per line	+	
Kernel CALL	+	
Empty branch	+	
Unused FORM parameter	+	
Function module recommendations	+	
CALL TRANSACTION authority check	+++	Nicht identisch mit Standardprüfung
Avoid use of SELECT-ENDSELECT	+	
Code metric: number of statements per proc. block	+	Ist im Standard vorhanden, dort aber nicht Remote-fähig

Da das Projekt Code Pal for ABAP relativ jung ist, können wir noch keine fundierten Empfehlungen zu den Prüfungen abgeben. Dies wird ergänzt, sobald wir mehr Erfahrung sammeln konnten.

5.6 Vorbereitung auf Migrationsprojekte

5.6.1 Migration auf HANA-Datenbank

Für alle Unternehmen, die derzeit noch ein ECC-System nutzen und ggf. auch keine HANA-Datenbank einsetzen, gibt es Prüfungen, damit Sie Ihre Entwicklungen rechtzeitig auf den Datenbankwechsel vorbereiten können. Die ausgelieferten Prüfvarianten der SAP heißen „Functional DB“ und „Functional DB Addition“, Verfügbarkeit (siehe Hinweis 1912445).

Wie der Name schon vermuten lässt, ist „Functional DB“ Pflicht und „Functional DB Addition“ optional.

Die Variante „Functional DB“ hat das Ziel, Eigenentwicklungen zu bewerten, die auf spezifische DB-Funktionen angewiesen sind und daher bei der Migration auf eine HANA-Datenbank angepasst werden müssen.

Die Prüfungen der Variante „Functional DB Addition“ können wichtige, potenzielle funktionale Fehler und SQL-Anweisungen mit niedriger Performance ermitteln. Die

Prüfungen dieser Prüfvariante sind nicht obligatorisch für die Migration von HANA, werden aber von SAP empfohlen.

Da die Ergebnisse dieser Prüfungen sich überlappen können, ist es ggf. sinnvoll, eine eigene Prüfvariante mit der Kombination der beiden Varianten anzulegen.

Für die statische Analyse von Performance-Aspekten steht die Prüfvariante „Performance DB“ zur Verfügung. Die Ergebnisse aus diesen Prüfungen sollten in jedem Fall zusammen mit den zur Laufzeit ermittelten Daten des SQL-Monitors betrachtet werden.

Es hat sich bewährt, sich schon frühzeitig mit diesen Prüfungen zu beschäftigen. Allgemeine Informationen zum ATC und der HANA-Migration finden Sie auch auf help.sap.com:

https://help.sap.com/viewer/7bfe8cdcfbb040dcb6702dada8c3e2f0/7.5.4/de-DE/2b99ce231e7e45e6a365608d63424336.html (abgerufen am 22.11.2019)

5.6.2 Konvertierung auf S/4HANA

Für eine Konvertierung nach S/4HANA stellt SAP folgende Prüfvarianten zur Verfügung, die unserer Meinung nach auGGch regelmäßig ausgeführt werden sollten (siehe dazu auch Kapitel 2):

- S4HANA_READINESS (nur lokal)
- S4HANA_READINESS_<*S/4HANA Version*> (remote)

Mittels dieser Prüfungen wird unter anderem die Prüfkategorie S/4HANA-Readiness aktiviert, unter der sich auch die Prüfungen auf die Simplification-Database befinden. Bei den S4HANA_READINESS_<*S/4HANA Version*>-Prüfungen ist in den Einstellungen die konkrete Release-ID für die entsprechenden Simplifizierungen eingetragen.

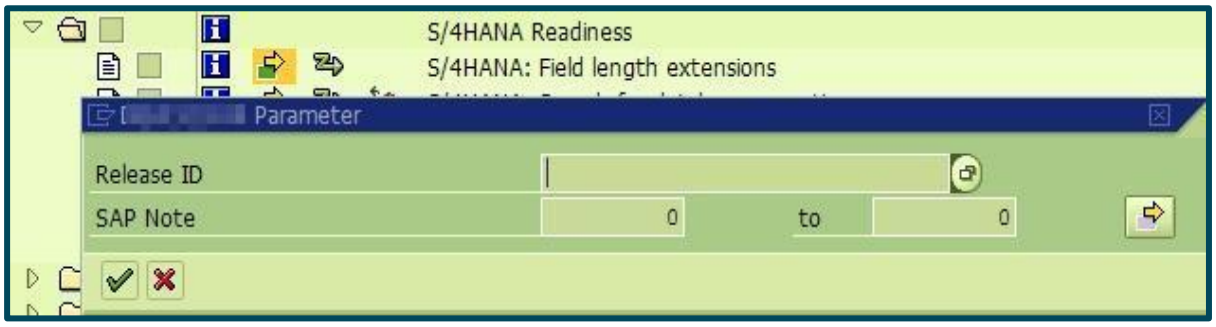


Abbildung 25 S/4HANA-Readiness-Release-Einstellungen

Für eine höhere Version werden automatisch alle darunterliegenden Versionen ebenfalls geprüft. Jedoch können Sie in einer eigenen Variante das Feld Release-ID auch leer lassen, dann benötigen Sie keine der speziellen Varianten. Um eine kontinuierliche Überprüfung des Systems zu bewerkstelligen, ist dies durchaus ein

Best-Practice. Unsere Empfehlung ist, alle S/4HANA-spezifischen Tests auszuwählen. Außerdem sollten Sie immer nach den aktuellen OSS-Hinweisen zu diesen Prüfungen suchen, da es hier regelmäßig Updates gibt.

- Empfohlene SAP-Hinweise für die Verwendung von ATC zur Durchführung von Remote-Analyse: https://launchpad.support.sap.com/#/notes/2364916 (abgerufen am 05.12.2019)
- SAP-Readiness-Check für S/4HANA: https://launchpad.support.sap.com/#/notes/2290622 (abgerufen am 05.12.2019)

6 Implementierung eigener Prüfungen

6.1 Wann brauche ich eigene Prüfungen?

In den meisten Fällen werden Sie voraussichtlich mit den von SAP ausgelieferten Standardprüfungen auskommen. Falls Sie aber etwas vermissen, können Sie Ihre eigenen Prüfungen implementieren.

Bevor Sie das tun, sollten Sie unbedingt prüfen, ob nicht doch eine der Standardprüfungen so eingestellt werden kann, dass sie Ihren Anforderungen entspricht. Am besten erstellen Sie sich dazu vorab passende Testobjekte, mit denen Sie testen können, wie die Prüfungen reagieren. Ähnlich wie bei ABAP Unit Tests hat es sich bewährt, dafür mindestens ein positives und mehrere negative Beispiele zu finden.

Nutzen Sie auch die vorhandenen Prüfungen in der Kategorie „Suchfunktionen“. In vielen Fällen können Sie damit um die Entwicklung eigener Prüfungen herumkommen. Ein Nachteil ist allerdings, dass die Findings hierbei für den betroffenen Entwickler meist nicht selbsterklärend sind.

Zu guter Letzt können Sie prüfen, ob Ihnen die abapOpenChecks oder der Code Pal for ABAP weiterhelfen. Hierbei handelt es sich um Open-Source-Projekte, die viele weitere Prüfungen bereitstellen und durch die Community (also auch Sie!) weiterentwickelt werden. Weitere Details zu den abapOpenChecks und dem Code Pal for ABAP finden Sie in Anhang B bzw. Anhang C.

Denken Sie auch daran, dass SAP selbst die Prüfungen immer weiterentwickelt. Ggf. gibt es also schon einen SAP-Hinweis, der die von Ihnen vermisste Prüfung enthält und in Ihr System eingespielt werden kann. Falls Sie dennoch zum Schluss kommen, dass die bestehenden Prüfungen Ihren Fall nicht abdecken können, können Sie die Ärmel hochkrempeln!

6.1.1 Fragenkatalog vor der Implementierung von eigenen Kundenprüfungen

Fragestellungen vor der Implementierung	Empfehlungen
Welche Prüfungen in meinen Guidelines werden vom Standard nicht abgebildet?	Vor Implementierung einer kundenindividuellen Prüfung ist es essenziell, die SAP-Standardprüfung zu verstehen.
Welche meiner Regeln in den Guidelines sind als eigene kundenindividuelle Prüfung umsetzbar?	Nicht jede Regel innerhalb der Guidelines kann als automatisierte Prüfung abgebildet werden. Daher ist ein grober Algorithmus bzw. ein Design notwendig, bevor Sie mit der Entwicklung einer eigenen Prüfung starten.

Fragestellungen vor der Implementierung	Empfehlungen
Wie viele False-Positives generiere ich mit meiner Prüfung (Wie exakt kann die Prüfung implementiert werden?)?	Um Falschmeldungen möglichst zu vermeiden, ist analytisches Verständnis des zu prüfenden Sachverhaltes notwendig. Je mehr False-Positives entstehen, desto länger benötigen die Entwickler, den Code zu analysieren. Zu viele False-Positives verringern die Akzeptanz der Entwickler.
Welche Konfigurationsmöglichkeiten benötige ich für meine eigenen Prüfungen?	Wie kann ich die Prüfungen sinnvoll parametrisieren (z. B. Einschränkung auf spezielle Objekttypen)?
Welche Objekttypen betrifft die Prüfung?	Es ist wichtig zu wissen, ob Data-Dictionary-Objekte oder Coding geprüft werden, da für die Implementierung unterschiedliche Basisklassen verwendet werden.
Lässt sich die Prüfung Remote-fähig implementieren?	Abwägen, wie hoch der Aufwand ist, um die Prüfung Remote-fähig zu machen (wegen lokaler Datenabhängigkeiten).
Wo finde ich die Daten zu meiner betreffenden Prüfung?	Ein tiefes Verständnis vom SAP-Standard ist notwendig. Beispiele hierfür sind: • aus welchen SAP-Tabellen oder Methoden Daten ermittelt werden können, bzw. • für Coding: Wie die Statement- und Token-Tabellen zu interpretieren sind
Was ist das Ziel meiner Prüfung?	Stoppen des Transportes oder Hilfe beim Code-Review

6.1.2 Allgemeine Vorgehensweise

Die Erweiterung von Prüfungen basiert auf Objektorientierung und Vererbung von SAP-Standard-Prüfklassen. Sie können als Grundlage für Ihre Prüfungen auf verschiedene Basisklassen zurückgreifen, die der Code Inspector zur Verfügung stellt. Dazu gehören z. B. diese Klassen:

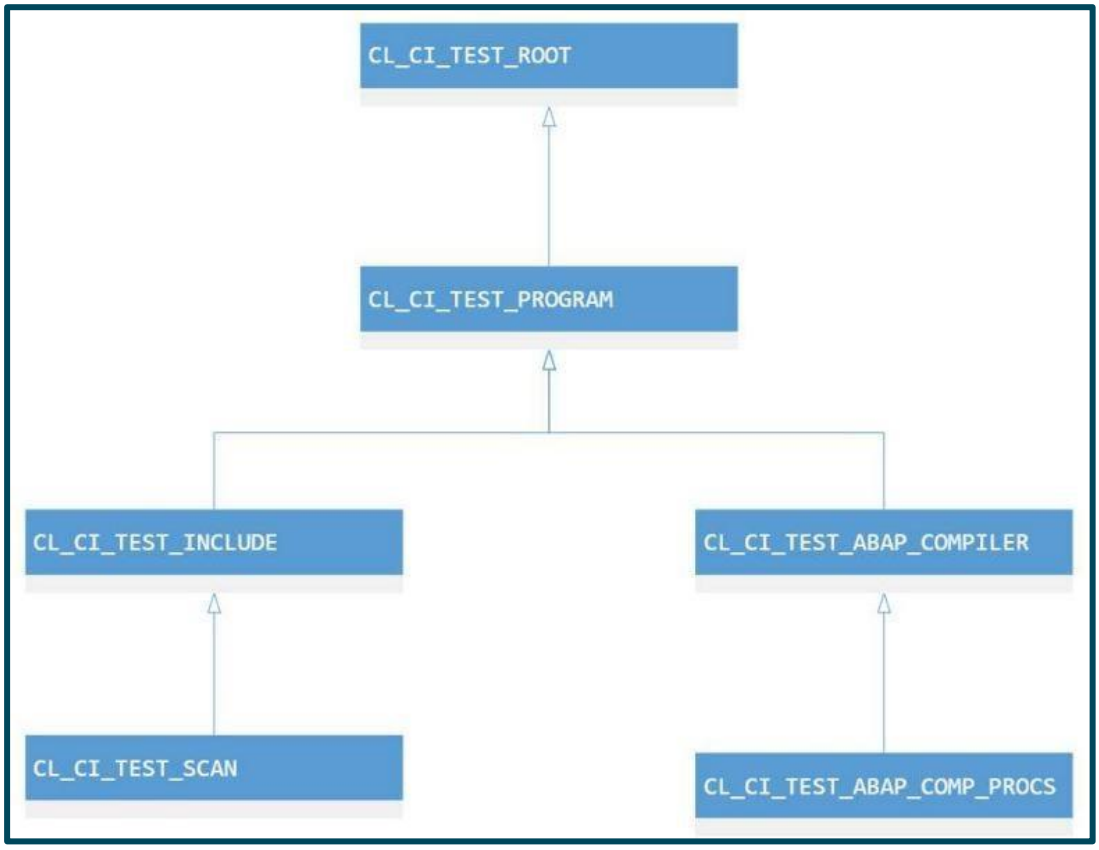


Abbildung 26 Hierarchie der Prüfklassen

Die Implementierung folgt diesen Schritten:

1. Prüfkategorie anlegen
   SAP bietet schon vordefinierte Kategorien an, wie beispielsweise Sicherheits- oder Performance-Prüfung, die man nutzen kann. Will man die selbst entwickelten Prüfungen in einer eigenen Kategorie zusammenfassen, erzeugt man eine neue Klasse als Kopie von CL_CI_CATEGORY_TEMPLATE. Darin ändert man dann den Titel und die Beschreibung, und über die Transaktion SCI und Verwaltung von Tests kann man die Kategorie auch aktivieren und sie in der Variantenkonfiguration sichtbar machen.

