Evolutionary and Pervasive Services

Elia Conchione

Architect, Soluta.Net, Italy, EU
econchione@soluta.net

Abstract. The strategy is to have a system that does not move data across the architecture when it is not required. The goal is also to support different architectures, namely: Information, Model and Application. The first is to be stored and maintained where the information is actually created (by the service providers), the Model in a separate repository; models are horizontal respect to the information and are common across services. Application architecture is managed by the Registry and it 's used as a directory of services. This separation of concerns as implemented in this project helps to avoid data replication and it aims at providing a better alignment between IT and the business.

Keywords: Modelling, negotiations, e-procurements, MDA [1], p2p, peer to peer, metamodelling, agreement, SMEs.

1 Small Organisations and e-Business

The SMEs are the backbone of European economy, in the enlarged European Union of 25 countries, some 23 million SMEs provide around 75 million jobs and represent 99% of all enterprises.

The subprime mortgage problem, which emerged in the summer of 2007 in the United States, has led to global financial market turmoil and growing concern about a U.S. economic slowdown. This situation of course don't help our SMEs.

Inside this scenario, it has become increasingly important the ability of enterprises to manage innovation and the creation of ecosystems to cater for short-term, short notice collaboration. Enterprises of the future will be nodes in innovation ecosystems requiring flexible, spontaneous collaboration activities.

Being more follower than leader of the change process, small organizations seem to need favorable conditions to accelerate the diffusion of the Internet and adoption of ICT technologies [2] [3] and thus to avoid a digital divide between larger and smaller enterprises and among geographical areas [4].

The digital divide that really matters, then is between those with access to a network and those without. The good news is that the gap is closing fast. The EU has set a goal of 50% access by 2015, but a new report from the World Bank notes that 77% of the Europe's population already lives within range of a network.

2 The Two Digital Divides

At the Lisbon summit in March 2000, the European Union representatives set the goal of becoming the world’s most dynamic and competitive knowledge-based economy

L. Telesca et al. (Eds.): DigiBiz 2009, LNICST 21, pp. 149–155, 2009.

by 2010 with the need to promote an 'Information Society for All', and to address the issues of the digital divide in the adoption of Internet and e-business use [4].

Comparing big and small company, the gap as defined using these per-head measures looks enormous, the growth rates tell a different story. Over the past 25 years, computer penetration has been increasing faster in SMEs than in holdings, which has not been surprising given the software saturation in complex organization. But the same is also true of internet usage, which grew by around 13% per quarter in high-income firm in the late 1990s, compared with 10% per quarter in low and middle-income one. Holdings are ahead, but the SMEs are catching up fast.

3 There Must be a Better Way

When it comes to enterprise software, in particular, this way of delivering technology creates a somewhat perverse set of economics. Software is a service at heart, albeit an automated one, but it is sold much like a manufactured good. Company have to pay large sums of money up front, bear much of the risk that a program may not work as promised, and cannot readily switch vendors.

IT firms, for their part, have to spend a lot of resources on marketing and distribution, rather than concentrating on developing software that works well and is easy to use. In our markets it is a great advantage to be first, so vendors are tempted to release programs even if they are still riddled with bugs. And because equity analysts rightly consider software firms a risky investment, such firms must grow quickly to justify their relatively high share prices, pushing them to sell more programs than customers need.

All this explains several peculiarities of the software business[5]. One is the fact that many of the licences sold are never used, a phenomenon known as "shelfware". More importantly, many software firms have grown so fast, often mortgaging the future, that they collapse when they reach Euro 1 billion in annual revenues, sometimes never to recover. Then there is the end-of-quarter rush, spurring many firms to do anything to get deals signed and meet analysts' expectations [6].

4 Major Obstacles

Adoption of a new technology is often expensive for many reasons. The benefits gained by the user and the costs of adoption are notable. These benefits are simply the difference in profits when an organization migrates from an older technology to a new one. Benefits may also include such "non-economic" factors as the satisfaction of being the first on the block to be converted from the older technology to the new one [7].

Large organizations that have large market shares are more inclined to undertake adopting a new technology, both because they have a greater opportunity to gain profits from the adoption but also because the availability of funds to these organizations is greater [8].

More profitable organizations are more likely to have the financial resources required for purchasing and adopting a new technology. In addition, they may be better able to attract the necessary human capital and other resources that are necessary [9].

For all previous reasons the small organization should be supported in removing a set of obstacles. This requires a strategic long-term vision and planning. The key challenge for policy makers is to identify the European and national policies which directly address the removal of these obstacles, through specific actions.

The *e*Europe 2002 Action Plan was endorsed by EU Member States at the Feira European Council in June 2000. The Action Plan's objective includes an action to encourage SMEs to 'go digital'. Within this action, main obstacles SMEs face as they engage in e-business has been identified [10].

In IT vision, the lack of suitable technical and managerial staff with sufficient knowledge and expertise is a major barrier.

This shortage affects technical knowledge related to ICT, but also entrepreneurial and managerial expertise needed for operating in a networked economy.

