# ERP change management for innovation and sustainability applied to User Interfaces

A.Selmeci, I. Orosz, T. Orosz and Gy. Györök
Óbudai Egyetem AREK, Székesfehérvár, Hungary
selmeci.attila@arek.uni-obuda.hu; orosz.istvan@arek.uni-obuda.hu; orosz.tamas@arek.uni-obuda.hu; gyorok.gyorgy@arek.uni-obuda.hu

***Abstract*— Enterprise Resources Planning Systems, ERP, have been successfully applied in business since their first appearance. Business portfolios of ERP Suppliers, such as SAP or Microsoft, have gradually been extended and improved to fulfill customers' requirements almost in any functional areas using defined standard business processes. Market leader ERP software providers, however, have to innovate their products continuously. Once a business or technological solution is being integrated in a software component, a further crucial requirement is to provide a long term support and also to take innovation into consideration. The present paper introduces our new approach, a customer-oriented ERP change management methodology, firstly applied to Microsoft Dynamics and SAP User Interfaces.**

## I. INTRODUCTION

ERP Systems apply several user interface technologies to manage the Systems in efficient and easy-to use way. ERP change management issues should not only deal with the applications of innovative IT solutions, but also be concerned in sustainability. ERP change management projects are considered to be collections of complex requirements, each consisting of basic business solutions. The main characteristics of basic business solutions are the weighted importance of innovation and sustainability. A collection of projects reflects how the role of innovation and sustainability changes over time. The present paper exemplifies such variety of User interfaces for SAP and Microsoft Dynamics, correspondingly.

## II. BASIC ARCHITECTURE OF THE SAP SYSTEMS

SAP solutions are based on the SAP Web Application Server technology. This technology provides open, scalable and robust infrastructure for running and developing dynamic applications. This is the core element of the SAP ERP (successor of R/3) system as well. The earlier R/2 releases had two layers, where the business logic and the storage as a monolith element took place on a mainframe, and the presentation layer appeared on a terminal. The communication capabilities were limited to LU 6.2 for other systems. R/3 systems inherit the real-time capability as we can see in the 'R' letter, but is redesigned for three-tier client-server architecture (see Fig. 1).

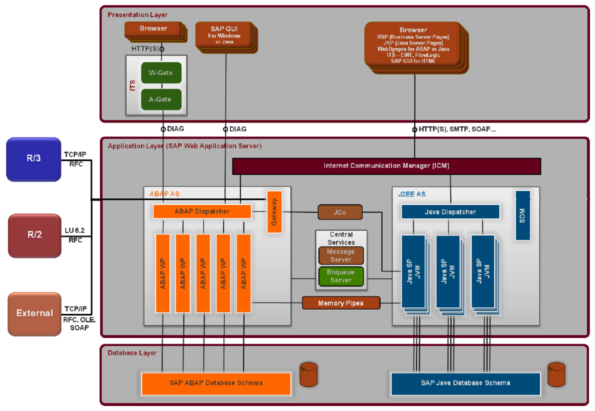


Figure 1 WebAS high-level technical architecture

This architecture means that the application is divided into three layers:

- Database layer: this is a single component (except some supported grid database systems), where the business data, and the whole repository are stored.
- Application layer: it is responsible for controlling and processing the applications. The business logic is running in this layer providing system-oriented services and ensures connectivity to presentation and other software components. For Web application this layer can be separated to more parts determining the connectivity, presentation, business logic itself, integration and persistent layers. (See details on Fig. 2 and later.)
- Presentation layer: it is the front-end, the Graphical User Interface, that runs on PCs, workstations, Web browsers or even on mobile devices.

Fig. 1 shows above the today's high-level technical architecture of the SAP Web Application Server.

