mySAP CRM

Detailed View:
SAP Mobile Application Studio
(Release 4.0)
Content

Introduction

Challenges

SAP Mobile Application Studio within mySAP CRM

Technical Capabilities of SAP Mobile Application Studio
- Building enhanced Mobile Client Applications
- Version management
- Integration into CRM system landscape
- Software Logistics
- Multilanguage and translation support

Conclusion
Content

- Introduction
- Challenges
- SAP Mobile Application Studio within mySAP CRM
- Technical Capabilities of SAP Mobile Application Studio
  - Building enhanced Mobile Client Applications
  - Version management
  - Integration into CRM system landscape
  - Software Logistics
  - Multilanguage and translation support
- Conclusion
mySAP CRM Release Roadmap 2002 / 2003

mySAP CRM 3.x
- Complete operational multi-channel SAP CRM
- Support of Connected customer-centric end-to-end processes across the mySAP.com E-Business platform
- Leading SAP CRM Analytics
- People-Centric SAP CRM - Ease of use through pre-configured roles for employees, customers, partners

mySAP CRM 4.0
- Leading-edge scenarios for numerous industries
- Industry-specific people-centric roles
- Enhanced SAP CRM core functionality for marketing, sales, service and analytics
- Interaction Center WebClient
- Collaborative channel management scenario

Industry-specific processes
Today’s complex customer problems require a deployable CRM solution that can directly address specific challenges regardless of where or when they occur in the cycle of interacting with, selling to and servicing an organization’s customers.

mySAP CRM blends deep functional capabilities in the core areas of Marketing, Sales and Service with award-winning Analytics that are directly embedded into the primary interaction channels with which organizations engage their customers.

All of this enables the closed loop interaction cycle, underlined mySAP CRM’s unique value propositions.

mySAP CRM is built on an open, reliable, secure and scalable technology platform

The strong and complete services offering of SAP helps to quickly implement mSAP CRM and supports the ongoing optimization of the solution environment.
The CRM server with the CRM middleware and CRM applications is at the core of the mySAP CRM system landscape. It takes advantage of all the benefits of SAP Web Application Server, including load balancing, high availability, integration, and security. The CRM server can run as a stand-alone solution or with various back-end systems. If SAP R/3 is the back-end system, only a plug-in is needed to handle data exchange with the SAP CRM server. CRM middleware handles data exchange with third-party systems.

mySAP CRM supports thin clients for user access. Without additional software installation on the client side, all relevant functions are accessible in a standard browser. You can also use SAP GUI for Windows.

mySAP CRM supports two types of mobile scenarios: laptops and handheld devices. Laptops employ the solutions for mobile sales and mobile service, which are synchronized via the CRM middleware with the SAP CRM server applications. Handheld solutions can be accessed in online or offline mode.

mySAP CRM is integrated with various other SAP solutions. For example, SAP® Business Information Warehouse increases the value of each CRM installation with reporting and data warehousing tools for analytical CRM. Efficient marketing planning and accurate ROI calculations are great advantages. It is also integrated with SAP® Advanced Planner & Optimizer (SAP APO), which allows service to the customer along the whole supply chain. Quick available-to-promise (ATP) checks for configurable and nonconfigurable products, production on demand, and delivery in time help satisfy customer needs.

The exchange of information between mySAP CRM and other components, such as SAP BW and ERP back-end systems occurs over the CRM middleware.
Challenges - SAP Mobile Application Studio

- Easy enhancement and new development of Mobile Client Applications
  - One tool to develop business logic as well as screen design

- Support of multi-user development: two or more users can access the same object and continue developing simultaneously without interfering with each other’s work

- Software logistics support
  - Transport of development objects between system landscapes
  - Easy deployment of new or enhanced applications to the laptop computers in the field

- Integration into the CRM system landscape
Content

- Introduction
- Challenges
- SAP Mobile Application Studio within mySAP CRM

- Technical Capabilities of SAP Mobile Application Studio
  - Building enhanced Mobile Client Applications
  - Version management
  - Integration into CRM system landscape
  - Software Logistics
  - Multilanguage and translation support

- Conclusion
Business Scenario

You are a member of the project team for an SAP CRM Mobile Sales implementation project at SAPCOE Inc.