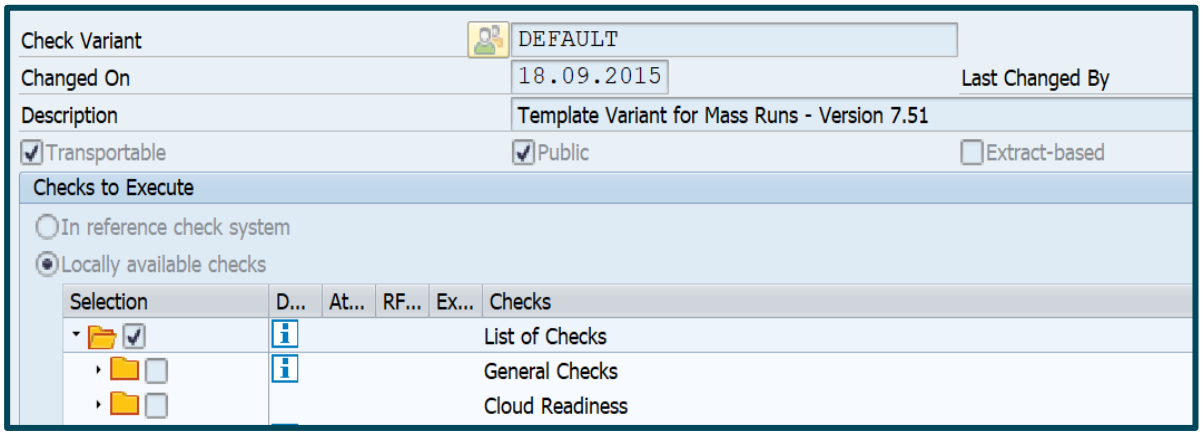


Abbildung 27 Prüfkategorien

2. Die nächste Aktion ist es, die Prüfung selbst zu implementieren. Für die Implementierung stellt SAP Beispielklassen bereit, die Sie als Kopiervorlage nutzen können. Die Klasse CL_CI_TEST_SCAN_TEMPLATE ist für die Prüfung von ABAP Code vorgesehen, für andere Entwicklungsobjekte können Sie die Klasse CL_CI_TEST_ROOT_TEMPLATE verwenden. Die Templates sind größtenteils leer, Sie müssen einige Methoden mit Coding füllen.

   a. CONSTRUCTOR (verpflichtend):
      i. Im Konstruktor sind einige Attribute zu setzen, die bereits zu Teilen im Root-Template zu sehen sind.
      ii. Tipp für jene, die ATC-Remote verwenden: Das REMOTE_RFC_ENABLED-Flag muss gesetzt sein, um die Prüfung über das Remote-ATC-System verwenden zu können.
   b. RUN (verpflichtend):
      i. Abhängig von der Art der Prüfung wird hier die Objektkollektion durchgeführt. Aus Performance-Gründen ist es manchmal notwendig, die Objekte in der Run-Methode zu sammeln (in einer eigenen Tabelle vom Typ SCIR_TOBJ - als Klassenattribut) und in der Methode RUN_END zu prüfen.
   c. ANALYZE_PROC:
      i. Hier können Sie den ABAP Code prüfen. Neben dem geparsten Quelltext und den Kommentaren stehen hier auch die Parameter des Aufrufs zur Verfügung. Üblicherweise wird in dieser Methode auf bestimmte Schlüsselwörter im Quelltext geprüft.
   d. IF_CI_TEST~QUERY_ATTRIBUTES (optional):
      i. HAS_ATTRIBUTES muss im Constructor gesetzt werden, wenn die Prüfung weitere Attribute benötigt.
   e. GET_ATTRIBUTES/PUT_ATTRIBUTES (optional):
      i. GET_ATTRIBUTES/PUT_ATTRIBUTES wird verwendet, um Eigenschaften zu liefern bzw. zu setzen
      ii. Sie können die Klasse CL_CI_QUERY_ATTRIBUTES nutzen, um diese Attribute per Pop-up bei der Erstellung der Prüfvariante im Code Inspector zu pflegen
   f. IF_CI_TEST~DISPLAY_DOCUMENTATION (optional):
      i. Wenn das Attribut HAS_DOCUMENTATION im Konstruktor gesetzt ist, wird diese Funktion für die Anzeige der Klassendokumentation verwendet.
   g. ADD_INFO:
      Zur Ausgabe der Prüfergebnisse verwenden Sie die ADD_INFO-Methode.
      Falls Ihre Prüfung nicht von CL_CI_TEST_ABAP_COMP_PROCS abgeleitet ist, nutzen Sie dafür die INFORM-Methode aus der

Basisklasse Ihrer Prüfung.

3. Nachdem Sie die Implementierung abgeschlossen haben, verwenden Sie in der Transaktion SCI das Menü Code Inspector > Verwaltung von > Tests, um Ihre Prüfung und Ihre Prüfkategorie zu aktivieren.

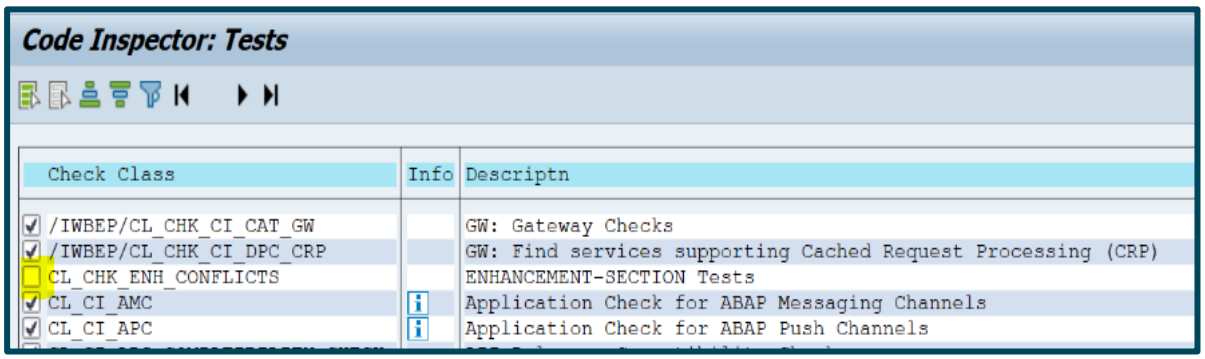


Abbildung 28 Verwaltung von Prüfklasse

4. Textelemente und Dokumentation (SE61) für Kategorie und Prüfung pflegen. Zuletzt pflegen Sie bitte die Textelemente und Dokumentation in der Transaktion SE61 für Kategorie und Prüfung.

5. Wie teste ich meine Prüfung?
   Es gibt zwei Optionen, die Prüfung zu testen:

   a. Schreiben von Unit-Tests zu jeder implementierten Funktion
   b. Anlage eines Testpakets, das Elemente enthält, die Ihre Prüfung finden sollte

Detaillierte Anleitungen und Beispiele finden Sie in folgenden Referenzen:

Referenz A: „How to write an ATC-Check“ (Stand 2018):
https://www.sap.com/documents/2018/09/905bfdab-1a7d-0010-87a3-c30de2ffd8ff.html
(abgerufen am 13.11.2019)

Referenz B: „How to Build a new Check for the Code Inspector“ (Stand 2004):
https://blogs.sap.com/2018/09/06/remote-code-analysis-in-atc-how-to-write-an-atc-check/ (abgerufen am 13.11.2019)

Falls Sie in die Thematik der Eigenentwicklung von Prüfungen und Objektkollektoren tiefer einsteigen möchten, empfehlen wir Ihnen die Lektüre des „Praxishandbuch SAP Code Inspector” (Randolf Eilenberger, Frank Ruggaber und Reinhard Schilcher, SAP PRESS). Leider ist dieses Buch auf dem Stand 2011.

6.1.3 Beispiele für Eigenentwicklungen

- Prüfung von Namenskonventionen für DDIC-Elemente (rekursiv)
- Prüfung von Namenskonventionen für Objekte, die nicht im ausgelieferten Standard geprüft werden (Beispiel: Erweiterungsimplementierungen)

- Prüfung auf Verwendung von obsoleten Klassen oder Funktionsbausteinen
- Vorhandensein von Übersetzungen in Default-Sprachen (DE/EN)
- Prüfung, ob Felder eines Append der ZZ-Namenskonvention von SAP folgt (siehe auch abapOpenChecks: Prüfung 33)
- Prüfung, ob Tabellen als Column-Store aufgebaut sind (HANA-System)
- Prüfung, ob diverse Objekttypen verwendet werden. Damit können diverse Objekttypen beim Transportrelease blockiert werden.
- Beispiele: Smartforms, SAP-Scripts
- Existenz von Kopfkommentaren
- Prüfung auf Zuordnung von Berechtigungsgruppen für Tabellen oder Programme
- Beispiele von Objektkollektoren – Objektkollektoren können im Code Inspector genutzt werden, um individuelle Selektionen der zu prüfenden Objektmenge zu implementieren.
  - Selektion nach Zeitraum
  - STMS-Importqueue-basierter Kollektor
  - Hinweis 2525636 ATC: externes API, um Objekt-Provider-ID erweitert: https://launchpad.support.sap.com/#/notes/2525636 (abgerufen am 05.12.2019)

6.2 Allgemeines zu Remote-Prüfungen

6.2.1 Wann brauche ich eine Remote-fähige Implementierung?

In aktuellen SAP-Releases stellt der ATC neue Prüfungen bereit, die in älteren Systemen noch nicht vorhanden sind. Dazu gehören z. B. die S/4HANA-Readiness-Checks. Mit der Möglichkeit von Remote-Prüfungen können diese Prüfungen auch gegen Systeme mit älteren Release-Ständen gefahren werden, sogar solche, in denen der ATC noch gar nicht zur Verfügung steht (ECC 7.00-7.02). Mit einem zentralen ATC-System (>= 7.51) sind Sie in der Lage, den Code mehrerer Entwicklungssysteme mit einer einheitlichen, zentralen Prüfvariante zu validieren.

6.2.2 Was muss ich tun, wenn ich eine Prüfung Remote-fähig implementiere?

Vorbereitend kann es je nach Release-Stand im zu prüfenden System notwendig sein, spezielle SAP-Hinweise einzuspielen.

- Hinweis 2375864: „ATC-Remote-Prüfungen: Entwicklerszenario“ https://launchpad.support.sap.com/#/notes/2375864 (abgerufen am 05.12.2019)
- Hinweis 2364916: „Empfohlene SAP-Hinweise für die Verwendung von ATC zur Durchführung von Remote-Analyse“ https://launchpad.support.sap.com/#/notes/2364916 (abgerufen am 05.12.2019)

Bei der Entwicklung Ihrer Prüfungen sollten Sie folgende Punkte beachten:

- Im Constructor muss das REMOTE_RFC_ENABLED-Flag gesetzt sein.
- Die Datenkollektion und Prüfung werden in der RUN-Methode implementiert.

