Challenges in Moving from Documents to Information Web for Services

Rakesh Mohan, Biplav Srivastava*, Pietro Mazzoleni, Richard Goodwin

IBM Research
Hawthorne, NY 10532

Abstract

The service industry has traditionally relied on physical documents to communicate and manage their project and operational activities. As they adopt the object-centric view of Web2.0 technologies in their productivity tools, knowledge-based workers now have to work with inter-connected object-centric view of information where they earlier had to deal with only documents created from word-processing tools. The paper discusses the new technical and business challenges that must be addressed for making object-centric Web2.0 based tools successful for services.

Introduction

There is a silent revolution underway in how information is getting captured and exchanged today. More and more information that was earlier captured in static documents is now captured in richly integrated, dynamic webs of information (Berners-Lee 2001). This advance brings unprecedented advantages to users in terms of how easy it is to update, share, collaborate, keep current and make decisions with information. However, it also raises issues in the way users employ the information in their work, as work processes are still very document centric. Nowhere is the phenomenon more evident than in services. The research directions in relevant areas like semantic web focus on building the web but do not currently recognize the consumption aspect of the integrated information [McCool 2005, McCool 2006, Lu et al 2002].

Information in Services

Most services activities follow a method that consists of roles, processes and work products. For example, in a doctor's office, there are patients, receptionists, nurses and doctors. A common process is that a patient first makes an appointment with the receptionist. At the doctor's office, the nurse first inspects the patient to see if the right records are there and takes some simple tests like temperature and blood pressure. The doctor then consults with the patient and takes notes and fills out forms and writes prescriptions. The appointment record, nurse's test results, doctor's notes and the prescription are examples of work products. The information about the patient's problem, and what the doctor did about it, is captured in the work products. This method can range from implicit and flexible, such as in a small doctors' practice to explicit and regimented, for example in an insurance company. In the process, as work gets handed off from one role to another, the information is handed over (called hand-off) in the form of the work products, for example the prescription from the doctor to the pharmacist.

With the advent of electronic patient record, the information about the patient may evolve to a rich web of information, linking the results of doctor's visits, test results, to potentially information about the doctors, insurance and members of the patient's family. In this scenario, one has to decide both what subset of the information web of the patient record is handed off between roles, and where approval is required, such as by an insurance company, one also needs to decide what temporal snapshot of this information web is passed.

Figure 1 illustrates the inter-play between how information is captured and how people collaborate with it.

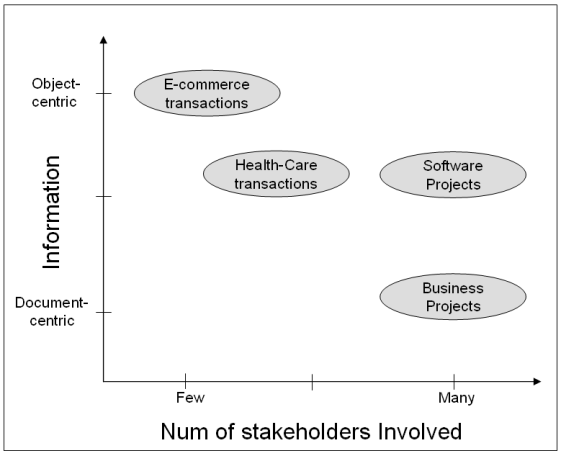


Figure 1: Information and Collaboration

In more complex scenarios, the issues of handoff, sharing of information and reaching agreements become much more severe. Consider projects like designing an airplane or implementing an enterprise scale software system. These projects involve hundreds of people working for years. There are many different roles, and organizations. The activities evolve over time and the richness of the information web and hand offs' between people and organization is extremely complex. These projects highlight the issues that appear when document centric work methodologies move to the information web.

*Corresponding author, can be reached at sbiplav@in.ibm.com

Let's consider the implementation of an Enterprise Resource Planning (ERP) system in more detail. An ERP system is a packaged application that provides common business functions and processes. The value proposition for adopting an ERP system is twofold: the packaged application embodies industry best practice processes, allowing an enterprise to leverage the experience of others; it is also less expensive and risky to license a packaged application rather than develop and maintain a custom application. According to AMR Research [AMR 2008], the total market size for ERP software is currently $34.4B. SAP leads with 42 %, followed by Oracle (23%), The Sage Group (7%), Microsoft Dynamics (4%), and others. While substantial, this market is dwarfed by the market for related services. AMR Research estimates that spending on services including consulting, integration and support for Oracle, SAP, and other business application vendors, called packaged enterprise application services, was $103B for 2007, and expected to reach $174B by 2012. We will use an SAP engagement as our example for the rest of the paper, since such engagements constitute over 50% of this market.

