Best practices for planning, deploying, and maintaining mobile applications with SAP NetWeaver Mobile 7.0

by Alexander Ilg and Karsten Strothmann

Alexander Ilg
Founder,
msc mobile ltd.

Karsten Strothmann
Regional Group Expert,
SAP AG

(full bios appear on page 146)

In the not-so-distant past, access to core business processes was restricted to users working in an office environment. Many users were therefore unable to participate in these processes because they did not have access to the required IT infrastructure due to the nature of their work (sales representatives at client locations and service technicians at customer sites, for example). This often led to gaps in the process that needed to be filled by additional processes, such as manual data entry, which, as you may be painfully aware of, can be a tedious and error-prone process. Mobile applications are designed to bridge these gaps by extending the reach of enterprise applications such as asset management, sales support, supply chain management, procurement, and timesheet and travel expense tracking to offsite employees via mobile devices, such as smart phones, PDAs, tablet PCs, laptops, etc.

As a first step toward bringing mobile functionality to users, SAP R/3 Release 4.6B introduced SAPConsole — an ABAP-based "screen scraper" technology that converts back-end applications running on an SAP server to a character-based format designed to run on the small screens of mobile devices. While SAPConsole continues to be widely used by many ABAP teams for ABAP-based mobile development,[1] this technology is limited by the need for devices to always be connected to the network because the application logic must reside on the server. This means that if you lose your connection, you lose the ability to use the application at all — the last thing you want when you are at a customer site trying to look up some urgently needed spare part in your inventory. To address this limitation, with SAP NetWeaver Mobile 7.0, the mobile applications reside on the device, enabling them to run regardless of whether they are connected to the network — if they are disconnected, any new entries are simply updated on the server when the device is reconnected. The Java-based programming

[1] Look for an upcoming article in *SAP Professional Journal* on SAPConsole application development.

Note!

Released with SAP NetWeaver 2004s (7.0), SAP NetWeaver Mobile 7.0 is the latest version of SAP's platform for enterprise mobility. It is the successor to SAP Mobile Infrastructure (MI), which was introduced with SAP Web Application Server 6.20. SAP NetWeaver Mobile 7.0 is almost identical to SAP MI 2.5 (SP 18 and higher). The major difference is that in SAP NetWeaver Mobile 7.0, the SAP WebConsole has been replaced with the SAP NetWeaver Mobile Administrator. So, if you are currently using SAP MI 2.5, most of this article is valid for you as well.

model of SAP NetWeaver Mobile enables you to easily integrate non-SAP back-end applications. SAP NetWeaver Mobile also offers sophisticated administration and development functionality.

This article introduces developers, consultants, and project leads to the SAP NetWeaver Mobile technology and outlines the ingredients for successfully planning, deploying, and maintaining mobile applications with SAP NetWeaver Mobile 7.0. Mobile projects are very different from traditional projects — they add new dimensions of complexity as a result of the distributed environments that they enable, which can encompass thousands of mobile devices (see the sidebar below for more on the key differences). Newcomers to this kind of project sometimes try to run mobile projects in a traditional way and often fail. This article aims to help you avoid this pitfall and set you up for success with your own mobile implementation project. While addressing the needs of the novice, this article also assists mobile experts by providing new ideas and perspectives that can be directly applied to current and future projects.

We will start with an introduction to the SAP NetWeaver Mobile architecture and how it works. Then we'll take a tour of the 10 key phases involved in a typical mobile project: gathering requirements and analyzing processes, creating a plan and assembling the project team, selecting the target mobile

What is so different about mobile environments?

Mobile landscapes are distributed environments that consist of a huge number of mobile devices — sometimes as many as 50,000. This adds a whole new level of complexity to your landscape. Instead of doing an update on one server, updates must be deployed on perhaps *thousands* of devices. This can be a big challenge for an administrator.

Mobile devices are typically geographically distributed and are not stationary. In addition, such devices are not always online — they are only occasionally connected, so direct or remote access to devices is not always possible. This can become a problem, for example, if you change a component on the server side that affects the clients as well. How can you guarantee that all clients are up-to-date with the server component when they are not always online?

And finally, compared to desktop PCs, mobile devices (PDAs and tablet PCs, for example) have limited CPU power and memory, less mature operating systems, and some hardware shortfalls, such as restricted battery life. Due to these factors, their capabilities and reliability are sometimes limited, which adds to the challenge of maintaining a consistent, smoothly running mobile environment.

device, choosing a connection type, setting up the mobile landscape, developing or enhancing the mobile application, securing your mobile landscape, testing the mobile application, training your users, and deploying the mobile application. We will also take a brief look at how these phases might be applied to an example implementation project. Last, but not least, we'll focus on a critical aspect of mobile projects that is often neglected (and can result in the loss of valuable data and a frustrated end-user base) — operating the mobile environment in the months and years following go-live.

> ***Note!***
>
> This article does not delve into how to actually code an SAP NetWeaver Mobile application but rather explores key considerations involved in each of the implementation phases. It is important to understand how mobile environments work before doing any hands-on development intended to run in such environments.

Whether you're a consultant or project manager seeking to gain insight into the mobile world, or a developer who wants a solid understanding of mobile SAP environments to ensure an optimal development approach, this article serves as a useful starting point. It provides insight into all aspects of mobile projects and enables you to see the full picture, which is needed to make your project a success.

Let's start by having a quick look at the SAP NetWeaver Mobile architecture and some of the functionality it offers.

SAP NetWeaver Mobile overview

SAP NetWeaver Mobile, which is based on the open industry standards Java and XML, provides the infrastructure that allows you to extend the reach of back-end applications, such as SAP Customer Relationship Management (CRM), Human Resources (HR), Product Lifecycle Management (PLM), and others, to mobile devices. Using SAP NetWeaver Mobile prepackaged and custom-developed mobile solutions allows users to access and modify back-end data wherever, whenever, on their own using mobile devices without the need to add a manual data-entry step to the process. SAP NetWeaver Mobile supports two modes for mobile solutions — "online" and "offline":

- **Online** solutions only work when the mobile device has a connection to the server — i.e., there *must* be reliable network access. These solutions are usually browser-based and do not require a software client installation on the device; they are generally referred to as *thin clients*. SAP NetWeaver Mobile online solutions are developed or enhanced in SAP NetWeaver Developer Studio using the Mobile Web Dynpro technology. SAP MSOn (Mobile Sales Online) is the only prepackaged online solution currently delivered by SAP.

- **Offline** solutions work without a permanent network connection — they only need to connect to the server from time to time to exchange (synchronize) data. Offline solutions require a software client installation on the device because specific code and data must be preinstalled in order to use applications without a network connection; they are generally referred to as *fat clients*. Offline applications use a Java client for tasks specific to offline solutions, such as software deployment and data synchronization. SAP NetWeaver Mobile offline applications are developed or enhanced in SAP NetWeaver Developer Studio using the SAP Mobile Development Kit (MDK). SAP offers a variety of prepackaged offline solutions (more on these in a moment).

While the Java-based development environment for online solutions offers more sophisticated tools and enhanced capabilities than were previously available, the real strength of SAP NetWeaver Mobile lies

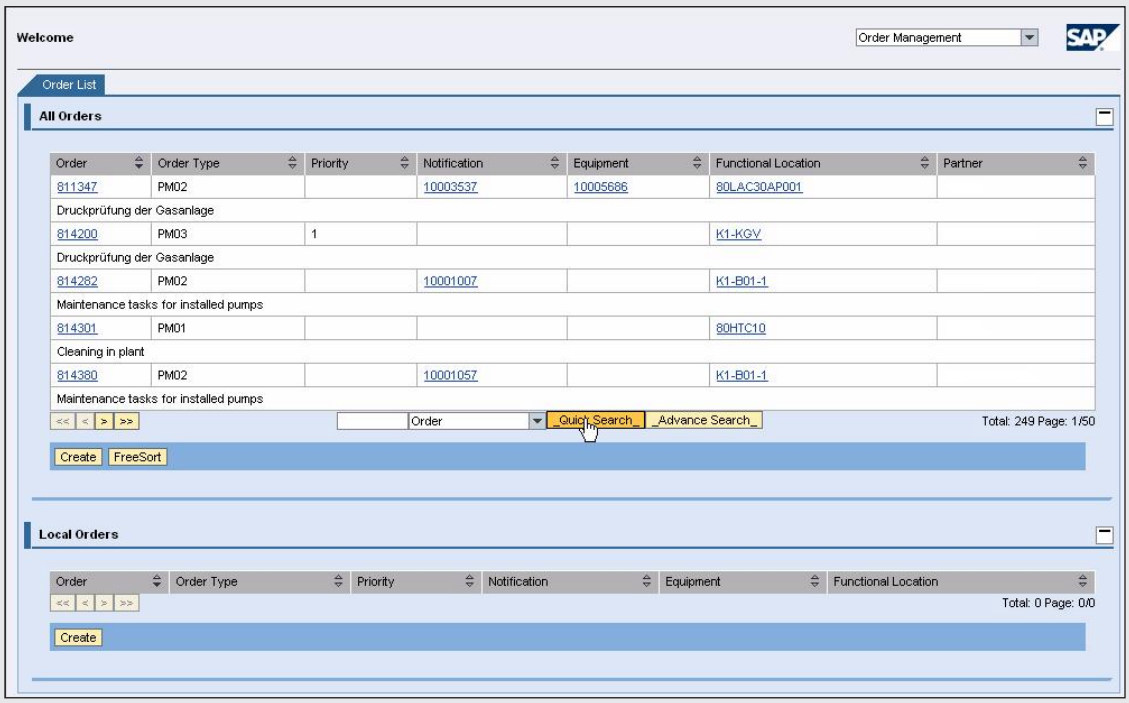


Figure 1 Laptop version of SAP xApp Mobile Asset Management (xMAM) 3.0

in its support for offline solutions. In many cases, it is either impossible or too costly to maintain a permanent online connection. For example, if you work in an environment with machines, they could block the GPRS (General Packet Radio Services) or 3G (third generation wireless technology) signal through scatter or magnetic fields and make a connection impossible. Or you could work in areas with insufficient network coverage. In some places, like hospitals, the use of wireless technology is banned entirely. In addition, online solutions often require high data transfer rates that are difficult to support on a constant basis. SAP CRM, for example, sometimes involves transferring data for thousands of customers and massive order histories. This is where the advanced data synchronization mechanisms provided by SAP NetWeaver Mobile for offline solutions come into play — whenever devices connect to the network, data and updates are exchanged with the back-end system, enabling business processes to execute even if there is no permanent connection to the network and limiting the duration of costly data transfers.

SAP offers a selection of prepackaged offline mobile applications for SAP NetWeaver Mobile.[2] These applications, which are based on xApp composite application technology, include:

- **SAP xApp Mobile Asset Management (xMAM):** xMAM, which integrates with SAP ERP, helps to reduce equipment downtime by giving technicians instant mobile access to work orders and equipment information from back-end applications, such as Plant Maintenance (PM) or Customer Service (CS), and enabling supervisors to manage all activities centrally. xMAM can be used with laptops (**Figure 1**), tablet PCs, PDAs (**Figure 2**), and Windows Mobile-based smart phones. SAP also offers a version of xMAM that is specific to utilities and provides a number of utilities-specific processes, such as periodic meter reading, in addition to standard xMAM processes.

