Acta Cybernetica **22** (2015) 183–210.

# Workflow processing using ERP Objects

Attila Selmeci*, Tamás Orosz†, and István Orosz‡

**Abstract**

Enterprise Resources Planning (ERP) Systems, and Workflow Management Systems (WFMS) evolved parallel in the past. The main business drivers for automation came from the ERP world, but the WFMS solutions discovered their own way figuring out the necessity of such applications without ERP as well. In our paper we follow only the usage of workflows in ERP systems. The central elements of built-in workflows are the ERP objects, which embed and handle the business data providing real life meaning of business objects as well. The capabilities of built-in workflow systems of such ERP solutions are presented via two market-leaders: SAP and Microsoft Dynamics AX. Both solutions are dealing with ERP objects in sence of the workflow management. We recognized and present the weaknesses and restrictions of the built-in workflow systems on these two ERP examples. In our paper we describe the results of our analysis of the interoperability of the built-in workflow systems as well and demonstrate the required add-on functionalities to provide usable cross-system workflows in such an environment. We also mention the possibility of using a built-in workflow as a full-featured WFMS.

**Keywords:** Workflow, ERP, BAPI, Business Evolution, Business Process Management (BPM)

## 1 Introduction

Today's World would stop, if ERP systems and other business application would not provide functionalities to manage and report data. In business application systems many business processes are running together, parallel and/or connecting to each other. These processes should be well organized to be able to manage the work at companies to achieve the goals. One tool for better management is the automation.

Workflow is usually regarded as "the computerized facilitation or automation of a business process, in whole or in par" according to the definition of the Workflow Management Coalition (WfMC), a nonprofit, international organization [2]

*AREK, Székesfehérvár, Hungary, E-mail: selmeci.attila@amk.uni.obuda.hu
†AREK, Székesfehérvár, Hungary, E-mail: orosz.tamas@amk.uni.obuda.hu
‡AREK, Székesfehérvár, Hungary, E-mail: istvan.orosz@sznet.hu

DOI: 10.14232/actacyb.22.1.2015.12

Workflow is automation of business processes, work steps in a certain sequence, where information, tasks or even documents are moving among the participants, who execute different actions on the objects following rules.

We acquired some experiences in various areas of different business application systems. We focused on the various approaches and capabilities of business solutions to understand and present the different level of interoperability of workflow environments. To see the steps and requirements leading to workflow environments we should first of all describe some basic terms using examples.

- A **business process** is built from a set of activities, which together realize a specific business goal, like general ledger accounting or procure to pay. The granularity of business processes is not strictly defined. With this varying granularity higher-level processes could be built as well.
- **Activities** can be called steps if we automate the process flow. As we see a Workflow is not only the process flow, but it contains the corresponding data, information, which are coupled to the process itself, and of course the actions, or execution steps which are the milestones of a Workflow. Technically an activity is a logical step of the whole process; we can think of it as an atomic element. These activities are not Workflow dependent tasks, so they can be executed manually as well, but in a process flow they are the main building elements, where the information or data is checked, displayed, updated or even deleted. An activity in a Workflow can be executed manually, if it requires human participant, like decision on acceptance or rejection. The other kind of activities requires machine resources, like sending a mail. These are executed in background without requiring dialog (or human) activities.
- The **participant** of a Workflow has his well-defined role in the process flow: when and what he should do.

The process chain is practically very rarely a simple sequence of tasks, but it contains at least decisions, as control points generate branches in the Workflow. The Workflow is only a theoretical process chain and a participant assignment with control steps (like sequential execution, choice, iteration or parallel execution). The technology enabler, which animates the workflow model to be a running, real process flow, is the Workflow Management System (WFMS). It does not only execute the process chains using software workflow engines, but it contains the possibility of plan, model, create and manage them as well. The WFMS solutions generally have graphical design/modeling tools as well. The workflow engine makes dialogue with participants and also contacts with application- and technology-elements. Practically we can distinguish five interfaces in a WFMS[12]:

- process definition tool (mainly graphical designer, modeller)
- invoked applications (application, which offer real business process steps to be executed)
- client application (surface for human activities)

- administering and monitoring tools (collecting technical, performance information on process flows, errors, etc.) and
- connection to external workflow engines (interoperability with other workflow systems)

There are standalone WFMS and built in ERP solutions also available.[13] In this paper we show the parallel evolution of business applications and automation, uncovering the point where the ERP systems are currently in the implementation of the Workflow. As the next step we describe, why we need communication and cross system workflows, and what are the enablers on philosophic and technology levels as well. In this topic we are interested in the invoked applications and connection to external workflow engines, as previously mentioned interfaces of the WFMS solutions. We introduce two built-in Workflow systems from the ERP arena (SAP and Microsoft Dynamics AX), where we show the different interface capabilities and possibilities. We have made our inquiry on cross-system workflow capabilities, where the process steps are executed in different systems. The interoperability of Microsoft's client products (like Outlook or Excel) with SAP is out of this process chain scope, because the steps are executed in this case only within SAP (not cross systems). There are several protocols, languages defined, introduced for business process communication or modelling like BPEL (Business Process Execution Language), Business Process Model and Notation (BPMN), etc. We explain the reason, why we do not concentrate on these opportunities in this paper. In the different sections and conclusion we present the several roles of objects, object orientation in business workflow environments.

## 2 Workflow evolution

As we have seen in the introduction the Workflow is mainly the automation of processes. Before going on we have to understand the steps from the single, manual task, through the Workflow, to the Business Process Management and beyond. Figure 1 illustrates the main drivers, why companies use Workflows in their environments:

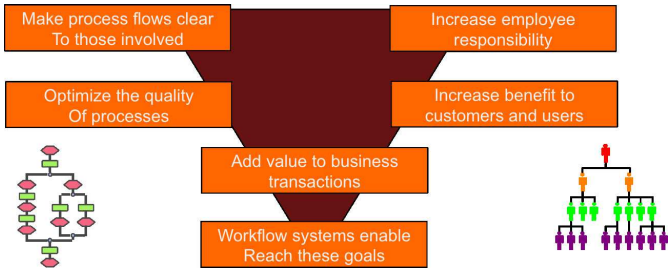


Figure 1: Workflow general concept

As we see there are more advantages for the employees and for the managers as well. The employees could easier and faster access the information in time and in place, less administrative tasks are involved, because the Workflow system should provide them in the background, and the processes could be simpler. In the same time the managers have better control over information, deadlines (essentially over the service levels), and with all these over the overall costs of the processes. Beyond the mainly cost and speed effectiveness the managers win flexibility in process changes, in organizational structure reconstruction and in some cases in the technology modifications. All these lead to more effective company level management, which is the main driver of automation.[1]

Some bigger stages of the evolution are described below:

- Workflow like thinking (1990):

  Standard monolith architecture runs the business tasks initiated by the users. The processes are not written to follow each other; only the executor knows what the next step should be. The workflow runs in the head of the end-user, he co-ordinates the process flow.

- Workflow as task sequence (1993-94):

  Similar to the previous stage, but the definition of process is written as a model and the tasks are executed according to the model as a sequence of tasks. No task management is implemented. The end-users are starting the application transactions or tasks directly in the system using the standard user interfaces.

  This level of workflow gives the thinking on arranging tasks to processes, defining process flows and even determining similarities between processes, or having same task, step in different processes. We can faintly recognize the reusability and building blocks.

- Task distribution within workflow(1996):

  This can be really called workflow, because the process definition contains not only the task sequence, but according to the individual steps different participants can execute the tasks. The work is divvied between the participants according to the business process requirements, like employee and manager, or requestor, approver, and purchasing expert. The participants should log into the systems (containing the data and the procedures) to execute the step, but the workflow system helps to launch the transactions, programs (with corresponding parameters) to be executed using the standard system user interfaces.