6.2.3 Wo muss ich meine Remote-Funktionsbausteine aufrufen (Methoden) und wann brauche ich sie?

In der RUN-Methode der Prüfung werden die Objekte im Fall einer Remote-Prüfung aus dem Satellitensystem mit Hilfe der RFC-fähigen Funktionsbausteine der Funktionsgruppe SABP_COMP_PROCS_E ausgelesen und ins zentrale ATC-System geholt (siehe Aufruf der Methode GET). Wenn die Funktionsgruppe SABP_COMP_PROCS_E noch nicht im Satellitensystem existiert, kann sie mit Hilfe des SAP-Hinweises 2270689 eingespielt werden.

https://launchpad.support.sap.com/#/notes/2270689 (abgerufen am 05.12.2019)

Falls Sie eine eigene Datensammlung brauchen, sollten Sie diese entsprechend als RFC-Aufruf innerhalb der RUN-Methode implementieren.

6.2.4 Was ist sonst dabei zu beachten?

Alle Remote-fähigen Prüfungen können nur statische Prüfungen durchführen. Für eine Laufzeitanalyse sollten Sie die dafür vorgesehenen Transaktionen wie SCMON nutzen.

Was sind häufige Hindernisse bei Remote-Prüfungen?

Beim Schreiben eigener Remote-Prüfungen sind folgende Punkte zu beachten:

- Schaffen Sie keine Abhängigkeiten von lokalen Daten zum globalen Prüfsystem (z. B.: Abfragen der TADIR-Tabelle im zentralen System, um Objektinfos zu erhalten).
  Um Dictionary-Objekte in Remote-Systemen zu analysieren, muss auf dem jeweiligen System ein RFC-fähiger Baustein erstellt werden, der die TADIR, DD02L…. auswertet. Um sich diese Objekte aus der Ergebnisliste des zentralen ATC-Systems anschauen zu können, muss der Hinweis 2831860 implementiert werden. Ohne den Hinweis wird immer versucht, das Dictionary-Objekt im zentralen ATC-System zu finden.
  https://launchpad.support.sap.com/#/notes/2831860
- Verwenden Sie keine Release- bzw. Kernel-abhängigen Funktionalitäten (RTTI).
- Verwenden Sie die Methoden der Klasse CL_CI_TEST_ABAP_COMP_PROCS, um Informationen am lokalen System zu erhalten.
- Weitere Informationen, die Sie aus dem lokalen System erhalten wollen, müssen Sie über ein Remote-fähiges Funktionsmodul implementieren. Die

RFC-Destination bei Aufruf ist in Ihrer Prüfung nicht verfügbar. Die aktuell gültige SourceID ist im Klassenattribut der Klasse CL_CI_TEST_ROOT=>SRCID verfügbar und muss noch in eine RFC-Destination gecasted werden. Dies kann mit CL_ABAP_SOURCE_ID=>GET_DESTINATION erfolgen.

Falls Sie bereits existierende eigene Prüfungen zu Remote-fähigen Prüfungen umarbeiten wollen, werden Sie nicht um die Implementierung von RFC-Funktionsbausteinen für Ihre Satellitensysteme herumkommen.

Sie sollten dabei auf folgende Punkte achten:

- Übertragen Sie möglichst kleine Datenmengen (z. B. bei Datenstrukturen nur die benötigten Felder).
- Falls Sie klassenbasierte Objekte übertragen, beschränken Sie sich möglichst nur auf Objektattribute, die Sie zwingend weiterverarbeiten müssen.
- Bei Existenzprüfungen von lokalen Daten sollten Sie diese auf dem Satellitensystem durchführen und nur einen booleschen Wert übermitteln.

6.3 Implementierung eigener Objektkollektoren

Prüfläufe bestehen aus zwei Komponenten, aus den in der Prüfvariante gesammelten Einzelprüfungen und der zu prüfenden Objektmenge.

Aktiviert man das ATC bei der Transportfreigabe, ergibt sich die Objektmenge implizit aus dem Inhalt des Transports. Möchten Sie Aussagen über die Code-Qualität eines Gesamtsystems treffen, so können Sie dies relativ einfach über die Repository-basierte Objektselektion bei der Anlage einer Konfiguration vornehmen. Alternativ können Sie über den Radio-Button „nach Objektmenge“ weitere, im CI definierte Objektmengen einbinden.

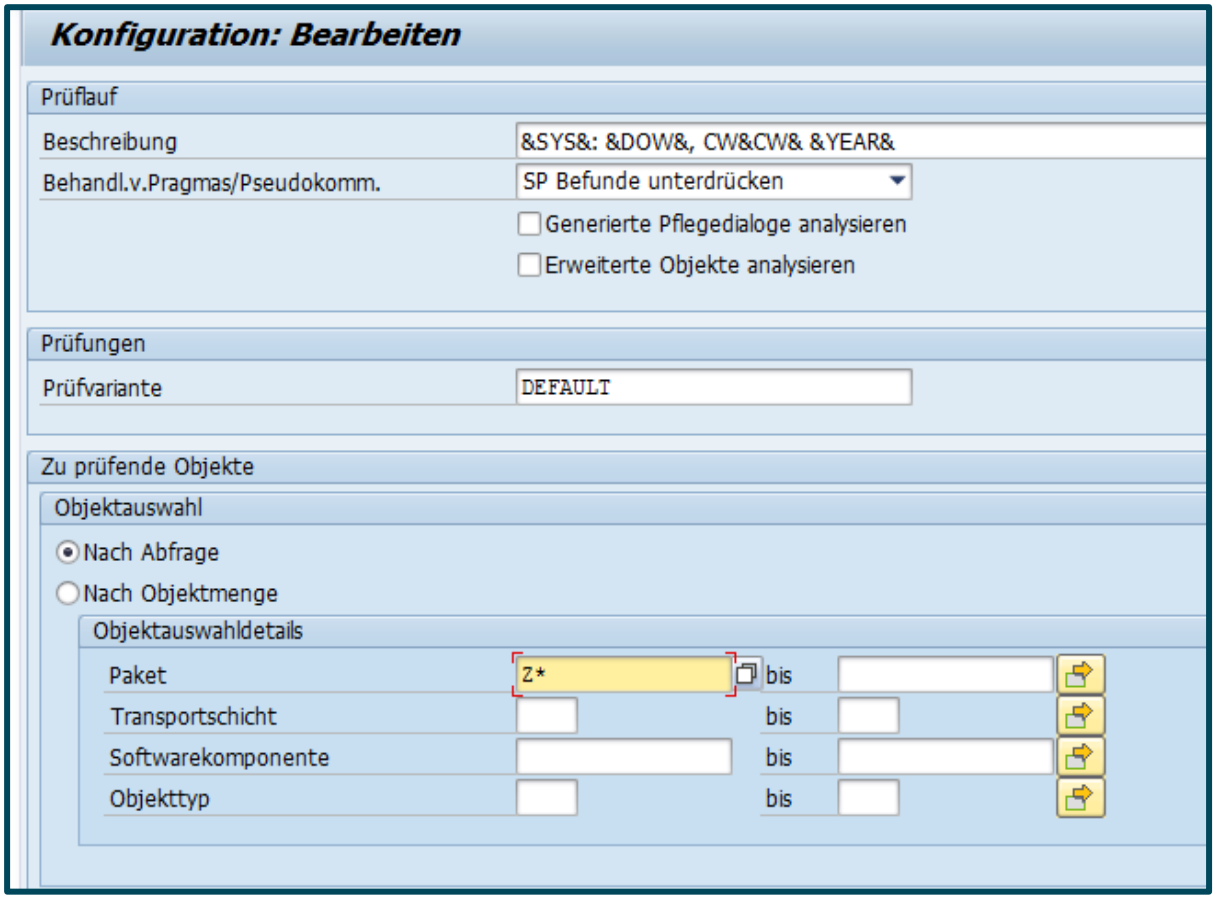


Abbildung 29 Selektion nach Objektmenge

Die „CI Objektmengen“ bieten nun die Option, eigene Objektkollektoren zu entwickeln. Diese können über den fünften Tab-Reiter in der Objektmengendefinition eingebunden werden.

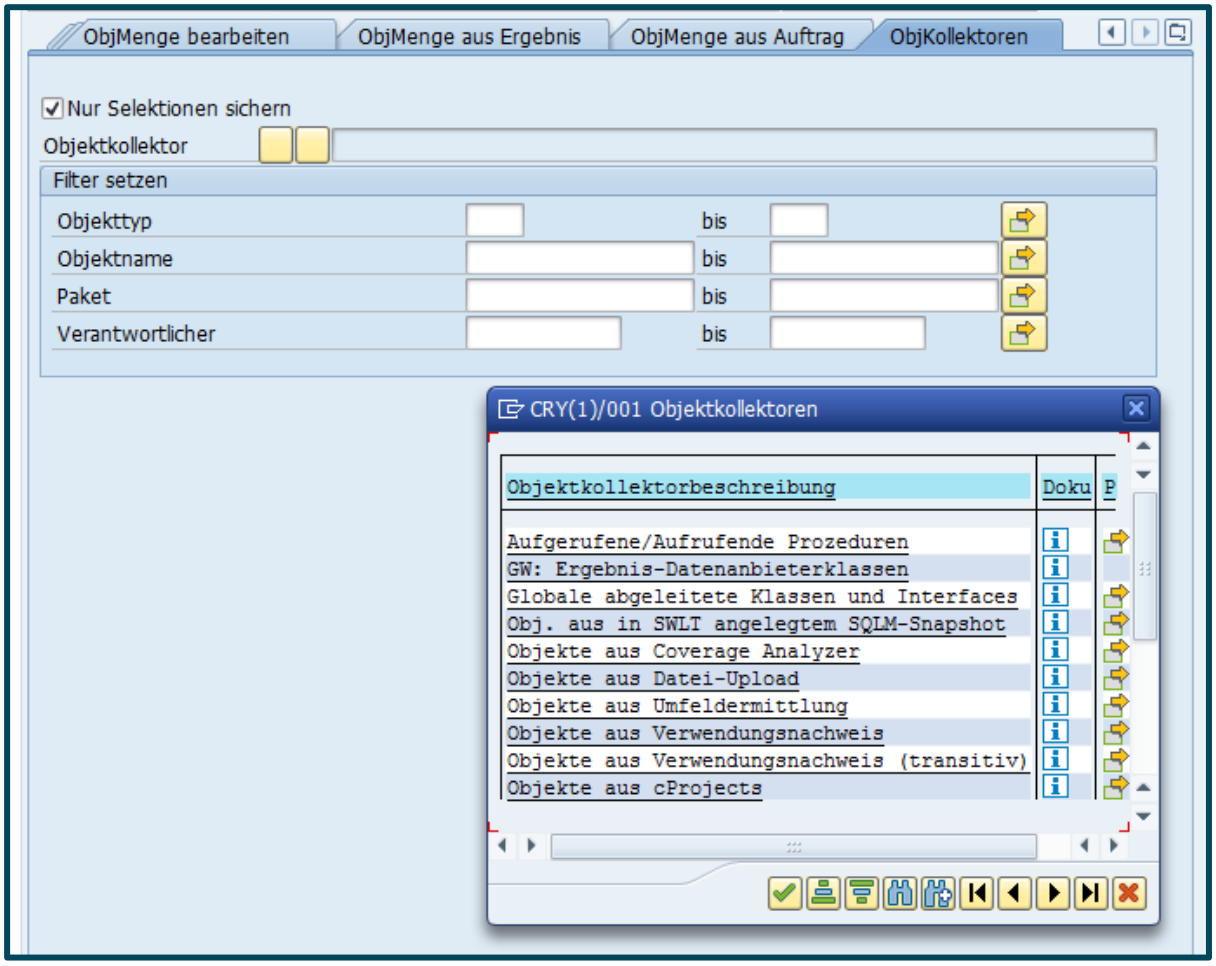


Abbildung 30 Auswahl Objektkollektoren

Die Auswahl der ausgelieferten SAP-Kollektoren vermittelt einen Eindruck davon, welcher Art solche Kollektoren sein können. Sie ermitteln Teilmengen der Software, die in einem besonderen, engeren Zusammenhang stehen.