[2] Note that the prepackaged solutions offered by SAP require a separate license.

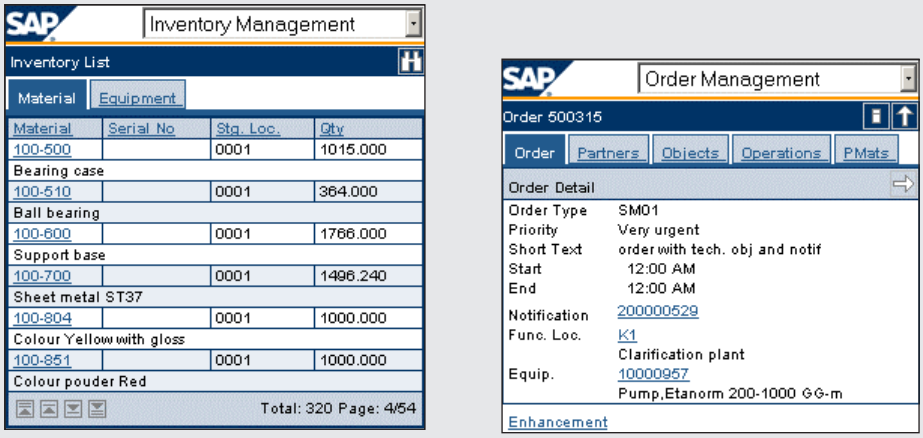


Figure 2 PDA version of SAP xApp Mobile Asset Management (xMAM) 3.0

- **SAP xApp Mobile Time and Travel (xMTT):** xMTT, which integrates with the HR and Financials (FI) modules of SAP ERP, enables employees in your company to record their working times and enter their travel expenses on laptops wherever they are, even when they are offline. Once they have a network connection again, they can synchronize their data to the back-end system.
- **SAP xApp Mobile Direct Store Delivery (xMDSD):** xMDSD, which integrates with the Direct Store Delivery (DSD) solution of SAP ERP, enables delivery drivers and field sales forces to quickly respond to customer needs for new and revised orders. This capability also helps with reducing material losses, providing companies with timely, accurate customer data, and reducing paperwork. xMDSD can be used with PDAs and with Windows Mobile-based smart phones.
- **SAP xApp Mobile Sales Handheld (xMSA HH):** xMSA HH, which integrates with SAP CRM, enables sales representatives to carry out their work anywhere, anytime, using a handheld device (i.e., PDAs and Windows Mobile-based smart phones) in offline mode.

> ***Note!***
>
> The prepackaged mobile applications do not necessarily support all target device types. Some applications, such as xMAM, support multiple device types, while other applications, such as xMTT, support only specific target devices, so be sure to check which device types the specific application supports when planning your implementation. This is also a consideration when custom-developing a mobile application — the type of device you need or would like to target will dictate your development approach.

As mentioned previously, these prepackaged solutions can be used as is, or they can be enhanced and customized using the MDK in SAP NetWeaver Developer Studio to meet the needs of your particular business. Customers and SAP partners can also build their own offline applications with SAP NetWeaver Mobile using the MDK. You can build custom applications to access any kind of back-end system, including SAP ERP, CRM, Business Intelligence (BI),

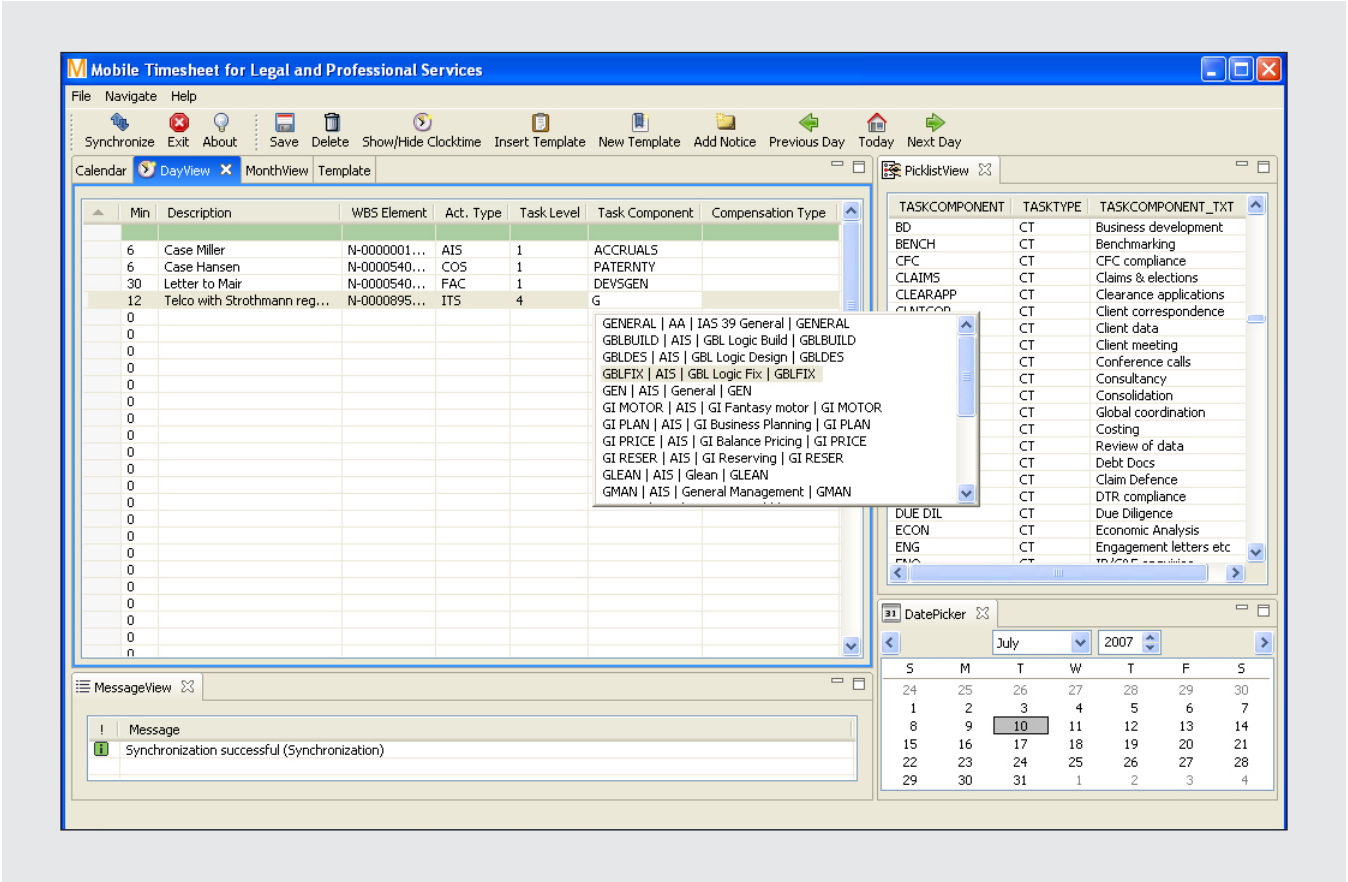


Figure 3 An example partner solution for SAP NetWeaver Mobile

and even non-SAP back-end systems. **Figure 3** shows a custom-developed SAP NetWeaver Mobile-based partner solution (a mobile timesheet for legal and professional services) that integrates with the SAP ERP modules HR and FI.

Let's now examine the underlying architecture that enables SAP NetWeaver Mobile and how it works.

SAP NetWeaver Mobile 7.0 architecture

In the case of offline solutions, SAP NetWeaver Mobile 7.0 is composed of a client part that resides on the mobile device and a server part that is embedded in the SAP NetWeaver Application Server (AS) ABAP and Java stacks, enabling access to the back-end systems. Therefore, a minimal SAP NetWeaver Mobile landscape consists of one or more clients (mobile devices), a middleware server (SAP NetWeaver AS), and one or more back-end systems (SAP CRM, HR, BI, etc.), as shown in **Figure 4**.

Let's take a closer look at the architectural elements of SAP NetWeaver Mobile:

- The **client component** of SAP NetWeaver Mobile is a Java-based framework that provides a runtime platform for the applications on mobile devices. It offers generic services and libraries through an API that is accessible by mobile applications. The client component can be deployed either on Microsoft Windows laptops and tablet PCs running JRE 1.4.x or on Windows Mobile-based PDA devices and smart phones running JRE 1.1.8. The Java runtime provides for this platform independence. As for the user

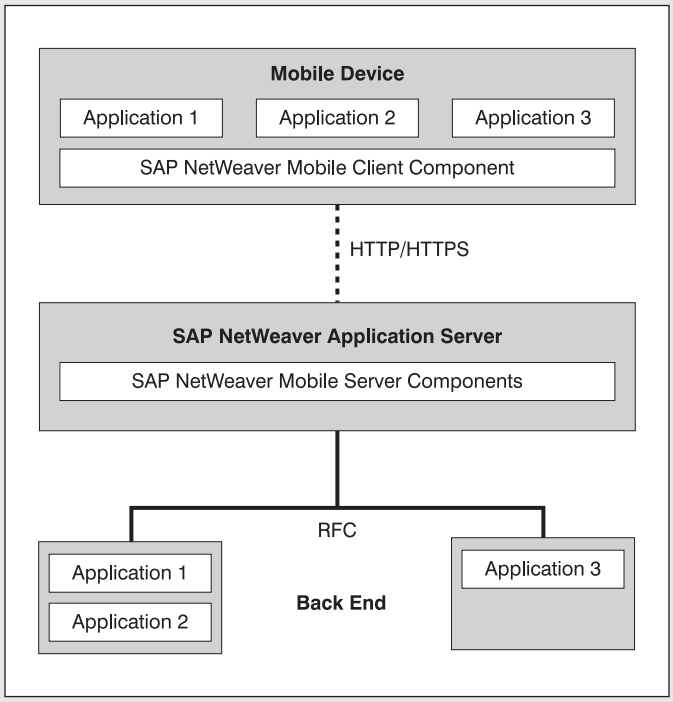


Figure 4 SAP NetWeaver Mobile 7.0 architecture overview

interface (UI), SAP NetWeaver Mobile offers two different UI programming models for the client: Java Server Pages (JSP) 1.1 and the Abstract Windowing Toolkit (AWT).

