Office of the CTO

Werner Aigner
Reiner Hammerich
Holger Meinert
Jochen Puzicha

Office of the CTO
Architecture Knowledge Transfer

Bernhard Gröne

SAP Architecture Whitepaper

Near-Term Architecture Strategy

Enterprise SOA Consumption

May 2007

Acknowledgements

The authors want to thank the colleagues who made such a comprehensive document possible, especially the contributors Stefan Baeuerle, Markus Cherdron, Wolfgang Degenhardt, Roman Hayer, Wolfgang Hilpert, Nicolai Jordt, Stefan Kaetker, Prasad Kompalli, Filip Misovski, Gordon Muehl, Guenter Pecht-Seibert, Uta Prigge, Christian Schloegel, Sebastian Speck, Franz Weber, Andreas Wesselmann, and all reviewers.

Table of Contents

Table of Figures

1 Introduction

1.1 Enterprise SOA

Customers have to innovate and change business processes frequently, driven by the business need to differentiate from competitors, to reduce cost by consolidation and standardization, and to adhere to regulatory compliance. The flexibility to adopt existing business processes and to create new ones is the key benefit that customers expect from enterprise SOA. Reduction of IT costs for implementation, integration, and change is clearly another strategic goal.

> *"Neither customers nor prospects care very much about SOA, but they are passionate about ease of use, functionality, information access, and cost of ownership." [Gart05]*

Therefore, providing enterprise services and the infrastructure to build and use them can only be one part of SAP's delivery on the enterprise SOA promise. The second, not less important part is to enable flexible service consumption and thereby to address the customers' individual needs for innovation and optimization, which proves the real value of enterprise SOA.

1.2 Architecture Strategy

This document depicts SAP's architecture strategy addressing service consumption and flexibility in enterprise SOA. It focuses on service consumers of special interest: tailored user interfaces, business processes, and composite applications. To provide a complete picture, the relevant aspects of service provisioning and business process platforms are included as well.

The architecture strategy is discussed in the context of SAP ERP, AP/A1S and technology provided by SAP NetWeaver. SAP CRM, SCM, SRM, coexistence scenarios of A1S and SAP Business Suite, and SAP Business One are not considered explicitly, although some of the aspects adhere to them as well.

The concrete roadmap including the realization plan and architectural details are not discussed here. Following the architecture strategy, this task belongs to the responsibility of the respective development organizations.

It is not likely that SAP will be able to realize the complete architecture laid out in this document in one release cycle, therefore we talk about the "near-term architecture strategy". It is an explicit assumption, though, that substantial parts will be available – in a phased approach – in the next releases and not only at the end of this implementation period.

2 Architecture Strategy – The Big Picture

This chapter introduces the cornerstones of enterprise service-oriented architecture (enterprise SOA) with a focus on three main consumption scenarios: user interfaces, business process management, and composite applications. In an ideal world, these scenarios would be independent from the service provisioning or platform side. More realistically, however, consumption and provisioning are flip sides of one coin and need to be considered, to some extent, together. Therefore the big picture laid out in this chapter also addresses important aspects of the provisioning side. The picture is refined in subsequent chapters on business process management, user interfaces and composite applications.

2.1 Enterprise SOA – Platform and Composition

Companies need a reliable backbone guaranteeing integrity for their business which they can adapt and extend to their needs in an easy and flexible way. This promise of enterprise SOA is addressed by different types of interconnected applications[1]:

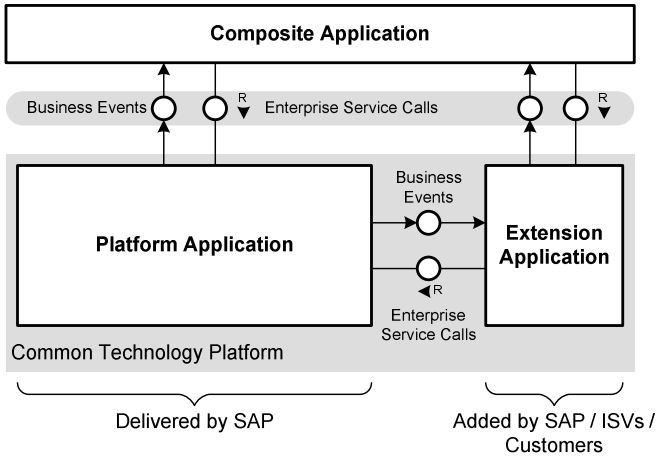

Figure 2-1 Application Types

- **Platform applications** deliver a high number of standard, highly integrated, mission-critical processes applicable to many industry or customer segments. They are designed to satisfy a high demand for integration, integrity and legal compliance – and therefore offer controlled flexibility through configuration and extensibility. SAP offers two platforms for large and mid-size enterprises: SAP ERP following the "enterprise SOA by evolution" approach, and AP/A1S[2] based on the "enterprise SOA by design" approach.
- **Extension applications** extend a platform to deliver integrated, mission-critical processes applicable to specific industry, customer or business segments. Examples could be "Transportation Management", "Talent Management" or "Bank Accounting" for the Financial Services industries. Extension applications share the key characteristics of platform applications, including their programming model, but are built to satisfy more specific business needs. An extension application is typically closely integrated with one platform: SAP ERP or AP/A1S.
- **Composite applications** are specifically built to satisfy more individual business needs ranging from simple extensions of platform and extension applications to new business processes ("next practices"). They allow companies to differentiate from competitors as well as to extend standard

[1] This classification is not yet aligned with marketing and other teams.

[2] In this document AP is considered a platform and A1S a platform application. Composition takes place either on A1S or on SAP ERP.

processes to individual needs. Consequently, there is a whole variety of different composite applications that can, in general, not easily be anticipated or pre-thought by SAP, and typically have a shorter lifecycle than platform and extensions applications.

Characteristics of different application types. There is no sharp borderline between the different types of applications. The following list of characteristics helps foster the understanding of the key differences.

Platform and extension applications:	Composite applications:
• cover the standardized, mission-critical business processes of an enterprise	• cover the long-tail or tailored processes for certain customer segments or user groups
• focus on integration, integrity and compliance	• focus on tailored user experience
• process legally binding business documents and fulfill strict data management and audit requirements	• can also deal with draft and collaborative data
• are configurable and can be extended in predefined ways	• are specifically built for dedicated use cases
• expose functionality through services, events and user interfaces	• consume existing functionality by calling services, reacting to events, and embedding user interfaces
• usually have a longer lifecycle, and more rigid upgrade and patch requirements	• usually have a shorter lifecycle, and are changed frequently

Service provisioning and consumption. Platform and extension applications expose services and events to allow building additional business processes, user interfaces or composite applications. In that sense, they do service provisioning whereas composite applications have a clear focus on service consumption.

2.2 Architectural Paradigms of Enterprise SOA

The logical structure of a system landscape according to enterprise SOA is shown in figure 2-2. It depicts the interacting types of applications as well as their key construction principles in the areas of user interfaces, business processes and composite applications which are refined in later chapters of this document.

The Main Building Blocks

The previous section introduced the two platforms, SAP ERP and AP/A1S that SAP offers in the near future for large and mid-size enterprises, extensions to these platforms and composite applications. Interactions between these applications are based on enterprise services and business events.

- An **enterprise service** is a service which provides business functionality and which is published by SAP in the Enterprise Services Workplace. Enterprise services are structured according to a harmonized enterprise model based on process components, business objects and global data types. They are well documented, guarantee quality and stability and are based on open standards.

- **Business event**[3] is an event that provides significant business information. In contrast to a service call, which is used to request or trigger an action of the receiver, an event informs an open number of subscribers about a (significant) change of state of a system.

[3] Until now, “Business Event” is a working title only.

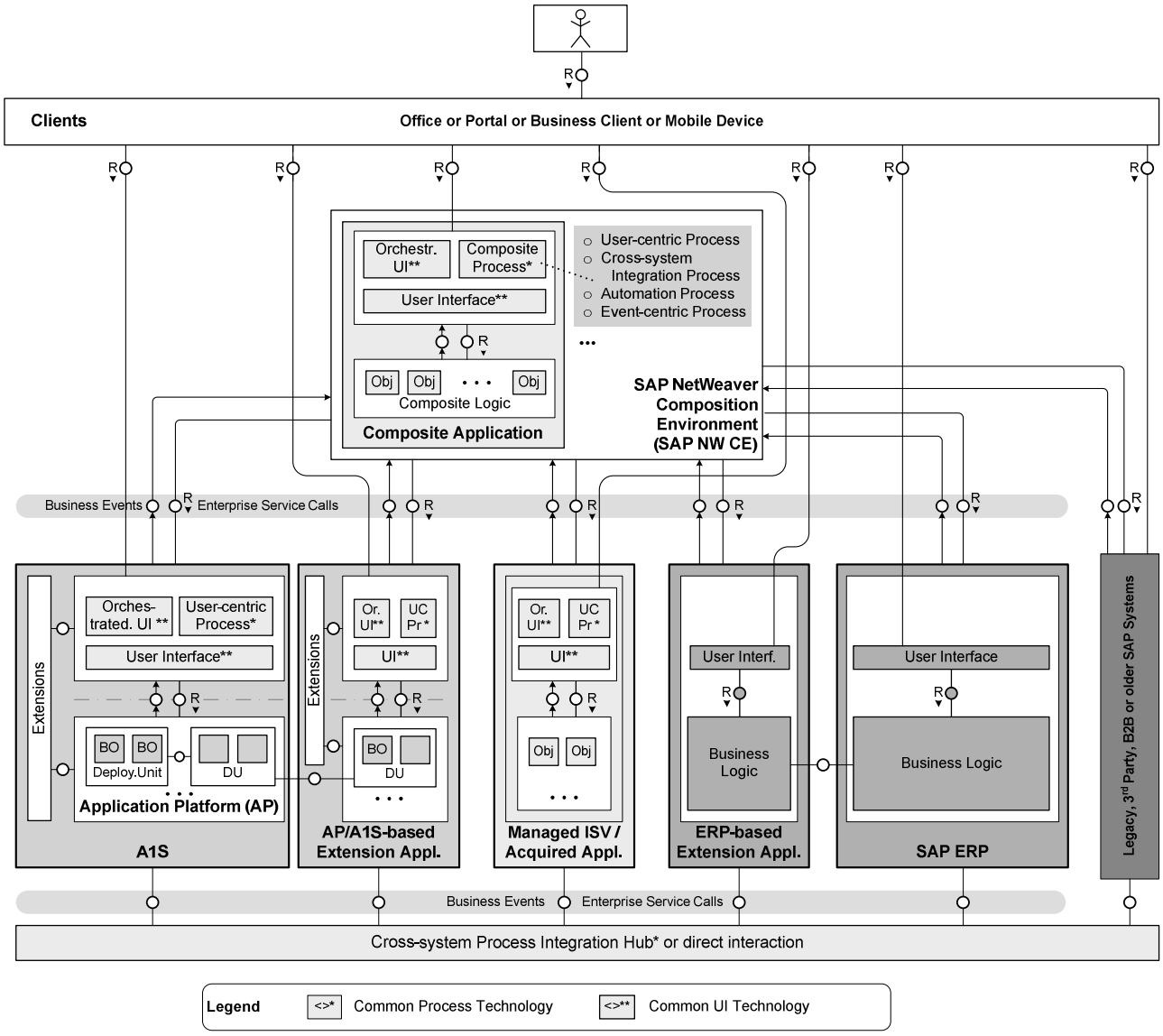

Figure 2-2 Big Picture of Enterprise SOA

Each of the two platforms comes along with its own programming model, business objects, process logic, user interface paradigm and set of enterprise services and business events. As a consequence, there will be two types of extension applications: extensions to SAP ERP following the SAP ERP programming model, and extension to AP/A1S following the AP/A1S programming model. Applications acquired by SAP, like Khimetrics or Triversity for retail or special partner applications covering white spaces in SAP's product offering, will fall into a third category of extension applications.