Besonders einfach lassen sich Kollektoren auf Basis im System vorhandener Objektkollektionen entwickeln. Beispielhaft seien hier das TMS, Traces oder auch UPL bzw. SCMON genannt.

Mit Hilfe des TMS lassen sich Objekte ermitteln, die aktuell bearbeitet werden, Objekte, die kürzlich bearbeitet wurden oder auch Objekte, die importiert wurden. In Kombination mit der entsprechenden Prüfung können Sie Aussagen ableiten, welche Befunde bearbeitet werden könnten oder ob die Objekte eines Imports noch korrekt sind. Letzteres betrifft u. a. Objekte, die nicht statisch beim Import geprüft werden.[8]

[8] Beispielhaft seien hier Adobe Forms genannt. Werden im Kontext DDIC Objekte eingebunden, die nicht in derselben Form im importierten System vorliegen, kommt es erst bei der Ausführung zu einem Fehler. Im Zusammenspiel mit einer – eigenen – Prüfung, die die Formulare prüft, können diese Fehler erkannt werden.

Die Aufzeichnung von Nutzungsstatistiken im Produktivsystem via UPL (Usage Procedure Logging) ermittelt die Teilmenge der verwendeten Entwicklungsobjekte. Prüfläufe mit diesen Daten zeigen ein genaueres Bild als die generische Selektion aus dem Repository. Zusammen mit den S/4HANA-Readiness-Prüfungen ermöglicht dies eine bessere Abschätzung des Aufwands eines S/4HANA-Migrationsprojekts.[9]

Die Entwicklung eigener Kollektoren ähnelt der Entwicklung von eigenen Prüfungen. Eine Kollektorenklasse erbt von CL_CI_COLLECTOR_ROOT.

Im Konstruktor werden Attribute gefüllt, mit Redefinition der Methoden IF_CI_COLLECTOR~GET_ATTRIBUTES, IF_CI_COLLECTOR~PUT_ATTRIBUTES und IF_CI_COLLECTOR~QUERY_ATTRIBUTES erfolgt das Handling der Laufzeitparameter. Mit der redefinierten Methode COLLECT wird die Ermittlung durchgeführt.

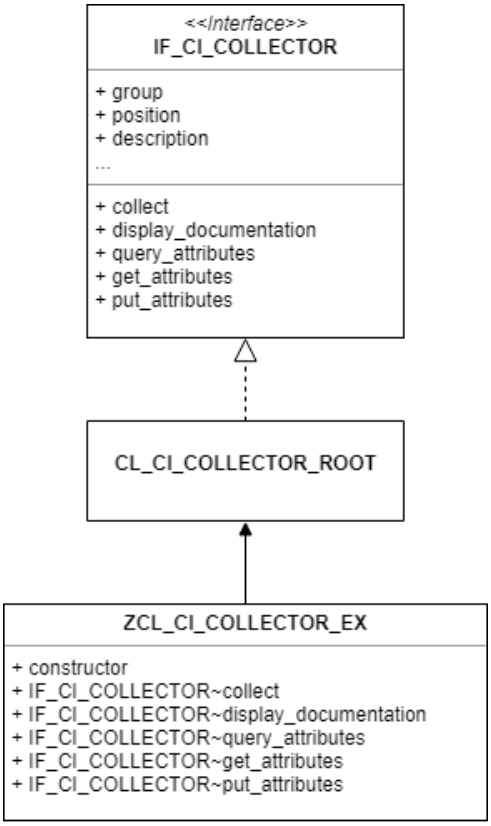


Abbildung 31 UML-Diagramm Object Collector

[9] UPL-Daten lassen sich mit Hilfe des Programms /SDF/SHOW_UPL deserialisieren und dann über die Objektmenge „Datei-upload“ für Prüfläufe zur Verfügung stellen. Dies ist aber zum einen ein ständiger manueller Aufwand und zum anderen lassen sich in großen Systemen nicht alle Datensätze auf einmal deserialisieren. Über eine Zusatzentwicklung lässt sich dies automatisieren.

Referenz C: „Code Inspector – Custom Object Collectors“

https://blogs.sap.com/2012/08/23/code-inspector-custom-object-collectors/ (abgerufen am 25.09.2019)

Referenz D: „Enabling ATC for remote checks“

http://saptechnicalguru.com/remote-atc/ (abgerufen am 25.09.2019)

6.4 ABAP-Unit-Tests bei eigenen Prüfungen

Bei der Entwicklung von eigenen Prüfungen sind Unit-Tests extrem hilfreich. Sie sollten sich dabei darauf konzentrieren, den Code so zu strukturieren, dass Unit-Tests möglichst einfach zu implementieren sind. Falls Sie im Unit-Test die Klasse CL_CI_CHECK_RESULT für den Aufruf verwenden, sollten Sie daran denken, dass Sie für die Ausführung der Unit-Tests geeignete Prüfvarianten anlegen müssen und diese dann auch mittransportiert werden müssen.

Alternativ können Sie sich auch mit den Möglichkeiten der abapOpenChecks auseinandersetzen, dort sind die Klassen klarer strukturiert und ohne Abhängigkeit von Prüfvarianten testbar. Auch der Code Pal for ABAP beinhaltet Unit-Tests und kann als Vorlage für ihre eigenen Prüfungen dienen.

Ein weiterer Vorteil der abapOpenChecks und des Code Pal for ABAP ist, dass man die neuen Prüfungen problemlos via GitHub mit der Community teilen kann.

7 Der lebende ATC im Unternehmen

ATC ist im Unternehmen eingeführt – was nun?

SAP hat mit ATC ein umfangreiches, dynamisches Tool entwickelt, welches gewartet und administriert werden muss. Die Erfahrung im Unternehmen hat gezeigt, dass es sinnvoll ist, ein ATC-Team, z. B. bestehend aus SAP-Anwendungsentwicklern, zu bilden (siehe Kapitel 4, Abschnitt „Einführung im Unternehmen“). Dieses Team ist – meist als Zusatzaufgabe – für folgende Aufgaben zuständig:

- Die Bewertung und ggf. Freigabe der Befreiungsanträge
- Beantwortung von Fragen („Was bedeutet dieser Befund?”)
- Aufgaben im Zusammenhang mit Upgrades oder Systemkopien (z. B. neues Aufsetzen des Qualitätssicherungssystems als Kopie)
- Ggf. Anpassung der Prüfungen durch:
  - Prioritätsänderung oder Anpassung der Konfiguration einzelner Prüfungen, z. B. weil es zu viele Befreiungsanträge mit False-Positives gibt
  - Zusätzliche Nutzung existierender Non-SAP-Prüfklassen, z. B. abapOpenChecks (siehe Anhang B) oder Code Pal for ABAP (siehe Anhang C)
  - Entwicklung eigener Prüfklassen (siehe Kapitel 6 „Implementierung eigener Prüfungen“)

Im folgenden Kapitel werden diese Aspekte detaillierter beschrieben.

7.1 Aktionen bei SAP-Systemkopie

Eine Systemkopie wird in der Regel erstellt, um ein Qualitätssicherungssystem oder Entwicklungssystem auf einen aktuellen Datenstand zu bringen. Für unsere Betrachtungsweise ist die Tiefe der Systemlandschaft (in der Regel 3- oder 2-Systemlandschaft) nicht relevant.

Dennoch gibt es mehrere verschiedene Konstellationen. Der grundsätzliche Gedanke ist, im Quellsystem der Kopie Einstellungen vorzunehmen, die dort vielleicht nicht benötigt werden, durch die dann aber im Zielsystem der Kopie Einstellungsaufwand gespart wird.

- Mit zentralem Prüfsystem (im **Remote-ATC**)
  - Vor dem Kopieren
    - Legen Sie eine zentrale Prüfvariante im Zielsystem an, die anschließend in das Quellsystem (Kopiervorlage) transportiert wird. Damit wird sichergestellt, dass die ATC-Konfiguration im Zielsystem (des Kopiervorgangs) nach erfolgter Systemkopie erhalten bleibt.

- Nach dem Kopieren
  - Set-up ATC-Konfiguration:

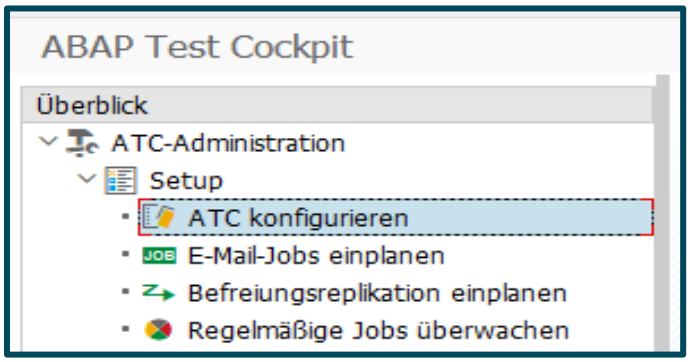


Abbildung 32 ATC-Konfiguration vornehmen (Transaktion: ATC)

  - Werden auf dem Quellsystem (z. B. Produktivsystem) keine ATC-Prüfungen durchgeführt, empfehlen wir, die ATC-Konfiguration analog dem Zielsystem anzulegen.
  - Zu den Nacharbeiten zählt das Anlegen von wiederkehrenden Aktivitäten, wie Jobs. Siehe Set-up-Menü in der ATC-Konfiguration (Abbildung 7-1).
- Detailliertere Informationen zur Konfiguration entnehmen Sie der jeweiligen SAP-Hilfe.

- Non-Remote-ATC
  - Vor dem Kopieren
    - Transportieren Sie die Prüfvariante in das Quellsystem der Kopie (als Kopiervorlage)
  - Nach dem Kopieren
    - Falls das geprüfte System neu aufgebaut wird: Einstellung der automatischen ATC-Prüfung bei Transportfreigabe (Transaktionscode: ATC -> Konfigurieren) sowie Transaktion SE03 „Globales Customizing Transport Organizer“, Einstellung: „Prüfung bei Freigabe“ im Quellsystem der Kopie
    - Falls das Master-System neu aufgebaut wird: neue Baseline nach Systemkopie erstellen.
  - Hinweise:
    - Genehmigende im Zielsystem werden mit der Systemkopie überschrieben. Tipp: Vor der Kopie einen Export der Genehmigenden erstellen und anschließend die Benutzer per Copy-and-paste einfügen.
    - Befreiungen und Befreiungshistorie gehen verloren!
    - Ergebnisse der ATC-Prüfläufe gehen verloren.

Fazit: Verwendung eines zentralen Prüfsystems (Remote-ATC) ist zu empfehlen!

Alternative für die letzten beiden Punkte: In Abhängigkeit des Systemkopie-Tools können ATC-spezifische Tabellen durch eine Systemkopie vom Überschreiben ausgeschlossen werden. Siehe auch Blog-Eintrag https://answers.sap.com/questions/10168548/atc-exemptions-and-system-copies.html (abgerufen am 22.11.2019)

7.2 Aktionen bei System-Upgrade

Vor einem System-Upgrade sollte zum Vergleich

- ein Screenshot der Standard-Prüfvariante (am besten auch mit den Details aus den Attributen/Parametern der einzelnen Prüfungen) erstellt werden
- eine Gesamtprüfung des eigenen Codings im System erstellt werden (insbesondere sollte die Statistik gesichert werden. Konkrete DB-Tabellen siehe auch Blog-Eintrag https://answers.sap.com/questions/10168548/atc-exemptions-and-system-copies.html; abgerufen am 22.11.2019)

Nach dem Upgrade sollten diese Aktionen wiederholt werden.

Durch den Vergleich können folgende Änderungen erkannt werden:

- neue Prüfungen
- geänderte Standardprioritäten
- geändertes Verhalten der Prüfungen

Daraus ergeben sich dann möglicherweise Änderungen an der Prüfvariante oder den Prioritäten.

7.3 Einheitliches Vorgehen bei Befreiungsanträgen

Falls es mehrere Genehmiger gibt, empfiehlt es sich, ein zentral zugreifbares Dokument anzulegen, wo generelle „interne Regeln“ bei der Behandlung von Befreiungsanträgen hinterlegt werden, z. B.:

- „Befunde zu generiertem Coding können immer befreit werden.“
- „Third-Party-Coding im Kundennamensraum kann immer befreit werden, aber Support durch Fremdfirma muss geklärt sein.“

7.4 Häufig gestellte Fragen im ATC-Alltag

Vor allem wenn ATC-Prüfungen obligatorisch bei Transportfreigabe sind, werden immer wieder bestimmte Arten von Anfragen an das ATC-Team gestellt. Hier wollen wir einige Hinweise geben.