- The **server component** of SAP NetWeaver Mobile is located on the middleware and spans the J2EE and the ABAP stacks of SAP NetWeaver AS. It handles the reception and transmission of data between the client (via HTTP/HTTPS) and the application back end (via RFC). Working in offline mode requires duplicate persistence of data via synchronization — data that is stored in the back-end system is duplicated to the mobile clients and, in the case of SmartSync synchronization, even to the middleware (see the sidebar on the following page for more on the SAP NetWeaver Mobile synchronization mechanisms). The server component ensures consistency throughout the synchronization of data between the components. Additionally, the server component is responsible for executing the deployment of software components such as mobile applications, patches, and drivers to the client devices. Deployment, which is managed using the SAP NetWeaver Mobile Administrator tool, includes all the steps that are required to enable a user to run a specific version of a mobile application (more on this later in the article). To enable administrators to monitor the landscape, SAP NetWeaver Mobile also features specialized central monitoring and tracing on the server component, which includes monitoring of synchronization as well as system availability via the SAP NetWeaver Mobile Administrator.

- For the **back end**, SAP NetWeaver Mobile utilizes BAPIs, which are included with all standard SAP systems and are used to retrieve data from the database and expose them as business objects. For use with SAP NetWeaver Mobile, these BAPIs are in wrappers that provide interfaces that meet the requirements of the SAP

SAP NetWeaver Mobile synchronization mechanisms

Synchronization is a mechanism for data exchange between mobile devices and back-end systems. The goal of mobile applications is to update the back-end system with data collected in the field. SAP NetWeaver Mobile offers two synchronization modes: GenericSync and SmartSync. GenericSync is best suited for simple data entry scenarios such as time recording. Because it is more powerful, SmartSync is recommended for more complex scenarios such as enterprise asset management.

GenericSync

GenericSync is a simple method for exchanging data between the mobile client and the back end. It allows easy linking of a mobile application to an SAP back-end system by enabling calls to any RFC-enabled SAP function module residing on the back-end system. The response data of the module called is transported back to the device without the SAP NetWeaver Mobile server component influencing the data. The device needs to actively pull the data from the back end, and the function module needs to retrieve the correct data for the current user and also update the state of the device.

While GenericSync allows for application-specific control of the synchronization process, there is no application data cached on the SAP NetWeaver Mobile server component and no delta determination mechanism for detecting and transferring changed data from client to server and vice versa. This can lead to either a long development process to implement this mechanism yourself or to an unnecessary transfer of data if a delta determination mechanism is not implemented.

SmartSync

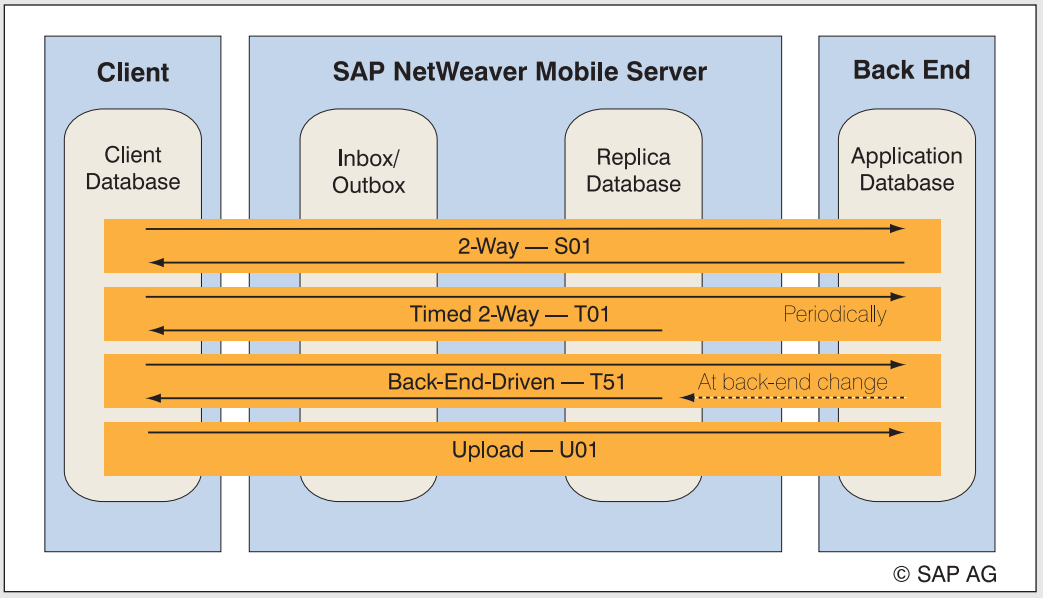
SmartSync is a full-blown data replication and realignment middleware solution based on extracts of business objects. (Business objects that can be synchronized in an SAP NetWeaver Mobile landscape are called SyncBOs.) SmartSync, which is built on top of GenericSync, intelligently transports data to the mobile devices. SmartSync tracks the current state of each device in the field and, using delta determination, guarantees that only changes made since the last synchronization are sent from the server to the client and vice versa. SmartSync detects conflict situations between client and back-end data and resolves them automatically.

SmartSync offers a number of different ways of synchronizing data with the back-end system:

- **2-Way SyncBOs** (synchronization type S01) immediately sends data to/from the back-end system and/or the mobile client. This ensures that the most up-to-date data is always passed on to the mobile client. This can lead to an increased load on the back-end system and a longer response time on the client device, but it is very efficient for frequently changing data objects.

NetWeaver Mobile synchronization mechanisms. The BAPI wrappers often contain additional mobile-specific logic or filter the data returned by the BAPI. Every ABAP-based system can be used as a back-end system (SAP CRM, HR, BI, etc.). If you deploy SAP applications for mobile business, the BAPI wrappers are shipped within a plug-in (available from the SAP Service Marketplace at

- **Timed 2-Way SyncBOs** (synchronization type T01) copies data from the back-end system to the middleware server component (and vice versa) using a periodically running batch job (replication). The data is cached in the middleware. Because there is no back-end access required during the actual synchronization of the mobile device, this synchronization type allows the shortest sync time, but the data may not always be up-to-date.
- **Back-End-Driven SyncBOs** (synchronization type T51), a.k.a. Server-Driven SyncBOs, copies delta data records from the back-end system to the middleware server component. Here, the back end is responsible for triggering data replication between the middleware and the back end. The greatest benefit is that the data traffic between the back-end system and the SAP NetWeaver Mobile server component is significantly reduced. The drawback is that it takes longer to implement due to additional required implementation steps.
- **Upload SyncBOs** (synchronization type U01) sends data from the mobile device to the middleware server component but not the other way around. You can create data on the device and send it via the middleware to the back end, but you cannot send data from the back end via the middleware to the client.



http://service.sap.com) for the back-end system supported by the specific application. (Keep in mind that login credentials are required to access the SAP Service Marketplace.)

Now that you have a foundational understanding of SAP NetWeaver Mobile, let's take a look at the key phases involved in mobile implementation project and the considerations to keep in mind along the way.

The 10 key phases of an SAP NetWeaver Mobile implementation project

Over the next sections, we'll walk through the 10 key phases involved in a typical SAP NetWeaver Mobile project (see **Figure 5**). Although we primarily focus on SAP prepackaged mobile solutions, the same phases apply almost identically to custom-developed applications. Please also keep in mind that we're focusing on the differences between traditional and mobile implementations in the most important phases typically found in mobile implementation projects. This is not intended to be a full-blown software process model that you should adopt; rather, it is intended to provide mobile-specific advice that you can adapt to the software process you already have in place.

Let's now explore each of the 10 key phases needed to implement a mobile application, starting with requirements gathering and analysis.

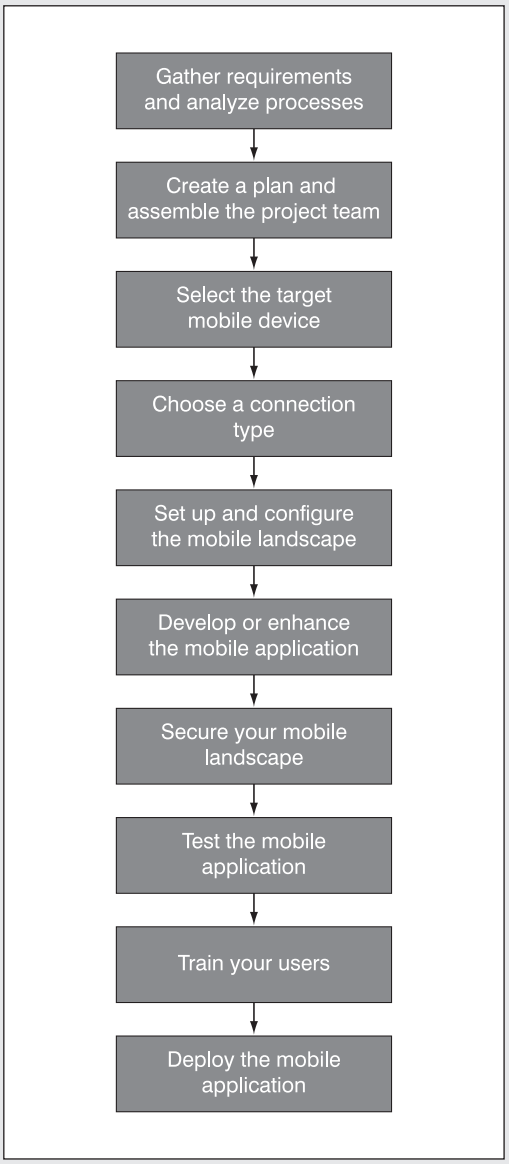


Figure 5 The 10 key phases of an SAP NetWeaver Mobile implementation project

Phase 1: Gather requirements and analyze processes

The first thing you need to do is analyze your current business processes to dictate your needs. The results of this analysis will decide the type of mobile application you will implement, which will be one of the following:

- A prepackaged SAP mobile application without any enhancements, if the prepackaged application fully fits your requirements
- A prepackaged SAP mobile application with enhancements, such as changing the UI or adding functionality, if the prepackaged SAP application covers almost all of your requirements
- A custom solution, if no prepackaged application covers your business process

When making this decision, you have to keep in mind that the cost of a mobile project increases with every enhancement or development you make. It's generally a good idea to use a prepackaged solution whenever possible and try to adjust it to your needs

through enhancements. Don't economize too much though — minimizing the number of clicks required for a heavily used mobile process through enhancement or development can significantly increase time and resource savings if you have a large user base.

To help with the actual analysis of your business processes, you can use modeling languages like UML as you would for traditional projects, but don't forget to consider the differences inherent to mobile environments:

- **Uncertainty of location:** In a mobile environment, it is not clear where an activity will be executed, and this uncertainty might have an effect on a business process. For example, a user might not be able to look up needed data if they are in a location with limited coverage, delaying continuation of the process until a connection can be established.
- **Putting the technology before the process:** The analysis of mobile business processes is often very technology-driven. During the analysis phase, people tend to try to make a business process fit their mobile technology instead of the other way around. For example, if end users are already equipped with a smart phone, the tendency might be to try and save time and money by keeping the device, even if the end result is a frustrating and inefficient process. It would be better to do an unbiased analysis first to see if a different type of device would work better.

Be sure to integrate these differences into your model to ensure that you select the target device that is best suited to your business and user needs.