The enterprise architecture of a company, however, also comprises legacy systems, systems from other vendors and older releases of SAP systems. Integration of these systems is typically done on a case-by-case basis and includes service and event enablement, integration of user interfaces in composite views or integration into business processes. These needs are addressed by the integration capabilities of SAP NetWeaver, examples being the SAP NetWeaver Portal or SAP NetWeaver Exchange Infrastructure (XI). ISVs can use composite applications to integrate their own applications into the SAP landscape of a company.

SAP NetWeaver Composition Environment (SAP NW CE) is the recommended environment for building composite applications on top of all the different types of applications or systems.[4] It is an integrated environment that offers a lean, standards-based way to build composite applications in Java and supports all major consumption scenarios in enterprise SOA. Based on Java EE 5, it includes tools for service composition, UI composition and process orchestration.

Independent Lifecycles and Loose Coupling

In general, platform applications, extension applications and composite applications must be designed for independent lifecycles to address the different frequency of changes and to decouple release cycles between SAP, ISVs and customers. Hence the interaction among applications must be based on the principle of loose coupling through stateless services and events. Stateful or transactional service calls are not allowed across application borders.

Compatibility is an important prerequisite for independent lifecycles of applications. To prevent that the use of a newer version of one application enforces the use of newer versions of other applications at customer side, the following rules have to be adhered to – within well-defined boundaries (e.g. two release cycles):

- Services, including interface and implementation/behavior, are only changed in a way that does not break existing consumers;
- Service consumers must be able to deal with different versions of the required services.

Platforms and extension applications are integrated according to the following principles:

- In case of SAP ERP, extension applications communicate with existing platform applications using RFC or ALE technology. New platform and extension applications will use enterprise services and business events.
- The integration between deployment units within AP and with deployment units of corresponding extension applications is based on the AP integration model using enterprise services, business events and process agents.
- Integration of SAP ERP-based applications with AP-based applications will be considered in a future version of this document.

Platform Extensibility

Composition on top of existing applications will be, in many real cases, not possible without affecting the underlying platform or extension applications:

- In many cases it is sufficient to adopt or reconfigure existing user interfaces or processes without having to develop a new composite application.
- In the other cases, it is necessary to provide new user interfaces, new services or events, or to add a field in a business object and its external interfaces. The ability to “hook” new composite logic into existing process logic might even require more extensibility features.

SAP ERP and AP/A1S differ considerably in their extensibility capabilities. In SAP ERP these capabilities are very heterogeneous and limited. Enhancements are only possible where extension points such as Business Add-Ins (BAdIs) are explicitly offered. In any case, very specific knowledge is required. AP/A1S is using a consistent set of models across all areas. On one hand, it makes the application’s process logic transparent, and on the other hand it allows extensions on a broad scale in a consistent and generic way. Although not all features are implemented in the first AP/A1S release (see discussion on process flexibility in chapter 3, for example), the basic concepts are in place for key customer use cases.

[4] Of course, composite applications can be built with other service-based products, such as the .Net environment.

Common Technologies

AP/A1S and SAP NetWeaver Composition Environment share technologies for business process management, and for creating and composing user interfaces. As a result, composite processes in SAP NetWeaver CE and user-centric processes in AP/A1S use one business process infrastructure (see chapter 3.4 for more details), which will also be used in future versions of the process integration hub evolving from SAP NetWeaver Process Integration (previously known as Exchange Infrastructure). Similarly, modeled user interfaces and user interface composition in both AP/A1S and SAP NetWeaver CE use one UI technology (see chapter 4.5).

Tools and Repositories

Increasing transparency while decreasing the cost of development and change is essential for the success of enterprise SOA. Prerequisites for this are the integration of tools and repositories as well as alignment of metamodels.

- SAP NetWeaver CE will provide one coherent set of metamodels and one integrated toolset based on Eclipse for the development of composite applications. Special user roles such as the business process expert may have additional modeling capabilities.
- Enterprise SOA by design in AP/A1S and SAP NetWeaver CE will be backed up by one integrated toolset and next generation enterprise repository that covers both, the composition and the provisioning side. It will allow the comprehensive modeling of business processes and user interfaces including extensibility of the platform. It is assumed that this environment is based on Eclipse and the SAP NetWeaver Modeling Infrastructure (MOIN) as underlying repository technology.

Today the Enterprise Services Repository (ES Repository) contains the service definitions offered by platform and extension applications including business-to-business (B2B) services used across enterprises, application-to-application (A2A) services for integration within distributable platform applications as well as between platform and extension applications, and “application-to-any” (A2X) services for composition. In future, business event definitions and information about user interfaces for use in composite applications will be provided as well.

2.3 Benefits

The architecture strategy presented in this document provides a high-level frame for enterprise SOA for customers, partners and SAP development. Although it seems fairly simple, it depicts the few crucial construction principles and cornerstones that are needed

- by customers to understand enterprise SOA and it’s impact onto their enterprise architecture;
- by partners to align their technology strategy and offerings with SAP’s architecture strategy;
- by SAP development to be able to consistently work out the details within the individual application and technology areas.

By showing how SAP’s core values, integrity and compliance of mission-critical applications, will be combined with flexible composition techniques into a comprehensive overall architecture, this near-term strategy helps positioning enterprise SOA inside and outside SAP and provides a strong answer to competitor initiatives.

3 Business Process Management (BPM)

Business process management (BPM) is an approach to define, execute and monitor business processes of a company in a flexible way, allowing for continuous adaptation and improvement. Gartner [Gart05] defined BPM as “a management practice that provides for governance of a business's process environment toward the goal of improving agility and operational performance. BPM is a structured approach employing methods, policies, metrics, management practices and software tools to manage and continuously optimize an organization's activities and processes."

It is important to realize that BPM is more than a specific software or technology. Customers consider both, SAP’s applications and process technology, as means to manage their business processes. In the following, SAP’s approach to support customers with enterprise software applications, tools, and technology for business process management is described. In the rest of this document, the term BPM is used to refer to enterprise software – applications, tools, and technology – to manage business processes.

The main cornerstone of the BPM architecture is the central and integrated toolset that allows modeling, managing, monitoring, and optimizing different types of seamlessly interacting processes using a common set of runtime services and events.

3.1 Process Types in Enterprise SOA

A variety of not necessarily disjoint business processes are relevant in enterprise SOA, including

- user-centric processes, more traditionally also known as workflows;
- application-to-application (A2A) processes;
- integration processes across enterprises (business-to-business, B2B);
- customer-specific processes to integrate their legacy or third-party systems;
- event-centric processes to understand, respond to and act upon business events;
- automation processes to reduce manual steps;
- processes to integrate real-world artifacts, such as processes using RFID technology.

End-to-end business processes are usually a combination of these process types.

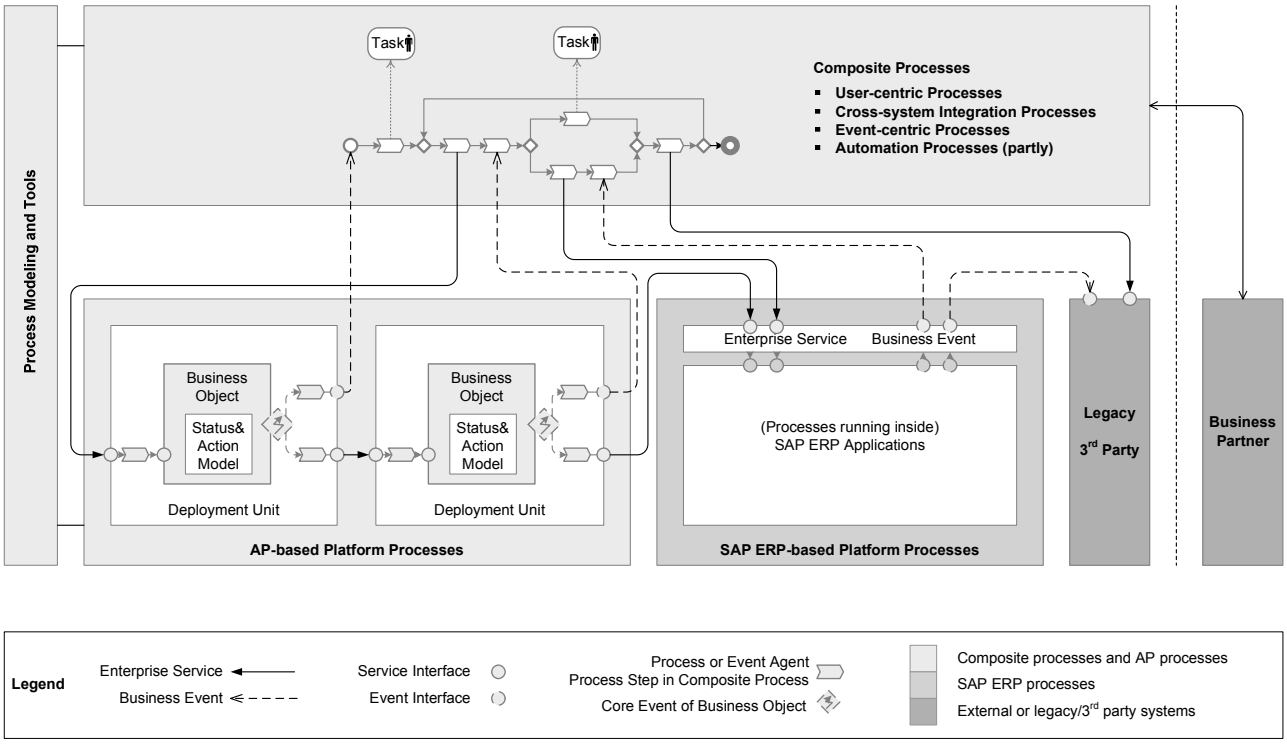

Figure 3-1 Types of Business Processes[5]

In line with the distinction between platform applications and composite applications, business processes are typically a seamless combination of two kinds of processes, as shown in figure 3-1 (see section 3.4 for more details):

- platform processes implementing proven standard business practices. They are delivered by SAP ERP and AP.
- composite processes implementing next practices or extending or adapting platform processes according to individual (customer) needs. They can be delivered as part of SAP solutions or can be built by SAP, customers or partners on top of SAP solutions.

One comprehensive toolset, one model repository, and a common runtime infrastructure are necessary to model, manage, execute, monitor and optimize business processes.

3.2 Status

Different generations of process technology are used by SAP ERP and by AP/A1S.

SAP ERP uses various process technologies, but most applications only make limited use of them. A prominent example is business workflow which is mainly used to govern human activities. The majority of processes in SAP ERP is defined by program code and is adapted by configuration.

In contrast, business process management is a main pillar of the AP/A1S architecture, including:

- process modeling in the ARIS tool which is integrated into the enterprise services repository (ES Repository);

[5] This figure is only an illustration and does not show all possible interactions or capabilities; for example, AP-based platform processes can trigger tasks, composite processes can throw events, platform processes can have direct B2B interactions. In particular, *the picture does not imply that there must always be a composite process that acts as central orchestrator!*

- business objects along with their status and action models defined in ES Repository;
- process integration between deployment units following an event-condition-action (ECA) paradigm implemented by the process agent framework (PAF);
- business task management (BTM) to address human interaction[6].

