

Need to Get Your Customizing and Testing Ready for a Global Rollout?

Use BC Sets and the Test Organizer to Smooth the Way!

Dr. Matthias Melich



Dr. Matthias Melich is Product Manager of SAP's Advanced Implementation Solutions department, whose tasks comprise the technical backbone for SAP's hosting offerings and generic customizing concepts and tools for the mySAP.com system. Matthias joined SAP in 1995, and since 1998 has been responsible for Product Management of Customizing Tools.

(complete bio appears on page 58)

Customers with numerous subsidiaries and installations often want to establish an SAP system landscape based upon a common *standardized core*, otherwise known as a *global template*, which reflects all of the R/3 settings and data that are specific to a corporate entity. Descriptions of common company processes, instructions regarding how (and how often) subsidiaries should implement (and update) the template, transport files, customizing settings, test descriptions, and test plans are all fair game for a global template. Some of these elements address the methodology that must be adhered to by subsidiaries as the corporate headquarters location rolls out the template, and some, particularly the customizing and testing elements, are about concrete R/3 settings and data.

It's this latter category of elements — the ones that require you to capture specific customizing settings and then, through proper testing, ensure that they can be properly invoked at subsidiary locations — that I will address in this article.

First, I will show you how BC Set technology, which was introduced in Release 4.5, can be used to capture the customizing settings that determine your company's standard business processes.¹ Then I will show you how to use the Test Organizer, a very powerful and very useful built-in testing facility introduced back in Release 3.1, to identify which processes to test and monitor their progress. But don't stop reading if preparation for a global rollout isn't on your current to-do list! Even if you are not undertaking a global rollout, this article

¹ For a comprehensive introduction to BC Sets, see "Enhancements in Customizing: Business Configuration Sets, the Customizing Cross-System Viewer, and the Activity Log," by Matthias Melich, in the November/December 1999 issue of the *SAP Professional Journal*.

Why Use Global Templates?

So, why should large companies now invest in global templates to accomplish their rollouts?

The main reason is reduced costs. If a company defines and implements a global template, the costs for subsidiary systems decrease because all local systems are (to a certain extent) identical (depending on the scope of the template), i.e.:

- ✓ The implementation of new subsidiary systems is accelerated because instead of building a new system from scratch, it can be jump-started via the global template.
- ✓ The support of local (standardized) systems is cheaper than the support of individual systems because the local support organization can draw on central knowledge.
- ✓ The knowledge transfer from headquarters to subsidiaries is easier because each local implementation can make use of centrally developed (standard) presentations and training material.

provides valuable information that can be applied in other customizing contexts, including:

- **Reuse of customizing:** If you are a consultant and you have to set the same parameters at each customer site, you may consider storing those settings in BC Sets (on your laptop) and uploading them to a customer system the next time you need them.
- **Preconfiguration:** If you are an SAP partner

specializing in a particular industry sector, you may want to consider BC Sets as a means to preconfigure a system.

- **Customizing variants:** Since BC Sets store customizing in a container separate from the original customizing tables, they can be used to produce variants of a process. This makes it possible for SAP partners to provide customers with a selection of variations of the same process, from which the customer can pick the one that fits best.
- **Documentation of parameters/validation:** Some companies need to thoroughly document which customizing settings were active in the system at a given point in time. BC Sets are perfectly suited for this task because they can be used to copy settings and store them in a safe place (away from the customizing tables).
- **Testing:** The Test Organizer can obviously be used for other purposes as well, such as upgrades. I recommend that you take a close look at this tool before spending money on a third-party testing tool, because the Test Organizer is part of SAP's standard delivery — i.e., you have it in your system, and it does not cost extra money!

Understanding the Process of a Global Rollout

A global rollout often proceeds like this:

1. Company headquarters develops a global template that defines which customizing values are to be standardized across the company's systems and what needs to be tested after template implementation. The customizing and testing parts of the template are usually developed in a reference

system. I will show you how this is done in this article.

2. After the template is completed, it is shipped to the subsidiaries, where it is implemented, localized, and tested. In this phase, subsidiaries sometimes stumble across definitions that are in conflict with local needs. In these cases, the subsidiaries have to be able to override the global settings with local settings.²
3. When the initial rollout is completed, headquarters starts to develop the next version of the template. The next version of the template can be independent of SAP release — i.e., it is possible that the same R/3 release is used as the system basis of the next global template. This process requires that the old release of the template be kept and that the new release be developed on the basis of the old one.
4. When the development process of the new version is completed, the template needs to be shipped to the subsidiaries, whose task is then to implement the new version of the template. The next sections outline the complete process.

In order to accomplish the production and implementation of a global template, headquarters must have the ability to:

- Capture settings that are common across the company, and identify settings that are mandatory for all subsidiaries.
- Put these settings into a container and roll them out to the subsidiaries, which must be able to apply these settings to their systems easily, without accidentally overwriting existing settings.

² This process will be the focus of my next article, where I will show you how BC Sets are applied in the subsidiary system and how they are tested. I will also show you how the subsidiaries can deviate from the template if they cannot work with definitions made by headquarters.

- Lock those customizing global settings that are *not* to be changed by the subsidiaries, and at the same time supply subsidiaries with a mechanism that enables them to digress from the template in a controllable fashion.
- Periodically check whether the subsidiaries are in compliance with the global settings.
- Deliver bug fixes for global customizing settings that proved to be wrong after the rollout.
- Deliver a new template version to the subsidiaries that can be imported into a system without going straight to the original customizing tables.

In terms of testing, both headquarters and subsidiaries must be able to:

- Define which processes need to be tested and how they need to be tested.
- Create reusable testing procedures that can be compiled flexibly and easily.
- Administer and monitor tests in a multisystem landscape.
- Match test cases and resources efficiently.
- Evaluate the progress and status of the testing activities.

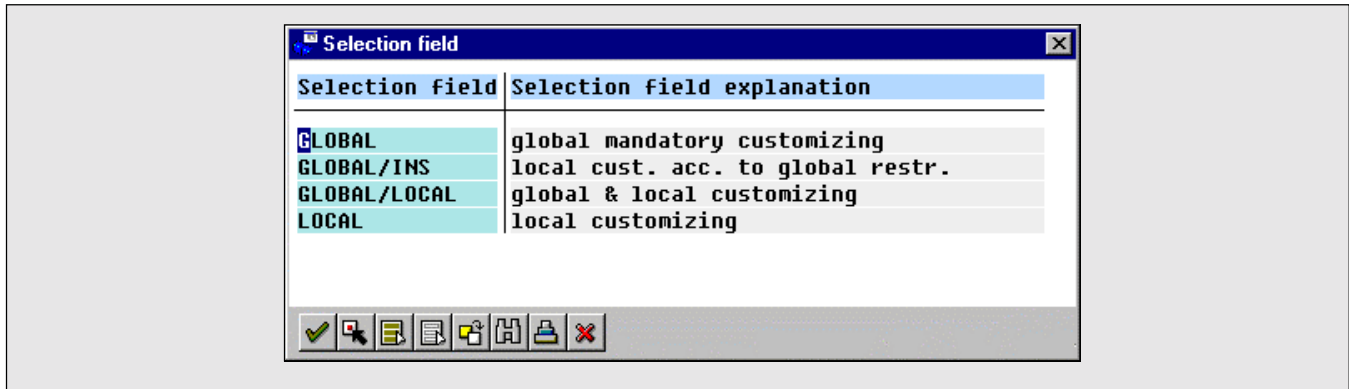
It is no coincidence that this is precisely the feature set you get with BC Sets and the Test Organizer!

Customizing in Global Rollouts Before Release 4.6C (and Before BC Sets)

As I mentioned earlier, BC Sets were introduced in Release 4.5, but with limited functionality. For

Figure 1

Selection Field Pop-Up



example, in Release 4.5 it is possible to create BC Sets and define which customizing settings are not to be changed in subsidiary systems. However, it is not possible in 4.5 to copy BC Set content into the system customizing tables (i.e., activation). It also is not possible in 4.5 to lock parameters for all subsidiaries in the subsidiary system or to convert customizing transport requests, both of which are possible in 4.6C. These were the last missing pieces needed to enable global rollouts.

And what about customizing prior to Release 4.5?

Before BC Sets

So how *do* you prepare customizing in pre-4.6C environments, when there were no BC Sets at all? The process consists of:

- Identification of the customizing activities that contain group customizing.
- Tagging of these activities with the help of selection fields. These fields offer the ability to assign attributes to customizing activities. The values of the selection fields can be defined by the company. **Figure 1** provides an example of how to define selection fields.

Here, the company has defined four values:

- **GLOBAL:** Activities with this attribute only

contain customizing that is centrally defined and must not be changed locally.

- **GLOBAL/INS:** Activities with this attribute contain customizing that is to be used as a template during localization — i.e., this customizing is defined by the company headquarters and re-created by the subsidiaries with local values.
- **GLOBAL/LOCAL:** Activities with this attribute contain global and local customizing.
- **LOCAL:** Activities with this attribute only contain local customizing.

Along with tagging the activities, the company had to document (using the activity notes) which parameters of the activity belong to the template, and also the status of those activities (i.e., “mandatory” or “default”).

When all the activities are tagged and the parameters described, this information is shipped to the subsidiaries, who then have to go through all global activities and customize the local system according to the status of the activity (selection field value) and the written documentation of the activity notes.

The shortcomings of this approach are obvious:

- **No link between standard customizing and standard processes:** Usually a process spans

several activities (see the next section for details). However, as the documentation is bound to just one activity, it is difficult to establish cross-activity customizing relationships. The only workaround to this problem is that you select one activity as the central place for the complete documentation of this process and tell the customizer that he needs to go to other activities and set parameters to define the complete process.

- **Unsatisfactory reuse:** The implementation of the template in the subsidiary system requires a lot of manual labor — you have to customize using parameter templates, you have to adjust customizing that comes into the local system from headquarters, etc. Much of this effort could be saved if it were possible to move the complete group customizing into the local system with a few mouse-clicks and then start your localization.
- **Higher chance of errors:** The manual labor in the local system is also a possible source of errors, which could be reduced if a “customizing move” procedure existed.
- **No deviation checks possible:** Because the standard customizing is insufficiently encapsulated in the template, it is very difficult to determine after the template implementation which values belong to the template and which do not. This delays changes in customizing at a later time because you have to first determine for each parameter whether or not you are allowed to change it.
- **No easy detection of source:** It is also very difficult to find out where the customizing in your tables originates. Did it come in with the template? Was it created manually? Is it part of SAP’s standard client customizing?
- **Focus on lines instead of fields:** The customizing delivery mechanism is the Change and Transport System (CTS). The CTS moves the standard customizing from the reference to the subsidiary system. The smallest customizing unit that the

CTS can handle is a table line. This is, however, not small enough for global rollouts because lines may contain standard mandatory values, default values that can be overwritten locally, and values that are not predefined by the headquarters. It would be helpful if the transport mechanism could be told to just move values belonging to the first two categories and leave out customizing of the third category. This is, however, not possible in pre-4.5 releases.

- **No attributes at the field level:** Another drawback of the selection field approach is that you cannot assign the aforementioned attributes to the customizing values. Selection fields only allow attributes on the activity level, not on the line level. The value attributes therefore have to be documented using the activity notes, but this obviously becomes tiresome. It would save a lot of documentation labor if you could attach the attributes directly to the values.

The Release 4.6C BC Set Technology

The BC Set technology remedies the deficiencies of the selection field approach by providing the ability to:

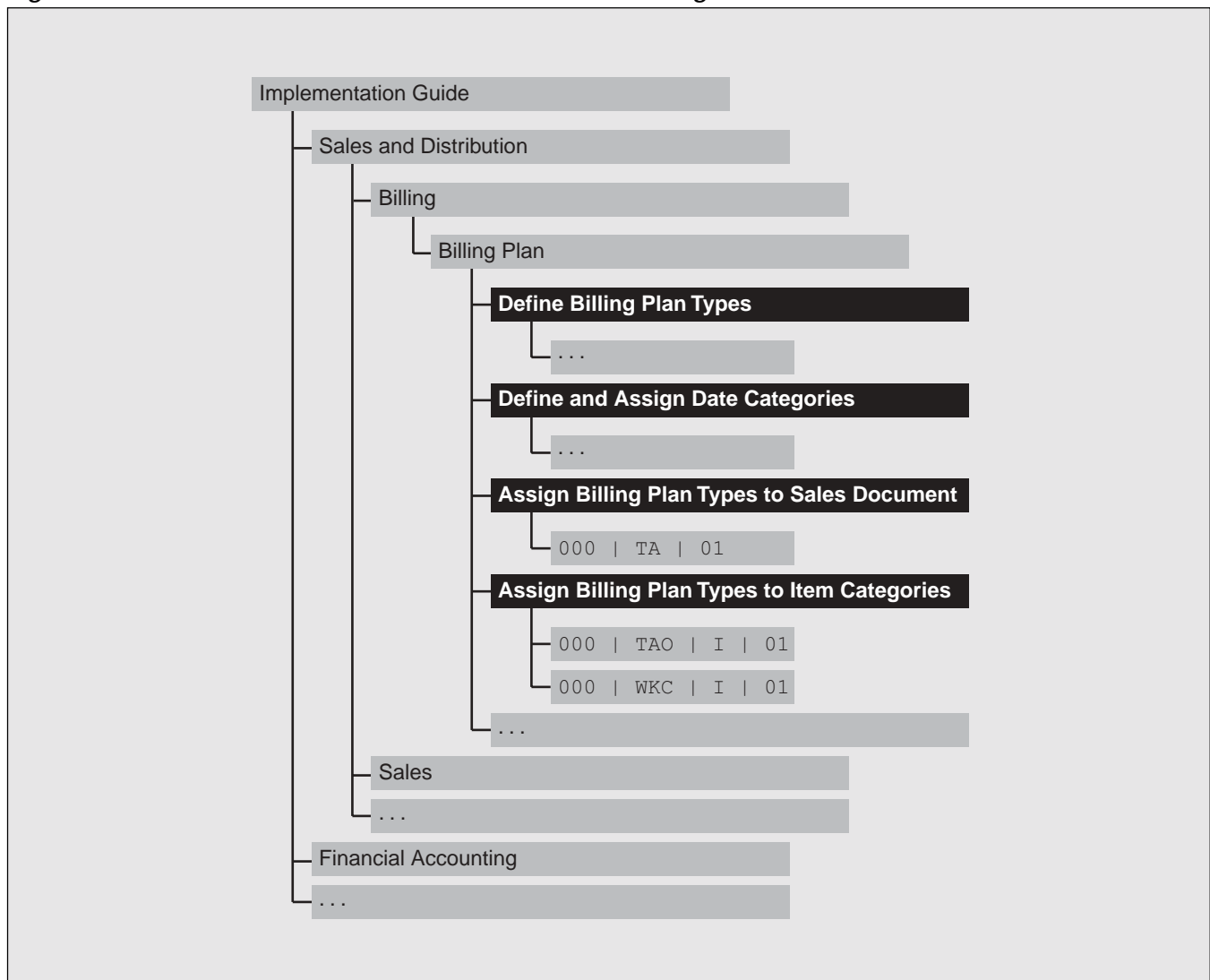
- Encapsulate settings across several activities
- Reuse these settings in the subsidiary system with a few mouse-clicks
- Focus on fields and not on lines
- Assign attributes at the field level

I will now explain the BC Set concept in more detail and how it relates to the tasks that you are faced with in a global rollout.