  This level allows separating activities within the company, not really thinking about the process itself but the element of the different processes. Different employees with different rights and expertise act as different participants in different processes. The processes can be fastened, e.g. the production of a good can be quicker and is changes the market. The processes cannot be

accelerated beyond a certain point without external information and input, but this leads us to a later stage.

- Service oriented workflow (2002):

  In the previous stage we have seen that the different tasks, steps can be clearly determined and isolated from each other. These tasks are faceless procedures with signatures acting as offered services from the backend systems. The services are called from the central process engine, which takes over the task of the previously used workflow systems and contains surface for user interfaces. The workflow running in the process engine is built on services, or better say to it is service oriented, and it does not depend on the different backend systems, since the services do not tell anything about the backend system, only the interface is available. Another interesting point is that the participant uses only the process engine, only the decisions are made there, but they are not executing backend tasks directly. The process engine executes all these tasks via the services.

  To handle such level of workflow there should be communication between the available backend systems. For such purposes EAI (Enterprise Application Integration) layer serves as the central component offering standard interfaces to the different technologies of the backend systems. This layer manages, monitors the data exchange in a higher, more optimal level by offering validation, conversion, filtering, multiple distribution, etc.

  A Service Oriented Architecture should offer among others a service repository, containing internal and even external (public) services (so-called Web Services) in practical cases using the standard UDDI (Universal Description, Discovery and Integration), WSDL (Web Services Description Language), and SOAP (Simple Object Access Protocol) protocols. The heart of the SOA environment is the process engine or central workflow system delivering not only the workflow system components, but user interface development tools as well.

- Workflows using composite applications and services (2005):

  The previous SOA based workflow offered UI design and process flow executing steps from different backend systems provided as services. SOA can be used for composite aplications as basis. A composite application can be called a cross system application as well, because the application has only a user interface and view control defined centrally, but the functions or in other word models according to the M-V-C (Model-View-Control) paradigm are created from service calls.

  The workflows using composite services step forward to define composite (cross system) services before offering them to the process engine layer. This concept brings a new layer into the middle between the backend systems offering simple services and the process engine having user interface capabilities. This layer is generally called Enterprise Service Bus (ESB) having an

enhanced and extended EAI engine. The ESB takes over the management of the service repository, because beyond the simple, backend offered or even externally published services, it defines new services built from the available services. These are called composite services. The ESB should be reliable and robust, because the outage of any simple service used in a composite service should be correctly handled to the process engine layer.

- Human driven workflows (2008):

  This is an intermediate step to the business process management solution. Here the participant actions are extended with preparation and execution after the human work. These extensions make possible to handle changes in organization or business level easier. The workflow environment and the company can react to the (market or business) changes if there is practicably no modification requirement on the computational logic, in the data model, and in deep level IT infrastructure (hardware and software). The business experts can easily change and redesign the processes if they have responsibility on the roles, assignments, service level measurements (like deadlines, escalations, reminders), user interface design and rights to add, remove manual steps. This separation can be achieved by dividing the whole solution between the ESB and process engine. The process engine contains all the activities, which belong to business relevance.

- Cloud computing workflows (2010):

  The last separation in the previous stage offers the physical separation of tasks and enablers, because the process engine with all its capabilities and responsibilities can build a separated environment, even cloud, connecting to the IT driven backend systems (internal end external) offering services (internal and external) and composite services. From the SOA level the real execution and model is not relevant for the business anymore, they think about the services only, not the storage, hardware, software solution. From their point of view changes on the business processes can be made without any programming, just redesigning is necessary. This leads to business level agility.

Why do we need workflow solution in an ERP system? To answer this question we just have to look above or into the history of the business application or the business requirements. The market wants always better, but cheaper goods, the producer should always optimize the processes with various techniques like reducing the number of employees, speed up steps inside processes using better control on execution, less waste by having higher quality control, etc. The business requirements from the companies around 2000 required the already existing paper-less document management, and the automated process control in the systems. At this point many software vendors already had solutions for business automation, but the concepts were not ready for everything, even if some of there were they could have been applied. Many trials for universal workflow system failed, so the ERP

producers decided to create their own implementation optimized to the solution. In the following two chapters we introduce two big software solutions, where the internal workflow engine functions according to the ERP workflow requirements.[6]

Table 1: Workflow evolution steps

| **Evolution step** | **Year** | **Main facts** | **Type** |
|---|---|---|---|
| Workflow like thinking | 1990 | Monolith architecture; only thinks about WF | 1 |
| Workflow as task sequence | 1993-94 | Sequence of tasks is defined; End-user executes tasks following the model | 1 |
| Task distribution within workflow | 1996 | More participant, different tasks, WF manages; business logic is in the application program | 3 |
| Service oriented workflow | 2002 | Process engine has own UI; business logic in decoupled services | 2 |
| Workflows using composite applications and services | 2005 | Service repository; ESB; business logic in Web Services | 2 |
| Human driven workflows | 2008 | Human activities and process steps are contracted; human step is the main driver | 2 |
| Cloud computing workflows | 2010 | Responsibilities and developments (Business and IT) are separated; ESB provides access for human activities via cloud; services are offered in the cloud for ESB | 2 |

Table 1 summarizes the bigger steps of workflow evolution detailed above including the later defined application type information as well. Not only the workflow concept evolved, but the operating systems, hardware and other software solutions as well. The ERP solutions had a reconstruction following also the hardware / operating system evolution.[13] Starting with the ERP II era, the central ERP workflows should step out the boundaries of the system and continue the workflow in other systems. A good example for that is the supply relationship management (SRM) solution. If someone would like to purchase something the requisition and approval should take place in the SRM system, but the real order is made in the

ERP system, the delivery information comes back to the SRM, but the ERP should receive the invoice. In such a complicated process, where different steps are executed in different components, the cross system workflow concept comes into the requirement and reality. These internal, but cross system workflows combine the different components into one big business solution.

Another more adequate example can be found in the supply chain management area, where the company knows the repletion of the shelves of the dealer or deliverer. In both examples the company should collaborate with the partner, or in technical level the systems of the company communicates with the systems of the partners. These communications are not only data exchange, but they could be workflow steps as well. This leads to the cross-system workflow requirement.

The evolution presented the three main application types of business process technologies:[13]

1. workflows involving humans,
2. workflows involving systems and applications,
3. transactional workflows.

According to our experiences the type 1 workflows are the early automations, the type 2 workflows are the EAI, BPM and SOA workflows, and the type 3 ones are the ERP built in workflows. The next two chapters explain the technology, possible power, and weakness of two ERP based internal workflow systems. We show that they converge to the type 2 as well. The automation was introduced for better performing execution and less human activities, but nowadays the WFMS brings more flexible environment to handle the business changes drove from the market easier. We will outlook a bit to the type 2 workflow systems not mentioning the currently available "off-the-shelf" products, but pointing out the role of these solutions in the market today.

# 3 Possibilities and architectures of Microsoft AX

The Microsoft Dynamics AX ERP solution can be regarded as a role based ERP solution, based on frequent workflow processing. The whole system has an object oriented architecture out of the box, so everything is designed and defined as interacting objects, which are organized into a hierarchy, called Application Object Tree (AOT) [8]. This architecture helps integrated workflow processing. Main activities, such as Payroll, Personnel, Budgeting, Shipping, which are all supported by a customizable role tailored user interface, which needs complex workflows through the system. For example, a purchase manager can track all activities from requisition to approval, if there is a workflow change because of absence or vacation, workflow can be redirected automatically. The whole workflow processing is emerging into a central software solution, not just being a major part of its own ERP solution. The current development stage is moving toward this nowadays, but the workflow

will be more of a common solution for any environments which are based on Microsoft ecosystems. So the ERP will be only acustomer of the workflow and not the main host, as interacting with more producers of the workflow. After the definition of business processes in an enterprise, it will be necessary to use the workflow as ahorizontal general tool, which connects the whole software/hardware environment, outside the ERP system as well. As developers make an abstraction of the business processes, they should not concentrate only on a monolith ERP system, but should take into consideration the information technology architecture in an enterprise. In this respect, the very trendy cloud computing phrase is still just a simple tool, and not the target. By this new workflow approach, the used techniques, existing enterprise flow and SOA elements will be defined newly, and not just refactored. This approach can extend the longevity of the lifecycle of the model.