To complete the AP/A1S approach, certain capabilities need to be added coherently. Examples are multi-step, multi-user workflows or modeling and extensibility capabilities to meet the process flexibility requirements.

On technology side, PTU SAP NetWeaver started the Galaxy project in April 2006 to innovate and consolidate SAP's process technology into a comprehensive, market-leading business process infrastructure[7] which also enables completely new scenarios. Galaxy will initially focus on scenarios for the NetWeaver Composition Environment (SAP NW CE), and the first delivery is also planned as part of SAP NW CE.

3.3 Strategy

SAP will provide a coherent and comprehensive business process management that integrates its application know-how with innovative technology. SAP's BPM will support human and system processes, structured and unstructured[8] processes, standard and exceptional human tasks, and will bring together the worlds of business and IT by a comprehensive process model. There will be a coherent metamodel, a comprehensive toolset and runtime services. The use of standards, such as the business process modeling notation (BPMN) or the business process execution language (BPEL)[9] will increase interoperability, acceptance, and ease of understanding among all stakeholders .

Features of SAP's BPM

SAP's business process technology needs to serve application and customer needs. It will support:

- the complete process lifecycle: discovery, ex-ante analysis, design, simulation, implementation, deployment, monitoring and management, ex-post analysis and optimization;
- different personas: "executive-to-developer", and levels of abstraction: "concept-to-execution";
- different types of processes: platform processes (AP only) and composite processes;
- different scenarios: from human-to-human to system-to-system processes;
- different levels of structure: from structured to unstructured or ad-hoc processes;

---

[6] For a more complete overview of process management in AP/A1S see [ESA06].

[7] In the following, the terms (business) process technology or infrastructure will be used to refer to the respective technologies delivered by SAP NetWeaver avoiding references to specific projects or technologies like Galaxy or others.

[8] Unstructured here refers to ad-hoc capabilities and to collaborative ways of working together without specifying who does what when.

[9] **BPMN** is a standardized graphical notation, based on a flowcharting technique, for depicting business processes (http://www.bpmn.org); an example is shown in figure 3-4 below. The primary goal of BPMN is to provide a notation that is readily understandable by all business users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes. BPMN is maintained by the Object Management Group (OMG; http://www.omg.org/).

BPEL is an XML-based language to define the sequence of Web service invocations in B2B or A2A environments (http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel). It is completely focused on 'orchestration': call, consume, and manage calling other Web services - and nothing more. BPEL is being standardized by the OASIS consortium (http://www.oasis-open.org); currently the version called WS-BPEL 2.0 is being finalized.

BPMN specifies a formal translation to BPEL. However, this requires that BPMN is used in a specific way; not all processes represented by BPMN can be mapped to BPEL.

- different paradigms such as control flow or event-condition-action rules;
- standards such as BPMN or BPEL.

Toolset and Runtime

A single central toolset based on the next generation Enterprise Services Repository will be provided to model and manage composite processes and platform processes running in AP in an integrated way. SAP NetWeaver will provide one set of runtime services to execute these processes.

Processes running in SAP ERP are integrated via enterprise services and business events only. The central toolset will not allow examining or controlling processes or business workflows residing inside SAP ERP applications. There will be no extraction or remodeling of these processes.

Process Models

Despite the differences in the applied process technology, it is imperative for customers to get a coherent view onto their processes – not necessarily on a detailed technical level but on a more business-oriented level. A joint process flow model layer will be provided for AP-based platform processes[10], composite processes and their combination. It will use BPMN and will be tightly integrated with the respective execution-level models. For customers, this will create a holistic model experience.

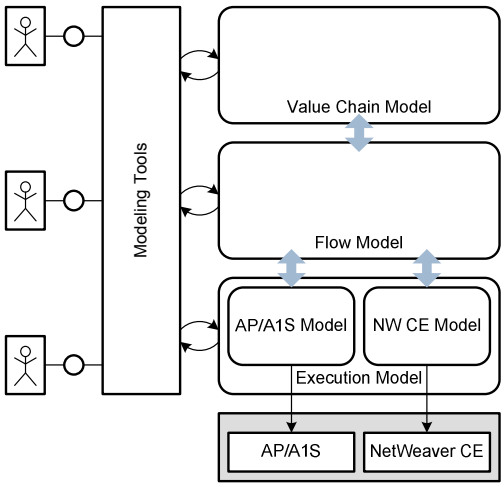

Figure 3-2 Business Process Models: From Concept to Execution

The modeling approach for business processes will support collaboration between business and IT. On the one hand, the models allow pushing business requirements to execution level; on the other hand business experts can discover what is executed in the system. As shown in figure 3-2, model refinement and abstraction will be supported by tools, but require corresponding processes and collaboration of people.

3.4 Platform and Composite Processes

In enterprise SOA, business processes are seamless combinations of platform and composite processes.

Platform processes provide proven standard business practices and are delivered by SAP ERP and by AP. Platform processes are built to satisfy a high demand on integration, integrity and legal compliance, and are

[10] As indicated above, this will not be possible for processes within SAP ERP.

typically mission-critical. Readers may think of sell-from-stock or make-to-order as examples[11]. The end-to-end complexity of platform processes is high but the number of variants for individual processing steps is limited due to the fact that they represent standard practices. Platform processes are designed as integral part of a process platform. They can be configured and extended,[12] but they can not be flexibly recomposed.[13]

Composite processes address the individual needs of customers. They complement or extend platform processes. Due to their specific nature, variation is enormous and can, in general, not easily be anticipated or pre-thought by SAP. For customers, composite processes are not only important to differentiate from competitors ("next practices"), but also to quickly support their business according to their or their business partners' individual requirements. Prominent subcategories of composite processes have already been introduced: user-centric processes, cross-system integration processes, automation processes and event-centric processes.

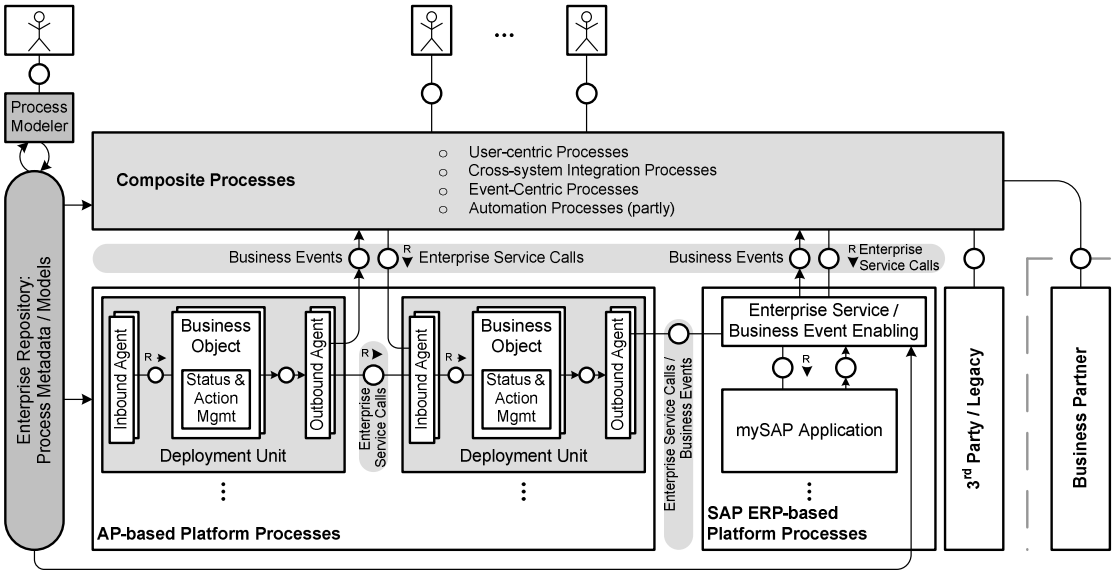

Figure 3-3 Platform Processes and Composite Processes

Business processes are executed in a federated way with all types of processes[14] seamlessly interacting through enterprise services and business events. There may not be an overall (composite) process that coordinates all interactions. The level of transparency and control of the processes will be different for SAP ERP on one hand and for platform processes in AP or composite processes on the other hand. This is due to the fact that process logic in AP is (mainly) explicitly described by models whereas in SAP ERP it is (mainly) defined by coding.

AP and composite processes will adhere to a coherent overall metamodel, a coherent central toolset and a common set of runtime services. Both will be integrated through common process flow models (see section 3.6 below), although they may use different subsets of the execution technology.

[11] Sell-from-stock or make-to-order are examples to foster the way of thinking about platform processes. They are standard practices with some variation and may even appear in combination.

[12] Extensions of platform processes can only be done in a governed way in order not to break the platform. Governance refers to where and which extensions are possible.

[13] In general, new or different platform processes with their requirements towards integration, integrity and compliance can not be created by simply recomposing existing parts but require more fundamental (re-)construction.

[14] This is also true for AP-internal processes when they cross the boundaries of deployment units as depicted in figure 3-3.

The huge end-to-end complexity of platform processes is handled in AP by a decomposition approach. The platform provides a set of well-integrated but federated components, the deployment units, which interact without central runtime orchestration. Each deployment unit offers services to change the state of its business objects. Status and action models encode which status transitions are possible. Event-condition-action rules, executed in process agents, define which status transitions result in which subsequent process steps. These process steps are executed through invocation of enterprise services of other deployment units. This approach combines central modeling and management with local process execution[15]. Status and action models as well as process agents will be centrally modeled[16] in a homogenous way, and the end-to-end view onto processes will be provided by models on higher abstraction level. Thus platform processes in AP will be modeled and managed as one integral part of the envisioned business process management. Enterprise services, business events, extension of business objects, extension of status and action models, and finally the event-condition-action mechanism allow extending and integrating AP-based platform processes with composite processes in a powerful way.

Composite processes are flexibly modeled according to individual requirements. They can trigger or can be triggered by platform processes, and they can insert additional steps in platform processes. They do not share the latter's high end-to-end complexity, but rather address smaller parts of the overall business process or orchestrate in certain cases. They will be centrally defined and executed. Initially, composite processes will be defined using a flowchart technique (such as BPMN or BPEL, which must be supported as the orchestration language in the Web services world); other techniques will be available later (for example, rules or state charts). Composite processes will be modeled and managed as one integral part of the envisioned business process management. Figure 3-3 only depicts the logical structure but not all "usages" of composite processes. They will not only be part of composite applications but will also appear as part of process or event hubs (including what is known from SAP NetWeaver Exchange Infrastructure). In contrast to today's reality, the new BPM approach will also enable a seamless integration of user-centric processes in these situations.

SAP ERP will participate in the envisioned business process management only in a lightweight fashion based on enterprise services and business events. This allows constructing composite processes on top of SAP ERP but does not allow visibility into or control over processes running inside SAP ERP. Business Workflow will not be replaced by new process technology but will continue to exist as is. Event enabling of the SAP ERP must be done similar to service enabling[17].

3.5 Flexibility of Platform Processes

The core value delivered by a business process platform is the reliable integration of a high number of standard, mission critical processes. While flexibility of platform processes is certainly required, it must not compromise integrity. Therefore, to a large extent, platform process flexibility is achieved by configuration within the boundaries of predefined and quality proven variations. Platform flexibility beyond the borderlines of predefined variation is enabled by different means – described below – in a way that platform integrity stays protected.

Most scenarios where customers want to adapt a platform fall into one of two categories:

- Integrate the platform into the customer landscape;
- Extend the solution scope of the platform (this is also being done by SAP, ISVs or SIs).

Integration in Customer Landscape. In principle, this adaptation is non-invasive and is addressed by the integration capabilities of SAP NetWeaver. A platform must be prepared for these integration needs by offering enterprise services and business events. For AP, both are intrinsic part of the "SOA by design"

[15] More details on process modeling and integration can be found in [ESA06].