Much like a photograph, BC Sets capture an image of customizing settings. They are a container that lets you store customizing information (i.e., values and attributes) separately from the customizing

Figure 2

Milestone Billing



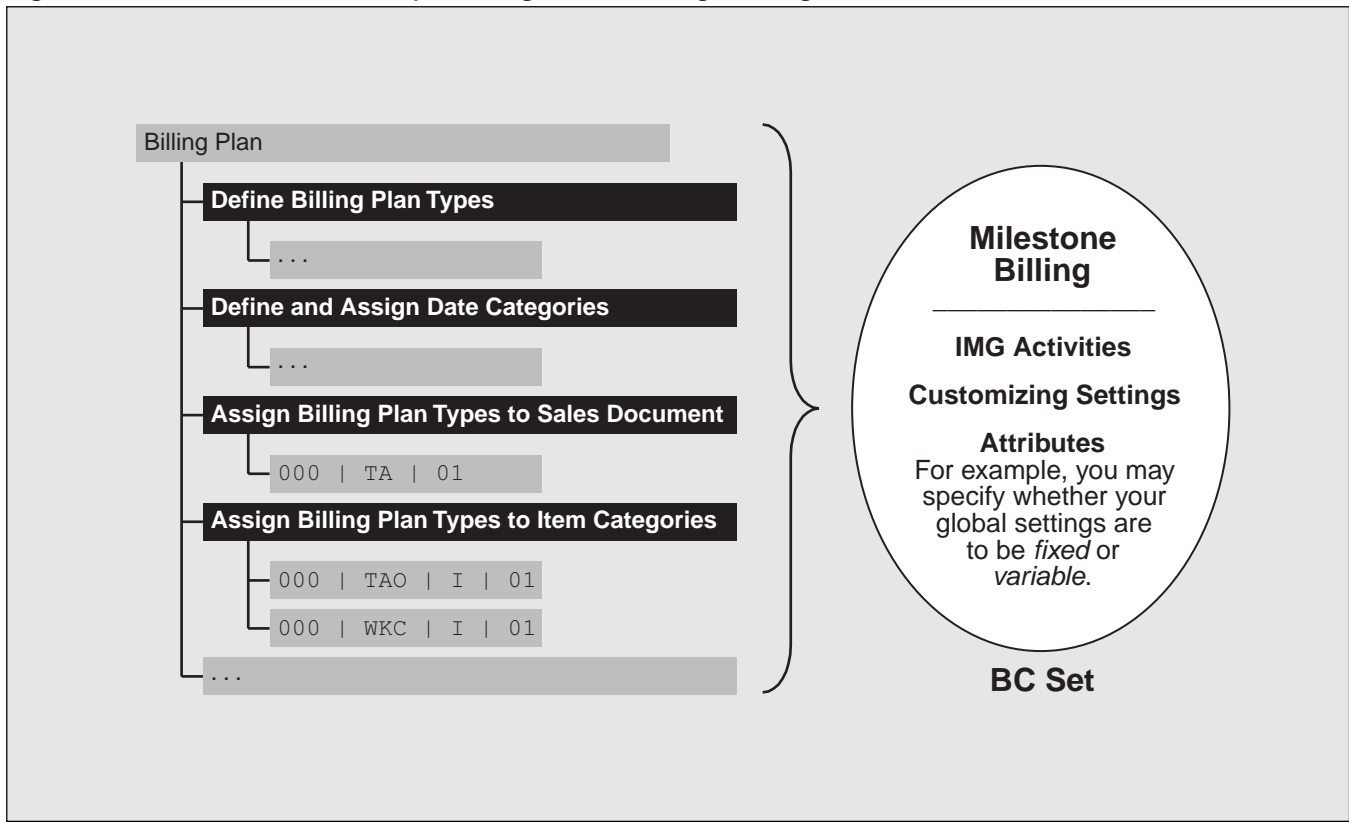
tables. BC Sets make it possible to enter values from several activities. Take, for example, a business function like “milestone billing.” If you want the subsidiary systems to perform payments according to a predefined schedule, you need to configure the reference system accordingly and integrate the customizing settings that define this business process into the global template.

This means two things: you need to know exactly which activities to go to in the Implementation Guide (IMG), and you need to know which customizing

values to enter in these activities. The diagram presented in **Figure 2** shows how the business function “milestone billing” is mapped to the various customizing tasks. The mapping of these types of business requirements is typically done by a trained consultant. In a global rollout, you should only need this consultant once — to define this process in the reference system.

When preparing a template without BC Sets, you have to:

Figure 3 Encapsulating Customizing Settings with BC Sets



- Assign selection fields to the activities (to show that they contain customizing, which is part of the global template).
- Save the individual lines of the customizing tables in a transport.

With BC Sets, you can first encapsulate all settings that belong to this specific function in one container. This allows you to establish the relationship between the business function and its customizing representation. This is shown in **Figure 3**.

The major advantage of using this “encapsulation” method rather than using the selection field method is clarity. A BC Set contains expert knowledge — it is the customizing equivalent of a business function. If all processes in a template are captured in BC Sets, the template customizing is well structured and meaningful. This means that you can iden-

tify at any stage of the global template life cycle what was part of the global shipment to the subsidiaries, which values were set during template implementation, what the relationship of customizing settings to business functions is, and so on. With the selection fields and the CTS, global customizing is just a heap of lines that is pushed from the headquarters’ reference system to the subsidiaries.

There are two additional advantages to using BC Sets rather than the selection field method in a global environment:

- BC Sets make it possible to add attributes to customizing settings. You may specify whether your global settings are to be *fixed* (i.e., mandatory and unchangeable for the subsidiaries) or *variable* (i.e., changeable for the subsidiary while applying the global customizing template in the local system).

- Customizing that is stored in BC Sets can be imported into a system without immediately influencing the customizing tables. This way you have a chance to see what enters the system before it overwrites your existing customizing. This is possible because the customizing settings that are included in a BC Set are stored in a container that is independent of the customizing tables. When you import a BC Set, the BC Set is moved from the transport file into this container without affecting the system's customizing tables.

BC Sets thus change the very nature of global customizing. The global customizing is stored in BC Sets, which remember what customizing activities are involved, which settings are included, and which attributes you assigned. This makes your global customizing more transparent.

Preparing to Use BC Sets

If you decide to use BC Sets to encapsulate group customizing, one of the first decisions you will need to make is how granular to make each BC Set. To my knowledge, there are two different schools of thought: the mosaic approach and the tile approach.

The *mosaic* approach creates a BC Set for each activity that belongs to a given process, and combines all of these BC Sets into one bigger BC Set that represents the business process. The *tile* approach includes all settings of all activities that are involved (in this process) in just one BC Set. The mosaic approach offers the advantage of better reuse of existing BC Sets. As the activity-based BC Sets are very small (they often contain only one line or part of a line), they can be reused in other process BC Sets (which make use of the same activity and the same parameters) more easily. The downside to this approach is that you wind up with a lot of BC Sets floating around in your system. The more there are, the more difficult it is to identify those BC Sets that need to be activated.

With the tile approach, the smallest units are business functions, not small, discrete activities, so there will be a lot fewer BC Sets to deal with. The downside to this approach is that there can be no reuse. If the same settings of one activity are part of two business functions, both BC Sets need to include these settings. This means additional development and higher maintenance effort. You are also faced with the problem that big BC Sets may run into time-outs during activation in the subsidiary system. (It is therefore advisable to test the activation of large BC Sets before including them in the global template!)

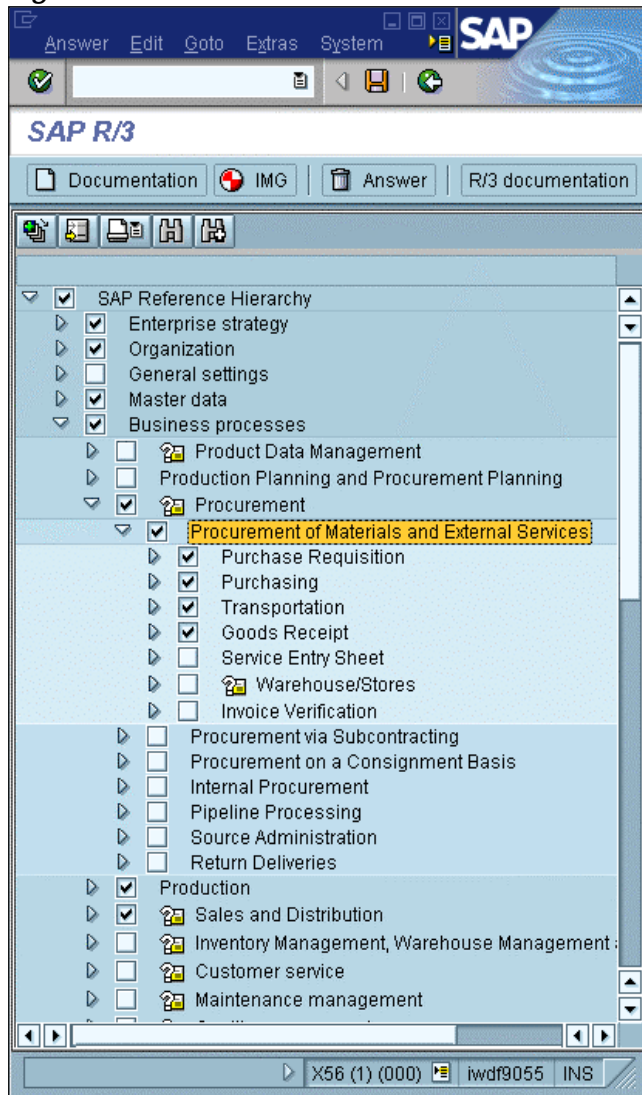
As both approaches have their pros and cons, it is difficult to say which should be used. In this article, I will assume that you are using the tile approach because these BC Sets can be created very quickly using transport requests. If you prefer the mosaic approach, I recommend you refer to my first BC Set article in the November/December 1999 issue of this publication where I demonstrate how to create small (activity-based) BC Sets.

If you are using the tile approach, you should take the SAP reference structure as orientation. You still need to determine what a process is, and the reference structure provides you with a structured list of processes — this is why I recommend using it for orientation. The SAP reference structure, which is available in the ASAP Question & Answer Database, structures the business processes that the SAP system can perform. **Figure 4** illustrates this hierarchy.

Following the Global ASAP methodology, the best level of granularity for BC Sets is the business process group level (this is the third level of the hierarchy, highlighted in Figure 4).

There is one last planning aspect that I would like to mention — naming BC Sets. Even if you choose the tile approach, the number of BC Sets in your system may be significant. It is therefore a good idea to come up with a naming convention before you start creating BC Sets. Having a naming convention in

Figure 4 The SAP Reference Structure



place makes BC Set creation and reuse easier because when entering the BC Set maintenance transaction, the BC Sets are listed alphabetically. If you use the naming convention I recommend, then the BC Sets will be grouped according to components and then business processes. This makes it easy to find your way around in the list. Based on my experiences, the name should contain the following:

- The component that the BC Set belongs to — e.g., APO, CRM, etc.

- The name of the business process — e.g., Vendor Managed Inventory (VMI), etc.
- A number (in steps of 10)

This leads to a name such as APO-VMI-10. The number at the end offers you the ability to create different variants or even patches for the same business process. So, for example, you could have APO-VMI-11, APO-VMI-12, and so on, if necessary.

✓ Tip

Do not include any release numbers in the name (neither an SAP release nor an internal template release number). This information belongs in the short text and not in the BC Set name because the name cannot be changed at a later date, whereas the short text can.

