Best Practices Document

Transition to SAP S/4HANA

Document Version: 1.0 – 2020-05-25

PUBLIC

Custom Code Management during an SAP S/4HANA Conversion

Document History

Version	Date	Change
1.0	2020-05-14	Initial version

Content

1 Custom Code Management as part of an SAP S/4HANA Conversion

Note on target audience

This document is available to all customers, however some of the content we recommend is accessible to Enterprise Support customers only. The standing guidance from SAP is that any customers who are embarking on major change – and the move to SAP S/4HANA definitely qualifies as that – should minimally have Enterprise Support.

Why touch custom code at all during your SAP S/4HANA conversion?

Moving from SAP ERP to SAP S/4HANA has a major impact on your custom code. This is due to the changes in the data structure (simplifications) as well as the underlying changes to the database as SAP S/4HANA is a whole new product rather than a new version of SAP ERP.

Essentially what you will do is

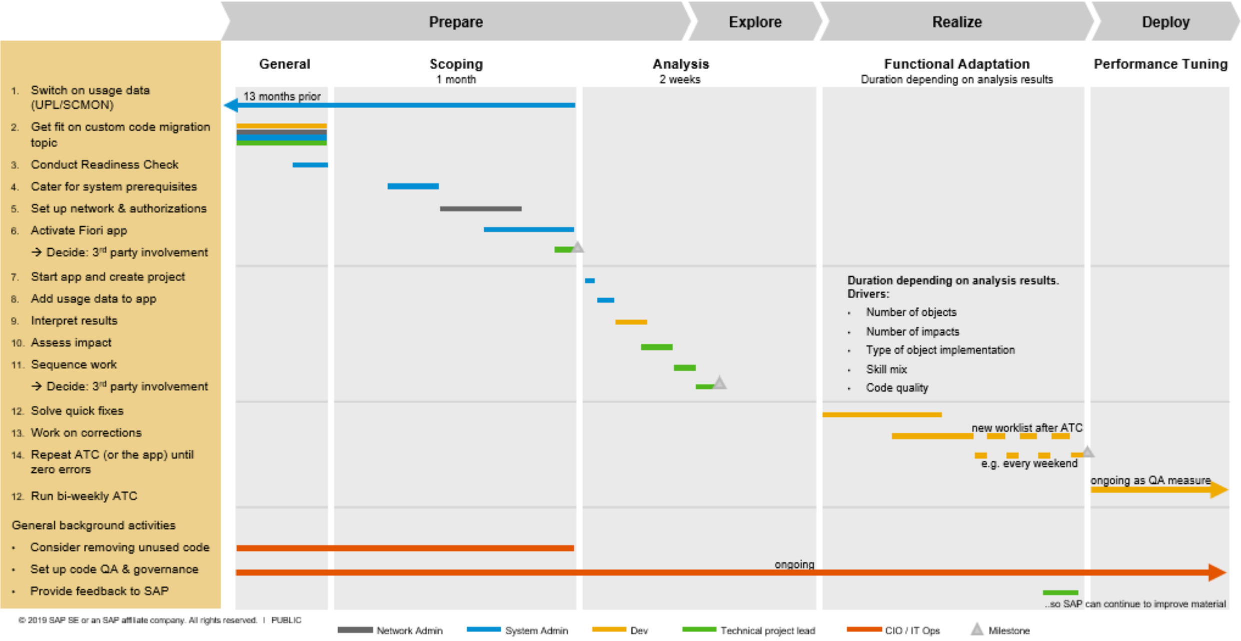
- scope an analysis of your custom code (including modifications and enhancements),
- conduct it, and
- take action according to the analysis results.

Prior to all that though, there is some

- preparation work to be done

to get everyone on top of the topic before the actual project work starts. And finally, you'll also have to plan how you want to

- keep your custom code healthy once you have arrived in SAP S/4HANA.