[16] This is true for status and action models, and is planned for process agents.

[17] The concrete decisions on event enabling of SAP ERP and the underlying technology have not been taken yet.

approach; for SAP ERP, both will be provided by an enabling layer on top of existing applications ("SOA by evolution").

Extension of the solution scope. In an ideal world, extending the solution scope could be done by solely using SAP NetWeaver Composition Environment. However, in most real cases the necessary extensions affect the underlying platform processes, and some cases might even not be solvable without digging deeper into the platform. The problem to be solved therefore is to enable the relevant extensions of platform process without threatening platform integrity, and ideally without coding.

In SAP ERP these extensions are only possible through configuration or through coding where specific exits are foreseen; in both cases, specific knowledge is required.

For AP, the following process extensibility features will be supported[18]:

- Add fields in an end-to-end way (from database through business logic to the user interface).
- Add individual, user-centric processes.
- Integrate external Web services.
- Add processes before or after platform process.
- Extend platform processes with new (composite) logic.

Within AP process models, extension points and patterns will be defined that govern and restrict customer adaptations. Thus, customers can extend AP, to a large extent, in a purely model-based way without breaking its integrity.

3.6 Flow Models for Application Platform and Composite Processes

Customers need one integrated way to understand and extend platform processes in AP, and to add or integrate composite processes. This need will be met by a process flow model layer joining AP processes and composite processes.

Process flow models, which are described by and extended version of BPMN, enable understanding the sequence of process steps without digging into execution-level models. By linking them to execution-level models, navigation between the models will be possible.

Process flow models comprise three components:

- Composite processes (the upper left, darker part in figure 3-4; the upper right, dark part depicts an external process step) which can be flexibly modeled according to the individual solution needs.
- Platform processes in AP (the light shaded steps in the lower layer of figure 3-4). The flow models for AP processes will be generated from the underlying execution-level models on demand without manual intervention. In contrast to composite processes, they can not be arbitrarily manipulated but contain the precise metadata where which extensions are possible. The extension points are depicted as circles attached to process steps in the lower layer of figure 3-4.
- Extensions of AP processes (the darker shaded steps in the lower layer of figure 3-4) as discussed in the previous section. They are created in the flow models based on the extension points and extension patterns offered by the existing AP processes.

[18] From a customer perspective, adaptation of user interfaces, reports and forms has the highest priority among all adaptation needs. The list represents the process view onto these requirements that have been worked out by A1S Solution Management and SAP NetWeaver Product Management based on customer input.

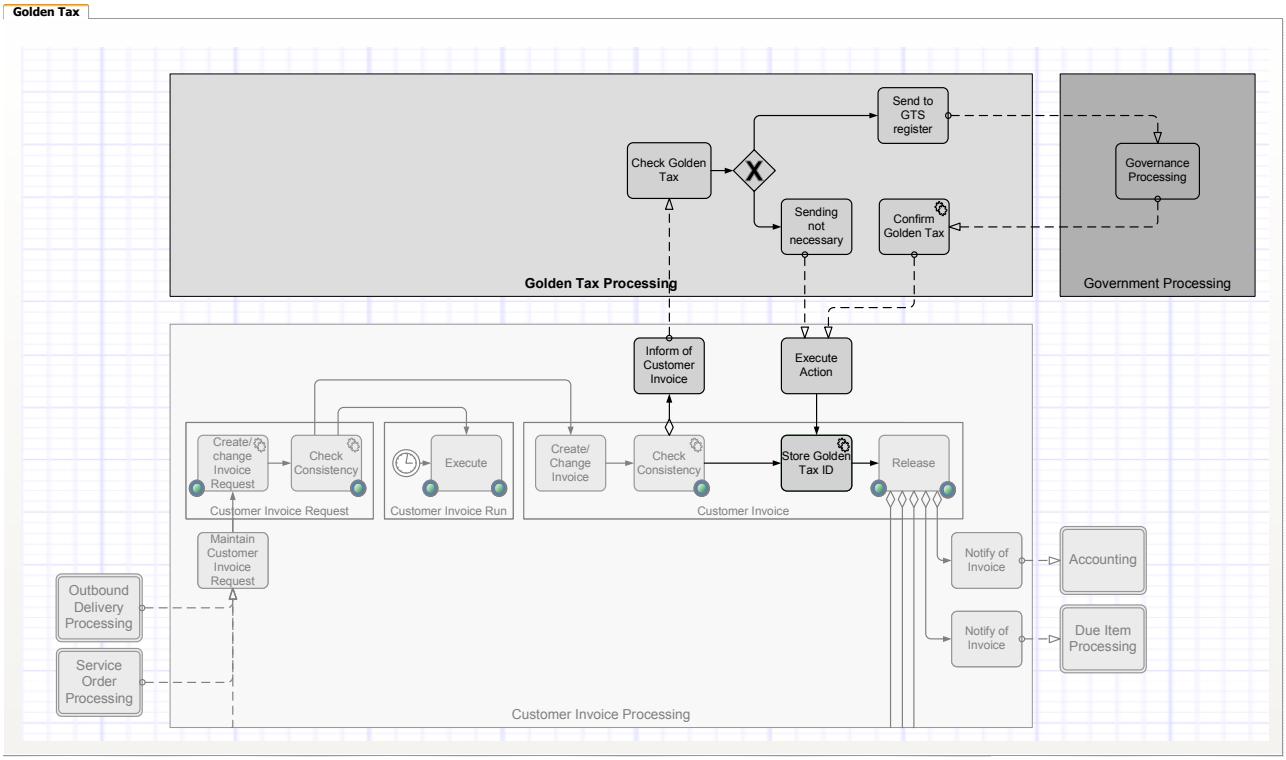

Figure 3-4 Flow Model of Composite and AP Process using BPMN

Purposes of Process Flow Models

One purpose is the extension of the platform solution scope by adding and integrating composite processes. Ideally, a composite process is described and refined top down from value chain model to flow model. Then the flow models of the relevant platform processes are included in a read-only manner. Based on these two ingredients, the relevant extensions of the platform processes are made and the connections between composite process and platform processes are created. Finally, the composite process is refined so that it is actually executable. From a customer perspective this must be a seamless and natural procedure not only to create full-fledged next practices on top of the platform but also to adopt the platform to specific needs.

Another purpose of process flow models is to give insight into platform processes in AP by generating the flow models focusing on relevant process steps and omitting technical details. This will also allow checking whether the execution models really meet the process requirements.

Finally, process flow models support a top-down modeling approach for platform processes as well: from value chain models through process flow models down to execution-level models. Top-down modeling will be the approach to define composite processes but is also envisioned in a later stage for platform processes.

4 User Interface

In a service-oriented world, user interfaces (UI) are one of the most prominent service consumers. In SOA, platforms expose functionality for consumers only via services and hide implementation and internal data behind service interfaces. Therefore the only way to implement a user interface is using these services. Since this architecture aims to guarantee consistency and compliance in the platform, one of the promises of enterprise SOA is to enable rapid user interface development and rapid changes on the user interface without endangering the underlying platform functionality. As a consequence, a platform needs to offer services, metadata and the infrastructure to build user interfaces on top of the given services. This service orientation applies to all kinds of services and user interfaces, no matter whether they are transactional, tightly or loosely coupled, stateless or stateful.

4.1 Status

The SAP user interface stack consists of several layers. First of all there is the client layer which performs the user interaction and rendering, for clients such as a browser and SAP NetWeaver Business Client. These clients are typically driven by the SAP NetWeaver Portal which serves as the UI integration platform for a wide range of applications. The applications themselves are implemented in a variety of programming models on different platforms such as SAP NetWeaver Application Server Java or ABAP. In an enterprise SOA world, the UI applications interact with the back ends through enterprise services.

SAP NetWeaver Business Client and the Portal

NetWeaver Business Client (SAP NW BC) is SAP's next-generation rich client. It combines features of a Web browser and the SAP GUI with a new and state-of-the art visual branding as well as significant performance benefits compared to SAP user interfaces running in pure Web browsers. A first version of SAP NetWeaver Business Client has already been released.

SAP NetWeaver Business Client must be seen as complementary access channel for the SAP NetWeaver Portal. For that purpose, the portal is modularized providing the core portal services separate from the portal user interface. The core portal services are leveraged by SAP NetWeaver Business client as well as by the portal user interface which is brought to the next level by introducing new Web 2.0 technologies such as AJAX.

Variety of UI Programming Models and Technologies

Today, a variety of UI programming models and technologies are used across SAP solutions. A set of categories group these programming models and technologies to provide guidance when to use which UI technology.

One of the goals of the overall UI strategy is to reduce – and in the end to replace – most of today's UI technologies [RoKr06]. It aims to finally consolidate application user interfaces on top of Web Dynpro technology, both in the Java as well as the ABAP environment.

For SAP ERP and New A1 moving from deprecated UI technologies to Web Dynpro ABAP is a huge ongoing effort since there is no automated migration. In order to redo the most commonly used SAP ERP user interfaces, Project Muse was announced. It combines the SAP NetWeaver Business Client with new Web Dynpro ABAP user interfaces. This aims to boost user productivity as well as to dramatically improve the visual appearance of SAP ERP and New A1 user interfaces.

In contrast to SAP ERP, the user interface of the A1S solution is in most cases defined by UI models based on patterns and floorplans, combining proven presentation and navigation solutions for typical user interface tasks. These patterns are implemented with Web Dynpro Java, and UI models are defined with SAP NetWeaver Visual Composer. However, there are still some challenges ahead regarding the consumption

model of services such as Service Adaptability, performance, resource consumption and interoperability with other UI technologies.

For SAP NetWeaver Composition Environment, the UI programming model consists of three major components, SAP NetWeaver Visual Composer, Web Dynpro Java and Standard Java technologies such as JSF. In contrast to AP/A1S, Visual Composer for SAP NetWeaver CE currently does not support UI patterns. The major challenge with the different programming models of the three major components is the limited interoperability especially at design time.

All of the stacks mentioned above also contain Adobe Forms as part of their solutions. These forms are used in both online and offline scenarios. In order to leverage Adobe Forms, the Adobe Form Designer is integrated into the design time environments. At runtime Adobe Forms add the Adobe Document Server to the solution landscape for accessing the form templates.

Controllers and Service Adaptation

A1S user interfaces consume services that are provided by business objects of the application platform (AP). One of the major findings is that, in most cases, user interfaces cannot use these services in the way the business objects exposes them. Similar experiences are also gained from iCOD development building composite applications for industries and earlier self service development,. To solve this problem without re-writing services, a service adaptation mechanism has been introduced for modeled user interfaces.

For pattern-based A1S user interfaces, the adaptation of services can be achieved twofold. During modeling the user interface, it is possible to do multiple service calls and join the results of these calls in order to fit them to the needs of the UI patterns. In addition to this kind of service adaptation, it is possible to create so-called controller objects which are implemented in ABAP and reside in the backend. A controller object typically covers capabilities such as transforming error messages into human readable messages, or displaying several business objects at a single screen. Very similar are the so-called transformed objects and transformation nodes which provide tailored views on business objects.

When consuming other service types such as stateless Web services, EJB services, or even classical RFCs, the service adaptation can be either implemented inside the business logic layer of a composite application or directly in the user interface layer. SAP NetWeaver Business Intelligence (BI) and SAP NetWeaver Exchange Infrastructure (XI) offer similar capabilities.