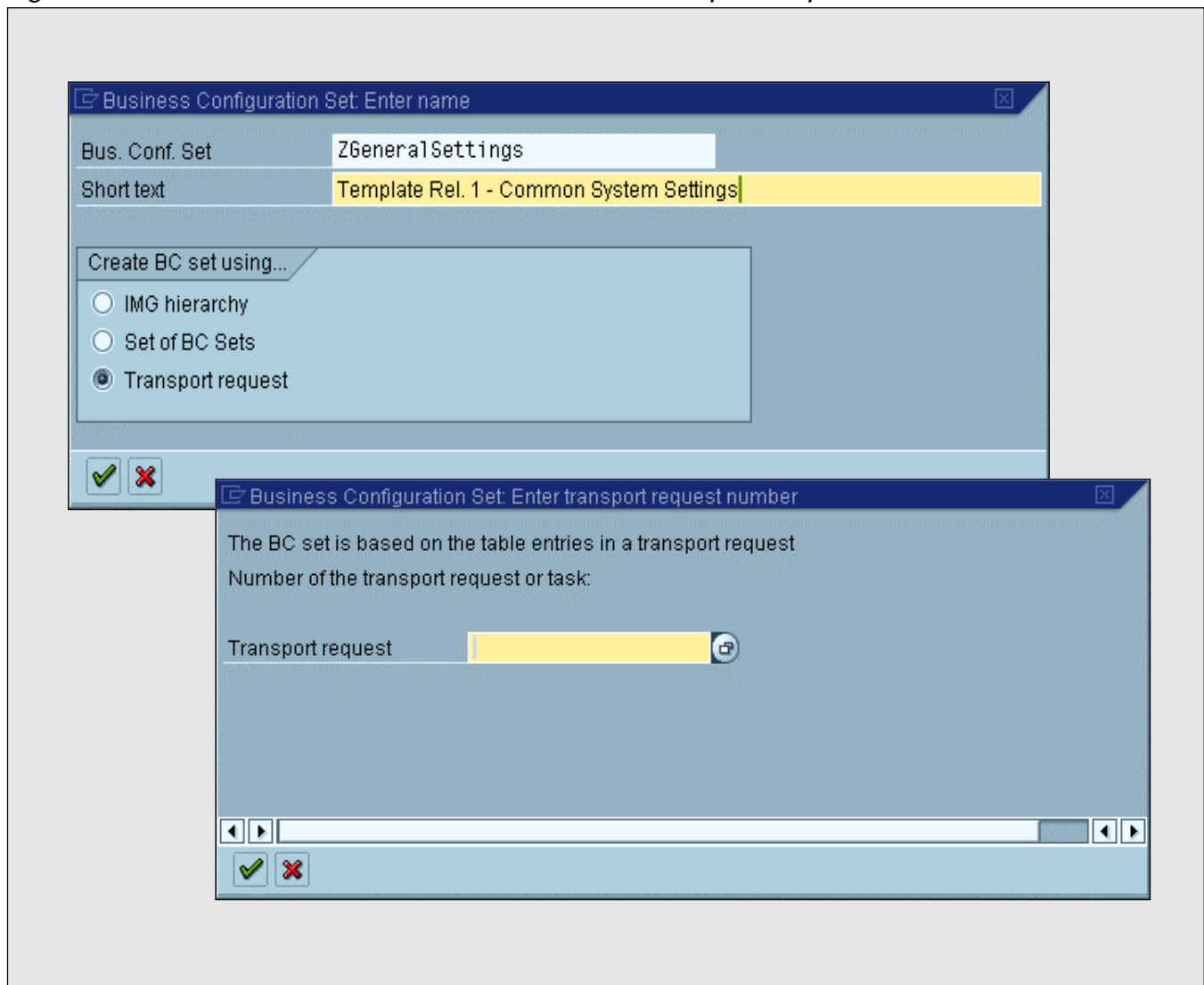
Creating a BC Set from a Transport Request

At this point, I will assume that you created a transport request for each business process that you want to include in the global template, and that you have customized these processes and saved the customizing settings in the appropriate transport request. I will show you in this section how this data is converted into a BC Set. This functionality was partially available with Release 4.5, but with 4.5 the conversion was not completely automatic. The information concerning where activities reside in the IMG was not stored in the transport request, and had to be supplied manually during the conversion for those activities contained multiple times in the IMG. With 4.6C, however, the conversion is fully automatic.

To create a BC Set from a transport request in 4.6C, follow these steps:

1. Enter the BC Set maintenance transaction (transaction SPPR3).

Figure 5 Create the BC Set, and Enter the Transport Request Number




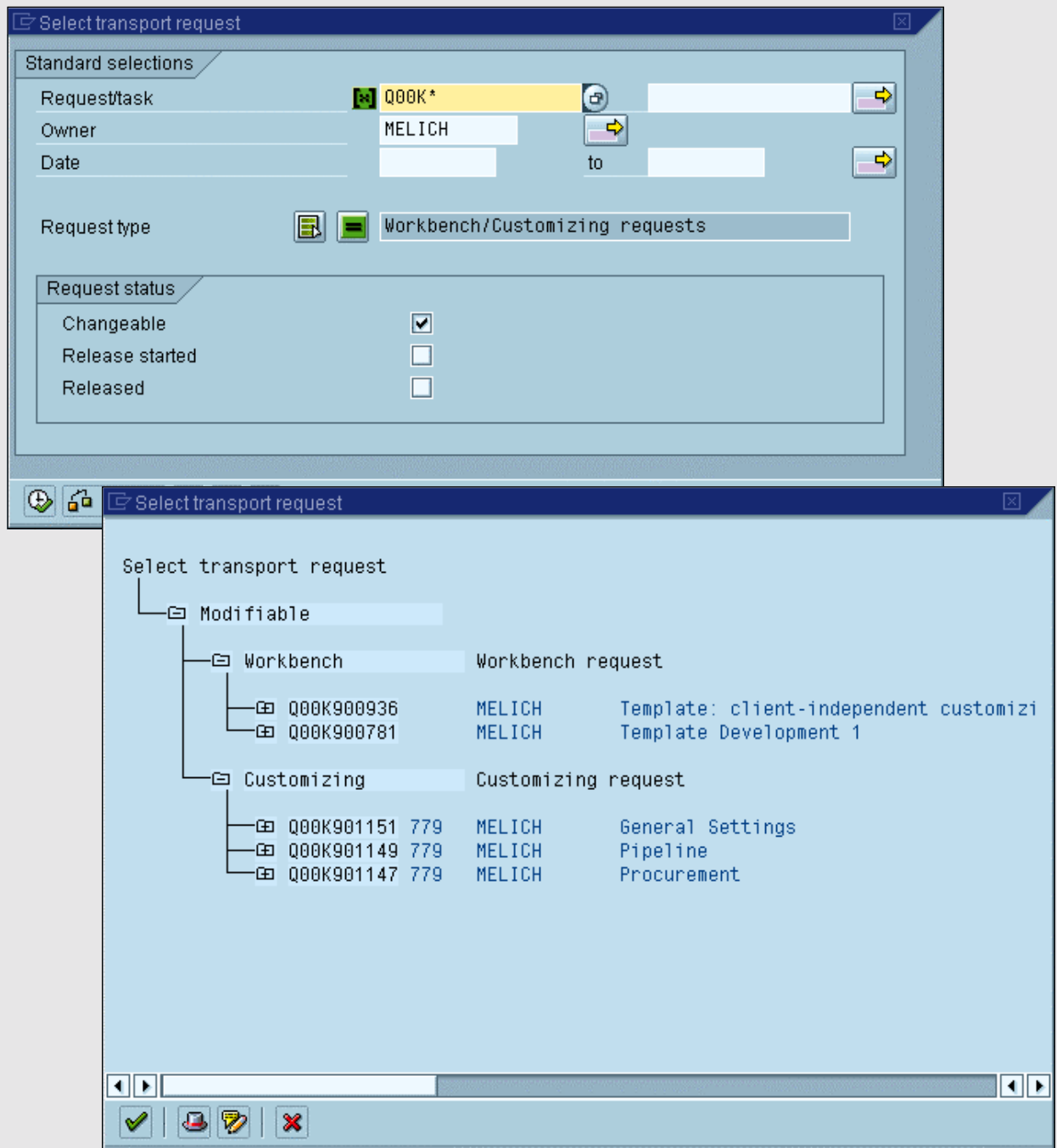
2. To create the BC Set, click on the  icon, which brings up the first screen shown in **Figure 5**.
3. Enter the BC Set ID. As you can see in Figure 5, I have entered “ZGeneralSettings.”
4. Enter a short text description. Here I have entered “Template Rel. 1 - Common System Settings.”
5. Click the **Transport request** radio button.
6. After accepting these entries, the second screen in

Figure 5 appears, which prompts you for the number of the transport request.

7. To view a list of the transport requests in the system, call up F4-Help,³ which launches the first screen in **Figure 6**, where you can fill in the appropriate fields.

³ A BC Set can only import one transport — i.e., you cannot merge two transport requests into one BC Set. If you need to combine information from two or more transports, import them into two or more individual BC Sets and combine the BC Sets into a larger BC Set.

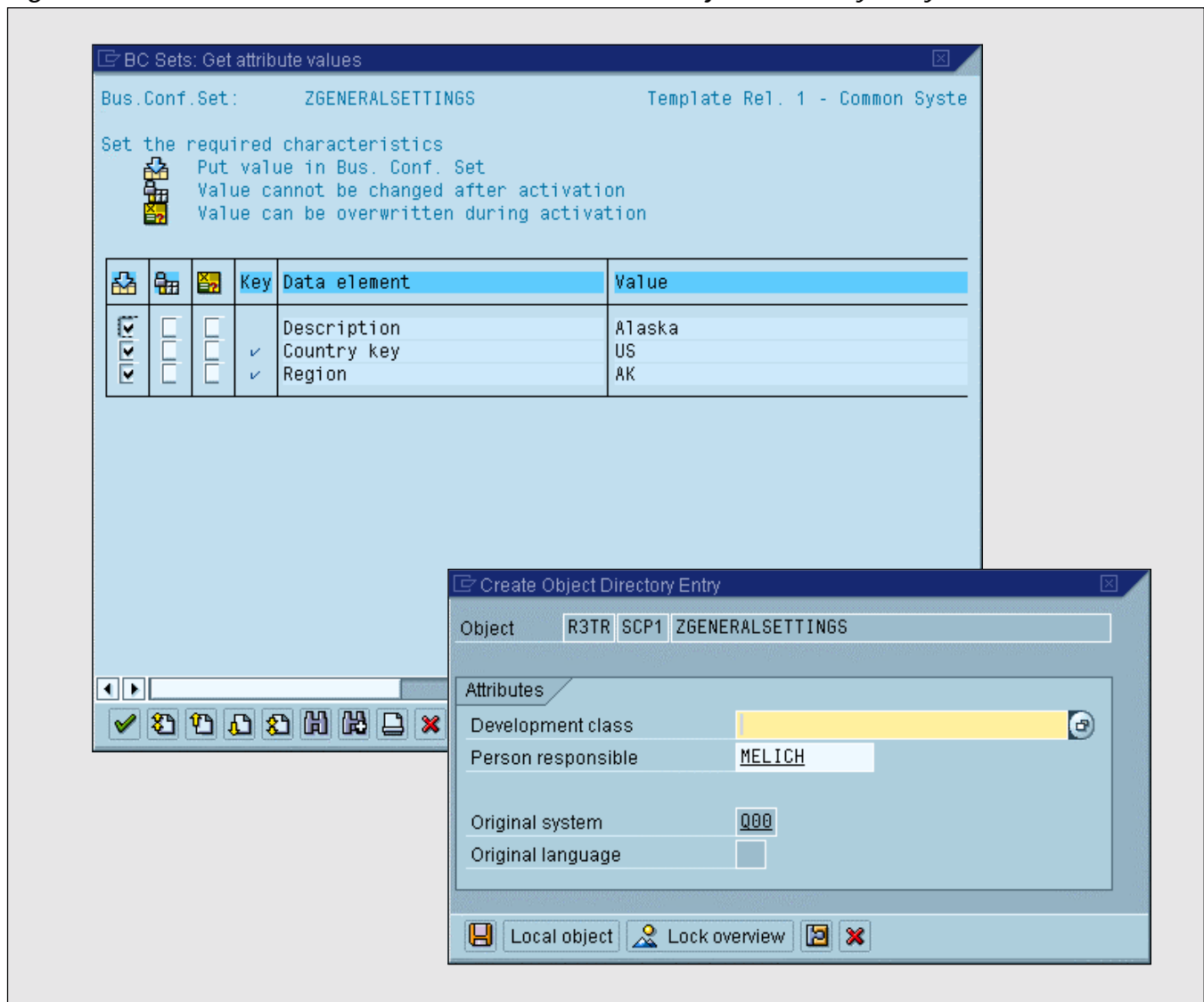
Figure 6 List All Transport Requests, and Select the Transport Request to Display



8. I have indicated that I am looking for all requests that I own, which results in the list shown in the second screen in Figure 6.

Here, I select one request — the General Settings transport request “Q00K901151” under “Customizing.”