SAP provides a comprehensive set of pre-defined business processes, reports and integration connectors [Matthes and Ziemer 1998]. In addition, SAP systems are highly configurable and allow the clients to run custom application code to meet requirements not addressed by the pre-defined business processes. Hence, SAP services engagements are intrinsically complex due to the sheer size of the system and can go for years involving hundreds of people. Nowadays, there is a push to deliver some parts of the service engagement using global delivery resources which adds another dimension to challenges in coordinating activities and information.

There are methodologies, e.g. SAP ASAP, to accelerate and promote best practices for services projects. Such methodologies are built around a collection of workproducts that make up a complete implementation method. A medium-size SAP project is composed of hundreds of workproducts today collected mainly using office productivity tools like MS Office and Open Office.

Tools for Capturing Information

On an SAP project, let us consider the pieces of information that need to be captured during the design phase for a business process. The information includes inputs and outputs, process variants, roles, steps to be followed; metrics to measure performance, policies to follow, transaction details, etc. Figure 2 illustrates this as a web of information objects, with their relationships.

It is common practice in the field to use office productivity tools to capture this information in a set of defined work products that each captures overlapping parts of the information in the model. We call this approach the document-centric approach. Figure 3 shows an example of how process information is captured in a typical SAP projects. The document consists of various sections, each specifying some aspect of the business process. Different role-players would create and exchange such documents, as a mechanism for coordinating their activities, and reaching and documenting agreement.

The benefits of document-centric approach are:

- Work Products can be created independent of any governance restrictions, as needed.
- Documents can be physically stored and use to document agreements.
- Documents can be passed-around to anyone who has access to the software compatible with document format
- There is no need for special tools to collect and distribute information.

With the increasing number of work products generated using those tools, it has been recognized that users need better ways to search, organize and reuse content. The introduction of text indexing and search addressed certain challenges in finding relevant content. However, it is still very time consuming and labor intensive to crawl through the search results in order to identify reusable content document by document.

The drawbacks of document-centric approach are:

- Information available to an individual can get stale and out-of-synch with the latest information available to the project.
- Documents cannot be consumed easily. For example, formatting can be changed, ad-hoc objects inserted
- Licenses for commercial word processors and format inter-operability are lingering issues, and the chosen technology varies with client preferences
- Intrinsic relations among information in the documents need to be independently (and manually) maintained by the users.