The idea behind Dynamics AX workflow 2012 R3 is to use the Windows Workflow Foundation, as a general unit solution [10]. Workflow Foundation provides a lot of capabilities which are used generally, where there is a need for workflow infrastructure. Being a low-level infrastructure basic element, Workflow Foundation has no direct access of or integration with Dynamics AX 2012. In the following figure 2 the workflow basic infrastructure is a so called abstraction layer which resides above workflow foundation and allows workflows that are specific to Dynamics AX to be designed, implemented, and configured in Dynamics AX 2012 and after then being executed by using the workflow foundation.

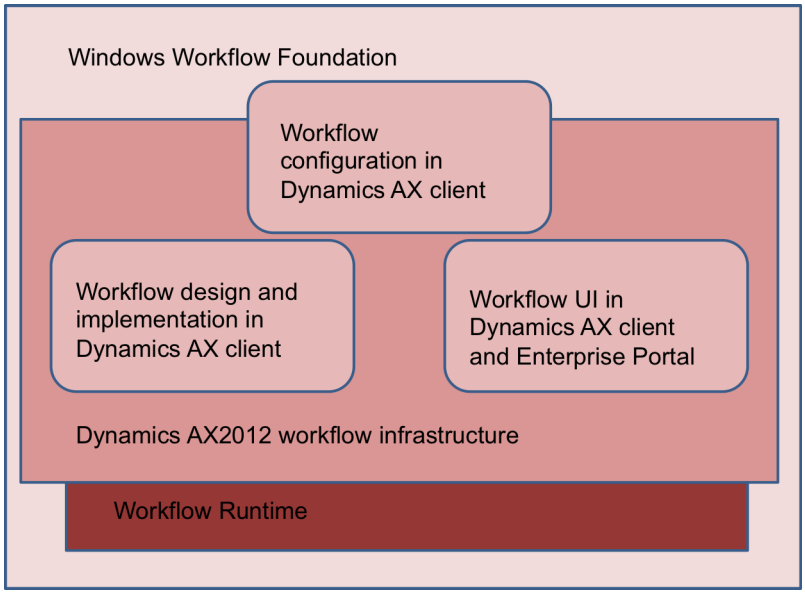


Figure 2: Windows Workflow architecture[9]

Developers are responsible for implementing the neccessary abstract workflow element to represent the business logic. Business process owners design the entire

workflow using a graphical workflow editor in the Dynamics AX 2012 client that is based on the WF Designer. The workflow runtime engine connects both the Microsoft Dynamics AX 2012 workflow infrastructure and the Workflow Foundation to each other; then instantiates and executes workflows. The system administrator has to maintain the runtime environments. [9]

The workflow document, which is sometimes referred to as the fundamental business document, is the root entry point for workflows in Dynamics AX 2012. Each workflow type and workflow element has to reference a workflow document because it accesses the data context for the workflow. A workflow document is an Application Object Tree (AOT) query supported by a class in the AOT (which is referred to as the workflow document class). The expression workflow document is used instead of query because it more accurately describes what the workflow is working on. The query used by a workflow document can be references to multiple data sources and is not restricted only to a single table. In AOT, a single query is able to reference data sources hierarchically. The workflow documents and workflow document classes are all located in the AOT in the Dynamics AX 2012 client. Workflows in the Dynamics AX 2012 use an expression builder the developer or the business process owner can use to make conditions that control the behavior of a workflow which is under execution. This expression builder uses the workflow document to get the fields which can be referenced in conditions. To make the result data set available within numerous conditions, developers or business process owners can add parm methods to the workflow document class, and later they can add specific X++ code to the parm methods to produce the derived data. After the workflow document can return those fields from the underlying query plus the data generated by the parm methods.

Workflow categories specify the association a workflow type has referring to a module. When a developer or a business process owner adds a new module to Microsoft Dynamics AX 2012, a new module and a new workflow category has to be defined, that references that module. Each workflow category is located in the AOT. The workflow type is the primary block that developers use to create workflows. The developer generates the workflow type by using the Workflow Wizard. This wizard automates the creation of the metadata required for a workflow type; what the developer needs to do is specifying the name, the workflow category, the query, and the menu items. Event handlers are the integration points that developers use to trigger business logic inside the workflow execution. Every workflow event is generated at the basic workflow level and the workflow element level. The elements of a workflow represent the activities within the workflow. The business process owner models these elements. An element can be a task, an approval, a subworkflow, a manual decision, an automated decision, a parallel activity with multiple branches, a line-item workflow, or an automated task. Developers implement these automated task elements, line-item workflow, and approval. The business process owners can use the other so-called "configuration only elements" in the graphical workflow editor. The following list describes each element:

1. Tasks are generic workflow elements that represent a single unit of work. The

developer defines the possible outcomes for each task.

2. Approvals are specialized tasks that allow sequencing of multiple steps and use a fixed set of outcomes.

3. Subworkflows are workflows that are invoked from other workflows.

4. Manual decisions enable the workflow to follow one of two possible paths based on an action taken by a user.

5. Automated decisions enable the workflow to follow one of two possible paths based on a condition.

6. Parallel activities contain two or more branches that represent discrete workflows, and they are executed simultaneously.

7. Line-item workflows are modeled within a workflow that exists for a business document that represents the master in a master-detail relationship. They enable specific workflows to be instantiated on line items that are associated with the master business document; for example, expense lines on an expense report.

8. Automated tasks are non-interactive and invoke X++ business logic synchronously. Manual and automated decisions, parallel activity, line-item workflows, and automated tasks are new in Microsoft Dynamics AX 2012. In addition workflow wizards have been added to make the creation of approval and task elements easier.

The main goal of workflows is the approval process of documents, where the document is one object and the triggering event is an object event. This object event is most likely a technical event, like starting a menu item, pressing a button, but not a higher-level business event. By checking the WFMS interfaces in this example the client is a Microsoft tool, e.g. Outlook, Excel or Dynamics AX client. The Windows Workflow Foundation calls the different business tasks from the AX, so it behaves as a WFMS, it takes the control and process flow without managing the data. As part of the operating system (Windows) it can be used for non-Microsoft applications as well to invoke any application task. As a process definition tool Microsoft uses Extensible Application Markup Language (XAML). The workflow definitions can be developed as a code, or as an XAML, or even as the mixture of them. By default the BPEL is not supported, though it was originally defined by Microsoft and IBM. [15] Microsoft offers an external library to be able to use BPEL in workflow design, but the BPEL will be converted to the internal representation and not used as it is.

There is an opportunity to use existing Microsoft technology for orchestrating workflows in Microsoft Dynamics AX 2012 instead of designing and implementing a specific functionality from scratch. In Microsoft Dynamics AX 2012, the WF framework was integrated into the AOS.