Figure 7 The “Get attribute values” and “Create Object Directory Entry” Screens



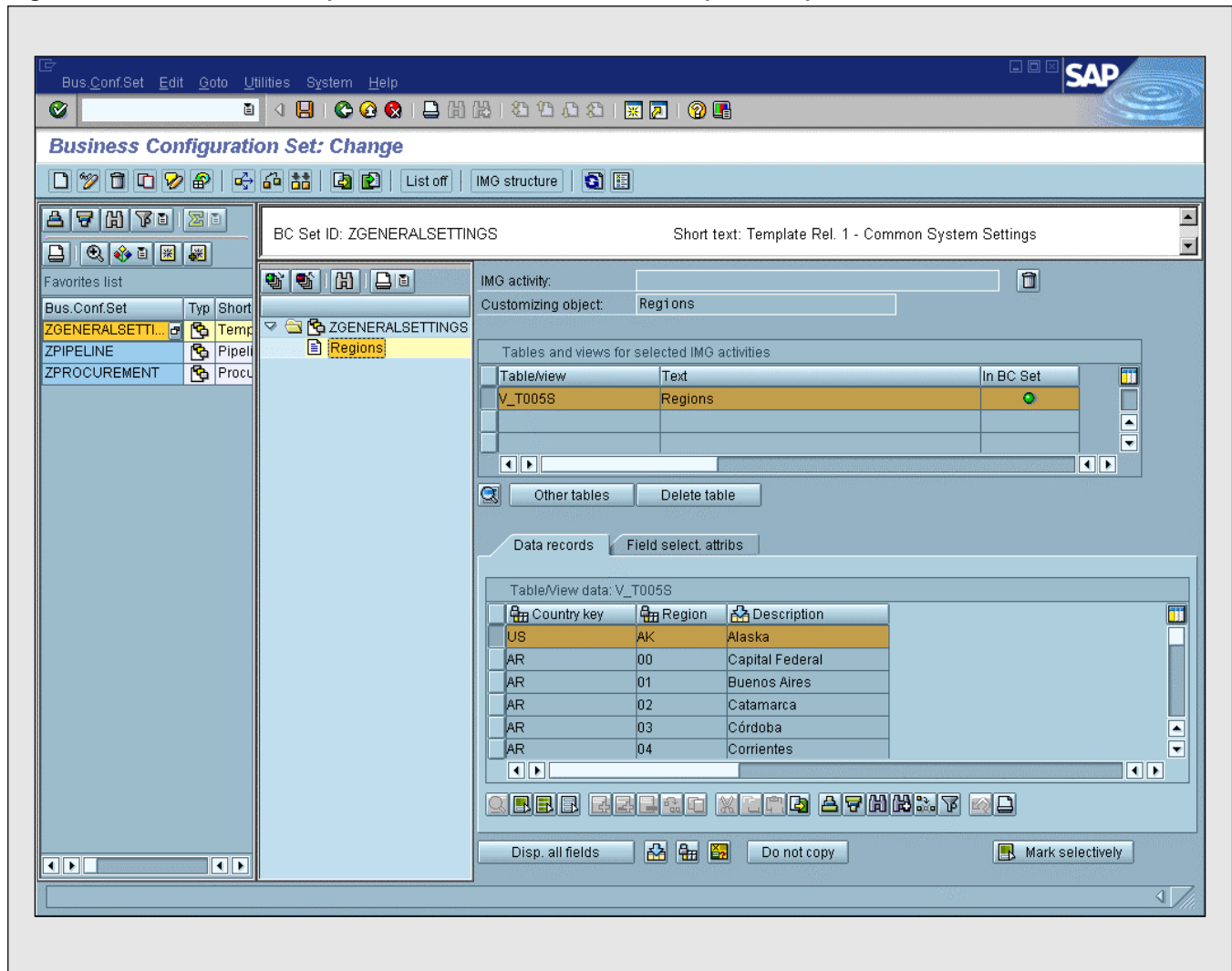
9. After accepting this choice, the system returns you to the “Enter transport request number” prompt (Figure 5) with the correct transport request number now filled in — “Q00K901151.” Accepting the transport request here brings you to the first screen shown in **Figure 7**.

The transport request only contains customizing data, but a BC Set additionally contains *attributes* for the customizing data. On the “Get attribute values” screen in Figure 7, you can add these attributes. The assignment of attributes can,

however, also be done in the regular maintenance mode of a BC Set. I suggest, therefore, that you just confirm this pop-up.

10. Finally, this brings you to the “Create Object Directory Entry” screen, which is the second screen shown in Figure 7. Remember that our task is to create a BC Set, and so far we have only selected the data source for the BC Set. Now we actually create the BC Set in the system — i.e., we register the BC Set in the system object list.

Figure 8 Completed Conversion of the Transport Request Content



This is where you have to assign the BC Set to a development class whose objects are transportable. You should assign a development class that contains objects that belong to the global template.

The conversion of the transport request content is now complete. **Figure 8** shows that:

- ZGENERALSETTINGS, the newly created BC Set, was added to the list of favorite BC Sets (because I decided that all BC Sets I create will appear in my favorites list — see the sidebar that

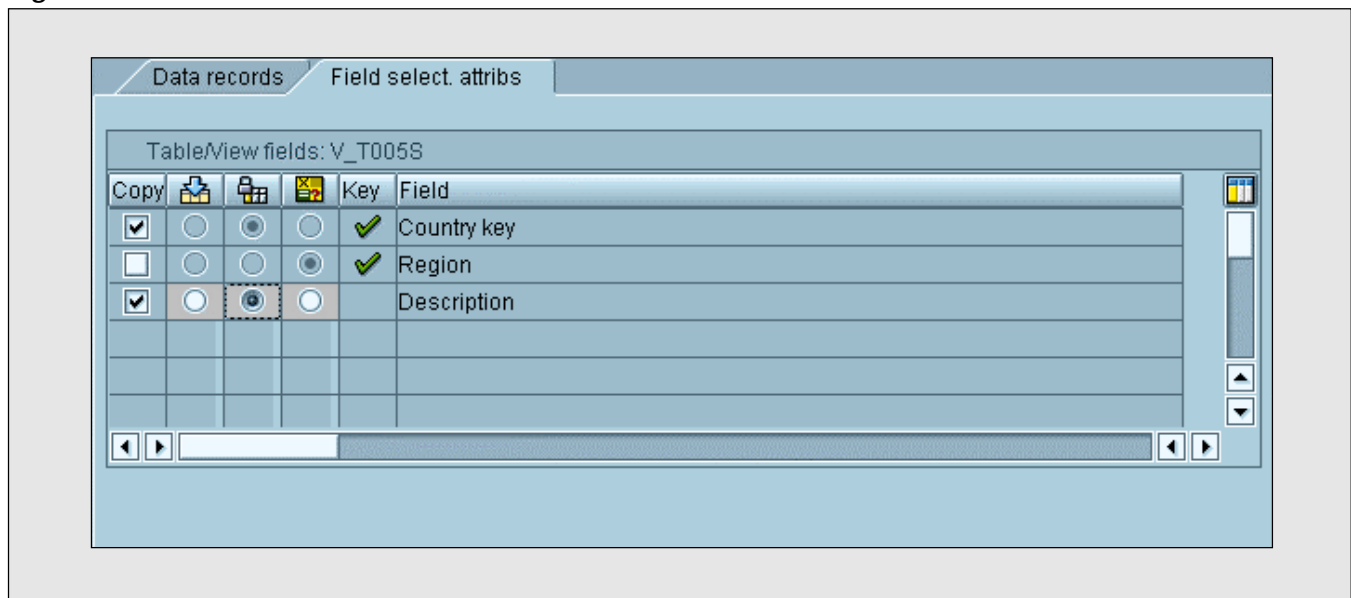
starts on page 44 titled “New in Release 4.6C — Favorite Management!”).

- The second column contains the names of the customizing activities that provided customizing data for the converted transport. In our example, all data stored in the request stems from one activity — “Regions.”

As you can see in Figure 8, highlighting the activity activates the content pad on the right side of the screen.

Figure 9

The “Field select. attribs” Tab



You can see that in the top table control, the name of a view has appeared, “V_T005S.” The BC Set maintenance transaction automatically traces which tables and views belong to the Regions activity and lists them.

In the table control at the bottom, the database content of the highlighted view is listed on the “Data records” tab. The BC Set maintenance transaction automatically lists all lines that are included in the BC Set at the top.

This makes it easy to see immediately what is in the BC Set.

The attributes are maintained on the second tab, “Field select. attribs,” as shown in **Figure 9**. Here, all fields of the view are listed vertically. In front of the field names, there are five columns:

- In the first column (the **Copy** column) are a series of check boxes. These check boxes determine the columns from which data is to be copied from the database table to the BC Set. In the global context, you would leave out all columns filled strictly with local values and for which

you do not even want to suggest a “default” value.

- The next three columns are connected via a series of radio buttons:

If the button is set to this column, then the value is marked as “default” — i.e., the global headquarters suggests this value, but the value is not mandatory.

If the button is set to this column, then the values in this column are “fixed” — i.e., mandatory for the subsidiaries.

If the button is set to this column, then the value stored in the BC Set is “variable” and can be changed during BC Set application in the local system.⁴

- The fifth column, **Key**, indicates which fields are key fields. Because this information is stored in the technical settings of the table/view, it cannot be changed here.

⁴ Details about how to change these values in the local system will be provided in my next article.

When setting a radio button for the “Region” entry, you will notice that you have to clear the checkbox of the **Copy** column in order to select the variable column radio button you want. The reason for this is that “Region” is a key field. From a technical point of view, a key field needs to be specified. You cannot leave a key field open — i.e., you cannot tell the BC Set not to include this field. If you do not want the value in this field to be reused in other systems, then this value must be changed during activation. For this reason, this field is automatically assigned the attribute “variable.”

When you try modifying the “Description” field line, you will find that you *can* set these radio buttons — i.e., you can assign any attribute to this field and you can also de-select this column from your BC Set. This can be done because the “Description” field is not a key field — i.e., it need not be present when writing information to the database.

The usage of BC Sets for a global template changes the way customizing is shipped to the subsidiaries. It is important that all transport requests converted to BC Sets are *not* shipped to the subsidiaries. All you have to do is include the BC Sets that contain the headquarters’ customizing in the template delivery. If you ship both the BC Sets and the transports, the customizing entries get into the system via the customizing requests — and this is not what you want! If the settings are in the tables, then a lot of the advantages of BC Sets are lost (like checking what gets overwritten, etc.). If this happens, you have to activate BC Sets after import. The fixed settings will then be locked, etc. We want to load the central customizing information into the subsidiaries using the BC Sets because this is the only way to exploit the full BC Set functionality.

Once the global customizing of the referenced system is captured in BC Sets, the next step of template development is the creation of the testing section of the global template. Headquarters needs to test whether the processes run as planned, and the subsidiaries must do the same after implementation of

the template in their systems. Therefore, we now move on to the testing phase.

✓ Tip

The BC Set maintenance transaction automatically detects which fields are organizational units and automatically sets these fields to “variable.” The reason for this is that organizational units tend to be different in different systems, even if the customizing is the same. Reusing customizing stored in a BC Set in a different system presupposes that organizational units can be adapted easily (without modifying the BC Set). This can be achieved with the “variable” attribute. If you do not want to change the value of an organizational unit during activation, you have to change the attribute from “variable” to “default” or “fixed.”

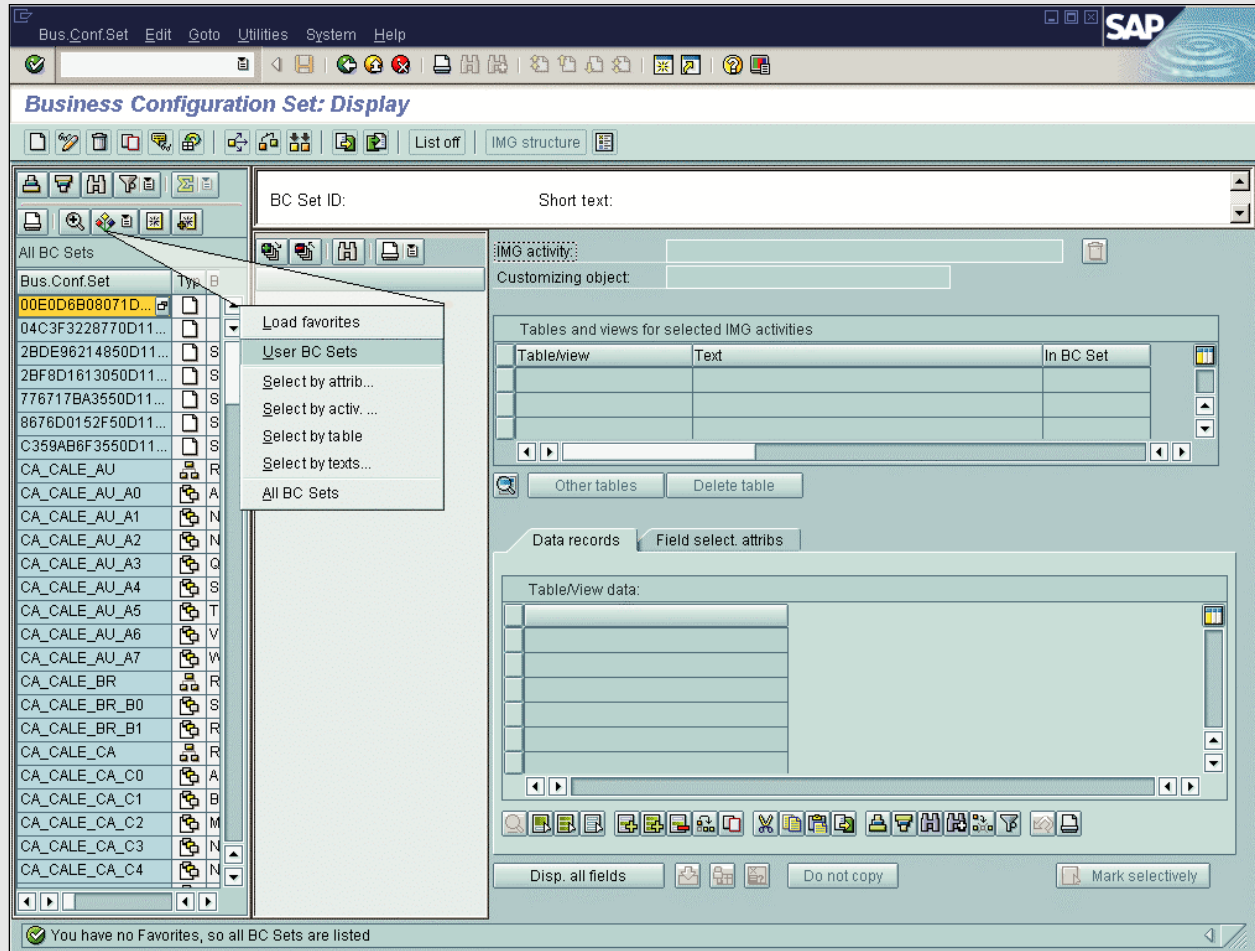
✓ Tip

In general, all activities that are based on table maintenance are compatible with BC Sets, but activities that are not based on this tool can cause problems, one of which is that customizing entries can sometimes trigger generation of tables, reports, etc. These activities cannot be handled by BC Sets because the BC Set activation does not trigger the generation methods in the target system. Any customizing that originates in these activities must be handled with transports and the selection field method because the transport system can start generations after importing customizing values.

