Developing Composite Applications with SAP Composite Application Framework
The Composite Application Framework (CAF) provides a programming model suitable for the development and deployment of composite applications according to the Enterprise Services Architecture (ESA).

CAF provides a comprehensive environment for modeling (metadata and tools).

CAF allows to build applications leveraging NetWeaver without the need to feed low-level APIs.

CAF helps developers and process owners to focus on implementing the business logic.

CAF comes with predefined generic patterns for reuse in different development projects.
Example: xFlights Demo

An employee has to fly to a conference. xFlights allows you to book flights.
Designing Across Traditional Functional ERP Areas

User Interface
Employee
Manager
Travel Agent

Business Scenario
Employee flies to conference.

Integration
CAF DB
Web Service
KM
HR, CRM, FI/CO
How To Build Apps Until Now?

- define Business Process
- draw Entity Relationship Diagrams
- create DDIC elements
- find & understand BAPIs, RFCs, APIs
- select Programming Language
- write coding (80% life-cycle-methods, 20% business logic)
  - ...debug, debug, debug...
How To Build Apps Until Now?

...and become familiar with...

- Enterprise Portal & Portal Development Kit
- NetWeaver Developer Studio
- ABAP Workbench
- Knowledge Management
- Backend-systems (R/3, CRM,...)
- 3rd Party-systems
- Exchange Infrastructure (XI)
- ...

Bottom line: we need developers with many skills – not easy to find
How To Build Apps Until Now?

...and the result is sometimes like this:
How To Build Apps From Now On?

Create…

```xml
<?xml version='1.0' encoding='ISO-8859-1' ?>
<XMI xmi.version='SAP_1.0' timestamp='Thu Mar 18 15:11:17 PST 2004'>
<XMI.header>
  <XMI.documentation>
    <XMI.exporter>SAP Meta Model Repository</XMI.exporter>
    <XMI.exporterVersion>1.0</XMI.exporterVersion>
  </XMI.documentation>
</XMI.header>
<XMI.content>
  <com.sap.caf.metamodel.Attribute xmi.id='010FE8D69A8247EC482BEEAA0BA8607D'
    isAbstract='false' attributeName='Manufacturer' languageDependent='0'
    timeDependent='0' hollow='0' DbFieldName='MANUFACTURER' typeJava='com.sap.caf.core.id'
    mandatory='0' keyType='0' cardinality='0' ___createdAt='20040316023208'
    ___createdBy='I804143' ___lastChangedAt='20040318231113' ___lastChangedBy='I804143'
    ___deprecated='0'/>
</XMI.content>
</XMI>

Generate…

```java
//create with public
throws
user =
if
How To Build Apps From Now On?

- **Model**
  - Entity Services
  - Application Services

- **Generate (from metadata)**
  - code
  - tables
  - Code could be platform independent (Java, ABAP, ...)

- **Integrate & Enrich**
  - backend-systems

- **Use predefined**
  - Patterns (UI, Guided Procedures – Benefit: Consistency! Once you know one, you know all)
  - Services
  - Data-types
Benefits

- Focus on
  - Modeling
  - Business Logic

- Enable
  - Programmers
  - Non-programmers (business process owners)
  - Faster redesign

- Standardize
  - Interfaces
  - Coding

- Reduce
  - Complexity
  - Development time

- Increase
  - Productivity
Architecture of CAF and Composite Applications

**Design Time**

**Process Modeler**
- Guided Procedures
- Activities

**UI Modeler**
- Web Dynpro
- Pattern Config.

**Service Modeler**
- Services
- Entities

**Metadata**
- Processes
- Workflows
- User Interfaces
- Services Entities

**Composite Application Framework**

**Run Time**

**Process Framework**
- Guided Procedures
- Work Centers
- Workflow Runtime

**UI Framework**
- UI Components
- UI Patterns
- WebDynpro Runtime

**Service Framework**
- Entity Services
- Application Services
- Persistence Manager
- External Services

**Integration Broker**
- Web Services

**DB**

© SAP AG 2004, Title of Presentation, Speaker Name / 13
Enable developers and business experts
- Model-driven, pattern-based approach to all layers relevant for Composite Application development

Combine infrastructure components
- WebAS
- WebDynpro for UI patterns and freestyle
- KM Documents and Collaboration
- BW for analytics
- BPM for workflow support

Reuse existing assets
- reuse, integrate and orchestrate existing functionality as services
- selectively fill gaps of missing functionality
Entity Service Modeler

- Is a tool for defining **Entity Services**, **Attributes** and their properties and relations to each other.