The above figure refers some till now not detailed component and protocols used in communications. In the early R/3 system, before WebAS SAP does not provided direct HTTP protocol, but only DIAG (Dynamic Information and Action Gateway) and RFC (Remote Function Call) were provided. The DIAG is SAP's own protocol to communicate between the application layer and the SAP front-end software, called SAP GUI. For real data exchange SAP uses the CPI-C (Common Program Interface for Communication) based RFC protocol. This is a program-to-program communication protocol, which was used and implemented in the R/2 and R/3 world. The figure mentions the R/2 connection via LU 6.2 protocol. CPI-C and RFC communications could use TCP/IP and LU 6.2 communication as well. The IBM's Logical Unit (LU) 6.2 is part of the System Network Architecture (SNA) protocol. SAP provided by RFC a possible communication layer where other programs, system or even own developed front-end applications could use the system services.

As of NetWeaver releases the SAP Application Server is expanded with Web server functionality as well and the whole SAP Web Application Server architecture can be separated as we mentioned above into five sub-layers according to the technical areas (see Fig. 2).

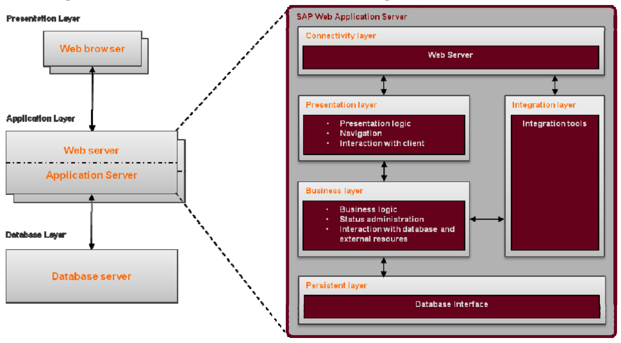


Figure 2 Sub-components of the Web Application Server

The 5 sub-components of the WebAS architecture:

- Presentation layer: presentation logic (view or screen definitions and switches), navigation, and the interaction with the client are defined here.
- Business layer: the business logic is running here, so the status of the applications are administered as well on this layer, and of course the service call like database connection or external resource usage are defined and maintained here.
- Integration layer: this layer opens the Web AS via integration engine to communicate via standard interfaces with other systems and solutions.
- Connectivity layer: this is the first layer facing with client or connected systems, application. The main service, which provides the Web server functionality and other standard protocols, is the Internet Communication Manager (ICM).
- Persistence layer: as we described above the Web AS ABAP environment guarantees database independent programming by Open SQL, which is implemented for the Java world as well and offers a variety of standard Application Programming Interfaces (APIs) to Java programmers, such as SQLJ and other Java technologies.

As we have seen the SAP Web AS with its two (ABAP and J2EE) personalities provides not only thick client capabilities (see it soon), but also many different Web UI options are offered out of the box.

## III. Frontend Possibilities of SAP Systems

SAP provides a universal front-end solution called SAP GUI (SAP Graphical User Interface). SAP at the beginning provided stand-alone SAP GUI solutions for Mac, OS/2, Windows and so on, but later decided to support three flavors according the available platforms. The SAP GUI family contains

- SAP GUI for Windows supporting MS Windows operating systems based on OLE and ActiveX controls
- SAP GUI for Java supporting all environments running Java runtime environment. This GUI is generally called platin GUI as well, because it is a platform independent solution.
- SAP GUI for HTML is a web based GUI frequently called as WebGUI because of the name of the service, which provides it.

SAP communicates with the SAP GUI client via DIAG protocol and in some cases RFC is used as well. The SAP GUI is a thick client; it must be installed onto the front-end workstation. The standard GUI for SAP is the SAP GUI and it was designed for end-users working with numbers, calculations and so on.

Each SAP application looks like the other, because the client software retrieves the screen element type, position, size, and of course the content and some other attributes, but the final design takes place on the front-end machine [1]. There are two main look and processing options of SAP programs: reports and screens. The two kinds of program are different because the reports are generally reading programs, which collect data and present them in a list. The screen or transaction programs are designed to maintain data in the SAP systems. An SAP transaction (Logical Unit of Work) leads through more screens (or dynpro-s as the original name refer to dynamic program) collecting different data for a special purpose before booking it into the database. Each screen contains beyond the element list and attributes, the so-called flow logic, which has (at least) two main parts the Process Before Output (PBO) and Process After Input (PAI) event blocks.