One more point about cross-system communications is missing. We have described that the WWF calls the Dynamics AX objects using the object model. If an external program would like to call these business functions, two possible ways are available: using the MS Dynamics AX business connector and writing C# code in .NET environment, or using Web Services. If we do not want to use Web Services, special wrapper modules should be created to be able to call (e.g. from Java) the modules. The Web Services offered by AX can be consumed in any language used in .NET environment. Dynamics AX call consume external Web Services as well. The following goals were achieved by this architecture:

- extensible, pluggable model for workflow integration
- scalability that accommodates the growth of workflow usage in Dynamics AX 2012 over time and provides options for scale up and scale out
- minimize the possible performance loss on transactional business logic when invoke workflows

# 4 Possibilities and architectures of SAP ERP

The Microsoft AX is younger software than the SAP, so we have to show a bit more from this always-renewing software solution. Many of the current SAP standard table structures and codes are coming from the mainframe era, where the first working early ERP system was the SAP R/2. Some parts of the basic business logic did not change too much in the last 20 years, so the todays applications can still rely on some old mechanism. The working architecture of SAP ERP have some layers above the operating system and database management system levels, which serve to manage the business functionalities and platform independency. The data model of this architecture is defined within the SAP system having no strict linkage to the underlying database level. A so-called database interface manages the mapping and SQL-translating between the SAP Open SQL and the underlying native SQL. There is no data model defined, used in the native database. The SAP data model stores much application specific, semantic information as well to help the application run smoother. The functional model has internally more layers, where the lowest layer contains the programs, function modules screens and transactions. The function modules are special, reusable development elements, because these are designed for specific functions like booking an item, or get employee data. Technically the function modules are organized into function groups, which program technically acts as an object oriented class having only static (class level) methods the grouped function modules which share the attributes defined in the class. Because of this behavior the function of a group can collect, manage the data they require during an active program session. Some of the function modules contain screens as well, but most of them are faceless execution codes. The programs and transactions are built upon the function modules, because the central functions are written using these forms to be able call them from more programs as well. Some oft

he function modules are enabled for remote invocation as well, implementing the remote function call (RFC) capability (the SAP variant oft he remote process call). The RFC enabled functions can be called from outside of the SAP system enabling application level linkage to other solutions.[5] The continuously renewed technical layer (called NetWeaver Application Platform) of the SAP systems makes possible to handle Web Services as producer or consumer as well. Each of these RFC enabled function modules can be called using the Web Service protocols (like WSDL, SOAP, etc.). The function groups as building blocks compose the middle internal layer of SAP applications. Almost each program, transaction calls functions to execute specific, standard tasks using them as system offered services. In the last 10-20 years SAP implemented the object orientation into the technology layer. Beyond the fully OOP (Object Oriented Programming) enabled classes we can identify many embedding classes for wrapping the function module logics. Figure 3 explains the RFCs on the right side and the object oriented wrappers described below.

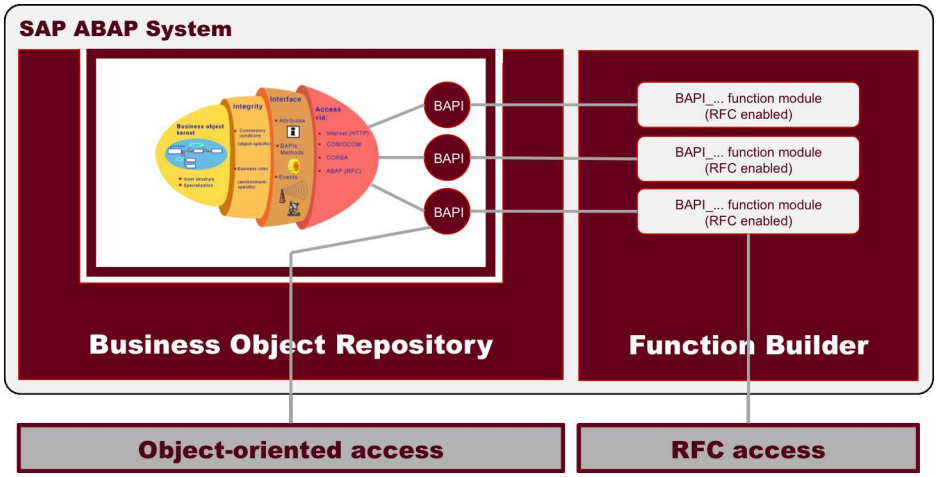


Figure 3: OO and RFC access of business data

SAP implemented the first Workflow environment in the 90's, but it was redesigned many times in the past. Before implementing the object orientation into the technology layer SAP recognized the necessity and usability of objects. To realize object management in the system SAP implemented a new layer above the business applications called Business Framework Architecture (BFA). In this initiative SAP introduced some new objectives and terminology, like business component, business objects and BAPIs. Business Objects are the most important elements of the BFA, because these set the object orientation into motion. Examples for business objects are an employee, invoice, or purchase order. Different relations between Business Objects can be defined, e.g. has-a, part-of, kind-of, is-a relations. As an example we can imagine that a purchase order should have a customer (association, has-a) and some items (composition, part-of). Each item contains a

material (association, has-a), but the stock material is a material (inheritance), etc. Real inheritance is not implemented, but the system uses this term and technology is also defined to manage inheritances. Some interfaces can be defined and used as well. SAP defined the name object type for class and Business Object for an instance of the class. The following element description is interpreted in figure 4 as well.

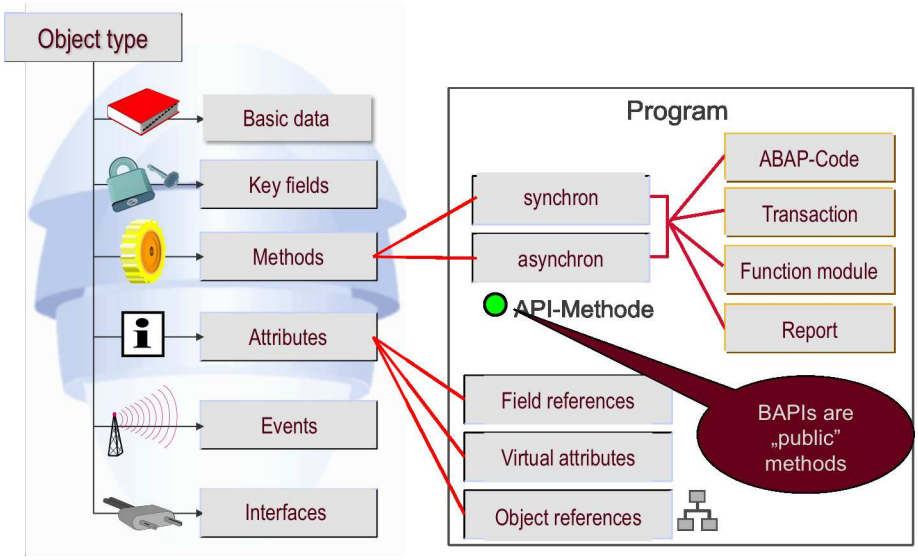


Figure 4: SAP BusinessObject Type and its elements

Each object type can have attributes, methods and events. Some of the attributes build the identification of the object. These compose the key of the object, like the material number for the object material. There is no real public-private distinction, but some oft he methods are public in a different sense. SAP internally can apply each of the objects with their attributes and methods. The code should contain the special Business Object libraries (includes) to be able to access, call, and execute the objects and their elements. It is because these techniques come from not object-oriented era, but with an object-oriented thinking. There are some special methods of the Business Object, which are the so-called BAPI (Business Application Programming Interface) methods. These methods are RFC enabled functions, so they can be called from outside of SAP. These are more possible invocation techniques available. The simplest is calling the method, as a standard RFC function using RFC libraries on the caller side (if it is not an SAP system). Using Web Service mechanism in the new systems, where the SAP offers the RFC enabled functions, so the BAPIs as well for calling as Web Services. The high level technique is object oriented, where the caller environment (like C#, Java, etc.) builds its own objects and calls the methods of the objects. In this case the externally referred object offers only the BAPI methods of the Business Object to execute.[11]

