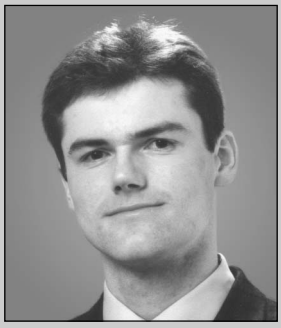
Extending SAP Business Workflow with Web Forms

Rainer Weber



Rainer Weber joined SAP AG in 1994, and since then, he has been working in the development of SAP Business Workflow. His principal areas of focus are interfaces to external systems, among others, and also Web interfaces. Rainer can be reached at rainer.weber@sap.com.

Imagine a manager at your company who does not interact directly with SAP transactions, and needs a simple forms-based interface for her daily work. This manager has a Web browser on her desktop, but not the SAP GUI for Windows. Now imagine that some crucial step in an SAP Business Workflow requires a decision of this manager. With Web forms (introduced in Release 4.0A of the SAP system), this manager can be brought into the fold of SAP Business Workflow by presenting an approval form. She can view and approve the work item directly from her browser.

Now picture the employees who are charged with entering SAP data, but who have not yet been trained on powerful SAP transactions. Here again, Web forms provide a solution. These employees could be provided with a Web form as their data entry interface. Instead of an SAP transaction, they would see a Web form, which is made up of a relatively short list of fields and a button for submitting the Web form, and thereby starting a workflow. And imagine you can radically reduce the development time by even generating the form, in both cases.

In this article, I will use the customer master data demo that has been delivered with the SAP system starting with Release 4.5a[1] to help teach you about all the little steps that are required to build these types of workflow scenarios. To create a customer master data record, in this scenario, a user simply links to a page on the company intranet, then fills out a Web form. This Web form automatically sets a workflow in motion that directs the completed form to another user who can verify the contents of this form-based work item and can grant approval for this record to be input as a new customer master record. If approved,

[1] The workflow used is WS20000102. It is not a full-fledged business example, but rather a demo designed to showcase the basics of leveraging Web forms for workflow.

a BAPI call writes the customer record to the appropriate database table. (This happens as a background step in the same workflow.) If the request is denied, or if writing the customer data to the database fails, an e-mail is automatically sent to the originating user. As is the case with the user who initiates the workflow, the user who receives the work item for approval does *not* have to go to his or her Business Workflow using SAP GUI for Windows to participate in the workflow. The approval process is done entirely from the Web browser. Familiarity with SAP transactions is not required.

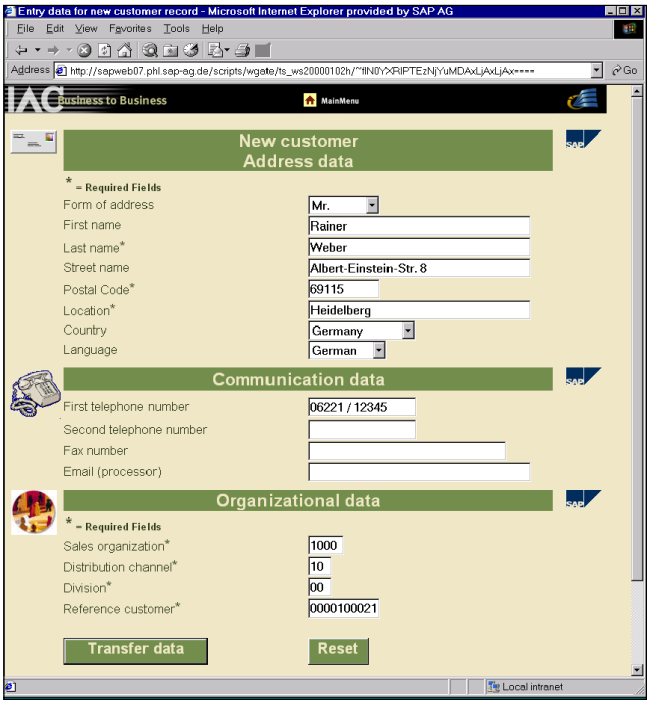
Building a Web Form Application — An Overview

Web forms do not *introduce* Web-enabled application support to the SAP system. Let's give credit where credit is due. SAP Internet Transaction Server (ITS) and SAP@Web Studio are the products that have made this possible. Web forms do not introduce workflow either. Workflow was part of the Release 3.0A offering. The Web Business Workplace followed in Release 3.1G.

Working in conjunction with ITS, SAP@Web Studio, and the Web Business Workplace, what Web forms *do* introduce are:

- An easy-to-use ability to trigger an SAP Business Workflow (e.g., submitting a customer record entry form that internally starts a workflow) from a Web browser, as shown in **Figure 1**.
- The ability (again, from a Web browser) to participate within an SAP Business Workflow with a form-based interface. For example, the started workflow generates a work item in the inbox of a person that checks the newly entered customer record data, telling him he should approve the request — by processing it he sees the same (or a variant) of that form, and may press buttons to approve or reject it. This is shown in **Figure 2** and **Figure 3**. Figure 2 shows the work item in

Figure 1 Customer Master Data Web Form That Originates the Workflow



SAP Internet Transaction Server (ITS)
ITS adds an HTML-based user interface to SAP applications. This runtime environment acts as a gateway between the HTTP server and the SAP system application server such that a user can access target SAP applications via a standard HTML browser and does not require the SAP GUI for Windows.

SAP@Web Studio
SAP@Web Studio is the development tool for the ITS runtime environment. It allows you to create, manage, maintain, and publish service descriptions and HTML templates. This is the tool you use for the development of the HTML user interface. It offers wizards for the generation of ITS services and HTML templates and an offline preview mode for rapid HTML design. This visual development environment is integrated with the SAP Change and Transport System and contains publish-and-run functionality.

Web Business Workplace
The Web Business Workplace (referred to as the "Web inbox" in earlier releases) is a sample Internet Application Component (IAC). It allows Web browser users to view and handle inbox entries (office mails and work items).