Phase 2: Create a plan and assemble the project team

The next step is to create a project plan. We won't go into the details of creating a plan, but it's very important to remember that mobile environments have a much higher degree of complexity than traditional environments, and this complexity will affect the configuration of your landscape as well as how you approach the rollout of the application for users.

Once the project plan is in place, you need to bring the project team together. On average you need a broader skill set than in traditional SAP projects because mobile projects include a wide variety of technologies from the entire SAP spectrum ranging from client-side technologies like Java to back-end technologies like ABAP.

Ensure that the following roles are represented on your project team:

- **Application specialist (back end):** The application specialist needs to have the relevant back-end business process and system (SAP CRM, HR, BI, etc.) knowledge for the specific solution being implemented in your project. For back-end enhancements, ABAP coding skills would be needed.
- **Mobile specialist:** The mobile specialist should know about the details of the SAP NetWeaver Mobile configuration and the mobile application installation, and should be experienced in technical blueprinting and deployment of mobile landscapes. The mobile specialist should also be familiar with the security aspects of a mobile landscape and should have a general understanding of network technologies such as GPRS, wireless LAN (WLAN), and 3G.
- **Technical specialist:** The technical specialist must have a solid understanding of SAP NetWeaver AS installation and administration.
- **Development specialist (front end):** The development specialist is required to know about any enhancements to the mobile application front end and/or back end, including knowledge of the following technologies:
  - Java, which is used to create the business logic on the SAP NetWeaver Mobile client side
  - The Mobile Client Application Framework (mCAF) enhancement concept, which is the framework on which the prepackaged SAP NetWeaver Mobile applications are built and

enables you to enhance these applications (we go into more detail on the mCAF when we discuss enhancements in Phase 6)

- JSP, HTML, and JavaScript, which are used to create the UI of an SAP NetWeaver Mobile application

- **Project manager:** As in any other project, a project manager is needed to coordinate the team members and ensure that the different pieces fit together in the end. Considering the complexity of a mobile landscape and the variety of technologies involved, the project manager should have at least a basic understanding of the technologies used.

If you cannot fill all these roles with appropriate personnel, you put your project at risk. Because most of these skill sets are complementary, it is very hard for team members to take over another role in addition to their own. An experienced Java front-end development specialist, for example, is not likely to have the required ABAP back-end knowledge or business process and customization know-how of an application specialist.

If you manage to put this team structure in place though, you are well-prepared for the next phase of your project — choosing the appropriate target mobile device.

Phase 3: Select the target mobile device

The selection of the target device depends on your usage requirements. **Figure 6** summarizes the supported devices and when it makes sense to use them. Environments in which proper handling cannot be guaranteed — environments where devices can fall from machines, get dirty, come into contact with tools like screwdrivers, etc. — require durable devices. In other cases, as in mobile CRM applications, for example, durability is not the key factor — key factors will more likely be the physical size of the device and its performance.

As you can see in the figure, for SAP NetWeaver Mobile applications, there are two main groups of devices from which you can select: Windows 32-bit (Windows XP/XP Tablet Edition/Vista) devices such as laptops and tablet PCs, and Windows Mobile-based devices such as PDAs and smart phones.

Let's take a closer look at some key factors when evaluating such devices:

- **Battery life:** How long does a battery have to last for end users to fulfill their tasks? End users need to be able to work without regularly being hindered by depleted batteries. In some environments changing batteries during the workday is not an option.

- **Performance:** What performance level is necessary in order to minimize delays and maximize the application's usability? The application's response time should be 3 to 4 seconds maximum for end users to enjoy working with it.

- **Memory:** Does the device have sufficient memory for the application you plan on running? For PDAs, it is recommended to have at least 128 MB of RAM for PPC 2003 devices and 64 MB RAM and 128 MB ROM for Windows Mobile 5.0 devices. For laptops this question is not as pressing because most laptops come with more than the recommended 128 MB.

- **Durability:** How durable does your device have to be? The more durable it is, the longer its lifespan is likely to be. You might spend more money replacing devices that are not durable enough than you would have spent buying durable devices in the first place.

- **Operability:** How much effort does it take to operate a device? Will users be able to use the device efficiently and comfortably? Laptops and tablet PCs have an advantage here with their larger size. Operability also goes hand in hand with accessibility.

- **Accessibility:** How easy will it be for support staff to remotely access a device to resolve problems? Windows XP/Vista-based devices allow for the

Device	Requirements	When to use	Advantages	Disadvantages
Laptop	Windows 2000/XP/Vista, 128 MB RAM recommended	In office environments or when users are able to carry a bigger device (e.g., when consultants record their travel expenses in xMTT using their laptops at hotels, clients sites, on flights)	High performance, easy to operate, large screen, good remote access support	Less portable, short battery life, can be fragile
Tablet PC	Windows XP Tablet PC Edition, 128 MB RAM recommended	When users are able to carry a medium-sized device (e.g., when the device can often be left in a service truck and is only picked up when needed rather than carried constantly); a good compromise between portability and performance	High performance, easy to operate, large screen, good remote access support, portable, durable	High cost, little or no keyboard, short battery life
PDA	PPC 2003, 128 MB RAM recommended; Windows Mobile 5.0, 64 MB RAM, 128 MB Flash ROM recommended	When portability is the major factor (e.g., when users have to carry the device all day long)	Portable, reasonable price	Lower performance, can be fragile, can be difficult to operate, small screen, limited remote access support
Ruggedized PDA	PPC 2003, 128 MB RAM recommended; Windows Mobile 5.0, 64 MB RAM, 128 MB Flash ROM recommended	When portability and durability are required (e.g., when users have to carry the device all day long in sometimes rugged environments)	Portable, durable	Very high cost, lower performance, can be difficult to operate, small screen, limited remote access support
Smart phone	Windows Mobile 5.0, 64 MB RAM, 128 MB Flash ROM	When portability and phone functionality are the major factors (e.g., when field service technicians have to carry a device all day long and be able to place and receive service calls)	Portable, reliable connectivity	Lower performance, can be fragile, small screen, can be difficult to operate

Figure 6 Devices supported by SAP NetWeaver Mobile 7.0

use of mature accessibility tools, such as PCAnywhere and WebEx, which make support a lot easier. Access to Windows Mobile-based devices can be much trickier.

- **Screen:** How easy is it for end users to read/manipulate data on the screen? Is the screen size sufficient for the application? Can users read the device in the expected ambient light conditions?

- **Data storage:** What data storage options does a device offer? Could you utilize other options for operating your mobile environment more effectively — could you transfer data to removable media, such as a USB drive, for example?

- **Stability:** Is the device stable and reliable enough? Stable and reliable devices reduce the possibility of data loss. Sticking to proven technology and experienced device providers helps.

- **Connection type:** How can you connect to the network with the device? Via cradle only or does the device have a built-in GPRS, 3G, or

WLAN modem? (We'll look at connection types in more detail in the next section.)

As you can see, there are many considerations involved in selecting a mobile device. Make sure you address all of these points when choosing a device.

Tip!

In addition to comparing the specifications, take the time to test the actual performance of the device. There are big performance variations between devices with the same specifications (CPU, RAM/ROM). Executing the same operation in xMAM on two different PDA models with the same high-level specifications can have drastically different results — one could take almost twice as long as the other. These disparities are caused by differences in the quality and the performance of the RAM and ROM used as well as the internal architecture of the device. A good memory controller, for example, can significantly improve a PDA's performance, and you probably won't find this data anywhere in the device specifications.

Keep in mind that there is never a perfect device for a certain use case. A mobile device is always a compromise and not all parties involved in the selection of the device will be completely satisfied with a device due to conflicting interests. Let's explore the different parties involved and their interests:

- **End users** like small and fast devices. In addition, the devices need to be as durable as possible to prolong their usefulness, they should not be too heavy, the battery needs to last for a long time, and even if the device is small, the screen needs to be as readable as possible. You can see where this leads — first of all, a lot of these points are contradictory (a small device only has so much space for a readable screen), and second of all, a device with most of these functionalities is certainly expensive.
- **Management** would like to buy devices that are as cheap as possible and have operating costs that are as low as possible. In addition, devices should work for at least three years without problems to keep the total cost of ownership low.
- **System administrators** prefer devices that are easy to operate and access, so the amount of administrative effort required is minimized.

Considering all these conflicting interests, how can you make sure that you choose the right device for your project? There are a few rules that can help you make the best choice possible:

1. **Involve all parties early in the project.** It's especially important to involve the end users, because they will ultimately make or break the project's success. If your end users are unhappy and won't use the devices, the project has failed, and the time and resource investments have been wasted.
2. **Test a number of devices.** Be sure to include different types of devices in your test, such as laptops, tablet PCs, and PDAs. Be open-minded when doing these tests. You might already have an idea of the kind of device you would like to go for, but it might turn out that the original device you had in mind is not the best device for your project.
3. **Always consider ruggedized devices** (if those devices make sense in your environment). True, the investment might be higher at first, but the total cost of ownership might be lower because those devices might live much longer than non-ruggedized devices.
4. **Make it a joint decision.** Get all involved parties to participate in and sign off on the decision. Even if the device is not the perfect answer for everyone, if everyone feels ownership over the selection, there is a better chance they will accept the necessary compromises.

Once you have selected the target mobile device, the next step is to decide on a connection type for the synchronization.

Connection type	Device	When to use	Advantages	Disadvantages
Cradle	PDAs and smart phones	As a workaround or for a very small number of users	Fast and reliable, low cost	Only available with network access; requires use of ActiveSync, a Microsoft-based synchronization software that can sometimes be hard to use
LAN/Ethernet	Laptops and tablet PCs	When users have access to a LAN	Fast and reliable, low cost	Network connection needs to be available; not possible for all device types (PDAs usually require an additional cradle)
WLAN (Wireless LAN)	WLAN-enabled PDAs, smart phones, tablet PCs, and laptops	When users have access to WLAN at synchronization time	Fast and reliable, low cost	Wireless network connection needs to be available, which limits the locations at which a synchronization can take place
Analog modem	Laptops and tablet PCs	As a last resort when there are no other options	Widely available (phone access is possible even in very remote areas)	Very slow, which limits the amount of data that can be transferred
GPRS (General Packet Radio Services)	Laptops, tablet PCs, PDAs, and smart phones	When end users work offsite and require flexible synchronization capabilities	Medium speed, widely available	Requires integration with a GSM (Global System for Mobile communication) cellular network; monthly contract costs; not always reliable; service differs depending on provider
UMTS (Universal Mobile Telecommunications System)	Laptops, tablet PCs, PDAs, and smart phones	When end users work offsite and require flexible synchronization capabilities	Fast	Requires UMTS coverage; availability limited primarily to metropolitan areas; monthly contract costs required; expensive

Figure 7 Connection types supported by NetWeaver Mobile 7.0

Phase 4: Choose a connection type

The SAP NetWeaver Mobile client can be synchronized over any TCP/IP network. The data is exchanged via standard HTTP or, if the connection is encrypted, HTTPS. Connection types include GPRS, analog modem, 3G, WLAN, LAN/Ethernet, and synchronization via a cradle/docking station for PDAs and smart phones (in this case, the device is using the network connection of the PC to which the cradle is attached).

