An ERP System Based on Peer-to-Peer-Networks and Web Services

Jorge Marx Gómez
Department of Computer Science, Technical University of Clausthal, 38678 Clausthal-Zellerfeld, Germany, Julius-Albert-Str. 4,
Tel: +49-5323-72-7115, Fax: +49-5323-72-7139, gomez@informatik.tu-clausthal.de

Oliver Krüger
Department of Computer Science, Technical University of Clausthal, 38678 Clausthal-Zellerfeld, Germany, Julius-Albert-Str. 4,
Tel: +49-5323-72-7115, Fax: +49-5323-72-7139, oliver.krueger@informatik.tu-clausthal.de

Claus Rautenstrauch
Department of Computer Science, Technical University of Clausthal, 38678 Clausthal-Zellerfeld, Germany, Julius-Albert-Str. 4,
Tel: +49-5323-72-7115, Fax: +49-5323-72-7139, rauten@iti.cs.uni-magdeburg.de

ABSTRACT

As the business world gets more and more dependent on digital technology, including information systems for resource management, even the small- to medium-sized enterprises have to install and maintain complex enterprise resource planning (ERP) systems. However, these are designed as an all-in-one solution, often implementing functionality not needed. Furthermore, ERP systems like SAP depend on very large-scale infrastructures like servers and networking technology, which are very expensive to install and to maintain. Customizing these large-scale systems to the needs of a small- to medium-sized business is nearly as expensive as the customization for a large enterprise and therefore not affordable for these companies. Because of this, in this paper we present a design for a distributed ERP system, which is based on Web Services and peer-to-peer technology. It is easier to install and to maintain and cheaper than the traditional solutions.

INTRODUCTION

Today's enterprise resource planning (ERP) systems are designed as a system compromised of databases and application servers which provide all desired functionality through a monolithic application. The whole system is then adapted to the company's needs by a complex process called customization. But there are several disadvantages of this classical design of ERP applications:

• High-end computers needed: Application and database servers require high-end hardware, especially if failover support is needed.
• Customization process is expensive: Highly skilled people have to go through the whole system and adapt it to the company's needs. This is a very lengthy and expensive process.
• Complex system management: Lots of servers have to be administered for which in turn skilled administrators have to be employed.

By market pressure, small- to medium-sized enterprises (SME) are forced to deliver highly customized and high quality products. Thus the use of ERP systems is necessary. However, these enterprises do not have the budget for installing, customizing and maintaining complex ERP systems.

A solution to this problem is application service providing (ASP). Service providers host and administer all equipment associated with the ERP system. In turn, the enterprises have to move their valuable data to the service provider’s data center. Problems of security, especially trust, have prevented this idea from becoming successful. The second option are pre-customized systems, which are addressing the customization process but require the same complex and expensive maintenance.

Our proposed solution, which we will be outlined here, is the use of an “out-of-the-box” computer with preinstalled software, which will use web services to expand the functionality of the whole system where needed and is much cheaper to install and maintain then the already available solutions without having to move valuable data to third parties.

BUSINESS COMPONENTS

The key element in our approach is a business component. A business component (BC) is a component which offers a specific set of business services from its business domain [Tur01]. The three main characteristics of (business) components are:

• Well-defined interfaces: In order to offer certain services to its environment this attribute is crucial. Communication with the outside world depends on standardized interfaces.
• Reusability: Due to its well-defined interfaces the BC can be reused in different contexts.
• Ability to be combined: Also due to its well-defined interfaces BCs can be loosely combined with other BCs into a business application system to fulfill complex tasks.

According to [Tur01] BCs pass through a life cycle:

1. Standardizing: First the BC needs to be standardized to get its well-defined interfaces (technically and domain-related).
2. Development: Standardized BCs can be developed by different (competing) vendors.
3. Adaptation: BCs require technical and domain-related adaptation in order to handle implementation-dependent incompatibilities and to customize its business services. In an ideal case the expense of this adaptation is minimal.
4. Composition: Different BCs are composed together to form a whole ERP system.
5. Evolution: Meaning the adaptation after its installation. From the users perspective this should not be necessary. In the case of an inappropriate BC (assuming a changing environment) the user should be able to just replace the BC with another one.
WHAT ARE WEB SERVICES?

Right now there is no generally accepted definition of what a web service really is (see [Ber03]). Within this paper we assume the definition of the workgroup “Development of web service based applications” by the Gesellschaft für Informatik (GI) as: “Web Services are self-descriptive, encapsulated software-components, which are offering an interface for remotely calling their functionality and can be loosely coupled by the exchange of messages. For achieving universal interoperability, standard internet technology is used for communication.”