7.4.1 Was bedeutet dieser Befund?

Hier kann das ATC-Team als Experten-Team dienen, evtl. mit einem unternehmensinternen FAQ-Dokument oder Wiki.

Bei den Befunden, die aus der erweiterten Programmprüfung (SLIN) kommen, sind oft die im ATC angezeigten Informationen nicht ausreichend verständlich. Hier empfiehlt es sich, für das jeweilige Objekt direkt die erweiterte Programmprüfung zu starten.

Hinweis: In Zusammenhang mit der SLIN ist auch interessant, dass Fehlermeldungen der Prio 3 in SLIN (dort linke Spalte, in rot) zurzeit auf Prio 3 in ATC (d. h. bei Freigabeprüfung nicht zu beheben) gemappt werden. Damit ist das ATC dann weniger strikt als die erweiterte Programmprüfung. (SLIN kennt die Unterteilung nach zwei Dimensionen: Fehler/Warnung/Info und Prios)

7.4.2 Checkliste bei unterschiedlichen Prüfergebnissen

Gelegentlich kann sich die Frage stellen: „Warum bekomme ich in System X das Prüfergebnis A und in System Y für das gleiche Objekt das Ergebnis B?“

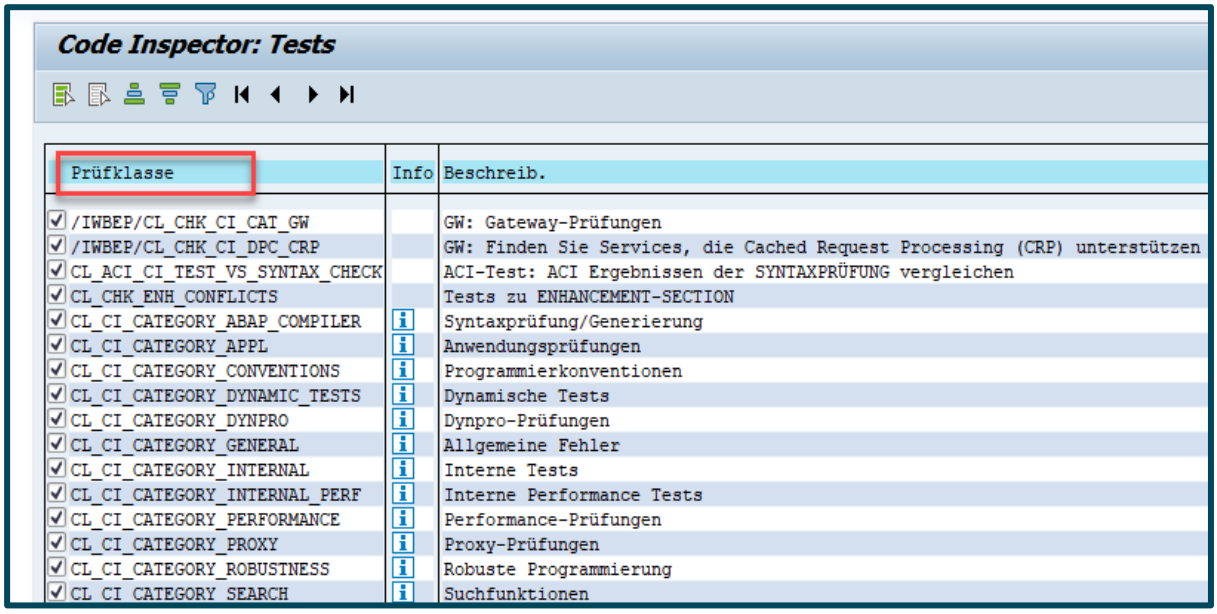
Hier ist zu prüfen:

- Sind die Objektversionen unterschiedlich?
  - Vergleich über Versionshistorie oder bei mehreren Objekten/Teilobjekten über Transaktion SREPO
- Sind Ausnahmen mittlerweile vorhanden, aber nicht in allen Systemen?
  - Ausnahmen stehen in Tabelle SATC_CI_EXEMPT
- Sind die Prüfvarianten unterschiedlich?
  - Achtung: Prüfvarianten müssen als transportierbar gekennzeichnet werden, und teilweise nach dem Import ins System noch über Menü im SCI (unter „Hilfsmittel“) importiert werden.
  - Technisch wird ein Programm mit der Namenskonvention z. B. Z_STANDARD==========VC erzeugt
- Haben die Systeme einen unterschiedlichen Stand bzgl. der ATC-Hinweise?
- Haben die Systeme einen unterschiedlichen Stand bzgl. Z-Codes (Erweiterungen) im ATC?

7.4.3 Debugging von Prüfungen

Aufgrund der unterschiedlichen Qualität der Prüfklassen kann der Wunsch aufkommen, das Verhalten zu debuggen.

In diesem Fall können Sie die Prüfklasse ermitteln, indem Sie in der Transaktion SCI unter Code Inspector > Verwaltung von > Tests die Prüfung identifizieren.



Prüfklasse	Info	Beschreib.
/IWBEP/CL_CHK_CI_CAT_GW		GW: Gateway-Prüfungen
/IWBEP/CL_CHK_CI_DPC_CRP		GW: Finden Sie Services, die Cached Request Processing (CRP) unterstützen
CL_ACI_CI_TEST_VS_SYNTAX_CHECK		ACI-Test: ACI Ergebnissen der SYNTAXPRÜFUNG vergleichen
CL_CHK_ENH_CONFLICTS		Tests zu ENHANCEMENT-SECTION
CL_CI_CATEGORY_ABAP_COMPILER		Syntaxprüfung/Generierung
CL_CI_CATEGORY_APPL		Anwendungsprüfungen
CL_CI_CATEGORY_CONVENTIONS		Programmierkonventionen
CL_CI_CATEGORY_DYNAMIC_TESTS		Dynamische Tests
CL_CI_CATEGORY_DYNPRO		Dynpro-Prüfungen
CL_CI_CATEGORY_GENERAL		Allgemeine Fehler
CL_CI_CATEGORY_INTERNAL		Interne Tests
CL_CI_CATEGORY_INTERNAL_PERF		Interne Performance Tests
CL_CI_CATEGORY_PERFORMANCE		Performance-Prüfungen
CL_CI_CATEGORY_PROXY		Proxy-Prüfungen
CL_CI_CATEGORY_ROBUSTNESS		Robuste Programmierung
CL_CI_CATEGORY_SEARCH		Suchfunktionen

Abbildung 33 Prüfklassen identifizieren

Anschließend setzen Sie einen externen Breakpoint in der Methode RUN der zugehörigen Klasse. Um direkt eine bestimmte Prüfung zu debuggen, können Sie in den meisten Fällen im Konstruktor der Klasse den Zusammenhang zwischen der Prüfmeldung und der „Kennung für Meldungs-Code“ (scimessages-code) ermitteln.

```
92  *-------------------------Call of a System Function: &1
93    FILL_MESSAGE '0001' 'W' text-101 '"#EC CI_CCALL'.
94
95  *-------------------------Call of Transaction &1
96    FILL_MESSAGE '0002' 'N' text-102 '"#EC CI_CALLTA'.
97
98  *-------------------------Use of a SYSTEM-CALL
99    FILL_MESSAGE '0003' 'W' text-103 '"#EC CI_SYSTEMCALL'.
100
101 *-------------------------Call Editor
102   FILL_MESSAGE '0004' 'N' text-104 '"#EC CI_EDITORCALL'.
103
```

Abbildung 34 Meldungscode ermitteln

Über Volltextsuche nach diesem Code finden Sie die zugehörige Methode (und evtl. lokale Klasse) in der Prüfklasse.

8 Einbindung von weiteren Tools

8.1 SQL-Monitor – SQLM

Der SQL-Monitor ist ein Tool von SAP, mit dem die Datenbankzugriffe über einen zu definierenden Zeitraum analysiert werden können. Er steht ab SAP ECC 6.0 EHP7 zur Verfügung. Das Logging der Datenbankzugriffe geschieht üblicherweise im Produktivsystem, und der Zeitraum sollte im Bereich von einer Woche bis maximal vier Wochen liegen. Wie bei jedem Analyse-Tool werden auch hier zusätzlich Ressourcen verbraucht, und Sie sollten den Einsatz vorher mit der SAP-Basis in Ihrem Unternehmen besprechen und eventuell überwachen lassen.

Unter diesem Link finden Sie grundsätzliche Informationen zum SQL-Monitor und auch die Voraussetzungen und Installationsanleitungen:

https://www.sap.com/documents/2013/10/92b57ae6-527c-0010-82c7-eda71af511fa.html (abgerufen am 05.12.2019)

Die Ergebnisse des SQLM können in der Transaktion SWLT, zusammen mit den Ergebnissen der ATC-Hintergrund-Jobs auf den Entwicklungssystemen, dediziert analysiert werden. Man kann so die Laufzeitinformationen des SQLM zusammen mit den Informationen aus der statischen Prüfung des ATC analysieren und so Performance-kritische Stellen identifizieren.

Außerdem kann aus Sicht des ATC, wenn dort Befunde im Umfeld der Performance vorliegen, im SQLM geprüft werden, ob diese überhaupt relevant sind und falls ja, mit welcher Priorität eine Performance-Optimierung erfolgen soll. Wenn z. B. die Zugriffe im entsprechenden Development-Objekt äußerst selten im Produktivsystem durchgeführt werden, muss man diesen Performance-Befund gar nicht oder jedenfalls nicht mit hoher Priorität bearbeiten.

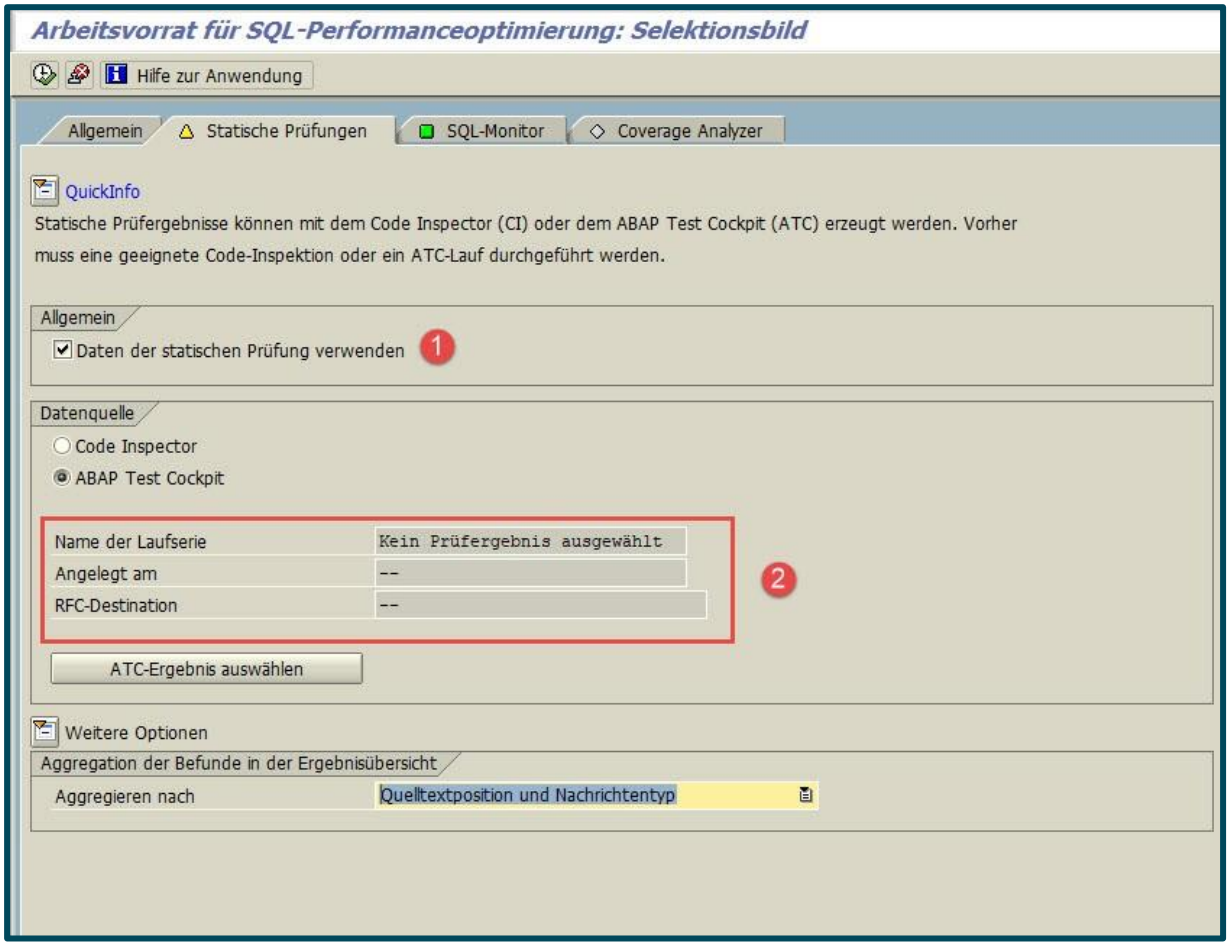


Abbildung 35 Transaktion SWLT (Analyse-Tool für den SQLM)

1. Hier auswählen, dass man statische Prüfungen mit analysieren will
2. Hier den ATC-Lauf angeben, bei dem z. B. Performance-Checks gemacht wurden

Im Zusammenspiel von ATC und SQLM kann man sehr effektiv Performance-Optimierungen erzielen. In der Praxis hat sich immer wieder gezeigt, dass solche Analysen als iterativer Prozess aufgesetzt werden müssen, also zyklisch durchgeführt werden sollten. Idealer Zeitpunkt sind z. B. größere Releases oder Monats-Releases. Wenn man im Unternehmen kürzere Deployment-Zyklen hat, sollte man diese SQLM-/ ATC-Prüfungen zumindest einmal pro Quartal fahren.

