Óbuda University

PhD Thesis

Enhancement of business information systems for small and medium sized enterprises, while transferring the business processes and services into computational cloud.

Kis- és közepes vállalatok üzleti információs rendszer funkcióinak bővülése, az üzleti folyamatok és szolgáltatások számítási felhőbe történő áttelepítése esetén

by

István Orosz

Supervisor

Tamás G. Orosz, PhD, habil.

Székesfehérvár, 2024.

Members of the Defense Committee:

Members of the Comprehensive Examination Committee:

Date of the Defense:

Abstract

As cloud-based technologies are evolving and takes place in everyday IT technology, it is obvious that Enterprise Resource Planning (ERP) systems are also part of the change. Cloud based information technology has created a need for a new software abstraction layer above the traditional implementation layers, and therefore radically changed the way how Enterprise Resource Planning (ERP) systems are implemented and developed over the various hardware and software abstraction layers. The complex part of the changes is that ERP systems mirror an end-to-end business process of a company. This paper focuses on the small and middle size companies in the area, considering the fact that their resources are more limited than the large-scale ones, therefore more focus must be put on code refactoring and reusing. As these organizational processes and the company objectives, goals commonly form the information system (IS) management, it forms a complex task for business process reengineering (BPR).

I introduced a new method for identifying the code parts which can be reused during the uplift process. I also made a proof of concept for validating the new process, and demonstrated is results. I managed to prove the new process in real life projects, which are demonstrated in my thesis, expect part which are under live intellectual property coverage.

In the next chapters I follow up my new method during the BPR process, which is a complex one, as customer needs, company goals and even already existing proceedings must be aligned together, which often difficult to reconcile. Another difference is that implementation of agile project management methodology, which brings new aspects into consideration: moving away from the waterfall model which is based on a static type software development progress a towards an evergreen agile solution [20]. This needs a change in the attitude on the project management and software development side also: instead of searching for and trying to reach the always 100% flawless solution, this method tries to iterate towards a continuous improvement for the project and the end product.

Previous lifecycle management processes from the assessment phase unto the post go-live and business as usual support phase handled the business logic as one common entity with its implementation. Which practically means, that the question of code reusability has now a new role as in the well-known on-premise model. I have introduced a new method of identifying and encapsulating those software parts, which later can be reused in a cloud SaaS environment. As the SaaS model has brought a new abstraction layer into the operation model of cloud-based software, the question of refactoring becomes more important, because the core business logic remains the same (or very similar).

This thesis introduces a change management methodology demonstrated through a specific ERP system, cloud-based Microsoft Dynamics 365 Finance and Operations, from the viewpoint of business process lifecycle maintenance implementation project. This implementation process is suitable for supporting long term maintenance and the evergreen property of the software product. This schema can result in longer lifetime for the software product, as it can be fully segregated from the constantly changing physical implementation layer. As the constant implementation of new technologies is present in the recent IT architecture, the introduction of this new abstraction layer is very useful, because unlike the technologies behind the basic fundamental of the business processes do not change this rapidly.

Enterprise Resources Planning (ERP) Systems, together with Workflow Management Systems (WFMS) have evolved in the past parallel. When we speak about workflow we mean usually as the computerized automation or facilitation of a business process, which covers a part or a whole [51]. Workflow is an automated part of a business process, which executes the work steps in a certain order; information, tasks or even documents are distributed and moved from one participant to the other, who then execute actions following rules on the objects. The main business need and drivers for this kind of automation of course came from the ERP world, but after then WFMS solutions have discovered their own way without ERP, as figuring out the necessity of such applications. Nowadays all the Tier 1 ERP solution providers have their own WFMS solutions standalone from their original mainstream ERP system.

In the final part of the thesis, I demonstrate the connection between the ERP architecture, and the workflow systems during the cloud based operation. The central elements, which carry and handle business data and meaning, are the ERP object. The interoperability of the built-in workflow systems will be presented as key point, then as well describing the required add-on functionalities, tools to provide usable cross system workflow integration in

such an environment. The possibility of using a built-in workflow will be examined, also as a full-featured WFMS. The Workflow is just a theoretical chain of processes and participants assigned with control steps, although the technology enabler is the WFMS. Workflow Management Systems does not only execute the process chains with the aid of software workflow engines, but it has the ability to manage, create and plan them also.

Kis- és közepes vállalatok üzleti információs rendszer funkcióinak bővülése, az üzleti folyamatok és szolgáltatások számítási felhőbe történő áttelepítése esetén

Orosz István

Kivonat

Ahogy a felhő alapú IT megoldások beszivárogtak a mindennapi életbe, úgy ez alól a változás alól a vállalatirányítási rendszerek (ERP) sem tudták kivonni magukat. A változás ezen területen lassabb, hisz egy vállalatirányítási rendszer éltciklusa 10+ években mérhető, és a befektetett összeg nagysága is indokolja a lassú változást. A felhő alapú IT fejlesztés és üzemeltetés újabb absztrakció réteget(ket) jelent a már meglévők fölé. Ebből fakadóan gyökeresen megváltoztatja a módot ahogy egy ERP rendszert fejlesztünk, üzemeltetünk a már meglévő hardveres és szoftveres absztrakciós rétegek fölé. A feladat komplexitása részben abból fakad, hogy egy ilyen rendszer egy vállalat teljes termelés és értékesítési folyamatát lefedi, ezáltal nemcsak a konkrét fejlesztési feladat ami bonyolult, de maga az üzleti folyamatok modellezése sem tartozik a legegyszerűbb feladatok közé. Disszertációmban a kis és közepes méretű vállalatokra koncentrálok, ahol lényeges megjegyezni hogy a erőforrás felhasználásuk sokkal korlátozottabb mint a nagy méretű vállalatoké, ezáltal sokkal nagyobb hangsúlyt helyeznek a kód újra felhasználásra, egész egyszerűen nem tehetik meg hogy egy ERP verzióváltást zöldmezős beruházáshoz hasonlóan kezeljenek.

Bevezetek egy új eljárást az újra felhasználható kód azonosítására, valamint végig vezetem azt egy proof of concept validációs eljáráson. Valós projecteken is sikerült bizonyítani az eljárás működőképességét, amelyet a disszertációmban is bemutatok, kivéve az adott iparági project szerzői jogvédett tartalmát.

Ezek után bemutatom hogy az így felhő alapú üzemeltetésre átemelt ERP rendszer bevezetése milyen módosításokat igényel az üzleti folyamatok újra tervezése oldalán, mivel egy összetett folyamat, mivel összetevői közt megtalálhatóak a vevői igények, céges célok, és a már létező és sikeres üzleti folyamatokat kell egymáshoz hangolni, miközben néha

egymásnak ellentmondó céloknak kell megfelelni. Az agilis project vezetési metodológia bevezetése újabb különbséget jelent a korábbiakhoz képest, a korábbi generációkra jellemző vízesés modell felváltása, ami lényegében egy statikusabb jellegű project menedzsment felől egy dinamikusabbra történő áttérést jelent, és végeredményként egy örökzöld megoldást eredményez [20]. Lényegi különbséget jelent ez nemcsak a project vezetésében, hanem az alkalmazott fejlesztési eljárásban is: ahelyett, hogy a 100% tökéletességű és teljesen letesztelt végeredményt engedjük ki az éles rendszerb, tudomásul vesszük az apróbb eltéréseket és folyamatos fejlesztési ciklusokkal törekedünk a cél elérésére.

Az ezt megelőző életciklus menedzsment eljárások az üzleti logikát az implementációval együtt kezelték, a tervezéstől egészen éles bevezetésen és a támogatási fázison keresztül. Ebből következően a kód újra hasznosításnak teljesen új szerepe van egy újabb absztrakt rétegekre bontott felhő alapú üzemeltetési modell-ben mint az előző évtizedek adatcenteres megoldásban. Az általam bevezetett eljárással azonosítható és körül határolható azon szoftver alkotóelemek, amelyeket egy SaaS üzemeltetési modell esetén újra felhasználhatunk.

Végül bemutatom hogy az így felhő alapú üzemeltetés az ERP rendszerek a velük kompatibilis Workflow menedzsment rendszerekkel együtt milyen változtatásokat igényelnek. WFMS rendszeren egy adott üzleti folyamat teljesen vagy részlegesen automatizált végrehajtását értjük [51]. A WFMS rendszerek létrejöttének fő hajtóereje az ERP rendszerek világa volt a kezdetekben, de később a számítási felhők elterjedésével leváltak a fő ERP rendszerekről és ma önálló életet élnek már.

Acknowledgement

I would like to thank to my supervisor Dr. habil Tamás Orosz for his regular professional level help during my doctoral years.

I dedicate this work to my loving family Tündi and Lívia, to my Father, for their love and support. This dissertation would have not been possible without their continuous support.

Contents

Abbreviations

AD Activity Diagrams

ADM Architecture Development Method

AOT Application Object Tree

API Application Programming Interface

AUML Agent Unified modelling Language

BAU Business as Usual

BAPI Business Application Programming Interface

BPE Business Process Engine

BPEL Business Process Execution Language

BPM Business process Management

BPMI Business Process Management Initiative

BPMN Business Process Model and Notation

BPMN Business Process Model Notation

BPMS Business Process Management Systems

BPO Business Process Owners

BPR Business Process Reengineering

BPR Business Process Reengineering

BPR Business Process Re-engineering

CEP Complex Event Processing

CIM Computation Independent Model

CRUD Create, read update and delete operations

CSP Communicating Sequential Processes

DEVS Discrete Event System Specification

DEVOPS Development and Operations

EAM Enterprise Architecture Management

EPC Event Driven Process Chain

ERP Enterprise Resource Planning system

IaaS Infrastructure as a Service

IDE Integrated Development Environment

IPMS Performance Measurement Systems

IS Information Systems

JDOM Java Document Object Model

MDA Model Driven Architecture

MDD Model-Driven Development

M-V-C Model-View-Control

PaaS Platform as a Service

PIM Platform Independent Model

POC Proof of Concept

PSM Platform Specific Model

RBPMO Role-Driven Business Process modelling

RFC remote function call (SAP)

RPC remote process calls (Windows)

SaaS Software as a Service

SME Small and Middle sized Enterprises

SOA Service Oriented Architecture

SOA Service Oriented Architecture

SOAP Simple Object Access Protocol

SOS Structural Operational Semantic

SQL Structured Query Language

TCO Total Cost of Ownership

UML Unified modelling Language

WFMC Workflow Management Coalition

WFMS Workflow Management Systems

WSDL Web Services Description Language

XAML Extensible Application Markup Language

XML Extensible Markup Language

Introduction

My goal is to develop a new methodology for identifying and isolating reusable code snippets in cloud ERP systems, which can be generally applied in ERP solutions for small and medium sized enterprises. The methodology enables the transition of ERP systems to a SaaS model while reducing development time and total cost of ownership, while keeping the business logic unchanged. I have used the Microsoft Dynamics 365 environment as a tool to validate the method, where I demonstrate the viability of the approach through case studies and prototypes, however the method is general in scope and not limited to a specific product, but formulates a generally valid methodology for developing and optimizing ERP systems for small and medium enterprises. Microsoft Dynamics 365 is only included as a tool to validate the scientific results.

Cloud based software technology changed radically the way how Enterprise Resource Planning (ERP) systems are operated through its lifecycle. The on-premise type of release-to-release lifecycle management driven by main version changes from pre-alpha to gold release through several stages, was transformed into a continuous release and maintenance management. Although the major business processes have remained the same, but how can be examined which software elements need to remain the same, while moving towards to agile type continuous lifecycle support? It is a key question to ensure the continuous support of the software product. Due to the sudden improvement in change management, this is described as Software-as-a-Service (SaaS) type lifecycle management methodology.

As from the implementations take place nowadays, Business Process Reengineering (BPR) is not the necessary result of this process. This thesis introduces a change management methodology demonstrated through a specific ERP system, cloud-based Microsoft Dynamics 365 Finance and Operations, from the viewpoint of business process lifecycle maintenance implementation project. This implementation process is suitable for supporting long term maintenance and the evergreen property of the software product.

Cloud based information technology has created a need for a new software abstraction layer above the traditional implementation layers, and therefore radically changed the way how Enterprise Resource Planning (ERP) systems are implemented and developed over the various hardware and software abstraction layers. The traditional update methodology which drives from release to release governed by the main version change (from pre-alpha to gold release) was substituted with a continuous release management. Inside the Software as a Service (SaaS) model, the core of the business logic is realized over the physical implementation layers. Although the infrastructure which lies behind seems to have longer lifecycle, with the invention of the SaaS operating model the software product achieves longer support and maintenance lifetime.

This schema can result in longer lifetime for the software product, as it can be fully segregated from the constantly changing physical implementation layer. As the constant implementation of new technologies is present in the recent IT architecture, the introduction of this new abstraction layer is very useful, because unlike the technologies behind the basic fundamental of the business processes do not change this rapidly. The SaaS-type lifecycle management does means that the technology independent implementation parts are separated from the business process implementation. Previous lifecycle management processes from the assessment phase unto the post go-live and business as usual support phase handled the business logic as one common entity with its implementation. Which practically means, that the question of code reusability has now a new role as in the well-known on-premise model. This thesis later introduces a new method of identifying and encapsulating those software parts, which later can be reused in a cloud SaaS environment. As the SaaS model has brought a new abstraction layer into the operation model of cloud based software, the question of refactoring becomes more important, because the core business logic remains the same (or very similar).

Platform as a Service and Software as a Service operational models make it a little more complex although the differences could be small for the first sight. Software as a Service (SaaS) service model provides a more sophisticated operational method, offering a lot of advanced features [20]. Infrastructure as a Service: provides the lowest level of hosted services, the service level is responsible for the virtualization, DB storage and servers, network. It is the easiest cloud-based operation level which steps over the datacenter on-demand solution. Platform as a Service: one level above the IaaS we can find this platform, this service level is responsible for the operating system(s) (almost all virtual ones), middleware interface(s) and the delivered runtime software modules. PaaS platforms offer the control over the application

and its data for the end users. Software as a Service: the customer(s) must take care only modelling the core business logic over the delivered software solution, and use the IaaS cloud-based infrastructure as the final implementation service. This the total inverse of the on-demand datacenter model.

Model Driven Architecture development approach is based on specific models, which roles as the basic foundation of design, development and for the whole operation lifecycle. It basically separates the core business logic from its technical implementation. MDA makes another level of abstraction above the already implemented layers, in which the software delivery organizations can create their own OOP objects to represent business logic in a software implementation independent way. MDA has three major abstraction layers:

- Computation Independent Model, contains the business domain model
- Platform Independent Model, describes the system functionality in implementation method independent form
- Platform Specific Model, system specification according to the implementation technology

These three abstraction layers were successfully identified for SaaS implementations from the MDA point of view. Platform Specific Model can be implemented in several different methods, while this article does not scope all the different implementation strategies. The upgrade tasks from the on-premise version to SaaS cloud based version can use the same OS, virtualization technology and middleware, so the PIM->PSM transformation can be done easier, focusing mainly on the technology independent tasks.

Cloud based computing disciplines shows the fifth phase of computing paradigms, from mainframes, personal, to client-server and web-based. Computational clouds can be deployed as public or private. Private clouds are the natural choice for hosting ERP systems which are operated by middle sized (or bigger) companies. This article focuses on the differences after the post go live support which are generated by the different behaviour of the cloud based SaaS support methodology. PaaS platforms are based on the basic idea that it enables third party solution delivery and implementation according to the SaaS model, meanwhile hiding all the complexity of the underlying datacentre and middleware architecture. An MDA (Model Driven Architecture) based development targeted for a SaaS cloud environment is able to

improve the flexibility, robustness and lower to TCO (Total Cost of Ownership) of the software product. All the populated services in the development pipeline can be used in a technology independent manner [8].

SaaS is most appropriate choice to model and ERP system as another abstraction layer, than to develop the software solution for the computational cloud directly. This way we can decouple and distinguish the core business logic from the infrastructure providing layers, therefore enhancing the whole lifetime and heavily decreases the direct dependence from the future technology changes.

MDA based implementation of a cloud software solution based on SaaS can benefit from these properties as services are defined in a technology independent way. Moreover, the method of encapsulating and implementing the core business logic through a technology independent process can improve the future reusability factor of the code [30]. When analysing this new abstraction layer, the question of the performance overhead is valid and interesting task for the future work, just because of the fact that from now every operation must go through some parts of the cloud. For the first sight it seems a very limited added network traffic, latency and extra computational work, they are also added steps which did not present in the on-premise version of an ERP systems. We need to take into consideration the overall solution security and data privacy. These issues are so serious, that till now this is the root cause why a lot of costumers do not want to their ERP solution including their business data into the cloud [38].

Enterprise Resources Planning (ERP) Systems, together with Workflow Management Systems (WFMS) have evolved in the past parallel. When we speak about workflow we mean usually as the computerized automation or facilitation of a business process, which covers a part or a whole [51]. Workflow is an automated part of a business process, which executes the work steps in a certain order; information, tasks or even documents are distributed and moved from one participant to the other, who then execute actions following rules on the objects. The main business need and drivers for this kind of automation of course came from the ERP world, but after then WFMS solutions have discovered their own way without ERP, as figuring out the necessity of such applications. Nowadays all the Tier 1 ERP solution providers have their own WFMS solutions standalone from their original mainstream ERP system. In this thesis the main focus is on the ERP architecture, a technology which implements built-in workflow

systems. The central elements, which carry and handle business data and meaning, are the ERP object. This paper will present the capabilities of such a built-in workflow system of ERP solutions via two Tier-1 ERP solution providers: SAP and Microsoft Dynamics D365 Finance and Operation. Both are dealing workflow management around ERP objects. Through some examples, both the weakness and restrictions of the built-in workflow systems will be presented on these two ERP examples. The interoperability of the built-in workflow systems will be presented as key point, then as well describing the required add-on functionalities, tools to provide usable cross system workflow integration in such an environment. The possibility of using a built-in workflow will be examined, also as a full-featured WFMS. To see how these steps and requirements will lead to workflow environments, some basic terms must be described first of all:

• **Activities:** they can be called steps; if the process flow is automated. A Workflow can be considered as not only the process flow, but it also contains the related information and data, which are coupled to the process itself, and the execution steps and the actions; those are the milestones of a Workflow. One can consider the activity as an atomic element, so technically an activity is the most minimal logical step of the whole process. Activities can be executed manually as well, so these are not Workflow dependent tasks, but as they are where the information or data is checked, displayed, updated or even deleted in a process flow, so they are the main building elements. If an activity requires any human interaction, like decision on acceptance or rejection, they can be executed manually in a Workflow. We can distinguish another kind of activities those require machine resources, like starting a batch job or sending a mail. These can be/are executed in without requiring dialog (or human) activities in a background tier.

• **Business process** is made out of a pre-defined set of activities, which are coupled together for a specific business goal, like procure to pay, sales order generation or general ledger accounting. The granularity of these processes is not pre-defined, and they can vary inside the same business organization also. On the top of this varying granularity, we can build up higher-level or embedded processes also.

• **The participant** of a Workflow has a well-defined role: when and what he can or should do. Inside the Workflow, these participants are usually part of the organization security hierarchy, as they get access to sensitive data also.

It is very rare that the process chain in the Workflow is practically just a simple sequence of tasks, it usually contains decision points, as the control points will generate branches. The Workflow is just a theoretical chain of processes and participants assigned with control steps, although the technology enabler is the WFMS. Workflow Management Systems does not only execute the process chains with the aid of software workflow engines, but it has the ability to manage, create and plan them also. The mainstream WFMS products usually have some kind of a graphical modelling and design tools also. The workflow engine contacts with applications and technology elements and makes dialog with participants also.

Methodology and literature review

Parallel to the evolution of the ERP systems, the functions and the requirements of the audit process also went through a deep improvement.

Citations

It is very interesting to see, which articles got the most citations – also interesting to draw some conclusion which are the most important ways to go forward with future studies and research. Also, it gives good feedback on the impact of the articles and helps focusing the later articles. It must be noted, that I have never choose a focus for the next article on the basis of the past citations, all articles were prepared according to the research plan. Here is the short list of the citations, based on the status July 2024 in Google Scholar:

Key Performance Indicators used in ERP performance measurement applications A Selmeci, I Orosz, G Györök, T Orosz 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and …	40	2012
The role of data authentication and security in the audit of financial statements L Szívós, I Orosz Acta Polytechnica Hungarica 11 (8), 161-176	10	2014
Software as a Service operation model in cloud based ERP systems I Orosz, A Selmeci, T Orosz 2019 IEEE 17th World Symposium on Applied Machine Intelligence and …	9	2019
Information technology systems in logistics and roles of ERPs L Duma, I Orosz 2012 IEEE 13th International Symposium on Computational Intelligence and …	7	2012
Company level Big Data Management T Orosz, I Orosz 2014 IEEE 9th IEEE International Symposium on Applied Computational …	6	2014
Code reusability in cloud based ERP solutions I Orosz, T Orosz 2017 IEEE 21st International Conference on Intelligent Engineering Systems …	4	2017
Software as a service in cloud based ERP change management I Orosz, T Orosz 2017 IEEE 15th International Symposium on Intelligent Systems and …	4	2017
Business process reengineering project in local governments with ERP I Orosz, T Orosz 2012 7th IEEE International Symposium on Applied Computational Intelligence …	4	2012

Fig.1. Google Scholar citations for some of my articles, at least with four hits.

Note: citations must be validated against MTMT and IEEE search engines also.

Google Scholar citations are based on profile pages which were set up by the authors for their work. This is an automatic process, and because it is based on a public google profile, it cannot be used as an authentic source of truth, although it is very useful as it is easy to access and has some cool features.

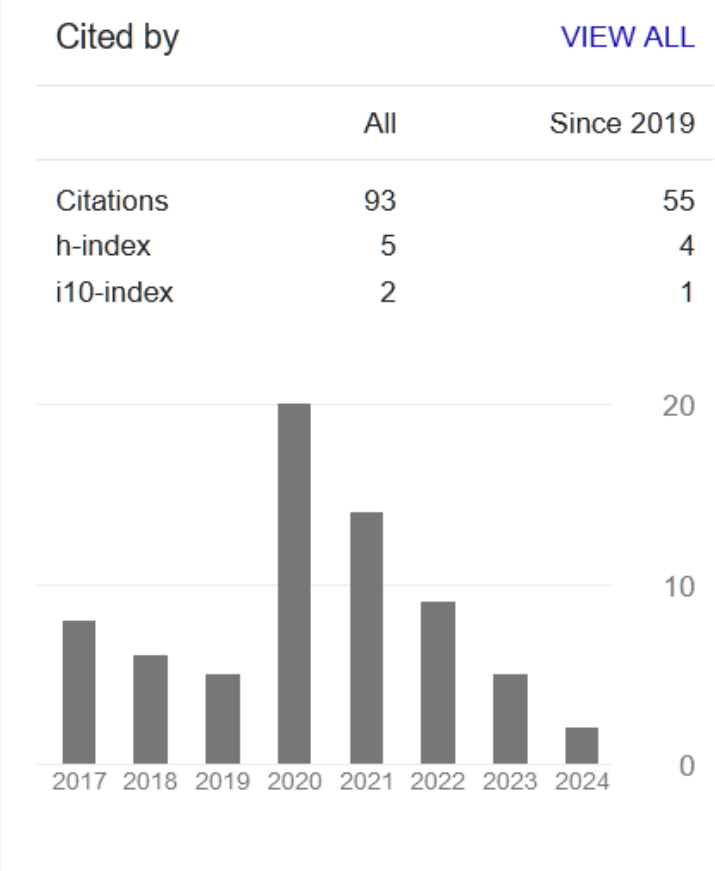


Fig.2. Google Scholar citations since 2014.