For actually implementing web services, there are many different concepts, protocols and models available, which are being standardized by popular institutions, especially the World Wide Web Consortium (W3C) and the Organization for the Advancement of Structured Information Standards (OASIS).

In our proposed solution, following, on XML-based standards will be used to achieve maximum interoperability and standard-compliance:

- Simple Object Access Protocol (SOAP)
  SOAP is a standard lightweight protocol for exchanging messages, especially for invoking methods, between applications. SOAP can be used on top of many protocols, for example SMTP or HTTP. Depending on top of which protocol SOAP is used, the message exchange can be asynchronous (e.g. by using SMTP) or synchronous (e.g. by using HTTP). Many toolkits are right now available for SOAP and it is supported by the many development environments like J2EE by Sun [Sun03], PHP [PHP03] or .NET by Microsoft [MS03].

- Web Service Description Language (WSDL)
  WSDL, as defined by the W3C [W3C03], is a language for describing the capabilities of a specific web service and propagating its interface for invoking method calls. WSDL descriptions of web services are normally generated automatically by the development environment and then distributed to all interested parties, e.g. by making them accessible via the World Wide Web.

- Universal Description, Discovery and Integration
  This technique, which is explained in more detail in [Oas02], is used for retrieving web services proving a specific needed functionality. For example, UDDI repositories can be established, in which web services can register themselves, including their WSDL description and their purpose, exposed functionality etc. (see [Bei+02, p. 278]).

The general concept and relationships of these protocols can be seen in figure 1.

Fig 1: Web Service protocols and their relationships [IBM03]

WHAT ARE P2P NETWORKS?

A Peer-to-Peer-Network (p2p-network), as illustrated in figure 2, is a set of equally treated nodes (peers) in a network, which are capable to offer resources to each other without requiring central coordination. (see [ScF03, p. 313]). Milojcic et. al. are using a similar definition: “The term peer-to-peer (P2P) refers to a class of systems and applications that employ distributed resources to perform a critical function in a decentralized manner.” [Mil+02, p. 1].

Therefore, the three most important properties of p2p-networks are (see [ScF03, S. 313ff.] and [Mil+02, S. 12ff.]):

- Resource-sharing: Each peer can act as a server as well as a client within the p2p-network. It therefore can access and offer resources from/to other peers.
- Decentralized network: The resource and network management requires no central instance. Thus no central control can be imposed on the network. Consequently, the peers are communicating directly with each other.
- Autonomy: Peers are able to impose their own security policies and to choose when they offer what resources to whom.

Further properties of p2p-networks can be:

- Ad-Hoc-Connections: P2P-networks are a constantly changing environment. Peers can and will join and leave the network any time they choose, to get connected. Instant messaging networks like ICQ and AOL instant messenger are good examples for this: As users will dial into the Internet, they will connect to the network. When they disconnect they will leave the p2p-network as well.
- Anonymity: P2P-networks can guarantee anonymity for their users if they are designed accordingly. Normally messages are then relayed by many peers so that the destination peer does not know who initially send the message.
- Distributed Resources: The peers can be utilized more efficiently because unused resources like CPU cycles can be offered in the network. This can be used to improve the Return-On-Investment.
- Failover-Support: The network on the whole is not affected, if one peer fails. Equivalent resources can be offered by other peers, resulting in high-availability of resources.

P2P-Networks can be split up into four categories:

- Distributed Computing: CPU intensive tasks are distributed to all nodes in the network which are solving the problem and sending their results back to the initiator. These networks are a good way for using otherwise unused CPU cycles on modern over-sized office computers. An example for this kind of network is seti@home where radio frequency data is analyzed in the hope of finding a proof for extraterrestrial life.
- File-sharing: File-sharing p2p-networks are used for distributing data between the peers resulting in a large data storage including
the storage space of each peer. Those networks make it possible
to aggregate very large amounts of data easily exceeding the
TB-byte barrier by using standard computers. Popular examples for
this kind of networks are Napster and Gnutella.
• **Collaborative Systems:** P2P-networks can be used for communi-
cating with others, like messaging services. Popular examples for
these are AOL Instant Messenger or ICQ.
• **Platforms:** Platforms are providing a complex system for various
kinds of services which can be developed upon them. Examples
for this kind of networks are Sun’s JXTA and Microsoft’s .NET
My Services. (see [Mil+02, p. 7f]).

We are planning to develop a system which fits into the first and
third category: Workflow, in terms of processing data, can be distributed
to other peers, like doing optimizations. On the other hand, digital
contracts and data queries can be performed which is collaboration
between corporations.