The Business Objects are the main building blocks of the SAP Business Workflows. Each step of a workflow is a method of a Business Object and we can start a method by waiting for a specific event of a Business Object (type). An SAP Workflow is a so-called multi step task, where each step can be a single step task (referring to a method of a Business Object) or an embedded multi step task as well. According to the mentioned workflow definition, in chapter I, the single tasks can be executed by a human being or by the workflow engine. So the task can be dialog or background. We have to determine for each task, who is allowed to perform it. SAP has a bit complicated decision on this using distinguished agent sets. From program execution point of view the agent should have right to execute a task. It is better to assign the task to a user group who are able to execute it. This assignment can be done according to an authorization role. During the workflow run the engine can evaluate who are allowed to execute the task. For example the manager of the requestor is a responsible agent. There could be more responsible agents as well (e.g. groups of book keeping clerks). The intersection of the possible and responsible agents is the group of users who receive the Work Item (the task or single step during runtime) in their workflow inbox. This group of users is called recipients. The possible agents are static information attached to the task, but the responsible agent list can be determined during runtime by so called rules. Rules are defined separately providing agent list using HR (Human Resource) hierarchies, or any programmed code to find agents according to parameters. This manner of rules makes them reusable in the whole workflow environment as well.[4]

In a workflow definition many opportunities are available to define the process flow. Firsts of all the workflow during runtime has a global status containing the global variables, information used in the tasks or even roles. For easier, but standard usage of data handling SAP provide various containers. A container holds and handles simple field variables, but structures, arrays or even Business Object can be stored and manipulated during runtime. The previously mention global storage area is called workflow container, which lifetime is equal with the workflow lifetime. The following containers have also roles during runtime:

- Rule container: Used to define the possible agents for a step to be executed. It reads data (parameters) from the Workflow container. Lifetime is only till the task is called.
- Work Item container: the single step receives input data from the Workflow container and puts back the output. It handles additional data as well, e.g. information displayed to the agent. These field values can be used in the short and long text of Work Item displayed in Workflow inbox of the recipients. Lifetime is during task call.
- Methods container: It communicates only with the Work Item container to receive the data required for the method call and sends back the results to the Work Item container.
- Event container: it is used to fill the event parameters. It is used before triggering a Workflow even from outside the Workflow system (from SAP

programs or out of the SAP using RFCs or Web Services). Lifetime of the event container is started before the event is created and ends when the data is loaded to the other (e.g. Workflow container).

Figure 5 describes the above defined container usage during runtime.

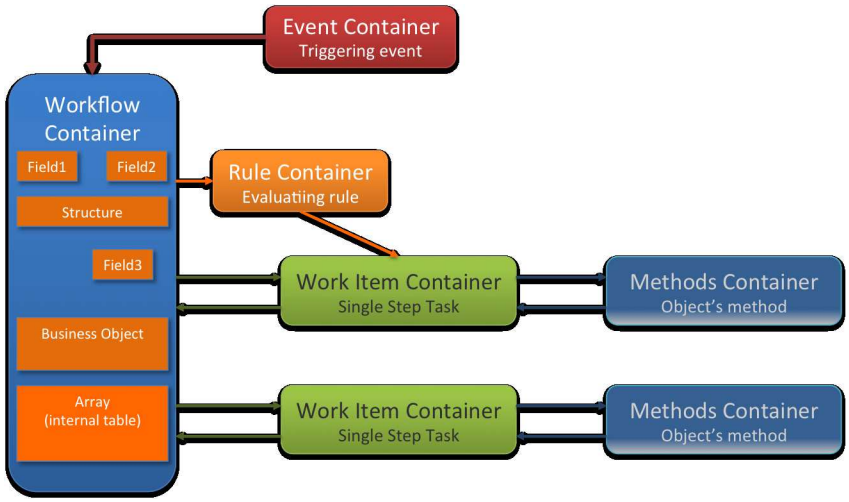


Figure 5: Containers in SAP Workflow

This container technique is very useful, because any data can be provided and stored without special mechanisms. In a newer, modern system these would be handled by using XML files, but here the SAP ABAP environment provides many features to store the content internally without converting the data into XML format. The meta-data descriptions are coming from the ABAP Dictionary, or Business Object definition. In the newer releases real object oriented classes and their methods can be used as single step tasks. This bring the solution to a modern form, but we have to rely on the old fashion Business Object as well, because the business data and function are embedded into those logical objects and object types, and only the new functionalities are covered by OOP classes.

Before going on we have to understand that the SAP Workflows use many different step types to build the Workflow template (design time object). There are different groups of step type we can distinguish. In the following list we shortly collect them:

- Executive steps: Activity (standard dialog or background task), Send mail (delivered standard task for mail sending, the text and receivers should be defined), Sub workflow (a multi step task, it starts another Workflow in place), Ad hoc anchor (run time defined sub Workflow), Web Activity (for external bidirectional communication using http based SOAP messaging).

- Control steps: Loop (UNTIL and WHILE loops), Condition (according to a condition on a container filed value it drives to the true or false branches), Multiple condition (according to container value a decision is made by the engine and runs the workflow forward on one of the defined threads), User decision (similar to the Multiple choice, but here the user decides, from the different options listed in the task), Fork (parallel branches, start and end points),
- Special steps: Container operation (the value of a container element can be changed), Event creator and Wait for event (starting and waiting for events, with these step types another workflow can be triggered, or the workflow can be suspended for a while till a specific event is raised), Process control (this step jumps out from the process flow and the workflow can be finished immediately; works like an exit command)

Each of the step types is customizable, but not changeable, except the Activity step type. (Technically they can be changed, but in practice there is no need to modify the method assignment, etc.) This is the step type, which can be used to execute our own tasks. As we described earlier a task runs always a method. The method is generally an element of a Business Object, but in the newer releases an OOP class method can be used as well. Various types of methods can be defined within a Business Object: Function Module, Transaction call, BAPI, RFC to call outside from the system, or manually developed local code. This last one gives high freedom to the programmer, but the code is not reusable in other places within the system. As you can see in a method we mainly use existing, already implemented functionalities. As we described earlier the BAPI-s are special function modules, so the calling mechanism similar to the function modules or to the RFCs. If we use transactions (but sometimes in case of using function modules as well) we can have not only faceless modules, but also standard SAP dialog transactions having screens (dynpros). As in the real OOP world methods can be instance dependent or static as well.

The methods having own screens are not really usable for external usage, because only the data should be transferred. For this purpose another method programming type is also available, the so-called BDC (Batch Data Communication) or Batch Input method. This executes the whole dialog transaction with all data validations in background according to a parameter and button pushing description, like a macro. If real interoperation is needed between systems or workflow engines, this kind of technique slows down the solution and could lead to unwanted process stops. If we want to use cross system workflows the Web Activity step type should be used. The interoperability is detailed in the next chapter.

As a whole the SAP Business Workflow is a very useful processing engine, which executes mainly internal Workflows. It is really business driven, and the main building blocks are business functionalities, which are high level enough to build the process flow. The whole solution is defined according to the object-oriented requirements, but only the new releases can implement real class methods in the workflow steps. The authorization and responsibility to execute a task are

separated, but in a well-configured environment they complete each other to have the right persons to be able to execute the task. SAP does not support directly BPEL models, if you want to use that Process Integration (PI), EAI like tool should be implemented as well. On the other hand SAP workflow offers interconnect capabilities in workflow, detailed in next chapter.

# 5 Cross-system workflows and Business Process Management

As we see in the practice and according to our research the business nowadays needs at least cross-system workflows or even Business Process Management solutions. Before going into detail we have to see a bit what the two example ERP solutions offer in that area. According to the above-described features, in an SAP system during process run any method (of a task) can call RFCs (e.g. to retrieve some information from another SAP system) or even Web Services (using SOAP protocol and internally the methods of proxy classes) to communicate with non-SAP systems. With these mechanisms we can call functions, methods of other system, which execute something (e.g. query or store some data). Depending on the connected system we can even start an external workflow, if the system offers such remotely callable services. The real way to start a workflow remotely or to do something to a request coming from a remote system is the usage of Web Activity step type tasks. These tasks should be based on a standard Business Object, called XML_DOC. Figure 6 shows the logical process flow using the xml based communication. The communication interface inside the SOAP or XML messaging is the Wf-XML. This is an open standard and some Workflow engines implemented it already. Four main methods can be used: starting a remote workflow or allowing a remote system to start a local workflow, and both can be executed synchronously or asynchronously as well. These enable the any system or application, which implements the Wf-XML interface to start a remote workflow without any special additional component.