You are using the SAP Mobile Application Studio to enhance and modify the mobile sales system.

To adjust your mobile sales system to SAPCOE’s requirements, you want to:

- Change the look and feel of the application
- Redesign some screens
- Add fields, icons, and checkboxes
- Hide complex functions
- Extend the business logic
How Does the SAP Mobile Application Studio Simplify Your Work?

- Visual modeling reduces manual coding to a minimum
- Easy to learn and use
- Effortless generation of Mobile Client Application
- Change management and modification support
- Software logistics
- Multilingual and translation capability
- Storage of metadata in Mobile Application Repository
What kind of technology does the Mobile Application Studio use?

- Business Object Model (UML)
- Object-oriented data binding
  - Automatically follows links and references
  - Reproduces changes to referenced objects and instances
- Object-oriented repository
Content

- Introduction
- Challenges
- SAP Mobile Application Studio within mySAP CRM
- Technical Capabilities of SAP Mobile Application Studio
  - Building enhanced Mobile Client Applications
  - Version management
  - Integration into CRM system landscape
  - Software Logistics
  - Multilanguage and translation support
- Conclusion
You can enhance the standard application ...

Standard:
SAP CRM Mobile Sales

Modification
SAP Mobile Application Studio

Enhanced:
SAP CRM Mobile Sales

...Or design an entirely new application:

Design
SAP Mobile Application Studio

Generate

New Application

As a rule, the starting point of a Mobile Client Application (MCA) implementation project is the delivery standard – that is, a template providing the full range of MCA features. Normally, the project team does not design an application “from scratch.” After the superfluous interaction components have been removed, your application is enhanced as required. There is an extension of this idea called Industry Templates for delivering solutions for verticals. This is work in progress.
The MAS Framework covers the complete business logic of your application.

Based on this business object model, most interaction component programming can be done using declarations. Layout definitions and data flow definitions are completely declared.

On the basis of these declarative definitions, the interaction components already provide a wide range of generic functions:
- Data binding between business objects and tiles in a tile set
- Data flow automation inside the interaction component hierarchy
- Navigation handling (navigation bar, hyperlinks, scrolling through collections)

These generic functions can be enhanced by customized behavior definitions if necessary.
### SAP Mobile Application Studio: Typical Scenarios

#### Context:
- Designing a mobile solution
- Enhancing mobile solution functions

<table>
<thead>
<tr>
<th>To do this...</th>
<th>You use these MAS tools...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design the screens (tile, tile set, business component, application)</td>
<td>Object Browser, Tile Wizard and Tile Set Wizard</td>
</tr>
<tr>
<td>Model the underlying business logic (business objects, relationships, queries, business rules)</td>
<td>Object Browser</td>
</tr>
<tr>
<td>[Change management:] Track changes, view previous versions, analyze differences, resolve collisions</td>
<td>Change list Browser, Collision Browser, Version Tree</td>
</tr>
<tr>
<td>Generator</td>
<td>Generator</td>
</tr>
<tr>
<td>[Software logistics:] Transport development objects to other repositories or systems, manage upgrades</td>
<td>Upgrade Console, Transport Tool</td>
</tr>
<tr>
<td>Create system messages, translate interface text</td>
<td>Message Editor, Translation Tool</td>
</tr>
<tr>
<td>Logically group development objects to make them industry-specific</td>
<td>Template Modeler</td>
</tr>
</tbody>
</table>
Easy Screen Design: Two-Part Procedure

1 **BEFORE**

With a single drag & drop, you create a screen. No additional manual interaction is required.

2 **AFTER**

Save and generate.
Wizards Support More Complex Steps

- Use drag and drop
  - To arrange tiles in a tile set
  - To model anchors between tiles or tile sets
New Generation Features

New: Start the generation
- From Object Browser
- From Generation List
Mobile Client Application Development

- Design time environment for declarative programming (SAP Mobile Application Studio)
- Object oriented repository for storing application declarations (Mobile Application Repository)
- Generators for generating application-specific runtime objects
- Generic runtime framework as execution environment for application-specific objects that have been generated

Summary: Mobile Client Technology provides:
Design time environment (SAP Mobile Application Studio) for declarative modeling:
  - visual modeling tools (designers)
  - programming environment (editors)
  - translation tools