8.2 ABAP Call Monitor – SCMON oder /SDF/SCMON

Der ABAP Call Monitor dient der Aufrufprotokollierung von ABAP-Entwicklungsobjekten, z. B. von Funktionsbausteinen, Reports oder Methoden. Neben der Anzahl der Aufrufe wird – als Verbesserung zum Vorgänger UPL (Usage Procedure Logging) – auch der Kontext der Nutzung protokolliert, ebenso die Zeitscheibe, in welcher der Aufruf stattgefunden hat.

SCMON steht in Systemen mit einem ABAP-Stack ab 7.50 zur Verfügung und kann in ältere Releases (> 7.00) per ST-PI-Add-on eingespielt werden (SAP-Hinweis 2224462).

Folgender Blog-Eintrag beschreibt die Systemvoraussetzungen sowie die Funktionsweise genauer:

https://blogs.sap.com/2017/04/06/abap-call-monitor-scmon-analyze-usage-of-your-code/ (abgerufen am 01.10.2019)

Die so gewonnenen Statistiken können genutzt werden, um einerseits besonders wichtige, andererseits nicht mehr benutzte Eigenentwicklungen im System zu identifizieren und so eine Priorisierung für die Abarbeitung von ATC-Befunden zu definieren. Aufgrund der hohen Granularität und des daraus resultierenden Datenvolumens ist ein langfristiges Halten der nicht aggregierten Daten normalerweise nicht möglich.

Im Solution-Manager-System oder mit der im Folgenden beschriebenen Transaktion SUSG können die Daten jedoch aggregiert und langfristig vorgehalten werden.

8.3 Nutzungsdatenaggregation – SUSG

Um eine belastbare Aussage über die Nutzung eines Entwicklungsobjektes treffen zu können, ist ein Überwachungszeitraum von mindestens 13 Monaten notwendig. Wie oben ausgeführt, ist dies mit den vom SCMON erstellten Tages-Zeitscheiben nicht möglich. Die Transaktion SUSG wurde geschaffen, um die SCMON-Daten zu aggregieren und in eigenen Tabellen zu persistieren. Sie ist ab Release 7.40 SP14 verfügbar, kann aber in ältere Releases (>= 7.00) eingespielt werden (SAP-Hinweis 2643357). Unabhängig davon ist auch eine Aggregation im Solution Manager möglich.

Folgende Blog-Einträge beschreiben die Systemvoraussetzungen, die Installation sowie die Funktionsweise genauer:

https://blogs.sap.com/2019/02/27/aggregate-usage-data-in-your-production-system-with-susg-transaction/ (abgerufen am 01.10.2019)

https://blogs.sap.com/2019/02/25/susg-how-to-aggregate-and-analyze-the-results-of-scmon-abap-call-statistics-to-do-s4hana-custom-code-housekeeping/ (abgerufen am 30.08.2019)

Im Gegensatz zum UPL, das für die Datensammlung und für die Aufbereitung der Daten den Solution Manager (und das integrierte BI) voraussetzt, ist die Transaktion SUSG unabhängig von weiteren Systemen und weniger komplex. Die aggregierte Datenhaltung der SUSG spart Platz gegenüber der UPL-Lösung und ist für die Bewertungen im Rahmen von z. B. S/4HANA-Konversionsprojekten völlig ausreichend.

8.4 Custom Code Lifecycle Management – CCLM

Das Custom Code Lifecycle Management ist die Zusammenführung der o. g. Tools im Solution Manager ab Rel. 7.2 SP05. Mit unterschiedlichen zusätzlichen Aspekten und Dashboards erhält man so die Möglichkeit, die kundeneigenen Entwicklungen dauerhaft zu überwachen und somit den gesamten Lifecycle zu analysieren, um frühzeitig zu erkennen, welches Coding nicht mehr genutzt wird. Auch die Software-Qualität kann im CCLM aus den Ergebnissen des ATC überwacht werden.

Die Aufbereitung kann sowohl für das Qualitätsmanagement als auch für das übergreifende Management zur Verfügung gestellt werden.

https://blogs.sap.com/2019/05/03/cclmcustom-code-life-cycle-management-configuration-in-solution-manager-7.2/ (abgerufen am 01.10.2019)

https://wiki.scn.sap.com/wiki/display/SM/SAP+Solution+Manager+WIKI+-+Custom+Code+Management (abgerufen am 01.10.2019)

8.5 SAP-Fiori-Custom-Code-Migration-App (ab S/4HANA 1809)

Mit dem Einsatz eines S/4HANA-1809-Systems als zentralem ATC steht die Fiori-App Code Migration zu Verfügung. Mit dem Einsatz dieser App können die Ergebnisse der ATC-Prüfungen mit den gesammelten Usage-Daten aus den Live-Systemen oder dem Solman abgemischt und so eine Bewertung der Code-Anpassungen für S/4HANA vorgenommen werden.

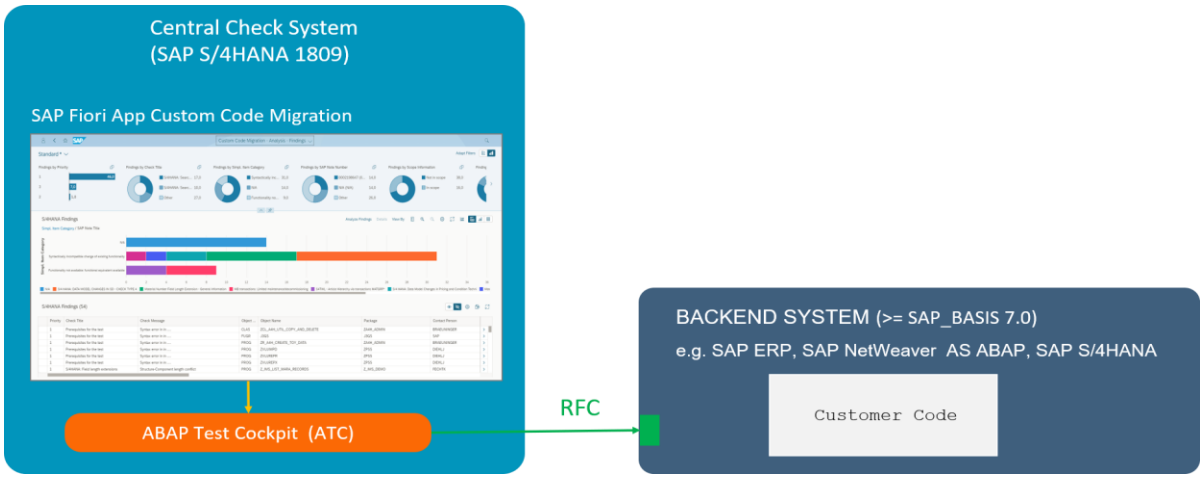


Abbildung 36 S/4HANA 1809 als Remote-ATC (Quelle: SAP SE)

Die von SAP bereitgestellte Simplification Database hilft Ihnen dabei, für S/4HANA notwendige Anpassungen frühzeitig zu identifizieren. Dafür laden Sie die Simplification Database (SAP-Hinweis 2241080) herunter und spielen sie mit der Transaktion SYCM in das zentrale ATC-System ein. Je nach Release-Stand ist es meist notwendig, in den beteiligten Systemen noch entsprechende SAP-Hinweise einzuspielen (vgl. SAP-Hinweis 2436688).

Danach können Sie mit der Prüfvariante S4HANA_READINESS_REMOTE die angeschlossenen Systeme überprüfen.

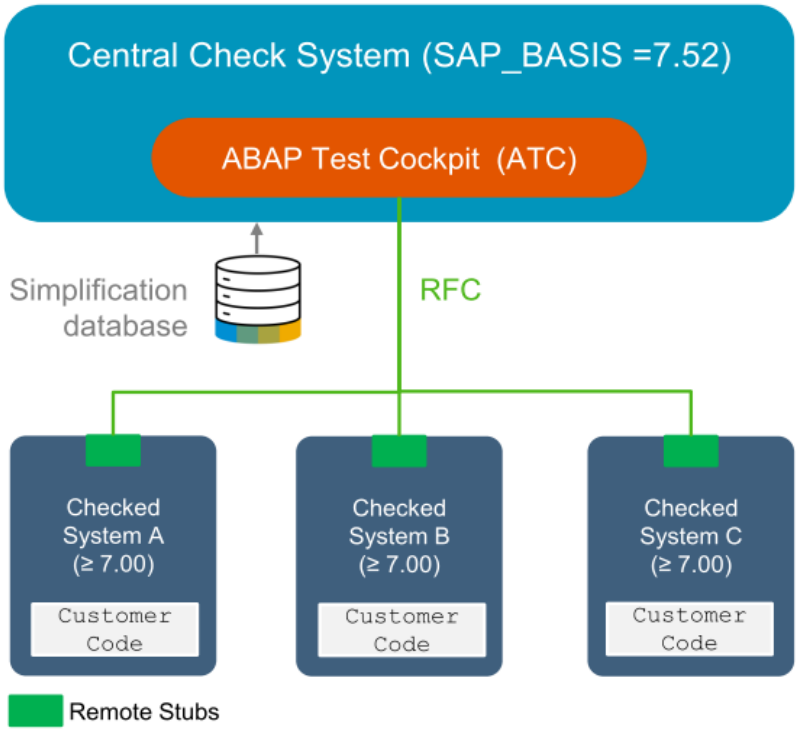


Abbildung 37 Remote-ATC-Prüfungen mit der S/4HANA Simplification Database (Quelle: SAP SE)

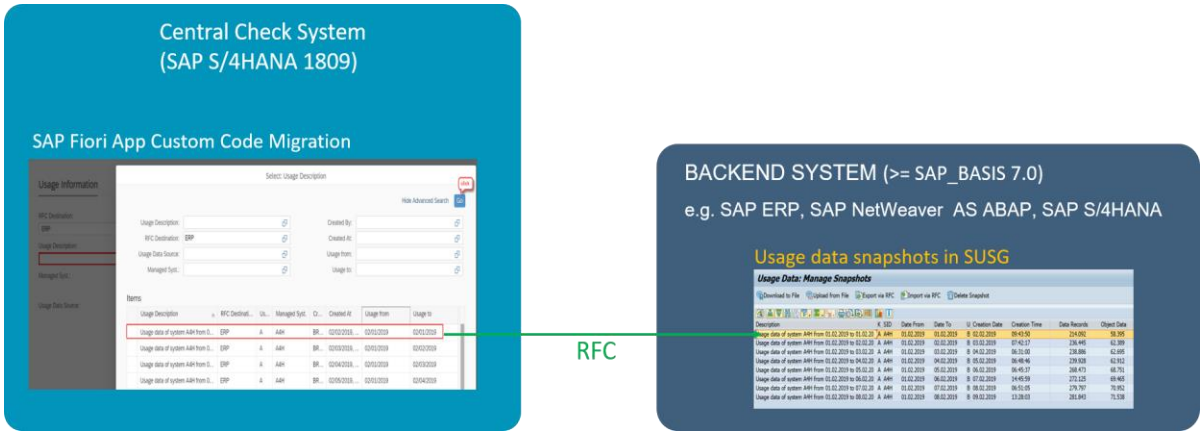


Abbildung 38 Integration der Usage-Daten (Quelle: SAP SE)