The biggest advantage of this site is the automatic maintenance, not just for your own work, but for co-writer states also.

MTMT list of publications

Here is the list of my publications currently available in MTMT (Magyar Tudományos Művek Tára / Hungarian Science Bibliography) available at: https://www.mtmt.hu/

[1]O. István, S. A., and G. T. Orosz, "Software as a Service operation model in cloud based ERP systems," in 2019 IEEE 17TH WORLD SYMPOSIUM ON APPLIED MACHINE INTELLIGENCE AND INFORMATICS (SAMI 2019), 2019, pp. 345–354.

[2]O. István and G. T. Orosz, "Code reusability in cloud based ERP solutions," in 2017 IEEE 21st International Conference on Intelligent Engineering Systems (INES), 2017, pp. 193–198.

[3]O. István and G. T. Orosz, "Software as a Service in Cloud based ERP change management," in 2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), 2017, pp. 181–186.

[4]I. Orosz, "Business process sustainability in cloud based SaaS environments," in AIS 2017 - 12th International Symposium on Applied Informatics and Related Areas organized in the frame of Hungarian Science Festival 2017 by Óbuda University, 2017, pp. 171–175.

[5]I. Orosz and T. Orosz, "Microsoft Change Management applying comparison of different versions," ACTA TECHNICA JAURINENSIS, vol. 7, no. 2, pp. 183–192, 2014.

[6]O. Tamás and O. István, "Optimization of Enterprise Business Processes with Big Data Management," BULETINUL STIINTIFIC AL UNIVERSITATII POLITEHNICA DIN TIMISOARA ROMANIA SERIA AUTOMATICA SI CALCULATORAE, vol. 59, no. 2, pp. 105–110, 2014.

[7]O. István, "The role of system specific rules in migrating master data," in 8th International Symposium on Applied Informatics and Related Areas: AIS 2013, 2013, pp. 12–16.

[8]O. I. and O. T., "Business Process Reengineering project in Local Governments with ERP," in 7th IEEE International Symposium on Applied Computational Intelligence and Informatics, SACI 2012, 2012, pp. 371–376.

[9]G. T. Orosz, I. Orosz, B. Túri, and A. Selmeci, "Local Governments-specific BPR mini-project with SAP applications," in IEEE 10th Jubilee International Symposium on Applied Machine Intelligence and Informatics, SAMI 2012, 2012, pp. 119–123.

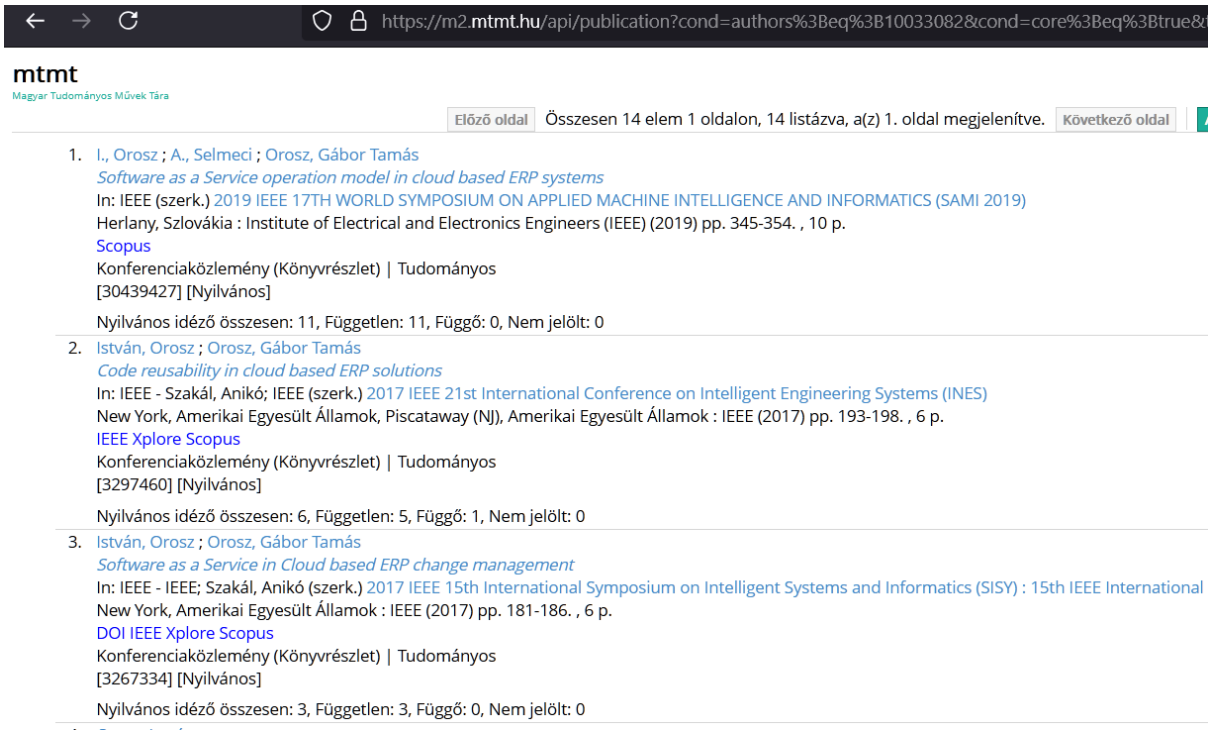
[10]O. Tamás, O. István, and S. Attila, "Continuous ERP Strategy Development and Implementation at Enterprises," in 7th International Symposium on Applied Informatics and Related Areas: AIS 2012, 2012, pp. 35–39.

[11]D. O. T G, S. A., O. I., and G. Gy., “ERP change management for innovation and sustainability applied to User Interfaces,” in Proceedings of the IEEE 16th International Conference on Intelligent Engineering Systems 2012, 2012, pp. 341–346.

[12]I. Orosz, B. Túri, and G. T. Orosz, “Inherited SAP Development Concepts using genuine IT programming tools,” in CINTI 2011 - 12th IEEE International Symposium on Computational Intelligence and Informatics, 2011, pp. 561–566.

[13]G. T. Orosz, A. Selmeci, and I. Orosz, “SAP BAPI as a Break-through and Future Communication Enabler,” in Proceedings of AIS 2010 [elektronikus dok.] : International Symposium on Applied Informatics and Related Areas : ... in the frame of Hungarian Science Festival 2010 [AIS 2010], 2010, pp. 48–53.

[14]G. T. Orosz, E. Sonn, B. Túri, and I. Orosz, “Application of Reliable and Comfortable SAP Development Tools,” in Proceedings of AIS 2010 [elektronikus dok.] : International Symposium on Applied Informatics and Related Areas : ... in the frame of Hungarian Science Festival 2010 [AIS 2010], 2010, pp. 68–73.



Research methodology – creating a POC

This is one of the most important stages when doing preparation for an enterprise level project. Independent from the POC outcome, it is worth mentioning that it gives answer to these questions:

1. The project board can get a view on is it worth spending money and resources for the project implementation.
2. Avoid to loss resource and budget on a project which are not viable or market ready.

Important to note, that use of Microsoft Dynamics 365 Enterprise Resource Planning system is only for the use test case, that thesis presents methodology which can be adapted to any other ERP system regardless manufacturer.

Some parts of the article and the thesis contains results oriented in software development and change management areas.

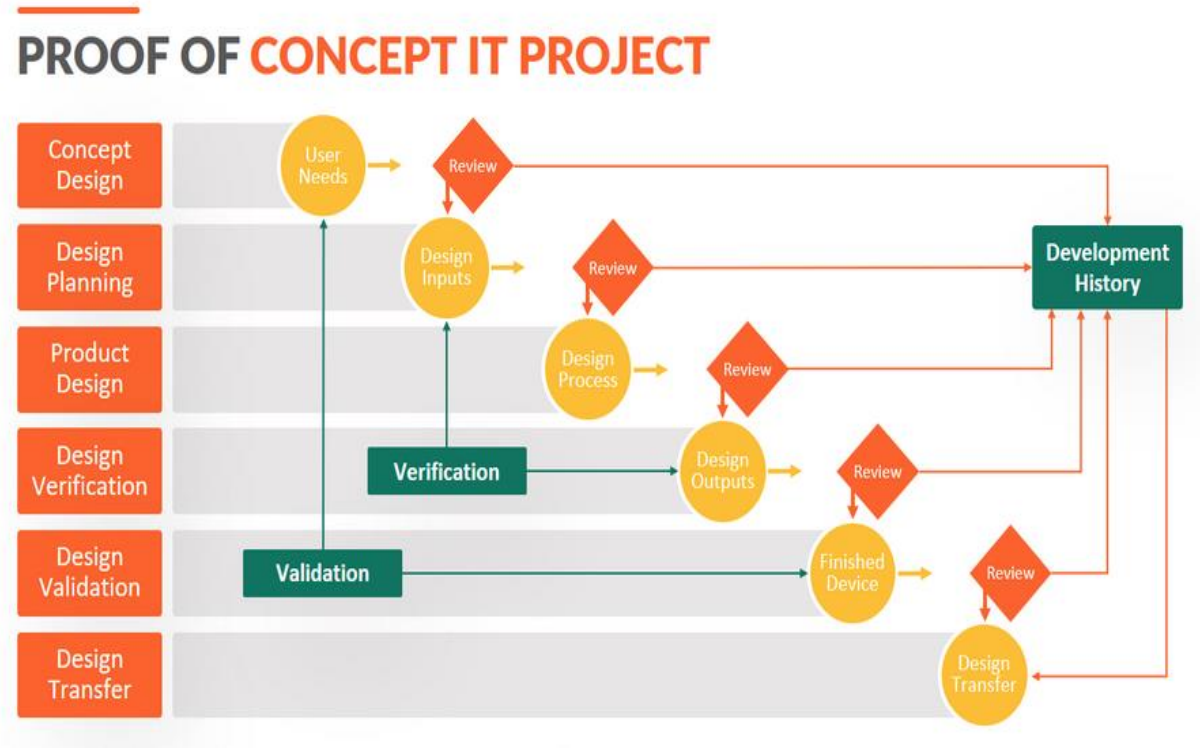


Fig.3. Lifecycle of a POC IT project [59].

The used research methodology was always to build at least a Proof of Concept for the statement. In some cases, it was possible not only to build a POC model, but to use the results in live scenarios during my several projects during these years.

In general, we build a POC when the idea is unique, so we cannot use already existing examples. It is a theoretical demonstration of a concept, which can help in determining that the idea is viable, without putting more effort in the implementation. Also, important to get some knowledge about the concept's potential, future maintenance needs, test whether it will work as visioned. We will get some information about the potential technical issues which can arise during the implementation, and how it will serve the organizational business goals and processes.

Other main points we can achieve by building a POC:

- Verify and validate the adequacy of the used technology in the specific computational could.
- Obtain feedback on future day-to-day processes and operation.
- Validate the cost/benefit projections of the project.
- Rethink resource need projections.
- Possible test for third party interfaces.
- Verify the system stability and viable for daily operations.
- Verify the interoperability and system design.
- Verify the system and its cloud-based architecture performance.
- Produce output samples.
- Provide results for full scale implementation.
- Helps planning the necessary knowledge transfer sessions.
- Helps creating test cases, KPI to measure the final product success.

- Establish a project and quality management framework.
- Establish the training and communication framework.
- Develop quality acceptance framework. [40]

Important to mention, that we did not build any prototype version of the POCs, as it was not in the scope of the thesis. Some of the POCs built, were realized into industry live projects later, so that was where the validation happened. These are the main steps creating a POC:

1. Define the idea which worth for creating a POC.
2. Define the KPI which will measure the success of the POC.
3. Build the POC.
4. Perform test cases.
5. Evaluate the KPI.

We could not use any free POC templates, as the ones we were built were very unique and connected closely to an industry solution. There are several different advantages of building a POC from the client point of view, and from the implementation team's point of view. On client side, it is a suitable time for thinking through the business idea behind, gather materials which can be relevant, define your hypothesis and ideas, as well as the objectives and KPIs [59]. For the POC implementation phase, company has to find a suitable implementation consultancy partner, with the necessary technical and process management capabilities. It is also worth mentioning, that throwing away and idea in this phase is a natural step, so is acceptable if a POC do not survive this phase and do not become a prospect for a real project.

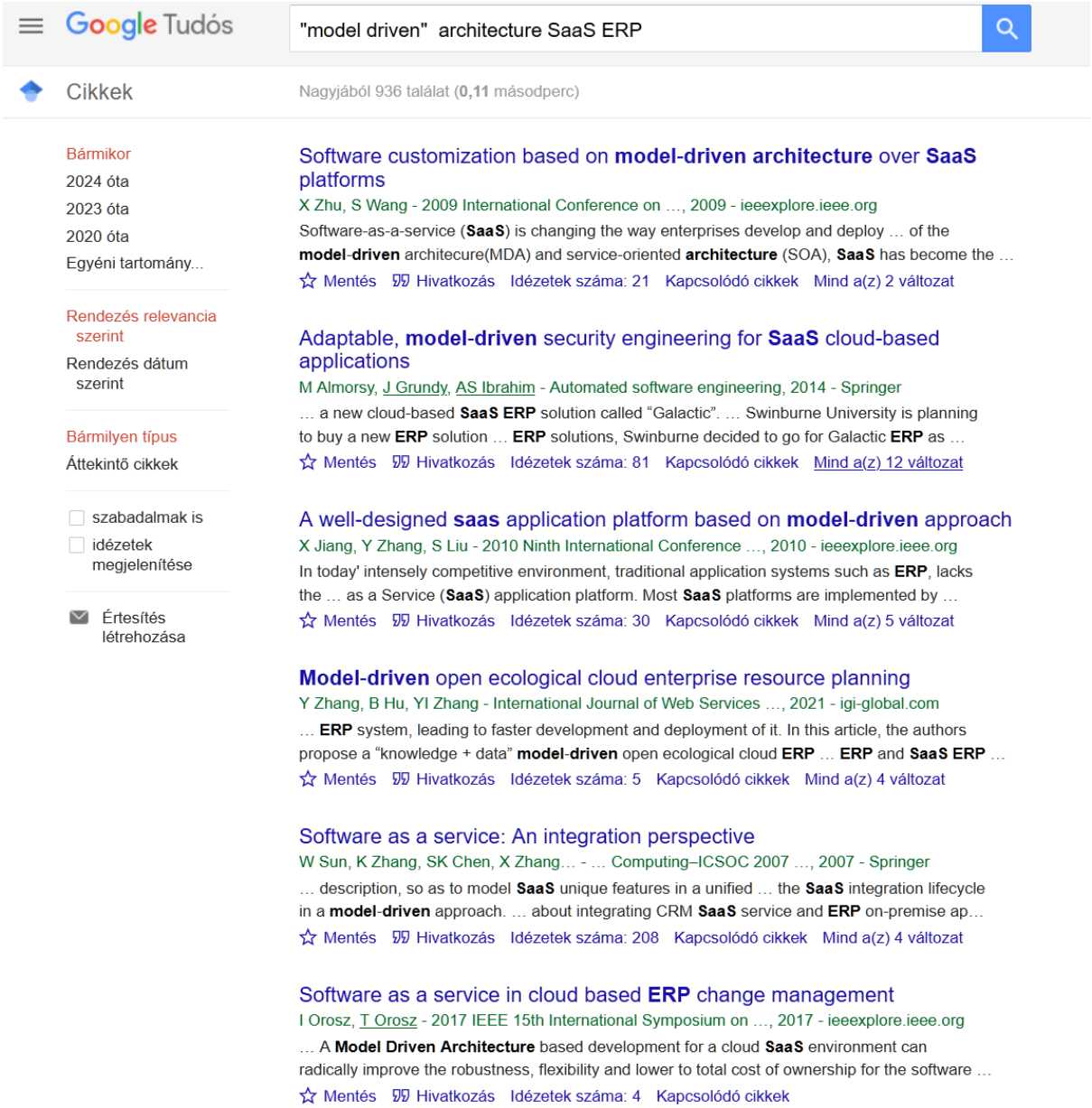
The implementation team also has some advantages during the POC phase of a project, like do a deep fit-gap analysis. It is also a compulsory step to do identify the risks and the goals, to determine the development KPIs and the objectives. It is also worth mentioning, that analysing the feasibility of the project and if not applicable to offer an alternative solution is also part of the POC phase. As the last preparation for the implementation phase, documentation is needed with the possible outcomes.

Uniqueness of the solution

My independent scientific contribution lies in the development of a general methodology for identifying and reusing code components within cloud-based ERP environments. Although the I applied Microsoft Dynamics 365 as a validation tool, the approach is universally applicable to ERP systems in small- and medium-sized enterprises. This methodology optimizes the transition to a SaaS model by reducing development time and operational costs while preserving essential business logic. I have independently developed a generalized methodology for identifying and encapsulating reusable code components in cloud-based ERP systems. Although Microsoft Dynamics 365 was used as a validation platform, this methodology is broadly applicable across ERP environments in SMEs. It demonstrates significant improvements in operational efficiency and cost-effectiveness, providing a robust framework for ERP transitions to cloud-based models while maintaining core business functions.

After broad research of the literature, I can conclude the uniqueness of the solution presented in the Code reusability in cloud-based ERP solutions, and later chapters. There are no publicly available articles presented, where the Computation/Platform Independent Model and Platform Specific Model are demonstrated together with of a Proof of Concept using different abstraction layers of SaaS cloud computing platform. Although MDA approach is used generally in organizing these kinds of classifications, it is considered to be an industry wide basic technology nowadays [79].

I made deep research in Google Scholar; the result was that my article is in the 5th place when rendering the result for relevance:



None of the articles in the first 20 (that was the most relevant list) describe the same method as my thesis, so I can be sure of the uniqueness of the solution presented in the Code reusability in cloud-based ERP solutions, and later chapters. I made similar research on the publicly available other sources, with the same result.

Code reusability in cloud-based ERP solutions

It is a very complex scenario to calculate the total cost of ownership of an Enterprise Resource planning system. Those agile methodology-based systems, which are currently used make the whole calculation even more complex, as the agile methodology operates in development cycles, and there is no predefined number of the cycles.

- **Research Gap**: While many businesses aim to optimize cloud-based ERP systems, the broader principles of code reuse across multiple platforms and lifecycle stages remain underexplored, particularly regarding their long-term financial and operational impact.
- **Research Question**: How can general principles of code reusability in cloud-based ERP systems improve the total cost of ownership (TCO) and streamline project timelines for Small and Middle sized Enterprises (SMEs)?
- **Hypothesis**: Implementing methodologies for code reusability in cloud-based ERP systems significantly reduces development time and operational costs while maintaining system integrity.

It is important to note, that use of Microsoft Dynamics 365 Enterprise Resource Planning system is only for the use test case, that thesis presents methodology which can be adapted to any other ERP system regardless manufacturer.

Own publications behind this chapter

Before moving on to discuss the topic of this chapter, let's summarize the own work behind. There are four articles which are corresponding:

- István Orosz, Tamás Orosz, Software as a service in cloud based ERP change management, 2017 IEEE 11th International Symposium on Intelligent Systems and Informatics (SISY 2017), Page(s): 000181 - 000186,

- István Orosz, Tamás Orosz: Code reusability in cloud based ERP solutions. Proceedings of the IEEE 21st International Conference on Intelligent Engineering Systems (INES 2017): Larnaca, Ciprus. IEEE, 2017. pp. 193-198

- Tamás Orosz, István Orosz, Company level Big Data Management, 9th Jubilee IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI 2014). Budapest, Magyarország: Óbudai Egyetem, Page(s): 299 - 303

- . Orosz*, A. Selmeci**, T. Orosz: Software as a Service operation model in cloud based ERP systems. IEEE 17th International Symposium on Applied Machine Intelligence and Informatics, 2019

These four articles contain the base research which later become foundation for this chapter. Each of them reflects the different aspect of the same area, but all in all they provide a complex view of code reusability tasks inside ERP systems.

A typical periodization of the full lifecycle implementation looks like the following [18][19]:

- Diagnostic
- Analysis
- Design
- Development
- Deployment
- Operation

Development phase is one of the most expensive part is the development phase, usually specialized teams do this activity. it is a vital question therefore, when it comes to a major version upgrade, how can we find a way to identify which objects of the code can be reused with no major modifications? Before cloud era, an easy answer to this question was given by

the on-premise operational model. Between two major version upgrades the operational model has not been changed, so all of the code upgrade could take place in the same abstraction level.

This is the standard architectural model which was used by the traditional client-server like architectures, where the following elements could store code elements:

- Client (in most cases it was a thick Windows client)
- Application server
- SQL database

By the time when the ERP systems occupied their place in the cloud, this operational model was totally changed. As physical infrastructure was moved from the datacenter to the cloud infrastructure meaning that the whole infrastructure-based operation is not in the scope anymore. Bandwidth and network latency became a critical point of operation on the other hand.

	On Premise	Cloud
Security	No dedicated internal IT group for security	Dedicated IT resources are needed to handle security internally
Costs	Cost are in correlation with the required resources, can be changed rapidly	Cost is in correlation with the size of the infrastructure
Maintenance	No dedicated IT support group	Dedicated IT support group

		with the company

Table.1. Comparison of on premise and cloud based operational models

Platform as a Service and Software as a Service operational models make it a little more complex although the differences could be small for the first sight. What makes the biggest difference is implementing a new abstraction layer for the business core logic.

It is the Platform as a Service (PaaS) operating in the cloud, which enables third party software solution service delivery, and hides the detailed complexity of the below implementation layer. This means that the cloud implementation layer covers the whole middleware and datacenter technical solution. So, while hiding the whole architecture implementation from the solution in the cloud, it offers platform services.

Software as a Service (SaaS) service model provides a more sophisticated operational method, offering the following key features [20]:

- The software solution and its data components are stored in a common place, widely known as “the” cloud solution.
- Clients access the software solution via internet using a thin client, and they can be charged on usage base – instead on license base.
- The whole infrastructure implementation is present on the vendor side, and its resource needs can be reconfigured by the customer without any hardware side or low-level software side changes.
- Customers always use the latest version of the software, which is called evergreen solution, provided on-demand and do not have to bother with version upgrade.