OO Repository (Mobile Application Repository) for storing the application declarations (i.e. development objects):
  - version management
  - distributed development
  - Transports

Generators for generating application specific runtime objects (i.e. application code, HTML files, etc.) from these declarations

Deployment tools for installing / upgrading these applications on laptops

Generic runtime framework as execution environment for the generated application specific runtime objects
  - User Interface Layer
  - Business Object Layer
  - BDoc Layer
To enable **concurrent users** to access an object and continue developing without interfering with each other's work

- To ensure that **no information is lost** when collisions occur

- To enable users to **view multiple versions** of a development object – for example, if they want to compare the status of an object at different times

- To obtain a **complete change history**

---

**Version Management**

To allow efficient change isolation during development, many versions of the same object need to be created, each in an isolation context. Further, structures are needed to detect collisions of changes and mechanisms to resolve them must be available.

Version Management or Configuration Management is a methodology by which one can maintain the revision history of a mutable object. You can track the changes an object undergoes during the process of development. In a distributed development scenario this becomes an important feature. Any version management system generally, should have the concept of **branching**. A branch or a variant is an alternate development path. This is very much essential in parallel development, where more than one person can work with the same Object.

For each development object multiple version branches can exist which keep track of concurrent changes. This special versioning technology can be useful in case of emergency repairs, parallel concurrent development or modification of SAP delivered objects by customers and partners. By using a new branch for customer modifications to SAP objects the system offers a powerful yet easy to handle modification service. Every branch is assigned to exactly one repository. For example branch "main" is assigned to SAP for objects that SAP delivered. One can only create versions of objects within a branch which belongs to the corresponding repository (which means that a branch is changeable in exactly one repository). Per development object we have exactly one owning repository (changes in others are only possible using side branches). This supports parallel and distributed development in a system landscape (in the same or different repositories).

There are two states of versions i.e **open version** and **closed version**. An open version is one where you can edit the object. It is a private version. A closed version is one that cannot be edited. It is a public version. The baseline or the repository version forms the basis for the future changes. This baseline is a **rolling** one. Whenever a change list is released, the latest version of the object becomes the new baseline.

In the sample version tree shown above, version 5 was still open (that is, not released) when User 2 began working on the same object simultaneously. As a result, the system offered to create a side branch. User 1 then released version 5, and resumed working on the object in version 6. As soon as User 2 tries to release his change list, a collision will occur.
The current version of an object in a repository is specified by the repository baseline:

- This version is called the baseline version.
- ONLY closed versions are in the baseline.

**Version Management – Repository Baseline**

The Baseline or the Repository Version forms the basis for the future changes. This baseline is a rolling one. Whenever a change list is released, the latest version of the object becomes the new baseline. This rolling baseline concept reduces the merge effort due to parallel development.

The baseline provides every user the latest closed versions of all repository objects.

When you close a development object version (by releasing the change list), you change the baseline as well. The baseline makes the changes visible for all.

Newly created objects (those with no closed version) do not appear in the baseline.
Version Management – Concurrent Development

Branching - A branch or a variant is an alternate development path. For each development object multiple version branches can exist which keep track of concurrent changes. This special versioning technology can be useful in case of emergency repairs, parallel concurrent development or modification of SAP delivered objects by customers and partners. Every branch is assigned to exactly one repository. For example branch „MAIN“ is assigned to SAP for objects that SAP delivered. One can only create versions of objects within a branch which belongs to the corresponding repository (which means that a branch is changeable in exactly one repository). Per development object we have exactly one owning repository (changes in others are only possible using side branches). By using a new branch for customer modifications to SAP objects the system offers a powerful yet easy to handle modification service. Branching is more or less ‘forced’ by policy and/or other conditions, rather than by choice.

Collisions - When two versions have been created out of a common predecessor and development has progressed in parallel, then we say that there is a collision between these two versions.

◆ Predecessor/Successor Collision - Create an Object in a repository and transport it to the other repository. To modify this object in the target repository, it has to be done on a side branch and the baseline shifts to the side branch. This is modified on the main branch itself in the source repository and the baseline shifts to the new version here. When this is transported to the target repository then there is predecessor/successor collision.