1. Switch on usage data (UPL/SCMON)
2. Get fit on custom code migration topic
3. Conduct Readiness Check
4. Cater for system prerequisites
5. Set up network & authorizations
6. Activate Fiori app

→ Decide: 3rd party involvement

7. Start app and create project
8. Add usage data to app
9. Interpret results
10. Assess impact
11. Sequence work

→ Decide: 3rd party involvement

12. Solve quick fixes
13. Work on corrections
14. Repeat ATC (or the app) until zero errors

12. Run bi-weekly ATC

General background activities

- Consider removing unused code
- Set up code QA & governance
- Provide feedback to SAP

Duration depending on analysis results. Drivers:

- Number of objects
- Number of impacts
- Type of object implementation
- Skill mix
- Code quality

© 2019 SAP SE or an SAP affiliate company. All rights reserved. I PUBLIC

Project plan – custom code activities in sequence

For customers with just one ERP in place (in a 3-tier landscape), heading for a conversion, here's SAP's recommendation how to best tackle these steps. We've done all the reading for you and sifted through the material such that we can point you to specific sections of the documents that are out there.

Browser recommendation: Google Chrome

2 Preparation work: Knowing your custom code, and knowing about custom code migration

In order to scope your custom code analysis properly, it's good to know your data, and it's even better to know everyone in your project team has been trained on the custom code migration topic.

As for knowing your data, it's a good idea to collect usage data from your system for ideally 13 months before your migration. 13 months so you also have the data for reports running merely annually. You're less than 13 months before project start? Well, switch it on nonetheless to at least find out about the reports used weekly and quarterly. Your system admin can do this by switching on "Usage Logging" (SCMON) and aggregation of collected usage data (SUSG) in order to store it for a long time in the system. These blogs tell you how:

Blog: ABAP Call Monitor – Analyze usage of your code

Blog: Aggregate usage data in your production system with SUSG transaction

The two abovementioned blogs are intended for customers with no SAP Solution Manager in place.

If you happen to have an SAP Solution Manager though, consider enabling "Usage Logging" (SCMON) within the Custom Code Lifecycle Management (CCLM), including the aggregation of the collected data. More details here:

How-to Guide: Custom Code Lifecycle Management 72 SP06 (pages 17-20. When also looking for CCLM setup, start on page 9.)

Apart from knowing usage, you also want to know which subject areas of your custom code are actually relevant for your SAP S/4HANA conversion.

This is what the SAP Readiness Check (RC) does for you in its Custom Code Analysis section. It displays which subject areas regarding your custom code have to be adjusted, and what the effect of the relevant subject on your code is.

Note, though, that the individual affected code objects are not listed in the SAP Readiness Check. These will be obtained later with another tool (SAP ABAP Test Cockpit: SAP ATC – no worries, all covered further down this document). However, to gain a deeper insight on the affected subject areas at an early stage, you can conduct an SAP Readiness Check, which works as follows:

SAP Note: 2758146 - SAP Readiness Check 2.0 & Next Generation SAP Business Scenario Recommendations or SAP Innovation and Optimization Pathfinder

As for knowing everyone is on top of the topic: There are several roles in your team who will be involved in the custom code migration topic. These are:

Main roles:

- S/4 Migration project lead
- Development
- Technical project lead
- IT Operations

Supporting roles:

- Network administrator
- System administrator

For the main roles, it makes sense to get fit on the custom code migration topic before embarking on the actual work.

We recommend the following activities:

- S/4 Migration project lead: Read overview white paper
    - White Paper: Custom Extensions in SAP S/4HANA Implementations
- Main roles: Take the following Meet-the-Expert Sessions for a topic overview (5 Sessions, duration: 1h each)
    - Custom Code for SAP S/4HANA Conversions: Methodology & Tools
    - Custom Code for SAP S/4HANA Conversions: ABAP Test Cockpit
    - Custom Code for SAP S/4HANA Conversions: SAP Fiori App
    - Custom Code for SAP S/4HANA Conversions: Result Analysis
    - Custom Code for SAP S/4HANA Conversions: Adaptation Activity
- Development / IT Operations / Technical PL: Register for the following Expert Guided Implementation (EGI) course offering:
    - EGI: Custom Code Lifecycle Management: Make Your Custom Code Efficient
    - EGI: SAP S/4HANA: Custom Code Impact Analysis
- Development / Technical PL: ABAP and SAP HANA delta changes
    - OpenSAP course: ABAP Development for SAP HANA

      Duration: 4 weeks (4 - 6 hours per week)
    - Blog: Getting Started with the ABAP RESTful Programming Model
    - Slide deck: Implementation patterns for CDS Views
    - DSAG document (German only): Einsatz des ABAP Test Cockpit

Our ambition with this document is to comprehensively describe the custom code management process before an SAP S/4HANA conversion for customers with just one ERP (in a 3-tier landscape) in place.