Figure 2 A Work Item Is Created in the Inbox for the Request

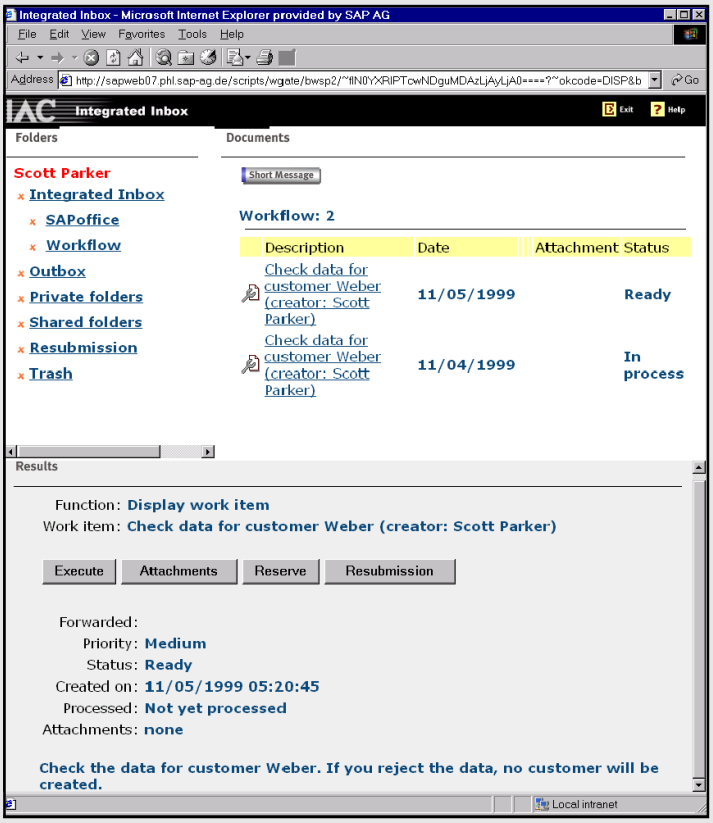
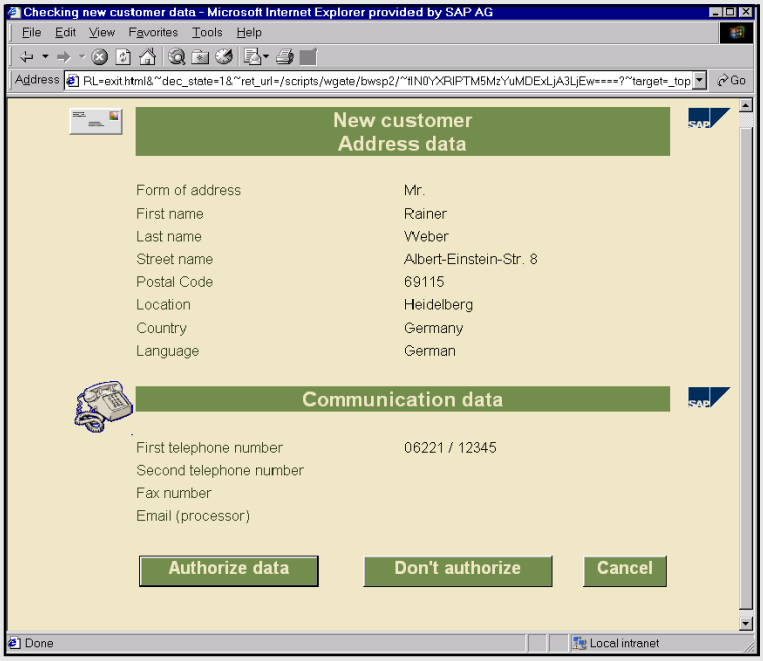


Figure 3 The Approval Form for Processing the Request (Work Item Execution)



the Web Business Workplace. Figure 3 shows what happens when you press the "Execute" button displayed in Figure 2.

You can think of a Web form as a representation of a container in the sense of SAP Business Workflow. It holds the data that is typically passed between a user and the workflow. Data is always taken from the container, processed, and then written back to the container. If required, the actual SAP business objects are created (or modified) in subsequent steps in the workflow, typically background steps, which do not involve a dialog.

You can think of a Web form as a representation of a container in the sense of SAP Business Workflow. It holds the data that is typically passed between a user and the workflow. Data is always taken from the container, processed, and then written back to the container.

In our customer master data example, the user calls a Web form and enters the data. Like all Internet Application Components, Web forms all have an associated ITS service, or just "service" for short, which, along with a service description, includes HTML templates, similar to the way an SAP application is characterized by a transaction and screens. (The HTML templates map the SAP screens to the Web. They define the mapping of objects of an SAP screen to objects in an HTML document, such as input fields and push-buttons; thus merging live SAP data into an HTML page.) And, just like a transaction, the service contains several kinds of attributes — like, which SAP system is used, how a connection is to be established with that system, and the SAP transaction that is to be called — which are captured in a service file.

Much of the work that is required to create a Web form is automated, as you will see in the sections that follow. What is *not* automated, and never could be, are the baseline design decisions. *You* need to decide which fields should appear on the form and how that form should be laid out.

Start to finish, what does a developer need to do in order to create a Web form that either starts a workflow or executes a work item? **Figure 4** details the steps you would take to create a Web form that starts a workflow. **Figure 5** details the steps you would take to create a Web form that executes a work item. There are quite a few steps to do, but they are just small ones, each one requiring just a few minutes, except for the last step — beautifying the templates. The time spent here depends on how much design effort you want to invest. All in all, there is not a great deal of difference between Figures 4 and 5. (I've highlighted the few areas in Figure 5 where the paths for creating a Web form to start a workflow versus a Web form that supports a work item execution diverge. The shaded areas that appear in Figure 5 are identical to those shown in Figure 4).

Basically, while in the SAP system, you define a workflow (or a standard task for work item execution), a transaction to start it, and its corresponding screens, then switch to the SAP@Web Studio development environment, where you define an ITS service and HTML templates for those SAP screens. Here, you will find that the Web Studio wizards do most of the development work for you, automatically generating an ITS service and HTML templates. You also employ SAP@Web Studio to generate the rough (very rough!) layouts of your HTML templates. Don't rely on this tool to beautify these templates. There are ample HTML editing tools out there designed expressly for this purpose.

In the sections that follow, I'll walk you through the steps you would take to create a Web form that starts a workflow, and I'll point out what would be different if you were creating a Web form to execute a work item.