Software reusability in Cloud based SaaS model

This chapter focuses on the software code reusability of a cloud based software solution uplifted to a Software as a Service (SAAS) operation model, shown in Fig.1.

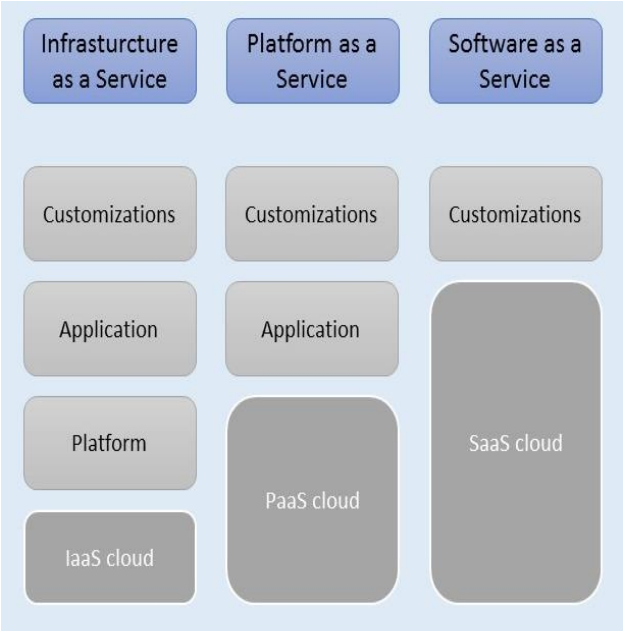


Fig.1. ERP cloud models [24]

This operating model introduces a new abstraction layer, which practically means that the software product has to adapt to new requirements. Therefore, some (or in worst case scenario all) layers of the original on-premise version have to be rewritten. The main question is how to determine the minimally needed code, which needs modification [30]?

The actual software product, on which I have made my experiments because of the complexity of the codebase, is the Microsoft Dynamics 365 cloud version, using Microsoft Azure cloud as Platforms as a Service (PaaS), which offers the solution as a service all in one.

PaaS operating platforms can make it possible for software providers to build and deliver services suitable to the SaaS model, meanwhile hiding all the highly complicated underlying middleware components and the whole datacenter network and server architecture [22].

- Infrastructure as a Service: provides the lowest level of hosted services, the service level is responsible for the virtualization, DB storage and servers, network. It is the easiest cloud-based operation level which steps over the datacenter on-demand solution.
- Platform as a Service: one level above the IaaS we can find this platform, this service level is responsible for the operating system(s) (almost all virtual ones), middleware interface(s) and the delivered runtime software modules. PaaS platforms offer the control over the application and its data for the end users.
- Software as a Service: the customer(s) must take care only modelling the core business logic over the delivered software solution, and use the IaaS cloud-based infrastructure as the final implementation service. This the total inverse of the on-demand datacenter model.

Model Driven Architecture (MDA)

Model Driven Architecture development approach is based on specific models, which roles as the basic foundation of design, development and for the whole operation lifecycle. It basically separates the core business logic from its technical implementation. This is the key point, how it can act as the starting point for distinguishing those code parts which can/must be refactored.

Practically MDA can be divided into these three subtasks:

- Computation Independent Model, contains the business domain model
- Platform Independent Model, describes the system functionality in implementation method independent form
- Platform Specific Model, system specification according to the implementation technology

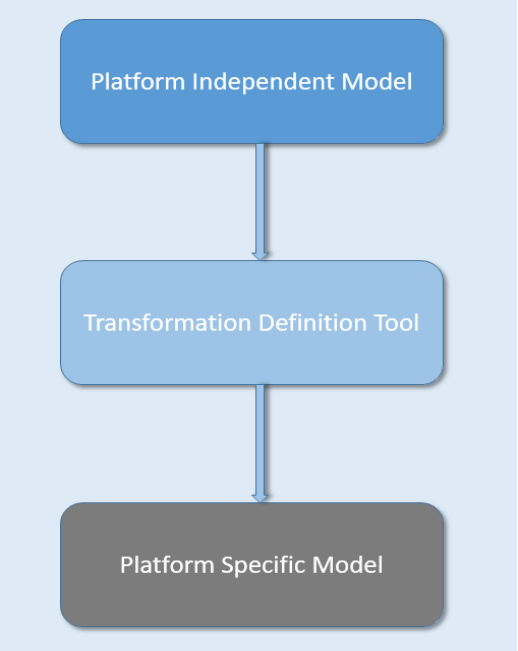


Fig. 2. Model Driven Architecture layers

MDA makes another level of abstraction above the already implemented layers, in which the software delivery organizations can create their own OOP objects to represent business logic in a software implementation independent way. The modelling language for this business abstraction layer is important for our goal: to identify the specific code parts. Those properties which describe the requirements are vital to identify the similar business processes and attach them to the implemented code parts.

Platform Independent Model definition

There are several ways to define the implementations of a Platform Specific Model, analysing them is not in the scope of this paper. Instead of it focuses to find a close to optimal way how the PIM model can be used to get the maximal common set of objects between the upgrade paths.

- **Research Gap**: Generalized methodologies for leveraging platform-independent models (PIMs) in the migration of legacy ERP systems to cloud environments are not sufficiently explored, particularly in the context of long-term operational sustainability for SMEs.
- **Research Question**: How can PIMs be applied to facilitate the seamless migration of legacy ERP systems to cloud-based SaaS platforms, minimizing disruption to core business functions?
- **Hypothesis**: Utilizing Platform Independent Models in ERP migrations significantly reduces migration complexity and risks while preserving essential business logic.

Own publications behind this chapter

Before moving on to discuss the topic of this chapter, let's summarize the own work behind. There are two articles which are corresponding:

- István Orosz, Tamás Orosz, Software as a service in cloud based ERP change management, 2017 IEEE 11th International Symposium on Intelligent Systems and Informatics (SISY 2017), Page(s): 000181 - 000186,
- István Orosz, Tamás Orosz: Code reusability in cloud based ERP solutions. Proceedings of the IEEE 21st International Conference on Intelligent Engineering Systems (INES 2017): Larnaca, Ciprus. IEEE, 2017. pp. 193-198

Three abstraction layers were successfully identified for SaaS implementations from the MDA point of view. The requirements for the system in a high level could be specified by

CIM. PIM can describe the operation on a platform independent manner, while hiding the technical operations of the platform. PSM defines the platform specific model, then adds details of the low-level platform details. MDA can map one single PIM in the cloud to different PSMs, as the transformation goes from a platform independent view to more platform specific mirrors of the solutions.

Any UML (Unified Modelling Language) description was used to describe the changes, as being industry standard. With the aid of UML, all the CIM, PIM and PSM layers are portrayed. This paper will demonstrate examples from Dynamics 365 Finance and Operation deployed in Azure cloud and will describe the transformation with the already existing on-premise version of the same ERP system hosted in datacentre.

The CIM model definition has to be extended with the suitable property definitions, which have to be exactly matched to the requirements list. This requirements list comes from the catalogue of the business processes has to be matched against the requirement model of the CIM without losing any valuable details. When the right requirements model is situated in CIM, it is able to identify what kind of requirements has to satisfy that code representation in PSM, and how can the code elements be identified, those which can be stayed untouched after the transition into SaaS model? The next chapter shows an elementary example, where the suitable PIM model could be built up and attached to the PSM model through the necessary transformation tool.

When doing a first-round analysing of the current codebase in the on premise version, a small part of the UML class schema looks like this:

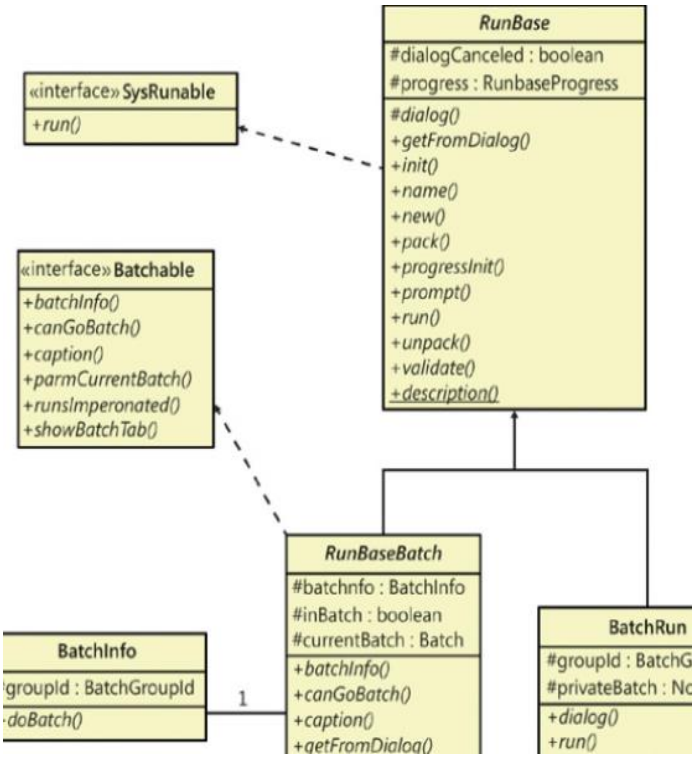


Fig. 3. Class schema of the RunBase class on Premise PIM model [23]

The UML model above shows a part of the entity-relationship data model, with all the elements of the shared and private objects. The model also contains some of the code classes, interfaces, with the operations, attributes, return types and parameters types. This PIM model contains enough information to decide the question: how can the two different object models be compared during the upgrade process, are they compatible?

While the code upgrade process runs, the currently existing business processes are only in focus for refactoring, the new ones are certainly out of scope. Newly added features, which can replace the old ones are also not touched. The refactoring process only takes care of the common set of business requirements, so this also decreases the number of possible objects. When the whole object hierarchy diagram is ready in the UML format, all the objects are described by the state of the object attributes, and the behaviour of the objects are described by the operations. These objects are working in association and dependency on each other,

interacting with the other objects in their dependency graph. Aggregation and composition represent a stronger level of association, while objects owning parts from other objects. Composition is a one-step higher-level relationship than aggregation, which means the whole part relationship.

Business Process Upgrade using PIM Model based approach

In order to present the possibility of reducing the unnecessary code changes during upgrade from on-premise version to cloud-based SaaS version of the same software solution (only ERP systems are touched in this article, although the methodology can be adopted to other IT systems also), the previous example will be examined.

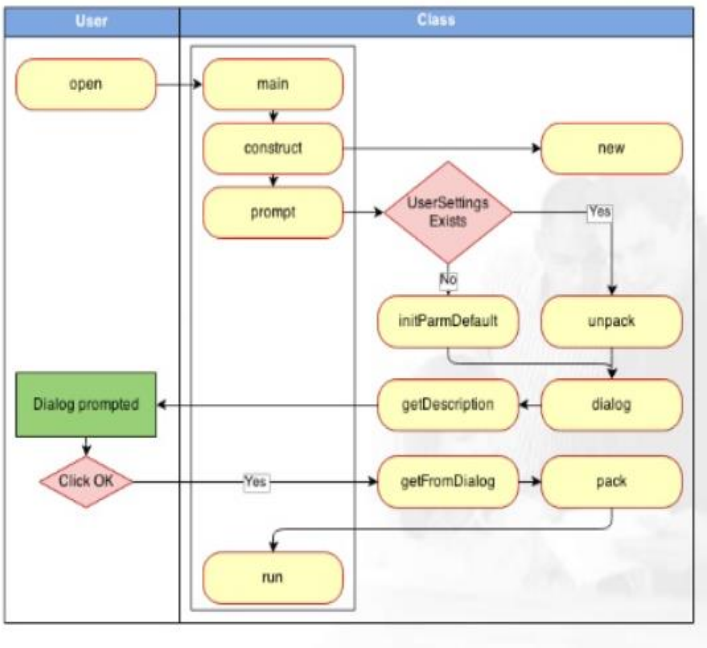


Fig.4. Runbase class operations in on premise PIM model [27]

The PIM for a SaaS cloud software solution in the UML format contains all the objects, their attributes and operations. Accessor methods for the attributes are not described in details, because these tasks will be part of the MDA mapping that is where these attributes will be translated and associated with the right accessor methods. The PIM model will give only a static snapshot of the system.

The other aspect, which could help in identifying the code parts which are responsible for implementing similar requirements are the attributes: what is the input and return data formats, how could the code interact with its environment?

All of these business requirements will remain the same while upgrading from the on-premise version to SaaS cloud-based version, so the first condition is fulfilled, as it is mainly a technical upgrade. As long as the business requirements are the same no new business logic needs to be implemented, therefore no code change must be done by the PIM model. According to the PIM model the parameters and the operations must be identical in both software versions:

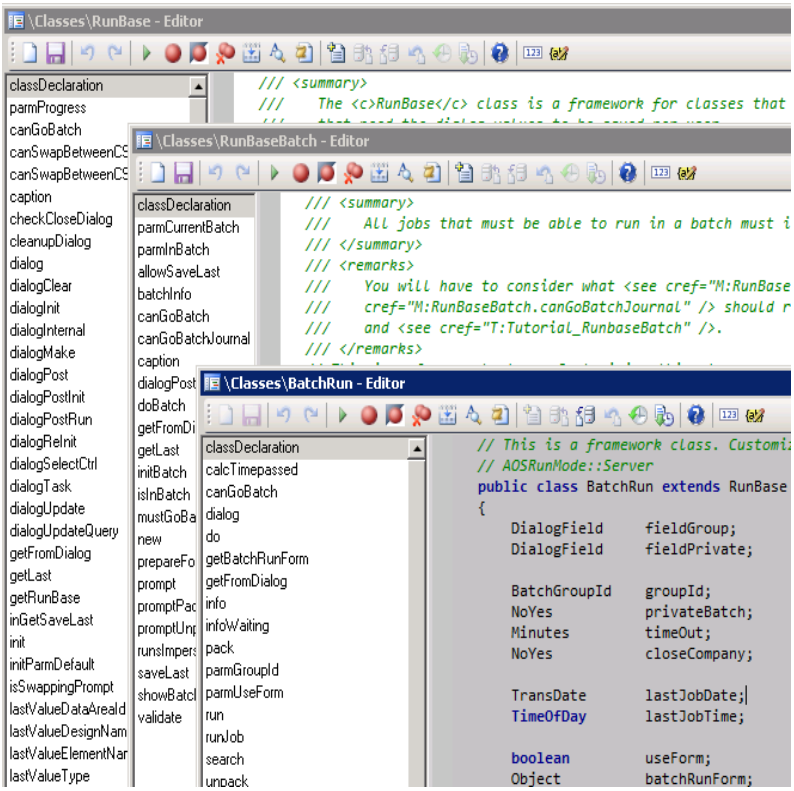


Fig 5. PSM implementation of the RunBase class

The business case behind the PSM implementation of the RunBase and its connected classes is that they are frequently used in running scheduled batch jobs, for example sending out sales invoices during the night. The most important business requirements are:

- Schedule a batch job per company, user, etc., this can be repetitious as well.
- Logging of the result messages
- Setting up the initial parameters on a user interaction dialog

- Output actions can be handled like emailing, format conversion, etc.
- Error logging and exception handling for later usage
- Resource handling through batch groups
- Run like a service
- High level of parallelism

Platform Specific Model can be implemented in several different methods, while this article does not scope all the different implementation strategies. The focus of the article is to find a proper method in which the Platform Independent Model can be used to specify the maximum number of common objects between the upgrade paths.

If the comparison is done for the two PIM models, they look very similar. Generally speaking, it is according to the expectations because all the business requirements remain the same, therefore the expected result is to get similar operations models finally. The final and exact compare for all the couple of thousands of upgradable objects is done automatically.

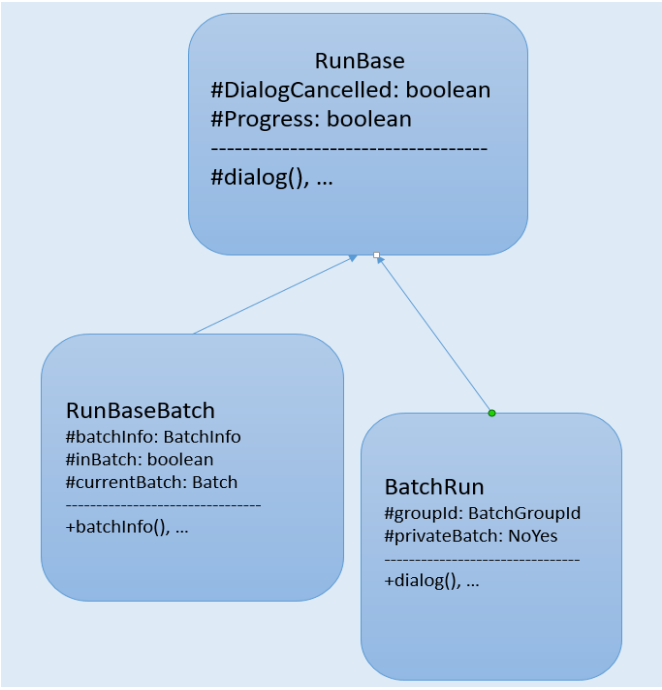


Fig.6. Runbase class operations in cloud base SaaS PIM model

This example shows that the PIM model contains enough additional information to make a proper decision about refactoring the objects. While the Platform Specific Implementation layer is untouched, the implementation can be done in different ways, as the process and the used models by the PIM will not reflect any system implementation low level properties of the software solution.

As the SaaS model has brought a new abstraction layer into the operation model of cloud-based software, the question of refactoring becomes more important, because the core business logic remains the same (or very similar). This new abstraction layer separates the business logic from any of the transformation or infrastructure specific layers, and makes possible for the customers to focus only the business processes.

Although the infrastructure which lies behind seems to have longer lifecycle, with the invention of the SaaS operating model the software product achieves longer support and maintenance lifetime. This is because the basic business processes, like posting a transaction into the General Ledger or incoming invoice approval seems more or less stable in the past decades, they were parts of the basic elements of doing business even before the introduction of the Enterprise Resource Planning Systems. The SaaS operating model is called evergreen: this means it makes possible for the customers to always use the latest version of the software running behind, without bothering the underlying infrastructure. Which covers the following activities: middleware, applications runtime, virtualization, servers, operating system, storage, networking, security [31]. Not only the workload and the responsibility go to the right resources, but customers can finally deal with the only thing they can do much better: tailoring the business functionality.

The PIM model is only enough to make decision regarding the code refactoring if:

a) the object hierarchy is well defined,

b) and the requirements list is up to date according to the business requirements [28][29].

It contains all the necessary information about the behaviour the properties and of the code objects, therefore on the decision side there is no need to deal with the underlying layers and the transformations. The upgrade tasks from the on-premise version to SaaS cloud based version can use the same OS, virtualization technology and middleware, so the PIM->PSM transformation can be done easier, focusing mainly on the technology independent tasks.

Software as a Service in Cloud based ERP change management

The right structure of the full lifecycle release and maintenance management of an ERP system is a key point of every business entity. Business requirements are always changing, nowadays the change is towards agile methodology, including new role like DevOps in the scope. The previously used model which was widely spread in the last decades, was based on the waterfall-like development release stages where they formed the milestones of the release life cycles.

- **Research Gap**: While Agile methodologies are increasingly being adopted, their specific impact on ERP upgrades, especially within SMEs, is under-documented.
- **Research Question**: How does the adoption of Agile methodologies in ERP systems affect project success and efficiency, compared to traditional waterfall approaches, particularly in SMEs?
- **Hypothesis**: Agile project management in ERP upgrades leads to higher stakeholder satisfaction and reduced time-to-market compared to traditional methodologies.

Own publications behind this chapter

These two articles contain the base research which later become foundation for this chapter. Each of them reflects the different aspect of the same area, but all in all they provide a complex view of change management tasks inside ERP systems.

- Orosz, I. and Orosz, T. (2014) "Microsoft Change Management Applying Comparison of Different Versions", Acta Technica Jaurinensis, 7(2), pp. pp. 183-192. (2014)

- Attila Selmeci and István Orosz, Software as a Service operation model in cloud based ERP systems, (CS)2 - The 11th Conference of PhD Students in Computer Science, Szeged 2018

Those steps acted an updated version of the previous step for both verification and development purposes [20]. Cloud based computing disciplines shows the fifth phase of computing paradigms, from mainframes, personal, to client-server and web-based. Computational clouds can be deployed as public or private. The main difference is in the security, as private clouds can be accessed strictly by one organization, but public clouds can be access widely in general, e.g., Gmail.

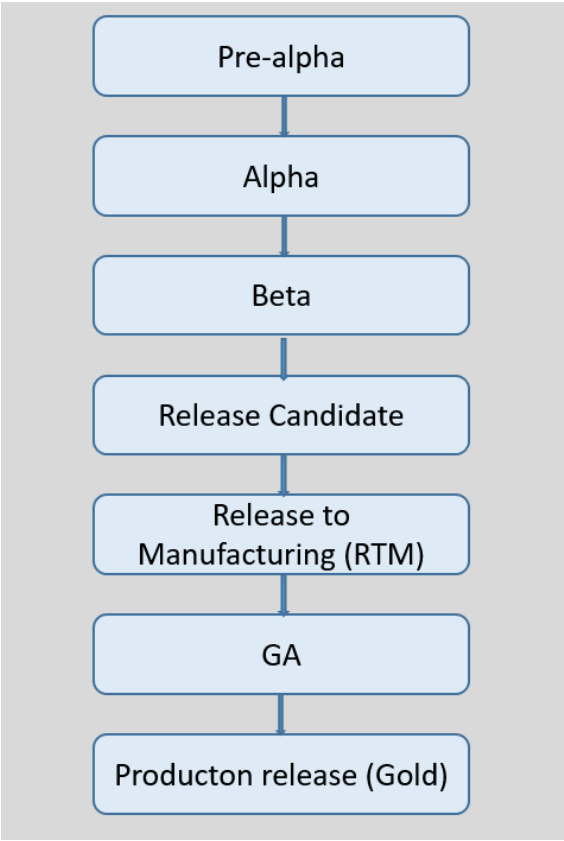


Fig. 7. Software development steps

Private clouds are the natural choice for hosting ERP systems which are operated by middles sized (or bigger) companies. Those activities which were usually done in the pre alpha stage, e.g. requirement analysis, software design, development and unit testing, will still remain the same. The fist software version where non-developer testing is taken place is the Alpha. Important to mention, that Alpha does not always contain the whole scale of functionalities. The first version with full feature set is Beta, but it can still contain serious problems.

Software products sometimes stay in this support stage for many years, which is because of the lack of project resources and limited support responsibility. Some software products are open to public in beta phase, which is called open beta, with the intention of getting more testing by a wider community.

