VERSION: VNEXT

DATE: 2007-08-17

AUTHOR:

Jonathan Heller

Juergen Kremer

Hermann Lueckhoff

SAP ARCHITECTURE BLUEPRINT

Duet vNext

PROJECT NAME/CPROJECT TITLE:	**DUET**
SPONSOR/PROJECT INITIATOR:	**DOUG MERRITT**
PROGRAM/PROJECT LEAD:	**UDO WAIBEL**
LEAD ARCHITECT:	**JUERGEN KREMER**
DEVELOPMENT:	☑ SAPLabs, mainly in India, Israel
	☑ Partner/ISV Microsoft

Document History

Version	Date	Status (Comments)
0.9	17-August-2007	Draft version
1.0	31-August-2007	Version submitted for OCTO review

1 Market and Product Background of Project/Program

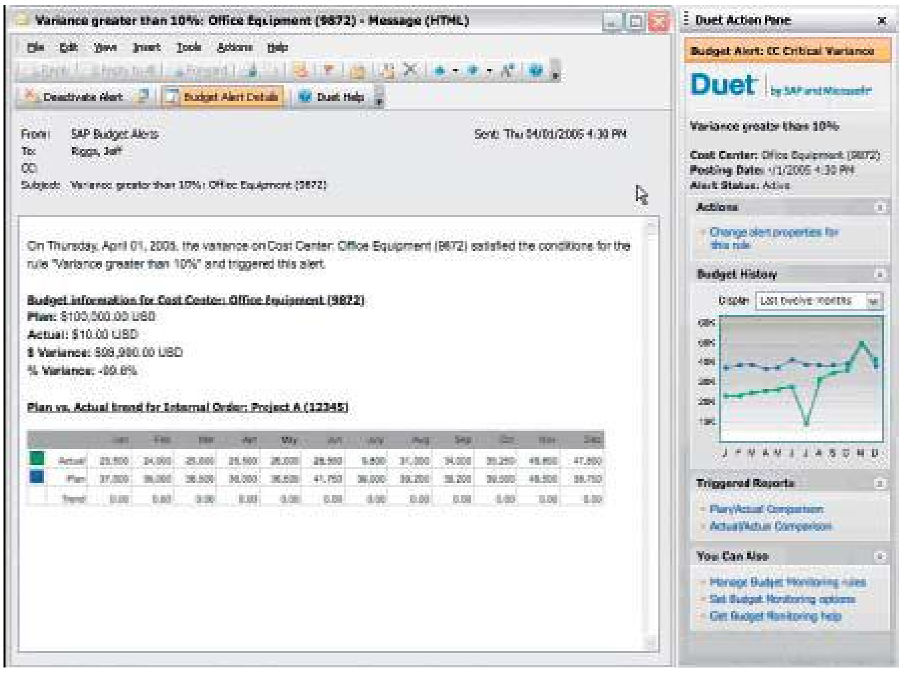
Planned release date:	Decision pending	
Underlying SAP NetWeaver release:	SAP NetWeaver 7.1	
Used SAP NetWeaver stacks:	☑ ABAP	☑ J2EE ☑ Net
Use cases targeted by the project/program:		
• Enable manager to control budget and receive reports within their Microsoft Office client • Handling leave process from Microsoft Outlook calendar • Manage appointments/handle time recording from Microsoft Outlook calendar • Employee/co-worker information with contextual information as Microsoft Outlook contact • Enable recruitment manager activities including interactions with recruiters and job candidates		
Strategic goals SAP wants to achieve with the project/program:		
• Integrate SAP functionality with the Microsoft Office environment • Increase number of SAP users • Enable customer to ensure process compliance • Proof of concept for consumption of enterprise services		
Mandatory software capabilities to address goals, use cases, and target market:		
• Robust runtime infrastructure to support Duet applications including seamless security, identity management, and role synchronization capabilities. • Centralized deployment and configuration infrastructure to manage a Duet landscape and its applications. • Development tools to improve Duet application development productivity.		

2 Architecture

2.1 Mission

Duet™ software for Microsoft® Office and SAP® applications – designed, developed, and supported by Microsoft Corporation and SAP AG – brings together the software of these two companies to give information workers a new level of information access and utility. In effect, Duet turns every Microsoft Office desktop into a personal window onto SAP enterprise applications.

.


Variance greater than 10%: Office Equipment (9872) - Message (HTML)
File Edit View Insert Tools Actions Help
Deactivate Alert
Budget Alert Details
Duet Help
Duet Action Pane
Budget Alert: CC Critical Variance
Duet by SAP and Microsoft
Actions
Change alert properties for this rule
Budget History
Display Last twelve months
Triggered Reports
Plan/Actual Comparison
Actual/Actual Comparison
You Can Also
Manage Budget Monitoring rules
Set Budget Monitoring options
Get Budget Monitoring help

From: SAP Budget Alerts
To: Riggs, Jeff
CC:
Subject: Variance greater than 10%: Office Equipment (9872)
Sent: Thu 04/01/2005 4:30 PM

On Thursday, April 01, 2005, the variance on Cost Center: Office Equipment (9872) satisfied the conditions for the rule "Variance greater than 10%" and triggered this alert.

Budget information for Cost Center: Office Equipment (9872)
Plan: $100,000.00 USD
Actual: $10.00 USD
$ Variance: $99,990.00 USD
% Variance: -99.9%

Plan vs. Actual trend for Internal Order: Project A (12345)

		Jan	Feb	Mar	Apr	May	Jun	July	Aug	Sep	Oct	Nov	Dec
🟩	Actual	23,500	24,000	25,000	25,500	26,000	26,500	9,800	31,000	34,000	35,250	45,850	47,850
🟦	Plan	37,000	36,000	38,500	36,000	36,500	41,750	36,000	39,200	38,200	39,500	48,500	38,750
	Trend	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Variance greater than 10%

Cost Center: Office Equipment (9872)
Posting Date: 4/1/2005 4:30 PM
Alert Status: Active

Figure 2-1: Budget Variance Alert

Figure 2-1 shows Duet user interface provided by Microsoft® Office Outlook® and Duet™ Action Pane.

2.2 Architecture Overview

2.2.1 Introduction

The underlying principle of all Duet applications is to enrich standard Microsoft Office objects (i.e. Microsoft Outlook groupware items, Word and Excel documents) with business data originating from SAP backend systems (SAP Business Suite systems). By attaching this data payload and providing respective user interface extensions directly embedded into the hosting Office application, Office can be used directly to access and manipulate enterprise data and related processes without leaving the familiar Office user interface.