◆ Main Branch Changeable Collision - This collision is reported when a side branch version is released first for an object in its repository. You cannot have a baseline on the side branch for an object in its repository. It has to be on the main branch only.

◆ Translation Collision - This is in case of Text Objects. Suppose you transport a text object from repository R1 to R2. In the R1 you create a translation object for this text object for a particular language. In R2 also a translation object for the imported text object for the same language is created. Now when there is a transport of this object from one repository to the other, there will be a translation collision.

Resolving Collisions – Merging
Whenever two versions of an object are created as a result of parallel lines of development from a single common predecessor, it is necessary to combine these two versions into a single version so that changes along the parallel development paths are not lost. Merging does this.

◆ Merge Copy: Here when the source version is copied into target version, all modifications, additions or deletions done in the source will be copied into target version and it becomes the new baseline version.

◆ Merge Primary: Here only the new objects created or modified will be reflected in the target version. All the objects deleted will be retained.

◆ Merge Secondary: Here only the new objects created will be reflected in the target version. All the objects deleted or modified will not be reflected in the target version.

◆ Discard Merge: Of the two versions colliding, one of them will discard the changes it has undergone in favor of the other.
Managing Change Lists and Versions

In the context of parallel and distributed development, it is important to track changes in the system.

A change list consists of a set of development object versions and the operations performed on them.

- New
- Modify
- Delete

Change lists enable:
- Change tracking and control
- Development isolation
- Change distribution

Each change list has an owner.

Managing Change Lists is the process of managing changes that take place to the data objects so that they undergo modification in an orderly fashion without destroying the data's integrity. In the context of parallel as well as distributed development, it is very important to track the changes in the system. It is the key to communication of information to different phases of development and execution. Managing changes involves recording the details of each and every operation on the development object versions in the Application Repository through change lists.

A change list consists of a set of development object versions and the operations performed on them. All operations on the development object versions are done via the mechanism of change lists and hence can be tracked. Each user makes his modification in the private domain of a change list. Typically whenever a development object is added to a change list, a new version of the object gets created and all the operations are done on this version of the object.

A change list will be the key to Transport and Replication of changes across different development repositories. To synchronize two repositories, it is enough to apply their change lists to each other. Therefore it can be said that the basic unit of transportation is a change list; which encapsulates the whole set of data modifications that were performed on the objects.

Since MAR supports a vast set of operations on the objects, it is also necessary that the change manager supports these changes. Therefore it is necessary that the change manager defines the type of operations that can be performed in a change list.

To operate on an object, it has to be added to a change list. The possible operations are

- **New** - A new Object is added to the change list.
- **Modify** - An already existing object is added to the change list for the purpose of modification.
- **Delete** - An already existing object is added to the change list for the purpose of deletion.

Each change list has an owner.
Managing Change Lists to Isolate Development

Isolated development:
- Users should not interfere with each other's work.
- The same user should be able to work on two different solutions independently.
- A user's change should not affect the baseline, in which a tested version of the object is available.
- Users should see only the changes they have made or are making.
- Users can test their changes locally.

Change lists

Development Isolation

When a user has added a particular development object version to his/her change list, that development object version cannot be added to other change lists (including the other change lists that the user himself/herself has created). When you try to edit the same object in another change list, then the system creates a new version of this development object.

The Active Context is the currently active change list and can be the baseline itself. Changes are always done in the context of a change list and the change list that is currently active is called as the Active Access Context. If no change list is currently active (typically after a release operation or logged onto the baseline) then the baseline is said to be the Active Access Context. No change can be done when the baseline is the Active Access Context, since the baseline is a public view and no object can be modified, created or deleted in the public view. Changes done to the development object version or any new object in a change list will be visible only in the access context of the change list. They won't be visible in the baseline or the other change lists.

When a user logs on (should specify the change list), the access context is calculated.

There is a concept of ownership of development objects. But this is at the repository level and not at the user level. A development object can be modified only in the repository that owns it. In case a development object needs to be changed in a repository that does not own it, it cannot be done directly in the baseline. A repair branch needs to be created and the modification needs to be done there.
Managing Change Lists – Tracking & Control

In MAR, managing change lists leverages on the Baseline Model of Version Management. The baseline is a public view version set. The versions which are in the private view (open versions) are placed in a change list. Whenever the change list is released, the baseline of the repository is updated with the new versions (the release implicitly closes the old versions) thus making the private versions public.