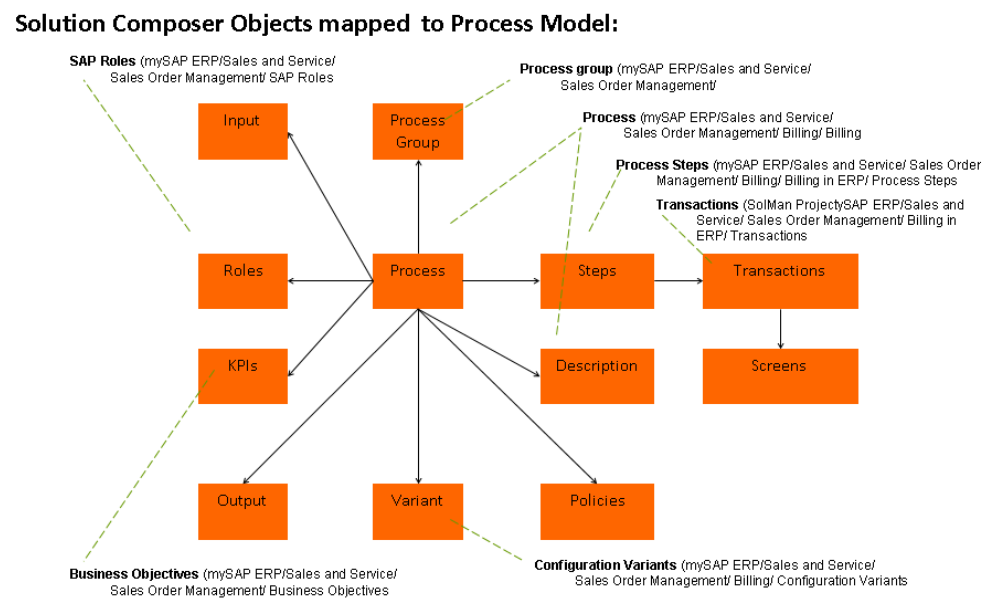
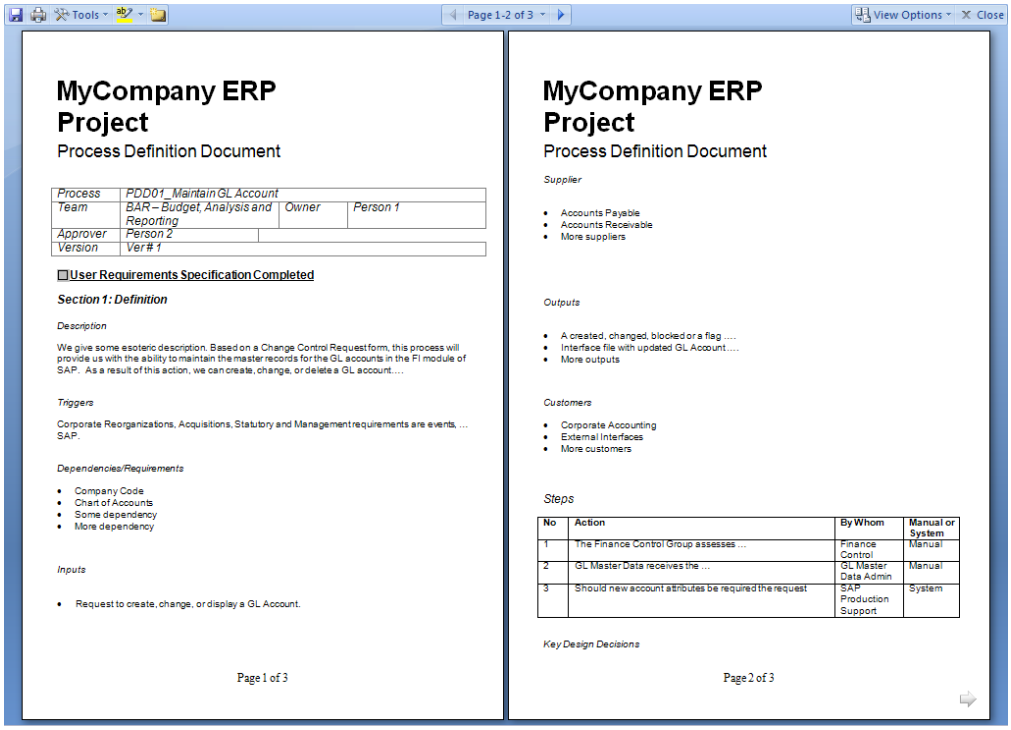


Figure 2: Object-Centric Representation of Process Information for SAP Implementation Illustrated with Content from SAP's Solution Composer and Solution Manager Tools



MyCompany ERP Project

Process Definition Document

Process	*PDD01_Maintain GL Account*		
Team	*BAR – Budget, Analysis and Reporting*	*Owner*	*Person 1*
Approver	*Person 2*		
Version	*Ver # 1*		

☐ **User Requirements Specification Completed**

Section 1: Definition

Description

We give some esoteric description. Based on a Change Control Request form, this process will provide us with the ability to maintain the master records for the GL accounts in the FI module of SAP. As a result of this action, we can create, change, or delete a GL account….

Triggers

Corporate Reorganizations, Acquisitions, Statutory and Management requirements are events, … SAP.

Dependencies/Requirements

- Company Code
- Chart of Accounts
- Some dependency
- More dependency

Inputs

- Request to create, change, or display a GL Account.