As per Duet 1.5 the vast majority of all Duet applications is still based on Outlook. Duet applications extend standard Outlook objects (e.g. mail messages, appointments, contacts, task) by adding application-specific data payload to Outlook objects. By doing so the semantics of an Outlook object is extended into the enterprise space (e.g. an Outlook task or email representing an approval step within an SAP business workflow) while preserving the same Outlook interaction model the end user is already accustomed to. Outlook objects with such extended semantics are also referred to as Duet items, Duet objects, or more technically bound items (Outlook items bound to SAP business objects).

Duet items are attached to a schema that describes the structure of the data extensions. Such a schema definition attached to a standard Outlook object constitutes a Duet bound item type. Duet allows defining meta data specific for each bound item type that describes which additional UI elements should be rendered whenever a bound item of the respective type is processed within an Office application. It also specifies which web services calls (create, update, delete (CUD)) will be used to synchronize any changes of the bound item on the client side with the associated backend business objects, and which transformation rules apply from the bound item representation to the web service signature and vice versa.

Example: An end user might open a workflow approval notification in Outlook. This bound item originates from an SAP backend system and has been pushed to the end user via various components of the Duet infrastructure. It contains SAP business data in Outlook custom properties, which are described by a respective schema. The Microsoft Office Add-on recognizes that this approval notification is a bound item based on certain values of predefined Outlook custom properties ("Bound Item Type"). The Office Add-on then looks up its meta data any UI extensions defined for this bound item type (approval item). It then interprets the meta data by rendering whatever UI elements are described therein (e.g. a task pane with specific content, or additional tool bar buttons). When the user hits the "Approve"-button, the item is marked as updated by Outlook. The data synchronization component of the Office Add-on will pickup all updated bound items eventually and will parameterize and call the appropriate CUD web services as specified in the meta data.

A Duet application can be thought of a collection of functionally related Duet items, which includes all necessary meta data, client-side code, content for the SAP NetWeaver Data Orchestration Engine (DOE), and backend logic necessary to make such Duet items functional. The Duet infrastructure described in this document is designed to support all aspects of such applications including runtime, design time, and deployment.

2.2.2 Overview

The architecture of Duet consists of three main elements (see Figure 2-2):

- An add-on to Microsoft Office on the user's client machine: The Office add-on handles communication between the client and the SAP Duet Add-on server. It contains a client runtime engine that provides a number of services such as deployment of metadata and .Net assemblies, data caching, queuing of outgoing service requests, etc. The add-on itself is provided by Microsoft.
- A box called "Duet Server", which consists of SAP and Microsoft components: The Duet server is used at deployment time to deliver all needed metadata, UI descriptions, and respective .Net assemblies to the client. At runtime all client requests go directly from the client to the SAP Duet Add-On server. The Duet server consists out of two parts, one provided by SAP and one provided by Microsoft.
- An add-on to the SAP Business Suite environment: SAP Duet Add-on consists of a DOE instance to provide typical services of a data synchronization middleware like data conflict detection and resolution, consolidation data base, or rule-based data distribution. The DOE instance is extended with certain Duet-specific services like:
    - o Service mapping (aka content-based routing), which allows selection of the appropriate backend system for a given service request.
    - o Groupware item management service allowing to create, update, and delete (CUD) Duet items.
    - o Transformation service to transform CUD request into DOE-based object presentation (Inbound Adapter) and vice versa.

It is important to point out that the native DOE client software (which e.g. is usually part of a mobile application) will not be part of Duet. DOE provides a layer to abstract from the specifics of a given data synchronization protocol (DOE channel handler). This concept will be used to wire up the DOE server part with the existing Duet client infrastructure using the CUD communication pattern already in place.

Figure 2-2 shows the above mentioned elements.

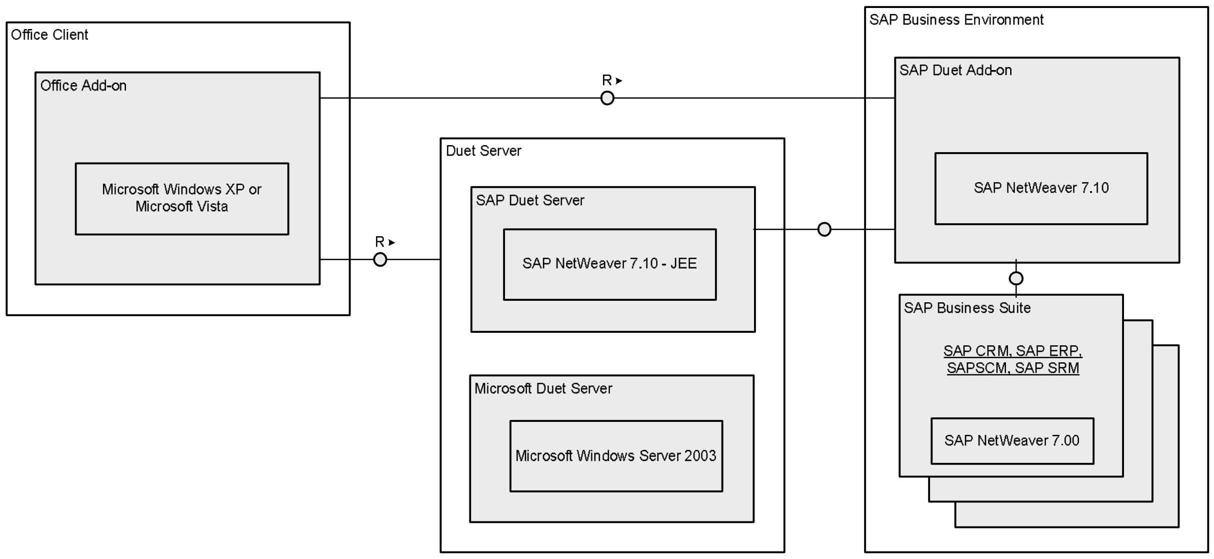


Figure 2-2: Duet overview

In the following each component is explained in more detail.

2.2.3 Office Add-on

The Office Add-on handles communication between the client and the SAP Duet Add-on server. If communication is using SAP LogonTicket, the Office Add-on first calls over to ticket issuer component for retrieving a valid SAP LogonTicket.