Common to all stacks is the absence of a generic query methodology. While AP/A1S has introduced so-called Query Response Nodes (QRN) which are technically the same as business object nodes which retrieve a list of business objects. SAP ERP needs to expose a specific service for each individual query. Both approaches do not allow the definition of the query in a way that the consumer of the service can generically access any kind of information including meta-data.

Unfortunately, the different frameworks for service adaptation and controller logic do neither share the same model nor do they share runtime engines. This creates silos of functionality on top of these technologies with limited interoperability.

Usage of User Interfaces in Composite Views and Guided Procedures

As of today, UI applications can be used within iViews in composite views, such as portal pages and object based navigation (see chapter 5). The Portal Content Directory (PCD) serves as single point of entry for the definition and discovery of iViews which are the representative of this kind of reusable UI applications. Content administrators compose portal pages from these iViews and define navigation between iViews using object based navigation. An iView is a very general abstraction and enables flexible embedding of almost any content that can run in a Web Browser in a loosely coupled manner.

UI applications need to implement a predefined interface in order to be used within guided procedures for parameter and context sharing. However, unfortunately not all SAP UI frameworks offer the capability to expose this interface and to map the context of guided procedures to the UI application. There is also not a common directory or repository that can be used for discovery of user interfaces that can be used in a guided procedure. Although guided procedures and portal iViews couple UI applications in a loosely coupled manner either in a sequence or in parallel, they do not share a common abstraction for a UI application.

4.2 Strategy

Figure 4-1 gives an overview of the UI strategy. The main objectives are a commitment to a multi-channel architecture based on core portal services and user centric services. Composite views such as portal pages are a composition of UI applications which are loosely coupled. “Loosely coupled” in the UI context means that several UI applications do neither share a single transaction nor references to the same business objects and data. Typically they can run in parallel and isolated from each other. Among loosely coupled UI applications, only a limited amount of data can be passed back and forth. The concept of loosely coupled UI applications also applies to interactive steps in user centric services and object based navigation. Compared to portal pages, the main difference is that those are running in a sequence.

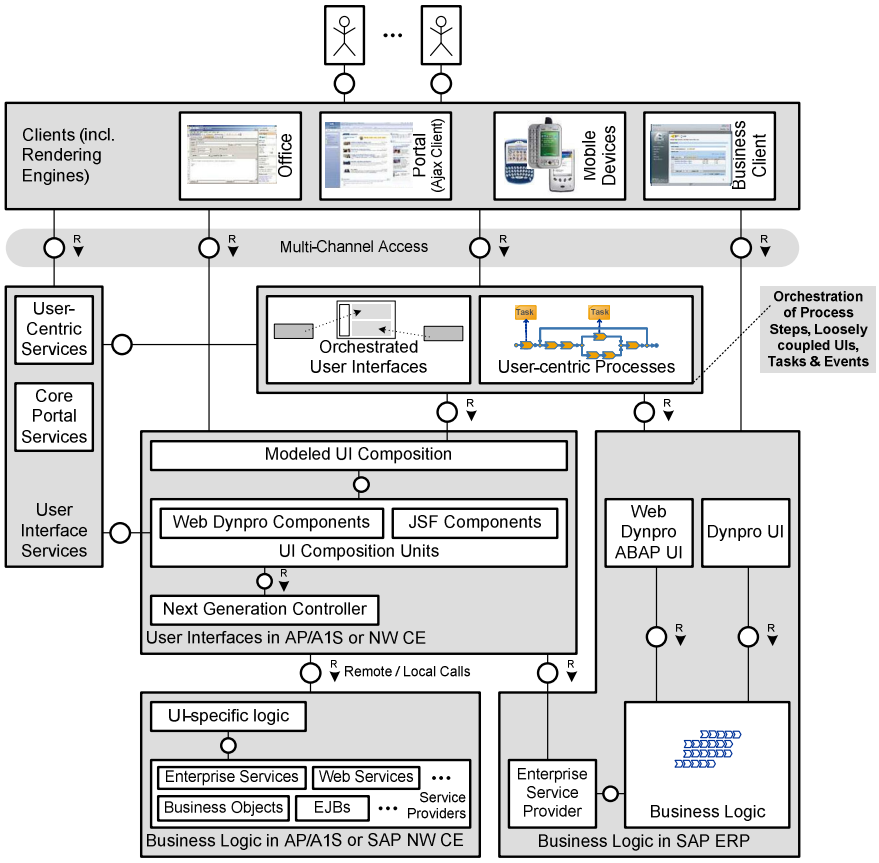

Figure 4-1 User Interface Strategy

In contrast to loosely coupled composite views, a single UI application follows a tightly coupled UI model. Tight coupling in the UI context means that different components of the UI application can share the same transaction and also share data via references in both read as well as write scenarios. The UI programming model for tightly coupled UI applications distinguishes between modeled UI composition and the development of UI building blocks. UI building blocks are either implemented in Web Dynpro Java or a Java standard technology such as Java Server Faces (JSF). Both can make use of a controller. This controller

connects to the different types of services offered either by the platform, by SAP NetWeaver Composition Environment, or through external Web Services – see section 4.6 for details.

Multi-Channel Architecture (MCA)

As shown at the top of Figure 4-1, a variety of user interface clients, such as rich clients, browsers, PDAs, or office applications such as Duet may access business applications. However, the core portal services and the user centric services are the foundation and are shared across the channels.

Trends and hypes in the UI rendering domain are coming and going frequently. To be as flexible as possible to adopt new UI trends whenever necessary, SAP will therefore continue to abstract the concrete UI technologies in the application models as well as in the core portal services with the consequence that some platform specific functions cannot be leveraged. As an example, this enables SAP to seamlessly integrate Web 2.0 capabilities such as Ajax or online collaboration features into the UI stack on different levels and for different devices without touching the application level models.

Integration of User Interfaces

The user interfaces of platform or composite applications can be integrated in user-centric processes, portal pages, and so on. This integration is achieved by loose coupling of different UI applications running either in parallel (portal page with several iViews) or in a sequence (user-centric process). In order to ease the modeling of portal pages and user-centric processes, there will be a catalog[19] for discovery of re-useable UI applications..

UI Programming Model and the Next Generation Controller

As of today the application user interfaces usually depend on the programming model of the services and require a tight coupling, since a certain type of service or controllers are used.. However, solutions on AP and SAP NetWeaver CE adhere to a common UI programming model, which includes UI modeling, Web Dynpro for Java, Java Standards and the next generation controller. The goal is the interoperability of the UI models at runtime as well as at design time. In order to break the dependency between user interface and the programming model used for the implementation of services a controller layer is introduced.

The task of the next generation controller is to map and adapt the different types of services between the user interface domain and the application domain. It transforms the different programming models of the services into the UI domain; see section 4.6 for more details.

As a consequence of the selective service enablement of SAP ERP, new SAP ERP user interfaces will be implemented as Web Dynpro ABAP applications as long as these UIs are integral part of SAP ERP. Classical Dynpro technology is used when existing applications have to be extended. In contrast, composite applications with an independent lifecycle built on SAP NetWeaver CE technology adhere to the UI programming model of SAP NW CE.

User interfaces of both SAP ERP and AP/A1S together with SAP NetWeaver CE must allow for easy UI integration in user-centric processes and portal pages.

Developing User Interfaces

For solutions on AP as well as composite applications built with SAP NetWeaver CE, user interfaces are defined by models, in which UI building blocks are composed and configured, as shown at the left-hand side of figure 4-1. UI modeling also includes the definition of the transactional boundaries for the UI applications.

[19] As of today it is not clear which catalogue and repository this will be

The UI building blocks are either implemented as Web Dynpro Java Components or through Java UI standards.

There will be a single tool framework based on Eclipse for developers to work on all UI models and navigate among the entities of these models for development and modeling on top of AP/A1S and SAP NetWeaver CE. For user interfaces in SAP ERP, the tools reside inside the ABAP Workbench. Visual Composer will focus on the needs of business analysts, non-developers and portal content administrators.

4.3 Multi-Channel Support and Client Strategy

A variety of clients can access SAP applications using appropriate channels. This multi-channel approach not only spans different devices but also includes offline applications such as Duet or mobile solutions. [20]

Device-specific Client-side Rendering

One component of the multi-channel architecture is the device-specific rendering engine. It uses the rendering capability of the corresponding client technology. For example, the browser loads an Ajax client from SAP NetWeaver Portal which renders HTML; the SAP NetWeaver Business Client has to be installed on a PC, but can use native technology for rendering; the same goes for Office (Duet) applications that use XAML, a Microsoft-specific technology.

UI Services

Nevertheless, the foundations of these UI rendering technologies are the UI services provided by SAP NetWeaver through user centric services and core portal services. These services offer all functions that are needed for the different channels. They serve as single point of entry for all channels. Therefore the service enablement of the core portal functions and user centric services are the key activities to introduce multi-channel support. The browser-based Portal UI client as well as the NetWeaver business client are the most prominent access channels for end users and will be a strategic component in SAP's product portfolio [HMM06].

All online applications should use the same UI programming model and modeling methodology for the different channels. However, it is not envisioned to go for a "one size fits all" approach. In general, user interfaces for multiple devices need multiple UI models due to different screen size, performance, rendering capabilities and, of course, for different groups of end users.

As mentioned above, the multi-channel approach specifies that rendering engines are implemented on client side using the client's native technology. This opens the opportunity of having client specific options for presenting generic UI services to the user. As a result, these services behave much more user friendly and hence it much better fits the end users expectation. Especially for capabilities such as channels pushing alerts and task, online collaboration, search, or office and desktop integration, this approach gives an opportunity for a deep adoption of native technologies for a certain clients. The drawback of this solution is that user interactions for generic services need to be re-implemented for each client.

4.4 Integration of User Interfaces

To include user interfaces in different loosely coupled scenarios, the user interfaces of platform applications as well as those of composite applications should be exposed. The most prominent scenarios for inclusion are

- User-centric processes

[20] However, support for offline applications is not in the focus of this document.

- Portal pages
- Object Based Navigation

This applies not only for the user interfaces from solutions on top of AP but also for the user interfaces exposed by SAP ERP such as Web Dynpro ABAP applications as well as classical Dynpro transactions.

In order to enable a comprehensive usage of user interfaces in these scenarios, there will be a common way to call a UI application, to pass data back and forth, and to share context between loosely coupled UI applications. For a flexible construction of loosely coupled scenarios to include existing UI applications, a catalog of UI applications is required that serves as a single point of entry for their discovery. This will allow to easily compose new portal pages and user-centric processes.

It is a common principle that loosely coupled user interfaces share a context which can be passed between the different UI applications, no matter whether the UI applications are executed in a user-centric service, a portal page or used as navigation target in object-based navigation. This enables context-aware behavior of the applications as well as context-aware generic services such as search and navigation. In order to easily use common contexts in loosely coupled scenarios, there will be tools and runtime support for the definition of a context, the mapping of context data into a UI application and the modification respectively enrichment of the context from within an UI application.

4.5 UI Programming Model

Solutions on top of AP and SAP NetWeaver CE will adhere to a UI programming model comprising reusable UI building blocks as well as modeled UI composition. Both approaches are based on Web Dynpro Java. However, overlapping features in both approaches such as UI controls, expressions or UI components (see below) need to be harmonized between the different models. In the end, both approaches will converge into the SAP UI programming model for solutions on AP and SAP NetWeaver CE.