I also do not recommend including data from tables in generated BC Sets, because you do not know whether the same tables were generated in the target system before activation.

New in Release 4.6C — Favorite Management!


The difference between the BC Set maintenance transaction in Release 4.5 and Release 4.6C becomes immediately visible upon startup — a “favorite management” section is now displayed on the left-hand side of the screen and the BC Set ID and its short text are displayed at the top of the right-hand side, as you can see in the 4.6C screen shot below.



The transaction code has also changed: in 4.5, the BC Set transaction was called with SCPR2, in 4.6 it is SCPR3 (for those of you who liked the older version better, SCPR2 is still available in 4.6C).

Let me briefly show you how to use the new favorite management feature.

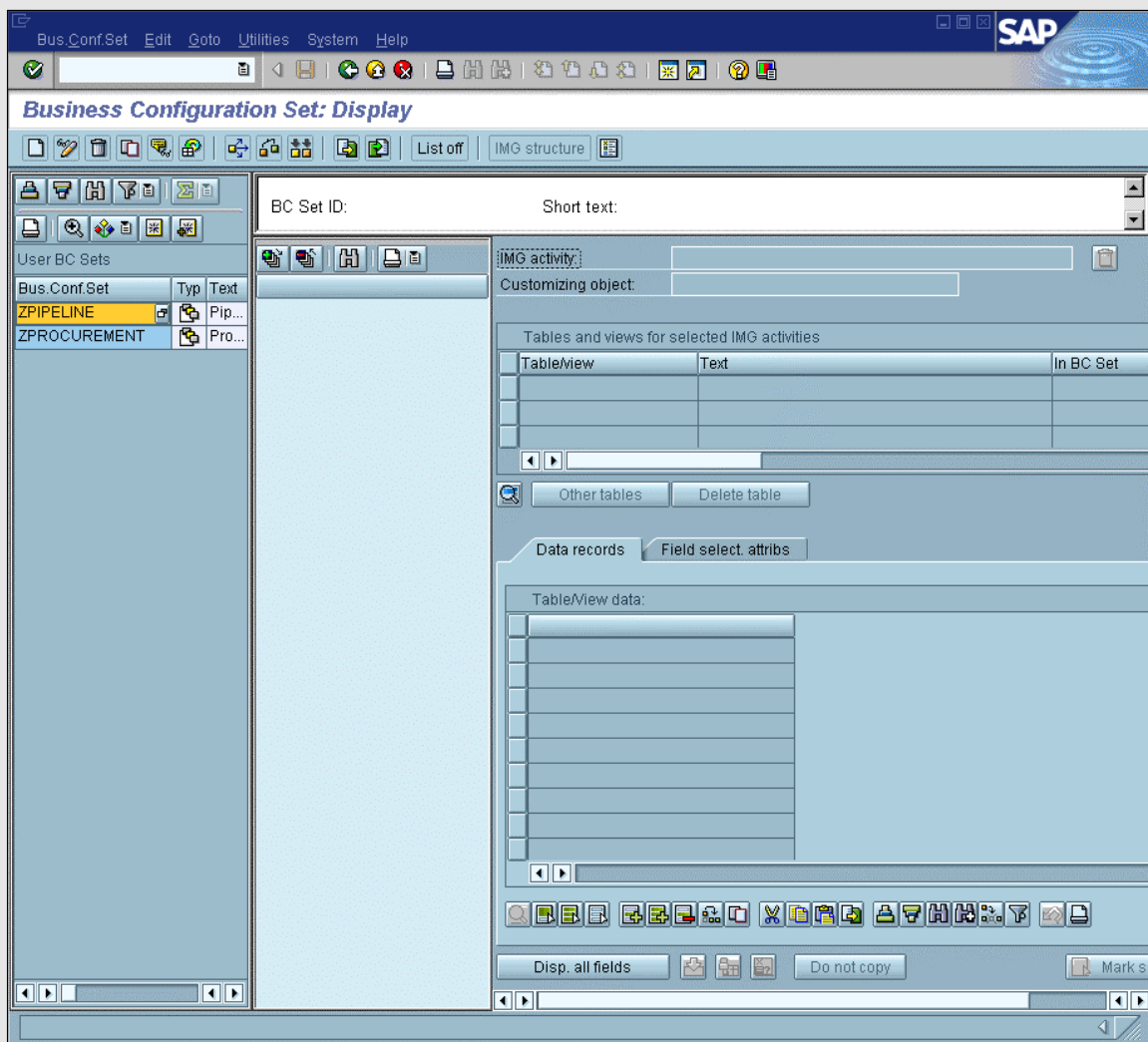
When you call SCPR3 for the first time, all BC Sets available in the system are displayed. As the number of BC Sets will increase considerably over time, it is a good idea to limit the favorites to those sets that you created yourself.


To limit the favorites, click on the  icon in the favorite management section, which brings up the selection menu shown in the previous screen shot.

To limit the display to those BC Sets that you created yourself, select “User BC Sets” from the selection menu. The favorite management section on the left-hand side of the screen now displays a limited list — in this case, ZPIPELINE and ZPROCUREMENT — as shown in the screen shot below.

✓ Tip

*If you want to hide the favorites during BC Set maintenance, click on the pushbutton **List off** at the top of the screen. The favorite list is hidden until you click **List on** or until you re-enter this transaction.*



To save this list as your favorite BC Sets, click on the  icon in the favorite management section. From now on, the BC Set maintenance transaction will always load your favorites list at the beginning.

Essentials of the SAP Testing Facilities

The implementation of a global template at a subsidiary requires thorough testing. The local systems need to ensure that:

- The customizing template delivered by headquarters is correct and works the way it was intended
- The local and the central customizing work hand-in-hand

These tests can be handled with the Test Workbench, which is shipped for free as a part of every R/3 system. The Test Workbench consists of:

- CATT (Computer-Aided Test Tools), which is used to record user input and automatically replay that input in the environment that is to be tested. If the replay works, then the function also works in the new environment.
- A tester transaction called “Perform Test” (transaction STWB_WORK). This tool is basically a worklist that contains all the tests a tester has to perform in the system.
- The Test Organizer, which contains several tools used to organize and manage testing.

Here, I will focus on using the Test Organizer and the tester transaction. Automated tests created and maintained with CATT will not be discussed.

The basic unit in testing is a *test case*. A test case describes how to test a simple function or a specific part of a business process. The SAP testing environment can handle the following test case types:

- Manual test case — This test case contains detailed descriptions of the steps testers are supposed to perform when logged on to the test

system. (This is the test case type I will focus on in the discussion.)

- CATT procedure — This test case contains an automated test created using CATT.
- Function module test — This test case is used during programming to test whether a single program routine works as intended.
- Remote R/2 test — This test case is used to handle test procedures in an R/2 system.
- External application — This test case is used to include tests performed by SAP external test tools like Mercury Winrunner.
- Referring — This test case is used to manage tests located in other R/3 systems.

Typically, a global template defines a lot of business processes that require testing after deployment. As templates tend to contain customizing for quite a number of generic company processes, a lot of test cases are needed, too. To facilitate the management of test cases, the Test Organizer provides *test catalogs*. A test catalog is basically a pool of test cases, a place in the system where you go in order to find out which test cases are available. The test cases inside a test catalog can be organized in folders and subfolders, which you are free to define.

If you want to make use of the test cases — i.e., if you want to start testing for a specific purpose (like the installation of a global template) — you generate a *test plan* from one or several test catalogs. A test plan thus contains a selection of all test cases in the system needed to test for a specific purpose. All test status and result information is stored with the test plan (and not with the test catalog). This is why you keep all test plans even after the testing activities are finished, because they document what you have done. From a technical point of view, a test plan is generated from a set of test catalogs. You can generate multiple test plans from one test catalog, and if you have started testing using a test plan and discover that

some test cases are missing, you can regenerate the test plan without losing the status and result information.

✓ Tip

It is a good idea to organize the test cases either according to the SAP application hierarchy or according to business processes. If you capture the group customizing in BC Sets according to business processes, then it makes sense to build test cases for these processes because this makes testing easier. The subsidiaries can then apply a BC Set in their system, and the test cases that go along with this process are also available in the local system.

If you fear that the number of test cases you need will be too large for one test catalog, you can define several test catalogs in your system to distribute the volume.

Before testing can begin at the subsidiary, the test cases and testers have to be matched — i.e., you have to answer the question of who tests which functions. This is the point at which the subsidiary has to decide:

- Which parts of the template should be tested?
- How many resources are required to complete the testing in a given period of time?
- How do the resources and the test cases have to be matched?

For the matching, the Test Organizer provides the concept of *test packages*. Test packages are those portions of a test plan that are assigned to users — i.e., a test plan is sliced into pieces that can be handled by individual testers. The matching process is technically handled with the transaction “Manage Test Plan” (STWB_2). The testing itself is done with the tester transaction “Perform Test” (STWB_WORK).

This transaction provides the testers with an environment that gives them access to the test cases and their descriptions, and allows them to set a status and enter documentation for each test case.

Both of these tasks will be explained in detail in my next article because they are used by the subsidiary to perform the template testing after the implementation of the template — i.e., the activation of the BC Sets.

In the next sections I will show you the headquarters part of testing⁵ — i.e., how to:

- Create a test catalog and define a catalog-specific structure.
- Create test cases and assign them to the catalog.
- Generate a test plan from your test catalog.

Both test catalogs and test plans can be transported. I recommend that the company headquarters defines a company test catalog and generates a test plan. This test plan, together with the test catalog, is shipped together with the global customizing to the subsidiaries, who can extend or change the test plan.

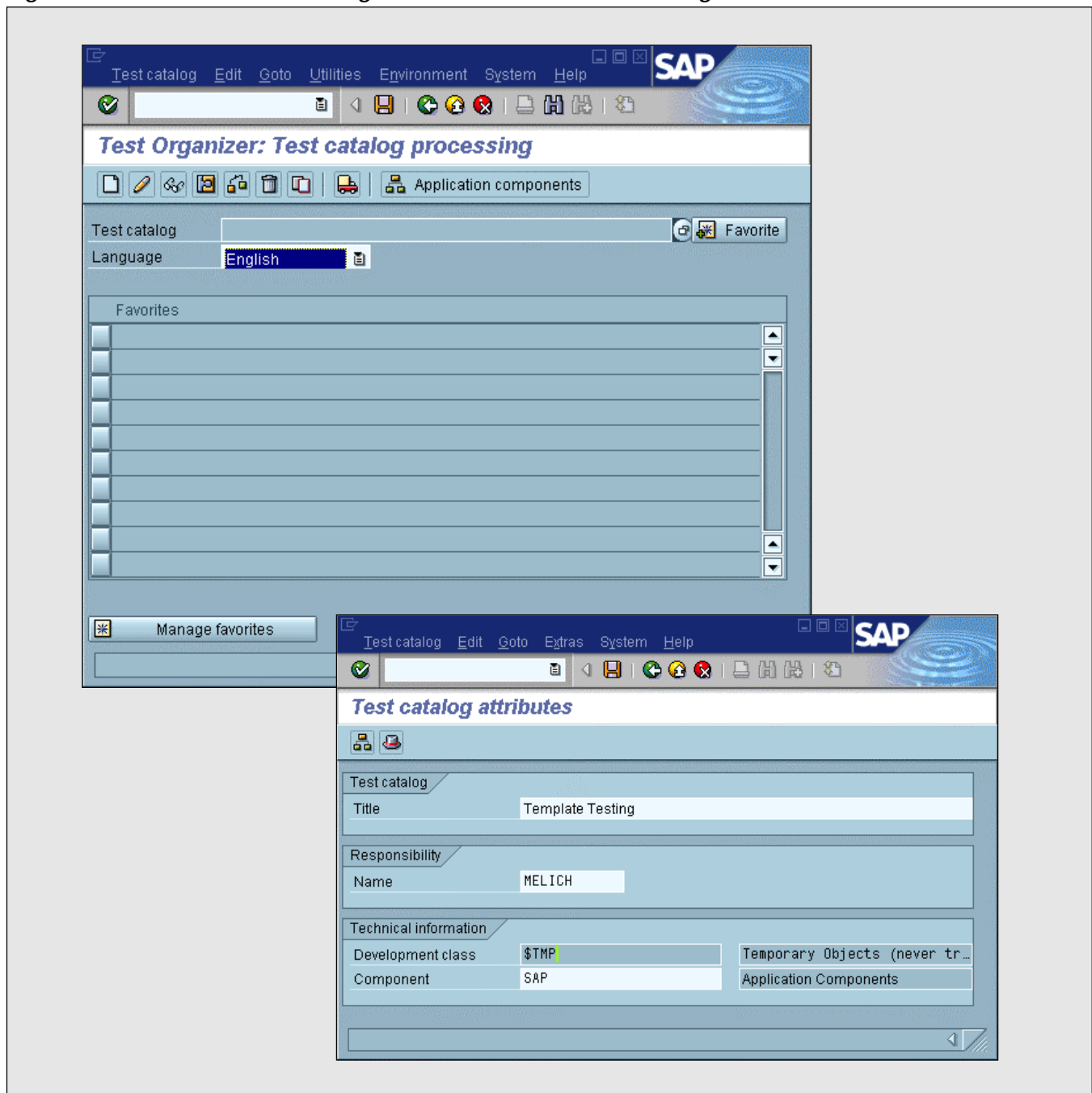
Creating a Test Catalog

The first step that headquarters has to take to prepare the testing of the template at the local sites is to define the overall scope of the template testing — i.e., define for which processes test procedures have to be created — and define how the pool of test cases is structured and ordered within the test catalog to ensure an effective assembly of the test plan.