As we described the SAP applications look similar. It is because SAP recommends using a standard style guide not only for standard SAP delivered programs and applications, but for customer developed once as well. For that SAP owns a Web site http://www.sapdesignguild.org/ containing the relevant information for programmers or designers.

SAP realized that the users want to have a bit easier, colorful, manageable surface, so as of release R/3 4.6 SAP implemented the so-called Enjoy controls to give new visual design for the end-user. The Enjoy controls are implemented as ActiveX controls in case of SAP GUI for Windows, but for JavaGUI SAP used JavaBeans instead.

SAP implemented Text Editor, HTML and picture viewer, tool bar, hierarchical tree controls and the so-called ALV (ABAP List Viewer) control, among others.

SAP GUI gives enhanced opportunities implemented on client side as well. In this case depending on the client PC setting the same SAP transaction, screen can look different. Two options are available currently:

- GuiXT
- GUI Scripting.

With the GuiXT screen elements can be hidden, moved, and new one pasted on the screen. This program runs with the SAP GUI process and modifies the look according to the script definition if available for the current screen. As an example an image can be placed to the screen be the following script row:

```
Image (20,30) "C:\Images\maci.jpg"
```

The local image file **C:\Images\maci.jpg** is displayed in row **20**, column **30**.

SAP GUI Scripting do not change the screen layout as GuiXT does. The main different is that the GUI scripting will not change the surface, but a script can be created manually or even by recording a SAPGUI execution and the script content can be change to execute again with other data.

SAP developed for non-standard display environments the SAPConsole client to support character-cell terminals, including radio frequency (RF) devices or even web-equipped devices. This is not a SAP GUI, but a different approach to handle special UI-s.

## IV. External, Developed Frontend options

As a communication layer SAP provides RFC enabled function to the outside world. These functions can be called from own developed programs or applications. The architecture figure shows above that the SAP communication through the gateway process, which handles the program-to-program communication to partner applications. SAP offers for external developers connectors to make easier the communication with the SAP:

- RFC Library: this is the classic RFC connector, which offers with its RFC API C-routines to create external RFC capability for own developments. RFC library makes it possible to create server or client RFC capable programs as well is C/C++ language.
- SAP Java Connector: with the SAP JCo Java application can communicate with SAP systems.
- SAP Connector for Microsoft .NET (SAP NCo): It makes possible to call SAP RFCs or BAPIs directly from .NET applications even it is written in Visual Basic, C++ or C#.

With these connectors it is possible to develop any required UI for SAP forgetting the standard SAP GUI and its possibilities. The basis of the developed UI is the RFC modules, which could provide data from and to the SAP system.

SAP introduced the Business Framework Architecture (BFA) to provide an object-oriented layer on the SAP application functionality. The public methods of Business Objects are the BAPIs (Business Application Programming Interfaces). For sustainability SAP freezes the interface definitions of the BAPIs. The older programs and external UIs can still use the old BAPIs. When an interface modification is needed, SAP implements a new method, new BAPI with the new signature.

SAP can be used as OLE (Object Linking and Embedding) server or as OLE client. For UI development we can use SAP as an OLE server. Visual Basic or even Excel can be used directly to call up SAP, log in and use according the login authorization the (remote) services of the system. SAP itself uses this as well for Business Warehouse (SAP BW) UI, because the SAP Business Explorer (BEx) is an Excel based solution (delivered as a front-end component with SAP GUI as well).

## V. SAP Web Frontend Options

SAP implemented around middle 90's a special Web enablement for the systems. During that time SAP had no Web Application Server technology as Fig. 2 shows, but a separated component was introduce to convert the SAP DIAG protocol to the Web HTTP protocol. This converter component is the Internet Transaction Server (ITS). SAP offers three programming models.

Earlier we mentioned that SAP GUI family has an SAP GUI for HTML, the so-called WebGUI. This is provided as a service by the ITS, which converts the standard SAP content to HTML Business functions.