Every phase after Beta can be considered as possible release candidate, therefore they have to provide the full list of functionalities, so this is the ideal part where dedicated testing teams can be used. In ideal cases this testing phase is already implemented in the client's future production environment to simulate the BAU phase as close as possible. One of the final steps is the Release to Manufacturing, at this point the product is digitally signed, so it is ready for live operation. General availability contains no additional software validation, the purpose of this step to prepare the product for the market availability. Final step is the Production release, where the final product, which reaches the market, is also digitally signed [19].

After the whole process is done, and the software product goes live in the production environment, the post go-live support is started. This can last for years, in some cases like multinational companies and banks sometimes more than a decade, until the product reaches its and of lifetime and is not supported anymore. This article focuses on the differences after the post go live support which are generated by the different behaviour of the cloud-based SaaS support methodology. This operational model offers a smoother method to implement significant changes during the lifecycle, without major effects on client side. In the current IT infrastructure nowadays, this cloud based operating model can be considered as one of the biggest changes.

There is a huge difference between the foundation of the two different software development and maintenance process, although it is worth to mention that in both processes both developers and software architects are able to use the same tools. For development and modelling purposes on low level the same elements could be used. Each approach focuses on the modularity, and even the same unit tests can be used in both cases, but the final goal is different.

Software as a Service key features, shortlisted (worth mentioning that sometimes it is referred as on-demand software, but that phrase does not cover every aspect) [22]:

- Software and its data are stored in one common place most cases in a cloud solution
- Customer facing infrastructure can be reconfigured with no hardware or datacenter side changes – it is just a change of resources attached to the customer

- Customers uses the latest version, provided as on-demand, no need to work on software life cycle steps and support needs. Evergreen option.
- Customers not charged on licenses fees but on subscription and usage base
- Infrastructure needs are present on not present on the customer side but on the vendor side
- Customers do not have purchase computing capacity in a datacenter – instead of they just configuring their resource needs
- Clients can access the software via a thin-clients solution, most likely a web browser

This article does not focus on Microsoft Dynamics 365 Finance and Operation on premise version, as it implements the original datacentre methodology. Microsoft has reengineered this into a SaaS model version, porting the application code in the described manner.

Software as a Service full lifecycle management model

This article focuses on the differences between the lifecycle models of the traditional datacentre solution to the SaaS model regarding mainly to code reusability and development process [28]. As a living example, Microsoft Dynamics 365 Finance and operation cloud version support and development lifecycle is in the focus, because it uses a SaaS operational model which is based on Azure cloud solution. Microsoft Azure services in the cloud is often referred as Platforms as a Service (PaaS), delivering solution stack as a service and computing resources all in one.

PaaS platforms are based on the basic idea that it enables third party solution delivery and implementation according to the SaaS model, meanwhile hiding all the complexity of the underlying datacentre and middleware architecture [21]. Even when the advantages of cloud computing are very clear, there are some points to consider before the decision on moving to cloud:

- SLA (Service Level Agreements) between vendor and the customer
- Security: if it is a public cloud solution, this must be one of the most sensitive points to decide about
- Question of privacy
- Outage and failover tolerance of the software
- Legal requirements, mostly when the software has to process transactions from more than one country (e.g. in Russia, you have to store local finance data physically inside the country)

Although all of these issues can be solved in a SaaS environment, data centre based on premise operating methods still a slight edge, not mention if security is a privileged priority.

An MDA (Model Driven Architecture) based development targeted for a SaaS cloud environment is able to improve the flexibility, robustness and lower to TCO (Total Cost of Ownership) of the software product. All the populated services in the development pipeline can be used in a technology independent manner [8]. As one model can integrate the whole business logic, which is independent from the layer of the technical solution means that a business requirement's core logic can be reused in future releases more easily, which improves the reusability and refactoring of the code [29].

According to the US National Institute of Standards and Technology, the exact definition of a Cloud based SaaS operating model is a service model, where the vendor applications run on cloud-based infrastructure, and this solution is offered to the customers as a service and not as a product [24]. Customers can reach their applications through a thin client, which is mainly a browser. Another property which is typical that customers cannot manage the configuration belongs to the underlying infrastructure layers. Based on the model view, it is interesting to see that SaaS models can sit on a higher level of abstraction compared to the development models where the software product has direct access to the technical layer. That means that cloud bases SaaS development, can separate the business logic and the underlying infrastructure from the unwanted effects of changes which can enhance the full lifetime of the software product.

A Model Driven Architecture based development for a cloud SaaS environment can use all of the populated services in a technology independent manner [25]. As the whole business logic can be integrated into one independent model this method helps code reusability and refactoring [29].

D365 as Software as a Service

Many of the involved and complicated software systems can be handled and used in the SaaS service model, which is a totally matter-of-course next step in the evolution of ERP software systems. The appropriate cloud operational model can be selected right to the business needs:

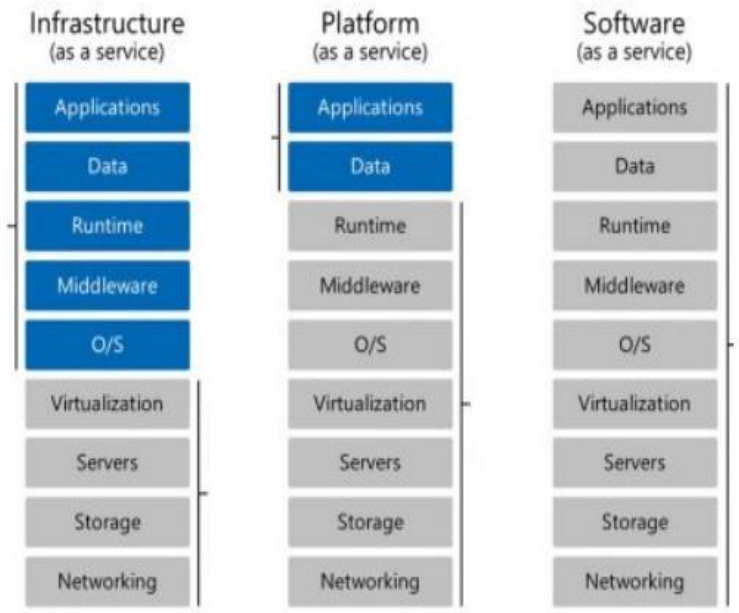


Fig.8. ERP cloud models [32]

These are the main cloud operational models for the Dynamics AX365, as for the first tier ERP systems also:

- IaaS: as this service level can take care of the virtualization, servers, storage and networking therefore it provides the minimal set of hosted services. Can be considered as the entry for the cloud-based solutions over the on-demand solutions based on datacentres.
- PaaS: one level over the IaaS, adding operating system, middleware interfaces and the runtime software modules to the offered services. Controlled environment offers for the date and its application for the customer.

- SaaS: customer has to do the business processes modelling and implementing its software solution, as the whole underlying cloud-based infrastructure is offered as a service. One can see it as the total inverse of the on-demand model.

The usage of different cloud operation models is a widely accepted commercial model, although this paper focuses mainly on the SaaS model.

Identifying the different abstraction layers for a SaaS model

SaaS is most appropriate choice to model and ERP system as another abstraction layer, then to develop the software solution for the computational cloud directly. This way we can decouple and distinguish the core business logic from the infrastructure providing layers, therefore enhancing the whole lifetime and heavily decreases the direct dependence from the future technology changes.

MDA based implementation of a cloud software solution based on SaaS can benefit from these properties as services are defined in a technology independent way. Moreover, the method of encapsulating and implementing the core business logic through a technology independent process can improve the future reusability factor of the code [30].

Previously there were three different abstraction level in SaaS cloud implementation were identified successfully. The computation-independent part of the software solution could be specified by the CIM (Computation Independent Model). It is able to describe the high-level requirements for the system. Medium high-level description of the solution can be gotten from the PIM (Platform Independent Model). It can offer a platform independent view of the whole operation lifecycle, meanwhile hiding all the platform technical details. Platform specific view of the solution can be got from the PSM (Platform Specific Model), which also describes the low-level details of the platform. The MDA (Model Driven Architecture) specific view can map one single PIM resides in the cloud to more different PSMs, while the process spreads from high level platform independent view to low level platform specific view of the solution.

Business Process Re-engineering using cloud-based SaaS approach

We need to examine the most sensitive question in order to demonstrate the possibility of using an ERP system as SaaS model: do we have to change the already existing business

processes or can we leave them the same, while only the underlying abstraction layers of the platform have to be changed.

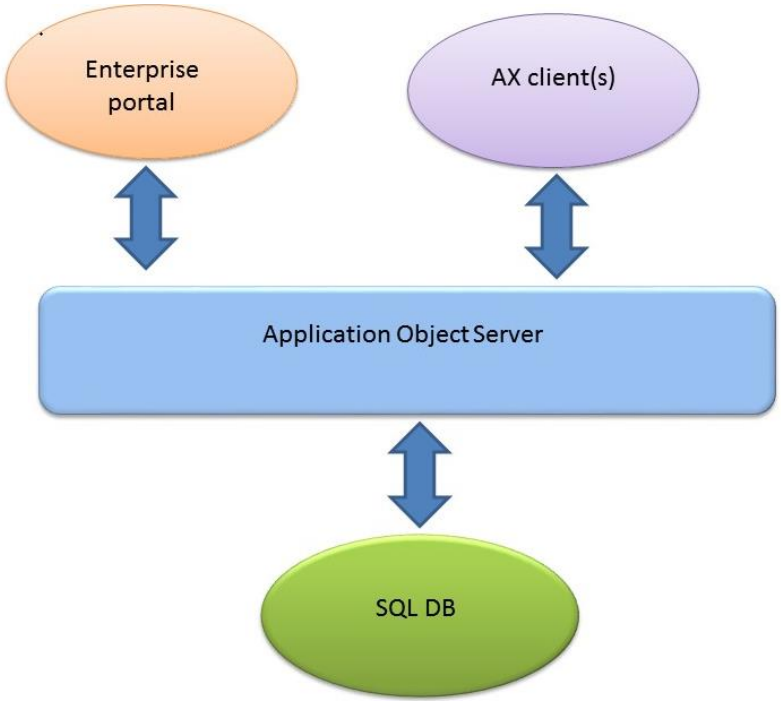


Fig.4. Three tier implementation layers of Dynamics AX on premise version

Need to take into consideration, that the original on-premise version of the ERP software was already a client server solution with three-layer, which also can be viewed as some kind of abstraction layers, the answer can be simple.

The final implementation of the Dynamics 365 ERP cloud-based SaaS version would include these abstraction layers:

1. Possible more PSM(s) containing one possible physical close implementation method, therefore more PSM(s) can be assigned to one PIM. More different PIM-to-PSM transformations are possible.
2. PIM layer containing the so-called core business logic in a way which should be fully independent from any technical implementation method.

3. CIM layer containing the software solution independent business model.

It has to noted, that these abstraction layers are different from the ones used in the on-premise model, which are based more on the actual implementation model. Therefore, direct one-to-one matching is not possible between the standard client-server model and the cloud based one program code.

One has to create these layers from scratch, as the original on-premise model could contain the business logic on every possible layer. For example, one could hardcode business level rules in a stored procedure on the SQL level. The following sub-chapters try to implement a possible way to identify and after decouple in every implementation layer the core business logic in the premise model. The most suitable tool seems to be the UML (Unified Modelling Language) description, useful for better er understanding of the object connections. All the CIM, PIM, PSM layers will be identified, and be described on how they could correlate with the original model, where it is possible. If the proper correlation is not possible, new model has to be built.

Cloud based ERP model as SaaS

The original implementation of Microsoft Dynamics 365 Finance and Operation was made with the intention to fit in the standard cloud based operational models. Therefore, to align with the Software as a Service methodology, it had to utilize the abstraction layers defined previously.

A Computation Independent Model

The CIM model would include one Use Case Diagram on the ERP system, which is aimed to model the business side functional requirements. These requirements must be transformed into the requirement model, but without losing any details. This Use Case diagram must include every piece of the final system functionality; therefore it has to specify the requirements which business has towards the software product.

B Platform Independent Model

The PIM model should include all the implementation steps and the platform independent realization of the CIM model, as a result it consists of objects and the connected relations. Basically, it needs the two most important properties of every object: the behaviour and the state of the object.

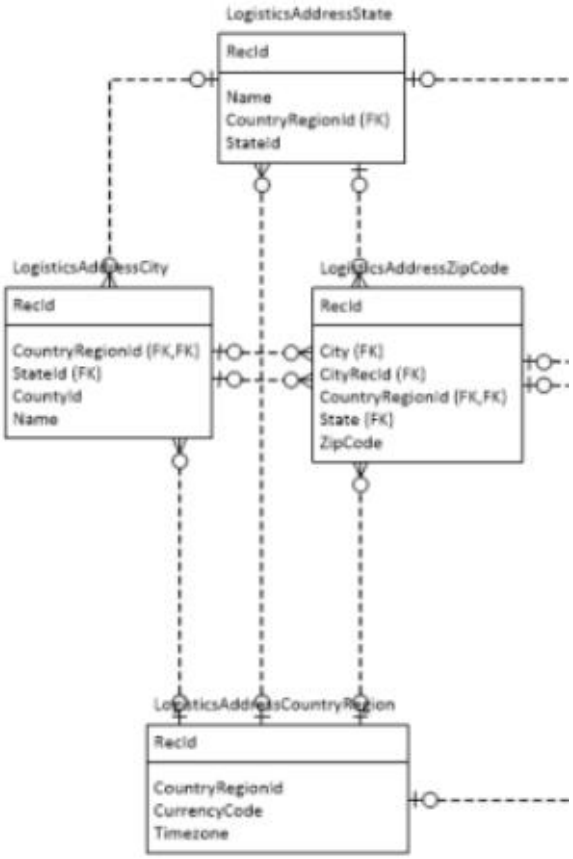


Fig.5. Part of the class structure of Dynamics AX [33]

Programmatically these objects are represented by classes in the codebase, and the interaction is described by the dependency between them. Those classes, the relations, the attributes and communication between them is shown at Fig. 5, one example only. The full attributes and connection could contain tens of thousands of classes. Although connections, attributes and classes may look low level technical enough, they cannot be considered as one-to-one correlation to the implementation layer.

C Platform Specific Model

Basically, the PIM model has to describe all the technical details, which are needed to specify the target implementation platform. This layer should include the and database specific details and the o ERP specific parts, together with the necessary connection towards third party applications.

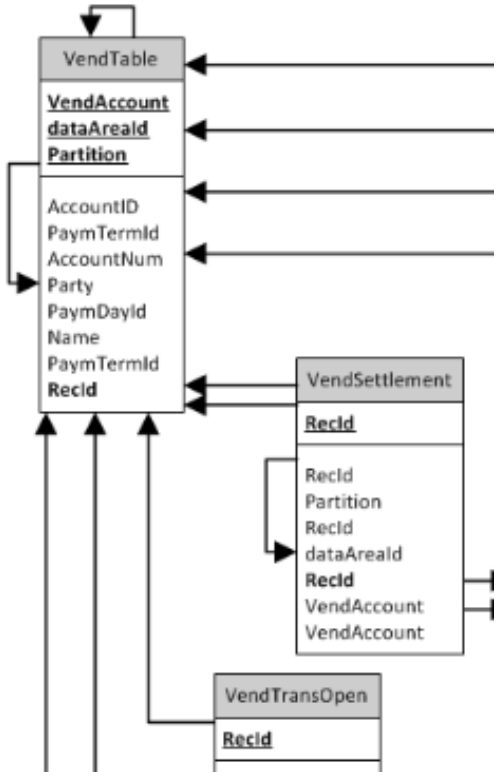


Fig.6. Part of the physical database model of Dynamics AX [34]

A PIM must have the specific transformation rules for a PSM, from the source (PIM) to the target (PSM). A cloud based PIM can be implemented by several PSMs, although all the abstraction layers have to protect the core business logic. In a result, all the functionalities must behave the same in all different platforms.

Conclusion and possible extensions

When analysing this new abstraction layer, the question of the performance overhead is valid and interesting task for the future work, just because of the fact that from now every operation must go through some parts of the cloud. For the first sight it seems a very limited added network traffic, latency and extra computational work, they are also added steps which did not present in the on premise version of an ERP systems. We need to take into consideration the overall solution security and data privacy. These issues are so serious, that till now this is

the root cause why a lot of costumers do not want to their ERP solution including their business data into the cloud [38]. These are the main questions which are worth analysing.

The fact that cloud-based ERP implementations lose some control over the business own data, compared to the datacentre solutions. But we need to take this control loss versus the evergreen property of the cloud based solutions. Needs to take into consideration, that the on premise version aften contains mixed type of technologies from the past, making the operational costs even higher. Theoretically speaking, the cloud based solutions are better planned and designed, making them more futureproof.

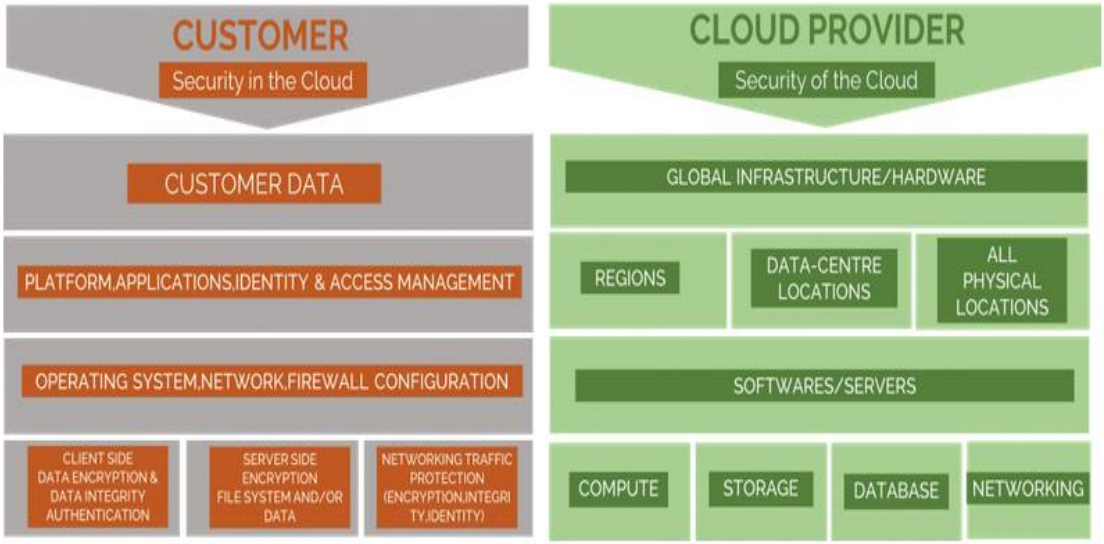


Fig.7. Customer and cloud provider security responsibilities [39]

Software as a Service was previously shown and analysed as a mainstream ERP approach method as of today, focusing on how to convert the standard on premise model into a cloud based one, the methodology used is based on identifying and creating different abstraction layers. For the future work, it is an interesting and valid question, how can one be sure, that the functionality is the same, covering the same business processes with the same input/output, presented in different PSM implementations of the same PIM model. Another interesting query, how can we compare the performance of the two different implementations, as they are operating on totally different platforms. The biggest Tier 1 suppliers of ERP systems (SAP, Microsoft, Oracle) all struggle with finding better approaches on the efficient

implementation, long term operation and handling of evergreen updates. This chapter highlighted some of the main topics for this challenge, while missing out a lot of others like security, infrastructure robustness, etc. Hybrid, public, private environments, data security and privacy are still heavy question to solve in cloud environments.

To follow the latest customer requirements, medium and small sized companies have to move their operation into cloud, while doing a full-scale business process reengineering (BPR) cycle. This will also mean that the change management processes will have affect not only on the daily business operations but possibly changes the actually used ERP systems also. Effective processing the high amount of data needs more effective processes, that is where big data algorithms, business intelligence came into sight – and they are heavily backed up by the cloud architecture. This will result even more new challenges in ERP system connected business environments. One of the key questions which if not in the focus of this article is security, handling sensitive personal data, and GDPR related areas. With the usage of machine learning it is possible for the first time in history to create personal profile so deep. ERP system usually contains so sensitive data for a company, that protecting them is a key point.

ERP Change Management Applying Comparison of Different Versions

Software change management methodology is always a complex area, even we are working within the same ecosystem. Business requirements are always changing, nowadays the change is towards agile methodology, including new role like DevOps in the scope. This section covers one of the most challenging change management tasks in a multinational company: ERP system change management, and tris to cover new aspects and new strategical approach to make it more efficient.

- **Research Gap**: Despite the growing body of literature on cloud migrations, there is insufficient research on generalized methodologies that ensure long-term operational sustainability during the transition of ERP systems for SMEs.
- **Research Question**: How can companies ensure sustainable long-term operations when migrating ERP systems, like Microsoft Dynamics 365, to cloud environments?
- **Hypothesis**: Migrating ERP systems to the cloud, utilizing frameworks like Microsoft Dynamics 365, enhances operational sustainability through evergreen system updates and reduced maintenance requirements.

Own publications behind this chapter

These four articles contain the base research which later become foundation for this chapter. Each of them reflects the different aspect of the same area, but all in all they provide a complex view of change management tasks inside ERP systems.

- Selmeci, I. Orosz, Gy. Györök, T. Orosz: Key Performance Indicators Used in ERP Performance Measurement Applications pp. 43-48. , 6 p. In: Szakál, A (szerk.) 2012

IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics, SISY 2012, Subotica, 2012, September, 20-22

- István Orosz, Túri Balázs, Tamás Orosz, Selmeci Attila, Local Governments-specific BPR mini-project with SAP applications, SAMI 2012: IEEE 10th International Symposium on Applied Machine Intelligence and Informatics, Page(s): 119 – 123
- István Orosz, Tamás Orosz: Business Process Reengineering Project in Local Governments with ERP. BULETINUL STIINTIFIC AL UNIVERSITATII POLITEHNICA DIN TIMISOARA ROMANIA SERIA AUTOMATICA SI CALCULATORAE 57 (71) : 4 pp. 253-258. , 6 p. (2012)
- Selmeci Attila, Orosz István, Orosz Tamás, Györök György: ERP change management for innovation and sustainability applied to User Interfaces. In: Szakál, Anikó (szerk.) Proceedings of the 2012 IEEE 16th International Conference on Intelligent Engineering Systems (INES). Lisbon, Portugal, (2012) pp. 341-346.