These tasks are solved with the test catalog maintenance transaction. The steps are as follows:

1. Go to the Test Organizer via the menu path **Tools** → **ABAP Workbench** → **Test** → **Test Workbench** → **Test Organizer** (transaction STWB_1).

⁵ Again, testing at the subsidiaries will be detailed in my next article.

Figure 10 *The Test Catalog Maintenance and Test Catalog Attributes Screens*

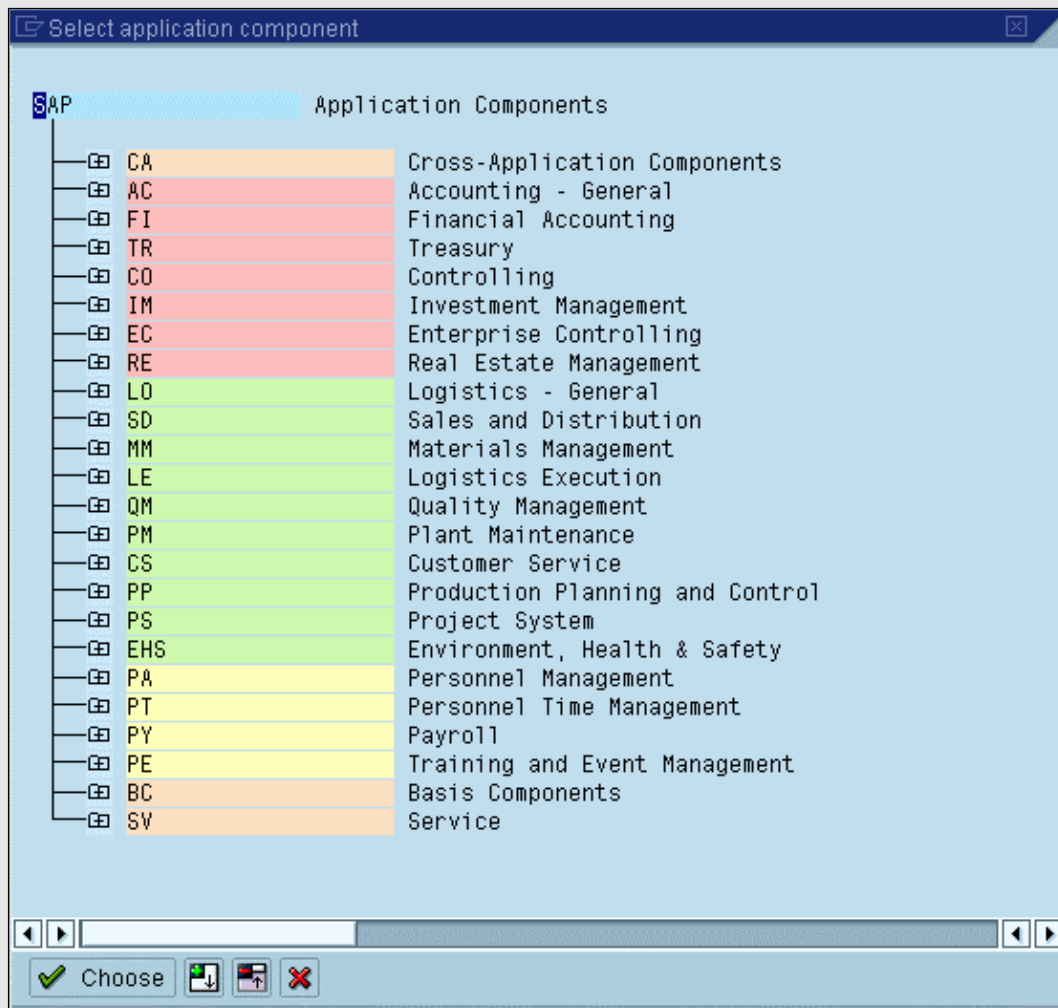
The first screen in **Figure 10** shows the initial screen of the test catalog maintenance transaction.

2. Click on the  icon to create a new test catalog.

The "Test catalog attributes" screen (the second screen in Figure 10) appears. On this screen, you have to enter the title of the test catalog, a development class, and the application component.

Figure 11

Select an Application Component

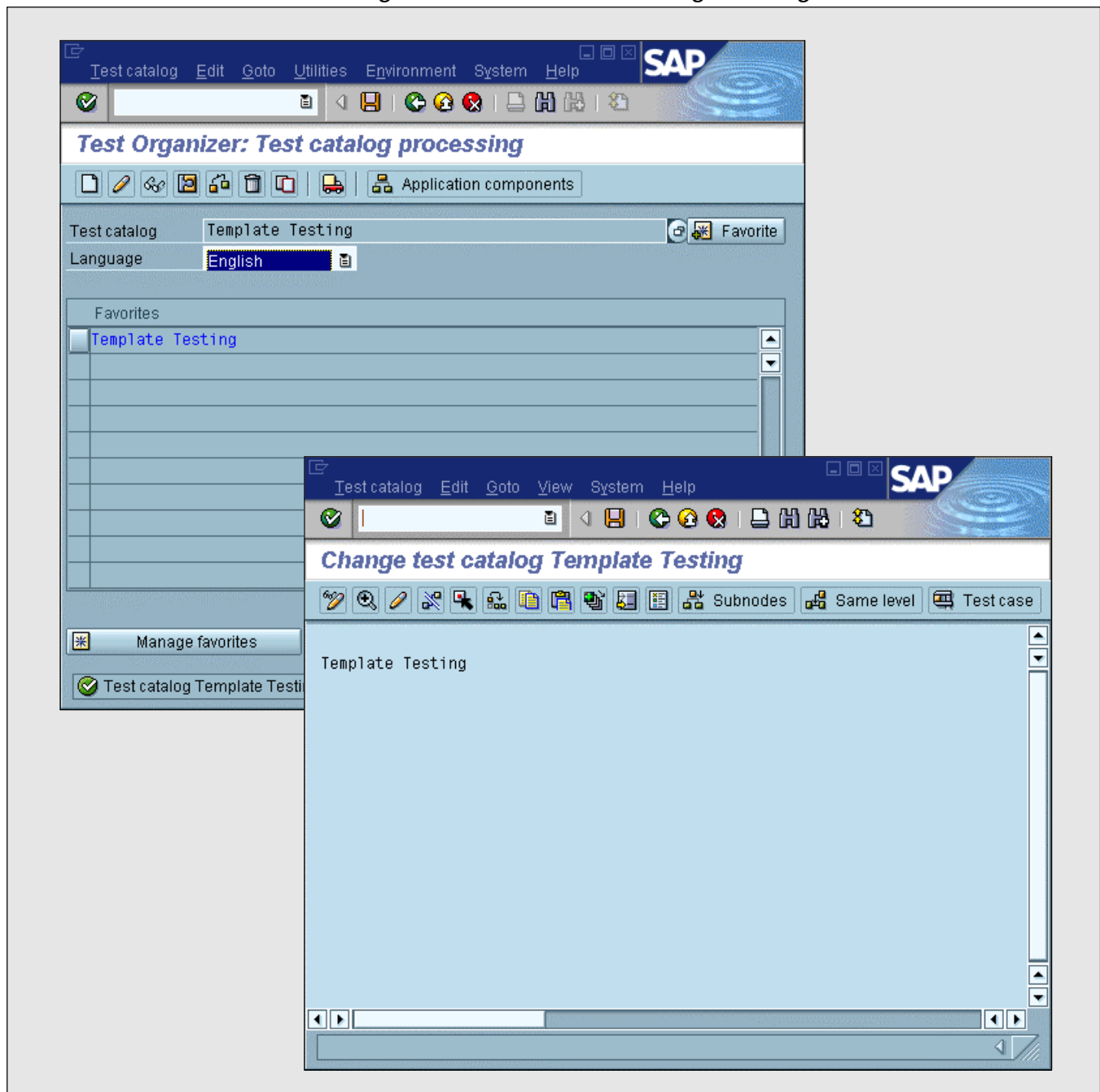


I chose “Template Testing” as the title and “\$TMP” (local object) as the development class because I created these screen shots using an SAP 4.6C test system. At a customer’s site, you would choose a customer development class. For the component you may select one of the components listed in **Figure 11**. I selected “SAP” here because this test catalog is supposed to contain test cases for all components.

The first step to take to prepare the testing of the template at the local sites is to define the overall scope of the template testing and define how the pool of test cases is structured and ordered within the test catalog to ensure an effective assembly of the test plan.

Figure 12

Initial Screen of Test Catalog Maintenance with the Test Catalog Name, and the Test Catalog in Change Mode





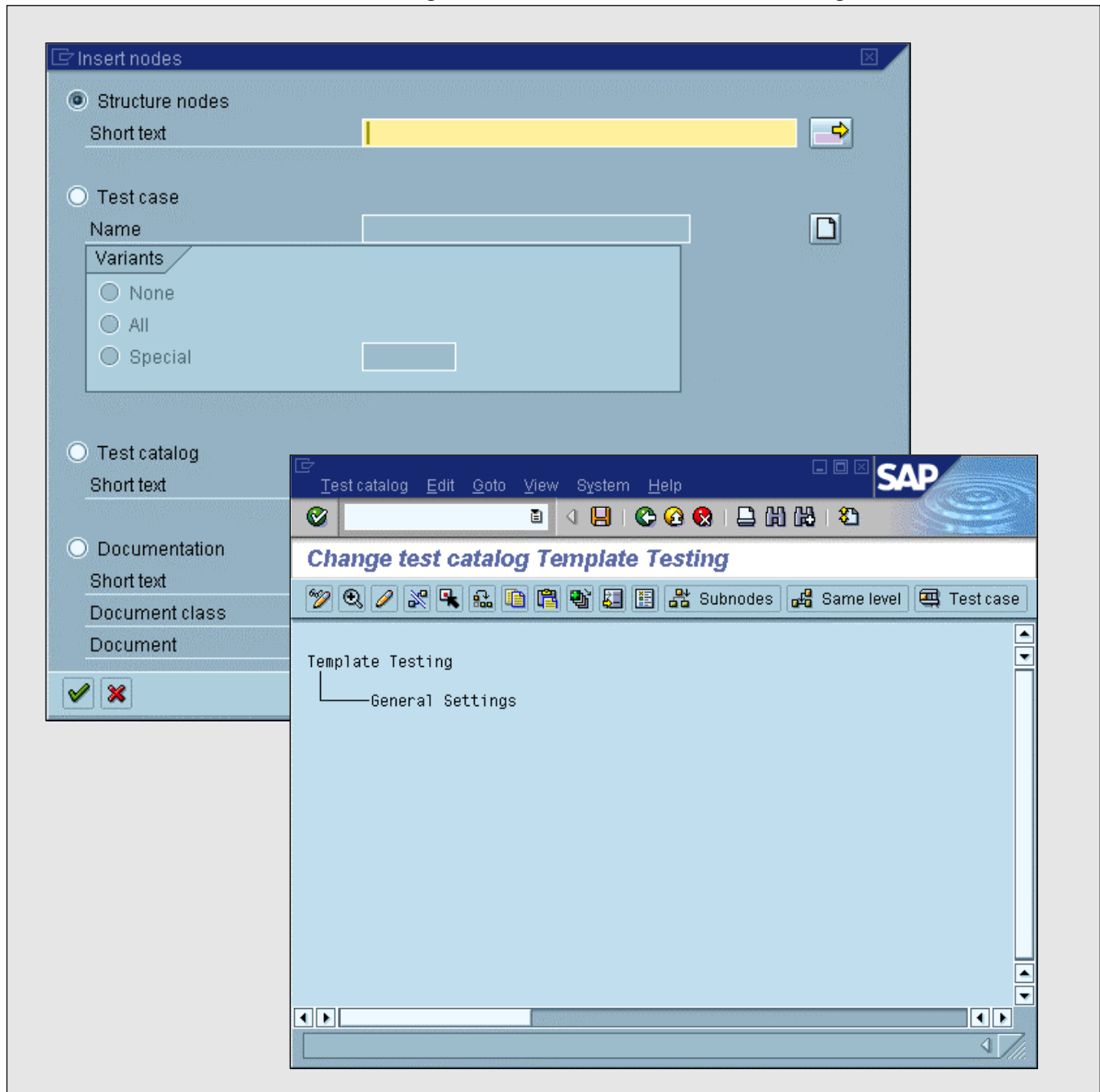
3. Click the  icon, which brings you back to the initial screen, only as you can see in the first screen in **Figure 12**, the name of the test catalog now appears — “Template Testing.” Click the  icon to enter the test catalog in change mode, shown in the second screen in Figure 12.
4. Next, via folders and subfolders, you define the internal structure of the test catalog, which is where the test cases will be stored. Naturally, this structure can always be enhanced or changed. To insert a folder, click the **Subnodes** button, which launches the “Insert

Figure 13

*The “Insert nodes” Screen, and the
“General Settings” Folder Inserted in the Test Catalog*



nodes” screen (the first screen shown in **Figure 13**). Here, select **Structure nodes** and enter a name for the folder — I have called mine “General Settings,” and as you can see in the second screen in Figure 13, this folder is inserted into the test catalog.