The Office Add-on provides three main services: a rendering service, a message processing service, and a data sync service.

The rendering service interprets XML-based meta data describing layout and content of Office UI artefacts (e.g. the Office task pane, Office tool bar). It also interprets so-called DuetML documents to render forms, which are usually placed inside regions of the task pane. DuetML is an XML dialect specifically developed for Duet applications, and resembles early versions of XAML (WPF).

The message processing service (CMP) processes messages created by the request handler of the Microsoft Duet server component using WebDAV protocol. All Duet-based messages are delivered to a hidden Outlook folder (invisible for the end user) with the Duet enterprise payload attached as a raw XML blob. The message processing service reads the Duet message in this hidden folder, transforms the XML payload into Outlook custom properties according to mapping rules specified in the respective meta data, and places the processed messages as Duet items into the appropriate Outlook user folder.

The sync service is responsible for queuing all created and changed Duet items and for processing the queue items by submitting respective create, update, and delete (CUD) web service calls to the backend.

A detailed description of the Office Add-on and its architecture can be found at Duet - Microsoft Planning Guide.

2.2.4 Microsoft Duet Server

Microsoft Duet server contains two main components, meta data service and request handler. Meta data service is responsible for distributing Duet content to all Duet clients in a given landscape in a role-aware manner. It also handles .Net assemblies, which are referenced by this meta data (so-called code-behind). The provisioning of the .Net assemblies to the client is done via the deployment component that could be seen as part of the meta data service.

The request handler serves as web service facade of Microsoft Exchange Server. It mainly provides CUD services to manipulate Duet items. It communicates with Microsoft Exchange Server via WebDAV control messages.

2.2.5 SAP Duet Server

SAP Duet server contains the components service mapper, service provider, ticket issuer, meta data repository interface and tools. The service provider is used at deploy time for propagating the configured meta data from the SAP part over into the Microsoft meta data service. It supports a range of transformations of this meta data defined by so-called instruction files e.g. to provide role-awareness, localization, or merging runtime data into static XML.

As the client communicates via web services the web services end-points in a landscape need to be defined. This is done in the service mapper component.

The ticket issuer issues SAP LogonTickets based on Kerberos tokens, which are used for authenticating to the SAP servers. That means that all calls from the client are executed in a dedicated user context.

Tools are used in Duet within the administration environment to allow the administrator to configure the Duet landscape and customize some aspects of applications, such as links or labels and tool tips. This administration environment is a Java WebDynpro based application.

Meta data for Duet applications is delivered by SAP and made available within Meta data repository interface as part of the administration environment. It allows customers to configure certain aspects of Duet applications (e.g. visibility of certain controls) or the availability of a Duet application in a given landscape. These configurations can be then published to the Microsoft Duet server for further distribution to the relevant Duet clients.

2.2.6 SAP Duet Add-on

In the current version of Duet, the SAP Duet Add-on contains Java components, which will become obsolete with Duet vNext. Obsolete components are item handler, service map and service bundling. These components will be replaced by a new SAP NetWeaver 7.10 component called SAP Data Orchestration Engine (DOE), which will be extended by Duet-specific services.

The inbound adapter exposes the web services, which are called from the Duet client. It is also responsible to transform data passed with the web service calls into DOE messages. These DOE messages are processed by DOE to create and update DOE object instances that reside in the DOE consolidation database. Multiple channel handler interfaces are the runtime representation of the DOE, supporting synchronous and asynchronous calls.

The groupware channel handler builds the bridge between DOE and the outbound adapter creating the groupware items and is an implementation of the DOE channel handler concept. It is a way to abstract from specific communication protocols between a DOE server and a DOE client. allowing DOE to communicate with different clients using different protocols (e.g. DOE native client protocol, RSS, syncML, etc.). The groupware channel handler transforms various DOE message types (e.g. an update or notification message) into appropriate groupware CUD requests.

The outbound adapter contains the former item handler functionality i.e. sending items from the SAP side to the Microsoft Request Handler. It also abstracts from certain Microsoft Exchange server-specific aspects in the Request Handler API for a possibly broader future support of different groupware platforms.

The backend adapter is responsible for propagating changes to DOE object instances to the respective backend system using CUD requests. It uses the underlying DOE object instance to parameterize these calls. It may also contain certain release dependent and application specific logic. For each connected backend system there will be separate backend adapter.

Communication flows at runtime are depicted in Figure 2-3:

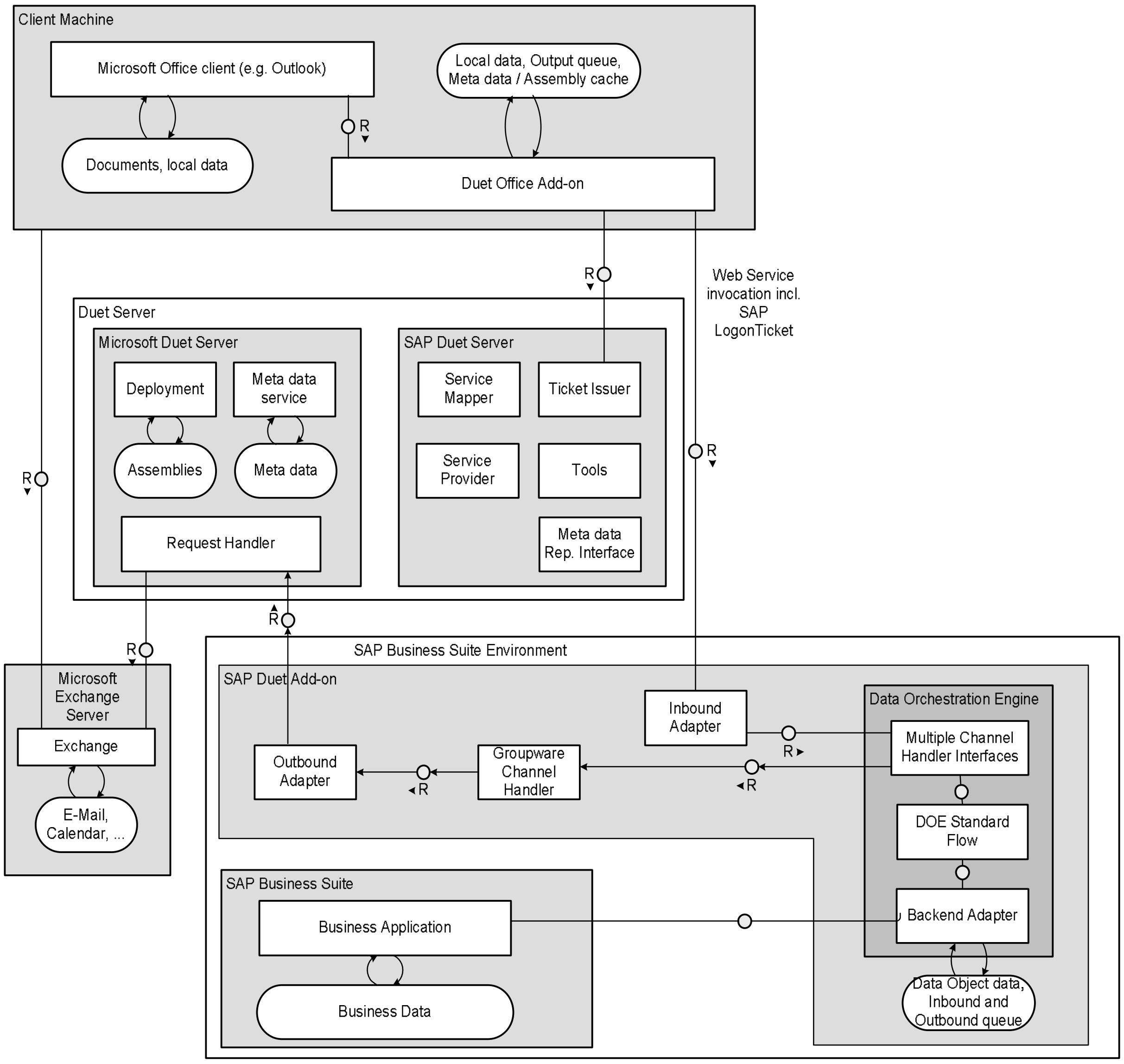


Figure 2-3: Architecture Overview Diagram (Static view at Runtime)

2.2.7 Request Sequence

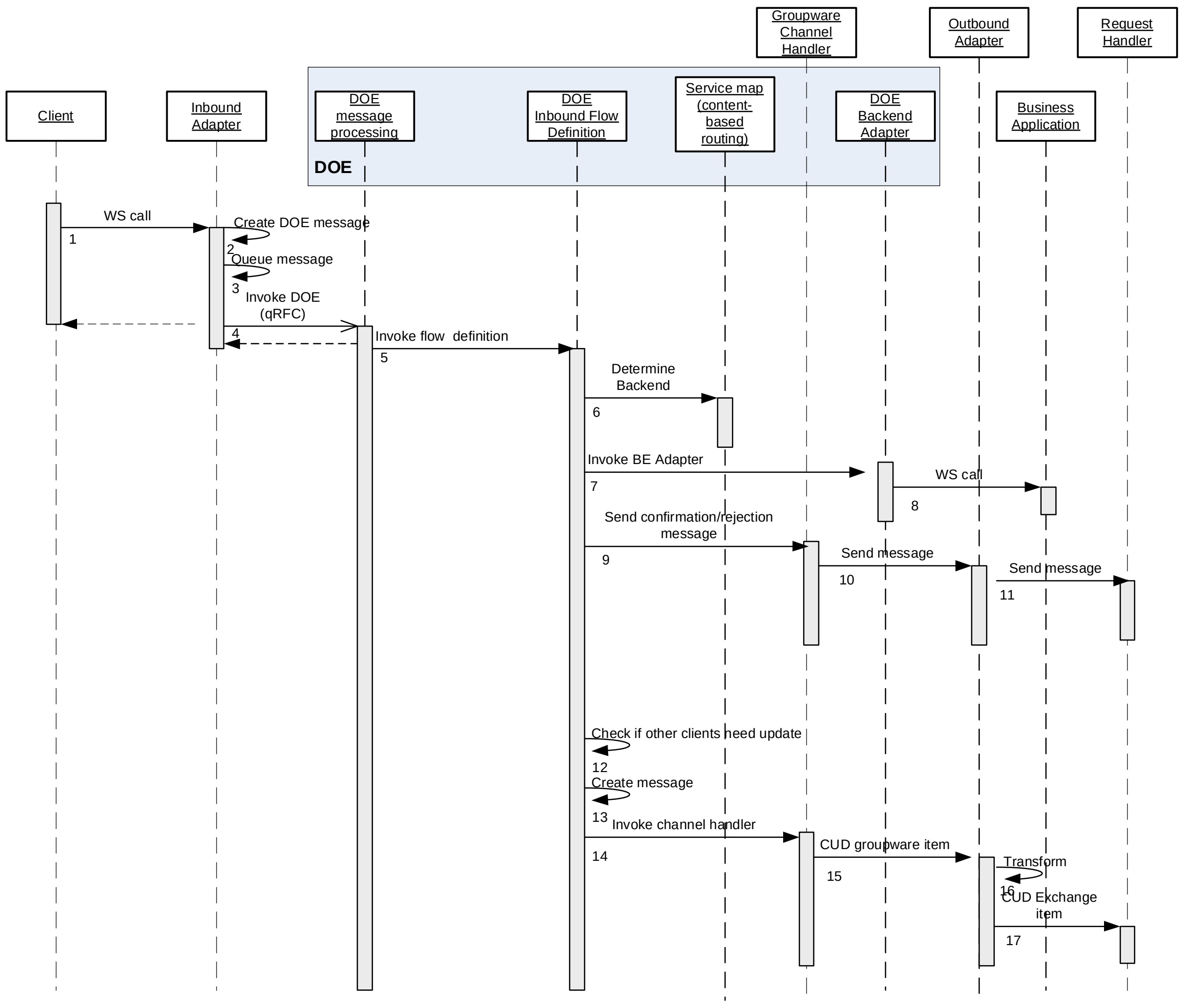


Figure 2-4: Architecture Diagram (Sequence diagram)

The sequence diagram above signifies a high-level view of operations involved when a Duet item is updated on the client. The operations on the client component are not modeled for the sake of simplicity. In our example the update web service call (1) is issued from the synchronization engine, which is part of the Duet `[Client]`. This web service call goes to the `[Inbound adapter]` component, which is responsible for transforming the web service call into a valid DOE message. . The resulting message is queued (3) and passed on to DOE with a qRFC call (4). `[DOE message processing]` represents the entry point into DOE processing. It invokes `[DOE Inbound flow definition]` (5), which is a sequence of customizable operation steps to manage the processing of inbound messages. There are different flow definitions for messages coming from the client (like in this example) and messages pushed from the backend. `[DOE Inbound flow definition]` calls `[Service map]` to determine which backend system the resulting web service call should be directed to. The appropriate `[DOE backend adapter]` associated with this backend system is then invoked (7). The respective CUD web service call is executed (8) with parameters mapped from the underlying DOE object instance. Based on the success of the call either a confirmation or rejection/error message is generated and the `[Groupware adapter]` (9) and `[Outbound adapter]` (10) are invoked to create a respective Exchange message (please note that confirmation messages will be ignored in most cases). This message is then handed over to `[Request handler]` (11) and will eventually make its way via Exchange server to the `[Client]`.