The Active Context is the currently active change list and can be the baseline itself. Changes are always done in the context of a change list and the change list that is currently active is called as the Active Access Context. If no change list is currently active (typically after a release operation or logged onto the baseline) then the baseline is said to be the Active Access Context. No change can be done when the baseline is the Active Access Context, since the baseline is a public view and no object can be modified, created or deleted in the public view.

Change list

A change list defines a set of development object versions and operations that were performed on the object versions. It helps define the set of objects which are be checked-in in one shot; which in turn influences the contents of the baseline version set. Each user makes his/her modification in the private domain of a change list. Typically whenever a development object is added to a change list, a new version of the object gets created and all the operations are done on this version of the object.
Managing Change Lists – Tracking & Control

If you have to change a development object, you must either use an existing change list and focus on it or create a new change list.

When a development object is open in a change list (either yours or another person’s) for editing, and you try to edit the same object in another change list, then the system creates a new version of this development object. Use this development object for your editing. (This new version is added to your work area.)

Once all the required operations are done, the change list has to be released. You must release a change list in order to make changes visible to others.
Managing Change Lists - New Repository Baseline

- Once all the required operations are done, the change list is released. All object versions have been closed by releasing the change list. The changes are visible to others. The baseline now gets altered to reflect this change. This is called the rolling baseline.

User can now work with the new versions of objects 2 and 5.
The Mobile Client Application (MCA) is the application running on the Mobile Sales or Service employee’s mobile device. The mySAP CRM Mobile Sales and Mobile Service components support a company’s mobile sales and mobile service employees, respectively, providing full access to all required data on laptop computers. This data is kept up to date through regular data exchange using the Middleware component of the central CRM System. (NOTE: There is no Middleware Server as such; the Middleware is a component of the CRM Server.)

Mobile Sales and Service users carry a full-blown PC application, including an application database, on their laptops. They connect to the CRM System as necessary via phone or network to exchange data accumulated and stored in queues at both ends. In the Communication Station, DCOM calls from Mobile Clients are transformed into remote function calls (RFCs) that go to the Middleware. This data distribution follows a flexible set of replication rules, where business objects are assigned to Mobile Clients according to values in criteria fields.

The Middleware has an RFC connection to an OLTP R/3 System, and possibly to other systems such as SAP Business Information Warehouse or a third-party application. Up to Release 3.0, these third-party systems were connected via ASCII adapter; as of 3.0, they can also be connected via XML adapter.

The CRM Middleware has a Consolidated Database (CDB) which houses the Client-related business data and application-specific metadata for the enterprise (or portion of the enterprise).

The SAP Mobile Application Studio (MAS) is a set of tools used for developing and/or customizing the mySAP CRM Mobile Applications. During replication between the MAS and the Middleware, application-specific upgrade data is sent. This upgrade data is then distributed from the Middleware to the Mobile Clients.
- Multiple MAS Clients can connect to the Mobile Repository Server (MRS) via Open DataBase Connectivity (ODBC).
- The MRS is connected to the CRM Server via the Transport Agent.
The Client Upgrade Source is normally installed on the Mobile Repository Server, and should only be used centrally by an authorized person.
Content

- Introduction
- Challenges
- SAP Mobile Application Studio within mySAP CRM
- Technical Capabilities of SAP Mobile Application Studio
  - Building enhanced Mobile Client Applications
  - Version management
  - Integration into CRM system landscape
  - Software Logistics
  - Multilanguage and translation support
- Conclusion
Transport vs. Client Upgrade

- **Transport** is the Propagation of development objects (design time objects) in a distributed development landscape. This means that development objects like source code or tile definitions are distributed between multiple application repositories connected to different SAP servers. The Mobile CRM transport functionality is built on top of the functionality of the R/3 transport system. This has two advantages:
  
  - Customizing on the CRM Server and ABAP-Workbench changes can be transported together with changes done in the Mobile Application Studio in one change request.
  - The predefined transport paths between CRM Servers can be reused.
  - Periodically running transports can synchronize different repositories automatically.