On the other hand the Microsoft Dynamics AX offers direct events to start workflow via the standard Windows Workflow Foundation (WWF). According to our study we figured out that Microsoft Workflow solution and the Dynamics AX ERP system architecture is more technical, than the SAP solution. Deeper, simpler and close to technology options are available. As we described above the events are low level, basic elements, which trigger basic logics in the Dynamics AX system. These can start internal, simple workflows, but with some programming, complex flow controls could be set up as well. There are no predefined protocols or techniques to call Workflows in remote systems or consume Workflow invocations from other systems. But the architecture of the Microsoft ERP system and the used workflow solution offers many low level possibilities to communicate with external workflows. First of all we have to understand that the Dynamics AX implemented originally a simple workflow engine, but afterwards pushed the task to the WWF built into the Windows server running the ERP system. This leads us to two recognitions:

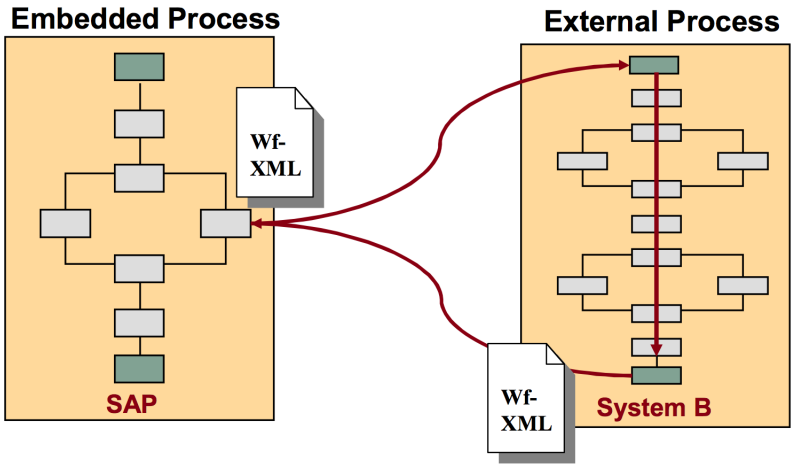


Figure 6: SAP XML_DOC usage

- The WWF handles the flow management, scheduling, background or dialog task calls, etc.
- The Dynamics AX offers only services, functions, methods, etc. to call, without controlling the flow.

These drive us to understand that WWF can easily, using standard behaviors, call external methods (external C# or VisualBasic methods) or even Web Services (proxy classes should be defined according to WSDL (Web Service Definition Language) information), and receive external calls as well to start or continue a workflow. The communication schema can be defined as in Figure 7.

If we want to handle the workflow interoperability in ERP level, we have to use other methods or develop. There are multinational companies, which evolved by merging smaller companies one by one. It is essential to develop a cross ERP solution for example between SAP and Dynamics AX.

The solution requires a middleware layer, which bridges data between SAP and the Application Integration Framework (AIF) within Microsoft Dynamics AX 2012, in this solution the middleware layer is the BizTalk Server. BizTalk Server uses its built in native Microsoft Message Queuing (MSMQ) and SAP Windows Communication Foundation (WCF), which provides reliable, high performance two-way data transactions between the two solution tiers. Microsoft Dynamics AX 2012 includes a development structure called AIF, which enables integration and duplex communication between business processes; also it supports connectivity to a wide variety of external data sources.

AIF also contains a wide range of services that allow the rapid development of customized user interfaces. For every built-in services, the standard Create, Read, Update, and Delete (CRUD) operations are available. Most of the services that are

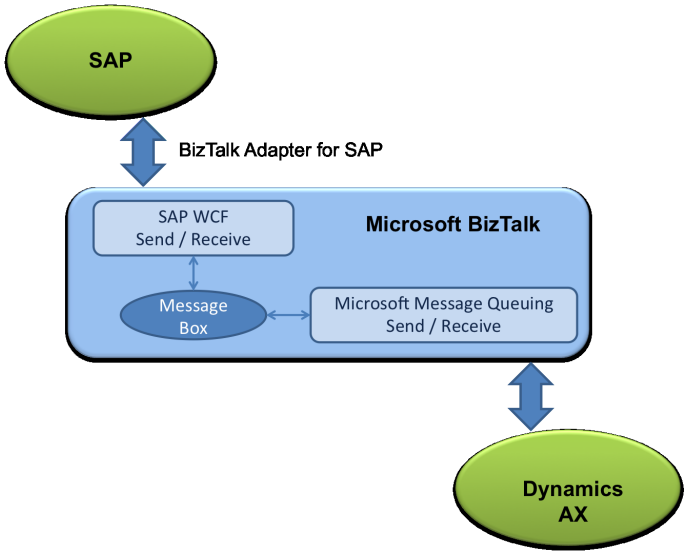


Figure 7: BizTalk connection schema

prepackaged in Microsoft Dynamics AX can be used as-is. Extensible Stylesheet Language Transformation (XSLT) retains all transformations and mappings within Microsoft Dynamics AX, which makes data transactions to be more easily deployed. When the incoming messages arrive from SAP, they are routed again using the Microsoft Dynamics AX 2012 format according to the mappings stored in Dynamics AX.

Data mapping uses in reusable SAP Intermediate Document (IDOC) messages. This allows for reusability of the same data by multiple applications, providing scalability for future development. In addition, with using the SAP WCF adapter that BizTalk Server provides, data requests directly in SAP with using a so-called queue table, that allows the SAP side to manage asynchronous connections internally rather than continuing to flow data back through BizTalk every time a message is put into the queue. As being part of the Microsoft Dynamics AX architecture, the code, which is written inside Microsoft Dynamics AX is always .NET compatible. Not only SAP application level loosely coupled connection (ALE Application Link Enabling) and data sources can be used, but direct RFC or embedded Web Service calls as well. The RFC calls do not have standard message based surfaces, but the Windows Communication Foundation translates the content to the internal form of BizTalk.

The IT solutions for workflow engines are strongly different in these two examples ERP system, but we can feel that each of the solutions tries to focus on the business requirements of the new world. As we discussed in the first two chapters the evolution of workflow is not finished at the internal ERP workflows solutions, but cross-system and external workflow engines are required. According to our real

World experiences the companies omitted the SOA (Service Oriented Architecture) way and immediately started with usage of Enterprise Service Bus (ESB) solutions. It means they implement a high level EAI (Enterprise Application Integration) solution for the data exchange between the many different systems. The EAI offers solution friendly adapters/connectors for the connected components using their own protocols, and reduce the communication using internal storage (keeping data content for multiple use or till next update), filtering (not all data, but only some records or columns are transferred), and conversions (e.g. field concatenation, trimming, splitting are necessary for the different receiver components).[14]

The table 2 summarizes the capabilities of the selected and observed ERP systems in the cross-system workflow arena we can see the following communication options in high level.

Table 2: Inter-communication

| **Connection** | **Experience** |
|---|---|
| AX - AX | In this case we are not talking about the underlying WWF layer and its features, but only the Dynamics AX level. Some options are available for communication between the remote workflows: using .NET Business Connector (enabling to access any object, like tables, classes, forms, etc.), using Web Services (with some development), or even low level server-to-server communication (not recommended). |
| SAP - SAP | Between SAP systems the well known RFC (Remote Function Call) technology and protocol is the easiest way for handling remote workflows. Fortunately many of the workflow functions (SWF_* functions) are remote enabled, so they can be called from other systems as well. To raise an event to start a workflow the SWF_CREATE_EVENT remote enabled function can be used. (The Business Object Type definition should exist on both sites to be able to raise the event of that Business object.) On the other hand, as each RFC enabled functions, these can be called as web service as well. Finally the Wf-XML method can be used between SAP systems as well. |
| SAP - AX | No direct connection exists, but(see below) |

Inter-communication possible when onw of the following is used:

1. Developed Middleware application using .NET Business Connector and SAP .NET Connector (enabling RFC connection) can be a simple project based solution

2. Web Service invocation from both sites with programming. On the Microsoft Dynamics AX site it is advisable to choose the WWF layer than the ERP

itself. (Because in ERP system more development efforts are needed to enable the workflow, which is controlled anyway on the Windows Workflow Foundation layer.)