`[DOE Inbound flow definition]` then checks whether there are other clients using the DOE object instance (12) that has been updated by the incoming DOE message (e.g. CRM contacts shared across multiple users). In this case another DOE message is created (13) aimed to update all Duet items associated with the DOE object instance. `[Groupware channel handler]` and `[Outbound adapter]` are invoked (14,15). `[Outbound adapter]` finally sends the message to `[Request handler]` (17) from where it is passed on to Exchange server and replicated to the respective Outlook clients.

A different request sequence would be invoked in the case of a backend system initiating an update of a Duet item. This case is intentionally left out of this document for the sake of keeping it compact.

2.3 Design Time Aspects

Duet offers several options for customization and development of Duet applications.

Developing a full fledged application which requires coding in .NET and ABAP is supported for SAP internal development only. By contract, no client side coding by customers is allowed until a LoBi-based Duet when Microsoft will provide the required development environment. The Duet client side source code developed by SAP is not provided to the customers.

Duet offers the customer with the ability to customize the application via the administration environment (using a WebDynpro based UI) or to develop his own Duet application based on a template provided by SAP. Template based development does not require coding and is essentially configuration of meta data.

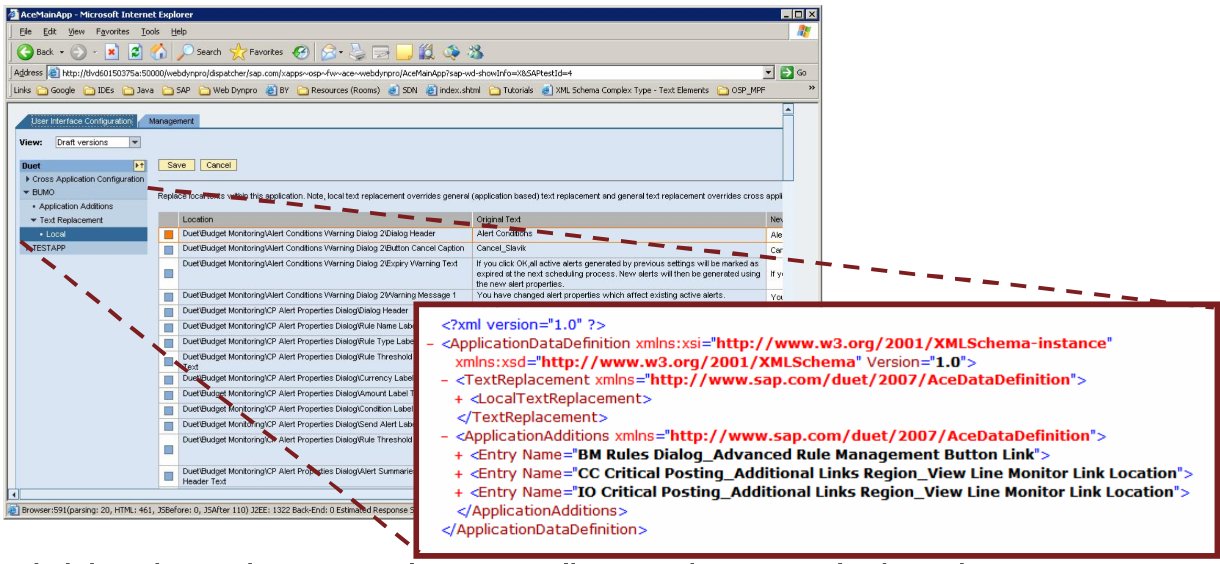
With the introduction of DOE enhancements to DOE data objects such as the introduction of additional fields will be supported via DOE design time and/or administration environment.

2.3.1.1 Application customization exposed to customers

Application customization (i.e. meta data modification) is done via the administration environment on the SAP Duet server and maintained in Java dictionaries. The customization exits are defined by the application developer, as visualized in Figure 2-5. Customization can control the look and feel of the application, for example text labels, tool tips or visibility of certain controls. The administrator also has the ability to add/modify links to external systems or files (e.g. URLs to the SAP Enterprise Portal, file system or any web site) and control the mapping of business data to Microsoft Outlook standard fields, for example the source of the address field in a contact.

A set of changes can be grouped together into a single batch and committed as a whole. The administration environment triggers the deployment of the modified meta data to the clients via the Microsoft meta data service.

The administration environment will ensure the compatibility of the DOE data model with the Duet meta data definition allowing the administrator to propagate the changes to the client. The system makes use of DOE versioning capabilities of the data model and the coexistence of multiple versions (old and new) at the same time.


AceMainApp - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://tlvd60150375a:50000/webdynpro/dispatcher/sap.com/xapps~osp~fw~ace~webdynpro/AceMainApp?sap-wd-showInfo=X&SAPtestId=4
Links Google IDEs Java SAP Web Dynpro BY Resources (Rooms) SDN index.shtml Tutorials XML Schema Complex Type - Text Elements OSP_MPF
User Interface Configuration
Management
Duet
Cross Application Configuration
BUMO
Application Additions
Text Replacement
Local
TESTAPP
Save
Cancel

View: Draft versions

Replace local texts within this application. Note, local text replacement overrides general (application based) text replacement and general text replacement overrides cross appli