Figure 4 Creating the Web Form That Starts the Customer Master Data Workflow

Where	Activity	Description
SAP System	Define a workflow with a "suitable" container.	In this step, you define all the data (i.e., import container elements) that needs to be passed to the workflow. In our customer example, this would be the name and address information that is used to create the customer master record. It is up to you to decide whether or not you want to use all the fields in the corresponding SAP table or just a subset of those fields.
	Generate a Web-compatible SAP transaction and a module pool (program) from within the task transaction.	This is a generic step that is the same for all Web forms, regardless of whether it supports a workflow start or work item execution. What is specific to our customer example is that the module pool would contain an application-specific screen for the input of customer master record data. Those data fields are generated by the workflow container import elements defined in the preceding step.
SAP@Web Studio	Use the Web Studio's Service Wizard to generate an ITS service.	Here, you determine in particular which transaction is called. In our example, we would be specifying the transaction we generated in the previous step.
	Use the Web Studio's Template Wizard to generate the HTML templates.	This step generates the Web counterparts of the SAP screens that were specified in the second step. In our example, this step would generate the HTML template for entering the customer master data.
	"Publish" the ITS service and HTML templates on the ITS.	"Publishing" the ITS service and HTML templates means that you establish the connection between the definition side and the runtime side of your Web form application.
Web Browser	Start the Web form with the URL to verify its operation.	Enter the customer data in the screen, and press the submit button. If everything proceeds according to plan, you should get a message that a workflow has been started under a particular ID.
HTML Editor	Beautify your HTML templates!	At this point, the elements on your Web form's screen are displayed in various shades of gray. And there are no graphics. So employ your favorite HTML editing tool, or better still, enlist the assistance of an experienced designer, to beautify the Web form's screens and to make them more appealing and intuitive to users.

Figure 5 Creating a Web Form That Executes a Customer Master Data Approval Work Item

Where	Activity	Description
SAP System	Define a standard task with a "suitable" container.	All customer data fields are read-only. Here, we add an element, such as ProcessingState, to capture whether the customer master data record has been approved or rejected.
	Generate a Web-compatible SAP transaction and module pool (program) from within the task transaction.	This is a generic step that is the same for all Web forms, regardless of whether it supports a workflow start or a work item execution. What is specific to our customer example is that the module pool would contain an application-specific screen for checking the master record data. Those data fields are generated by the task container elements defined in the preceding step.
	Insert form step in Workflow.	Insert the approval standard task into the workflow (in our example, the customer master record workflow we defined before).
SAP@Web Studio	Use the Web Studio's Service Wizard to generate an ITS service.	Here, you determine which transaction is called. In our example, we would be specifying the transaction we generated in the second step.
	Use the Web Studio's Template Wizard to generate the HTML templates.	This step generates the Web counterparts of the SAP screens that were specified in the second step. In our example, this step would generate the HTML template for entering the customer master data.
	"Publish" the ITS service and HTML templates on the ITS.	"Publishing" the ITS service and HTML templates means that you establish the connection between the definition side and the runtime side of your Web form application.
Web Browser	Use the Web Business Workplace to verify the operation of your Web form-based work item.	Run the workflow such that it generates a work item in your Web Business Workplace, and execute that work item. The Business Workplace knows the URL for executing the work item. This URL is implicitly called when you execute the work item, which makes testing very easy.
HTML Editor	Beautify your HTML templates!	At this point, the elements on your Web form's screen are displayed in various shades of gray. And there are no graphics. So employ your favorite HTML editing tool, or better still, enlist the assistance of an experienced designer, to beautify the Web form's screens and to make them more appealing and intuitive to users.

Development Activities in the SAP System

When creating a Web form that starts a workflow, the first thing to do is to define a workflow. (When creating a Web form that will execute a work item as part of an established workflow, you define a standard task.) You simply define container elements for all the fields that you would like to appear on the Web form.

If you have designed conventional SAP Business Workflows, you will find that the Web form-based workflow calls for a bigger container. With conventional SAP Business Workflows, you just pass object references in the container. You have access to all the attributes of these objects, or you can process the objects.

That's not the case with Web forms. A container that is associated with a Web form-based workflow does not use object references. Remember, the actual SAP business objects don't get created (modified) right away. Instead, we put the flat data into the container and create (modify) the business objects via subsequent background calls if needed.[2] Because the number of database records stored is the number of form fields multiplied by the number of instances (work items), we recommend that you do not use Web forms for complicated activities that would require very large forms.

[2] This is a key difference between a workflow started by a Web form and a workflow started as a typical SAP Business Workflow. Conventional SAP Business Workflows are usually triggered by an event, and with that event you hand off data to the workflow as an object reference. With a Web form, the data is passed in "flat" form. That is, for each field on the form, you will have a corresponding container element of that size. For an object, you would just pass the object reference and have all the object's attributes available. If you would like to pass an "object" in a Web form, you can only pass the attributes of the object, which you would have to resolve earlier. Only "flat" container elements (referring to data dictionary structure fields) are supported. This is why Web forms work well for situations, like our customer master data example, where a small amount of data is being passed, and do not work well when large amounts of data need to be passed.

For our customer master data records example, all the fields that should later appear as input fields of a form for starting a workflow are defined as import container elements. In our customer master example, these are container elements for the name and address data. The binding is done implicitly by the name of the container element and the name of the form field. So, in fact, no binding is defined explicitly.