- These objects and their definitions serve to generate tables, table-relations, DDIC-elements and basic methods for reading, writing, changing, deleting & searching data (**life-cycle methods**).

- Additionally, infrastructure like **collaboration**, **authorization**, **logging & traces** and storage in multiple locations is integrated and **interfaces** are created.
```java

dataAccessService.activate();
}
} catch (Exception e) {
    CAFPublicLogger.traceThrowableT(Severity.DEBUG, location, method, e);
    throw new CreateException();
}
} finally {
    CAFPublicLogger.exiting(null, Person.JIRM_REQUEST, method, locat

    public Person create(String firstName, String lastName) throws CAFCreateException {
        java.lang.String method = Person.JIRM_REQUEST + ":create(\n        java.lang.String user = sessionContext.getCallerPrincipal().getName();
        CAFPublicLogger.entering(user, Person.JIRM_REQUEST, method, locati

        if (!checkPermission(CAPFPermissionName.create, null, user, Person.(
            Object permissionExceptionArgs = (user, CAPFPermissionName.cre
            CAFPublicLogger.exiting(user, Person.JIRM_REQUEST, method, loca

            throw new CAFCreateException("BO_PERMISSION", permissionExcep

            Person be = new Person();

            // Set the administrative attributes
            Date time = new Date();
            be.setCreatedBy(user);
            be.setCreatedAt(time);
            be.setLastChangedBy(user);
            be.setLastChangedAt(time);

            PersonPK primaryKey;
            try {
```
Application Service Modeler

- The Application Service Modeler is a tool for creating services.
- These services contain the business logic and connect to the UI- and entity service-world.
- Additionally, infrastructure like collaboration, authorization, and logging & traces is integrated and interfaces are created.
- Services can be reused and published as web-services.
User Interface Patterns

- User Interfaces can be built “freestyle” or “patterns” can be reused. Patterns are predefined and often used UI templates. Examples are ObjectSelector- (“Search”), ObjectEditor-, Dashboard- (“formatted result list”) and many others.

- Reusing patterns accelerates development and simplifies usability for users.
## User Interface Patterns – Runtime

### Persons found

<table>
<thead>
<tr>
<th>Person</th>
<th>Last name</th>
<th>First name</th>
<th>Created By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mozart</td>
<td>Wolfgang Amadeus</td>
<td></td>
<td>Administrator</td>
</tr>
<tr>
<td>Lanner</td>
<td>Joseph</td>
<td></td>
<td>Administrator</td>
</tr>
<tr>
<td>Schubert</td>
<td>Franz</td>
<td></td>
<td>Administrator</td>
</tr>
<tr>
<td>Bruckner</td>
<td>Anton</td>
<td></td>
<td>Administrator</td>
</tr>
<tr>
<td>Strauß</td>
<td>Johann</td>
<td></td>
<td>Administrator</td>
</tr>
</tbody>
</table>

1 of 5
Guided Procedures – Definition

- A framework for **modeling and managing** processes that involve access to multiple backend systems
- Designed to implement business processes with **greater ease and speed** across multiple applications
- Enables users without specialized software development skills to easily set up and execute **collaborative business processes**
- Allows the invocation of various types of applications and services within a process (Web Dynpro, RFC’s…)
- **Navigates the End User through UI-driven, people centric, distributed and collaborative business processes**
- Enables offline process execution via **Interactive Forms**
Guided Procedures – Features

- **GP design time** – set of functions enabling business experts to model processes and create reusable components