	Location	Original Text	New
🟧	Duet\Budget Monitoring\Alert Conditions Warning Dialog 2\Dialog Header	Alert Conditions	Ale
🟦	Duet\Budget Monitoring\Alert Conditions Warning Dialog 2\Button Cancel Caption	Cancel_Slavik	Car
🟦	Duet\Budget Monitoring\Alert Conditions Warning Dialog 2\Expiry Warning Text	If you click OK,all active alerts generated by previous settings will be marked as expired at the next scheduling process. New alerts will then be generated using the new alert properties.	If y
🟦	Duet\Budget Monitoring\Alert Conditions Warning Dialog 2\Warning Message 1	You have changed alert properties which affect existing active alerts.	You
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Dialog Header		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Rule Name Labe		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Rule Type Labe		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Rule Threshold Text		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Currency Label		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Amount Label T		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Condition Label		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Send Alert Labe		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Rule Threshold		
🟦	Duet\Budget Monitoring\CP Alert Properties Dialog\Alert Summarie Header Text		

Browser:591(parsing: 20, HTML: 461, JSBefore: 0, JSAfter 110) J2EE: 1322 Back-End: 0 Estimated Response S

```
  <?xml version="1.0" ?>
- <ApplicationDataDefinition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" Version="1.0">
  - <TextReplacement xmlns="http://www.sap.com/duet/2007/AceDataDefinition">
    + <LocalTextReplacement>
    </TextReplacement>
  - <ApplicationAdditions xmlns="http://www.sap.com/duet/2007/AceDataDefinition">
    + <Entry Name="BM Rules Dialog_Advanced Rule Management Button Link">
    + <Entry Name="CC Critical Posting_Additional Links Region_View Line Monitor Link Location">
    + <Entry Name="IO Critical Posting_Additional Links Region_View Line Monitor Link Location">
    </ApplicationAdditions>
  </ApplicationDataDefinition>
```

Figure 2-5: Administration environment and corresponding metadata customization exits

2.3.1.2 Template based application development

Starting from Duet 1.5, customers will be able to develop their own application based on a template. The first template introduced with Duet 1.5 will be the approval pattern, which will enable customers to expose arbitrary approval processes based on SAP Business Workflow as Duet items.

A Microsoft Visual Studio plug-in based on Microsoft Domain Specific Language (DSL) serves as the development environment. The plug-in provides a template based environment focusing on UI definition, handling the Duet complexities transparently in the background. In Duet vNext further applications templates will follow, such as event management or business activity (calendar based activities).

Each template requires a software component provided by SAP. This component provides the runtime support for the template containing all the required code as well as the common meta data parts used by all the template-based application instances.

The template design time environment ensures that the template-based application references a certain runtime component version and complies with the required meta data definition schemas.

2.3.1.3 Application development environment (SAP internal)

Currently there is a variety of technologies involved to build a Duet application.

<u>Duet application consists of three parts:</u>

- Client side part, which includes Duet meta data, DuetML and/or XAML and code-behind packaged as a Java SCA and deployed to the SAP Duet Server.
- Middleware part, which includes DOE content developed in the DOE workbench. (ABAP) DOE content consists of DOE object definitions as well as inbound and outbound flows.
- SAP Business Suite enhancements (in ABAP). These enhancements are optional and mainly used for simple service composition in case the appropriate application functionality is not available or only available via non-compliant APIs.

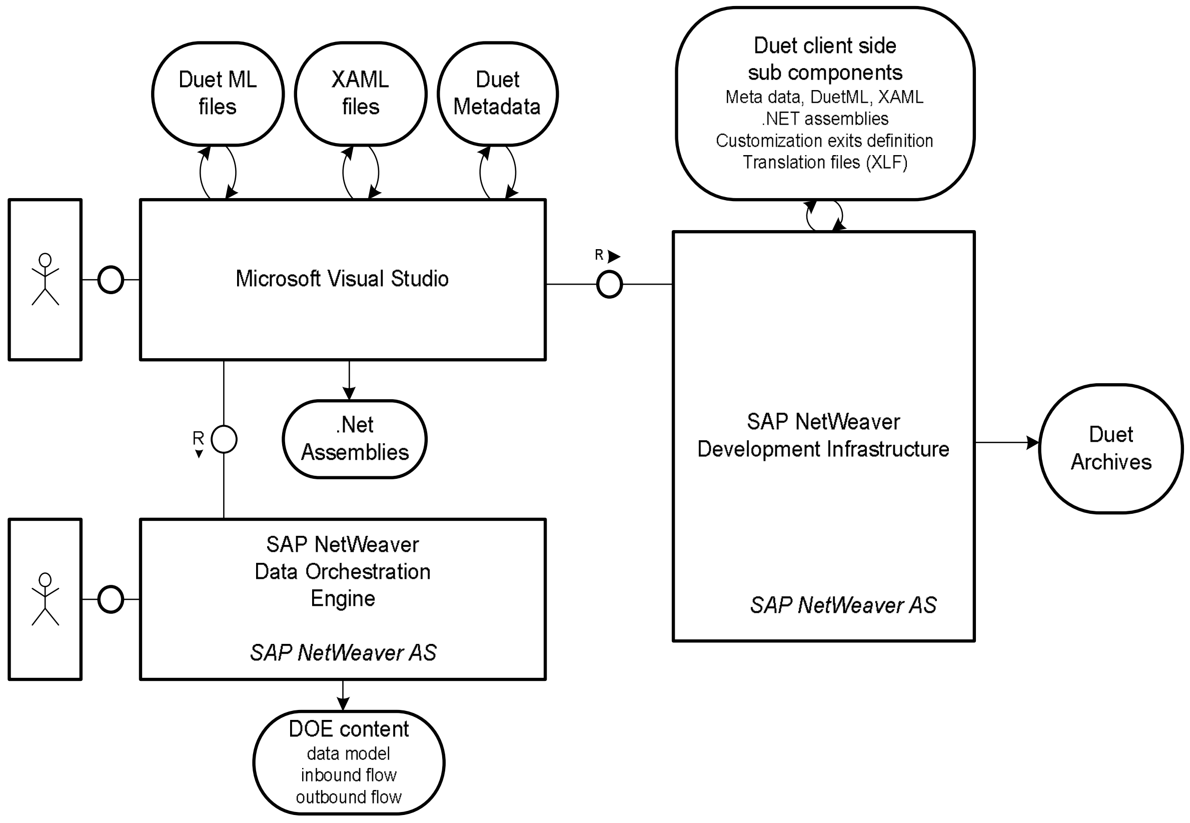


Figure 2-6: Architecture Overview Diagram (Static Structure at Design time)

Client side application development is done in Microsoft Visual Studio. Several Visual Studio plug-ins assist the developer in editing meta data and defining the UI (XAML or DuetML).