Figure 7 provides an overview of the connection types supported by SAP NetWeaver Mobile 7.0 and when to use them.

The connection type you choose very much depends on the amount of data that is synchronized. If users synchronize only small amounts of data, a low bandwidth, such as that provided by GPRS, can be sufficient. If the users synchronize large amounts of data with the middleware server, a higher bandwidth, such as that provided by WLAN or UMTS, is recommended.

Keep in mind, however, that the stability of the connection is more important than the bandwidth that is available. Broken synchronizations because of interrupted connections lead to an overhead and user

frustration because the whole synchronization needs to be repeated. If the network is stable, these broken connections can be avoided. A stable, low bandwidth network is always preferable to an unstable, high bandwidth one.

This means that you should test the network of your choice. While there shouldn't be many differences between different LAN/Ethernet connections, the stability of GPRS can be very different with different network providers and in different locations. The same problem exists with analog modems, 3G, and WLAN. In addition, for GPRS and 3G you need to know the network coverage. Is a connection available everywhere your user might want to synchronize?

Let's take a quick look at some key factors when choosing a synchronization connection type:

- **Reliability and stability:** Is the connection reliable enough? What about stability? Unreliable and unstable connections can lead to frustration among end users, as well as errors that are hard to find and resolve.
- **Speed:** Is the connection speed high enough for your needs and for the amount of data that needs to be synchronized? Low connection speeds can again lead to frustrated end users because they might have to wait a long time for the synchronization to finish.
- **Costs:** What about the costs for synchronizing your data? Some connection types like LAN have almost no cost, while others like UMTS come with (sometimes expensive) monthly charges. Make sure that the cost for your connections are well calculated in advance and that they fit into your budget.
- **Availability:** Is a connection type available wherever end users need to synchronize? UMTS, for example, is currently not available everywhere.
- **Hardware:** Does the selected device support the chosen connection type? Plain LAN access is usually not suitable for PDAs, unless you are using the Microsoft ActiveSync synchronization technology and a cradle.
- **Support:** If you choose an external provider, make sure that the carrier you use is able to support you in case of issues. This applies primarily to providers of GPRS or UMTS connections.

Once you have determined the connection type, you can then set up and configure your landscape, which we will review next.

Phase 5: Set up and configure the mobile landscape

In order to set up your mobile landscape with a hardware configuration that will support the implementation, you need to size it and determine your scalability needs. We look at these two critical tasks here.

Sizing

Sometimes project teams don't understand the importance of sizing. Even the best mobile solution is of little help if the performance of the middleware or back end is poor. By performing a proper sizing, you can make sure that your landscape works in an optimal way and can handle the load.

SAP offers information about sizing for SAP NetWeaver AS and also for mobile clients in the following documents, which are available from the SAP Service Marketplace:

- "Sizing Guide for Mobile Business Applications" (http://service.sap.com/mobile)
- SAP Note 874006

The best way to minimize synchronization times and keep the system fast and responsive is to reduce data transfer volumes. Wherever possible, avoid downloading unnecessary data to the mobile middleware and to the mobile device. For example, if you can download only the data for a user's region in contrast to the worldwide dataset, you can significantly reduce the data volume.

Depending on the type of synchronization (GenericSync or SmartSync), the critical components are either the middleware or the back-end system. In general, use the following guidelines:

- **SmartSync** applications usually create more load on the middleware, especially on the database server. Adding application servers yields minimal benefit, but adding more power to the middleware database server may produce significant gains.
- **GenericSync** applications tend to cause more load on the back-end system rather than on the middleware server. The reason for this is that each request has to go to the back end because no data is replicated to the middleware.

While the recommendations for sizing are highly dependent on your particular organization, in general keep in mind that a realistic sizing can only be done if the following information is available:

- Number of concurrent parallel synchronizations
- Number of users
- Amount of data per user
- Type of synchronization (GenericSync or SmartSync)
- Type of SyncBOs used (if SmartSync is used)

Scalability

Scalability is an important factor to keep in mind for the middleware and the back-end system. It will be a lot easier to adapt to either a growing number of mobile users or a growing amount of data per user if your landscape is scalable.

The following components should be scalable in an SAP NetWeaver Mobile landscape:

- **SmartSync** environments require the middleware database to be scalable. SmartSync is database-intensive, so it is more important to have a scalable database server than a scalable application server.
- **GenericSync** environments do not cause significant load on the application or database server of the server component middleware. Here, the middleware is mostly used to pass data from the client to the back end and vice versa. The majority of the workload, generated by the RFC calls from the middleware to the back-end system, is on the back-end system and the network. Sufficient dialog work processes should be available on the back-end system. Each RFC call requires a free dialog work process, so if the number of parallel RFC calls exceeds the maximum number of available dialog work processes, the performance and reliability of the synchronization will be drastically compromised.

Once you have sized your landscape and settled on your scalability requirements, you can set up and configure your landscape accordingly.

With the hardware configuration required to support your mobile implementation in place, you next need to address the mobile application you'll be using.

Phase 6: Develop or enhance the mobile application

In this phase, either you will design, develop, and implement a custom application, or if you are using a prepackaged application, you will need to implement the necessary enhancements, if there are any. We'll take a brief look at these tasks here.

Developing custom applications

Developing a mobile solution for a small device like a PDA or a smart phone can be a challenge, much like the early days of computing when memory and CPU power were valuable commodities. Developers had to watch every bit they added to memory and think twice about every line of code to make sure no performance was lost. As RAM and hard disk memory got bigger and cheaper over the years, there was less of a need to be so vigilant about consuming resources. With

Tool	Used for
ABAP Workbench	Creating the BAPI wrapper and function modules in the back end
SyncBO Builder	Defining the SyncBOs (middleware transaction merep_sbuilder)
SAP NetWeaver Developer Studio	Creating or enhancing the mobile client application with the Mobile Development Kit (MDK) view
JUnit	Open source tool for creating unit tests to ensure the quality of the client application (http://www.junit.org)
FindBugs	Open source tool for detecting bugs in Java source code (http://findbugs.sourceforge.net/)

Figure 8 Development tools used in SAP NetWeaver Mobile application development

mobile devices, we once again face this challenge and must relearn the art of efficient development.

The development of a custom SAP NetWeaver Mobile application is a complex process and involves several components: the client application, the SyncBOs on the middleware, and the BAPI wrapper in the back end. Covering the entire development process is beyond the scope of this article, so we'll provide a general overview to give you an idea of what's involved.[3] We'll start with a high-level look at the steps you have to follow to implement a SmartSync or a GenericSync application. **Figure 8** summarizes the tools used, including some open source tools for debugging that we have found helpful in our experience.

The development of an SAP NetWeaver Mobile application based on SmartSync usually involves the following steps:

1. Implement the BAPI wrapper in the back-end system.
2. Create the SyncBOs on the middleware server.
3. Develop the mobile client (no synchronization logic is required, as this will be handled by the SAP NetWeaver Mobile framework).
4. Test and debug the application.

[3] For details on developing mobile applications, see the MDK information available at SDN (http://sdn.sap.com).

For a GenericSync application, the following steps are required:

1. Implement function modules in the back-end system. These function modules will read and write the data containers that are exchanged with the mobile client.
2. Develop the mobile client application. In this case, the synchronization logic (what to put into the data containers and how to read their content) has to be written on the client as well.
3. Test and debug the application.

Regardless of the approach you use, before you start the development, you need to know the following:

- **The target platform:** The target platform for the application not only has technical implications (Windows laptops and tablet PCs require JRE 1.4.x, while Windows Mobile-based PDA devices and smart phones require JRE 1.1.8), it also determines how the UI will need to look — PDAs have a much smaller screen size than laptops, for example, so the UI will need to be much simpler.
- **The UI type:** You will also need to decide whether you want to develop the UI using JSP or AWT. While JSP-based applications are displayed in a Web browser, AWT applications have a native UI. Our recommendation is to use AWT, as the performance and the usability are much higher.

Enhancing SAP-provided prepackaged applications

Let's say you would like to use one of the SAP-provided applications, but there are components in the application you would like to change to meet your particular business needs. Perhaps you want to adjust the UI by hiding some links that are of no use to your end users or add a new functionality, such as additional checks on data entered by end users. You can easily do this by enhancing the application. Most of the SAP mobile applications — xMAM 2.5 and 3.0, xMAM utilities 3.0, xMSA HH 5.0, and xMTT 2.0 — are based on a common framework called the Mobile Client Application Framework (mCAF).[4]

The mCAF facilitates the enhancement of the prepackaged SAP applications by offering an easy way to migrate the enhancements you've made to new service releases of the application. In the past, the new version of the application would overwrite the enhanced version, and you would have to re-create the enhancements in the new version. With the mCAF, you can just apply the enhancements to the new service release by following these simple steps:[5]

1. Make a backup of your enhancements.
2. Import the new service release into SAP NetWeaver Developer Studio.
3. Copy your enhancements into the project.
4. Export the new Web archive (WAR) file of the application.

It's really that easy! The mCAF makes all of this possible by adhering to the Model-View-Controller (MVC) design pattern, which separates the application data from the UI so that changes to one do not affect the other (see the sidebar on the following page for more details).

While the details of enhancing an application are beyond the scope of this article, we'll provide you with a brief overview of the steps involved.[6] To create enhancements, you need to do the following:

1. Import the WAR file of the prepackaged SAP application via the MDK into SAP NetWeaver Developer Studio.
2. Create the triggers for your enhancements:
   - zcore.configure: Here you define the business logic you would like to enhance.
   - view.configure: Here you define the UIs and workflows you want to change.
3. Create the enhancements:
   - For business logic changes, create your enhancements by extending existing Java classes or by creating new ones.
   - For UI and workflow changes, extend the mCAF view files and the corresponding Java classes and JSPs.
4. Test, run, and debug your application from within SAP NetWeaver Developer Studio.
5. Export a WAR file that includes your enhancements.

> ***Note!***
>
> When you enhance a mobile application that should run on a PDA, you need to make sure that you only use Java classes that are available in Java 1.1.8. Otherwise the application will not run on the PDA. The best approach would be to install an additional JDK 1.1.8, which is available free of charge from Sun Microsystems (http://java.sun.com).

[4] The SAP mobile applications xMDSD and xMTT 1.6 are based on the AWT and microITS UI technologies, respectively, rather than on the mCAF. For details on how to enhance these applications, see the enhancement guides available at the SAP Service Marketplace.

[5] For more details on migrating enhancements using the mCAF, please refer to the application-specific enhancement guides from the SAP Service Marketplace.