5 The Solution

The proximity to sources of information and services is thus key to success, especially for small and medium enterprises. The proximity to facilities and information flows has been considered very important in the past years, and it is still useful to understand the growth and competitiveness of companies in the present.

The evolutionary and pervasive service platform implements the concept of Digital Business Ecosystem (DBE) [11] [12]: companies that join the community can register their services and make them discoverable by the other members of the community. The DBE thus realizes the concept of extended dynamic cluster.

It is very important to stress that this approach focuses on services and not on the technological infrastructure that is used for implementing and running business services: small and medium enterprises lacking the physical and technological infrastructure to promote and distribute their services are only required to formalize the interface of their services in order to be able to exploit the potential of the system. Once they do so, their services become visible and usable inside the ecosystem.

By using a peer-to-peer infrastructure, the project provides a service registry that collects information about business services provided by all the subscribed organizations and makes them discoverable and usable.

The main concept about this system is that anybody in the community can get to know the existence of a service and be able use this service; the importance of a service does not depend on the dimension of the enterprise which exposes it, but rather on its perceived usefulness. The ranking mechanism for services promotes the services that are used most frequently, or that have received a positive feedback by their users. Any organization in the community can decide to build its own service by using and modifying models of services that have a top ranking, introducing a next step in the evolution of the ecosystem.

A business service in the evolutionary and pervasive service platform can be whatever: it can be a flight ticket reservation system as well as flowers delivery online service. By joining the community, companies can enlarge their market visibility and improve the quality of services provided by cooperating with other actors.

6 Evolutionary and Pervasive Service

This is the solution and the implementationdesigned for the Evolutionary and Pervasive Service Platform; this solution takes into account the project solutions defined in the previous specifications and summarized as follows:

- decoupling between: model (IT and Business), service data, service endpoint
- alignment between business and IT: “What you see on the platform is what you get when you use it”
- support to the EOA (Ecosystem Oriented Architecture) paradigm: dependency, versioning, metamodel bases

In the architecture description we also present components that are not part of the Evolutionary and Pervasive Service Platform. These elements provide functionalities that are used by system actors so as to improve the components integration.

7 Architectural Strategies

The strategy is to have a system that does not move data across the architecture when it is not required. The goal is also to support different architectures, namely: Information, Model and Application. The first is to be stored and maintained where the information is actually created (by the service providers), the Model in a separate repository; models are horizontal respect to the information and are common across services. Application architecture is managed by the Registry and it 's used as a directory of services. This separation of concerns as implemented in this project helps to avoid data replication and it aims at providing a better alignment between IT and the business, details follow below.

Inside the Evolutionary and Pervasive Service Platform, registered services are associated to models that are stored in the Repository.

A registry entry contains only attributes that are used for: consuming the service, retrieving the business model instance, retrieving information about the service vendor, retrieving the WSDL associated with the service, retrieving models that are associated with the service

The service is thus described by models that are associated with it. An user that wants to find a certain type of services must first of all search the models that contain the desired contents, and then search for all the services associated with these models.

The Registry works like a DNS server, because it doesn't contain any meta-data that describes services but rather contains only information for associating a service to a model or an ID. Like a DNS server, it must have a fully decentralized architecture for ensuring availability and reliability, and also an automatic synchronization mechanism.

The Registry also manages the lease for registered services and uses this information to keep them discoverable. We believe this approach avoids the risk of an overload of “keep alive” requests from service providers willing to renew their services. Every registry node can use an ad-hoc algorithm for checking service availability and then unregister those that do not respond.

The Repository is used for storing models [13] and their descriptions, so it must use a database to save information and allow queries on meta-data.

We also introduce the concept of template (borrowed from Jini) to improve models search.

To describe models we use tags which are similar to folksonomies [14]. This is a way for creating a model description that is driven by the community growing around models development.

We deem the use of standards very important in a project, and that using WSDL for web [15] [16] services, the JAXR [17] API for storing models and the SQL language for performing queries is a guarantee of quality.

We use a data warehouse for performing data mining on the data which is collected by inspecting business model instances. The data warehouse can also contain model meta-data for performing powerful queries on clients' requests.

The information contained in business model instances is collected by a service crawler that retrieves them by querying the service provider, inspecting the business model instances and reporting information to the data warehouse.

8 Interactions between Components

Interactions between components are explained using *following picture,* Figure 1, which describe the steps that are performed by operations provided by the registry and by the repository.