The EWT (Easy Web Transaction) is a transaction, where the status of it takes place in the SAP system, because the basis of an EWT is a standard SAP transaction. Each of the screens (Dynpros) of the SAP transaction is translated in design time to so called HTML Business templates, which contain the screen element definitions using HTML Business functions. These templates can be modified and enhanced according to the Web requirements (e.g. images can be placed instead of buttons, etc.).

SAP ITS offers a third programming model, where the status of the transaction is stored on the ITS side. The ITS Flo Logic manages event flows, where the ITS can call back to the SAP system to retrieve or store data if the process flow requires it, but the screen definition and the flow logic is totally defined on the ITS side. ITS can use only remote enabled function for this purpose, so only RFCs and BAPIs can be used for this task.

As the Figure 2 shows the new SAP Web Application Server itself contains a built in Web server and with it offers beyond the old DIAG and RFC protocols many new a standard protocols like HTTP, SOAP, SMTP, etc. The main component of the Web enablement is the Internet Communication Manager (ICM) and the framework (ICF) around it. The services offered by a Web Application Server are listed in the ICF. As a special service SAP offers the so-called Integrated ITS as a service as well. So the Figure 1 have the ITS offerings in the presentation layer because of the service. The `webgui` service is also offers as a standard service without installing any other components for it.

In the ABAP Web AS SAP introduced the page based server side scripting Web programming tool, the Business Server Pages (BSP). This was a real internal development tool and environment for Web UIs using local (e.g. database, file system) or remote (other SAP or non-SAP systems remote enabled functionalities) resources for data retrieval and storage. The BSP has two possible scripting language: JavaScript and ABAP. Because of the ABAP scripting capability this Web tool has much higher interest than ITS ever (the old, experienced ABAP programmer can learn and program it easily) had. In the last releases SAP finished the development of standard Web transaction using BSP, but SAP still recommends using it for free-style Web programming for the customers.

SAP implemented first in J2EE Web AS and later in the ABAP Web AS as well the WebDynpro technology. The WebDynpro name come from the old, standard SAP screen name: dynpro. It uses the ModelView-Controller paradigm, and the definitions are stored as meta data and the system generates the corresponding code from this meta-data to create the real WebDynpro components. The WebDynpro components are reusable and according to the M-V-C paradigm they are sustainable because the look, the navigation, control and the business data management (model) are separated. The best way to keep the component unchanged during modification is to define the model as methods of a class holding the state of the WebDynpro component. The methods are embedding tools for real services calls. A service call can be local real service, database extraction, managing files, or any remote service (RFC, BAPI from other SAP system, or Web service) consuming. WebDynpro cannot be used so freely as the BSP, because of the meta-data concept. Meta-data serves as the definition of the layout, screen elements, navigation, data exchange via hierarchical data store called context. The developer writes ABAP code only in special cases, like the real business logic takes place, or entry checks are needed. This makes easier to use different UI device for WebDynpro, because from the meta-data different, device dependent code can be generated by the SAP system. On the other hand it leads to restricted surface option, only pre-defined elements can be used and the outlook is also pre-defined (though if SAP NetWeaver Portal is used, the WebDynpro elements can take over the portal design for having same style).

## VI. How to Create Sustainable UIs

As we worked out the technical capabilities we can declare that the basis for creating sustainable backend functions for UI-s are the well-defined RFCs and BAPIs, which can be called as WebServices as well. The main point of the sustainability in time when we have changes in the system is to have encapsulating, remote enable proxy services. These services can build the model level of the M-V-C paradigm to make a reusable and embedding layer for the UI-s. This makes possible not to change anything in the developed UI if any modification occurs behind (like upgrade, service modification), because the only the content of the embedding service should be modified if needed. This guarantees a more sustainable environment, because the end-user does note fill any changes on the UI, even if the back-end functionality is modified [2-3].