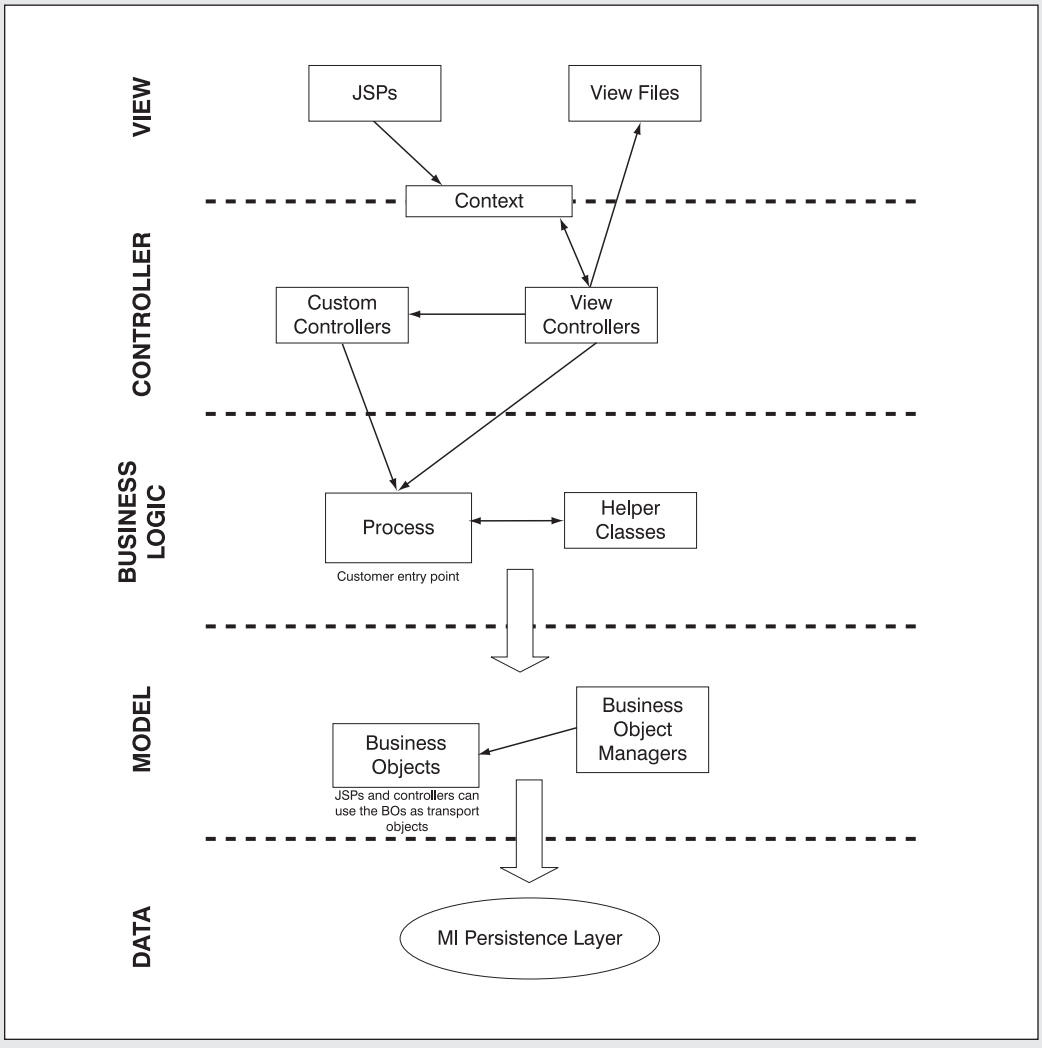
[6] For details on enhancing mobile applications, see the mobile application enhancement guides, which can be downloaded from the SAP Service Marketplace.

Mobile Client Application Framework (mCAF)

The Mobile Client Application Framework (mCAF) is based on the Model-View-Controller (MVC) concept. According to the MVC design pattern, an application consists of a model layer, a view layer, and a controller layer.

The goal of the MVC concept is to remove data (model) and UI (view) interdependencies, so changes to the UI do not affect the data handling and the data can be reorganized without needing to change the UI. The MVC design pattern solves this problem by decoupling data access and business logic from data presentation and user interaction, and introducing an intermediate component: the controller.

The following diagram summarizes how this works.



Let's take a closer look at the key elements of MVC:

- The **view** layer displays context data provided by the controller layer. View files describe the navigation flow and are stored in XML format. The compiling of the JSPs that actually hold the source code of the screens generates the different application screens. One advantage of this approach is that customers can use their own JSPs without any need to recompile the controller code.
- The **controller** layer is made up of Java classes. For each JSP there is a Java class — a view controller — that implements methods for the different events that can be triggered from the JSP. The view controller accesses the model layer and uses business objects from the model layer to transport data to and from the context of a view. View controllers are responsible for preparing the context of a view, for controlling application navigation by handling view events from JSPs, and for processing user input. Custom controllers are responsible for sharing data among views and components.
- The **model** layer consists of business object managers that encapsulate business process logic. This is the entry point for view controllers and for customers who want to reuse the business logic encapsulated in these objects. Business objects are used to transport the data fetched by the business object managers to the view layer, and they don't contain business logic — they are just containers for data.

The mCAF implements the MVC concept and facilitates enhancements with a clear separation of the different layers, which decouples data access and business logic from data presentation and user interaction. This makes it easy to enhance certain parts of the application without changing the behavior of other parts.

Now that your mobile solution is ready to go, it's time to make sure that your landscape is secure so that you can use it.

Phase 7: Secure your mobile landscape

Security is an important factor to consider in highly distributed environments. With the distributed nature of mobile landscapes, it is critical that you prevent security breaches by making sure that you secure data on the devices and that someone cannot misuse a stolen device to access your company network.

Securing a mobile landscape can be managed in a number of ways, including using encryption for the synchronization and for the data on the mobile device, using passwords on the mobile device, and configuring the RFC connection to the back end. Let's review each of these options, which are summarized in **Figure 9** on the next page.

Encrypting the data transfer/ synchronization

The data exchanged between the SAP NetWeaver Mobile client and the middleware can be SSL encrypted. This is recommended if the synchronization is done via an unsecured network like the Internet. The disadvantage of the encryption is that it requires server and client time and computing power, which can lead to problems on small devices and

Security method	Requirements	When to use	Advantages	Disadvantages
Encrypting the data transfer/ synchronization	Installation of the SSL add-on on the mobile client	Synchronization over an unsecure network (Internet)	Secure communication	Slows down the performance of the synchronization
Encrypting the data on the mobile device	Use of DB2e on the mobile client	If the mobile application contains highly confidential data	If the device is lost or stolen, data cannot be read by directly accessing the database	Slows down the data access and therefore the whole solution
Using passwords on the mobile device	Configuration of the password functionality on the mobile client	If the mobile application contains confidential data; depending on the synchronization frequency, different levels of password checks can be activated	Only authorized users can log into the application; if the device is lost or stolen, data cannot be read without logging in	Depending on the settings, passwords might have to be entered quite often
Configuring the RFC connection to the back end	Sufficient user rights in the middleware and back-end systems	Always required; trusted connection is recommended	Always required	Difficult password handling if no trusted connection is used

Figure 9 Methods for securing your SAP NetWeaver Mobile landscape

when a large number of users connect to the server at the same time. If your users connect via the company network or via VPN (Virtual Private Network), the SSL encryption is not required.

> ***Note!***
>
> You need to install the SSL add-on on the mobile client before you can use SSL. The SSL add-on is available from the SAP Service Marketplace.

Encrypting the data on the device

It is possible to encrypt the data that is stored on the client device. However, this will lead to lower application performance, as data access is significantly slower.

> ***Note!***
>
> Encryption is only possible if you select DB2e for the persistence layer; file I/O does not support it.

Using passwords on the mobile device

When you start the SAP NetWeaver Mobile client, you have to log into the device via a user name and password that you create the first time you log in.

You can also configure the SAP NetWeaver Mobile client to require a password for synchronization. (The administrator needs to complete this configuration before deploying the SAP NetWeaver Mobile client.) The following password options are available:

- **Local:** The synchronization password corresponds to the local login password and does not need to be entered for synchronization.
- **atSync:** The synchronization password does not correspond to the local login password and must be entered for each synchronization.
- **Once:** The synchronization password does not correspond to the local login password and must be entered once for the first synchronization after each login.
- **Single Sign-On (SSO):** The user never needs to enter a password. The user information will be received from a ticket-issuing system (for example, SAP Enterprise Portal).

The default configuration is atSync, although if the users synchronize frequently, this configuration can be cumbersome because users will have to reenter their passwords repeatedly. In this case, a local configuration might make more sense, although keep in mind that even with a local synchronization password, the password is not actually stored on the mobile device. The SAP NetWeaver Mobile client is calculating a hash key from the password and storing it on the mobile device. If the user enters the password again to log into the SAP NetWeaver Mobile client, the hash key is created again and compared to the old hash key. If they match, the user is allowed to use the client. If you have an SSO environment in your company, it is a good idea to implement the SSO feature of SAP NetWeaver Mobile as well.

Configuring the RFC connection to the back end

The communication between the middleware and back-end server is done via RFC. There are three options for configuring the RFC connection:

- **Trusted Connection:** If the middleware and the back-end system are configured as trusted systems, the communication does not require a password check. The systems trust each other, so only the user names will be passed to the other system. If the user exists there, access is granted.
- **System User:** With this option, a system user will be used in the configuration of the trusted connection. The user name and password must be specified in the RFC connection. This option may not work for all GenericSync applications, as the application could use the passed user information as a criterion for how to store the data.
- **Login User:** With this configuration, the logged-in user (the SAP NetWeaver Mobile user logged into the SAP NetWeaver Mobile client) is used to connect to the back-end system. If this option is selected, the passwords of the user must be the same in both the middleware and the back end.

A trusted connection is the recommended configuration because it is the most flexible one. With a trusted connection, the passwords in the middleware and the back end don't need to match, which makes things easier because users can change their passwords on both systems independently.

Once you have secured your landscape, you are ready to test the application in preparation for deployment to your users.

Phase 8: Test the mobile application

In mobile projects, it is particularly important to ensure the quality of a solution prior to go-live. While SAP NetWeaver Mobile offers a number of advanced monitoring and tracing tools that greatly facilitate the detection of problems and the deployment of fixes, it's still best to avoid problems as much as possible beforehand. Bugs in enhancements to SAP prepackaged applications or in your own developed mobile applications are harder to detect than in traditional IT environments, and it is a bigger effort to fix those bugs because patches have to be deployed to every single device out in the field.