The meta data specifies various aspects of a Duet application, or more narrowly a Duet bound item type, e.g. its schema, which UI elements (action pane, tool bars, etc.) are active, web services used to update the state of a bound item, transformation rules to apply before and after web service calls, mapping rules to relate bound item data to certain standard Outlook custom properties, specification of cached web service call, etc. The meta data format is based on Microsoft IBF (Information Bridge Framework) and has been extended due to a number of Duet-specific requirements. Despite the relative richness of the meta data additional .Net assemblies are usually necessary to directly interact with the object model of the Office host application (Outlook, Word, Excel).

A special Duet project type organizes the project into a “Duet Standard structure” and integrates the various Duet plug-ins (provided by SAP and Microsoft) into a single environment. Microsoft code-behind and XAML forms are edited via the built-in Visual Studio tools. DuetML can be generated out of an existing Win Form. DuetML is a deprecated technology introduced with Duet 1.0 and will be replaced by XAML (the new UI markup language coming with .Net 3.0)

Since there is no standard SAP support for the translation of .Net assemblies, Duet design time environment offers design time support for the extraction of the needed strings together with their contextual information and the upload of the data into SAP NetWeaver Development Infrastructure in an XLF format. The standard Microsoft support for translation / localization does not meet SAP requirements since it does not retain the contextual information in the extracted strings.

All Microsoft-built content is put into the SAP NetWeaver Development Infrastructure (NWDI) via script (from the perforce production share) and translation is added to those. Delivery is done via Duet archives, which are created by the NWDI central build server.

The DOE content and the SAP Business Suite enhancements are developed by SAP standard tools and follow the standard SAP procedures for these environments.

2.4 Rationale for including DOE into Duet Architecture

Integrating DOE into Duet is the most substantial architectural change introduced with Duet vNext. There is a number of reasons for this decision:

- Current Duet architecture does not have a common robust mechanism to detect and manage conflicting update requests for a given business object, which could compromise data integrity. As Duet increasingly covers more complex scenarios with business objects shared across multiple users and with full 2-way sync capabilities required (e.g. SAP CRM) this problem will become more threatening over time. In case DOE would not become part of Duet an equivalent framework component would have to be developed by the Duet team.
- Current architecture doesn't offer a solution for handling and managing the rule-based distribution of data from the SAP Business Suite applications to the clients.
- Each application caters to its own needs, data distribution rules are hard coded, some reside in Java and some in ABAP with very limited to none customization capabilities.
- Duet has become part of the Information Worker Solution comprised of Mobile Business Applications (MBA), SAP NetWeaver Enterprise Search, and Duet. As both Duet and Mobile Business Applications are offline-enabled applications with very similar data sync requirements using a common data sync middleware component (DOE) is an obvious choice. TCD as well as TCO considerations – especially for customers deploying Duet and Mobile together – are important factors for driving such architectural convergence.
- At its current stage Duet infrastructure does not provide a robust mechanism to support customer extensions over the complete application life cycle. By leveraging DOE life cycle management capabilities Duet will be able to cover extensions on business object level. Again, this mechanism will be shared with Mobile ensuring basic coherence across those two application families.
- Using DOE as the Duet middle layer will allow us to remove Java-based runtime components still mandatory with Duet 1.5 architecture. Avoiding Java components as mandatory parts of the Duet runtime will reduce overall complexity and thus TCO.
- By 2009 Microsoft will roll out its new framework for developing Office Business Applications (OBA) code-named LoBi. LoBi architecture will also include a data sync middle layer very similar to the architecture proposed in this document. Part of our LoBi adoption strategy is establishing DOE as an integrated part of a pre-LoBi Duet technology stack. Our goal is to position DOE as the data sync middleware of choice for all SAP-based LoBi applications and to work with Microsoft on an integration layer that will ensure interactivity between DOE and LoBi in the best possible way. Not adopting DOE will force us into adopting Microsoft LoBi

data sync middleware, which will come with a similarly high price tag on the one side and will inhibit Duet-Mobile architectural convergence for years to come.

2.5 Total Cost of Ownership

One SAP Duet Add-on server can serve multiple ABAP backend; and one SAP Duet server can server multiple SAP Duet Add-on servers.

Investigations were done to have DOE based on SAP NetWeaver 7.0 as the SAP Business suite is based on SAP NetWeaver 7.0. But the costs are to high for down port, hence the SAP Duet Add-on server will be a server of its own. But from a DOE perspective an own server is recommended in regards to DOE performance.

2.6 Deployment

Deployment in the Duet case has two flavors: the deployment of the Duet components itself and the deployment of Duet content in a landscape.

2.6.1 Deployment options

The Office Add-on needs to be installed on every Duet client machine. Microsoft recommends the usage of System Management Server (SMS) infrastructure.

The two parts of the Duet server (Microsoft part and SAP part) can be installed on one machine. The SAP Duet server contains components packaged as one Java deployable and can be deployed on any SAP NetWeaver 7.10 Java system existing in a customer landscape.

Multiple SAP Duet Add-ons can be managed with one SAP Duet Server. In this case the configuration is propagated by the SAP Duet server to the connected SAP Duet Add-ons. Depending on the customer needs one or multiple SAP Business Suite systems can be connected to one SAP Duet Add-on. It s also possible to deploy multiple SAP Duet Add-on systems connected to SAP Business Suite systems. Reasons to deploy multiple SAP Duet servers can be the geographical or functional distribution of systems at the customer side. Minimal setup is one SAP Duet Add-on in the entire landscape.

Since SAP Duet Add-on is based on SAP NetWeaver 7.10 it will be necessary in most cases to install a separate SAP NetWeaver 7.10 instance as the SAP Business Suite is on running on SAP NetWeaver 7.0. Please see also the comment made in chapter 2.5. Since SAP Duet server contains components that are packaged as one Java deployable it could be deployed on any 7.10 Java system already existing in a customer landscape.

2.6.2 Duet Content Deployment

Duet content is delivered via SAP standard deliveries and managed in the SAP Duet Server. Customization is done in SAP Duet Server and according to the customizing the meta data (including UI definition) is deployed to the Microsoft Duet parts within a Duet landscape. The SAP Duet Server controls the version of the deployed meta data and ensures that it is in sync with the version of the referenced DOE objects.