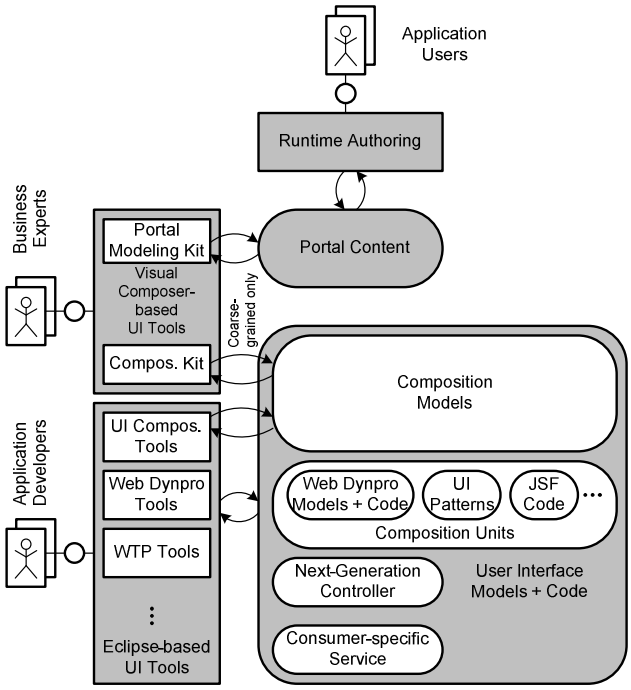

Figure 4-2 UI Programming Model for tightly coupled user interfaces in A1S & SAP NetWeaver CE

One Common UI Programming Model for AP and for SAP NetWeaver CE

Since SAP will offer further solutions on top of AP besides A1S, and since SAP will build composite applications based on SAP NetWeaver CE, the alignment of the UI programming models into one coherent and comprehensive model is a strategic goal [PMH06]. This UI programming model supports modeled user interfaces with easy-to-use integration of Web Dynpro Java components as well as Java UI standards.

Modeled UI Composition

Modeled UI composition enables composition of UI applications from coarse-grained UI building blocks in a tightly coupled environment. Tight coupling in this context means that it is possible to run in the same transactional bracket and to share references to data and business objects among different UI building blocks. UI composition combines flexible UI modeling as known from Visual Composer with the idea of UI patterns. The UI model can be seen as the glue between the UI building blocks. Hence, UI composition tools are required to combine building blocks implemented in different UI technology in a single, tightly coupled user interface. These tools are required for developers and therefore should be made available in Eclipse. UI Composition in Visual Composer will focus on the scenarios where non-developers adapt existing UI models to their needs. Both environments implement the same UI composition model.

Modeled UI composition is based on building blocks with standardized contracts defined as so-called composition units [GCH06]. Composition units define the subset of capabilities of UI building blocks needed to compose UI applications from them. As a consequence, modeling a complete UI application results in assembling composition units. In contrast, portal pages are a composition of loosely coupled iViews instead of tightly coupled composition units. UI composition also includes the methodology to connect the modeled user interfaces to the next generation controller for reading and writing data as well as for modeling service invocation.

UI Building Blocks

UI building blocks are implemented either in Web Dynpro Java, supported by a comprehensive tool set in Eclipse, or through Java standard UI technologies (in figure 4-2 depicted as “JSF”). Most of the UI building blocks are generic and configurable in order to adapt them to a certain usage scenario. Typical configuration options are the visible fields, the grouping of fields on the UI, some layout options and so on. Therefore, a configuration tool is also part of a UI building block – the tool allows configuring the usage of the UI building block in a UI model. This configuration tool can either be modeled or be implemented in the same technology as the UI building block itself. The combination of UI building block and configuration tool forms a self-contained unit.

However, some specialized UI building blocks can also be used without configuration since their behavior and presentation is completely determined by their specific task. Anyhow, there is no major difference between generic UI building blocks and specific UI building blocks, neither in terms of how to use them nor in terms of how to implement them.

Here is an example of what modeled UI composition and UI building blocks should comprise in future:

When thinking of an A1S floorplan or VC Freestyle UI model, the UI composition tools and the meta model deal with the glue between the UI building blocks. In this context it means:

- Selecting a floorplan which serves as a template of the user interface
- Discovering and usage of UI building blocks in the context and constraints of the selected floorplan
- Configuration of the UI building blocks in the context of the selected floorplan and UI application

- Modeling of data flow, data binding and navigation between UI building blocks and the controller. This also includes conditional branching dependent on state and state changes as well as calculation of fields in the binding and flow chain.

- Lifetime management of the UI building blocks at runtime

- Tracking of dependencies to UI building blocks for a comprehensive deployment strategy.

A UI building block deals with the implementation of a certain generic or specialized functionality. Typical examples are working with a list such as ALV, editing a record such as a form, or providing navigation between different aspects of a business object via a tab strip. In this context it means:

- The model or implementation in code of the functionality. For a building block such as ALV, this also includes generic functions such as sorting, printing, filtering and so on.

- The proper execution of the configuration of the UI model. This can either be achieved as interpretation of the configuration at runtime or through a generation step.

- Reading and writing of data according to the UI model including invocation of service calls.

Next Generation Controller

A user interface will consume services of the underlying business logic through the next generation controller, which provides flexible adaptation capabilities. The controller exposes a standardized interface to the UI building blocks allowing them to use different service provisioning technologies such as business objects, enterprise services, Web services, or EJBs. As a consequence, UI building blocks on SAP NetWeaver CE offer a much broader reuse when using the controller, while user interfaces on AP/A1S can benefit from a more flexible SAP NetWeaver Visual Composer modeling.

SAP ERP UI Programming Model

SAP ERP will implement new functionality using Web Dynpro ABAP Foundation. There is no further product-wide, general-purpose modeled UI composition approach in SAP ERP . Only selected Dynpro applications will be ported to Web Dynpro ABAP. The ABAP workbench is still the environment to develop UI applications. Nevertheless, user interfaces implemented in both Web Dynpro ABAP and Dynpro will be enabled for usage in loosely coupled scenarios.

However, due to the dual stack reality, Web Dynpro ABAP will not be enabled to expose composition units for the usage of Web Dynpro ABAP UI building blocks in the UI composition tools. Since the UI composition runtime is Web Dynpro Java, it will not be possible to use UI building blocks implemented in Web Dynpro ABAP in tightly coupled UI scenarios.

4.6 Next Generation Controller

A typical pattern solving the problem that a user interface should be decoupled from the business logic (provided by services in SOA), is the introduction of a controller. This also applies to distributed services [HKCS06], including:

- Mapping data including metadata passed to and returned from service invocations.
- Concatenation of data and calculation of new data including metadata.
- Chaining service invocations and creating views and projections on the result sets.
- Querying datasets including joins, views and projections
- Mapping errors and other information to human-readable message.

- Transparent transformation of different types of services and data sources into the user interface domain
- Managing state
- Providing exits for more complex coding

In general, most of the capabilities of a controller can be defined in an easy-to-use modeling environment. However, in order to keep the controller model simple and easy to use, special cases will require additional application-specific code. In addition, the ease of use depends on the underlying services and meta data. The consumption of a simple Web Service most probably requires more application-specific code and additional modeled behavior compared to the consumption of a rich enterprise service. More details can be found at [NGC07]

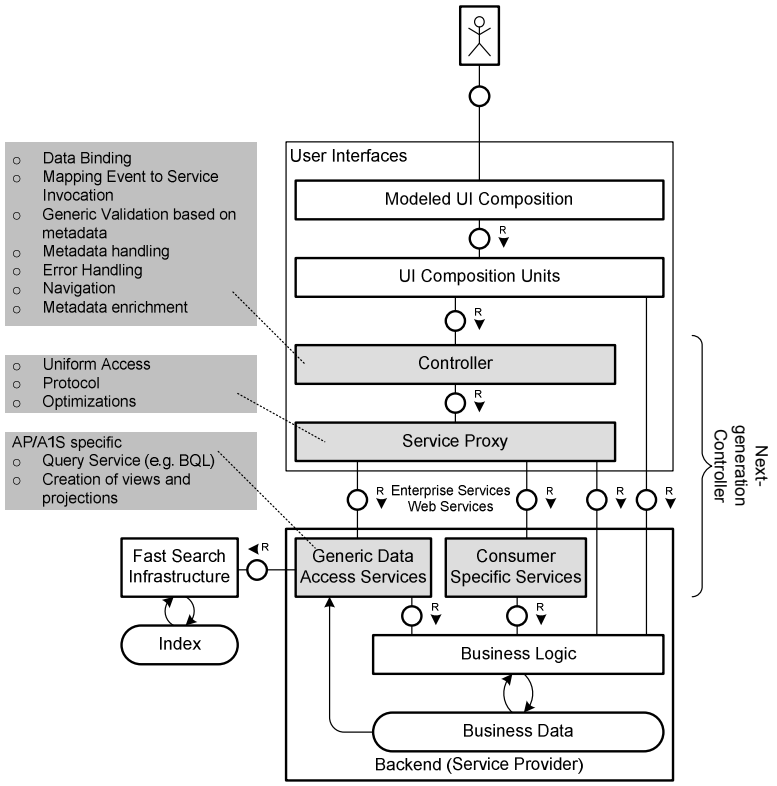

Figure 4-3 Next-Generation Controller

In the future, SAP will offer a UI composition environment and a UI model runtime that are able to consume different types of service (with constraints) from one single UI model. Therefore UI building blocks must be independent from the service types, keeping controller functionality as generic as possible. The next generation controller is the place where different service types and their capabilities (including the capabilities listed above) are mapped to the user interface domain and vice versa. The goal is to eliminate all dependencies from the UI domain to a certain service type and vice versa. In order to achieve this goal, the next generation controller must obey to interfaces and contracts defined by the UI domain as well as the interfaces and contracts defined by the service types. Hence, the controller plays the role of a mediator between these domains that maps and transforms between the domains. The future integration of a new type of service is therefore transparent for the UI modeling approach enabling broad reuse of the UI modeling capabilities. One of the controller's main tasks is the exposure of services, data, metadata, actions, etc. for extremely easy, model-driven consumption in the user interface. This will dramatically reduce complexity in the UI modeling domain and boost efficiencies when it comes to rapid development or easy changes of user interfaces.

The term “Next Generation controller” should not be seen as a new layer of software that is added to the UI stack. It is more the availability of a set of capabilities which can be located on different layers of the software stack. In the UI layer there is a controller that might be integrated into the UI programming model serving data binding, event mapping, generic validation, and error handling. Also on the consumer side there must be some mapping of different types of services into the UI domain, e.g. inside a service proxy. On the other hand, there is a need for service adaptation capabilities executed on the service provisioning system that might be deeply integrated into the service infrastructure, fast search infrastructure or even the data base. Finally there is a need for consumer-specific services on top of the service platform AP which bundles service calls and other functionality into a consumer specific service.

In order to reach the goal of a common controller approach, several existing UI frameworks have to be revisited, resulting in a decision to consolidate, align, or phase out the framework. [21]

[21] The incomplete list of the most popular framweworks that are affected by service adaptation and a common controller approach are Web Dynpro and CMI (Common Modeling Interface), Service Adaptation in Visual Composer, A1S Controller Objects , Composite Application Framework (CAF), ESF / GCP, Galaxy and Guided Procedures, Composed Pattern runtime and others.

5 Composition and Composite Applications

One of the most important benefits of enterprise SOA and its service-enabled platform applications is the possibility to develop individual, tailored applications on top of them – composite applications.[22]

Composite applications are dedicated to a specific business need of the user, focusing on usability and flexibility. In contrast to platform applications, scalability and legal compliance for example are not that important. Composite applications can be built by partners and customers as well, therefore the number of composites will exceed platform and extension applications by factors and they will have a shorter life span.