The best way to avoid problems is to invest time and resources into good quality management. The key to managing software quality is to identify the correctness, completeness, security, and quality of software through software testing — both *white box* testing,

Application scenario	Tests to perform
Prepackaged SAP application like xMAM or xMTT with minor enhancements	• Black box unit tests of your enhancements should be sufficient • Go top-down to integrate and test your enhancements one after the other • Be sure to perform a system test as a last step to make sure that your enhancements work and that there are no side effects
Prepackaged SAP application like xMAM or xMTT with major enhancements in lower layers	• Black box and white box unit tests • Use a hybrid approach to integration testing — combine the top-down and the bottom-up approaches to best fit your specific enhancements (i.e., use a top-down approach but use bottom-up in lower layers if needed) • Be sure to perform an extensive system test as a last step to make sure that your enhancements work and that there are no side effects
Custom-developed SAP NetWeaver Mobile application	• Black box and white box unit tests • Go bottom-up to integrate and test one component after the other, ensuring that lower levels of the application work as planned • Be sure to perform an extensive system test as a last step to make sure the application works as expected

Figure 10 Key tests for mobile application scenarios

which is an internal view of the test object, and *black box* testing, which is an external view. White box testing (structural testing) is used for testing internal paths through the software and therefore requires some knowledge of the coding — the tester needs to know in detail how the software works. Black box testing (functional testing), on the other hand, is used for testing the software purely from the user's perspective and therefore requires no internal knowledge of the item being tested. Black box testing is useful for simplifying the testing of complex software.

So what is so different about testing mobile applications? Methodology-wise, there is no difference. You probably start by doing some black box and white box unit tests. After you have successfully tested all the units you will probably start by doing some integration tests of the units. The type of integration test you perform — bottom-up or top-down — will depend on whether you created or enhanced the application:

- If you have developed an application from scratch, you'll want to use the bottom-up methodology — i.e., an incremental testing strategy starting from the very lowest levels and then working upward as you integrate successive layers.
- If you have been enhancing an SAP prepackaged application, you'll want to use the top-down methodology — i.e., an incremental testing strategy in which you start at the top part of the application to be enhanced and work your way down through the underlying components. The underlying components are mostly SAP standard components, but some are placeholders for custom-developed enhancements. These custom-developed enhancements are first simulated by stubs (dummies) until they are later replaced by the enhanced components. So, for example, you would use the SAP prepackaged application as a starting point, add your UI enhancements first, and then work your way down through the stubs for the lower-level components, integrating one component after the other by replacing the stubs.

After integrating all the units, you will then perform a system test to ensure that everything is working properly and there are no adverse effects from the enhancements. **Figure 10** summarizes the most important testing to perform for mobile application scenarios.

What makes testing mobile applications different from testing standard applications is that the system

Training area	Recommended training
Device	Using the hardware, including everyday things like charging batteries, starting up the device, or connecting to the network
Operating system	Operating system basics, such as using the virtual keyboard on a PDA
Application	Application-specific training, such as giving a functionality overview or learning how to use the application in detail
Synchronization process	What options for synchonization are there ? How can the end user use these options?
Support process	What happens if issues arise? Where can the user go for help?

Figure 11 Key training areas

landscape is more complex, there are more technologies involved, and more things need to be tested. Executing the necessary test cases is also often more time consuming because it might require synchronizations. Keep the following recommendations in mind for handling the additional challenges of testing in a mobile landscape:

- **Write unit tests:** Ensure that each piece of your source code is covered by a unit test to ensure quality.
- **Test on the target device:** Testing on a developer workstation using a device emulator gives an inaccurate impression of the performance. In addition, the device emulator might not exactly match the target device.
- **Test repeatedly throughout the development cycle:** Errors can be found and issues can be identified early in the process.
- **Plan more time for testing:** Mobile environments are more complex than traditional environments. This results in longer test execution times.
- **Involve key users early:** Key users can test the application on the target device and provide feedback on the behavior of the application and its performance.
- **Include a pilot phase:** Because errors in mobile environments are hard to fix, it is smart to have a pilot group with a small number of users run the application for productive use.

The last point cannot be stressed enough: prior to the big rollout, a pilot group should run the application for productive use. Based on the feedback from the pilot group, the project team has the chance to perform final modifications. At the same time, the administration team managing the landscape will be able to familiarize themselves with supporting the solution while working with a small number of experienced users.

With testing complete and your application ready to go, it's time to train your users in preparation for deployment and go-live.

Phase 9: Train your users

If you introduce new software, such as SAP NetWeaver Mobile, and new hardware devices, you need to train your users in both areas. New hardware, new operating systems, and new applications can be challenging for anybody, but especially for those who may not work with computers on a regular basis.

Many service technicians are technically skilled, but may have never worked with a tablet PC before, for example. Therefore, the need to train them in the use of a tablet PC and its operating system is of great importance.

Figure 11 summarizes the most important areas in which you need to be sure to train users.

Keep in mind that users learn best when actually doing things hands-on. It might be a good idea to

either train your end users in a lab environment or have a trainer onsite with them for their first day using the application.

We have finally reached our goal. There's only one thing left to do — deploy the application.

Phase 10: Deploy the mobile application

The rollout of the new mobile application needs to be well prepared. This is especially important if users are being exposed to the selected mobile device and the application for the first time. Remember, the first impression is very important! It is very difficult to change a user's mind once he or she has decided an application is problematic.

Because mobile applications span several systems (the mobile device, SAP NetWeaver AS, and the back-end systems), software logistics for mobile applications can be challenging. SAP NetWeaver Mobile offers a sophisticated application deployment concept to centrally manage such a fleet of devices.

The administration of the entire mobile landscape is done with the help of SAP NetWeaver Mobile and relies on tools provided in the SAP NetWeaver Mobile Administrator, shown in **Figure 12**. (The SAP NetWeaver Mobile Administrator has replaced the SAP WebConsole used for administration purposes in earlier releases.) The system administrator can monitor data exchange and administrate the mobile landscape using a single browser window. The SAP NetWeaver Mobile Administrator also allows for centrally controlled software deployment and updates without user interaction, and supports complex software assignments through the use of hierarchy groups, which are a logical grouping of multiple users to which you can assign applications or setup packages.

It is important to understand that the synchronization mechanism of SAP NetWeaver Mobile handles not only the transfer of application data but also the actual deployment of software components to the target mobile devices. With the initial synchronization, the required software components are downloaded to the target mobile device — this is how applications such as xMAM are installed. After the initial installation, a second synchronization synchronizes the SAP NetWeaver Mobile application data to the device. All of this happens outside of the user's perspective — he or she simply clicks on a button.

In order to deploy additional applications, upgrades, patches, and add-ons such as databases, device drivers, and security libraries, the administrator must first upload the corresponding software components to the middleware and then assign them to a target device, user, or group of users. Once a target user synchronizes his or her mobile device, the assigned components are downloaded to this specific device and installed without any further interaction from the end user. In this way, you can deploy mobile software components to specific mobile devices.

SAP NetWeaver Mobile offers two ways to roll out solutions to your users: the classic method and the setup packages method.

The classic method is to manually execute several synchronization cycles and device reboots. The components to deploy on a device or a group of devices are assigned using the SAP NetWeaver Mobile Administrator. The advantages of the classic method are that it is more straightforward and requires less work from the administrator. The downsides are that it might require device reboots and the process might not be clear to the end user.

With the classic method, you also need to think about the initial installation of the SAP NetWeaver Mobile framework on the devices. The framework is installed on a laptop or tablet PC via a setup.exe file and on a PDA or smart phone via a CAB file. For a 32-bit Windows device such as a laptop or a tablet PC, it makes sense to use a software management solution (such as Microsoft SMS, enteo Netinstall, or IBM Tivoli) to install the framework. The management of PDAs, on the other hand, is something completely new, and largely depends on the number of devices you need to install. If it is just a small number, then we recommend doing the installation by hand via a secure digital (SD) card. (Note that the administrator should do this, not the end user.) If you plan to install a large number of PDAs, we suggest

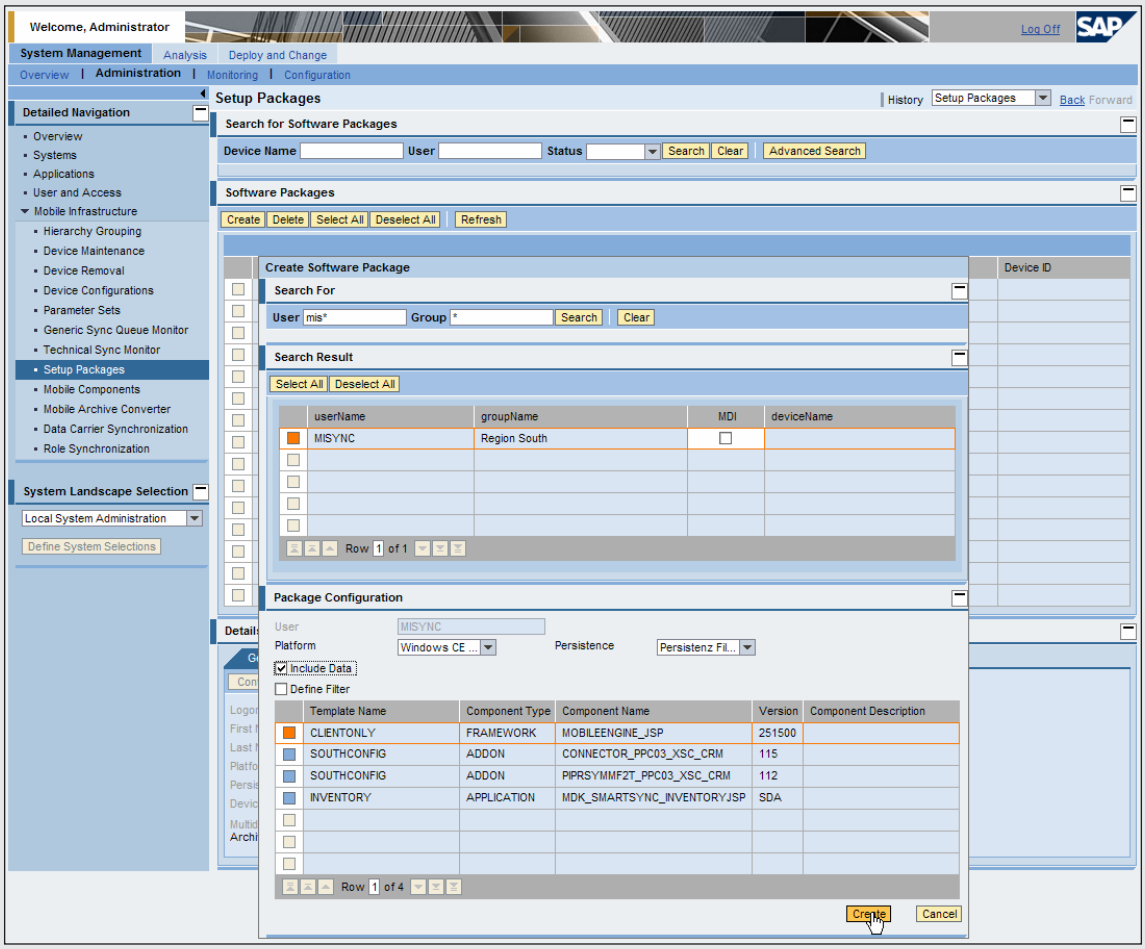


Figure 12 The SAP NetWeaver Mobile Administrator

using a software management system. Most solutions for the 32-bit Windows platform also provide add-ons that facilitate the management of PDAs. We advise preconfiguring the framework on mobile devices so that the configuration work does not need to be performed by the end user. An administrator can add the necessary settings that tell the mobile device which server to connect to.