Still, you might want to do some further reading or look for a community to ask questions in. Our recommendations are:

- SAP Wiki: SAP Fiori SAP S/4HANA Extensibility topics

  a good blog to start with in this section is

  SAP Fiori for SAP S/4HANA – How to extend a SAP Fiori app in SAP S/4HANA
- Questions during the course of the project:

  SAP community: ABAP Testing and Analysis

If you are looking for structured learning on custom code management, see the Enterprise Support Value Map on the topic.

Training completed – what's next?

A word of reassurance: We are aware that there will be a time lag between your training and the actual doing later on. That’s why further down this document, when it comes to the actual doing, we will point to the specific moments in the documents or videos that contain the actual how-to.

3 General: Paving the way for clean custom code in the future

Custom code is also a topic of interest for your CIO and IT Operations: Your S/4 conversion represents a near-perfect occasion for removing unused code. Even if this turns out to be an activity you might rule out, for example if you only have very little custom code to begin with, it still makes sense to do so based on an informed decision.

For decommissioning unused custom code, your IT Operations can refer to the following white paper:

White paper: Custom Code Decommissioning: Best Practices (in CCLM) (pages 1-9).

(If you happen to also have an SAP Solution Manager, then the rest of the document covers how to decommission code using SAP Solution Manager.)

Likewise, this is the moment to seriously start thinking about how you want to ensure an ongoing quality of your custom code in the future. Set up a team that includes both quality experts and developers. Work with it to introduce a quality process and define your own guidelines that your Development organization subscribes to.

For reference on setting up custom code guidelines and Quality Assurance, your IT Operations can get inspiration from the following programming guidelines assembled by the German SAP user group DSAG. Overall it makes sense to set up guidelines in a joint initiative between IT Operations and Development to obtain buy-in from both parties.

DSAG Recommendations: Best Practice Guidelines for Development – Useful tips for ABAP Development (especially Chapters 2, 4, 5)

4 Scoping: Setting everything up for a smooth analysis

As always with SAP, there are several options to achieve your goal. However, for conversion customers with just one ERP in place, you might use ATC on an SAP NetWeaver 7.5.2. We at SAP recommend you make use of the Custom Code Analysis Fiori App because it offers you additional features such as the custom code quick fixes that can be solved in a near-automated fashion. You also do not need to have an SAP Solution Manager in place for this.

In order to be able to use the Custom Code Analysis Fiori App, you need to have a check system in place, for which there are two on-premise options, and one cloud option:

- On-premise, you need to have an SAP S/4HANA (>=1809; we're using the Custom Code Migration Guide 1909 in this document as it is the latest available release at the point of writing) in place.
  - Either as an empty system
  - Or as an SAP S/4HANA fully activated appliance within your firewall.
- As your cloud option, you would use the SAP Cloud Platform ABAP Environment (often referred to as SAP Cloud ABAP Platform).

Either way, this system does not contain your custom code but needs to be connected to your current ERP so you can conduct your analysis. This system will be referred to as your check system.

For setting up your check system to do the custom code analysis, your system admin can refer to:

- **On-premise – empty SAP S/4HANA system:**

  Blog: SAP S/4HANA System Conversion - Custom code adoption process

  (specifically 3rd paragraph "Technical Infrastructure for custom code analysis". This also refers to the following blog.)