Creating Test Cases

The next step is to create test cases with test descriptions that lay out standard testing procedures for the globally defined business function. The steps are as follows:

Figure 14

The “Test case attributes” Screen

Test case attributes: Change test case COUNTRY_DEFINITL...

Test case: COUNTRY_DEFINITIONS

Management data | General data | Restrictions

Header data

Test case: COUNTRY_DEFINITIONS

Title: General Country Definitions

Type: Manual test case

Responsibility

Name: MELICH

Type:

Development information

Development class: \$TMP

Component: SAP

Status: Active

Release: Not released

Temporary Objects (never transpor... Application Components)


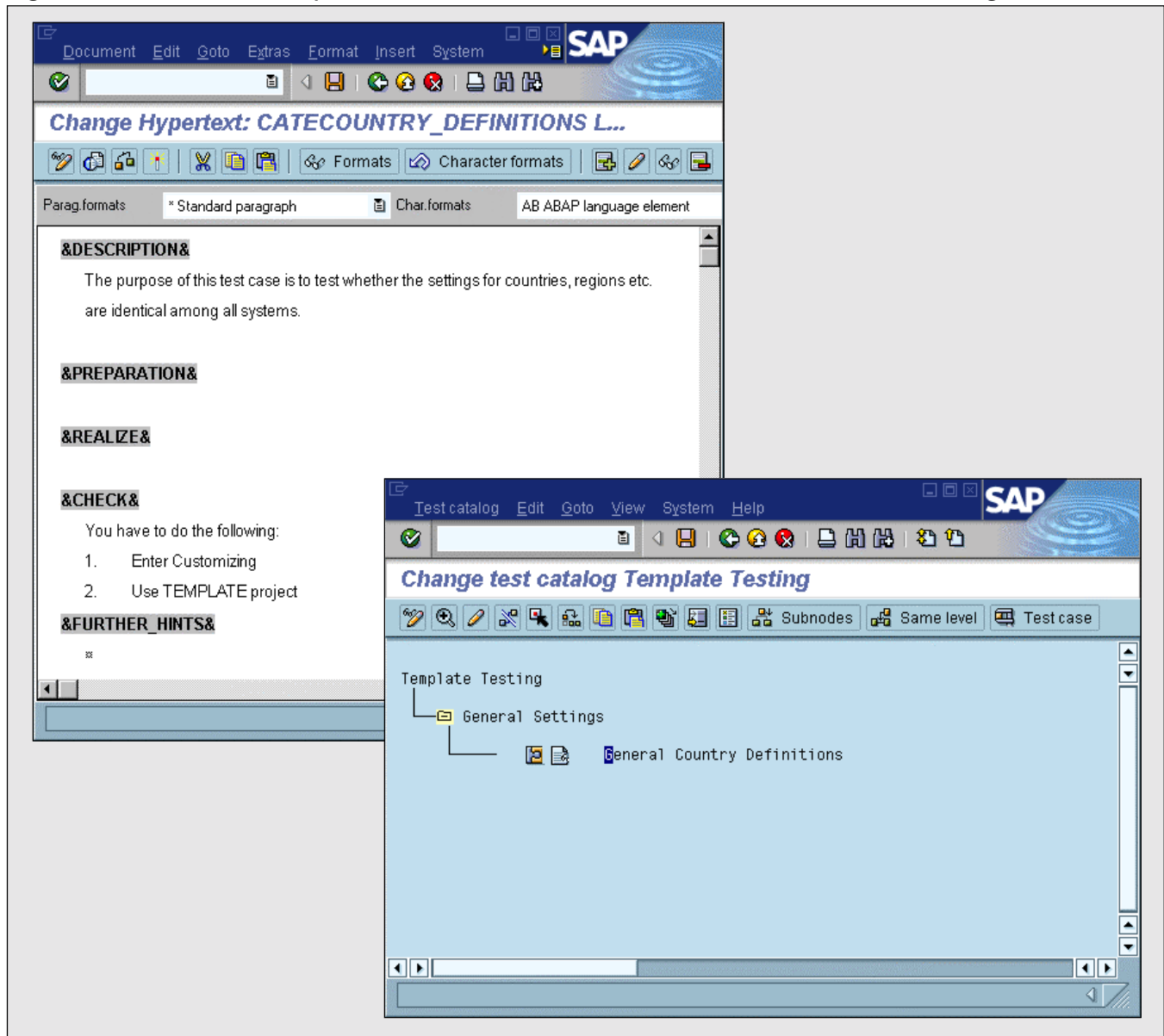
1. To create a manual test case in this folder, position the cursor on the “General Settings” folder shown in Figure 13, and click the **Subnodes** button. This brings you to the “Insert nodes” screen again. This time, select **Test case**. Enter the name of the test case — I have called mine “Country Definitions” — and select “None” for variants, because we do not plan to create a variant for this test case. Then click the  icon, which launches the “Test case attributes” screen (Figure 14).
2. Here you enter the header data of this test case, as well as other information, such as:
 - On the “General data” tab, the time required for testing the function described in this test case and the name of the transaction, report, or function module
 - On the “Restrictions” tab, the dependencies of this test case with regard to SAP release, language, or country, a specific hardware


Figure 15 Add a Description to the Tester, and a Test Case to the “General Settings” Folder



environment, and the use of this test case (individual, application, platform test)

To keep this example as short as possible, I simply enter the title of the test case, “General Country Definitions,” and I leave the test type as is — i.e., “Manual test case.”

3. For a manual test, you have to add a detailed description of what the tester is supposed

to test. To create this description, click the  icon, which launches the first screen shown in **Figure 15**.


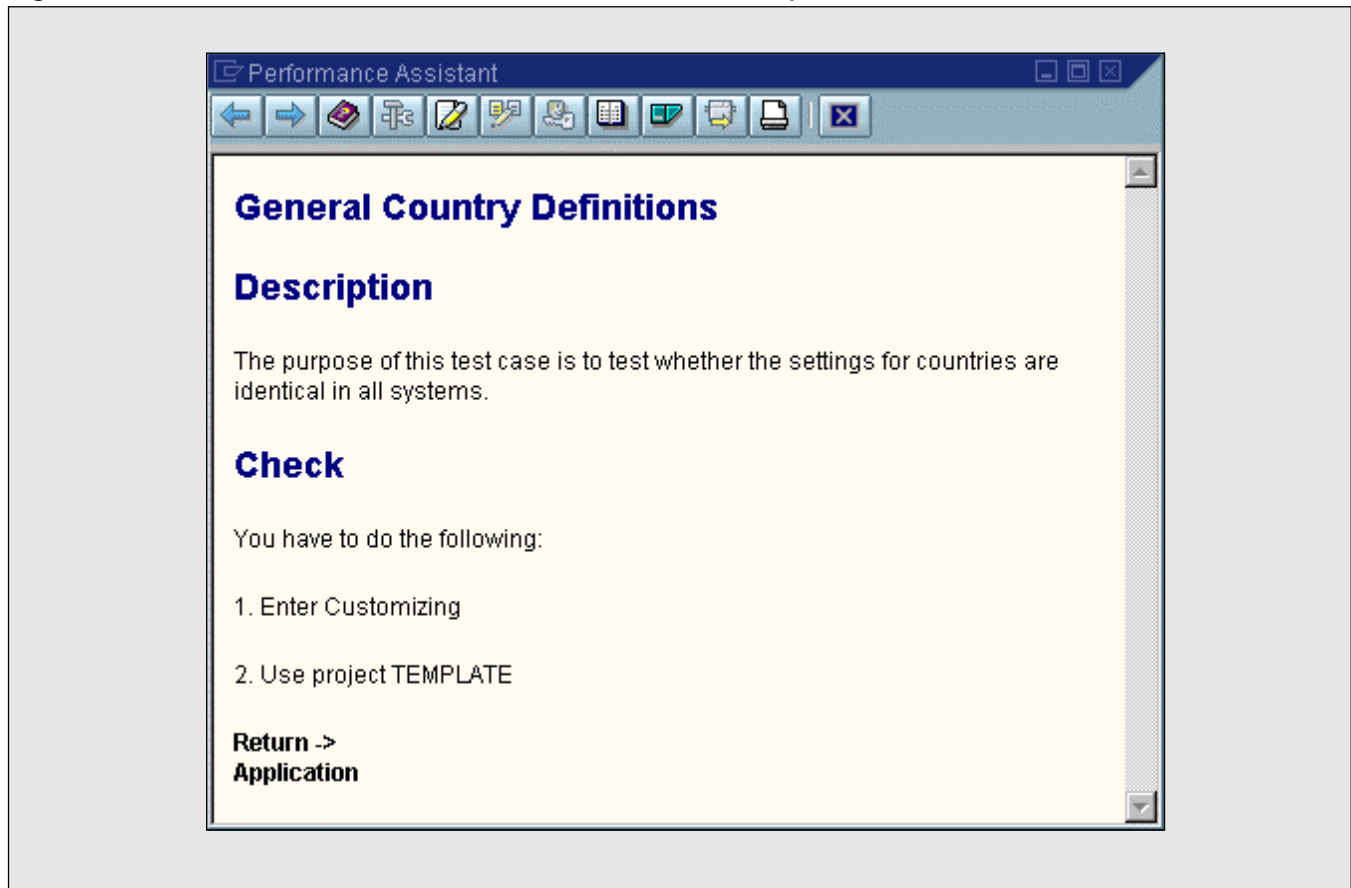
4. Once you’ve created the description, click the  icon twice, which brings you back to the screen originally shown in Figure 13, only now you can see that the test case has been added to the “General Settings” folder (see the second screen in Figure 15).

Figure 16

The Test Case Description



5. In front of the test case are two icons. Clicking the first icon (📄) calls up the test case attribute screen; clicking the second icon (📖) allows you to read the test case description, as shown in **Figure 16**.

Generating a Test Plan

Once the test catalog is established, which contains the superset of all test cases associated with the template, the company headquarters can take the next step — the definition of a test plan that includes at least the minimal scope of what should be tested locally. Predefining the test plan allows the subsidiaries to jump-start the testing procedures after the template implementation. As the content of the test

plan can be changed at any time (even while the testing is in progress), the subsidiaries always have the chance to adapt the testing focus according to the local needs.

In this section, you will see how a test plan is generated from a test catalog. The steps are as follows:

1. Go to test plan maintenance via the menu path **Tools → ABAP Workbench → Test → Test Workbench → Test Organizer → Manage Test Plan** (transaction STWB_2). **Figure 17** shows the initial screen of test plan maintenance.
2. To create a test plan, click the 📄 icon, which brings up the screen shown in **Figure 18**. Enter

