Is Enterprise Application Integration still a multi-million investment burden for companies?

Daniel Kutáč*
daniel.kutac@intersystems.cz

Abstract: This paper describes recent situation in integration platforms market.

First, it briefly analyses history of business integration from „rip & replace" approach to IT modernization through an era of deploying large integration suites by Enterprise Application Integration (EAI) pioneer companies. Subsequently it focuses on a new trend of reducing integration cost by employing compact modern unified products of EAI market newcomer companies suitable even for small and medium enterprises with smaller budgets and shorter timeframes.

Key Words: ERP, Integration Suites, UBIP, Workflow

1 History of Application Integration

We are not going to define term "Enterprise Application Integration" here. Many others did that already, let's name [1] for all. We will, instead, summarize historical development of integration originally standalone applications into one compound, interconnected unit.

1.1 Pioneer Enterprise Applications

Originally, enterprise applications of 1960-1970th were designed to serve just one purpose – to replace tedious, manual, repeating work. They used to be at the beginning relatively simple in their functionality and in user interaction via terminals. At the time of origin of these systems there was no notion of interconnection needs of these "isolated application islands".

Later, in 1980th, together with introduction of more advanced and faster communication technologies, companies realized that they need not only partial picture of individual departments but need consolidated information from all applications. Hand made, manual consolidation from old days proved to be not sufficient and too much error prone for achieving new goals.

1.2 ERP Application Era

As a solution for the growing consolidation needs, a ERP (Enterprise Resource Planning) applications evolved. They usually originated from single purpose applications by reprogramming and extending their functionality so they covered most aspects of the enterprise activity. These packages usually encompassed resource management, production management, mostly with accounting bindings. Leading representative of these packaged applications became company called SAP.

* InterSystems B.V, Vlnitá 31, CZ 147 00 Praha 4

Nearly all 1990th were characterized by massive deployment of such "whole covering" solutions. But, towards 2000 it became clear, that deploying such universal solution is not beneficial. Quite a big deployment attempts lasted much longer, cost much more money and brought less functionality that expected.

That was one of reasons for companies, together with industrial recession, to look for other alternative solutions. One of requirements was that such solutions shall retain existing, specialized applications, and leverage from their proven functionality and join them together and bring a new value. That was the dawn of integration tools massive deployment.

1.3 Integration Tools

With certain level of simplification, we can say that integration tools embrace following levels:

1.3.1 Data Integration

Data integration is base pillar of integration tasks. Data must be located (using database connections, files...), identified (tables, files...) and cataloged. This is accomplished by building common metadata model that abstracts data source from their physical location.

1.3.2 Application Integration

Application integration task it to bind them together at the data level in real-time, if possible. Application integration embraces e.g. connection of CRM (Customers Relationship Module) with ERP system and distribution management system, interconnection of order processing system with selected customers' systems, building a web portal for customers or in-house use or so. Last, but not least, application integration includes new functionality development or development of intermediary connectors between proprietary and new systems.

1.3.3 Business Processes Integration

Business Processes Integration represents highest level of application interconnection; not only at data level but at operational level by defining, implementing and managing business processes. Existing processes can be streamlined by determining and implementing their software model. This results in cost cutting and more efficient workflow management.

1.3.4 Business Activities Monitoring

Even most perfect business processes interconnection would hardly make any sense without continuous knowledge of processes state. Monitoring is responsible for automatic - driven by predefined rules - reaction resulting from any non standard situation.

The above given categorization of integration levels also describes integration history, from oldest task - data integration, until most recent - business processes monitoring.

Very first solutions, that can be considered direct ancestors of nowadays integration platforms, were so called message brokers. Message brokers are tools providing data interchange between integrated application and guaranteeing information delivery according to predefined rules. These tools can be found as back as in 1970th and still represent a core of any integration tool.

Following companies profiled during years as leading integration solutions providers:

Tab 1.3.4.1 Leading integration solutions providers

IBM	WebMethods
SeeBeyond	Vitria
Tibco	MQSoftware
BEA	

All products of listed companies share one common characteristic – they are conglomerates (suites) of originally partial integration level solutions. Namely first three of those listed. This is result of historical development, as many of integration tools were developed by small companies and later swallowed by bigger ones and incorporated into their products.

Integration suites bear in their nature one big disadvantage, separate sets of metadata, user interfaces and configurations for each and every tool from the suite. Thus, integration by using suites brings a need to integrate first of all the individual tools of the suite and only then, integration of applications themselves. Needless to say this makes the whole integration process very complex and time consuming.