  Blog: Remote Code Analysis in ATC – Technical Setup step by step (including how to set up system)

  Custom Code Migration Guide (specifically Chapter 2.1 Using SAP S/4HANA 1909 as Central Check System)

- **On-premise – SAP S/4HANA fully activated appliance:**

  SAP Note: 2041140 - Order an SAP S/4HANA fully-activated appliance for on-premise deployment

- **Cloud – SAP Cloud ABAP Platform:**

  Blog: ABAP custom code analysis using SAP Cloud Platform (section "Technical Setup")

For the creation of network users and authorizations, your network admin can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapters 2.1.1.2 Configuring the User – 2.1.1.4 Configuring the Connection Between Central Check System and Checked System)

For activating the Fiori app, your system admin can refer to the Custom Code Migration Guide once more. In addition, there is a useful section on the SAP Fiori App in the Meet-the-Expert (MTE) session: Custom Code for SAP S/4HANA Conversions.

Custom Code Migration Guide (specifically Chapters 2.1.1.2 Configuring the User – 2.1.1.4 Configuring the Connection Between Central Check System and Checked System)

MTE: Custom Code for SAP S/4HANA Conversions - SAP Fiori App (Minute 9:40-19:40)

This is also a first good time to decide if you want to tackle the custom code topic on your own, or if you want to get help from outside, that is, from your partner, or from SAP. Your Account Executive or Enterprise Support Advisor can help you identify the appropriate service offerings.

5 Analysis: Unlocking the secret of how much there is to do

Now that you have your SAP S/4HANA check system in place and have connected it to your current ERP, all complete with authorized network users, it's time to embark on the actual analysis. Essentially what you do is **start the app, specify the target release, add the usage data from your previous SCMON/SUSG prep work, and read the analysis results.**

For starting the app, your system admin can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapter 2.1.3.1 Creating a Custom Code Migration Project)

For adding usage data, your system admin can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapter 2.1.3.2.1 Adding Usage Data to your Custom Code Migration Project)

For reading the analysis results, your lead developer can refer to the Custom Code Migration Guide. In addition, there is a useful section on results analysis in the Meet-the-Expert (MTE) session: Custom Code for SAP S/4HANA Conversions.

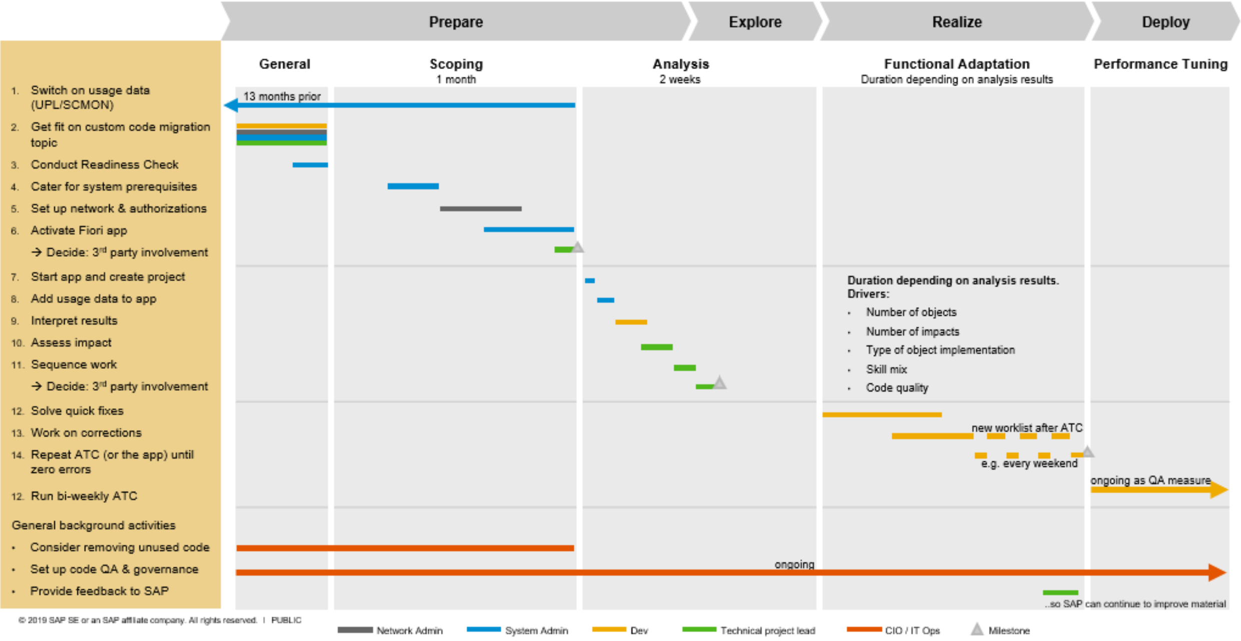
Custom Code Migration Guide (specifically Chapter 2.1.3.3 Analyzing the Findings)