**COMBINING WEB SERVICES AND PEER-TO-PEER-
NETWORKS**

Normally, each p2p-network implementation has each own, unique
and special protocol, which is used only within this network. These
protocols may be optimised and specifically tuned for the given
application. However, designing good protocols is a complicated task.
Furthermore, software which should work with the new system has to
implement its special protocol. Because we want to uniformly access
data and functionality and want the communication between the systems
as easy and standards-compliant as possible, we will not invent a "yet-
another"-protocol. Instead, we will use web services based on SOAP to
provide resources and normal web service clients to use them. No extra
protocols are needed and implementations can utilize existing class-
libraries and development-environments.

The web service offering the needed functionality then can be
retrieved by querying an UDDI directory. This way any company or any
other person can extend our system by a new feature simply by providing
a new web service implementing the desired functionality. Thus new
business components can easily be added to the system. Because a
specific functionality can be provided by many service providers, each
user of our system is not reliant on one vendor. In case of technical
failure or dissatisfaction with the provider, a user may easily switch to
another one. This way, a new marketplace for services is created, where
services can be bought and offered. But another, more traditional
marketplace will be transferred into the p2p-network as well: All suppli-
der production companies, sellers and resellers can participate and
compete in this network by exposing their catalogues and prices,
availability dates etc. to their potential costumers.

**BUILDING A DISTRIBUTED ERP-SYSTEM
Requirements**

The ERP system should be capable of supporting all resource-
specific planning for SMEs, which are trying to survive in the market
by highly customized and high-quality products. For this reason, the ERP
system needs to be easily adaptable. Furthermore, it needs to be highly
reliable, because very important business processes are dependant on an
ERP system. Because of the intended users, the system needs to be very
cost-effective and not dependant on an own staff of technically skilled
people for administering and supporting the system.

**Participants**

The participants in this system are companies offering goods
(seller), reselling goods (reseller) and the ones who are offering specific
services for this system, like special business components (service
providers). Everyone can participate in any role, as long as he installs
a server within his network which is able to access the required resources
for the task. The server will act as a peer in the p2p-network and needs
to expose any functions as SOAP-web services and register them
accordingly in an UDDI-directory.

![Fig 3: Structure of the proposed system](image-url)
ERP systems to stay competitive but cannot afford the classical all-in-one solutions. However, these can install and use our ERP-system without spending much money and without the need to build up expensive technical staff for support.

They simply have to buy one server system which will act as a gateway to the p2p-network and will query for the data and will use further web services to extend its functionality if available.

Availability is assured by the spread of the web services through the p2p-network, thus if one web service fails, another may take over.

The web services can access and use their own database, so the average database size is decreased. Backup and restore operations are therefore faster and easier to perform.

Another side-effect is the blurring between inter- and intranational integration: If two enterprises want to cooperate by using the web service-based system, it is technically not more difficult than joining two business units this way.

Our system is very flexible, because it mainly consists of loosely coupled web services which are providing the desired functionality. Therefore, the system can be easily extended and customized, simply by adding and/or using web services. The complex customization process now becomes an easy subscription to web services.

CONCLUSIONS AND OUTLOOK

The new system and its design are a promising for small- to medium-sized enterprises. We are right now involved in implementing the system and will roll it out at selected sites.

We hope that this way, a new generation of ERP-systems is born which is easier to install and maintain than traditional monolithic systems often requiring the restructuring of a company to work efficiently.

Because the whole system is based on published and free standards, it is easy for everyone to utilize the system: Agents may roam the p2p-network for specific information and even adapters to other, proprietary systems are possible.

REFERENCES


[Oas02] Organization for the Advancement of Structured Information Standards (OASIS) (Hrsg.):


[W3C03a] World Wide Web Consortium (W3C) (Hrsg.): Web Services Description Language (WSDL) 1.1, August 2003

A Disaster Management Specific Mobility Model for Flying Ad-hoc Network
[www.irma-international.org/article/a-disaster-management-specific-mobility-model-for-flying-ad-hoc-network/156480/](www.irma-international.org/article/a-disaster-management-specific-mobility-model-for-flying-ad-hoc-network/156480/)

Next-Generation Optical Access Networks
[www.irma-international.org/chapter/next-generation-optical-access-networks/113055/](www.irma-international.org/chapter/next-generation-optical-access-networks/113055/)

Hypervisor-Based Server Virtualization
[www.irma-international.org/chapter/hypervisor-based-server-virtualization/112514/](www.irma-international.org/chapter/hypervisor-based-server-virtualization/112514/)

Fuzzy Decision Support System for Coronary Artery Disease Diagnosis Based on Rough Set Theory

Two Rough Set-based Software Tools for Analyzing Non-Deterministic Data