3. Using EAI (Enterprise Application Integration) solution, like BizTalk to enable the standard interface usage. This was exactly described above.

The EAI is extended or enhanced, because other integration level can take higher roles as well. Not only the data exchange, data integration is important, but the information integration as well. In technical language we can say that not only the data is managed, but the functions as well. With these capabilities we are speaking not system level data and functional model (using internal integrated data and implemented function collection), but solution or even company level data and functional models. The data exchange is done via the EAI solution, but the function, services offered are collected into service repositories connected to the Enterprise Service Bus. From this bus any component system can read the relevant data or call the required (external) service (function). This level is offered by SAP ERP and Microsoft Dynamics AX as well, but each can interconnect with a third party ESB environment as well. It is important from our point of view, because with this offering we can use any of these ERP systems as external Workflow engines using their own engines, but executing as tasks others (via ESB) offered services. Here we have to make difference between the two ERP solutions. SAP offers an application level object oriented environment via Business Objects. Most of the important methods of the Business Objects can be called externally as service (or even Web Service) also. SAP offers many RFC enabled functions to be called as Web Service as well. These can be collected as single services in the service repository. SAP recognized early in the Web Service era, that these functions are too deep, to simple to build an application from them. It drove SAP to introduce the Enterprise Services, which are higher-level services with full definition, documentation executing a specific business task. Microsoft Dynamics AX, as we described pushes out every workflow engine capability from the ERP system to the existing and functioning Windows Workflow Foundation. With this step Dynamics AX realize the usage of an external workflow engine as standard behavior. To make the solution longer usable the engine and the workflow steps should be separated and the workflow engine should call the tasks as external methods any time. Dynamics AX offers only services to be called from the WWF engine. With this capability, though the ERP has no more its own internal workflow engine, but reaches immediately the independence from the workflow engine itself.

# 6 Conclusion

In our study we wanted to analyze the ERP business object capabilities, behavior and usage in workflow systems to demonstrate the level of solutions in the evolution stairs. We have considered only two built-in ERP based workflow systems with their simple capabilities. We did not involve any real WFMS systems, nor any

other existing workflow technologies, protocols, modeling standards, because they are not really available in the observed solutions. The standard interoperability between SAP Workflow and Microsoft client tools are also out of scope of this paper. Our goal was to inquire

- the cross-system capabilities of ERP built-in Workflow systems based on ERP objects,
- the role of the objects in these solutions,
- the waknesses and strength of the built-in workflow systems and
- the usability of such built-in workflow system as WFMS systems (central workflow engine).

We have to understand that BPM is not a technology and only an evolution grid of workflow management, but the company implementing it has to be able to think about flexible, reconfigurable business processes. The IT only gives some technology to be able to execute the business requirements. The modern company environments do not contain only an ERP solution, but many other components according to the ERP II concept. Some workflows are starting in an SRM system continue in the ERP and completes in the SRM again. On the other hand, these components are more collaborative, e.g. some of these like SCM (Supply Chain Management) or SRM (Supply Relationship Management) where partner systems should collaborate and exchange data or even call external functions (like component replenishment, managed inventory, contract management, etc.). If a company can achieve the stage, where not only the data sharing (EAI), but the service access and executions (ESB) is also centralized, we can say it has an agile IT. Because the IT environment enables the business to react on changes very quickly by redesigning the processes. Before going to the next stage, one important fact should be mentioned. The layers of the workflows using composite services required many restrictions and virtualization from the IT, but opens new world for the management to react quickly to the business changes. Of course it means parallel living, similar composite services according to the time changing, and special interfaces for the different layers, and backend systems. In practice the EIA layer does this, but in theorem inter-communication surfaces should be provided to handle the differences between systems, solutions, when we want them to communicate or even work together. These could be called adapters for inter-communication. (This kind of adapter is also needed if two workflow systems would communicate.) As we have seen in our study and test the analyzed technologies are almost ready to run real Business Process Management environment, but the technology and application level approaches are still fighting.

We summarize our results according to our original goals. Firt of all we have to mention that there are huge differences between the two ERP systems in thinking, technology (object orientation), modernity, approach, event handling, etc.

The Dynamics AX with its modernity can touch only from technology point of view the Business Process Management (BPM) level easier than SAP does. SAP

implements the application level services, so a BPM Workflow engine can build the cross-system application from bigger step using larger granularity of reusable building blocks.

**Cross-system capabilities or Workflow interoperability**

One of the challenges for a built-in workflow system is the capability of interoperability. With the today's technologies, modeling tools and protocols we have the possibility not to use internal workflow solutions, but special purpose solution should be implemented. The selected ERP vendors offer central EIA, ESB, and BPM solutions also as separated products. Because of the price of such solution it is a real scenario to connect different internal workflow systems directly.

As we mentioned earlier, these solutions are totally different in concept, object handling, event handling, modeling and other areas as well. We discovered the direct workflow level communication possibilities of the two selected ERP solutions.

SAP with its SAP Business Workflow provides internal, system level process automation. The processes, triggering events and steps to be executed are designed according business driven concept. SAP Business Workflow has its own, internal modeling tool, which is not UML or BPEL compliant. It was designed and developed for internal purposes. It follows the WfMC directives about the possible task types and main architecture points. SAP opened the systems to the World by offering its standard communication protocol RFC (Remote Function Call), and the remote enable BAPI modules as public methods of the Business Objects used in Workflow steps. With these old fashion techniques SAP could handle Workflow level interconnection. With the newer releases SAP implemented the Web Service capability using two layers: each RFC enable function can be invoked as web service, and object oriented web services can be developed. SAP manages the standard web service protocols, like WSDL, UDDI, and SOAP, providing possibility to consume external web services as well. To bundle these to the workflow SAP created special Business Objects with BADI elements offering external workflow triggering and receiving functions. These interoperability features are based on the so-called Wf-XML communication protocol, but technically e.g. the Workflow triggering standard function is also RFC enabled and so, can be called from outside SAP as well.

The Microsoft Dynamics AX solution approaches the workflow management from technical side. The whole solution is object oriented in thinking, concept and implementation as well. The workflow is a technical management, process flow of technical objects, using object methods. It is hard to follow the real, high level data and process flow, because the technical elements should be developed and manipulation, not the business level objects. The triggering mechanism is mainly dialog activity dependent, like pushing a button, selecting a menu item. On the other hand the objects are reused many times in the environment. Same workflow can be triggered from more points as well. The newer versions of MS Dynamics do not contain own workflow engine, but they rely on the operating system level standard, independent workflow engine (Windows Workflow foundation). Here we can recognize the reusability again, that Microsoft develops different products and

they are using the features offered by the others. The WWF does not offer too much remote communication possibilities, but enables consuming and producing web services. For external service call WWF offers the so-called ExternalDataExchangeService class delivered among others under the Windows system environment (namespace System.Workflow...). This namespace offers the some Workflow dependent classes to manage and coordinate the workflow runtime and workflow instances. In the .NET framework using C# programming language this method can be used. Internal Workflows can be deployed as web services for interoperability support like ASP.NET web services. The ASP.NET cookies are working like a proxy service around the workflow instance. All that means the consumer should support ASP.NET cookies as well. The other way around works trough the class InvokeWebServiceActivity offered in the same System.Workflow.Activities namespace.