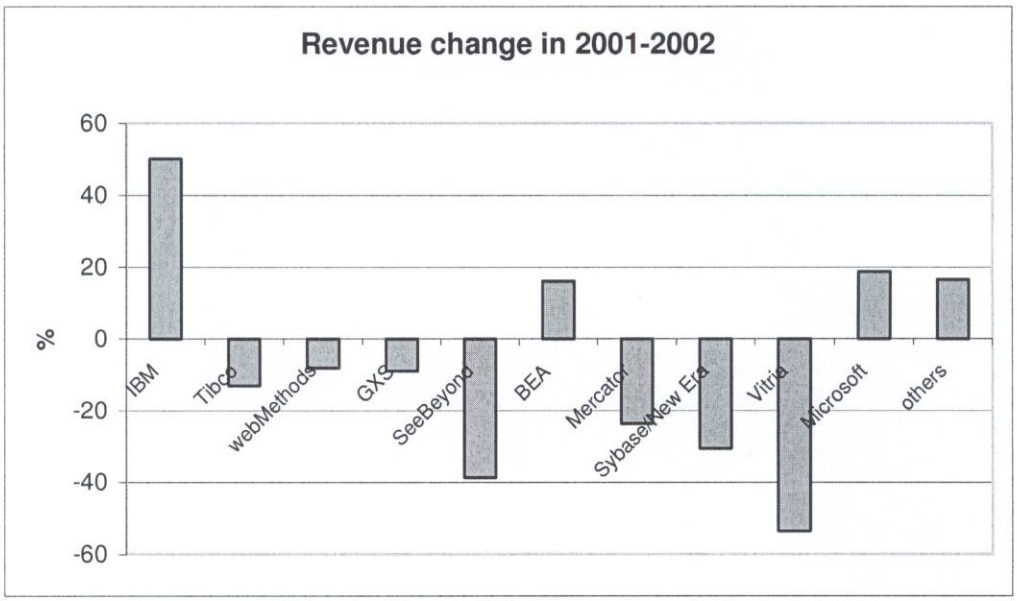
A brief description of basic characteristics of some of integration tools can be found in [2].

It is obvious, that companies have no power and skills to manage integration using suites on their own. They have to rely on external companies – integrators. Integrators have deep knowledge of various business environment and they know at least some of the above listed products. However, integrator services are very expensive and after initial revenue growth from integration services a break-event appeared some 3 years ago.

2 2002 Integration Market Broke

Somewhere around 2002 integration market broke. It turned out that companies were unable to spend multimillion amounts for top level consulting services and for deployment of very complex integration suites. This fact demonstrated itself especially by traditional implementation solutions providers. Following graph shows, that except IBM, who is both supplier and consultant, most other traditional suppliers revenue stagnated or even fell down [3] [4].

Fig. 2.1 EAI solution providers revenue change



The graph also illustrates positive growth for BEA and Microsoft who both joined the market recently.

In line with emerging XML and Web Services, many companies, who were originally known as application providers, like SAP, Magic Software and others, presented own integration tools as extensions to their applications creating "embedded integration" business. However, even though these solutions are more compact then classic tools, their architecture is still using traditional suite approach.

3 UBIP

The motivation of InterSystems Corp. for entering EAI solutions market was driven by its customers. The company has been more and more involved in scenarios, where its customers who used InterSystems post-relational database and application server Caché, utilized Caché engine to build routing and transformation components to connect applications written in Caché with other applications. Having learned from mistakes of traditional EAI providers, InterSystems decided to design its integration solution as one, uniform, compact set of tools that looks like a single application yet providing all functionality that can be found in classic integration solutions. This was origin of Ensemble, InterSystems integration product. InterSystems are calling the architecture of its integration tool "UBIP" - Universal Business Integration Platform.

The principal characteristic of UBIP is use of centralized, multidimensional, multi-view (Objects, SQL, XML) data storage with virtual machine. Unlike in other integration tools, database is not separated (federated) out of the execution engine, but embedded into this engine making one, indivisible core of the product. This core, that stores all metadata and exchanged data itself, and in which all processes are executed, communicates by means of native scripting languages with all Ensemble components

This paper, however, is not going to describe Ensemble in high-level technical details, those who are interested, can consult e.g. [2].

According to [5], UBIP architecture is characterized by these words "By abstracting the relevant interfaces, data and business logic models of these independent, back-end processes into a single, consistent and efficient canonical form, the complexity associated with integrating heterogeneous systems is dramatically reduced".

We will explain in following paragraphs meaning of previous sentence.

All exchanged (integrated) data is transformed by means of inbound adaptors into persistent classes representing messages. These messages are entering exposed business services and then, on the other end, leave Ensemble as output messages (again persistent classes) via outbound adapters where they are transformed into the format and technology of receiving application.