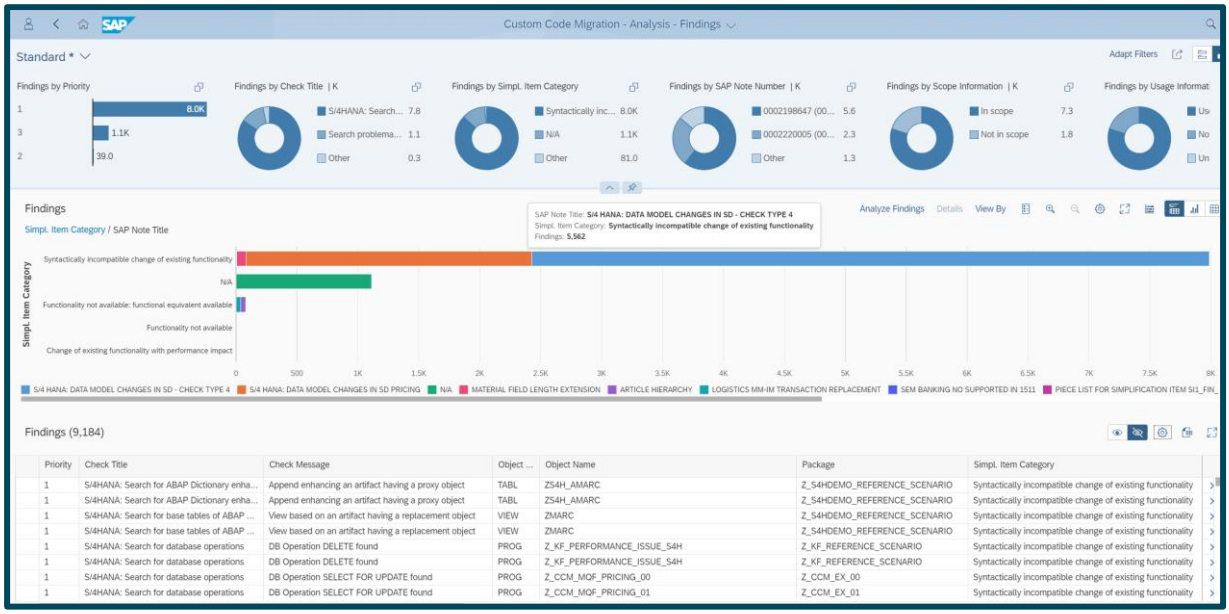


Abbildung 39 Analyse der ATC-Befunde (Quelle: SAP SE)

Anhand der eingespielten Usage-Daten erstellt die App erstmalig einen Vorschlag, welche Objekte sich im Scope für die S/4HANA-Migration befinden. Der Scope lässt sich aber jederzeit ändern.

Die Fundstellen können Sie z. B. nach Paketen, SAP-Hinweisen, Priorität oder Verfügbarkeit von Quick-Fixes gruppieren und so den Objektumfang eingrenzen.

Zusätzlich wird ein Transport angelegt, der alle zu löschenden Objekte (nicht im Scope) aufnimmt. Bei der Systemumstellung auf S/4HANA werden dann durch den SUM (Software Update Manager) die nicht mehr verwendeten Objekte automatisch gelöscht.

Hinweis: Wenn Sie die zu löschenden Objekte als Back-up noch benötigen, empfehlen wir Ihnen den Einsatz von abapGit (https://github.com/larshp/abapGit), um die Objekte in einem git-Repository abzulegen.

Blog-Eintrag zur Fiori-App:

https://blogs.sap.com/2019/02/27/custom-code-analysis-for-sap-s4hana-with-sap-fiori-app-custom-code-migration/ (abgerufen am 01.10.2019)

Dokumentationslink:

https://help.sap.com/viewer/9a281eac983f4f688d0deedc96b3c61c/201809.000/en-US/f3b749efe0674d669fb1ecf9704fcfd3.html (abgerufen am 16.08.2019)

8.6 Eclipse-ADT-Quick-Fixes

Ab S/4HANA 1809 und ADT in Eclipse 3.0 stellt SAP für die Custom-Code-Adaptation sogenannte Quick-Fixes bereit, um betroffene Code-Stellen automatisch anzupassen zu können.

Diese Quick-Fixes sind in erster Linie technische Anpassungen, die durch die Simplification im S/4HANA-System sowie durch die SQL-Anpassungen für HANA bedingt sind. Zu ersterem gehören u.a. DB-Zugriffsmethoden in den neuen Datenmodellen im SD, Preisfindung und Hauptbuch sowie Anpassungen des Codes an lange Materialnummern.

Bei der Ausführung der Quick-Fixes wird ein Wizard gestartet, der einen Code-Vergleich anbietet. Dort kann man auch sehen, wie im Einzelnen der Quell-Code angepasst wird.

Viele Quick-Fixes sind auch massentauglich, d. h. die gleichzeitige Anwendung auf mehrere Fundstellen ist möglich.

https://blogs.sap.com/2019/06/25/custom-code-adaptation-for-sap-s4hana-faq/#_Toc470164440 (abgerufen am 16.08.2019)

https://blogs.sap.com/2018/10/02/semi-automatic-custom-code-adaptation-after-sap-s4hana-system-conversion/
(abgerufen am 16.08.2019)

8.7 Ausblick: ABAP und Continuous-Integration

Mit dem Einsatz von abapGit können viele neue Möglichkeiten in der Entwicklung zum Einsatz kommen. Unter anderem ist dabei auch Continuous-Integration (https://de.wikipedia.org/wiki/Kontinuierliche_Integration) ein wesentlicher Aspekt und bedeutet, dass schon während der Entwicklung laufend ein Gesamtsystem gebaut und automatisiert getestet wird.

Das kann z. B. für die Entwicklung von REST-APIs interessant sein, damit Code-Änderungen im Backend sofort zu einem integrativen Test der Applikation führen und die Ergebnisse dieser Prüfungen direkt in der Entwicklungsumgebung visualisiert werden können. Sinnvolle Tests im Rahmen der CI sind u. a. die Ausführung von ABAP Unit Tests und die Prüfungen durch den ATC.

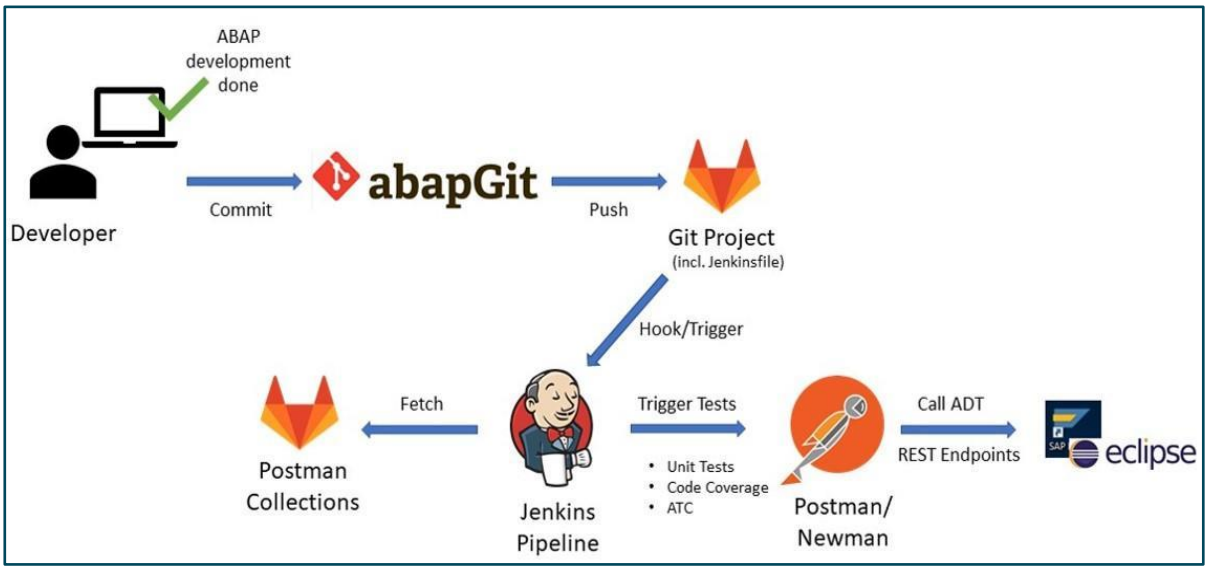


Abbildung 40 Beispiel für einen CI-Workflow mit abapGit (Quelle: Christian Lechner)

https://github.com/lechnerc77/ABAP_CI_PIPELINE_BASE (abgerufen am 05.12.2019)

https://github.com/andau/abapCI (abgerufen am 05.12.2019)

https://blogs.sap.com/2018/08/20/abap-continuous-integration-plugin-now-with-plug-and-play/ (abgerufen am 16.08.2019)

8.1 Relevante SAP-Hinweise

Bezeichnung	Hyperlink
SQLM	https://launchpad.support.sap.com/#/notes/1885926
ABAP-Aufrufmonitor SCMON (/SDF/SCMON)	https://launchpad.support.sap.com/#/notes/2679723 https://launchpad.support.sap.com/#/notes/2224462
Implementieren der Transaktion SUSG	https://launchpad.support.sap.com/#/notes/2643357 https://launchpad.support.sap.com/#/notes/2701371
Fiori-Custom-Code-Migration-App	https://launchpad.support.sap.com/#/notes/2599695
Custom-Code-Checks für SAP-Cloud-Platform-ABAP-Environment	https://launchpad.support.sap.com/#/notes/2684665
S/4HANA: Sammelhinweis für ATC-Prüfungen und Fiori-App Code Migration	https://launchpad.support.sap.com/#/notes/2436688
S/4HANA: Simplification-Database	https://launchpad.support.sap.com/#/notes/2241080
S/4HANA: Empfehlungen zur Anpassung von Custom-Code	https://launchpad.support.sap.com/#/notes/2190420
Aktivieren der Funktion für mehrere Quick-Fixes für ATC-Probleme in ADT	https://launchpad.support.sap.com/#/notes/2695592

9 Die Autoren

Manuela Graf

SAP Technical Expert for Development & Configuration, Infineon IT Services GmbH

Manuela Graf startete 2007 als ABAP-Entwicklerin im HR-Umfeld. Sie ist seit 2013 in einem zentralen Team bei Infineon beschäftigt, das sich mit Qualität und Governance im SAP-Umfeld auseinandersetzt. In dieser Funktion ist sie für die konzernweiten SAP-Entwicklungsrichtlinien, das Aufsetzen und Erweitern des ATC sowie das Prüfen der Entwicklungs- und Design-Qualität verantwortlich.

Frank Hampel

SAP inhouse Consultant SD & ABAP/OO Developer, KTR Systems GmbH

Frank Hampel arbeitet seit 2001 als SAP inhouse Consultant SD und ABAP/OO Developer, seit 2014 beim mittelständischen Maschinenbauer KTR Systems GmbH in der internen IT-Abteilung. Neben Prozessberatung und SAP-Rollout-Projekten für Tochterfirmen führt er Software-Entwicklungsprojekte im SAP R/3 mit Schwerpunkt im Modul SD durch. Besonders spezialisiert hat er sich auf die Entwicklung von SAP Interactive Forms by Adobe (IFbA). Standardisierung und gute Code-Qualität gehören zu seiner Grundeinstellung. Dabei sind Design-Patterns, Frameworks und unternehmens-weite Toolbox-Klassen gute Hilfsmittel. Seit 2016 ist er Teil des KTR ATC-Teams.

Florian Henninger

Senior Consultant SAP Development, FIS Informationssysteme und Consulting GmbH

Florian Henninger ist zertifizierter ABAP-Entwickler und seit 2010 im Bereich SAP-ABAP-Entwicklung tätig. Sein fachlicher Schwerpunkt liegt im Bereich Core Development/Enhancements, API-Bereitstellungen sowie dem Thema Output Management bei ERP-Einführungsprojekten und Bestandssystemen. Außerdem ist er Mitglied des Qualitätmanagements, bei der er unter anderem die Entwicklungsrichtlinien ständig weiterentwickelt.