When dealing with code management in the whole change management method, it somehow differs from the traditional methods. Business processes, like booking an entry in the General Ledger (GL) are not changed in the last decades, but the technical implementation has changed a lot. The whole change management process has to be a managed one, which leads the whole organization towards business process reengineering. These steps will form a project, which takes place in the organization when it is finally introduced and approved [77]. Usual side affect of the change management process is the organizational change. The general purpose of a change management project is always to generate minimal impact to the organization, avoid and solve concerns. We can face several different types of change management:

- Changing the behaviour of a personnel
- Strategic and mission changes
- Operational and structural changes
- Technological changes
- Changing the behaviour of a personnel

The changes came from the ERP side are generally operation, structural and technological ones. Organization change management process always starts with a detailed landscaping of the current status, with a strong focus of the need and the ability to change. In parallel, the objectives, content and process should be defined together with the government and board structure of the process. Focus must be on the following areas:

- Performance KPIs like financial measures of the company
- Operational efficiency
- Leadership commitment
- Communication effectiveness

The perceived need to change leads to plan the right strategy in order to avoid failures both on budget and project goal level. A successful organizational change management project has to have focus on the following areas:

- The right management for proceeds, realization defining measurable participant efforts, including creating business case for their achievement. Also monitoring risks, dependencies, costs, return of investments. Cultural differences also must be taken into account.
- Use of effective communication patterns, which targets stakeholders for reason of change (why we do it?), benefits of the successful transformation (what is it in for you?), details of the change which is important for the transparency (when? Who is in charge? How much will it cost?)
- Effective training including “train the trainers” methodology which strengthens the attachment to the core project. Skills must be upgraded also for the future shape of the organization.
- Understand deeply the resistance from the employees (fear of the changing future), align them for the strategic direction of the organization.
- Provide personal coaching if needed to alleviate change related fears and concerns at the beginning of the project.
- Constant monitoring of the execution and implementation, constant fine tuning of the process [78]

These can be fit into the following steps:

- Diagnostics
- Analysis
- Design
- Development
- Deployment
- Operation

The future of the company structure has to be planned according to the following principles: integration, autonomy, segmentation. This is the optimal flow of change according to a Dynamics AX workflow process:

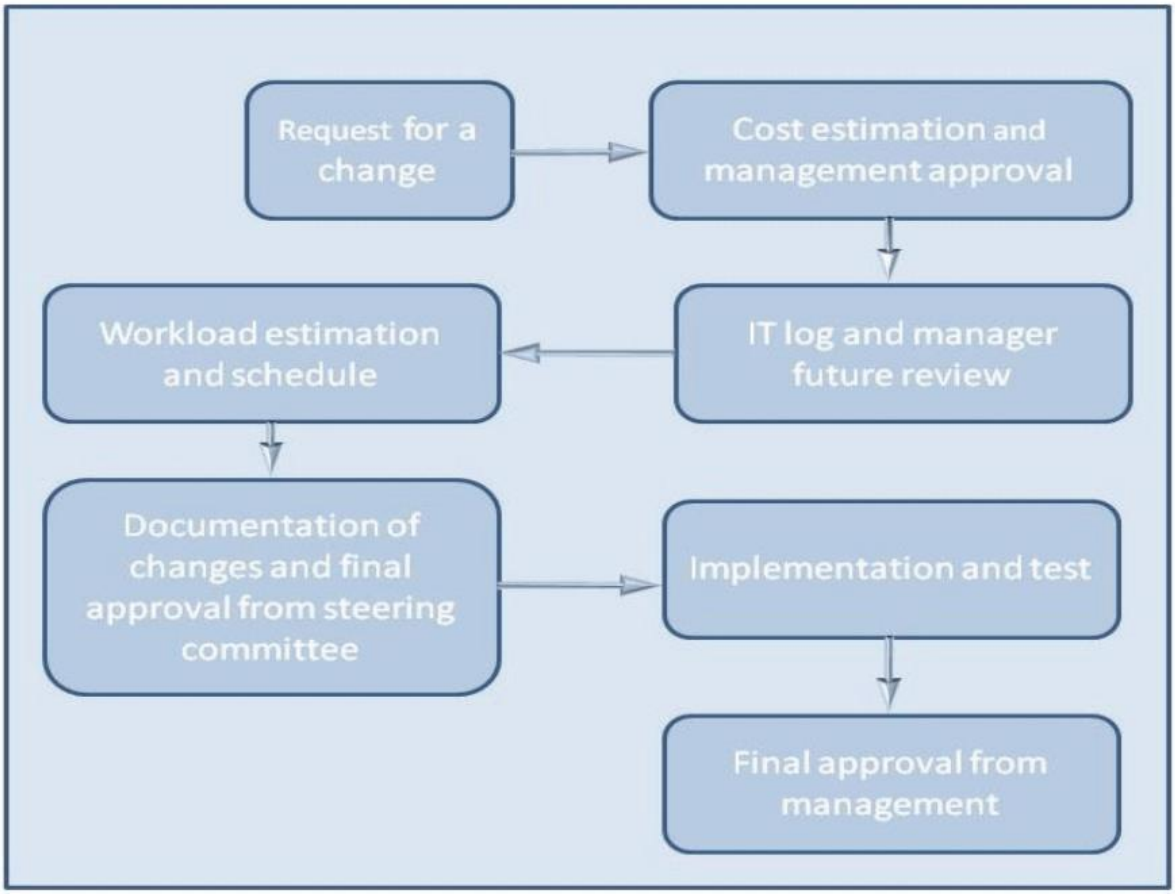


Fig.8. Technological changes flow [3]

It is also important to create a clear governance over the organizational change process, which is aligned to the project goals and objectives. The model used for the consultation governance also has to take into consideration the organisational culture, the values and the principles along which the company is driven.

In order to overcome needs of a BPR, Dynamics AX has implemented a methodology which can speed up this complex process, which is based on instantly implement the principals which were underlined by the business process recipes. Sustainability is one of the key factors of each model change processes, as it is a vertical factor covering all the basic steps of an ERP project lifecycle.

Optimization and upgrade processes are different in a was that business processes are intact in these project types. Sure Steps methodology covers the previously mentioned six main steps and two additional ones. These are the followings in details:

- Diagnostic phase analysis the customer process in high level. This step focuses on the project initialization, setup the project plan and the scope definition.
- Analysis has to identify the used business processes and documents them in high level, it is useful to have external consultancy for this step, as they can see these processes with fresh eye. The main goal of this step also to understand the business needs and do some initial modelling. Microsoft offers Sure Step Business Modeler for this purpose.
- Design steps has to find the most suitable way how to implement the business processes in the ERP system, in this case in Dynamics 365 FO in the cloud. Until this steps there could be multiple solutions for one topic, but this step has to identify what is the most suitable for the project. It is also the place for building Proof of Concept (POC) models also.
- Development step is the place for doing the code implementation, and unit, process and regression testing also. Data migration from legacy system has to be done as well, together with implementing the previously designed security structure. This steps end, when the regression testing is successful on the migrated data.

- Deployment step is the place for deploying the ready solution into the customer's own company cloud stack. This is also the phase for some extra testing mainly focusing on system level, like security testing, user acceptance test and load testing.

- Operation steps is the place when the whole system is deployed and tested ready in the customer cloud stack, which is in this case Azure DevOps. As D365 Finance and Operation has its evergreen feature, is can be considered as the final place for the software solution.

Sustainability and agile business change management

Today's software project management seems to leave behind the waterfall method, which was an appropriate project methodology when the project is much shorter than the organizational change period.

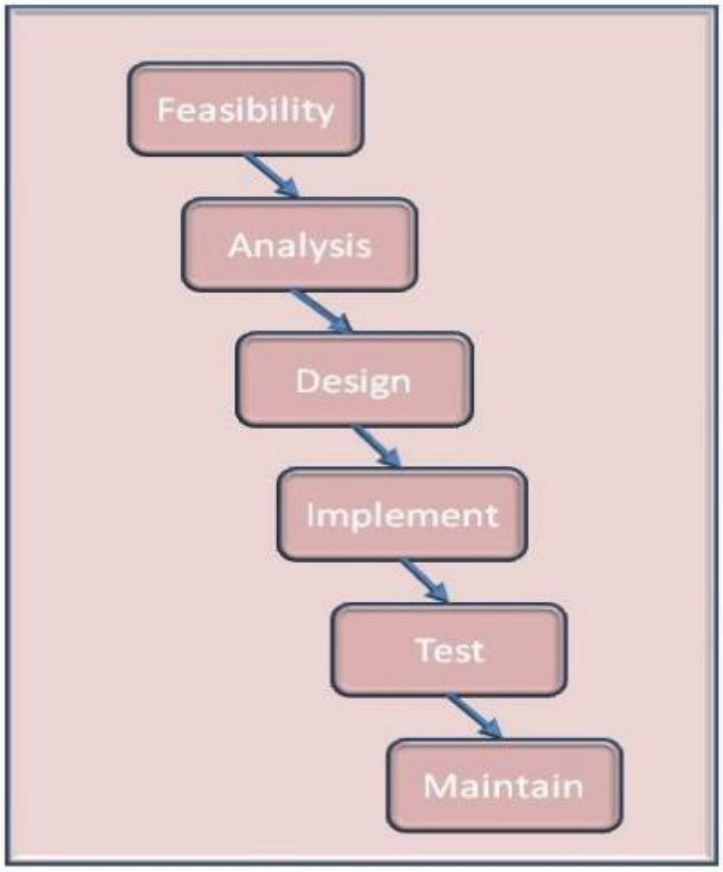


Fig.9. Waterfall methodology [3]

To put it into other words, if the organizational change is prevalent, it is easy to run into a situation when the project lasts longer, which means that the project might fail as a result of the changed requirements. Another issue is that the project steps have to follow each other in strict order, without no real possibility to step back or repeat a step if the result do not meet the quality bar. The early phase issues, like the ones e.g. at analysis phase, can affect the whole project for the implementation phase also and can jeopardize the project success. Typically,

waterfall is a good methodology for implementing a business change in a little changing environment.

When business has rapid need for change management, it is advised to use another type of project management methodology, the one what is in scope is the Agile implementation method. Typically, it gives greater control over the final product, as the direction of the development can be changed rapidly if there are new requirements or the older ones has been changed.

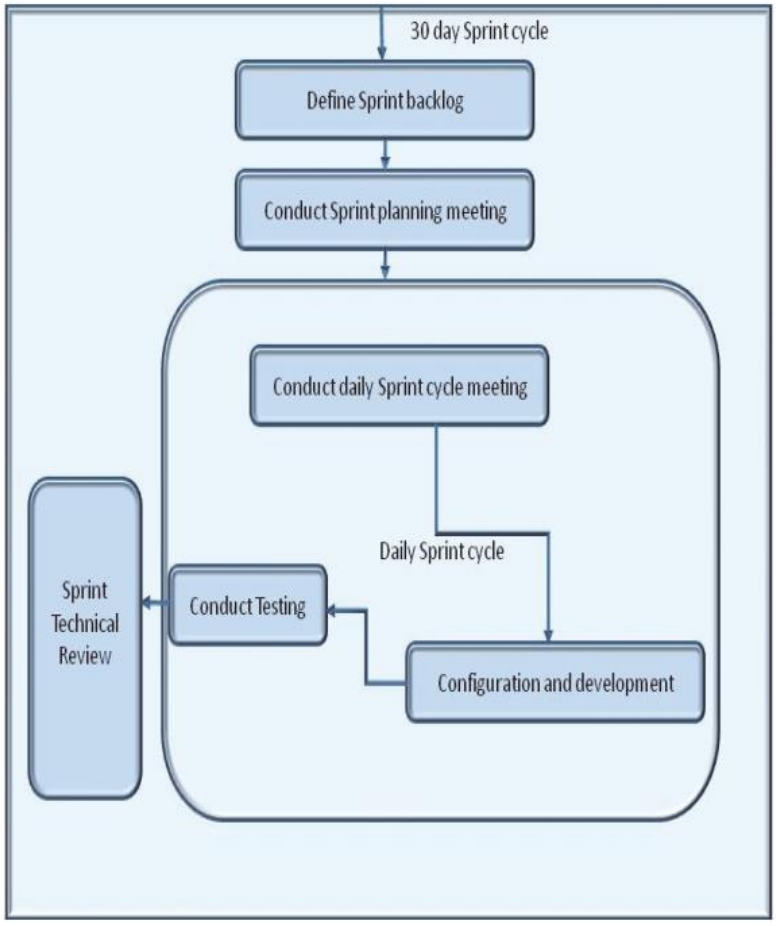


Fig.10. Spring cycle in an agile implementation [3]

Business change management in agile methodology

It tries to implement the end product similar to trial and error – tries to deliver in smaller chunks in repeated cycles instead of delivering the whole product in one monolith deployment step. If the result is not satisfactory to the customer, it is easy to run another development cycle, which also means the customer or end users are got involved into the project deeper, which also means that they understand the project goals deeper, which can result in better delivered quality and reduced costs. Worth mentioning that agile methodology has to come with strict time and material cost management, as it is easy to override both time and project costing.

Some explanation to Fig.10:

- Solution backlog, which contains the feature list of the proposed solution.
- Release backlog, first identify than prioritize all the features which belong to the sprint (usually 30 days), also with the time estimation for all the features (story point).
- Sprint backlog, which is the breakdown of the release backlog to feature priority and time estimation, sometimes divided into daily sprints.
- User story, which describes the business functions of a feature, also can contain user roles and test cases.
- Defect report, which identifies and reports bugs found during the implementation phase
- Daily stand-up meeting, which is a short- and effective-day starting meeting for the development and functional team

The usual agile implementation has to start with a detailed business process analysis followed by the high-level fit gap analysis. The output of the fit-gap analysis, the so-called Solution Backlog, is the key document for the project. Unless other project documents, mainly on the waterfall model, it is a living and always changing document.

These initial project steps could look familiar from other project management tools, but the implementation phase looks different: instead of doing the Design and Development in big monolith steps, agile methodology executes them in sprints (most often 30 days). It has its own document, the Sprint Backlog, which is derived from the Solution Backlog. Each development task is put into smaller pieces, usually maximum 16hrs of pure development work.

Workflow processing in the cloud ERP world

The business world as we know them today would hardly even exists if ERP system would stop functioning from one day to another. These are complex ecosystems, often consist of hundreds of sub systems, which works parallel in close connection to each other. The business processes which they represent are aligned with the company goals, therefore they have to be well organized and should provide consistent result even in different conditions. Automation is a key property when we try to improve these processes on management side.

- **Research Gap**: There is a lack of comprehensive studies that address the integration of workflow management systems (WFMS) across multiple ERP platforms, particularly in cross-platform automation scenarios for SMEs.
- **Research Question**: What are the challenges and benefits of integrating WFMS within ERP systems, particularly when operating across different cloud environments?
- **Hypothesis**: Integrating workflow management systems (WFMS) within ERP systems increases operational efficiency by automating repetitive tasks and reducing manual intervention.

Own publications behind this chapter

These four articles contain the base research which later become foundation for this chapter. Each of them reflect the different aspect of the same area, but all in all they provide a complex view of change management tasks inside ERP systems.

- Attila Selmeci, Tamás Orosz, István Orosz Workflow processing using ERP Objects. ACTA CYBERNETICA 22 : 1 pp. 183-210. , 28 p. (2015)

- István Orosz, László Szívós, The Role of Data Authentication and Security in the Audit of Financial Statements, ACTA POLYTECHNICA HUNGARICA (2014)

- Attila Selmeci and István Orosz, Workflow processing using SAP Objects, (CS)2 - The 8th Conference of PhD Students in Computer Science, Szeged (2012)

- István Orosz, Tamás Orosz: Change management and workflow processing using Dynamics AX objects. 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics, SISY 2012, Subotica, 2012, September, 20-22 pp. 249-254

In general workflow is considered as "the computerized facilitation or automation of a business process, in whole or in part", this definition is taken from Workflow Management Coalition (WfMC), which is a non-profit international organization, dealing with the industry standardization of best practices for workflow management [55]. According to this definition, workflow will be considered as the automation of a business process which executes work tasks in a sequence governed by conditions, while information, documents and subtasks can be circulated between participants (human or machine), triggering actions while obeying business rules.

Some experience is required while working with workflows in business applications. This part of the thesis focuses on different capabilities and approaches of these business solution to demonstrate the different levels and methods of interoperability for workflow management systems.

1. **Participant**: an entity in the workflow with a pre-defined role on his tasks.

2. **Activities**: these are called tasks or workflow steps, if the workflow is automated. Workflow can be seen as a common set of the process flow and its related data which are joint to the process, and the execution steps together. Activity is the most basic element of the workflow process; it represents a logical step in the process chain. Worth mentioning the each of these activities can be processes manually also. These manual steps are often representing a decision point, rejection or acceptance criteria where human interaction is needed. Machine executed steps are usually processed in the background, like sending a notification mail or posting an invoice to the general ledger.

3. **Business process**: after defining the participants and activities, we can define and build up the business process, guiding for a company objective. The detail level of business process definition is not general, complex processes can be split into sup processes, and one can create more complex processes with the aid of pre-defined simpler ones.

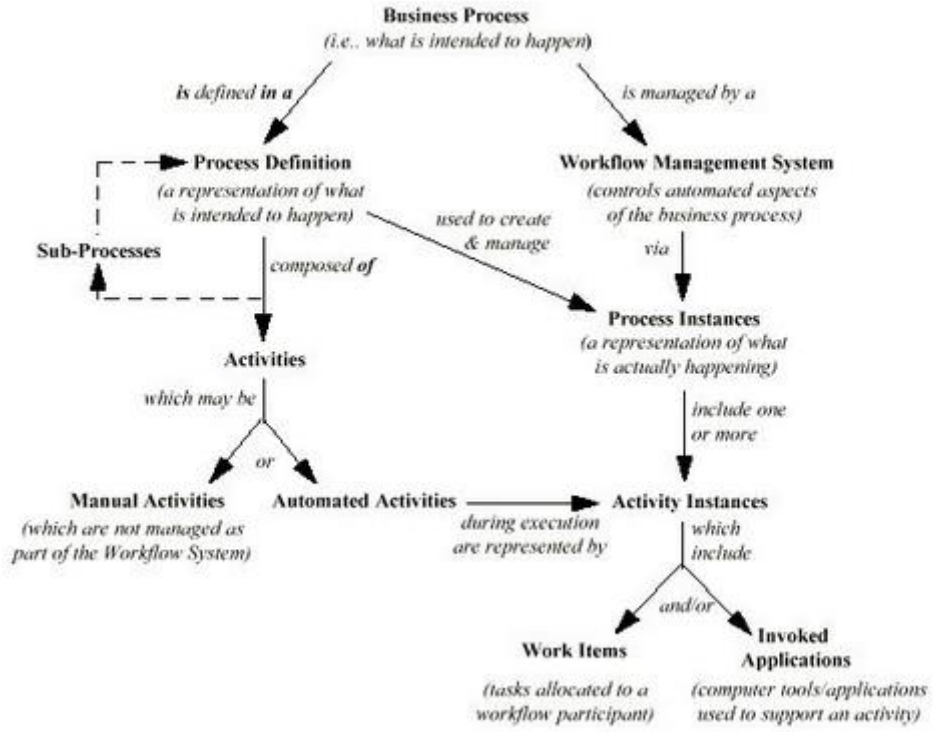


Fig.8. General concept of workflow [1]

As the business processes usually contain decision point, so do the corresponding processing chains also, as it is very rare that a business process is only a simple chain of activities. These controls point later will create branches in the workflow, making it a more precise mapping of the process. The single workflow is a theoretical process chain representing the business process with execution mechanisms like parallel or sequential execution, choice, iteration. It is the Workflow Management System (WFMS) which makes the workflow processing live, it is the technology enabler behind the theoretical model. One can plan, create and manage the workflow execution with the aid of the WFMS, these are the common features besides the execution of the process chain. The Workflow engine is responsible for making the dialog with

the participants, while contacting with other technology or application parts. WFMSs generally contain these five parts:

1. Client application, a graphical user interface (GUI) for human interaction
2. Process definition tool
3. Administering and monitoring tools
4. Invoked applications, which executes the real business process activities
5. Interface for external workflow engines

On the market there can are standalone WFMSs and ERP system built-in ones also available [52], both solutions can have its advantages depending on the project needs. Business applications and its automated execution have been evolved in the past parallel, reaching a point where ERP system can be considered a workflow implementation. This triggers the need for communication and cross system workflows, and later part of this chapter will show the enablers on technology side. Invoked applications and interfaces towards external workflow engines are also in the scope of this chapter, as it will also introduce the mechanism in depth for today's two popular built-in workflow systems from the ERP world, SAP and Microsoft Dynamics D365 Finance and Operation. Analysis was focusing on the cross-system possibilities, where different process steps can be executed in distinct business applications. SAP cooperation with Microsoft client product is out of scope, as in this case this is not a real cross system execution. Many of the leading description languages and protocols are of the scope also, like Business Process Execution language (BPEL), Business Process Modell and Notation (BPMN), as we lack spread.

History and ecosystem development

The introduction to workflows in the previous chapter showed that the main driver for them is the automation for process execution.

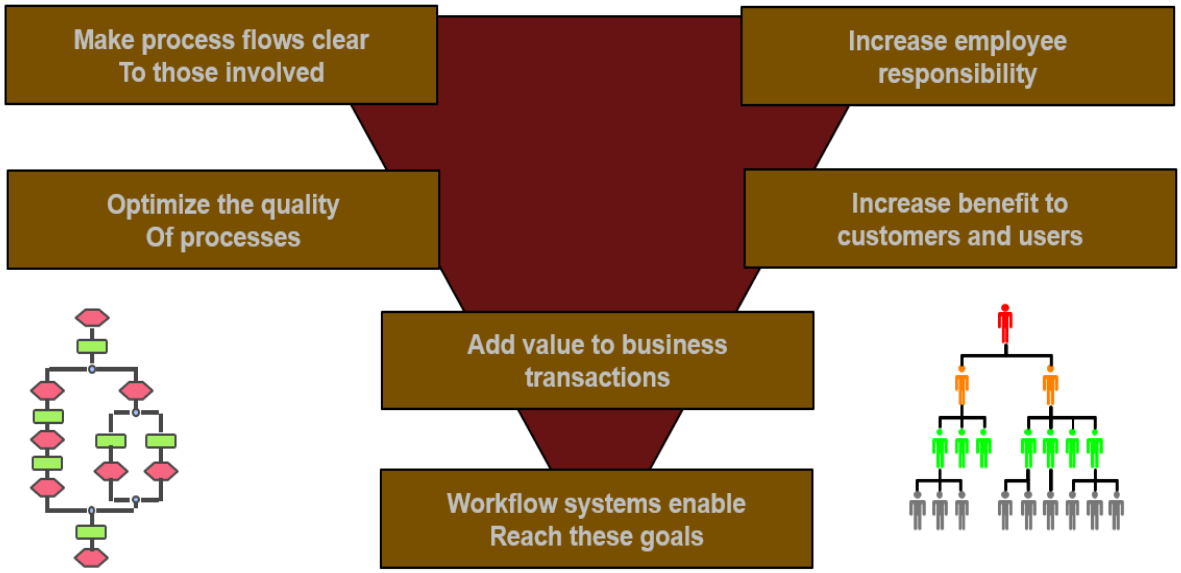


Fig.9. General concept of workflow [1]

The workflow benefits are valuable for employees and management level also. Employees can get less administrative workload as subprocesses can be executed in the background, necessary information circulates faster and via the WFMS they can get it faster. In parallel management level can have more control of the information flow, control deadlines and dates better, which overall means better cost control also. Not to mention the secondary affects, which means more flexibility in process changes, better high-level overview, and a step advantage via using latest technology. In general, these are the high-level driver for the implementation of a WFMS [41]. These are the last couple of decades in the development of the workflow systems:

- In early 90's users start to align their thinking like workflow. This is still the age of standalone ERP application running in datacenters, with no proper business rule formalization, so description languages. Only the end user knows the chain or steps, everything is executed manually.