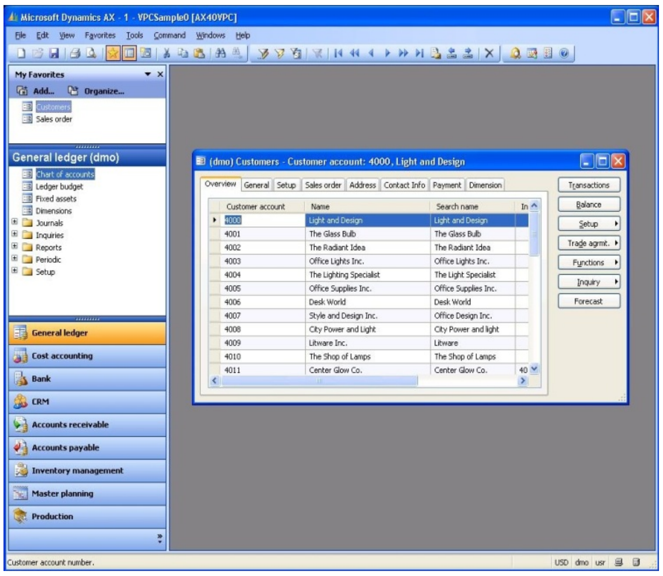
The opposite side also remains unmodified against the changes, if we consider a service or functionality on the back-end side stable, but we switch from a UI option to another one. As we learned above the almost all internal, standard SAP UIs can use any Workbench object, which provides functionality as model. But we can use remote-enabled function modules as well to implement model functionalities to guarantee the independence from the UIs (even external UIs). If we think about not RFCs, the standard Dynpro, ITS EWT, BSP or WebDynpro can be used as well. If we want to switch from one to the other the control and view layers should be redesigned, but the model can stay as it is. Using RFCs we have the opportunity to switch to ITS FlowLogic or use separated Web surface, which can be an external one or another SAP Web AS based UI (like ABAP or Java WebDynpro). This changeability on UI side leads us beyond the UIs, because it tries to uncover the power of design and development methodologies advised for sustainable and maintainable solution in a changing environment. These innovations for development strategies help us to generate better change management in a heterogeneous system landscape as well.

## VII. Dynamics AX User Interfaces

Microsoft uses a different user interface in Dynamics AX 2012, which is a logical way of development in AX 2009. The Enterprise Portal X++ Web User Interface Framework first shipped with Navision-Damgaard Axapta 2.5. It includes a set of nodes in the AOT, such as Web Forms, Web Reports, and Weblets, for defining the Web user interface components. It also includes a set of X++ kernel classes for reading and generating HTML from these elements and renders them as Web pages. In addition, it includes a set of Microsoft Dynamics AX Web Parts (Web Forms Web Part, Web Reports Web Part, Web Menu Web Part, and the Generic Web Part) and a set of out-of-the box application pages in Enterprise Portal that use these Web Parts.

With Microsoft Dynamics AX 2009, a new Web User Interface Framework based on ASP.NET, with managed APIs, Visual Studio-based development tools, and out-of-the-box application pages in Enterprise Portal built with this new Web User Interface Framework. For backward compatibility, the X++ Web User Interface Framework shipped along with the new ASP.NET Web User Interface Framework in Microsoft Dynamics AX 2009.

Since Microsoft Dynamics AX 2009 contains no dependencies on the Enterprise Portal X++ Web User Interface Framework, and future investment is focused on technologies built on the ASP.NET Framework-based Web User Interface, the Enterprise Portal X++ Web User Interface Framework will not ship in future Microsoft Dynamics AX releases Fig.1.