Development productivity and simplicity is a key to success. That's why the model-driven approach is so essential. Flexibility not only of composite applications but also of the underlying platform is also a key success factor. Composite applications are highly specialized consumers of services and events. Although they may provide services and events as well, they are typically not designed for reuse. In some sense composite applications are for the enterprise what Web 2.0 mashups are for the World Wide Web.

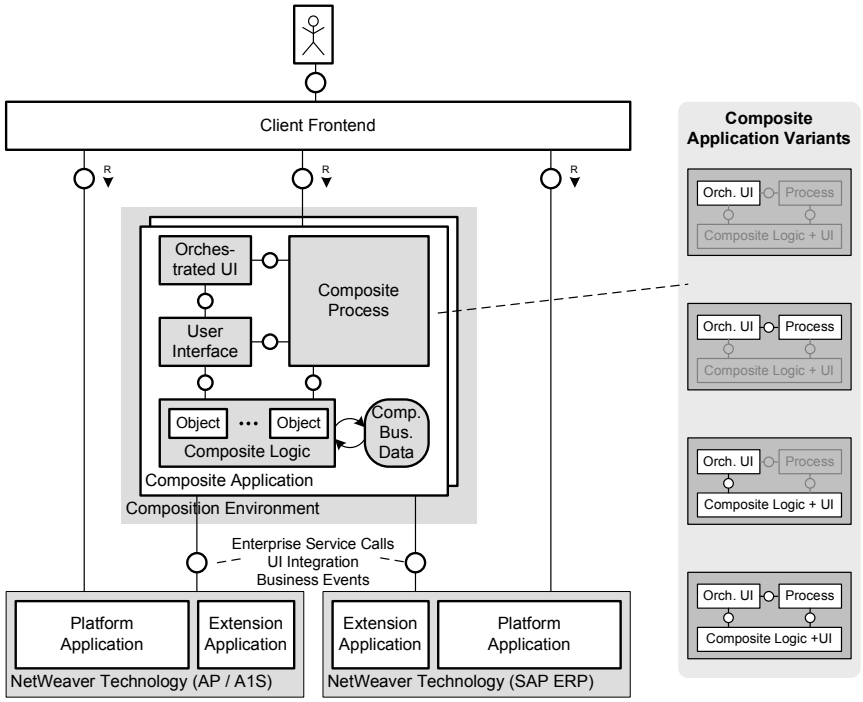

Figure 5-1 Composite Applications

Figure 5-1 distinguishes different types of composite applications, depending on their purpose. A simple composite application just bundles composite processes and orchestrated user interfaces for a dedicated platform, but very often business logic has to be adapted to the user's needs. This results in composite logic calling services of the underlying systems. To fill gaps of the underlying application, a composite application may require own objects with persistent data. The sophistication of composite applications with regards to

[22] Definition from Wikipedia (http://en.wikipedia.org/wiki/Composite_application): In computing, the term composite application expresses a perspective of software engineering that defines an application built by combining multiple services. A composite application consists of functionality drawn from several different sources within a service-oriented architecture (SOA). The components may be individual web services, selected functions from within other applications, or entire systems whose outputs have been packaged as web services (often legacy systems). Composite applications often incorporate orchestration of "local" application logic to control how the composed services interact with each other to produce the new, derived functionality.

own business logic and persistent data is determined by the capability and flexibility of the underlying platform.

Enriching the two main platforms SAP ERP and AP/A1S with composite applications is the main focus of SAP as indicated in figure 5-1. Composite applications may of course also invoke services from other sources to extend the functionality and scope.

Composite applications are designed for an independent lifecycle, since they can be built by SAP, partners, and customers. Of course SAP can decide to integrate a composite application into a solution (such A1S or SAP ERP). This should only be a packaging decision.

Besides addressing new user groups with new or extended business processes, composite applications also have implications on the technology, tools and frameworks side. This chapter describes the status and the strategy of composition and composite applications at SAP.

5.1 Status

Tools and Frameworks

In principle, composite applications can be developed in any environment such as .Net, Java or scripting languages. With SAP NetWeaver Composition Environment (SAP NetWeaver CE), SAP provides an own Java-based technology for building composite applications.

SAP NetWeaver 2004s was the first platform to offer composition tools such as the Composite Application Framework (CAF) and Guided Procedures. However the usage of these tools is not promoted due to missing integration and maturity, which would lead to high cost of development. Focus of SAP NetWeaver 2004s is to support service provisioning of SAP ERP 2005. SAP NetWeaver New York focuses on AP/A1S and on service provisioning, extensibility and flexibility of AP/A1S.

Based on the experience with SAP's existing composition capabilities and their usage in composite application development at SAP such as xApps, Composite Fast Track (CFT) and Industry Composites Development (iCOD) and at partners, SAP will have one preferred environment for building and running composite applications on SAP technology. The SAP NetWeaver Composition Environment, which will be shipped in 2007, is a new, integrated, coherent, and lean composition environment for SAP, partners and customers. It is based on Java EE 5 and other common standards, and focuses initially on lean consumption and consolidation of tools and frameworks. Subsequent releases will bring new innovative concepts such as the process infrastructure from the Galaxy project. In SAP NetWeaver CE there will be also future versions of the composite application framework (CAF) for service composition.

Application Development

On the SAP ERP side with its large installed customer base, several projects during the past four years focused on composite application development. SAP started with xApps, which were quite large applications mainly using conventional technology, and which moved now into standard application offerings. The Composite Fast Track (CFT) provided insight in developing small composite applications which were mainly single composite processes or orchestrated user interfaces.

Currently SAP's activities are mainly focused on iCOD, which targets industry-specific composite applications. iCOD started on SAP NetWeaver04s and has switched to SAP NetWeaver CE. For 2007 these composite applications will extend the core ERP processes which have a broad installed base like order-to-cash, procure-to-pay, manufacturing or asset lifecycle management. The industry focus will be on manufacturing industries.

In parallel, an ecosystem of partners is already developing composite applications which are certified by SAP.[23] The majority of these partners migrate their own solution to SAP NetWeaver and connect them to SAP via composite processes or orchestrated user interfaces. In 2007 it is the goal to let these partners move towards SAP NetWeaver CE.

On the AP/A1S side first activities to enable building composite applications on top have started. AP/A1S development together with NetWeaver developed a concept how composite applications, service provisioning and platform flexibility can play together.

5.2 Strategy

Composite applications allow covering the long-tail of processes

SAP will complement its enterprise SOA strategy with the ability for partners and customers as well as for SAP itself to build and offer composite applications on top of a stable, compliant, scalable and open platform. Composite applications will be used to adapt and individualize business processes to the specific needs of industries, micro-verticals and even customers.

The number of composite applications created will exceed the platform and extension applications by factors. Therefore it is essential that the creation of such composite applications becomes much more efficient with respect to design, deployment, and implementation aspects. SAP has chosen the approach of model-driven development and Java to achieve that goal. Also the life span of such applications is much shorter than the life span of the platform or the extension applications.

SAP NetWeaver CE will be SAP's technology offering for building and running Composite Applications

Composition on top of the platform may take place in different environments. With NetWeaver Composition Environment (SAP NetWeaver CE), SAP will offer a preferred environment to build and run composite applications, especially against SAP's business process platforms.

SAP NetWeaver CE comprises only those components needed to fulfill the task. Small footprint, integration of tools and frameworks as well as support of standards will drive down TCO and TCD so that smaller partners and customers will be able to efficiently build composite applications with SAP NetWeaver CE.

SAP NetWeaver CE will include the business process management (BPM) and user interface capabilities relevant for composition as described in the previous chapters. It is therefore essential that BPM and UI are aligned in a consistent and homogeneous composition environment.

In some cases SAP will offer composite applications on another basis than SAP NetWeaver CE to reach more users and get more usage to the platforms, for example with Duet in Microsoft Office environments.[24]

[23] The actual data can be found under the following path: www.sap.com -> Partners -> Global and Local Partner Directories – klick on "partner directory" –> choose "Search for Solutions" and choose "Composite Applications" as the "Certification Category"

[24] Composite Applications can be implemented in different technologies. Partners and customers might use other technologies for building composites. With NetWeaver CE, SAP will offer and promote an own composition environment and SAP will push the usage of SAP NetWeaver CE for that purpose.

Improving platform flexibility will allow more variations in Composite Applications

Not all tasks needed to build composite applications can be done decoupled from the underlying solutions. As already described in the Big Picture, BPM and user interface chapters, a certain degree of flexibility is needed inside the platforms too. The range and sophistication of composite applications varies for the different underlying platforms, because they offer different capabilities with respect to flexibility.

Flexibility is needed in various areas:

- Extensibility of data models[25]: Composite applications might require additional attributes for existing platform business objects with a certain business logic assigned to it.
- Flexibility on the user interface level: As already described in the user interface chapter, the platform application user interfaces must be able to participate in orchestrated user interfaces and exchange context. This requires adaptation of the user interface or even offering an alternative tightly-coupled transactional user interface on top of platform business objects.
- Business process flexibility refers to the ability to hook composite applications into, before or after platform processes without compromising the latter's consistency. This includes offering the proper enterprise services and business events, carrying new attributes along the process chain or providing the proper extension points in a very visible way.

The more flexibility the platform provides, the simpler the composite application can become. Probably some composites simply cannot be built at all without certain platform extensions. It will also be an advantage for SAP NetWeaver CE to leverage the platform capabilities. However SAP's two platforms SAP ERP and AP/A1S will differ considerably.

- For AP/A1S such flexibility mechanisms will be offered as described above and in previous chapters
- For SAP ERP such extensions are restricted to the existence of certain BAdIs[26]. Therefore in certain cases a dedicated service pack might be needed which then contains the specific code extensions on the SAP ERP side.

Composite Applications will be designed for a separate lifecycle

Due to their nature composite applications will be designed to have their own lifecycle independent from the platform. Therefore the platform has to obey compatibility rules and, in addition, has to allow deployment of extensions needed to run a composite application.

Of course composite applications can become part of a solution (such as A1S or SAP ERP). This should be only a packaging and lifecycle decision, and have no implications on the technical design of a composite application.

The main consequence derived from the separate lifecycle is loose coupling. Composite applications only communicate stateless and ideally asynchronously with the underlying platform applications. Interfaces must remain stable and compatible. Therefore, using AP/A1S core services, which are synchronous and possibly stateful, is not allowed for composite applications running on the composition environment.

In case a composite application, which has to be loosely coupled with the platform, requires additional functionality of the platform, tightly coupled parts such as transactional user interfaces or extensions to business object's data model have to be deployed to the platform.

[25] Extensibility of data models typically goes hand in hand with the extension of a user interface to enter the data including some value help or even more complex validation logic. Or the additional information is used in business process and will be used in enhancements of existing business processes.

[26] BAdI: Business Add-Ins, an extension mechanism

Composite Applications will be provided for both platforms – SAP ERP and AP/A1S

AP/A1S and SAP ERP will offer the same technical capabilities for building composites (exposing services, events and enable reuse of user interfaces) as far as possible.

However semantically, both will differ. There will be no common abstraction across both platforms. Differences in master data, organizational data and configuration data will lead to semantically different interfaces.[27]

As a consequence composite applications typically will either be built on top of SAP ERP or on top of AP/A1S. This does not mean that composite applications only use services and events from one platform. They can reach out to the web and to other applications and consume services or events from there. But one of the both platforms is the leading one.[28] If SAP or a partner wants to offer composite applications spanning SAP ERP and AP/A1S or running against both platforms, this feature has to be added to the Composite Application explicitly.

SAP will intensify the composite application efforts in the SAP ERP and in the AP/A1S space.

- On the SAP ERP side, the findings of iCOD will be applied and expanded from there.
- On the AP/A1S side SAP will enrich the solution with composite applications.