- Sometime around the middle of the decade formalization of the workflow started to create an initial model. While still there is no task management, users are executing steps manually and decision points are still evaluated by them. With the existence of the model, workflow processing makes a step ahead: similar processes are executed similarly, first steps of reusability.

- Second part of the 90', the appearance of the task inside the workflow. Different participants can execute the same task in the same sequence. Also, the different types of the participants appear in the ERP system: approve and requester. Different user roles are present in the system, users are divided into different roles which allows them to process the same workflow for e.g. posting a sales order. First real benefits of the workflow appear as processes can be executed in a shorter timeframe.

- Early 2000 years, so far all individual steps were clearly distinguished from each other, triggering different procedure from the back end WFMS. These services are called from the central workflow engine, which takes over the task from the previously mentioned workflow systems while containing the elements of the user interface. These workflows running in the processing system are service oriented ones, and the main difference from the earlier adaptations is that it does not depend from the backend system.

Therefore, the service is independent from the backend, only the interface is published. Participants use processing engine for decision making, but they do not execute backend tasks directly. All of them are processed via services. This kind of workflow processing needs a standard communication between the available backend systems. Enterprise Application Integration (EAI) communication standard serves this purpose, providing a common platform for integrating backend systems with different technologies. Data exchange is monitored by this layer with higher level, offering multiple distribution,

conversion and filtering. This kind of Service Oriented Architectures (SOA) have to offer external and internal services (often referred as Web Services) using UDDI (Universal Description, Discovery and Integration), WSDL (Web Services Description Language) and SOAP (Single Object Access Protocols) amongst others. In the middle of the SOA environment lies the central workflow processing engine, providing together the system components and the user interface parts also.

- Around 2005, workflows start to use composite application and services.

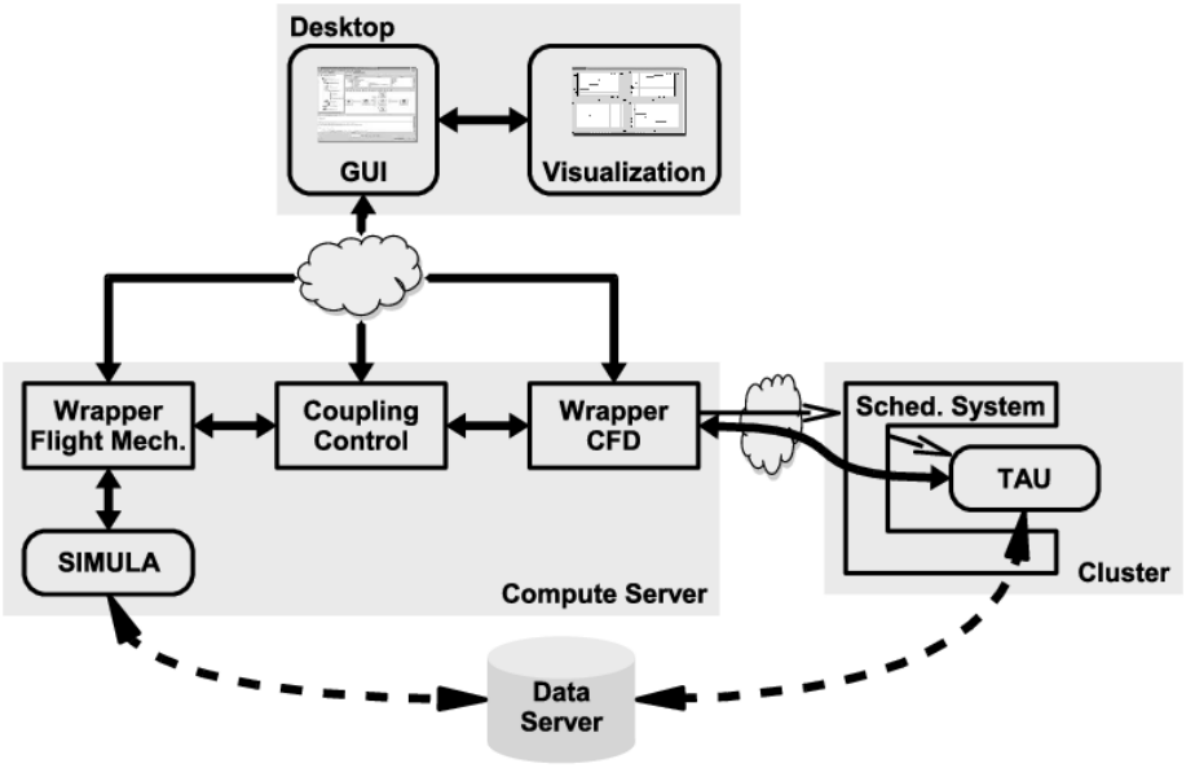


Fig.10. General concept of composite workflow [56]

Till this time SOA based workflow processing was able to provide UI and process flow executing while connecting different backend processing system connected with EAI. Another approach to use SOA is to consider it as a composite application, which only utilizes the GUI and the control mechanism, but the procedures call services which are executed according to the M-V-C (Model-View-Control) paradigm. Composite workflow systems are a step forward, they are right before the processing engine layer versions. The later concept brings another abstraction layer between the processing engine offering the GUI and the backend system offering processing services. This

middle layer is often called Enterprise Service Bus (ESB), with an extended EAI processing engine. ESB manages the whole service repository, and it can define new complex service from the already existing ones, which are called generally composite services. ESB has to be built up in a robust and reliable way, as outages of simple service could jeopardize the composite ones.

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

Fig.11. General concept of composite workflow [1]

- Workflows driven by human resources, from 2008: between the composite workflows and cloud-based solutions, this intermediate step is characterized by participant actions which are extended with some preparation and execution right

after the human interaction. This step makes change handling easier on organization or business level much easier. Process engine contains all the business relevant activities. The company with the workflow environment is able to react to the business changes even if there are no need for deep changes in the computational logic, data model, etc. The kind of flexibility can be achieved by grouping and separating the functions between the ESB and the processing engine.

- Cloud based computing workflows, from 2011: the last and currently up to date state of the workflow processing is the cloud-based operation model. As the physical separation of tasks and enablers took place in the previous chapter, as process engines can build a separate environment with their all capabilities and responsibilities, connected to the back-end systems while offering external, internal and composite services. From the SOA point of view, the physical execution system and the model is not relevant anymore, business want to control only the services. Software as service operation model seems to be a logical choice as the next evolution step, as it separates the offered services from the undelaying business logic and implementation layers. It improves the agility of the business as they are no longer tied to the programming model, they can interact only with the offered service.

Subsystems and ERP base workflows.

The main driver for using workflow comes from the business needs: there is a constant need for having cheaper processing through automatization of the possible steps. One of the most important aspects is to reduce the resource needs for repeating monotonous tasks which can be replaced by workflow steps. Other key driver is the constant need for cost reduction. By using workflows, expensive human resources can be freed up from the daily routine and divert to more creative and agile way of working. From around the early 2000's years business needs focused one the paper-less document processing and more and more automated part in the process control. That came along the fact that automated parts can be done in higher quality, e.g., sending of 200 sales invoices in mail is not a real challenge for a skilled office clerk, and can be considered as a waste of valuable working hours. Since these years, many process vendors already have some kind of a process automatization solution, but they were acting like separated islands in the whole IT landscape. Many first attempts failed for integrating these separated solutions, but the emerging needs of the ERP world made the final push.

ERP solution provider started to create their own industry standards for implementing workflows engines into their business flows, each Tier One provider started with their own standards (Microsoft, Oracle, SAP, etc.). Out of them, two major solution provider's internal workflow processing will be introduced in the following chapters. It is interesting to analyse how undelaying hardware, operating system and lower layer software solution changes affected the ERP system evolution, which we will also take a closer look [42]. After leaving the integrated ERP era, central workflow systems started to expand outside the core ERP system and became to act like an integration layer controlling information and task flow between other system too. Supply Relationship Management (SRM) is a good example for this natural evolution, the business process affects both system:

1. Purchase requisition approval takes place in the SRM.
2. As ERP is responsible for the stock and the purchase order administration the real reservation of the stock is taken place in ERP.

3. Then the delivery dates, routes, prices and other information came back from ERP to SRM, inform the purchaser.
4. Real invoicing are taken place in ERP, as it is responsible for purchase order ledger, and getting it aligned with the general ledger for audit reasons.

In such complicated processes the synchronization between the workflow actors is essential and must be a robust solution, as they can affect the company day-to-day business and live operation. Supply chain management is another complex example, where the even the logistics and shop floor systems come into sight, plus coordinated the flow of goods between suppliers and customers also, sometimes crossing country borders with the need of even more legal requirements like transport accompanying documents. These kinds of complex processes need not only raw data exchange but distributing workflow steps amongst different systems, which directly leads to cross-system workflow processes.

The direct evolution path, which were presented in the three main business process types are the following: [52]

1. Human interaction workflows
2. Workflows which involve applications
3. Transactional workflows

According to previous experiences, type 1 workflows are early automatizations, type 2 are EAI, SOA and BPM workflows and type 3 are usual ERP workflow implementations. In the next two chapter, two market leading ERP workflow solution will be analysed, that they converge towards the type 2. Originally the main driver for workflow automatization was to have better performance with less human interaction needed, but nowadays the WFMS main strength is to present business process changes faster on the market.

Architectural analysis of Microsoft Dynamics 365 Finance and Operation workflow engine

The Microsoft Dynamics AX ERP system based on its frequent workflow processing and advanced security system can be regarded as a role-based ERP solution. On development side, it is based on Object Oriented Programming paradigm, totally redesigned from the scratch, while all object is organized and can be reached from the Application Object Tree [47]. Previous data centre-based version, like Dynamics AX2012 had built in workflow processing, which is now a standalone product, has its own lifecycle in the cloud. Main business processes like budgeting, shipping and warehousing, payroll, personnel, etc. are all covered by custom tailored role-based user interface, which is supported by complex workflow engine through the system.

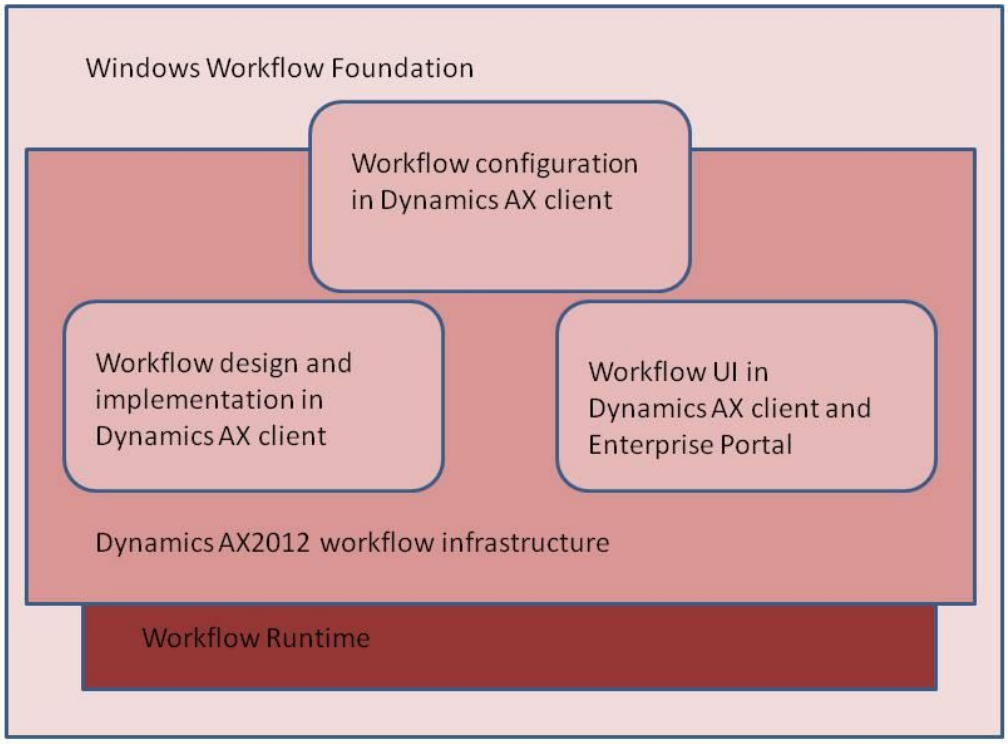


Fig.12. Windows workflow architecture for Microsoft Dynamics AX [48]

Let’s get one example: a purchase manager can control all the activate tasks from requisition to paying approval, if a change is needed in the participants because of absence or vacation, the workflow can be redirected automatically. What is important to mention is that the workflow solution extend the boundaries of the ERP system, but emerging into a central software solution, reaching out to other parts of the information flow. In the cloud base solution, D365 Finance and Operation, this goes even further: workflow became a standalone product, serving the needs of the entire ecosystem, not just one ERP solution.

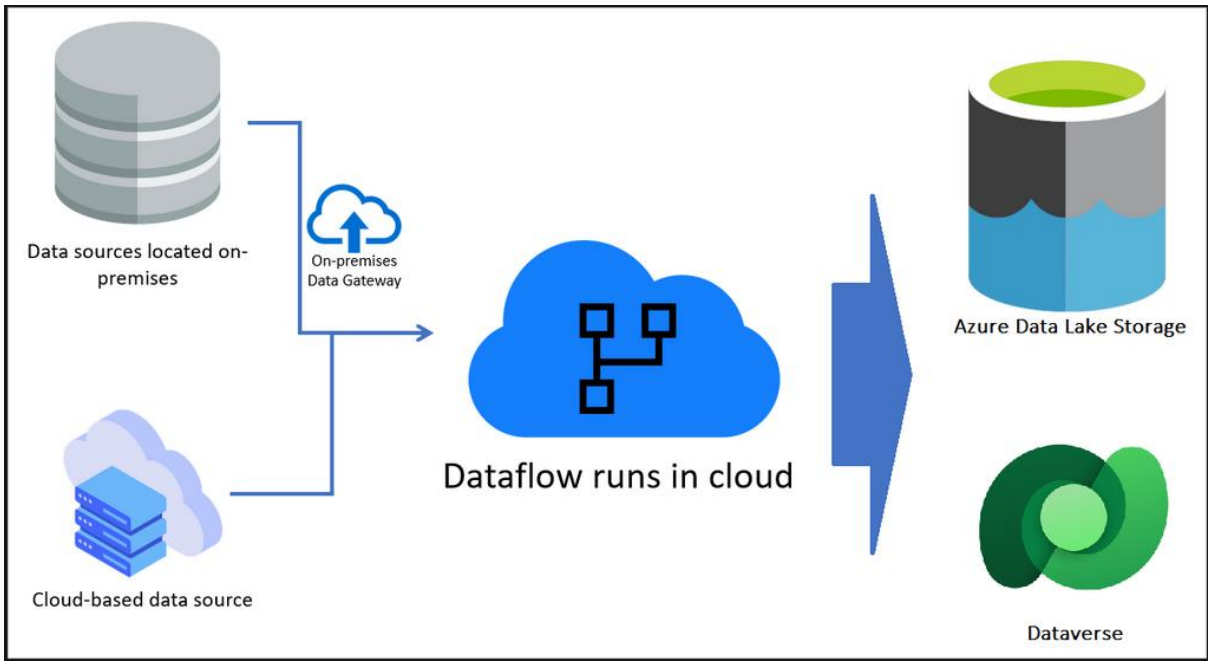


Fig.13. Workflow data flow in Azure DevOps [58]

As the whole workflow concept evolves, the ERP system is not anymore the main host, but more as a single customer of the workflow. After definition the business processes in the core ERP system, it is necessary to define the connecting workflows as a horizontal drive for the business process automatization. As business analysts make an abstraction of the business processes, they plan a monolith ERP application, but they should take into consideration also the IT architecture of an enterprise level company. As from this respect, cloud computing is only a tool, not the target. The already existing enterprise business flow, SOA elements has to be defined newly and not just refactored. This way can extend the longevity of the lifecycle of the ERP model.

Technical implementation of the workflow engine and its supporting objects

There is a fundamental difference between workflow usages of AX2012 and D365FO: as the first is a monolith windows application it uses the Windows Workflow Foundation shown on fig.12, the later utilizes Microsoft Flow in the Azure DevOps cloud solution. Both workflow solution also provides a wide range of elements, predefined tasks and roles for participants. Both has a common aspect also: they utilize an abstraction layer to hide the undelaying infrastructure from the ERP system, which is treated as a customer. In AX2012 the Workflow Foundation has no direct access to or integration with the core ERP, being a low-level infrastructure element. Workflow basic architecture lays above the workflow foundation and makes possible to define AX2012 specific workflows to be designed, implemented and configured, after then executed by the workflow foundation. Development phase is responsible to define the abstract workflow elements, tasks which later define the basic tasks of the business process. Business process owners (BPO) design the entire workflow with a graphical workflow editor, which is later passed to the WF runtime. The workflow runtime engine connects the AX2012 workflow infrastructure to the Workflow Foundation, then instantiates and executes the workflow, as runtime environments are handled and administrated by the system administrators [48].

The workflow document is the entry point for workflows in AX12012, therefore it is one of the fundamental documents of this ERP. All of the workflow documents in each types have to have a reference to a workflow document, as it can access the data content for the workflow. A workflow document is an object in the Application Object Tree (AOT) query, which supports a workflow document class. It is called as workflow document, as it clearly explains what it is all about, and the primary purpose of the workflow. This query can gather the information from more data sources hierarchically ordered to each other, worth mentioning that unlike the cloud-based solution, it is still uses ERP database as primary source. In AX2012 datacentre version, all workflow documents and its classes are stored in the metadata DB, inside the monolith ERP solution. To make these documents classes reusable by other workflows, developer assistance is needed, as the underlying structure change needs additional development work. This is not an agile way of dealing with business changes that is why cloud

based ERP solution started a totally new approach, where the physical implementation layer is totally hidden from the participants. Workflow categories are responsible for specifying the association between a workflow type and its corresponding module. When the BPO wants to define a new element for a workflow, these are the choices which can be used:

1. Tasks, which are atomic workflow points, representing one single unit of the workflow, developer defines the possible outcome for them.
2. Approvals, which allows executing a series of tasks, with a fixed set of possible outcomes.
3. Sub workflows, which are embedded workflows, defining a subtask which can be executed and reused in more than one workflow.
4. Manual decision, which enable human interaction and to make a choice between multiple outcomes.
5. Automated decision, which does the same as the manual ones, but without a human interaction, choice is made based on environmental variables.
6. Parallel activities contain more branches which can be executed simultaneously, which means they are independent from each other's outcomes.
7. Line-item workflows, which represent a master workflow document in a master-detail relationship. The enable a specific workflow task on a single line of a master document, for example exceeding the limit on an expense line of an expense report. The business relation between these documents has to be set up during the analysis phase.
8. Automated tasks, which are non-interactive ones, and invoke pre-defined business logic synchronously. These are new types of workflow element, together with the manual and automated decisions, parallel activities and line-item elements.

Workflow engines often user event handlers, which can be triggered by and internal or external trigger. Every event handler is generated on workflow element level, which represents the activities within the workflow. One the most common goals of the workflow is to approve documents, which is done by triggering the object event. This object event is usually a technical one, like pressing an OK or a menu button. Checking the WFMS interfaces, the client can be Dynamics AX or Outlook or any kind of element of the Microsoft software ecosystem.

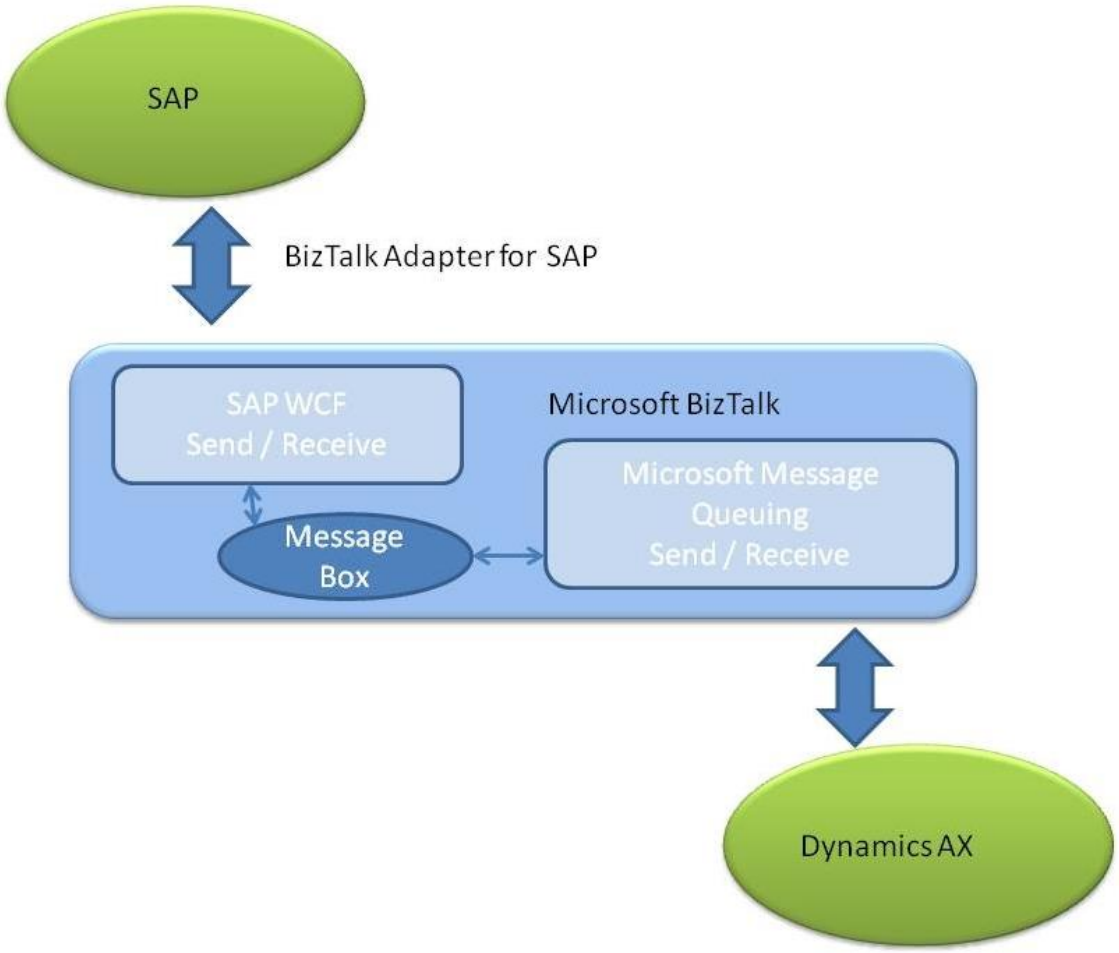


Fig.14. Biztalk communication schema cross SAP-AX system [1]

Cross system communication is done via WWF calls through AX objects, if an external solution, like SAP, wants to call them, two ways are possible:

1. Use the Microsoft Dynamics Business Connector via C# code in a suitable .Net environment
2. Use Web Services

Worth mentioning that Web service offered by Dynamics AX can be consumed by any language in the .Net framework. The Windows Workflow Foundation triggers the different business task from within the previously mentioned clients, so it acts like the WFMS and taking control of the process flow. While within the Windows server area, it can be called and used from non- Microsoft application also, while the standard interface is used. Extensible Application Markup Language (XAML) is used as the process definition tool, which extends the usability of the solution as being an industry standard. Workflow definitions can be defined as X++ code, XAML or the mixture of both. As out of the box BPEL is not supported, although it was originally developed and defined by IBM and Microsoft [54].