After the framework is installed on the mobile device, it is ready for use. The user can then log in and perform the first synchronization. During this and each of the following synchronizations, the SAP NetWeaver Mobile client is sending status information to the middleware server. The middleware checks if there are any mobile applications assigned to the synchronizing user. If this is the case, the application files are downloaded to the mobile device and installed. After a restart of the SAP NetWeaver Mobile client, the applications are available for use.

As you can see, while the classic method is quicker for development and testing purposes, it is a bit cumbersome because almost no administrative steps are necessary in the middleware. The overall faster and recommended alternative is to use *setup packages*. Setup packages are assembled on the SAP NetWeaver Mobile server using the SAP NetWeaver

An implementation project example

Let's take a look at how the 10 implementation phases might be applied at a utility company that wants to implement SAP xApp Mobile Asset Management (xMAM) 3.0 to enable their 800 field technicians and service engineers to manage service orders, notifications, and machine maintenance, while they are only occasionally connected.

1. **Gather requirements and analyze processes:** A group of key users is questioned about the current paper-based process. Based on the feedback from these key users, a first, rough mobile scenario is created.
2. **Create a plan and assemble the project team:** The project plan is created by the project lead, who also selects the different experts needed for the project team, including a PLM back-end specialist, an xMAM development specialist, a technical specialist for the middleware setup, and an SAP NetWeaver Mobile specialist.
3. **Select the target mobile device:** A ruggedized PDA is selected as the target device, as the team and the end users felt that a tablet PC would be too big and heavy. During the selection process, key end users are involved to make sure that the general end user population later accepts the selected device. The key end users were provided with five different preselected devices to test and choose from.
4. **Choose a connection type:** As most of the technicians visit the office only once a week, but need to synchronize their orders daily, 3G is selected as the connection type. This has the advantage that users can update their data after every order they work on. The team selected a mobile phone provider that allows a direct tunnel into the company network, eliminating the need for an SSL-encrypted connection, as well as the need for SAP NetWeaver AS to be opened to the Internet, which can be risky from a security perspective.

Mobile Administrator and can contain the SAP NetWeaver Mobile client, applications, add-ons, and application data, along with hierarchy group assignments that determine the users who will receive the package. The setup packages are then transferred to the designated mobile devices and extracted. The transfer can either take place by network connection or by using a CD or an SD card. Although this method takes more time in creating setup packages, the time invested is well worth it because it facilitates quick and easy installation for end users.

Keep in mind that the deployment step is complex and should be tested under the same conditions as the final rollout. It is a good idea to include some of your end users in these tests too.

Note!

Both the classic and setup package installation methods require that the mobile device have a JRE installed (JRE 1.4.x on Windows laptops and tablet PCs and JRE 1.1.8 on Windows Mobile-based PDA devices and smart phones).

We've now reached the end of our tour of the 10 key phases of a mobile implementation project (see the sidebar above for a look at how this all might look in a simple example project). But we're not done yet!

5. **Set up and configure the mobile landscape:** The technical specialist and the SAP NetWeaver Mobile specialist on the project team, together with the administrators of the system landscape, configure the mobile landscape. Because the administrators need to run the landscape after go-live, they were involved early in the process to secure the knowledge transfer.
6. **Implement enhancements:** The application's screenflow is changed and the UI is adjusted to reflect the utility company's corporate design. Multiple pages have been combined to reduce the amount of clicks that are required to go through the process. In addition, unnecessary functionality has been removed from the application to present the end user only with relevant functionality and screens.
7. **Secure your mobile landscape:** The data that is stored on the mobile device is not classified as extremely confidential, so it is not encrypted on the mobile device. The synchronization is also not encrypted as the device is directly tunneled into the customer network.
8. **Test the mobile application:** The same group of end users that chose the device using a small pilot project with real-world data test the solution. The resulting feedback is incorporated into the application and some additional changes are made based on the feedback.
9. **Train your users:** The project team creates end user documentation and training materials. All end users attend workshops where the application is demonstrated and users can practice using it on the target device. In addition, the key end users who already gained experience with the application during the pilot project can help support new users.
10. **Deploy the mobile application:** The project team together with the system administrators implement the initial deployment. The software is installed via setup packages on the 800 devices. In addition, copies of the setup packages are distributed to end users on SD cards so that they can be reinstalled if the devices crash.

There's one more critical aspect of mobile implementationprojects that is often neglected — operating the environment post go-live. We'll round out our discussion by taking a look at this important task next.

Operating a mobile environment

Introducing mobile applications to an existing system landscape adds a new level of complexity with respect to operations, since a system administrator cannot physically access the mobile devices. The demands and expectations regarding efficient and smooth operation are high, and disruptions to everyday field activities can have a major impact on the entire organization.

Operating a mobile environment means dealing with what we like to call the *mobile challenges*: mass rollout, deployment of patches, hot fixes, upgrades to the mobile landscape, monitoring, troubleshooting, data maintenance, and restoring devices. Sure, this has to be done in traditional IT environments as well, but what makes this much harder in mobile environments is the distributed nature of mobile landscapes. The SAP NetWeaver Mobile Administrator provides you with all the necessary tools to meet these challenges.

In addition to using the SAP NetWeaver Mobile Administrator, the following strategies can help ensure a smooth-running mobile landscape:

1. **Device selection is crucial** — if you choose the wrong device or drastically economize on the hardware, you might pay the price later. Have a look at a variety of devices and choose one that fits your needs. Invest time in this activity, especially when you think about procuring PDA-based devices. This will help keep operating costs low.

2. **Good planning is essential** and strategies on how to deal with potential issues must be in place. Mistakes are hard to fix because this might require rolling out the software to potentially thousands of devices again.

3. **Think carefully when administering productive devices** — you might accidentally delete a device that is still in use, or worse, you could delete a large number of devices that are still in use.

4. **Invest time and resources into support**, and ensure that your support personnel are able to access devices either directly or by remote login. If no such option exists, develop simple procedures that empower users to reinstall their devices to resolve issues.

5. **Educate your end users well** to prevent issues before they occur. This will reduce the workload for the support team, as well as improve the end user experience.

General guidelines for mobile projects

Following these general guidelines can make a major difference and highly increase the likelihood that a project is going to be a success:

- **Buy-in:** Mobilizing your environment requires a shift in the way a company does business and perform project implementations. Due to the various technologies that make up a mobile landscape, many different stakeholders are involved. Buy-in from all parties is required for a project to be successful.

- **End user acceptance:** End users are a key factor in the adoption of mobile applications. If end users enjoy using the mobile application and see an advantage in this new way of working, this will contribute significantly to the success of the project. Therefore, involve your end users as early as possible. This is especially important when selecting devices and adjusting the UI to suit end user needs. Involvement of end users from the very beginning is a key factor in the success of the project. This will ensure that the delivered solution matches the needs and expectations of the users.

- **Device selection:** The mobile device you select heavily influences performance, stability, and look and feel. Choosing a device that is too slow results in frustrated end users. Selecting a device that is not durable enough results in a lot of downtime. If the wrong device is chosen or if the choice of hardware is based on economizing, this could affect the performance of the application. Therefore, evaluate a variety of devices and choose one that fits your needs. Invest time in this activity, especially when dealing with handheld devices.

- **Project team expertise:** Whether the customer, a partner, or SAP drives the implementation, it is critical to involve expert resources in the project team because mobile technology is relatively new. This will ensure adequate expertise as well as proper knowledge transfer for a successful implementation and enable customers to operate and maintain the landscape themselves after go-live.

- **Quality management:** Quality management is very important because it can be a challenge to address issues in a mobile environment. Once the application has been rolled out, it is not always easy to access devices that are out in the field. Also keep in mind that the number of

A sneak peek at SAP NetWeaver Mobile 7.1

SAP NetWeaver Mobile 7.1 will be the next version of SAP NetWeaver Mobile. It offers significant improvements in terms of data synchronization performance and also comes with new tools that simplify development. It also has improved mobile device management capabilities to minimize the effort of managing mobile devices.

Some of the highlights of the new product are:

- Higher scalability to allow for the handling of more mobile devices and larger volumes of data
- New highly optimized data realignment algorithms
- Development of mobile components using the Web Dynpro framework, resulting in shorter development cycles and reduced total cost of development
- Improved mobile device management, monitoring, and security
- Ability to track security incidents, such as failed application login attempts, and take preventative action by blocking the synchronization of a device

To protect customer investments, SAP NetWeaver Mobile 7.1 comes with a compatibility layer for existing mobile applications. This layer allows you to continue using mobile applications built on earlier versions of SAP NetWeaver Mobile (e.g., SAP Mobile Infrastructure 2.5 and SAP NetWeaver Mobile 7.0).

SAP NetWeaver Mobile 7.1 is currently in ramp-up and will most likely be generally available in early 2008.

devices in a mobile environment can be very high. Proper planning is essential, especially when it comes to defining strategies for dealing with potential issues.

- **Support process:** Invest a high percentage of your time and resources in support and ensure that your support team is able to access devices either directly or through remote login. If this is not possible, develop simple procedures to enable users to reinstall their devices themselves to solve issues.
- **Sizing:** Make sure that server sizing fits the requirements and the performance is adequate. This keeps synchronization times low and ultimately makes the end users happy.

Conclusion

Mobile solutions are changing the way we think about business software. We have to think not only about desktops and servers, but also about laptops, tablet PCs, PDAs, smart phones, and other devices —

not to mention devices down the road that we can't even imagine at the moment, such as wearable devices.

According to Gartner, in 2004 2% to 5% of IT budgets were spent on mobile implementations; in 2008 it will be around 8% to 10%. In the US alone there are 60 million people — representing 45% of the overall workforce — who spend 20% or more of their on-the-job time outside of the office. To avoid gaps in business processes, it is a question of when, not if, mobile solutions will become mainstream. With knowledge of SAP NetWeaver Mobile 7.0, and how to successfully implement it, you will be well prepared for this next step in enterprise IT.

Alexander Ilg first came into contact with mobile software in 1997 when he implemented a mobile software solution for DaimlerChrysler. He has worked with SAP's mobile software since 2002 an d has been involved in numerous mobile projects, bringing more than 100,000 mobile users live. Alexander was part of the team that implemented the SAP NetWeaver Mobile xMTT and xMAM solutions. In 2006, he founded msc mobile ltd., which focuses on implementing easy-to-use mobile solutions. He can be reached at alexander.ilg@msc-mobile.com.

Karsten Strothmann is a Regional Group Expert at SAP AG. He specializes in the SAP NetWeaver Mobile-based applications xMTT, xMAM, and xMAM utilities, and has seven years of experience at SAP. In the last three years, Karsten has coached more than a hundred mobile projects, bringing thousands of users live. He holds a master's degree in Computer Science from Dortmund University and has been a key speaker at several SAP TechEd and SAP Skills Conferences. You can reach him at k.strothmann@sap.com.