Business services are Ensemble's API. These components are exposed as services as Ensemble is strictly using SOA (Service Oriented Architecture) concept. These services can interchange information with external applications using any technology, starting with COM, via .NET, Java to XML and Web Services.

Ensemble classes represent single, consistent and effective canonical form as cited above. Ensemble classes are objects per ODMG [6] definition. Persistent classes created as output of adaptors, are also Ensemble classes.

Business services are part of business processes model. Business processes are represented as persistent classes as well. Integration developers can describe these processes either by BPML (Business Process Modeling Language) [7] in XML or Graphical form or by Ensemble's Class Definition Language.

Data is exchanged between Ensemble components (business services, business processes, business operations, transformations etc.) by persistent classes – messages.

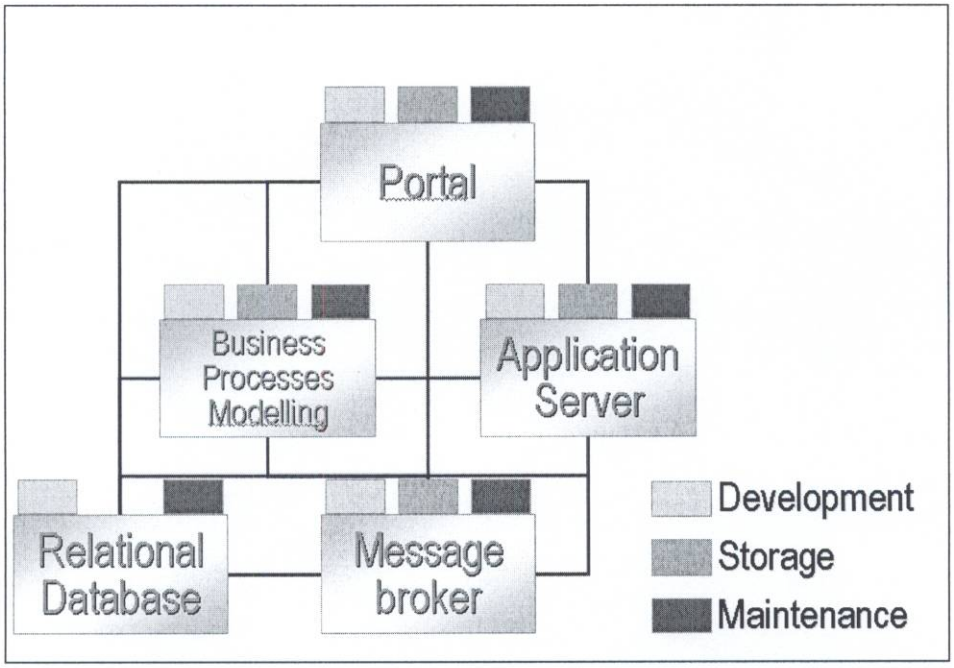
Business processes output is provided by calling business operations. Data is passed to operations by messages – persistent classes. Operations communicate with applications via outbound adaptors.

As it can be seen in the above text, all developers work in Ensemble consists of creating code in one of Ensemble native scripting language – ObjectScript [8] or Basic [9]. It is also possible to include SQL blocks into scripting language modules where feasible. All this dramatically decreases number of programming languages a developer has to know in order to utilize Ensemble. All Ensemble components (services, processes, operations, adaptors, etc...) represent a complete framework of classes that developers utilize when designing and implementing an integration scenario.

On the other hand, developers can easily build composite applications or reports using SQL, Java, .NET, Web Services, C++ or other technologies as Ensemble can talk to these directly via communication gateways or direct class projections into these programming languages.

Following diagrams compare UBIP architecture and integration suites.