Workflow defined in SAP ERP

As the previously mentioned D365 Finance and Operation ERP system is much younger application than the SAP, it is worth taking a closer look at the so-called industry standard solution. To start with the historical background, many database table names and relations, also menu function names came from the mainframe era, sometime they go back to the first working version SAP R/2. These parts mainly cover those business logic which did not change much over the past decades, so the old algorithms still can do their duty. There are some layers which serves the business logic over the operating system and database level, they are also responsible for the platform independent operations. This data model, which is defined in the upper layer, has no direct mapping to the physical DB layer, it also supports platform independent operation if needed. Database interface delivers the online translation between the SAP Open SQL and the undelaying native SQL engine. This native SQL layer has no real data model defined. All information about the DB model and application specific schema is stored in upper layer handled by SAP.

The internal functional model consists of more layers, each of them supplies various functionalities. Lowest one contains programs, transactions, functional module screens and forms. This layer covers functions which can be reused in various business processes, because they are atomic business elements, like booking a general ledger entry or access employee address data. These action items are grouped into functional groups, these groups are technically OOP classes, containing static functions, sharing the attributes of the class. Doe to this behaviour, the functional groups can reach out to the DB, and manage the content while their active sessions. Some of them contains screens as input/output forms, but most majority are lacking the user facing parts, contain only executing code. These code parts are built up with the focusing to reusability, some of them are enabled for remote calling also, with remote function call (RFC), which is the SAP terminology for the standard Windows remote process

calls (RPC). These RFC calls has similar behaviour to the standard RPC calls: they allow selected functions to be called outside SAP, enabling application level data and schema change [44]. This interchange layer is continuously evolved (NetWaver Application platform), it makes possible to handle Web Services either as a consumer or a producer also. Worth mentioning that each of the RFC enabled modules can be reached by various Web Service protocols like WSDL, SOAP, etc.

These functional groups commonly form the middle layer of the SAP application architecture. Almost all functional part of an SAP application uses them to execute the building blocks of a business process or workflow. In the last decades, object oriented programming took place in SAP technology layer as main programming paradigm. Beyond this clear OOP classes, there are many more embedded ones which wraps the functional module logics.

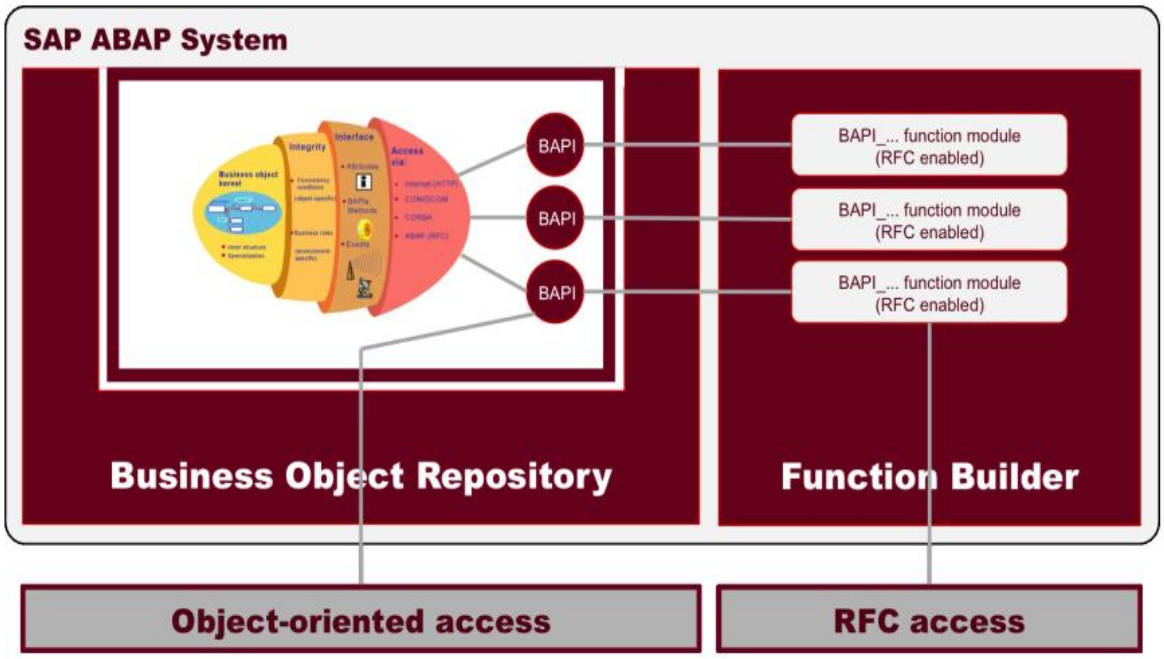


Fig.15. RFC and Object oriented access data access [1]

Since the first workflow implementation from the 90', SAP has re-implemented this solution many times. Object as a concept were already present in the technology layer even before SAP implemented clear OOP programming technology. It was possible with the

invention of the Business Framework Architecture, which was a new layer in the architecture. This new layer contained some new objects and methodology as business component and objects, BAPIs. Business objects became the most important elements, as they moved the object oriented methodology forward. Most typical examples for these objects are an invoice, an employee, or a sales or purchase order.

Between these business objects, several different types of relation can be defined, like part-of, kind-of, is-a, like a purchase order can have a customer (has-a), and items (part-of). Real inheritance between them is not supported, but this term is used, and there are ways to implement a behaviour which acts similar like inheritance. Some interface model can be used also, but this abstract type is not well defined according to the standards.

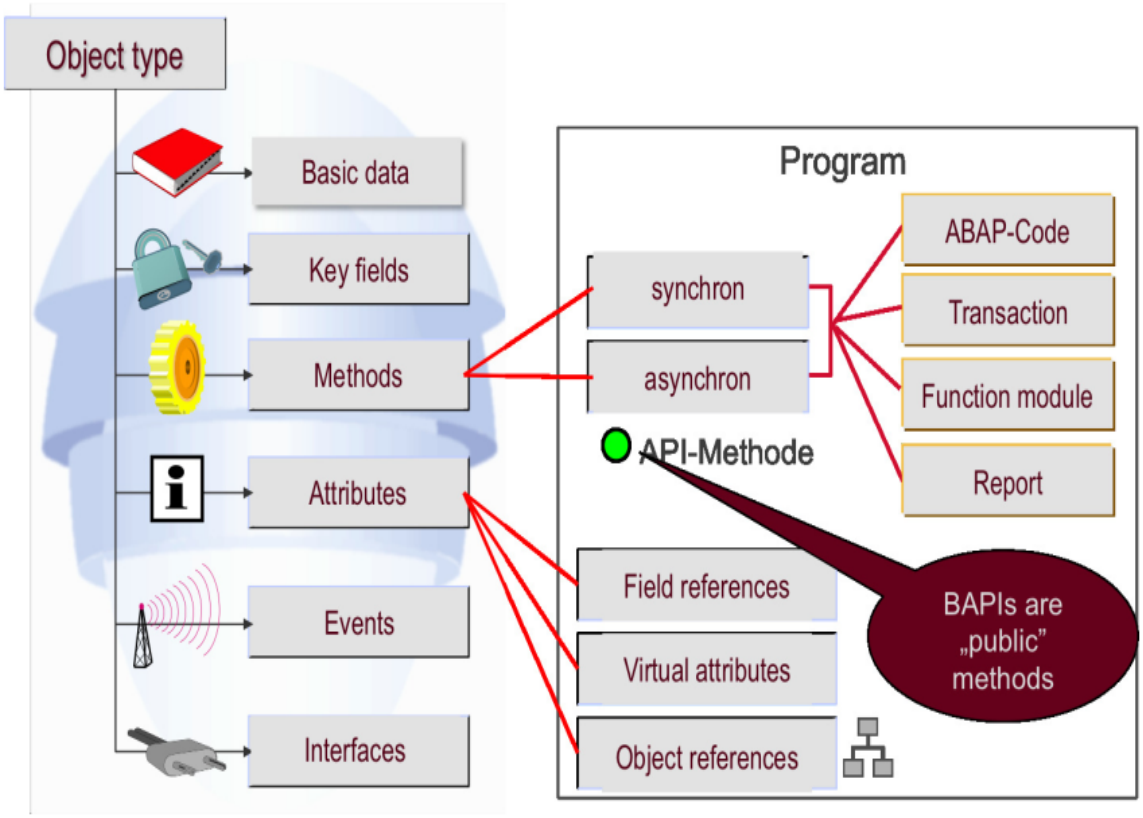


Fig.16. Business objects and their types [1]

Each object can have its own events, methods and attributes. Some attributes can identify the objects itself, and can act as a unique key, example is the material key to a key material. Another difference to the OOP world is that lack of real public-private distinction,

although there are some methods which can be considered as public is a different view. SAP applies internally each of these objects by attributes and methods, and the code must contain Business Object libraries to be able to call and execute objects and their elements.

This an OOP like thinking, implementing the logic behind the object oriented programming. BAPIs are special objects within the Business Objects, which RFC enabled so they are reachable from outside the SAP ERP application. SAP uses more invocation techniques, simplest is the simple method call. It is implemented through a standard RFC call, using RFC libraries on the caller side if the caller is not another SAP application. Newer applications can use web services, so BAPIs can be accessed through this way, where RFC is enabled. The technology in high level is OOP, where the caller side builds its own objects and calls the business methods [50].

These Business Objects are the most important building blocks of the standard SAP workflow. Each and every step in the workflow is based on these objects, and a special event can trigger the method. A workflow is a multi-step task, where each step is a single one connected to a Business Object method or can be an embedded multi step also.

According to the previously defined workflow step, a singled step can be executed by a human or can be an automated execution step by the workflow engine. Each step has to have an approval property, to determine who is allowed to perform. SAP uses a distinguished agent set methodology, which means that programmatically the agent has the right to execute a task. Task can be assigned to a group of users who will execute it, most commonly it is done in an authorization role.

On the bases of these authorization role, workflow engine decides the owner of the task runtime. Let’s say for example that the requestor’s line manager is the responsible agent. It is also possible that there are more responsible persons, e.g. a group of bookkeepers. The intersection between the responsible and possible agents is always the groups of users who will receive the workflow item, this group of users is called recipients. The group of possible agents are predefined and attached to the task as a property, but the responsible user group is calculated runtime according to the appropriate rule. These rules are predefined based on the HR (Human Resources) organization and dependency list, or can be done runtime by the program to find the right agent based on the parameters. By defining the rule this way, makes them reusable in most of the workflow environments [43].

Workflow execution by Business Objects

Process flow can be defined in many different ways in the workflow definition. Workflow during runtime has its own global status, variables, parameters which can be used for tasks or roles. These kinds of data storage SAP provide the containers. Unlike other non-structured data structure, containers are able to store not just simple fields but more complex ones like arrays, even Business Objects, these elements can be manipulated runtime. This global storage is called as workflow container, and is alive during the whole workflow execution. There are some other containers with important rules during the workflow lifecycle:

- Event container, which stores the event parameters. It can also store events triggered outside the workflow system, from other SAP application or even outside using web services or RFCs. The lifetime starts with the first event and ends when the triggered data is passed to the next container, e.g. workflow container.

- Methods container, which receives the data required for the method call, during this activity it receives and send data only to the work item container

- Work item container, communicates with the Workflow container, send and accepts data like information which is displayed to the agent, etc. Lifecycle is limited to the caller task.

- Rule container, gets data from Workflow container, main task is to define the possible agents for the current task. Lifecycle is limited to the caller task.

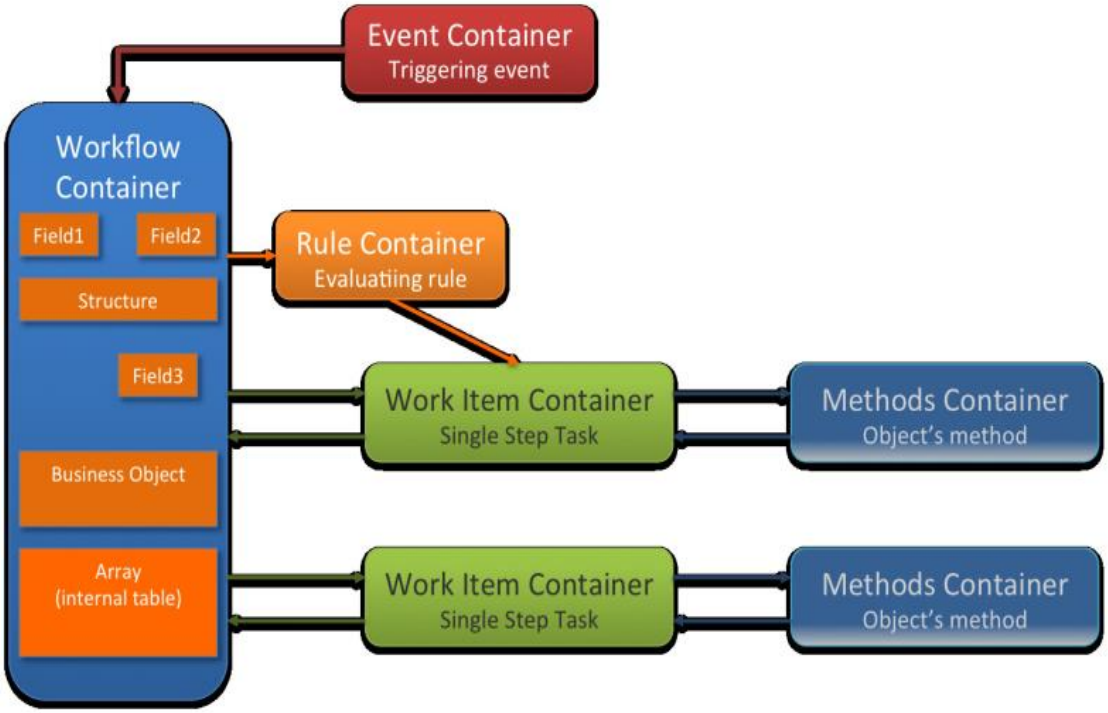


Fig.17. Container types [1]

The previously defined container technology has advantage of fast and simple data exchange between tasks. Recently it has been replaced by XML based data exchange, but in here SAP ABAP provides a lot of features.

The Business objects or the ABAP Dictionary provide the meta-data content. The Business objects or the ABAP Dictionary provide the meta-data content. Recent releases use pure OOP classes and their method to represent a single task. This a more updated architecture, but still Business Objects have to be used in some cases, as the already existing business cases embedded into these logical objects are not converted into object oriented classes, only the newer ones.

SAP workflows use many different steps for building up a workflow template, which is a design type object. We can distinguish the followings:

- Control steps, which can be a condition based on a control field and can lead to a logical decision. It also can be a loop; the two types are the while and until loops are implemented. Multiple conditions are also implemented, based on a container value the workflow can go on one of the pre-defined threads. It also contains user decision, which are similar to the multiple ones, but lets the user choose, and a fork which is an implementation of the parallel execution.

- Executive steps, which can be an activity like a background task or a dialog. Sending a mail from the system, which is a standard task where addresses are coming from the customer master. Sub workflow, which is responsible for starting a sub workflow as a multi-step task. Ad hoc anchor, which is a run time defined sub workflow. Web activity for out of system communication based on SOAP messaging.

- Special steps, which can be a container operation where the values of the container can be updated. Event creator and wait for an event step, which can wait for an event, start an event or suspend till a specific event is raised. Process control, which act like a real exit point, it jumps out of the process flow, and finishes the workflow.

All of these steps can be customized but not changed, expect the Activity one, which is a special one as it can be used to execute own tasks. As described earlier a task always runs a method, which in previous versions is a member of the Business Objects, in recent releases it can be an OOP object also. Inside the Business Objects different types of methods can be defined like: transaction call, BAPI, function call, RFC or own developed code. Worth mentioning, that the least one is not reusable in other places of the system, but gives huge freedom to the developer. It comes from the SAP philosophy, that the main building blocks are already existing ones with different parametrization. Although if OOP classes are used, methods can be instance dependent or static also.

SAP workflow as it is now executing mainly internal workflows, in its current state is a very useful business tool for automatizing processes. The main advantage is that is builds up from business process blocks, which can be reused widely in different workflows again and again. The idea is long based on object oriented thinking, but only the recent releases

implement this programming paradigm fully. It separates the authorization tasks from the responsibilities in a required manner, and in a well configured system they complete each other to find and delegate to the right person for the given task. BPEL models are not fully supported directly, if PI is a goal, a tool like EAI should be used. SAP workflow solution also support interconnect capabilities, which will be described later.

Business process management and Interconnect workflows

As our previous research shows, business needs nowadays turn to cross system processes, where more application is involved in the same cloud solution, or even Business Process Management solutions. In an SAP workflow process any task or method can trigger an RFC call to get information from another SAP system, or a web service using SOAP protocol to get the same from a non-SAP application. With the aid of these function, a workflow is able to reach data tags or call methods from other applications, or even start an outer workflow if it is allowed in the other system. All the tasks which aim to reach a remote system starts with the Web Activity step type task. This task should be based on a standard Business Object, which is the XML_DOC.

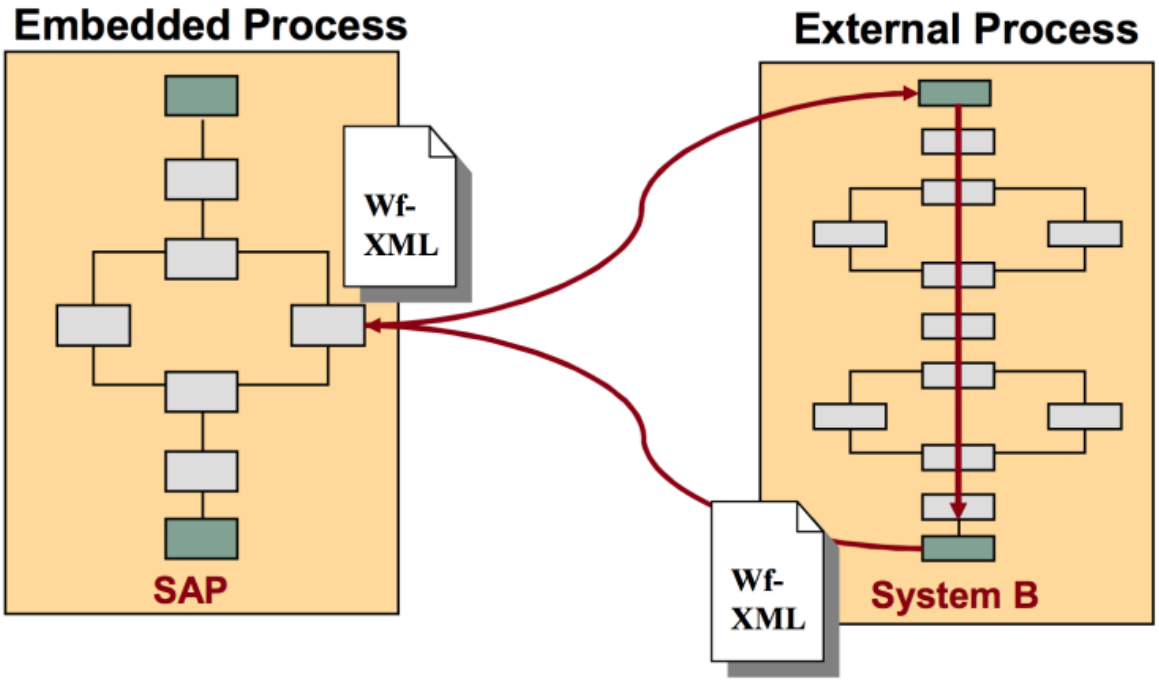


Fig.18. XML_DOC usage [1]

The communication uses a WF-XML interface inside the XML or SOAP message, which is an open standard with some implementation by commonly available workflow engines. There are four main methods which can be used:

- start a workflow
- allow remote system to start a local workflow
- synchronous execution allowed
- asynchronous execution allowed

These can enable any outer systems, which has the capability for WF-XML methodology, to work with remote workflows.

The other main solution which is in the scope of our research analysis is the Microsoft Dynamics ERP family, which uses another solution: for starting and controlling a remote workflow one has to use WWF (Windows Workflow Foundation). This solution is closer to the technical layer than the one used by SAP. As previously described, events are low level building blocks of a workflow, which can trigger basic steps in a Dynamics AX system. They can start workflows also by default, but with minimal development more complex ones can be started also. There are no predefined protocols to handle workflow invocations from other systems, but the solid architecture of the Dynamics ERP system offers solution to handle this.

First, we have to see that Dynamics AX first implemented a single workflow engine, later pushed the task, like an outsourcing, to WWF which is an integral part as a service of the Windows server ecosystem. This led to two findings:

1. From this point it is the WWF which handles all the flow management, scheduling and dialog calls, etc.

2. From this point Dynamics AX is only responsible for ordering and communicating with the services which WWF offers.

WWF can handle internal method calls like C# or Visual Basic methods, or use web services, where proxy classes should be defined to Web Service Definition Language, WSDL. Communication schema is defined in Fig. 19.