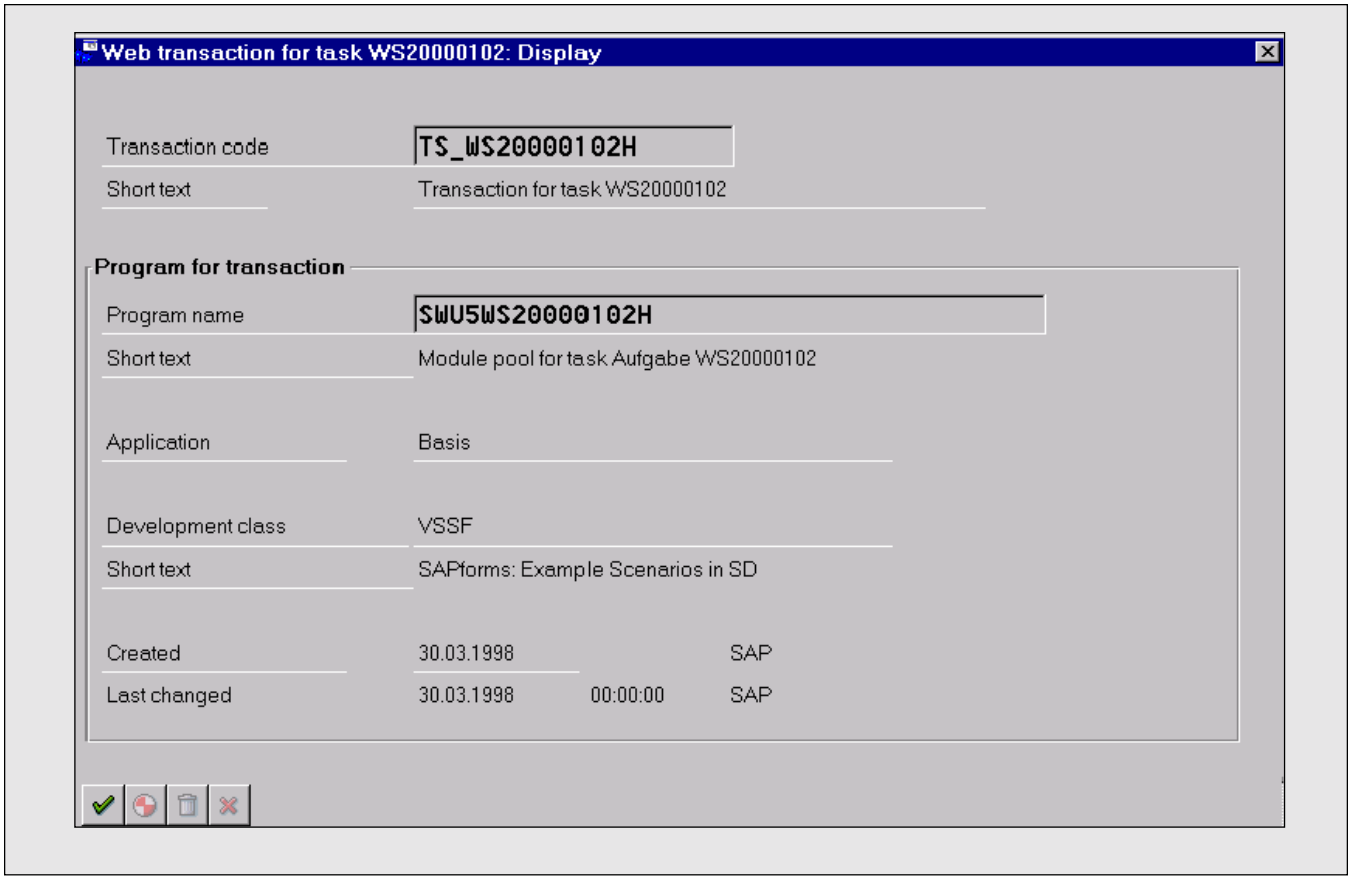
For a work item execution Web form, you need to refer to the object method FORM.HTMLProcess in the task, and then select the flag "Executable on Internet." The FORM.HTMLProcess method generically reads the container elements into the corresponding form fields, and after processing the form, updates the container with the form fields.

In our example, all the customer data fields are read-only fields, and the ProcessingState container element is written with "A" for "Approved" or "R" for "Rejected" when executing the work item. Selecting the "Executable on Internet" flag is the means by which you introduce the Execute button in the Web Business Workplace.

When creating a Web form that starts a workflow, the first thing to do is to define a workflow. (When creating a Web form that will execute a work item as part of an established workflow, you define a standard task.) You simply define container elements for all the fields that you would like to appear on the Web form.

After you have defined your task with a container that has all the container elements you want to use as form fields, you generate a Web transaction for that task by pressing the "Generate" button in the Web

Figure 6 Generating a Web Transaction and Module Pool for the Task



transaction dialog in the task menu. The Generate button also generates the underlying module pool (program). This is shown in **Figure 6**.

The module pool has four screens:

- **Screen 50** is used as a technical screen. It is used for parameter passing (e.g., the work item ID is transferred from the calling service, usually the Web Business Workplace) and is usually processed in the background, so you will never see it, except in testing situations.

- **Screens 100** and **150** are what you would think of as being "Web forms" (at least the SAP system counterpart for it). These screens contain fields for all the container elements and a "Transfer" button for sending the form to the SAP system. Screens 100 and 150 (shown in **Figure 7** and **Figure 8**) look similar — screen 100 is used for work item execution, and screen 150 is used for workflow start. (Each generated transaction is capable of doing both things, though usually only one or the other thing will be used: you just start a workflow, and execute a work item for a standard task.)

After you have defined your task with a container that has all the container elements you want to use as form fields, you generate a Web transaction for that task by pressing the "Generate" button in the Web transaction dialog in the task menu. The Generate button also generates the underlying module pool (program).

Figure 7 The Work Item Execution Screen (Screen 100)

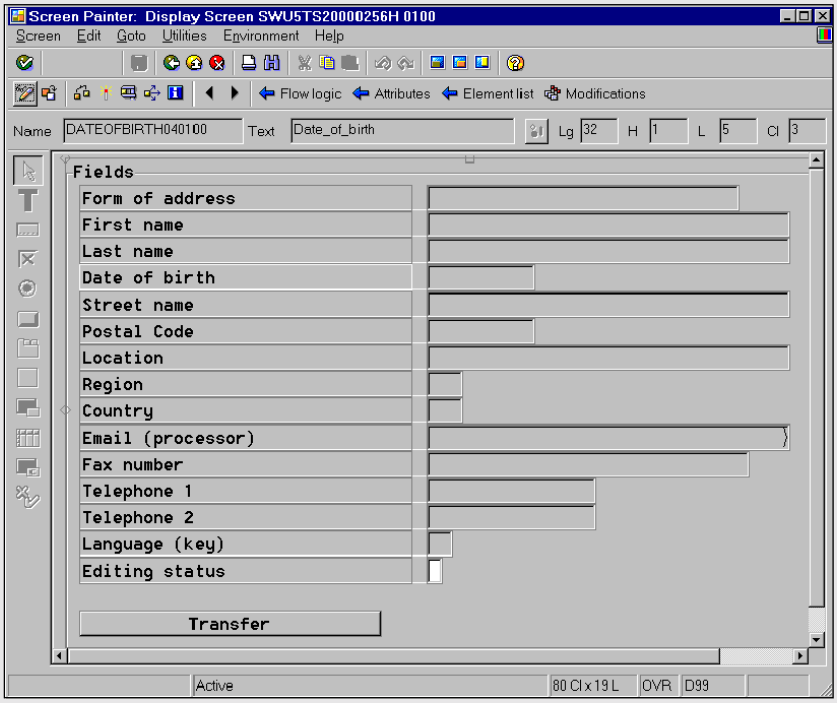


Figure 8 The Workflow Start Screen (Screen 150)