Oliver Hütköper

SAP Technology and Development Expert, Hauni Maschinenbau GmbH

Oliver Hütköper ist seit 2010 im Umfeld der SAP-Entwicklung tätig. Neben der Erstellung von kundeneigenen Entwicklungen stehen dabei insbesondere die technische Beratung in Projekten, die Steuerung von internen und externen Entwicklern sowie die Qualitätssicherung im Fokus. Er ist bei der Hauni Maschinenbau GmbH verantwortlich für das ABAP Test Cockpit.

Mario Kopp

Senior Software Developer SAP, Phoenix Contact GmbH & Co. KG

Mario Kopp ist seit 2004 im Umfeld des SAP Development bei Phoenix Contact beschäftigt und seit 1985 im Unternehmen. Der Tätigkeitsbereich umfasst die Anwendungs- und Interface-Entwicklung in ABAP. Zusätzlich ist er verantwortlich für die Software-Qualität im Zusammenspiel mit dem zentralen ATC und das Aufsetzen von Prüfvarianten und eigenen ATC-Checks. Des Weiteren umfasst sein Arbeitsbereich auch die Performance-Analysen von ABAP Code in den jeweiligen Produktivsystemen. Die ABAP-Entwicklungsrichtlinie wird von ihm und zwei anderen Kollegen ständig weiterentwickelt.

Dr. Christian Lechner

Principal IT Consultant & Innovation Management, minnosphere GmbH

Christian Lechner arbeitet seit 2005 als Entwickler, Architekt und Projektleiter in der SAP-Standard- als auch Kunden-individuellen Entwicklung. Mit der Nutzung des ATC kam er daher schon in vielen Bereichen in Berührung. Aktuell liegt sein Fokus im Kontext ATC bei der S/4HANA-Readiness von Kundensystemen.

Michael Lichtinger

SAP Technical Consultant, Clientis AG

Michael Lichtinger entwickelt seit 2006 in diversen Kundenprojekten in ABAP und SAPUI5 und konnte dort Erfahrungen sammeln. Neben der Einführung des ATC als Quality-Gate bei der Clientis AG hat er dieses auch bei einigen Kunden mit eingeführt.

Stefan Nothaft

Lead IT Consultant Products & Development, msg systems ag

Stefan Nothaft arbeitet seit 2004 bei dem SAP-Beratungs- und -Entwicklungspartner msg systems ag und entwickelt dort SAP-Anwendungen für individuelle Kundenprojekte. Darüber hinaus ist er an der Entwicklung von Produktlösungen im Bereich Insurance/Reinsurance und Food beteiligt. Bei seiner täglichen Arbeit als Software-Architekt in R/3 und S/4 legt er besonders viel Wert auf erstklassige Code-Qualität und setzt auf intelligente (Test-) Automatisierung. Im „Arbeitskreis Development" der DSAG teilt er sein Wissen als erfahrener Entwickler in der ABAP-Programmierung.

Sebastian Oswald

Junior Professional SAP Solution Development, Schwarz IT KG

Sebastian Oswald arbeitet seit 2018 bei der Schwarz IT KG als ABAP-Entwickler. Zu seinen Aufgaben gehören sowohl die Weiterentwicklung von kundeneigenen Lösungen, als auch die Einführung des remote ATC bei der Schwarz IT. Die stetige Erweiterung von Fachwissen und die Einhaltung und Erhöhung von Softwarequalität ist für ihn von besonderer Bedeutung.

Volker Pröbius

SAP Technical Development Manager, Robert Bosch GmbH

Volker Pröbius ist seit 1998 im SAP-Umfeld tätig und seit 2006 im Robert Bosch Konzern in der zentralen IT. Der Tätigkeitsbereich umfasst die Anwendungs- und Schnittstellenentwicklung in ABAP und UI5. Seit 2018 verantwortet er als Technical Development Manager die Entwicklungsrichtlinien und Software-Qualität einiger S/4HANA-Systeme.

Norbert Schulte

SAP inhouse Consultant BI/SAP Basis & ABAP/OO Developer, KTR Systems GmbH

Norbert Schulte ist seit 2006 im SAP-Umfeld tätig, arbeitet seit 2008 bei der KTR Systems und war maßgeblich beim Aufbau einer SAP-BI-Umgebung, von der Modellierung der Datenmodelle über die Berichterstellung und sämtliche Basisaktivitäten, beteiligt. Aktuell setzt er unternehmensweit ein Berechtigungskonzept, angelehnt an die DSGVO, um. 2016 initiierte er das Thema ATC bei KTR und ist Teil des dreiköpfigen ATC-Teams.

Susanne Schuster

SAP Technical Expert & Quality Gate, Infineon IT Services GmbH

Susanne Schuster ist seit 2013 im SAP-Bereich bei Infineon IT Services GmbH beschäftigt. Zu ihren Hauptaufgabengebieten zählen die Qualitätssicherung der Kundenentwicklungen auf den SAP-Systemen sowie diverse Prozesse im Bereich Design, Governance und Guidelines. Im Zuge der Qualitätssicherung wurde ATC für alle SAP-Systeme eingeführt. Außerdem befasst sie sich mit der Entwicklung kundeneigener Prüfungen, um interne Richtlinien und Best-Practices automatisiert zu kontrollieren.

Kay Streubel

Senior Consultant SAP Development, Clientis AG

Kay Streubel ist seit 2001 zertifizierter ABAP-Entwickler und hat in vielfältigen Kundenprojekten mit dem Schwerpunkt ABAP Erfahrungen gesammelt. Die stetige Weiterentwicklung des Entwicklungs-Know-hows und die Steigerung der Software-Qualität im Team liegen ihm besonders am Herzen. Als Mitglied des DSAG-Arbeitskreises Development beteiligt er sich aktiv an den Interessen der ABAP-Community.

Tim Trumpf

Software Developer SAP, Hansgrohe SE

Tim Trumpf arbeitet seit 2018 im SAP-Umfeld bei der Hansgrohe SE und entwickelt ABAP und UI5. Er konnte in kundeneigenen Entwicklungen Erfahrungen sammeln und führte ein Remote-ATC-System ein, welches er betreut.

Edo von Glan

SAP-Entwickler, Drägerwerk AG & Co. KGaA

Edo von Glan hat als Software-Entwickler für IBM und Commerzbank sowie sieben Jahre in der Produktentwicklung bei SAP gearbeitet.

Seit 2008 arbeitet er in der SAP-Entwicklungsabteilung bei Dräger, davon sechs Jahre in leitender Position. Sein Verantwortungsbereich umfasst die Applikations- und Schnittstellenentwicklung sowie Custom Code Lifecycle Management und Software-Qualität für die im Unternehmen zentral verwalteten SAP-Systeme.

Bärbel Winkler

Systemanalystin SAP-Basis/Programmierung, Alfred Kärcher SE & Co. KG

Bärbel Winkler arbeitet seit 2000 im SAP-Umfeld in der ABAP-Programmierung und Projektarbeit und ist seit 2011 bei der Alfred Kärcher SE & Co. KG in kundeneigene

Entwicklungen und deren Koordination eingebunden. Zu ihren Aufgaben gehört die regelmäßige Aktualisierung der Entwicklungsrichtlinien, die Verwaltung des zentralen ATC-Systems, die Beurteilung und Schätzung eingehender Entwicklungs-anforderungen sowie die Begleitung der Umsetzung bei extern durchgeführter Programmierung.

Anhang A: GitHub-Repository der Code-Beispiele

Nach dem Motto „Ein Bild sagt mehr als tausend Worte“ stellen wir Ihnen im Rahmen des Leitfadens auch konkrete Code-Beispiele im abapGit-Format zur Verfügung. Die Beispiele finden Sie im GitHub-Repository der DSAG unter: https://github.com/1DSAG/ATC-Best-Practice-Guide (abgerufen am 05.12.2019)

Der dort abgelegte Code unterliegt der MIT-Lizenz. Beachten Sie bei der Verwendung des Code die in der Lizenz beschriebenen Anmerkungen bzgl. Wartung etc.: https://github.com/1DSAG/ATC-Best-Practice-Guide/blob/master/LICENSE (abgerufen am 05.12.2019)

Ihr Feed-back ist natürlich willkommen. Sie können Rückmeldungen zum Projekt wie Bugs oder Feature-Requests über den üblichen GitHub-Mechanismus von Issues abgeben. Da es sich um ein Community-Projekt aus der DSAG heraus handelt, können Sie natürlich auch selbst in Form von Pull-Requests zum Projekt beitragen.

Anhang B: abapOpenChecks

Bei den abapOpenChecks handelt es sich um ein Open-Source-Projekt, das SAP Code Inspector/ATC-Prüfungen unter der MIT-Lizenz zur Verfügung stellt. Sie finden das Projekt auf GitHub unter https://github.com/larshp/abapOpenChecks (abgerufen am 29.10.2019).

Sie können das Tool mittels abapGit (https://github.com/larshp/abapGit) auf Ihrem SAP-System installieren. Die Installation ist auf der Seite https://docs.abapopenchecks.org/installation/ (abgerufen am 29.10.2019) beschrieben.

Eine Dokumentation der verfügbaren Prüfungen finden Sie auf der Seite https://docs.abapopenchecks.org/checks/ (abgerufen am 29.10.2019). Hier können Sie auch erkennen, ob die Prüfung RFC-fähig ist oder nicht. In den Prüfungen finden Sie – sofern relevant – auch einen Verweis auf das Community-Projekt Clean ABAP (https://github.com/SAP/styleguides/blob/master/clean-abap/CleanABAP.md, abgerufen am 29.10.2019), das weitere Informationen zur Motivation der Prüfung liefert.

Als Community-Projekt liefert die SAP natürlich keinerlei Support zu den abapOpenChecks. Die Installation und Nutzung der Prüfungen obliegen daher komplett der Hoheit der Nutzer des Codes. Rückmeldungen zum Projekt wie Bugs oder Feature-Requests können Sie über den üblichen GitHub-Mechanismus von Issues abgeben. Sie können sich natürlich auch mittels Pull-Requests am Projekt beteiligen.

Anhang C: Code Pal for ABAP

Beim Code Pal for ABAP handelt es sich um ein Open-Source-Projekt, das SAP Code Inspector/ATC-Prüfungen unter der Apache Lizenz, Version 2.0 zur Verfügung stellt. Sie finden das Projekt auf GitHub unter https://github.com/SAP/code-pal-for-abap (abgerufen am 11.06.2020). Ziel der Prüfungen ist es den Entwickler beim Erstellen von Clean ABAP (siehe https://github.com/SAP/styleguides/blob/master/clean-abap/CleanABAP.md - aufgerufen am 11.06.2020) zu unterstützen.

Sie können das Tool mittels abapGit (https://github.com/larshp/abapGit) auf Ihrem SAP-System installieren. Die Installation ist auf der GitHub Seite des Projekts beschrieben (https://github.com/SAP/code-pal-for-abap#how-to-bring-code-pal-into-your-system – aufgerufen am 11.06.2020).

Eine Dokumentation der verfügbaren Prüfungen finden Sie auf der Seite https://github.com/SAP/code-pal-for-abap/blob/master/docs/check_documentation.md (aufgerufen am 11.06.2020).

Als Community-Projekt liefert die SAP natürlich keinerlei Support zum Code Pal for ABAP. Die Installation und Nutzung der Prüfungen obliegen daher komplett der Hoheit der Nutzer des Codes. Rückmeldungen zum Projekt wie Bugs oder Feature-Requests können Sie über den üblichen GitHub-Mechanismus von Issues abgeben. Sie können sich natürlich auch mittels Pull-Requests am Projekt beteiligen.

Impressum

Wir weisen ausdrücklich darauf hin, dass das vorliegende Dokument nicht jeglichen Regelungsbedarf sämtlicher DSAG-Mitglieder in allen Geschäftsszenarien antizipieren und abdecken kann. Insofern müssen die angesprochenen Themen und Anregungen naturgemäß unvollständig bleiben. Die DSAG und die beteiligten Autoren können bezüglich der Vollständigkeit und Erfolgsgeeignetheit der Anregungen keine Verantwortung übernehmen.

Deutschsprachige SAP® Anwendergruppe e. V.
Altrottstraße 34 a
69190 Walldorf | Deutschland
Telefon +49 6227 35809-58
Telefax +49 6227 35809-59
E-Mail info@dsag.de

dsag.de