These deep technical aspect shows that we are not really dealing with the MS Dynamics AX layer, but the underlying operating system level embedded workflow engine. To really implement the cross-system workflow between SAP and MS Dynamics AX we have some possibilities:

- Starting SAP Workflow from MS Dynamics by means of RFC or web service call (based on .NET connector)
- To start a workflow in MS environment from SAP we have to use intermediate layer based on .NET (using ASP.NET cookies) or the separate EAI product: BizTalk.
- Invoking SAP Workflow using embedded web service call with .NET based InvokeWebServiceActivity class.

As conclusion we can state, that MS Dynamics AX via WWF offers less cross-system workflow possibilities, than SAP. SAP with its business level approach makes easier the connectivity.

**Role of ERP Objects, Thinking-approach, weakness, strength**

The ERP objects are different in the two ERP systems. SAP uses logical, but business objects, which deliver high level design capabilities. The internal workflow modeler is not dealing with the triggering method directly, but focuses on the object content and the possible methods of it, which can be used in the workflow as single tasks. The workflow holds the business object during run time and handles them like persistent objects. These Business Objects and object types are not always implemented as real object oriented objects and classes, but rather old fashion procedural subroutines. The Business Architecture Framework hides these differences and offers a management and programming layers using object oriented thinking and handling. The new versions of Business Objects are implemented with real OOP techniques. SAP provides off-the-shelf web service capability for public business object methods and also higher level enterprise services for better support of the SOA, ESB and BPM directions.

Microsoft Dynamics AX uses different level of ERP object, from technical elementary classes to higher level AOT classes as well. Unfortunately the workflow is not really business driven, so it was developed according to technical requirements following the technical rules, offering technology level features. The existing components are not integrated, but reused in many cases and many levels (like Windows Workflow Foundation). The workflow approach was originally based on new ERP level solution, but it offered very weak, simple capabilities. The newer versions are using the Windows level, "external" workflow engine offering more task and activity types, but it is further from the ERP object.

Figure 8 shows the different technical level workflow engine approaches. The schematic figure explains the differences in case of having two SAP or two AX systems on the same host.

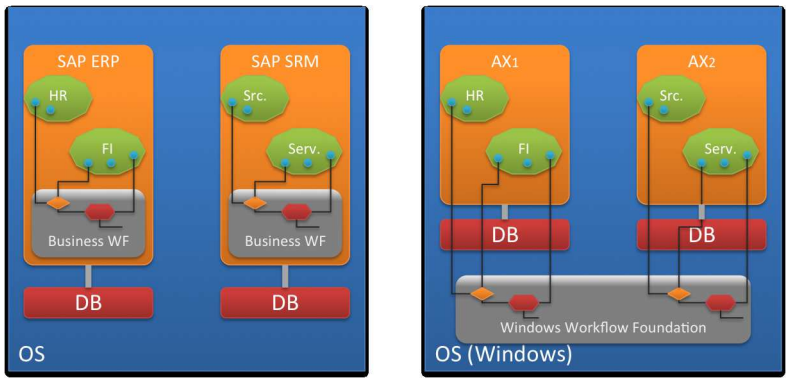


Figure 8: Different workflow engine approaches: SAP and AX

SAP is coming from the business world and the development approaches are derived from the business requirements. Fully integrated, e.g. own developed workflow engine exists in each SAP system. Microsoft Dynamics AX uses the common (for each application running on the same host) workflow engine. The Figure 8 represents with green areas the business components, like HR (Human Resources), the small light blue circles serve as objects (within SAP as Business Objects and their methods; within AX as AOT elements and their methods). The gray areas represent the workflow engine on both sides.

**Business Process Management, built-in workflow as WFMS**[3] In a BPM environment the ERP functionality mustn't show the original surfaces, transactions, but lower level, faceless functions, services should be provided, which execute specific steps, modifications, data retrievals in the backend system. These services should be offered as web services, or via EIA layer should be embedded into web service calls.

Microsoft Dynamics AX offers in the newer releases only services for workflow tasks, but uses "external" (not into the system built) workflow engine (WWF)

as default. This can help to move into a BPM environment easier. Many of the business logics and simple workflows are programmed into end-user UI screens and screen-elements (like menu items). These should be reworked before using them in BPM environment.

SAP has a lot of services to be used in a BPM environment, but the standard transactions are mainly screen based, so the business logic is not separated like in an MVC (Model-View-Controller) environment. SAP works on separating these layers to be able to more effective in BPM and faceless environments. (The standard RFC and BAPI functions provide too small granularity, but the Enterprise Services are sometimes too complex for daily use in a BPM environment.)

According to the requirements each ERP solution should be refactored by separating the user interfaces, business logics and data retieval or storing layers. The last two layers can be parts of the EAI (data communication) and ESB (service, function repository) centers.

The ERP Objects are the basic elements in both cases of the automated flows. They are used for other purposes as well independently whether they are real objects or only logical ones. According to our study we can say, that the observed ERP Workflow implementations are really useful and comprehensive within their environment. But in case of communication or using them as real FWMS, they cannot provide the required features and capabilities. Each product work according to their internal standards not following the currently available BPM standards. On the other hand both vendor provide additional solution (Microsoft BizTalk, SAP Process Integration or SAP Process Orchestration) for better workflow communication using standard (like BPEL, BMPN) and offering real WFMS solution. To reach the SOA, BPM and cloud based workflow concepts; the ERP systems should be changed to separate the business logic, business process from the visualization and from the internal process chains offering real, standalone services.

Our future plan is to compare off-the-shelf and open-source WFMS systems from connectivity, flexibility, UI and model design aspects.

# References

[1] Stohr, Edward A., and J. Leon Zhao. *Workflow automation: Overview and research issues.*. Information Systems Frontiers 3.3 (2001): 281-296.

[2] D. Hollingsworth. *The Workflow Reference Model.* Workflow Management Coalition, Document Number TC00-1003, Winchester, 1995.: 6

[3] Ko, Ryan KL. *A computer scientist's introductory guide to business process management (BPM)* Crossroads 15.4 (2009)

[4] Rickayzen, A., Dart, J., Brennecke, C. and Schneider, M. *Practical Workflow for SAP* Galileo Press (2002) ISBN 1-59229-006-X

[5] Orosz, T. *Analysis of SAP Development tools and methods* Intelligent Engineering Systems (INES) (2011)

[6] Srivardhana, T. and D. Pawlowski, S. *ERP systems as an enabler of sustained business process innovation: A knowledge-based view.* Journal of Strategic Information Systems 16.1 (2007): 51-69.

[7] Jablonski, S. and Bussler, Ch. *Workflow management: modeling concepts, architecture and implementation.* Cengage Learning EMEA (1996) ISBN-13: 978-1850322221

[8] Fife, M. *Dynamics AX 2012 Blueprints: Developing a Product Approval Workflow* Amazon Digital Services (2012) ISBN: B00GHXYCHQ

[9] Pocius, M. *Microsoft Dynamics AX 2012 Development Cookbook* Packtpub Publishing (2012) ISBN 139781849684644

[10] Birch, A. *Implementing Microsoft Dynamics AX 2012 with Sure Step 2012* Packt Publishing (2013) ISBN 978-1849687041

[11] Selmeci, A., Orosz, T. *SAP remote communications* The POLITEHNICA University of Timisoara, Romania (2012), Vol. 57(71), No. 4 ISSN 1224-600X pp., 267-274

[12] Dumas, M., Aalst, W.,HOFSTEDE, A. *Process-aware Information Systems* John Wiley & Sons, Inc. (2005) ISBN 978-0471663065

[13] Cardoso, J., Bostrom, R.P., Sjeth, A. *Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications* Kluwer Academic Publishers (2004) Information Technology and Management 5, 319338

[14] Selmeci, A., Orosz, T. *Usage of SOA and BPM changes the roles and the way of thinking in development* SISY 2012 Subotica, Serbia (2012), ISBN 978-1-4673-4751-8, pp 265-271

[15] Chappell, D. *https://msdn.microsoft.com/en-us/library/aa480215.aspx* MSDN (2005) Using Windows Workflow Foundation