Decisions which composite applications to build and were to invest are not taken yet beyond the 2007 plans.

An Ecosystem of partners will drive the topic of Composite Applications

Similar to Web 2.0 where the community plays a crucial role, SAP cannot cover the long-tail of business processes by itself. Therefore SAP will foster a partner ecosystem and encourage the creativity of the partners and customers. This will be done for SAP ERP customers and later also for AP/A1S customers as there the partner contribution is even more critical for success. Through that ecosystem, SAP will drive usage of the platforms.

Partners might use composite applications to integrate their own applications with SAP applications at the customer site. They might compete amongst each other or even with SAP.

There will be an integrated toolset for developers based on consolidated metamodels residing in one repository

Low cost and high efficiency of development of composite application is essential for the success in the ecosystem. To achieve this, a common repository, a consistent set of metamodels residing that repository and a set of integrated tools is needed.

- SAP will offer the Modeling Infrastructure (project name: MOIN) as the repository
- The metamodels for composite application development including the metamodels needed to support the platform flexibility will reside in that repository.
- For developers Eclipse will be the integrated tool environment where the standard Java plug-ins will be available as well as the tools around UI, BPM and composition. If special users like business experts need their own tools then they should get dedicated tools with reduced function set in their preferred environment additionally.

[27] There will be a semantical alignment according to the so-called PIC-process and global data types (GDTs) will be used where possible for both platforms.

[28] Composites can be built on top of other applications or platforms too like on top of SAP CRM. This is not yet in the scope of the document however. The use case of composites on top of non-SAP platforms is also not considered.

6 Conclusion

An architecture strategy for enterprise SOA has to comprise the service provisioning and the service consumption side. To address customer needs and expectations, SAP has to provide technology to build and run composites – views, processes, and complete applications alike – and deliver them side by side with partners.

Especially in business process management and user interfaces, SAP benefits from using the same toolset and underlying technology for both, platform applications and composite applications. The flexibility and integration that can be reached with this approach requires well-defined enterprise services and business events as well as platform and extension applications with appropriate extensibility features. According to this strategy, SAP will focus on:

- **Service Provisioning**
  Provide a good choice of enterprise services and business events on both platforms, SAP ERP and AP/A1S. Offer platform flexibility features on AP/A1S. Enable consumption of user interfaces in composite processes.
- **Service Consumption**
  Provide SAP NetWeaver Composition Environment (SAP NetWeaver CE) as the tool and technology environment for composite applications, integrating open standards and the aligned business process and user interface technology. Optimize service consumption in the UI. Deliver an interesting choice of composite applications.
- **Technology Innovation and Alignment**
  Integrate missing business process technology in both SAP NetWeaver CE and A1S (project Galaxy). Align the user interface technology of SAP NetWeaver CE and A1S (UI patterns, SAP NetWeaver Portal, SAP NetWeaver Business Client). Align SAP NetWeaver Business Client and Portal UI on top of service-enabled portal platform.
- **Design Tools and Infrastructure**
  Consolidate the multiple design tools into an Eclipse-based modeling environment. Consolidate the metamodels to a coherent and interoperable set. Move models from ES Repository and other storages for process and user interface models to a new MOIN-based[29] “enterprise repository”, and address their relationship to configuration, administration, monitoring, and others.

[29] MOIN stands for MOdeling INfrastructure.

7 Appendix

7.1 Process, Service and Event Infrastructure

Having discussed business processes and process models more from an application standpoint, this section takes a brief look onto the underlying SAP NetWeaver technology. There is not a sharp boundary between process, service and event infrastructure[30]. Therefore the key capabilities of all three are visualized together in the following Figure 7-1.

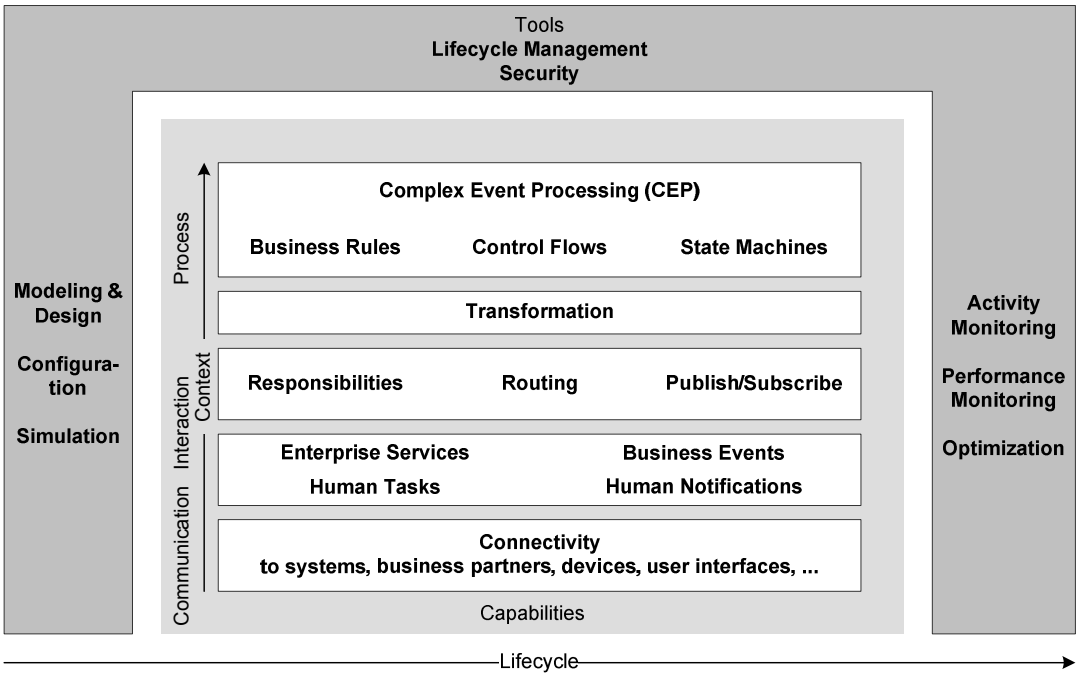

Figure 7-1 Process, Service and Event Infrastructure

An integrated tool environment must provide the ability to manage business processes across the complete lifecycle including modeling/design, configuration, simulation, deployment, execution, monitoring and optimization. Activity monitoring refers to the ability of taking actions upon occurrence of significant events; performance monitoring allows to measure key performance indicators of processes. Security requirements from authorization to secure communication must be met across the whole lifecycle as well.

Software systems and users are part of business processes. Connectivity to systems within and across enterprise boundaries and to devices for human or non-human usage is therefore a necessary enabling technology.

Four different types of interactions represented by enterprise services, business events, human tasks and human notifications[31] can be distinguished. Services and events are used for (direct) interactions between software systems, tasks and notifications are used by software systems to address humans. Services and tasks are the means to request or trigger the execution of an activity by a software system or human, respectively; therefore determination of responsibilities or routing of requests according to roles or

[30] Service and event infrastructure are often subsumed, sometimes together with other parts of process infrastructure, under the term enterprise service bus (ESB) or enterprise SOA infrastructure

[31] According to the proposal for the WS-Human Task standard by IBM and SAP, human tasks are services "implemented" by people and notifications are used to notify a person or a group of people of a noteworthy business event.

organizational structure is a prerequisite. Business events and human notifications, in contrast, follow a publish/subscribe model where interested parties (software systems or humans) subscribe to published information; in this model no assumptions can be made whether there are actual subscribers and which action they will take upon receipt of the information.

Single interactions or activities need to be combined in structured or ad-hoc ways to achieve a certain business outcome. Control flows and state machines are well-known technologies for coordinating multiple activities in a prescriptive (control flow) or reactive/event-driven (state machine) form. Control flows here refer to flowcharting techniques that specify the order[32] of performance of activities; state machines refer to techniques that specify transitions from one state into another based on the occurrence of events[33]. Business rules are used in different ways:

- Reaction rules (or event-condition-action, ECA, rules) define what to do under which conditions if something happened. Outbound process agents in AP are, in fact, a certain kind of reaction rules.
- Derivation rules define how to derive or infer new from given information. Determination of the responsible for a certain task is an example of a derivation rule.
- Constraint or compliance rules define policies that have to be met, for example: 4 eye approval is required.

Complex event processing (CEP) is an emerging technique focusing on extracting information from clouds of events created in enterprise IT and business systems. First commercial applications were in business activity monitoring[34], for example to monitoring conformance to service level agreements or to detect fraud. In essence, CEP applies event-condition-action rules to streams of events[35].

[32] Order is meant in the general sense: not only sequential but also parallel or conditional behavior may be described.

[33] Status and action management in AP is based on such a technique to describe which actions on business objects may or must not be executed depending on the current state. It is used in a passive way, though, and does not automatically trigger state transitions of business objects even if they were allowed.

[34] Currently, SAP's business activity monitoring is not based on CEP techniques. CEP will help to enable more and new scenarios centering on the resolution of significant events as, for example, envisioned by the Live Enterprise project.

[35] Streams of events can be partially ordered, for example by time, causality or aggregation. In contrast to "traditional ECA rules" the partial orderings will be taken into account in CEP.

7.2 Further Readings

[AMR06] *SAP Nears End of SOA Journey and Offers a Peek at a New Product*, Jim Shepherd, AMR Research Inc., December 2006 (http://www.amrresearch.com/Content/View.asp?pmillid=20043)

[BBGL07] Architecture of AP/A1S – The Book, Rüdiger Buck-Emden, Jochen Boeder, Bernhard Gröne, Martin Lünzmann, April 2007

[ESA06] *Process Modeling and Process Integration*, ESA Architecture Series 2006, Detailed View No. 2, SAP AG, Walldorf, February 2006

[Gart05] *Business Process Management: Preparing for the Process-Managed Organization*, Michael James Melenovsky, Jim Sinur, Janelle B. Hill, David W. McCoy, Gartner RAS Core Research Note G00129461, June 2005

[GCH06] Early Concept of Composition Units. Yuval Gilboa, Markus Cherdron, Reiner Hammerich. https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/e109c167-5650-2810-d6b8-faced4f9fe22/CompositionUnits%20Early%20draft.doc

[HKCS06] Connecting Backend System to Web Dynpro Components. Reiner Hammerich, Holger Koser, Markus Cherdron, Frank Seeger. https://portal.wdf.sap.corp/irj/go/km/docs/room_project/cm_stores/documents/workspaces/e109c167-5650-2810-d6b8-faced4f9fe22/Connecting%20Backends.doc

[HMM06] A High-Level Architecture View on the NetWeaver Business Client. Reiner Hammerich, Brian McKellar, Filip Misovski. https://portal.wdf.sap.corp/irj/go/km/docs/corporate_portal/WS%20PTG/Product%20Architecture/Architecture%20Projects/User%20Interface/NWBC_WhitePaper_v1.0.doc

[NGC07] High level Capabilities and Goals of the Next Generation Controller. <Still missing>

[PMH06] User Interface Position Statement on Visual Composer, WebDynpro & Smart Client. Gilad Parann-Nisanny, Paul Mullaseril, Reiner Hammerich. https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/corporate_portal/WS%20PTG/Product%20Architecture/Architecture%20Projects/User%20Interface/User%20Interface%20Position%20Statement.pdf

[RoKr06] The 255-List, Dan Rosenberg, Klaus Kreplin (https://portal.wdf.sap.corp/irj/servlet/prt/portal/prtroot/com.sap.sen.wcms.Cockpit.Main?url=/guid/f0a6befc-226b-2910-7182-d17b922d917b)

For more information on **Product Architecture,** see SAP Corporate Portal → Product & Technology Group → Architecture.