- **GP callable objects** – most fine-grained elements created in the design time; enable the execution of external applications and services within the GP framework

- **GP process flow** – CO’s are attached to actions (representing process steps at runtime) which are executed in blocks (representing phases at runtime) either sequentially, in parallel, or in a loop

- **GP data context** – the design time supports the definition of data persistency between process steps and enables mapping between the parameters of the process building elements
Guided Procedures – Features (cont’d)

- **GP roles** – allows the assignment of roles to certain actions; process role consolidation is possible

- **GP runtime** – process initiation from a process template and follow ist execution; process contributors are guided through the process steps to complete the tasks assigned to them

- **GP work items** – a contributor acts only when a work item appears in his work list (role based)

- **GP views** – set of views that show different aspects of a process (process overview/information/activities)

- **GP interactive forms** – enables the implementation of form based processes; both online and offline use of forms is supported
Guided Procedures – Features (cont’d)

- **GP administration** – system administrators can monitor and administer process instances and GP system data

- **GP workflow engine** – running on BPM runtime

- **GP translation** – make GP content available in multiple languages

- **GP transport** – enables the transfer of GP objects across SAP installations
Guided Procedures – Perspectives*

- **Control-flow perspective** – describes activities and their execution ordering (sequence, choice, parallelism, join synchronization); activities are atomic units of work

- **Data perspective** – layers business and processing data on the control perspective (data flow between activities)

- **Resource perspective** – an organizational structure e.g. human and device roles responsible for executing activities

- **Operational perspective** – describes the elementary actions executed by activities, where the actions map into underlying applications

*Van der Aalst et.al., Workflow Patterns*
Process Design and Implementation

Setup collaborative business processes using the Guided Procedures design time

■ Step 1: Design business process

<table>
<thead>
<tr>
<th>Business Process</th>
<th>Guided Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>Process</td>
</tr>
<tr>
<td>Phase</td>
<td>Block</td>
</tr>
<tr>
<td>Step</td>
<td>Action (CallableObjects)</td>
</tr>
</tbody>
</table>

■ Step 2: Re-use existing process templates
  ■ Leverage existing building blocks from the Guided Procedures Action Gallery

■ Step 3: Build missing actions
  ■ Implement the CallableObject interface for application components, e.g. CAF UI pattern or WebDynpro
  ■ If needed, create appropriate services in the NetWeaver Developer Studio

■ Step 4: Define the Guided Procedure for the business process
Process Context = Shared Data Storage

Process

Block

Rejection

Action 1 → Action 2 → Action 3 → Action 4 → Action 5

Course Request
Request Completion
Course Approval
Course Persistence
Course Confirmation

User ID
Firstname, Lastname…

Process Context
Why „Action“ Abstraction?

- Decoupling interface from implementation
- Separate the language of business experts (Action) from the language of developers (Callable Objects)
- Action embraces Callable Objects for execution and display
- Reuse of one Callable Object in several scenarios with easy adoption to new situations (e.g. Select User for next step)
Composite application development on SAP NetWeaver 2004s

- Ongoing: Composites and xApps by SAP
- Q3/2005: Light and medium weight composites by selected partners
- Q2/2006: Composite applications by partners and customers

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning, development, and shipment of Composite Apps at SAP</td>
<td>Light-medium weight Composite Apps built by selected partners</td>
<td>General roll-out</td>
<td>Ramp-up start mySAP ERP 2005 incl. SAP NetWeaver 2004s</td>
</tr>
<tr>
<td>TechEd</td>
<td>Partner enablement: training material and workshops</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Summary

- **Composite applications** empower companies to drive innovative business processes by leveraging existing IT investments.

- The **SAP Composite Application Framework (SAP CAF)** provides a methodology and toolset to efficiently develop and manage composite applications—following the Enterprise Services Architecture blueprint.

- **CAF Core** enables the modeling of Entity and Application Services. Entity Services encapsulate business objects including their local or remote persistency whereas Application Services compose and orchestrate services for implementing new business logic.

- **CAF Guided Procedures** allows the modeling, managing, and execution of new collaborative business processes.