Figure 17

The Test Plan Maintenance Initial Screen

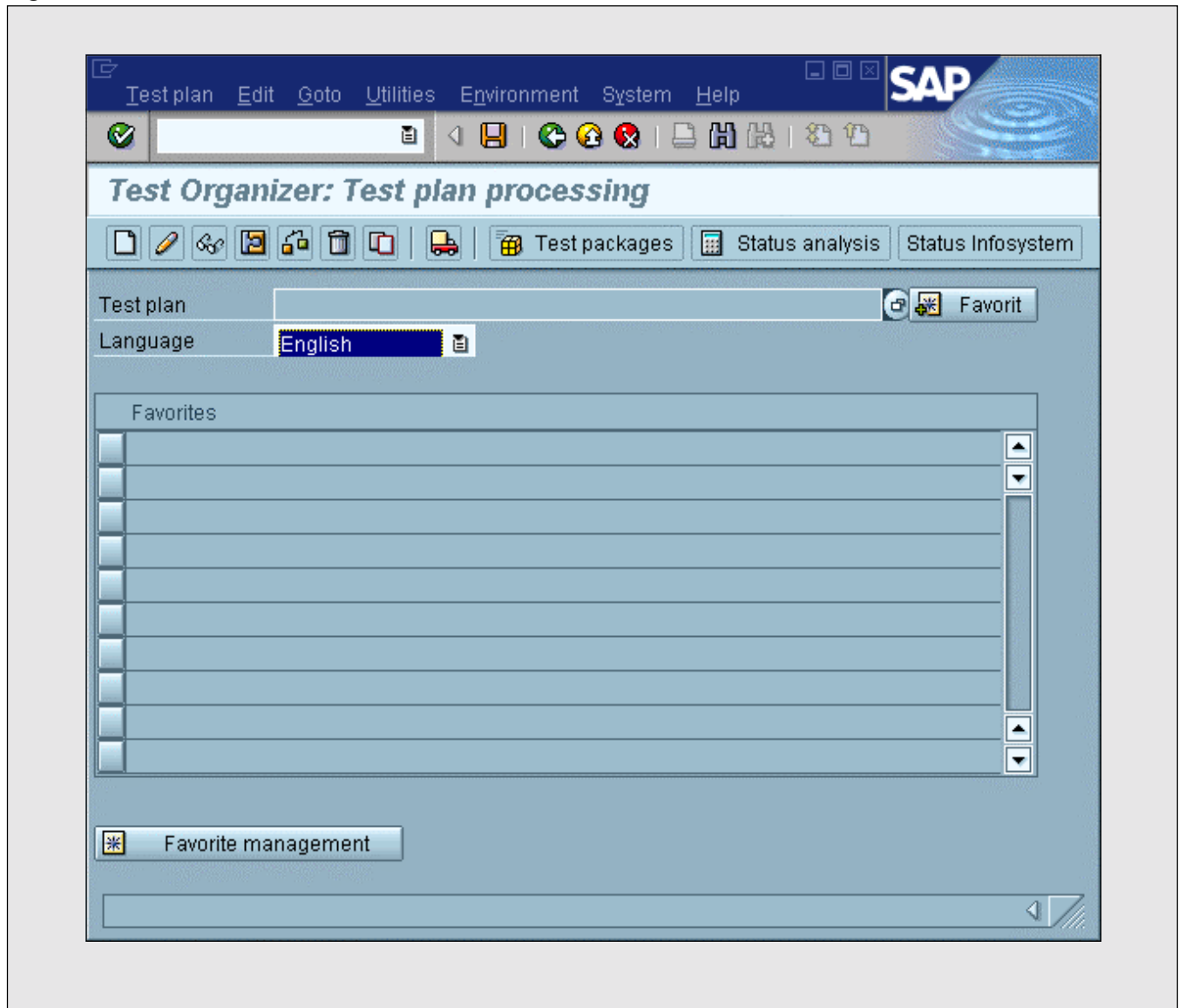


Figure 18

Enter the Name of the Test Plan

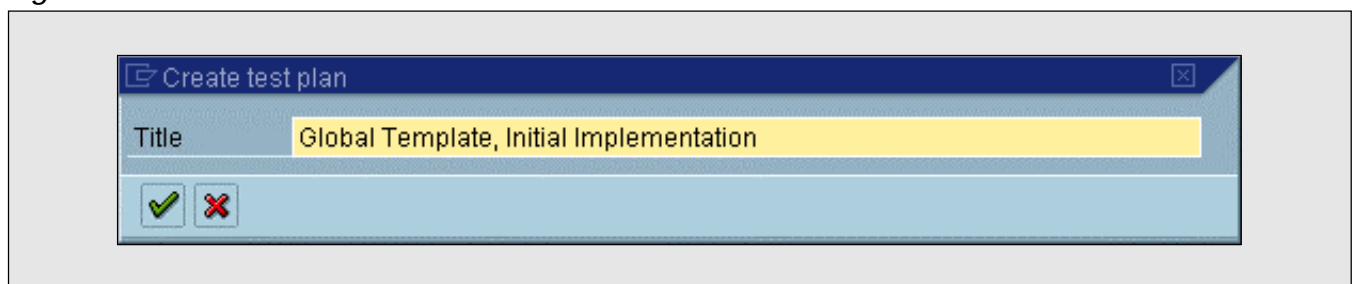
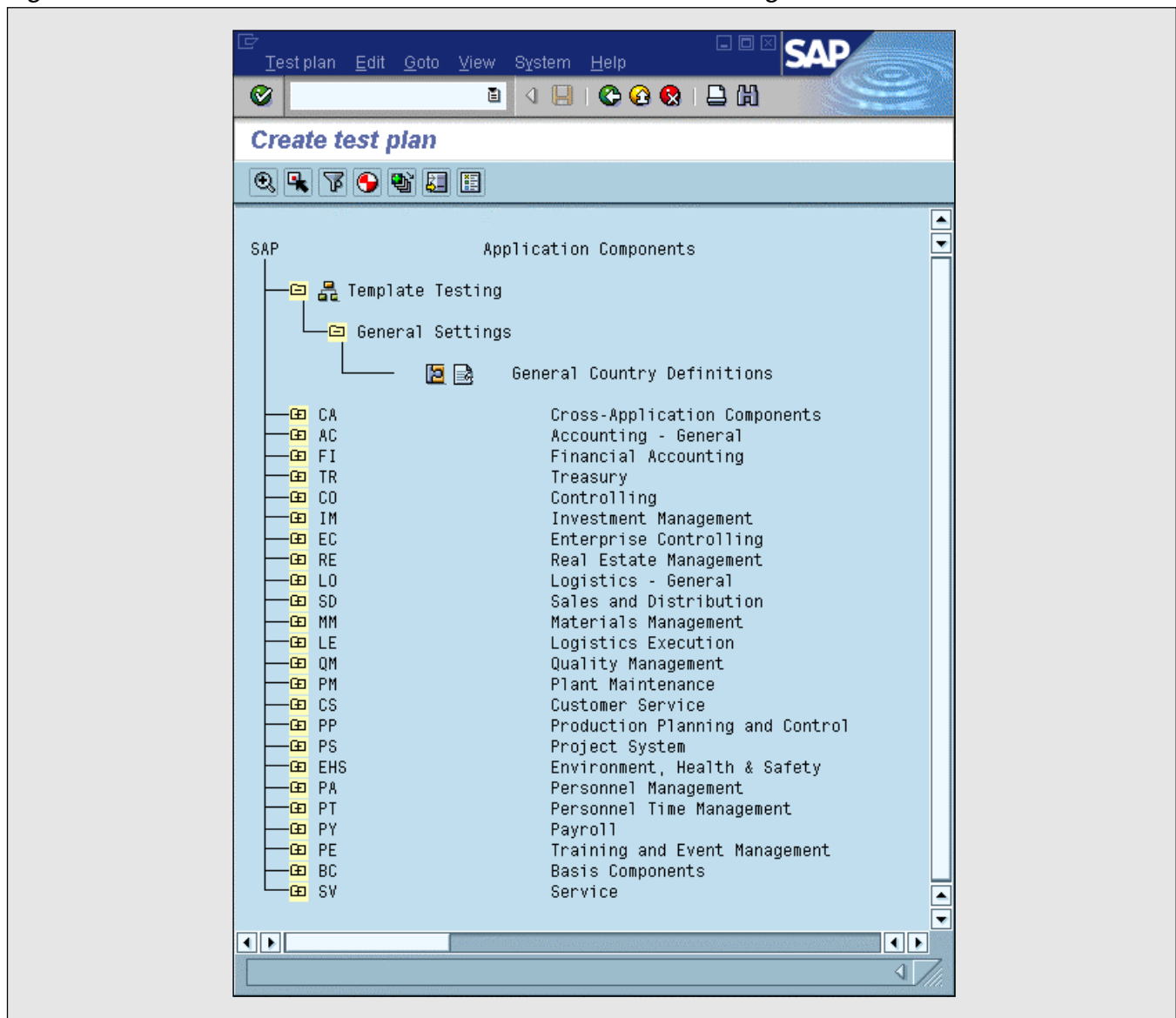



Figure 19

List All the Available Test Catalogs



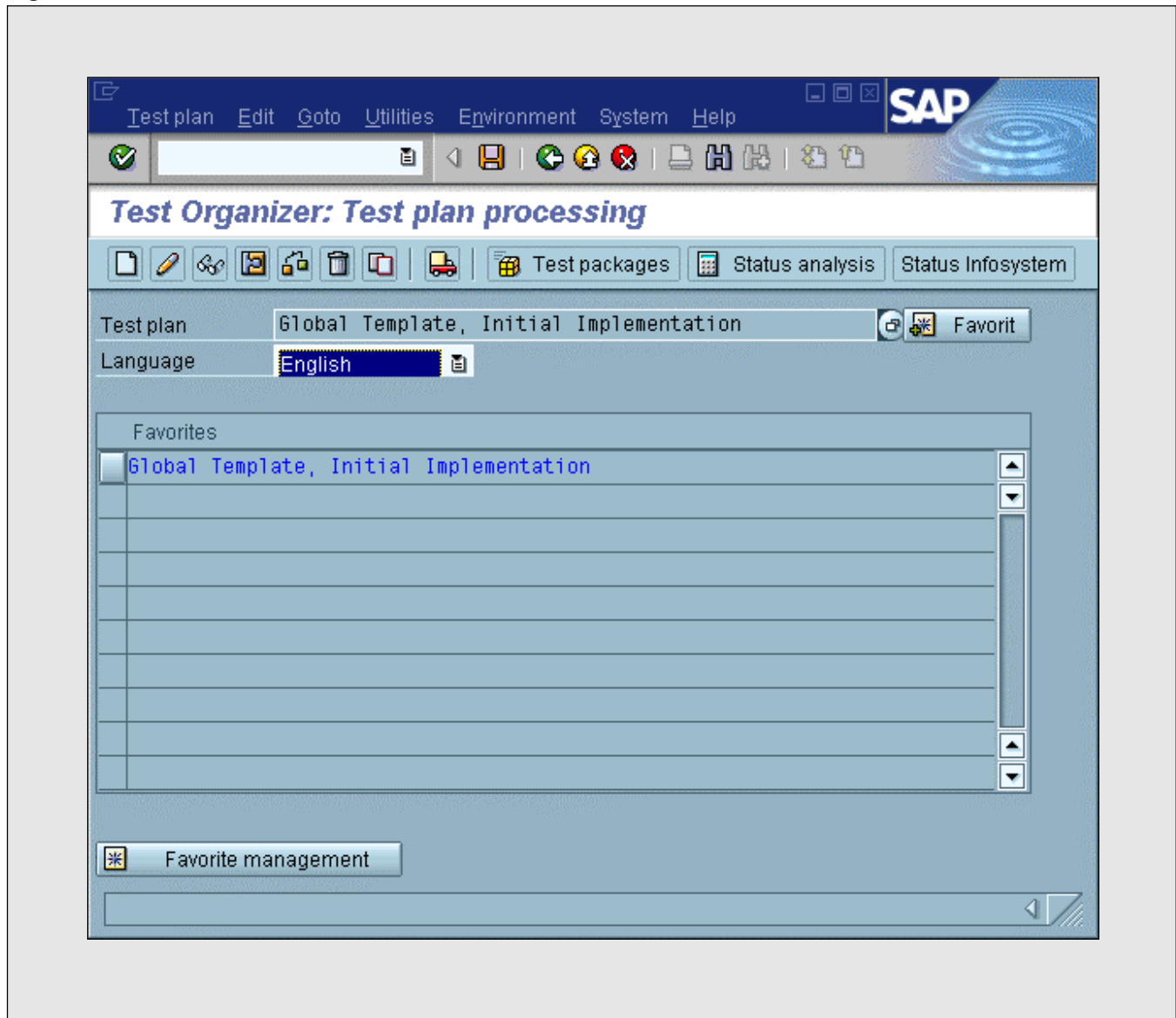
the name of the test plan that you want to create. Here I have entered “Global Template, Initial Implementation.”


3. Click on the  icon, which takes you to the screen in **Figure 19**. Here the system lists all test catalogs that are available in the system. As I did not assign the catalog “Template Testing” to a specific application component, this catalog is displayed immediately under the SAP node.


Selecting the test cases that you want to include in this test plan is easy because:

- You can drill down to the individual test cases in each test catalog.
- You can call up the attributes and the test descriptions for any test case.
- You can select individual test cases as well as folders from a test catalog.

Figure 20 Initial Screen of Test Plan Maintenance with the Test Plan Name



- You can pick from several test catalogs at the same time.
- 4. Here, I will pick a complete folder, so I position the cursor on the folder “General Settings” and click the  icon. The folder and its contents are highlighted.
- 5. If you are done with your selection — i.e., if you picked all test cases and folders that you want to

include in your test plan — click the  icon to generate the test plan. If the system is done, the entry screen of the test plan maintenance transaction is displayed again, only as you can see in **Figure 20**, the name of the test plan now appears — “Global Template, Initial Implementation.”

The generation of a test plan is the last step of the template definition for testing done by headquarters.

Conclusion

I hope that you have come away from this discussion feeling that BC Sets and the Test Organizer are valuable tools for managing the preparation of a global rollout. Once you have captured the group standard customizing according to business functions in BC Sets and created test cases, a catalog, and a plan, the next step is to put all of this information into transport requests and bundle them with add-on developments, process descriptions, etc.

In my next article, the focus of interest will shift to the subsidiary — i.e., I will show you how the subsidiary evaluates the content of the BC Sets, applies it to the local customizing tables, and starts testing. The latter involves the matching of test cases and resources, the running of the test cases, and the evaluation of the overall test progress.

If you want to practice the handling of BC Sets, I recommend you check SAP's course offerings. Our development team has set up a one-day training course that offers insights into the complete BC Set functionality. The course is called ASAP94, "Advanced Customizing Tools," and is available as of April 1, 2001.

For more information on customizing tools, visit <http://service.sap.com/customizing>.

Acknowledgements

I would like to thank Marc Oliver Schäfer and Sabine Reich for checking the correctness of the how-to sections in this article, and for their continuous proofreading.

Dr. Matthias Melich studied English Literature at the University of Rochester (New York), and Mathematics and English Philology at the University of Cologne. He also received his Ph.D. in Computer-Based Language Learning at the University of Cologne in 1993. He joined SAP in 1995 as a member of the Common Objects and Modeling team, and a year later joined the Archiving Solutions team. Since 1998, Matthias has been responsible for Product Management of Customizing Tools. In February 2000, the Customizing Tool Development team became an independent department called Advanced Implementation Solutions (AIS), whose tasks comprise the technical backbone for SAP's hosting offerings and generic customizing concepts and tools for the mySAP.com system. Matthias is now Product Manager of the AIS department. He is also the author of "Computer-Based Training for R/3 Archiving" and the SAP DocuSet "Data Modeling with the Data Modeler." He can be reached at matthias.melich@sap.com.