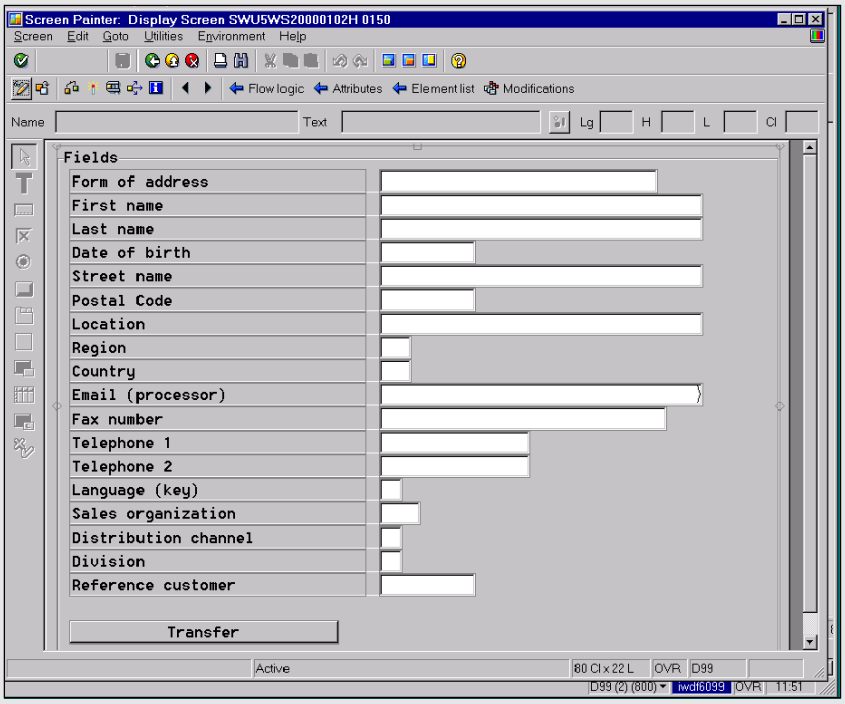
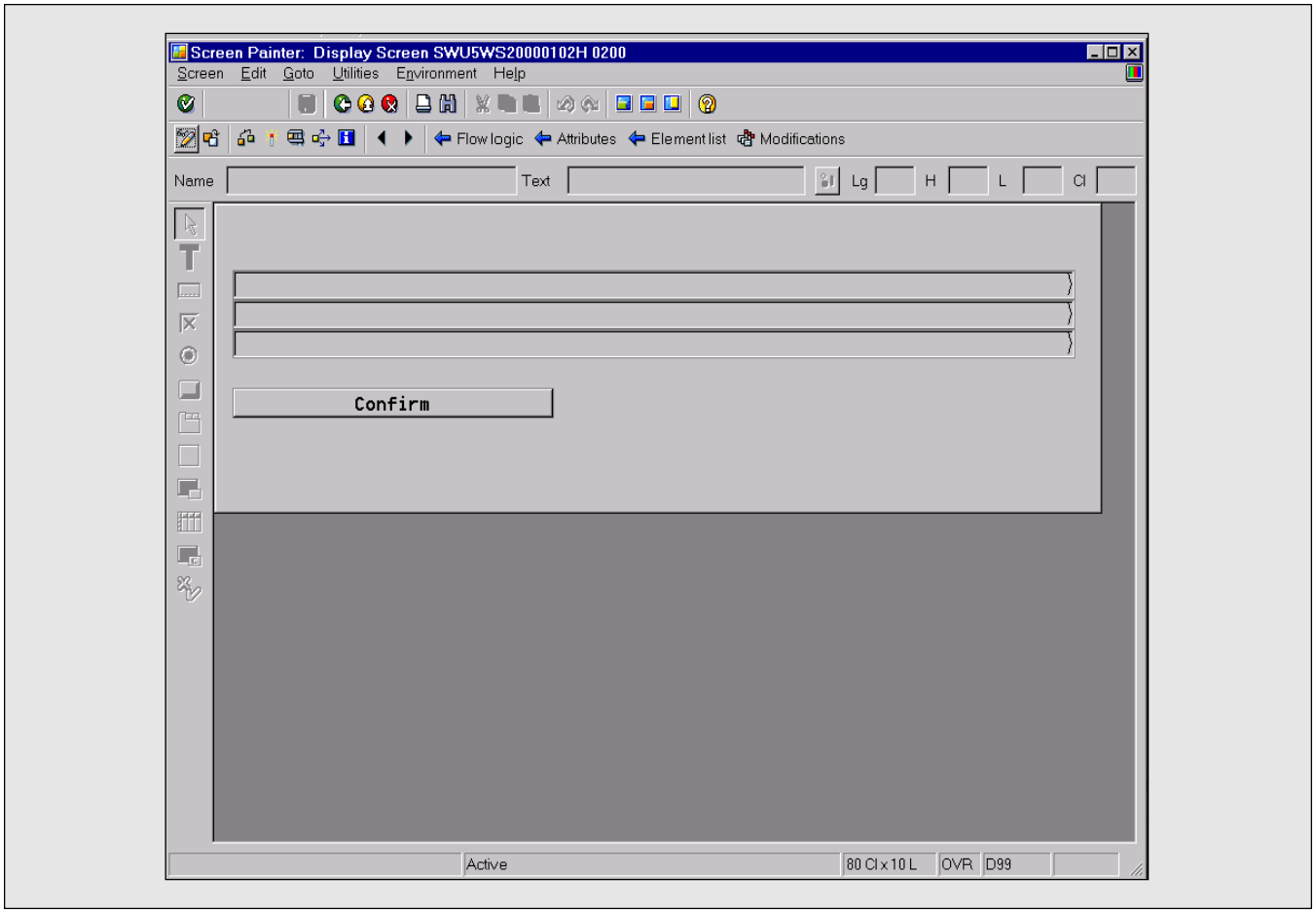


Figure 9 The Confirmation Screen (Screen 200)



- **Screen 200** is a confirmation screen and displays messages. This screen is shown in **Figure 9**.