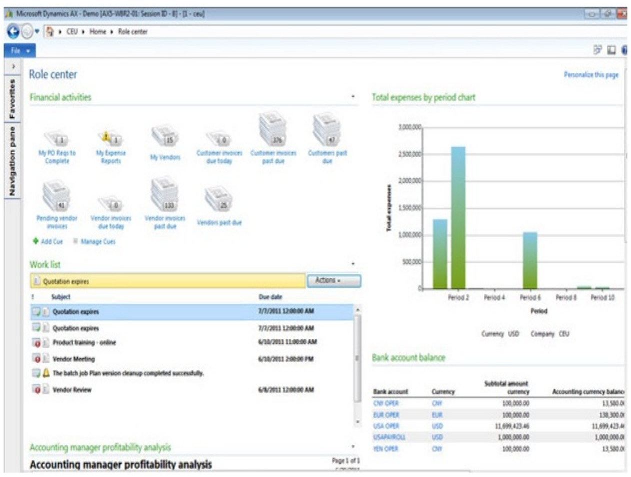


| Customer account | Name | Search name | In |
|---|---|---|---|
| 4000 | Light and Design | Light and Design | |
| 4001 | The Glass Bulb | The Glass Bulb | |
| 4002 | The Radiant Idea | The Radiant Idea | |
| 4003 | Office Lights Inc. | Office Lights Inc. | |
| 4004 | The Lighting Specialist | The Light Specialist | |
| 4005 | Office Supplies Inc. | Office Supplies Inc. | |
| 4006 | Desk World | Desk World | |
| 4007 | Style and Design Inc. | Office Design Inc. | |
| 4008 | City Power and Light | City Power and light | |
| 4009 | Litware Inc. | Litware | |
| 4010 | The Shop of Lamps | The Shop of Lamps | |
| 4011 | Center Glow Co. | Center Glow Co. | 40 |

Figure 1. General Ledger in AX 2009

Business deploying Dynamics to cover required functionalities needs continuous improvement and changes in functional and development framework of MS Dynamics System.

The next generation of Dynamics AX 2012 contains a new framework of user interface, which is called Role Tailored User Experience. Fig.2.



Work list

Quotation expires

| ! | Subject | Due date |
|---|---|---|
| | Quotation expires | 7/7/2011 12:00:00 AM |
| | Quotation expires | 7/7/2011 12:00:00 AM |
| | Product training - online | 6/10/2011 11:00:00 AM |
| | Vendor Meeting | 6/10/2011 2:00:00 PM |
| | The batch job Plan version cleanup completed successfully. | |
| | Vendor Review | 6/8/2011 12:00:00 AM |

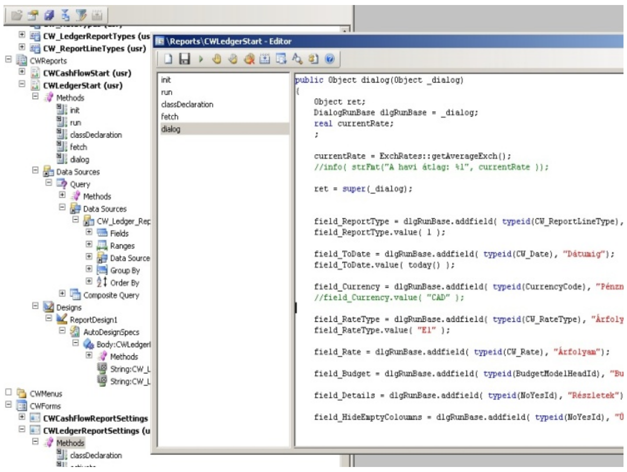
Bank account balance

| Bank account | Currency | Subtotal amount currency | Accounting currency balanc |
|---|---|---|---|
| CNY OPER | CNY | 100,000.00 | 13,580.0 |
| EUR OPER | EUR | 100,000.00 | 138,300.0 |
| USA OPER | USD | 11,699,423.46 | 11,699,423.4 |
| USAPAYROLL | USD | 1,000,000.00 | 1,000,000.0 |
| YEN OPER | CNY | 100,000.00 | 13,580.0 |

Figure 2. Role Tailored user homepage

This approach makes the user interface more friendly and useable for experienced Windows and Office users. The familiar experience extends even further to drive productivity for analysts working with data from Microsoft Dynamics AX 2012, interoperability with Microsoft Excel spreadsheet software and PowerPivot brings that data into those tools that analysts expect, with a native, natural experience enabling rapid, effective access to business system data. Such extensions significantly improve the efficiency of ERP System not only focusing on direct efficiency but also addressing the problems of the efficacy and effectiveness [4].