- **Client Upgrade** is the deployment and installation of executable software and other runtime objects like HTML files to the machines where the mobile clients are installed. The Client Upgrade functionality can be used to send the upgrade files to the mobile client machines and it ensures that the upgrade is executed at the correct point in time. Client Upgrade uses the BDOC distribution mechanism of the CRM Middleware. An upgrade is received by all mobile clients that are subscribed for a specific publication. The upgrade software on the client machine automatically executes the upgrade after it has received the new software. To avoid sending huge upgrades over a low-bandwidth network like a telephone line, the upgrade files can also be shipped on a CD. In this case, only the trigger to execute the upgrade is transmitted via BDoc.
Content

Introduction

Challenges

SAP Mobile Application Studio within mySAP CRM

Technical Capabilities of SAP Mobile Application Studio

- Building enhanced Mobile Client Applications
- Version management
- Integration into CRM system landscape
- Software Logistics
- Multilanguage and translation support

Conclusion
Multilanguage and Translation Support

The translation tool provides a convenient way to translate text into different languages.
Content

» Introduction

» Challenges

» SAP Mobile Application Studio within mySAP CRM

» Technical Capabilities of SAP Mobile Application Studio
  » Building enhanced Mobile Client Applications
  » Version management
  » Integration into CRM system landscape
  » Software Logistics
  » Multilanguage and translation support

» Conclusion
mySAP CRM - Customers By IBS

- Service Industries: 26%
- Consumer Products: 16%
- Process Industries: 18%
- Discrete Industries: 26%
- Retail: 6%
- Financial Services: 5%
- Not Assigned: 1%

Source: CMI, January 8, 2002
As it already has for so many other leading global brands today.

Choose mySAP CRM to enable customer-centric E-Business within your enterprise.
mySAP Business Suite
The Solution for the Extended Enterprise

- mySAP Business Suite is a suite of business applications and application & integration platform to enable companies to manage the entire value chain across business networks transforming into a more adaptive business.

- mySAP Business Suite consists of the following SAP solutions
  - mySAP CRM (Customer Relationship Management)
  - mySAP SCM (Supply Chain Management)
  - mySAP PLM (Product LifeCycle Management)
  - mySAP SRM (Supplier Relationship Management)
  - mySAP HR (Human Resources)
  - mySAP FIN(Financials)
  - mySAP Mobile Business
  - SAP NetWeaver

- **Best of Suite by Industry**
  Broader Best-In-Class business application (#1 or #2 Position of SAP in the respective market category) and platform footprint and industry specific capabilities and content for more than 20 industries.

- **Powered by SAP NetWeaver**
  Open application and integration platform with world class operations concept to run entire businesses and business networks in a secure and reliable fashion.
Prepackaged Integration
mySAP Business Suite provides business applications which are integrated by design and have open interfaces. The prepackaged integration between SAP business applications including SAP R/3 enable real-time end to end business scenarios.

Enabling Adaptive Business
SAP NetWeaver enables an open approach to adapt your system infrastructure to changing needs. mySAP solutions providing real-time data integration with the SAP R/3 transactions and leverage business analytics and planning applications to enable user to adapt the business to changing customer and market requirements.

Unique Attributes
Complete, Best-in-class Integrated business applications
The suite combines best-in-class proven business applications (SAP CRM, SAP SCM, SAP R/3 Enterprise,...) into a business and technology foundation powered by SAPNetWeaver. All applications are built on the SAP Web Application Server; common development, administration and security environments are used across all applications. The portal is the information delivery framework for all applications, be they SAP or non SAP. Mobile business applications in each solution are built on the mobile infrastructure to drive business mobility. The integration broker and Business Process Management provides process centric integration for SAP and non-SAP systems within and beyond enterprise boundaries based on open standards such as XML, Java and web services standards.

Extension beyond “ERP” capabilities in synchronization with ERP
The prepackaged integration to SAP R/3 enables companies to add CRM, SCM, PLM and SRM capabilities to their SAP R/3 installations and ensure low TCO and fast T2B.

Broadest business content
SAP is the only infrastructure provider that offers a broad and diverse connectivity and business content developed by SAP and its partners network, based on SAP’s intimate, longstanding knowledge of intricate business processes in more than 20 industries. This content shortens the development cycle for customers and drives down the cost of integration projects.