MTE: Custom Code for SAP S/4HANA Conversions – Results Analysis (section "Viewing the results of SAP Fiori Custom Code Migration Project" Minute 17:30-32:40 and "Analyzing the Impact Analysis result in Detail" Minute 35:00 – 48:30)

A quick remark on what the tool brings up: The tool will display all remediation changes or impacts that are required to be made to your custom code in the context of the SAP S/4HANA transition. Period. If you are looking for the true minimal work to be done, this SAP tool is your unbiased friend. It's therefore a perfect little helper to define the no-frills scope to be done, so you can also use it to double-check the results of other tools available on the market.

Once you have your analysis results available, the next step is for your technical project lead to assess the time the actual fixing work is likely to take, and sequence the work accordingly. While it is comparatively easy to tag a rough time assessment to the scoping and analysis work, providing an **assessment for the duration of the actual fixing work** is quite a different animal. It is heavily **influenced by the number of objects that require fixing, the number of impacts found, the type of object implementation, the skill mix of the developers involved, and the overall quality of your custom code.**

Now that you know the impact of the fixing work to be done, this is another good time to re-assess if you want to continue tackling the custom code topic on your own, or if you want to get help from outside, that is, from your partner, or from SAP. In that case, SAP Enterprise Support Advisory (for Enterprise Support customers) or your regular SAP service contact can help you identify the appropriate service offerings.



Prepare · Explore · Realize · Deploy

General · Scoping (1 month) · Analysis (2 weeks) · Functional Adaptation (Duration depending on analysis results) · Performance Tuning

1. Switch on usage data (UPL/SCMON)
2. Get fit on custom code migration topic
3. Conduct Readiness Check
4. Cater for system prerequisites
5. Set up network & authorizations
6. Activate Fiori app
   → Decide: 3rd party involvement
7. Start app and create project
8. Add usage data to app
9. Interpret results
10. Assess impact
11. Sequence work
    → Decide: 3rd party involvement
12. Solve quick fixes
13. Work on corrections
14. Repeat ATC (or the app) until zero errors
12. Run bi-weekly ATC

General background activities

- Consider removing unused code
- Set up code QA & governance
- Provide feedback to SAP

Duration depending on analysis results. Drivers:

- Number of objects
- Number of impacts
- Type of object implementation
- Skill mix
- Code quality

Project plan – custom code activities in sequence

As you can see from the project plan above, this constitutes the end of your analysis activities. Unless you want to decommission some custom code that you have identified as no-longer-to-be-used, you can now turn your attention to other conversion-related topics. You pick up the custom code topic again in the Realize project phase, that is, once you actually have a converted SAP S/4HANA DEV system in place.

6 Functional Adaptation: Rolling up your sleeves and doing the fixing

While all analysis has happened in the check system, the actual fixing work takes place in your newly converted SAP S/4HANA system. Your SAP S/4HANA DEV system newly converted from a system copy of your ERP DEV, to be more precise.

So, what is the work to be done here? It consists of first applying quick fixes, solving pseudo-comments, and handling exemptions. It is then followed by fixing the coding according to error classifications until the check tool produces no more errors.

In order to do this, you first need to activate SAP ABAP Test Cockpit (ATC) in your SAP S/4HANA system (that's the check tool just mentioned) and set up SAP ABAP Development Tools (ADT) in Eclipse.

For activating ATC in the SAP S/4HANA system, your system admin can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapter 3.2 Running Local SAP S/4HANA Checks. This also includes which check variant to pick.)