Helpful Hints on Developing Web Forms

✔ If you have already carried out the development steps, including generating and beautifying templates, and you notice that you forgot to include, for example, a field for a mobile telephone number, and later learn that this field must appear on the form, you will usually just re-do these steps. This is not hard to do because they use generation mechanisms. Only the beautifying step should be adjusted manually, because by re-generating the templates, the beautification work will be lost.

✔ You need to be mindful of the fact that Web forms do not work well in situations that require a large amount of data to be passed in and out of a container. As you put flat data into the container, the container table (used to store the container elements persistently) can become very large. Thus, Web forms should *not* be used in situations that would require large amounts of data to be handled by the form.

✔ The Web transaction and module pool (program) that is automatically generated for you will typically suffice. You may, however, find that you need to modify the generated program. You may, for example, want to display some user-related data — e.g., the actual amount of vacation time on an

employee absence form. If you want to fine-tune the transaction, use two special modules that are not overwritten when you re-generate the Web transaction; one is called before displaying the Web form, the other one after the Web form has been processed.

✔ The Web transaction and module pool is generated in a name space that is fixed during customizing. If this customizing has not been carried out, the generation will fail.

✔ Call the transaction directly from the command field in the SAP screen (/nTS_WS20000102H). If the transaction does not run here, it certainly will not run when called from the Web interface.

Insert a Form Step in the Workflow

This step applies only when creating Web forms for the execution of work items. A "form step" is simply a dialog step in the workflow that is executed via a Web form. You define a form-based task, and then create an activity in the workflow that refers to this task.

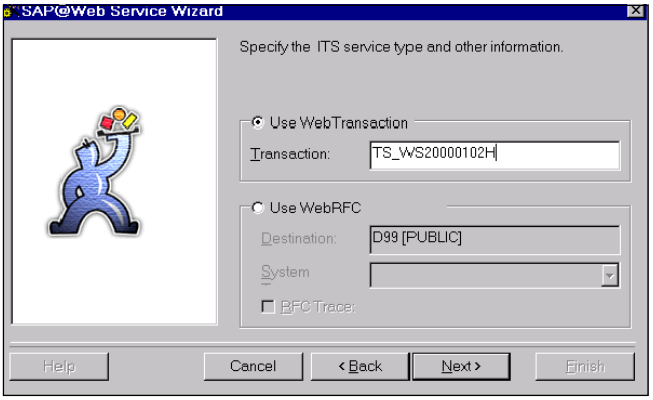
Web Studio-Based Development Activities

After generating the Web transaction from the task, you switch to the Web Studio development environment to generate and publish the ITS Service and HTML templates.[3]

[3] The activities that are performed in the Web Studio development environment will not be familiar to a developer with only SAP Business Workflow experience. The Web Studio development environment resembles the popular PC-based software development studios, not the development environment to which many SAP application developers have grown accustomed.

The wizard you use to generate the ITS service is the Service Wizard, shown in **Figure 10**. The process is easy, requiring you to enter just the following pieces of information:

Figure 10 The SAP@Web Service Wizard

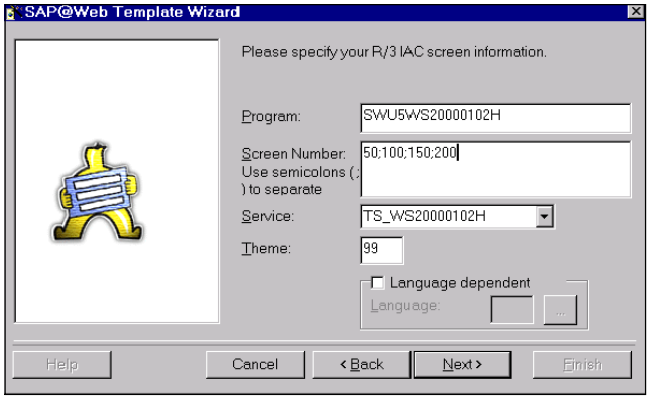


- **Which SAP system is to be used:** When calling this service, the ITS connects to that SAP system.

- **Which logon information is used:** In a workflow situation, a user's actions are usually done under his account. So when defining the ITS service for the Web transaction, the parameters for the userID and the password should be left empty, only the client is fixed. Thus a user is forced to identify himself with a userID and password. This user indentification is especially important in work item execution, because work items generally need to be executed under the userID, not just as an anonymous user. (The opposite would be to enter a fixed user's name, often some anonymous Internet user. The service and transaction would then be called under that user's userID).

- **Which transaction should be called:** In our case, this is the customer Web transaction we have just created.

Figure 11 The SAP@Web Template Wizard



HTML templates are generated using the Template Wizard shown in **Figure 11**.

Earlier, you created a module pool with one transaction and four screens. Here, you enter that information into the wizard. The wizard logs on to the SAP system, draws the data for these SAP screens, and for each screen, generates an HTML template. This is shown in **Figure 12**.

The template name is calculated from the module pool and the screen number as programname_screennumber.html — e.g., SWU5WS20000102h.html.