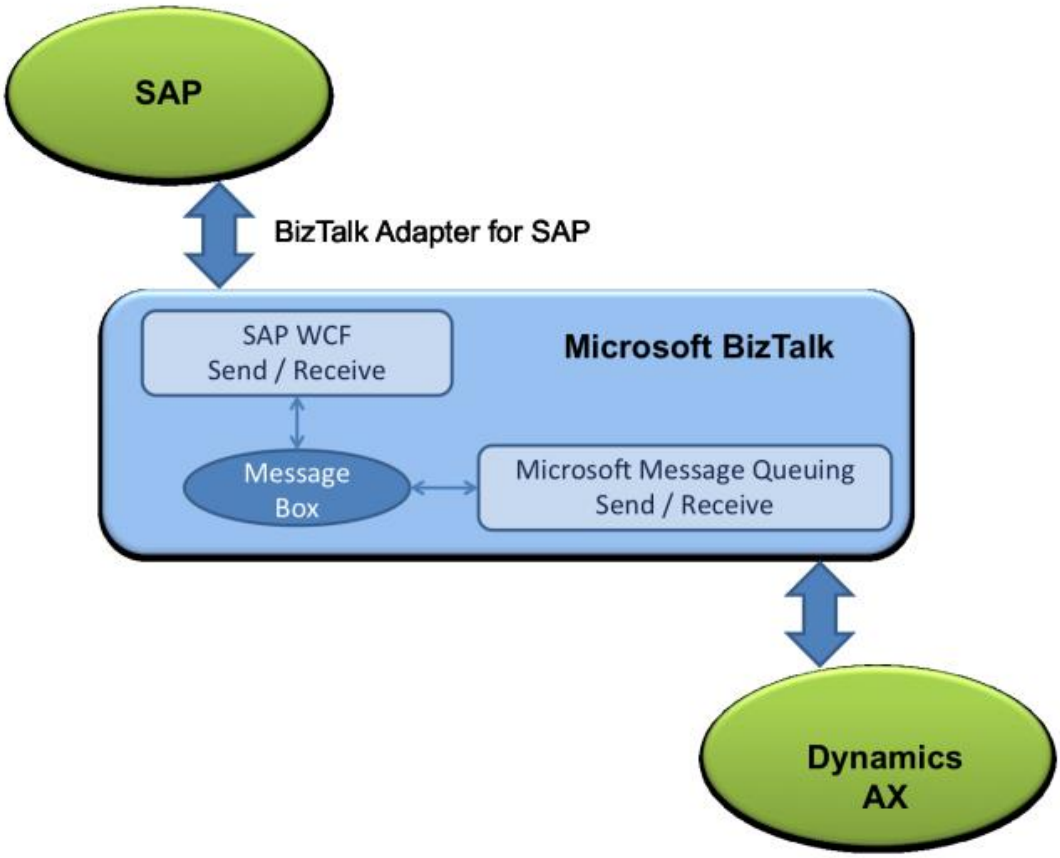


Fig.19. Integration middleware communication schema.

We have a strong business case behind the existence of the cross-application workflows: a lot of multinational companies evolved by acquisitions; therefore they have to solve the communication issue between different ERP systems and workflows. This is the most important business drive for these large-scale ERPs: they are able to operate in multi company environments by default.

The technical solution for the intercompany, multi ERP issue is the usage of middleware layer which bridges the data flow between SAP and the Application Integration

Framework inside Dynamics AX. As a middleware solution, Biztalk serves is the commonly used industry solution, which uses Microsoft Message Queuing (MSMQ) in connection with SAP Windows Communication Foundation (WCF) to provide reliable communication solution between the two tiers. Dynamics AX introduced Application Integration Framework, and in later cloud editions it uses OData, to provide flexible programming framework for different business needs. AIF and OData also offer a wide range of services supporting fast customizations and application building, as not just the standard Create Read Update Delete (CRUD) operations are available but more complex building block of basic business operations. Extensible Stylesheet Language Transformation (XSLT) can be used to create mapping for the integrations.

On the other hand, data mapping on SAP side is handled by Intermediate Documents (IDOC), allowing reusable and scalable document handling for the growing future needs. Intercommunication can be possible on the technical side if one the technologies are in use:

1. Custom middleware using .NET connector and SAP .NET connector, enabling RFC connection.
2. WEB service invocation used on both sides; WWF layer is preferable on Dynamics side.
3. Using an EAI compatible solution, like BizTalk

The EAI can be extended to include business logic also, the data exchange and madding can be done also, but the service offered are connected into service repositories, controlled by a Service Bus. This central element Service Bus is able to serve the service with the relevant data, also call the required (external) service functions. This feature is implemented in SAP and Dynamics AX also, and each of them are able to interconnect to ESB environment from third party providers also. It is an important feature, as these ERP systems are able to interconnect with external Workflow engines also, worth mentioning that SAP provides application level OOP environment through Business Objects. Many important methods from the Business Objects can be externally called as a service, or through a web service, these single services can be collected to a service repository.

SAP had a recognition from the early Web Service era, as these functions are too complex to build an application, therefore invented the Enterprise Services, which are fully

documented high-level services executing a specific business process. Microsoft Dynamics AX although evolved in another way, outsourcing the workflow capabilities into the already existing Windows Workflow Foundation, and its predecessors in the Azure Cloud. This decision separated the Workflow from the original ERP system and offered a different lifecycle for the Workflow to serve more generally as a cloud-based workflow engine.

Bibliography

[1] Attila Selmeci, Tamás Orosz, István Orosz Workflow processing using ERP Objects. ACTA CYBERNETICA 22 : 1 pp. 183-210. , 28 p. (2015)

[2] István Orosz, László Szívós, The Role of Data Authentication and Security in the Audit of Financial Statements, ACTA POLYTECHNICA HUNGARICA (2014)

[3] Orosz, I. and Orosz, T. (2014) "Microsoft Change Management Applying Comparison of Different Versions", Acta Technica Jaurinensis, 7(2), pp. pp. 183-192. (2014)

[4] Attila Selmeci and István Orosz, Software as a Service operation model in cloud based ERP systems, (CS)2 - The 11th Conference of PhD Students in Computer Science, Szeged 2018

[5] István Orosz, Tamás Orosz, Software as a service in cloud based ERP change management, 2017 IEEE 11th International Symposium on Intelligent Systems and Informatics (SISY 2017), Page(s): 000181 - 000186,

[6] István Orosz, Tamás Orosz: Code reusability in cloud based ERP solutions. Proceedings of the IEEE 21st International Conference on Intelligent Engineering Systems (INES 2017): Larnaca, Ciprus. IEEE, 2017. pp. 193-198

[7] Tamás Orosz, István Orosz, Company level Big Data Management, 9th Jubilee IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI 2014). Budapest, Magyarország: Óbudai Egyetem, Page(s): 299 - 303

[8] Attila Selmeci and István Orosz, Workflow processing using SAP Objects, (CS)2 - The 8th Conference of PhD Students in Computer Science, Szeged (2012)

[9] István Orosz, Tamás Orosz: Change management and workflow processing using Dynamics AX objects. 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics, SISY 2012, Subotica, 2012, September, 20-22 pp. 249-254

[10] Selmeci, I. Orosz, Gy. Györök, T. Orosz: Key Performance Indicators Used in ERP Performance Measurement Applications pp. 43-48. , 6 p. In: Szakál, A (szerk.) 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics, SISY 2012, Subotica, 2012, September, 20-22

[11] István Orosz, Túri Balázs, Tamás Orosz, Selmeci Attila, Local Governments-specific BPR mini-project with SAP applications, SAMI 2012: IEEE 10th International Symposium on Applied Machine Intelligence and Informatics, Page(s): 119 – 123

[12] István Orosz, Tamás Orosz: Business Process Reengineering Project in Local Governments with ERP. BULETINUL STIINTIFIC AL UNIVERSITATII POLITEHNICA DIN TIMISOARA ROMANIA SERIA AUTOMATICA SI CALCULATORAE 57 (71) : 4 pp. 253-258. , 6 p. (2012)

[13] Selmeci Attila, Orosz István, Orosz Tamás, Györök György: ERP change management for innovation and sustainability applied to User Interfaces. In: Szakál, Anikó (szerk.) Proceedings of the 2012 IEEE 16th International Conference on Intelligent Engineering Systems (INES). Lisbon, Portugal, (2012) pp. 341-346.

[14] László Duma, István Orosz, Information technology systems in logistics and roles of ERPs, 2012 IEEE 13th International Symposium on Computational Intelligence and Informatics (CINTI), Page(s): 115 – 121

[15] István Orosz ; Túri Balázs ; Tamás Orosz: Inherited SAP Development Concepts using genuine IT programming tools. 2011 IEEE 12th International Symposium on Computational Intelligence and Informatics (CINTI). 2011.561-566

[16] I. Orosz*, A. Selmeci**, T. Orosz: Software as a Service operation model in cloud based ERP systems. IEEE 17th International Symposium on Applied Machine Intelligence and Informatics, 2019

[17] Tamás, Orosz ; István, Orosz ; Attila, Selmeci: Continuous ERP Strategy Development and Implementation at Enterprises 7th International Symposium on Applied Informatics and Related Areas (AIS) 2012, Székesfehérvár, Magyarország, pp. 35-39. ISBN:978-615-5018-40-4

[18] Microsoft Dynamics Sure Step Methodology, https://technet.microsoft.com/en-us/library/aa496439.aspx

[19] Richard Murch, The Software Development Lifecycle - A Complete Guide, Amazon Digital Services LLC, ASIN: B007ZCRP1I.

[20] Clements, Kazman, Klein, Evaluating Software Architectures: Methods and Case Studies, 2002, ISBN-10: 020170482X

[21] Alp Oral, Bedir Tekinerdogan, "Supporting Performance Isolation in Software as a Service Systems with Rich Clients", Big Data (BigData Congress) 2015 IEEE International Congress on, pp. 297-304, 2015.

[22] Armando Fox, David A. Patterson: Engineering Software as a Service: An Agile Approach Using Cloud Computing, Strawberry Canyon LLC, 2013, ISBN 0984881247, 9780984881246

[23] https://www.microsoftpressstore.com/articles/article.aspx?p=2240847&seqNum=10, The MorphX Development Environment and Tools

[24] Peter Mell and Tim Grance: The NIST Definition of Cloud Computing. Version 15, 10-7-09, http://thecloudtutorial.com/nistcloudcomputingdefinition.html

[25] Enterprise Resource Planning software blog, http://www.erpsoftwareblog.com/2016/09/hosting-microsoft-dynamics-cloud-evaluating-iaas-paas-saas/

[26] Database Entity Relationship Diagram, https://community.dynamics.com/ax/b/axhari/archive/2014/08/09/ax-2012-database-entity-relationship-diagrams

[27] https://www.slideshare.net/HamdaouiAmine/microsoft-dynamics-ax2012-forms-and-tables-methods-call-sequences-30159669

[28] Aggarwal, K. K., et al. "Software reuse metrics for object-oriented systems." Software Engineering Research, Management and Applications, 2005. Third ACIS International Conference on. IEEE, 2005.

[29] Jeffrey S. Poulin, Ph.D., "The Search for a General Reusability Metric", Lockheed Martin Federal Systems Owego, New York, Proceedings of the Workshop on Reuse and the NASA Software Strategic Plan, Fairfax, VA, 24-27 September 1996.

[30] Martin L. Griss: Systematic Software Reuse: Architecture, Process and Organization are Crucial, Software Technology Laboratory, HP Laboratories, Fusion Newsletter, http://martin.griss.com/pubs/fusion1.htm

[31] Krešimir Popović, Željko Hocenski: “Cloud computing security issues and challenges”, MIPRO 2010 Proceedings of the 33rd International Convention, ISBN: 978-9-5323-3050-2

[32] Chao Jin, Wenjun Wu, Hui Zhang, "Automating Deployment of Customized Scientific Data Analytic Environments on Clouds", Big Data and Cloud Computing (BdCloud) 2014 IEEE Fourth International Conference on, pp. 41-48, 2014

[33] Microsoft Technet Metadata Service Class Diagram, https://technet.microsoft.com/en-us/library/gg845212.aspx

[34] Database Entity Relationship Diagram, https://community.dynamics.com/ax/b/axhari/archive/2014/08/09/ax-2012-database-entity-relationship-diagrams

[35] Jeffrey S. Poulin, Ph.D., "The Search for a General Reusability Metric", Lockheed Martin Federal Systems Owego, New York, Proceedings of the Workshop on Reuse and the NASA Software Strategic Plan, Fairfax, VA, 24-27 September 1996.

[36] Martin L. Griss: Systematic Software Reuse: Architecture, Process and Organization are Crucial, Software Technology Laboratory, HP Laboratories, Fusion Newsletter, http://martin.griss.com/pubs/fusion1.htm

[37] FSM. David Cannon (2011). ITIL Service Strategy 2011 Edition. The Stationery Office. ISBN 978-0113313044.

[38] Krešimir Popović, Željko Hocenski: "Cloud computing security issues and challenges", MIPRO 2010 Proceedings of the 33rd International Convention, ISBN: 978-9-5323-3050-2

[39] eCloudChain: Cloud more secure than the Data Centers? https://www.ecloudchain.com/how-secure-is-the-cloud/

[40] Proof of Concept -Pilot Guideline, William French, Dir of Div of Enterprise Applications, James Weaver, Dir of Div of Tech Engineering, 04/29/2010, GDL-EASS011 COMMONWEALTH OF PENNSYLVANIA DEPARTMENT OF PUBLIC WELFARE

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

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

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

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

[45] 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.

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

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

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

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

[50] 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

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

[52] 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

[53] 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

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

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

[56] Guy K. Kloss, Andreas Schreiber, Lecture Notes in Computer Science 4145:37-45, May 2006, DOI:10.1007/11890850_5, International Provenance and Annotation Workshop, IPAW 2006

[57] WorkFlow Management Coalition (WfMC) , CONTENIDO Objetivo Stándares de la WfMC Grafo de Actividades vs Grafo de Estados Ejemplos con Xflow. Published byTiffany Burnam, https://slideplayer.com/slide/3344876/

[58] Microsoft Workflow dataflow diagram in Azure Devops https://docs.microsoft.com/hu-hu/power-query/dataflows/overview-dataflows-across-power-platform-dynamics-365

[59] Lvivity consulting, https://lvivity.com/proof-of-concept-meaning

[60] Anil Kumar Gupta, "Quality Assurance for Dynamics AX-Based ERP Solutions: Verifying Dynamics AX customization to the Microsoft IBI Standards Studies". Packt Publishing, 2002, ISBN-10: 1847192912.

[61] Bae, N. and Aschroft P.: "Implementation of ERP systems: accounting and auditing implications", Information System Control Journal, 2004, Vol. 5, pp. 43-8.

[62] Brazel, J.F. and Agoglia, C.P.: "An examination of auditor planning judgements in a complex accounting information system environment", Contemporatry Accounting Research, 2007, Vol. 24 No. 4., pp. 1059-83.

[63] Fodor, J., Ősz, R., "Possible applications of fuzzy methodology in the educational process", IEEE 11th International Symposium on Applied Machine Intelligence and Informatics (SAMI), DOI: 10.1109/SAMI.2013.6480992, ISBN: 978-1-4673-5927-6, pp. 37-40, 2013.

[64] Forrester, Jay Wright, "Industrial dynamics". Waltham, MA: Pegasus Communications, 1961.

[65] Goldberg, S. and Godwin, J.H. : "Operational reviews on auditing ERP", The Journal of Corporate Accounting and Finance, 2003, Vol. 14 No. 4, pp. 63-5.

[66] Hunton, J.E., Wright, A.M. and Wright, S.: “Are financial auditors overconfident in their ability to assess risk associated with enterprise resource planning system?”, Journal of Information Systems, 2004, Vol. 18 No. 2., pp. 7-28.

[67] International Standard on Auditing 200: Overall objectives of the independent auditor and the conduct of an audit in accordance with international standards on auditing, IFAC. http://www.ifac.org/sites/default/files/downloads/a008-2010-iaasb-handbook-isa-200.pdf

[68] International Standard on Auditing 315: Identifying and assessing the risk of material misstatement through understanding the entity and its environment, IFAC. http://www.ifac.org/sites/default/files/downloads/a017-2010-iaasb-handbook-isa-315.pdf

[69] International Standard on Auditing 330: The auditor’s response to assessed risks, IFAC. http://www.ifac.org/sites/default/files/downloads/a019-2010-iaasb-handbook-isa-330.pdf

[70] Kanellou, A. and Spathis, C.: ”Auditing in enterprise system environment: a synthesis”, 2011, Journal of Enterprise Information Management, Vol. 24 No. 6, pp. 494 – 519.

[71] Kuhn, J.R. and Sutton, S.G.: “Continuous auditing in ERP system environments: the current state and future directions”, 2010, Journal of Information Systems, Vol. 24 No 1., pp. 99-112.

[72] Maedche, Alexander, "An ERP-centric Master Data Management Approach" (2010). AMCIS 2010 Proceedings. Paper 384. http://aisel.aisnet.org/amcis2010/384

[73] Messier W.F., Eilifsen, A. and Austen, L.A.:”Auditor detected misstatements and the effect of information technology”, International Jounal on Auditing, 2004, Vol. 8, pp. 223-35.

[74] Vendrzyk, V.P. and Bagranoff, N.A.: ”The evolving role of IS audit: a field study comparing the perceptions of IS and financial auditors.” 2003, Advances in Accounting, Vol. 20, pp. 141-63

[75] Wright, S. and Wright, A.M.: “Information system assurance for enterprise resource planning systems: unique risk considerations”, Journal of Information Systems, 2002, Vol. 16, pp. 99-113.

[76] Yang, D.C. and Guan, L.: “The evolution of IT auditing and internal control standards in financial statement audit. The case of the United States”, 2004, Managerial Auditing Journal, Vol. 19 No. 4, pp. 544-55.

[77] Filicetti, J. *PMO and Project Management Dictionary,* PM Hut. (August 20, 2007): Retrieved 16 November 2009 http://www.pmhut.com/pmo-and-project-management-d

[78] Phillips, J. R.: *Enhancing the Effectiveness of Organizational Change Management,* Human Resource Management, vol. 22(iss. 1/2), pp. 183-199, 1983 Retrieved 12/21/11 from http://onlinelibrary.wiley.com DOI: 10.1002/hrm.3930220125

Object Management Group® (OMG®), Modell Driven Architecture World, https://www.omg.org/mda/

Appendix A - Citations

Based on Google Scholar, as of July 2023, contains only the most cited article:

1. **Selmeci, I. Orosz, Gy. Györök, T. Orosz: Key Performance Indicators Used in ERP Performance Measurement Applications pp. 43-48. , 6 p. In: Szakál, A (szerk.) 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics, SISY 2012, Subotica, 2012, September, 20-22**

Citations shortlisted:

1. Real-time information systems and methodology based on continuous homomorphic processing in linear information spaces - M ZINNER, G Luhn, M Ertelt, M Austen - 2015 - Google Patents
2. Indicadores de gestión: toma de decisiones basada en inteligencia de negocios - FEC Rozo - Tecnología Investigación y Academia, 2013 - revistas.udistrital.edu.co
3. After a Successful Business Case of ERP–What Happens then? - B Johansson, L Karlsson, E Laine, V Wiksell - Procedia Computer Science, 2016 – Elsevier
4. Performance metrics in engineering change management-Key Performance Indicators and engineering change performance levels - N Kattner, T Wang, U Lindemann - 2016 IEEE International …, 2016 - ieeexplore.ieee.org
5. Beyond the factory paradigm: Digital nomadism and the digital future (s) of knowledge work post-COVID-19 - B Wang, D Schlagwein… - Journal of the …, 2020 - aisel.aisnet.org

6. Modelo para optimizar el proceso de gestión de negocio combinando minería de procesos con inteligencia de negocios desde almacenes de datos - JC GIRALDO, J JIMÉNEZ… - Revista …, 2017 - sweetpoison.revistaespacios.com
7. Impact analysis of enterprise resource planning post-implementation modifications - M Parhizkar - 2016 - openaccess.city.ac.uk
8. Decision-making support in engineering design based on collaborative dashboards: integration of business intelligence techniques - A Fradi, M Bricogne, M Bosch-Mauchand… - … on Research into …, 2017 - Springer
9. Performance issues of In-Memory Databases in OLTP systems - P Szpisják, L Rádai - 2016 IEEE 11th International Symposium …, 2016 - ieeexplore.ieee.org
10. Analysis of institutional and business stress under new technologies implementation - C Drumea - Bulletin of the Transilvania University of Brasov …, 2016 - webbut.unitbv.ro
11. Integrating Business Process Management and Data Mining for Organizational Decision Making. - J Giraldo, JA Jiménez, MS Tabares - Res. Comput. Sci., 2015 - rcs.cic.ipn.mx
12. Support of physics education by means of business informatics methods at University level - T Orosz - 2015 IEEE 10th Jubilee International Symposium on …, 2015 - ieeexplore.ieee.org
13. Causes of failure in the implementation and functioning of information systems in organizations - JR Figueroa-Flores… - International …, 2020 - pdfs.semanticscholar.org
14. Project management for manufacturing using PMI's foundation in PLM technologies - JE Montoya Cano - 2016 - repository.eafit.edu.co
15. Optimizing knowledge transfer in Physics education by fitting supplemental materials to individual learners - É Stefánkó, T Orosz - 2016 IEEE 11th International Symposium …, 2016 - ieeexplore.ieee.org
16. PRACTITIONER-ORIENTED VIEW ON SUCCESSFUL CRM ADOPTION AMONG SMEs-COMPARATIVE CASE EVIDENCE FROM THE B2B CONTEXT - K Kuusinen - 2017 - osuva.uwasa.fi
17. A Business Intelligence Model to support the engineering student life cycle analysis Recruitment - A Santoyo-Sanchez, JL Reyna-Gascón… - Technological Trends in … - cidetec.ipn.mx

18. Planeación de la producción en la línea de cajas secas de una empresa metalmecánica - MG Cano Sánchez - ri.uaemex.mx
19. PROPUESTA DE INDICADORES CLAVES PARA LA GESTIÓN DE PROYECTOS BASADOS EN METODOLOGÍA SCRUM UTILIZANDO PROCESOS DE ETL. - R VERDEZOTO BÓSQUEZ - 2019 - repositorio.uisrael.edu.ec
20. Incidencia de una estructura de gestión estratégica de proyectos de investigación en el incremento de capacidades organizacionales en facultades de administración - AA Del Rio Cortina - 2020 - repository.ean.edu.co
21. Modelo para optimizar el proceso de gestión de negocio combinando minería de procesos con inteligencia de negocios desde almacenes de datos - JCG Mejía, JJ Builes, MST Betancur - revistaespacios.com
22. Key performance indicators for adopting sustainability practices in footwear supply chains - M Moktadir, Y Mahmud, A Banaitis, T Sarder, MR Khan - 2021 - dspace5.zcu.cz
23. Measuring ERP System Success: Success Indicators and Structural Equation Modelling Approach - OA Ra, DK Mahalikb - Turkish Journal of Computer and Mathematics …, 2021 - turcomat.org
24. Metrics and Models for Evaluating the Quality of ERP Software: Systematic Mapping Review - MA Mohammed, AM Talib, IA Al-Baltah - Metrics and Models for …, 2020 - igi-global.com