For setting up ADT (Eclipse), your developers can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapter 3.3.1 Displaying Active Results)

For applying quick fixes and solving pseudo-comments, your developers can refer to the Custom Code Migration Guide once more. There is also a blog on semi-automatic custom code adaptation that deals with precisely this topic. In addition, there is also a Meet-the-Expert (MTE) session that includes a demo.

Custom Code Migration Guide (specifically Chapter 3.3.3.1 Applying Quick Fixes, and 3.3.3.2 Using Pseudo Comments)

Blog: Semi-automatic custom code adaptation after SAP S/4HANA system conversion

MTE: Custom Code for SAP S/4HANA Conversions – Adaptation Activity (Minute 31:20 – 42:00)

For handling exemptions, your developers can refer to this blog:

Blog: Remote Code Analysis in ATC – Working with Exemptions

For fixing the coding according to error classifications, there is another blog for your developers, plus the above-mentioned Meet-the-Expert (MTE) session once more:

Blog: ABAP Test Cockpit (ATC) for Developers in Eclipse

MTE: Custom Code for SAP S/4HANA Conversions – Adaptation Activity (Minute 18:10 – 31:20, 42:00 – 54:00)

As your developers work on the code fixes, it makes sense to re-run ATC periodically to receive an updated list of the work that remains to be done. A handy time to do this is every weekend during your code fixing period, and then working your way through the updated error list starting on Monday.

Finally, make sure you **keep your modifications and enhancements in mind**. Don't forget to perform the traditional SPAU, SPDD and SPAU_ENH activities for objects for which SAP provides a new version. For each of them, you will have to decide to either keep it or go back to the SAP standard:

Blog: SAP S/4HANA System Conversion – Custom code adaptation process

For running ATC until zero errors, your developers can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapter 3.3.4 Rechecking the Active Result)

Zero errors? You've done it! Your custom code is all nice and shiny! In order to make sure it stays that way, it makes sense to re-run ATC on a bi-weekly basis when your developers add new custom code.

For running bi-weekly ATCs, your developers can refer to the Custom Code Migration Guide:

Custom Code Migration Guide (specifically Chapter 3.2.3 Scheduling Local ATC Run Series, and 3.4 Rerunning Local SAP S/4HANA Checks)

For developing additional custom code, your developers can refer to the best practices laid out the following blog. We also recommend they check the ABAP Testing and Analysis Community for "Clean ABAP" and follow the programming guidelines assembled by the German SAP user group DSAG.

Blog: Best Practices for ABAP Development on SAP Netweaver 7.5x

SAP Community: ABAP Testing and Analysis ➔ check for Clean ABAP

DSAG Recommendations: Best Practice Guidelines for Development – Useful tips for ABAP Development

A quick remark on extensibility. Some errors you might find in your custom code will turn up because the related functions have been discontinued by SAP with no replacement (these are called simplification items, and the SAP Readiness Check you conducted in the very beginning of your custom code project has a dedicated section on that too). Your related custom coding will no longer work, so it's time for you to re-think, or (so much better in our opinion) re-design.

Re-thinking means you stick with the way you have previously run your process and re-write new custom code for this, for example by copying previous SAP standard and your own custom coding into your namespace and adjusting it.

Re-designing, by contrast, relates to the business innovations SAP brings in via Fiori, for example simpler processes, or new business value.

For SAP Fiori extensibility and the value it adds, see the SAP Fiori S/4HANA Extensibility Wiki and this blog:

SAP Wiki: SAP Fiori S/4HANA Extensibility

Blog: Fiori for SAP S/4HANA – Top 5 Fiori features that bring value to every Business User

7 General: Housekeeping

During the Prepare phase of your project, your IT Operations defined new quality guidelines together with a dedicated QA team. Undoubtedly, you will have learned some lessons during the project, and now it's time to adjust your guidelines to these learnings.

For reference on adjusting custom code guidelines and QA, your IT Operations can once more refer to the following programming guidelines assembled by the German SAP user group DSAG:

DSAG Recommendations: Best Practice Guidelines for Development – Useful tips for ABAP Development

www.sap.com/contactsap