Workflow automation, built on the .NET Workflow Foundation, brings user tasks to a central task pane, including tasks assigned to an individual, their role, or a specific functional queue.



```
public Object dialog(Object _dialog)
{
    Object ret;
    DialogRunBase dlgRunBase = _dialog;
    real currentRate;
    ;

    currentRate = ExchRates::getAverageExch();
    //info( strFmt("A havi átlag: %1", currentRate ));

    ret = super(_dialog);


    field_ReportType = dlgRunBase.addfield( typeid(CW_ReportLineType),
    field_ReportType.value( 1 );

    field_ToDate = dlgRunBase.addfield( typeid(CW_Date), "Dátumig");
    field_ToDate.value( today() );

    field_Currency = dlgRunBase.addfield( typeid(CurrencyCode), "Pénzn
    //field_Currency.value( "CAD" );

    field_RateType = dlgRunBase.addfield( typeid(CW_RateType), "Árfoly
    field_RateType.value( "E1" );

    field_Rate = dlgRunBase.addfield( typeid(CW_Rate), "Árfolyam");

    field_Budget = dlgRunBase.addfield( typeid(BudgetModelHeadId), "Bu

    field_Details = dlgRunBase.addfield( typeid(NoYesId), "Részletek")

    field_HideEmptyColoumns = dlgRunBase.addfield( typeid(NoYesId), "Ü
```

Figure 3. AX development environment

Microsoft .NET Framework is integrated with Microsoft Dynamics AX in the following ways Fig.3.

X++ code can now be compiled into the common intermediate language (CIL) of Microsoft .NET Framework, but it is only for the AOS server. This enables services and batch jobs to run faster.

.NET interop to X++ is now supported. Microsoft Visual Studio can now automatically generate C# (or Visual Basic) code for proxy classes that wrap X++ classes and tables. It means that the metadata is now open to Microsoft Visual Studio.

Events are now supported in X++. The AOT provides useful nodes that make implementation easier. Events can be used to reduce the need to change custom code when you upgrade to newer versions of Microsoft Dynamics AX. Event handler methods can be written in X++ or in any Microsoft .NET Framework language.

Attribute classes can be used to add metadata to classes, tables, and methods.

The X++ language has the new keywords **as** and **is**. You can use the **is** keyword to test for inheritance relationships during run time. You can use the as keyword to verify the correctness of your downcast assignments during run time. There are many steps in bringing traditional X++ and MorphX IDE to be closer associated with Microsoft .Net technologies and MS SQL Server Reporting Services.

Developing code that integrates with Microsoft Dynamics AX is now possible in Visual Studio. The full development and deployment scenario is supported for managed code. This means, that a developer can easily add X++ objects to your Visual Studio project and write managed code that accesses those objects, can create event handlers in managed code as well as X++, and can write .NET code to handle class events. Managed code assemblies are automatically deployed to the location that you specify: client, server, Reporting Services or Enterprise Portal. You can specify where the assembly is deployed by setting project properties in Visual Studio. After you deploy an assembly, you can then see the managed code classes via IntelliSense and call the managed code from X++. Managed code is now supported by the Cross-reference Tool in MorphX. This means you can see the same cross-reference information for both X++ and managed code. For example, when you access the Cross-reference Tool from a report, you can see what the report data source.

Just as an X++ class could inherit from another X++ class in Microsoft Dynamics AX 2009, a table can inherit from another table in Microsoft Dynamics AX 2012. As in earlier releases, the Common table is still the base table of every table. A table that inherits from a base table is called a derived table. The terms parent table and child table describe foreign key relationships, not inheritance.

## VIII. SUSTAINABLE DEVELOPING IN DYNAMICS AX

At the beginning of the developing status of the first Axapta release, designed by Damgaard Software, there was an idea of three tier solution to isolate the code from database and from the operating system. This idea was similar to ABAP in the case of SAP R/3. From the viewpoint of database neutrality, recent versions are available on Oracle and Microsoft SQL platforms.