Publishing the ITS Service and HTML Templates

After generating the ITS service and HTML templates, you "Publish" them on the ITS. "Publish" is a Web Studio operation (shown in **Figure 13**) that writes the ITS service and HTML templates into specific directories on the ITS, making them accessible via the Web. This is where the site definition is used: It tells the Web Studio to which computers

Figure 12 An HTML Template for Starting a Workflow

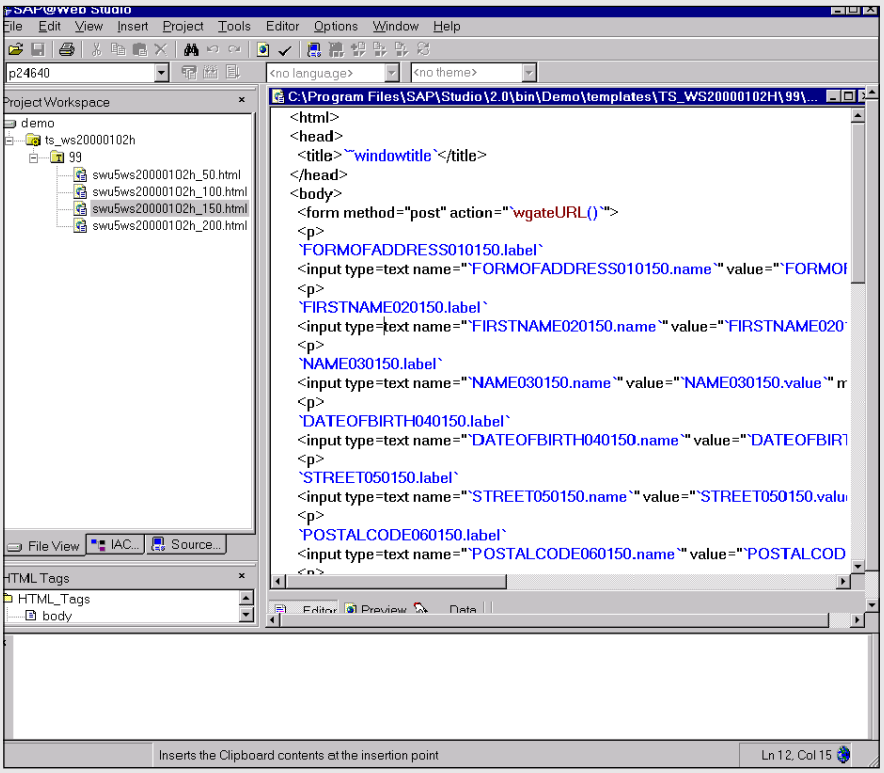
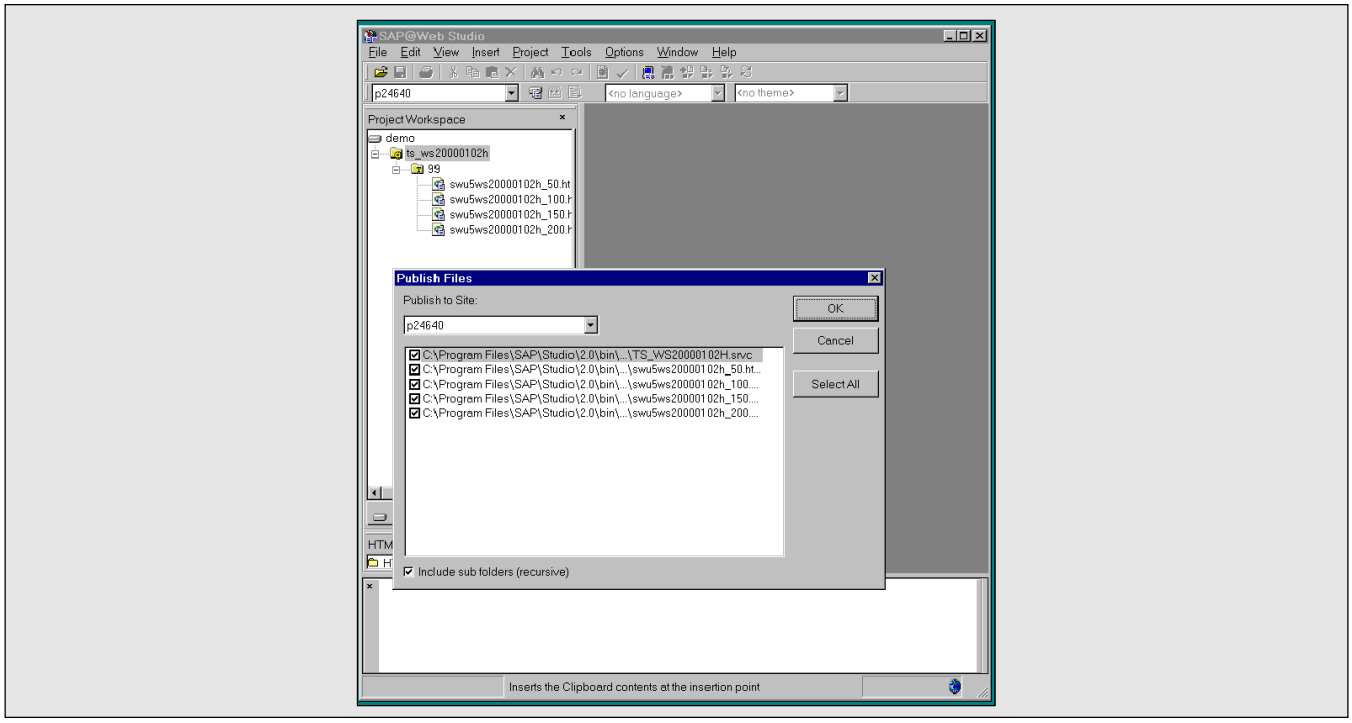


Figure 13 *The Web Studio "Publish" Operation*



(both the Web server and the ITS) the files should be written, into which directories they should be put, and what URL prefix is to be used to call the services.[4]

Helpful Hints on Generating and Publishing the ITS Service and HTML Templates

✔ Use cut-and-paste from the SAP screen to enter information about the transaction and the module pool into the wizard. This saves you keystrokes. It also helps you avoid the situation where you have mis-entered information. Then, carefully check that this cut-and-paste worked. You want to make sure that the strings are the same.

✔ Most developers do not want to produce language-dependent templates, but if you inadvertently fill in the *language field* when generating your templates, that's exactly what you'll get.

✔ It's a good idea to verify that the HTML templates contain all the elements you had anticipated. You can use the Preview menu item to check that your HTML templates approximate your target SAP screens.