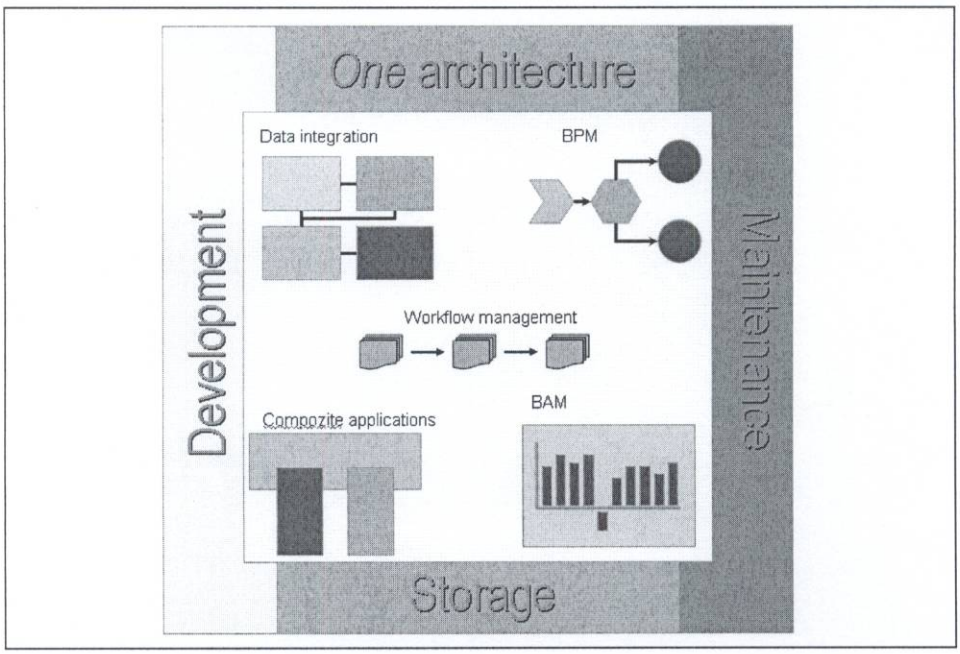
Fig. 3.1 Integration suite



An integration suite consists of tightly connected independent integration tools for different integration levels. As the diagram shows, each part of integration suite contains development tools, storage system for configuration settings and data and maintenance utilities. All these are usually based on different technologies, thus making the integration suite an element that itself increases "application entropy"* of the enterprise to integrate.

Fig. 3.2 UBIP Architecture

* read as "number of applications to integrate"



UBIP means that all integration levels are encapsulated into one homogeneous and compact product.

Besides fully automated business processes, there are often scenarios that require manual human intervention. Workflow management is integral part of UBIP architecture and as such is also implemented by persistent classes. Each action is recorded into built-in Ensemble database, each operator/user can see a list of open tasks. Based on task status assigned by operator, business process can continue execution by different path whether automatically or by another manual intervention.

Whole configuration and development of a scenario is done by a web based portal. The portal is built by a set of Ensemble classes. The portal is an open application, customizable by developers to best suit their needs.

Part of portal is Business Activity Monitoring (BAM) feature. Operators can define measures to monitor and watch them in graphical way as graphs, diagrams etc... These measures are, as all other parts of Ensemble, represented by persistent classes. Thus, operators have history of data/processes flow just next to their eyes.

One of the most important consequences of using object enabled storage for every message that passes Ensemble is that operators can instantly watch history, search for abnormalities and errors. Measures can be bound to alerts so when a predefined limit is broken an alert can be raised, e.g. as an email to operator.

The nature of UBIP architecture predestines Ensemble use in embedded integrations market. However, it can be used with the same ease for full scale EAI projects.

UBIP advantages brief summary

- Installation – there is just one installation package

- Compactness – all functionality is built in some 160-200MB depending on operating systems, including documentation
- Deployment speed – product is up and ready to use within some 10 minutes from installation start
- Minimal learning time – just learn API classes and how to use Basic or ObjectScript
- Built-in storage – enables auditing, business activities monitoring, enables business rules definitions and their change at runtime, no need to restart any process
- Technology independent solution, no need for Java or .NET or any other third party framework
- Fully independent of integrated applications

4 Conclusion

We have described briefly a history of applications integration, pointed out a breaking time period shortly after year 2000 that gave - together with emerge of service oriented architecture - birth to new generation of enterprise integration tools based on compact, unified and lightweight architectures. Among those, we have presented UBIP architecture, used by Ensemble, an integration product produced by InterSystems Corp.

References

[1] IDG: http://www.nwfusion.com/details/821.html

[2] Daniel Kutáč, Martin Zubek: *Ensemble – integrační produkt nové generace,* proceedings of conference Systémová Integrace 2004, Prague

[3] Global Information Inc.: http://www.gii.co.jp/english/wg12938_ application_integration_toc.html

[4] Network WorldFusion: http://www.nwfusion.com/news/2003/0714earnings.html

[5] *Business Integration Journal*, March 2004, http://www.intersystems.com/ensemble/analysts/BIJmarch.pdf

[6] ODMG: *ODMG 3.0 standard*, http://www.odmg.org

[7] BPMI: *BPML specification*, http://www.bpmi.org/bpml-spec.htm

[8] InterSystems Corp.: *Using Caché ObjectScript manual*, http://www.intersystems.com/cache/downloads/documentation/cache5docs/PDFS/GCOS.pdf

[9] InterSystems Corp.: *Using Caché Basic manual*, http://www.intersystems.com/cache/downloads/documentation/cache5docs/PDFS/GBAS.pdf