The following diagram describes Duet content distribution between the different components (remark: all runtime connections are removed, only deployment connections shown):

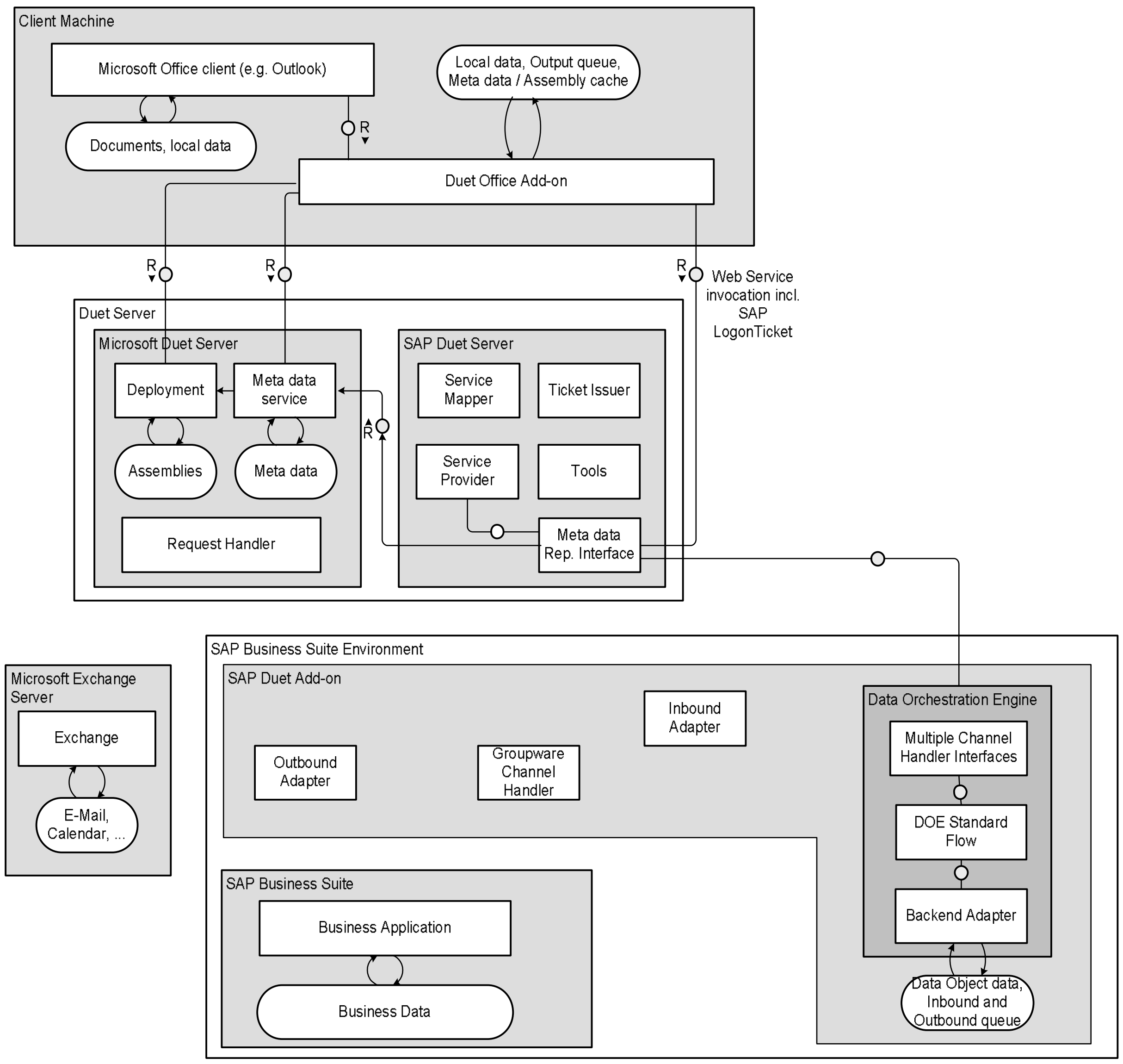


Figure 2-7: Duet content deployment

2.7 Architecture Documentation

Details explanation on Duet Architecture can be found at: Duet - Microsoft Planning Guide

For further details and information see: Corporate Portal -> PTU Emerging Solutions -> Products -> Duet

DOE related information at:

- KM documentation for DOE
- Building a sample DOE project
- Presentations and demo recordings
- More presentations and demo recordings

Additional information can be found at www.duet.com.

3 Open Issues, Outlook and Risks

Open Issues and Outlook

At this point in time there is considerable uncertainty about time lines and final release dates. Given the very latest developments and interactions with Duet partner Microsoft it is even unlikely that the architecture outlined in this document will materialize for a Duet 2.0 release. A more likely possibility seems to be that the next major release will be based already on Microsoft's LOBi framework. That would mean that a completely new client architecture would replace the current IBF-based client components. It would also mean that all data synchronization would no more go through Exchange server but rather through a dedicated data synchronization middleware. We intend to position DOE as a data synchronization middleware that would be capable to communicate directly with LOBi data synch clients. The same basic DOE mechanism (channel handler) would be used to implement the LOBi synch protocol. One underlying assumption is that all other components than the protocol-dependend channel handler would remain more or less intact. Even though this is a highly political topic with Microsoft we did get some indications from their side that they would be willing to support such an approach.

4 Abbreviations

BI	SAP NetWeaver Business Intelligence
CBS	Central build server
CUD	Create, update, delete
DOE	SAP Data Orchestration Engine
DSL	Domain Specific Language
DTR	Design Time Repository
DuetML	Duet Markup Language; markup language based on early versions of XAML to describe UI elements in the Duet context
IBF	Information Bridge Framework; predecessor of what is now called Duet platform (Microsoft part). IBF started as a framework to develop business extensions directly embedded into Office applications in a largely meta data-driven way.
LOBi	Line of Business integration; upcoming framework from Microsoft to develop Office Business Applications based on Office 14
MBA	Mobile Business Applications
OBA	Office Business Application – term introduced by Microsoft to refer to Office-embedded business applications (like Duet)

SCA	Software component archive
SMS	System Management Server
TCD	Total cost of development
TCO	Total cost of ownership
WebDAV	Web-based Distributed Authoring and Versioning
WPF	Windows Presentation Foundation
XAML	Extensible Application Markup Language; The new UI mark up language introduced with .Net 3.0 (WPF)
XLF	XML Localization Interchange File Format (XLIFF) format, in where the file has extension .xlf
XML	Extensible Markup Language