We can see substantial investments in all Dynamics ERP solutions, mainly in application user interfaces. The custom programming of user homepages goes forward .Net and SOAP web services. The consolidation of user interfaces goes to point in the future, where the user will be able to use the interfaces of different ERP products, regardless it is an AX, GP or other Dynamics ERP product. For software developer this trend means that generic .Net C# programmer should be able to modify the application via .Net interop. Microsoft Dynamics ERP project is trying one small step at the time by aligning all its inherited Corporate ERP application to have similar intuitive user interface and better exposure to .Net.

Developers are used to the new Microsoft platforms as .Net, CLR and also classical COM+. If a company has decent knowledge of MorphX X++ and has experienced Microsoft Visual Studio programmers, then this organization has the ability to successfully customize AX 2012 or 2009 versions. The newer releases and updates make the investment in developing necessary in the near future for every organization. Microsoft solutions often make objects and function, which were heavily used in previous versions, neglected in newer releases. This means that future version may not be compatible with the current .Net based programming module.

On the other hand MorphX and X++ developing environment have proven records to be more stable and neutral to the short term waves of computer technology trends. So it is recommended to insert the business logic more into these technologies. Let other companies to do the beta testing and debugging of new technologies, and when there is a solid version that will be the time to step up. There is a mixed market for Axapta version, current is AX 2012, but the AX v4.0, 3.0 etc. are still working. This case is somehow similar to the previous stage of Oracle, when Oracle Financials were competing with PeopleSoft.

Right now the way towards sustainable developments in Dynamics AX is using more MorphX and X++ for containing the business logic [5-6]. Newer technologies have to be adapted with care.

## IX. Conclusion

SAP and Microsoft ERP and other platform-based solution have many options to build a well-defined UI surface using difference data source and storage. Both solutions can provide services to consume by other application. With this property it is easy to create in a heterogeneous environment smoothly embedded new transactions, applications, which use services from other solutions for back-end functionalities. SAP can offer RFCs, BAPIs and Web Services for external use, so MS AX is a good candidate to profit from this offering for its developed UIs or UI extensions. The available RFCs and BAPIs services can be consumed only if the SAP NCo (SAP Connector for Microsoft .NET) is available for the developer. If no SAP connector is applied, Web Services (embedded RFCs and BAPIs as well) offered by SAP are consumable by AX.

SAP is not prepared for consuming services offered on any protocols. SAP can consume only provided Web Services and RFCs (and some other specialties not mentioned here). Definitely RFCs are provided mainly by SAP systems, but if a corresponding connector is in use, other applications, programs can also provide remotely callable functions for SAP. MS AX generally provides Web Services, which can be consumed by SAP applications based on SAP Web AS. But using the SAP NCo for Microsoft AX, not only Web Services, but also RFCs can be provided as well. When SAP consumes a Web Service or RFC provided by MS AX, different UIs, even standard ABAP Dnypro (screen based transaction) or WebDynpro can be based on them.

The interoperability and the service oriented new architectures and techniques make it easier to create application-wide transactions using UIs required. These UIs are maintainable and sustainable if the services behind are embedded to proxy classes or internal services.

## Acknowledgment

The authors gratefully acknowledge the grant provided by the project TÁMOP-4.2.2/B-10/1-2010-0020, Support of the scientific training, workshops, and establish talent management system at the Óbuda University.

Special thanks to Marianna Machata for her contributions.

## References

[1] Rita Mátrai, Zsolt Tibor Kosztyán: "A New Method for the Characterization of the Perspicuity of User Interfaces", Acta Polytechnica Hungarica Vol. 9, No. 1, 2012

[2] Igor Barbaric, *Design Patterns in Object-Oriented ABAP*. Galileo Press Bonn-Boston, 2010

[3] Rich Heilman, Thomas Jung, *Next Generation ABAP Development*. Galileo Press Bonn-Boston, 2007

[4] L. Muka; G. Lencse. "Developing a meta-methodology for efficient simulation of infocommunication systems and related processes", Infocommunications Journal, Vol. LXIII, No. 7. 9-14., 2008.

[5] Dynamics AX Library - http://social.msdn.microsoft.com.

[6] Dynamics AX Community Network - https://community.dynamics.com/product/ax/default.aspx