Use cut-and-paste from the SAP screen to enter information about the transaction and the module pool into the wizard. This saves you keystrokes. It also helps you avoid the situation where you have mis-entered information. Then, carefully check that this cut-and-paste worked. You want to make sure that the strings are the same.

[4] A prerequisite to developing Internet Application Components is the creation of a "site definition." This is a setting which only needs to be done *once* per ITS. It need not be created for each Web form. The site definition tells the Web Studio where to write the Service (i.e., to which computers and which directories) in order to run and test it. It is the link between the Web Studio development environment you installed on your development machine for creating Web forms and the ITS, which contains the runtime service and templates.

✔ If the site definition is not correct, publishing will fail. If you experience a failure, double-check the information that was provided with regard to the site definition. And be sure that your development machine (where the Web Studio runs) has access to the corresponding directories of the ITS and the Web server.

✔ If you want to use a search help for entering data values into form fields, you may use the HTML Business command **~searchhelp** for those fields, thus both validating the input to a field and displaying which values you may enter are also available via the Web.

Test the Operation of Your Web Form

It's now time to test whether the Web form you have created will indeed run on the Web. The quickest way of testing is to simply run the Web Service and enter the relevant data:

- If you are testing a Web form that starts a workflow, you set the parameter Web flag to "X," and press the submit button.

- If you are testing a Web form that executes a work item, enter the work item ID, then set the parameter Web flag to "X," and press the submit button.

But for an integration test, where you are testing to make sure that a call via the Web ignites the right sequence of events, you need to do a bit more.

To perform an integration test on a Web form that originates a workflow, enter the URL of the form:

```
http://<WebServer>/scripts/wgate/
<Service>/!?WEB_FLAG=X&~
OkCode=CONT
```

For example:

```
http://disz200.wdf.sap-ag.de/scripts/wgate/
TS_WS20000102H/!?WEB_FLAG=X&~
OkCode=CONT
```

- **disz200.wdf.sap-ag.de** identifies the Web server.
- **/scripts/wgate** identifies the executable of the ITS.
- **TS_WS20000102H** is a workflow-specific parameter that identifies the service of the Web form that is called (it is the same as the Web transaction name here).
- **WEB_FLAG=X&~OkCode=CONT**, which is also workflow-specific, indicates that the screen field WEB_FLAG is set to X and the Ok-Code CONT is passed to the transaction; this allows invisible processing of the first screen, screen 50.

As you can see, the URL you have to enter in order to test a Web form that has been created to originate a workflow is complex. Mis-typing the URL is a common mistake. Fortunately, this URL is used only during testing, so users are never exposed to its complexity!

To perform an integration test on a Web form that has been established for executing a work item, you define a workflow that uses the task of the Web form in some step. When you run that workflow and receive a work item for that step in your inbox and go to execute it, the Web form is called. You do not have to concern yourself with the URL here, because you do not call the Web form directly. It is called via the Web Business Workplace.

Go Back and Beautify Your Templates!

Technically speaking, your HTML templates are now fully operational. The problem is, these

monochromatic, gray templates are not very attractive. I suspect most of you will want to beautify them a bit, adding a company logo perhaps, a splash of color, graphics, etc.

Since this step is not tied to the SAP system, you may want to have it done by a Web page designer. Revising these templates is simply a matter of manipulating the HTML templates, screen 100 or screen 150 (depending on whether you use workflow start or work item execution) and screen 200, in the usual way of manipulating HTML files (screen 50 will not be visible). All the processing functionality will be preserved.

Technically speaking, your HTML templates are now fully operational. The problem is, these monochromatic, gray templates are not very attractive. I suspect most of you will want to beautify them a bit, adding a company logo perhaps, a splash of color, graphics, etc.

Helpful Hints on Beautifying Your Templates

✔ HTML templates are not real HTML pages. Rather, they are templates that are used as a basis for producing real HTML pages at runtime (an activity that is performed by the ITS). For that purpose the templates contain embedded HTML Business directives/commands. These directives/commands should be left untouched.

✔ Don't underestimate the value of a good Web page designer!

✔ With regard to code maintenance, the service and the templates can be checked into the SAP system, where you can take advantage of the usual transport mechanism for maintaining the service and templates (and also images, if they are used).

✔ Strictly speaking, Web forms only supply just one submit button. It would be nice, particularly in work item approval situations, to have two buttons — one to accept and one to reject. As you revise the aesthetics of your templates, you can introduce a neat trick to get around the one-button limitation. (We incorporated this technique in our customer example.)

Say, for example, you have a form to Accept or Reject data. You could introduce an "approval state" container element that is capable of getting the value "A" for accept and "R" for reject. You introduce the two buttons by means of HTML. Pressing the Accept button, for example, would set the approval state variable to "A" and deliver the Ok-Code for submit — i.e., you simulate the process of entering "A" for the approval state field, and then submit the form (see Figure 3).

Conclusion

Web forms should be used for workflows that do not require lots of different views and immediate validation between fields. Developing Web forms to support a workflow that lets a user fill in a vacation request form and then have that form be directed to the user's manager for approval is a good example of a simple workflow. Creating a material master record is an activity that is *not* well-suited for Web forms. This is an activity that requires many views to be filled in and would be better served with an SAP transaction running in the local SAP GUI or accessed through the Web browser.

So, keep it simple. Reserve Web forms for workflow situations that do not require large amounts of data to be handled, that do not have complex transaction processing requirements, and that do not have complex data validation requirements. Focus your Web form efforts on workflow situations that need to support inexperienced SAP users, where Web access is desirable, and where no immediate writing of business data is necessary.

That's not to say that you can't mix and match forms-based and transaction-based workflows. You can. You can devise workflows that support a mix of both. Incorporating Web forms in broader workflow scenarios can make life simpler for users who are not well-versed in powerful SAP transactions, and it opens up access to SAP workflow applications to anyone with a Web browser.

Irrespective of the drive to make SAP transactions available through Web browsers, Web forms have the advantage that they can be beautified using any standard HTML editor. This enables the forms to blend seamlessly into any company's intranet or public Web site.

Rainer Weber joined SAP AG in 1994, and since then, he has been working in the development of SAP Business Workflow. His principal areas of focus are interfaces to external systems, among others, and also Web interfaces. Rainer can be reached at rainer.weber@sap.com.