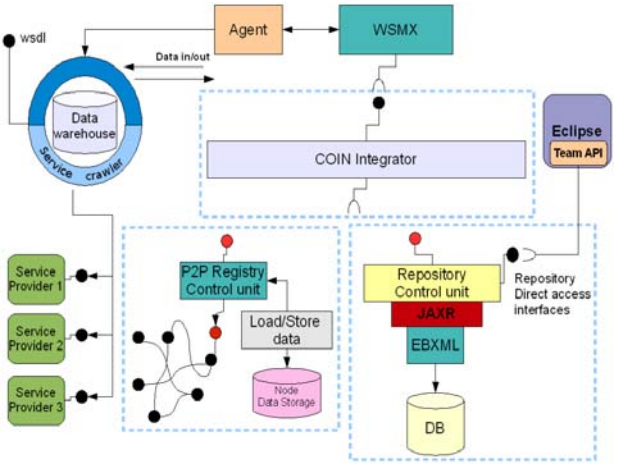

Fig. 1. Evolutionary and Pervasive Service Platform architecture

9 Registry

The Registry is used to store services that are available to users. Information is stored inside a RegistryEntry object.

The Registry is distributed over a P2P network, which provides the load balancing between nodes: every node contains ≈ *(1/n * total-data)* + *backups* (where n is the

number of nodes). This architecture is fully decentralized and so it doesn't have a single point of failure. When a node joins or leaves the network, the data is automatically redistributed to avoid data loss.

Registry functionalities are similar to DNS servers, which receive a domain name as input, and return the IP associated with it. The Registry, similarly, receives a ServiceID and returns the corresponding EndPoint which can be used for consuming the service.

The Registry also manages the concept of “lease”. After a service registration, a dedicated checker tests if the service is still alive once a predefined time slot is elapsed. The figure below illustrates this scenario [18].

The service provider must supply a special method that responds to a “check” request. If the service doesn't respond, the entry is deleted from the Registry.

The lease is managed at Registry side, because we want to avoid the critical situation of overload that may happen when all the services perform the lease renew at the same time. With the proposed solution, nodes manage the lease using a built-in ad-hoc strategy.

10 Repository

The Repository is used for storing models and meta-models.
The Repository [19] realizes the evolutionary concept, that is summarized below:

- supports the tracking of model versions, extensions and copies: models are never removed/modified, a change is a copy with a link to the original, model, models gain “ageing” concept
- supports model dependencies: solving “Versioning Hell“ is one of the goals that the repository is targeting, models gain “context aware” information, the more connected, the more valuable

Users can also get access to the Repository via a special interface that exposes the services using the WSDL standard. This solution allows custom plugins development, like a new one that performs the component integration between the “Repository control unit” and the “Eclipse Team API” [20].

11 Data Ware House

This module, that is external to the Evolutionary and Pervasive Service Platform, collects information from the repository and also from the business model instances that are located on the service providers.

Operations that involve data mining are performed inside this component for improving performances and effectiveness. Custom data mining algorithms can be implemented inside it. This choice improves performances because queries regarding service properties, like discount percentage, products amount etc., are directly submitted to a dedicated optimized repository.

The current status is maintained by a service crawler, that queries services providers and collects information inspecting the business model instances that are returned.

12 Summary and Conclusions

Evolutionary and Pervasive Service Platform is the founding principle that drives the entire architecture[21] of the Open Negotiation Environment, this allows the solution to functionally scale, to have software components that are not bind to any specific negotiation and are adaptable to custom needs. P2P designs have a lot of advantages this has provided also benefit in the daily organization of the work easing the interoperability issues among partners and software components.

The architecture adopts a decentralized approach to avoid having a central node (a single point of failure) this would jeopardize the adoption of the platform. The architecture is configured in order to behave like a self supporting organism that grow over time: nodes of participant organizations are attached to each other sharing resources.

The technologies for the decentralization are built around p2p protocols, RESTful and JAXR iso standard.

By using a peer-to-peer infrastructure, the project provides a service registry that collects information about business services provided by all the subscribed organizations and makes them discoverable and usable.

References

1. Many, Model Driven Architecture
2. Association for Computing Machinery, Transactions on Internet Technology
3. Many, Rest Representational State Transfer
4. Nachira, F.: Towards a Network of Digital Business Ecosystems Fostering The Local Development (2002)
5. Stanoevska-Slabeva, K., Faust, R., Hoyer, V.: D1.2 – Software Requirements and Use Cases (2007)
6. The Economist (2003)
7. Hall and Khan (2003)
8. Bronwyn and Beethika (2003)
9. Dorfman (1987)
10. European Commission (2000)
11. Digital Business Ecosystem Project
12. Pierfranco, F.: Architecture for Digital Ecosystems: Beyond Service-Oriented Architecture (2007)
13. Many, Omg Document Number: Formal/2007-12-01, Mof 2.0/Xmi Mapping (2007)
14. Many, Folksomy
15. Fielding, R.T., Taylor, R.N.: Principled Design of the Modern Web Architecture (2002-2005)
16. Pautasso, C., Zimmermann, O., Leymann, F.: Restful Web Services vs. Big Web Services: Making the Right Architectural Decision (2008-2004)
17. Many, The Java Api for Xml Registries (Jaxr),
18. Many, Hazelcast, Clustering And Highly Scalable Data Distribution Platform for Java
19. Many, Electronic Business Using Extensible Markup Language
20. Many, Eclipse
21. University of California, Irvine, Architectural Styles And The Design Of Network-Based Software Architectures, Doctoral Dissertation