Page 1 of 3

MyCompany ERP Project

Process Definition Document

Supplier

- Accounts Payable
- Accounts Receivable
- More suppliers

Outputs

- A created, changed, blocked or a flag ….
- Interface file with updated GL Account ….
- More outputs

Customers

- Corporate Accounting
- External Interfaces
- More customers

Steps

No	Action	By Whom	Manual or System
1	The Finance Control Group assesses …	Finance Control	Manual
2	GL Master Data receives the …	GL Master Data Admin	Manual
3	Should new account attributes be required the request	SAP Production Support	System

Key Design Decisions

Page 2 of 3

Figure 3: Document-Centric Representation of Process Description for SAP Implementation

The new wave of tools in services is adopting object-centric paradigm of the internet to manage information. Here, objects are inter-connected and represent the domain of discourse. The links between objects specify the relationships and constraints that are maintained. Service participants can see and update the latest information based on their privileges. Figure 2 had shown an example for SAP process information.

The benefits of an object-centric approach are:

- Up-to-date information available to everybody ("single view of truth")
- Document can be viewed as snapshot of a view on a sub-graph of the information web, at a given time. It can be created on-demand in any format.
- Instead of physical documents, links to objects can be shared. Intrinsic relations among information in the documents can be maintained in common repositories shared by all users.

The drawbacks of object-centric approach are:

- When documents are published for objects, they cannot be easily updated and then reflected back on the objects. This is because the objects may have been updated to a different state by the time changes from the document are completed
- Objects cannot be altered freely, by design. The inter-relationships and constraints among the objects need to be always consistent and they restrict the type of changes that a user can do.
- New tools, not familiar to users, are needed to collect and distribute information.

Position

The key problem with moving from document-centric approach to object-centric approach in services is streamlining the hand-offs between the role-players. In order to do that, the activities of the role-players have to be structured; the commitments/agreements during the hand-off have to be persisted over time and verifiable; and the system has to evolve as changes happen over time. The most relevant research areas like semantic web do not currently recognize the consumption aspect of the integrated information [Lu et al 2002]. In this paper, we describe the technical and business challenges that must be addressed to make object-centric approach successful for services. The issues are:

- Building a model for the object-centric system covering informational needs. It is important for this model to cover the crucial pieces of information along with their relationships. Changing the model becomes harder, if not impossible, after the system is built and data has been created.
- Building new tools which can support users in editing information as defined in the model as well as promotes best-practices for complex projects.
- Publishing object-centric information, along with attachments, into open formats so that they can meet the expectations of the document-centric external world.
- Supporting online as well as offline updates to information
- Harvesting content from legacy document-centric artifacts and using it to populate the new object-centric system
- Integrating information from different sources and tools

Building An Object-Centric Model

Model Driven Architecture [MDA 1996] is an established way of build large software systems, and a well defined model is key to this approach. However, a model for software systems that captures the complexity of large services projects (e.g., multi-year SAP customization projects) can quickly become overwhelming. Issues arise when changing one aspect of the model impacts other aspects. Inconsistent models lead to confusion and inconsistencies in the information captured. Limiting the explosiveness of the modeling language may reduce complexity and inconsistency, but at the cost of being able to express required relationships. It is an open question as to how to balance a more expressive modeling language with overly complex models.

In the end, the model represents the underlying data organization that is implicitly contained in existing work products and methods. Can we automatically extract a model from a business methodology? Inferring a model is complicated by the fact that consultants adapt the implicit model for each engagement.

Once system is established with a particular model, the users can start entering information associated with different objects. If they want to add new objects or relationships into the model, the relationships and constraints among the data that was already entered may get violated[1]. As a result, most approaches do not allow the model to be changed once the system is built. The downside to this approach is inflexible systems that cannot adapt to the needs of the situation. How to support required flexibility, while not invalidating previously collected information, is a challenge.

Tools for Creating Model-Guided Content

Once we have a model of the required information, it can be used to structure the collection of information to meet the needs of the project. For example, if requirements object is suppose to have a source and a business objective, the organization becomes clear to the consultants since

[1] The situation is not much different from databases where a table's schema cannot be changed once data is entered into it. To change, a new table is created with the new schema, the data transferred to the new table, and the old table is deleted.

each requirement object will have fields for each of these. Providing a structured way to collect information improves the quality of data that is entered in the system while also improving user's productivity. We see this as a major advantage of a model centric approach to consulting.

Publishing Object-Centric Information

A document from document-centric tools is a container for different types of information, e.g., textual content, figures, multi-media objects. They may be zip files (like Lotus Symphony or Open Office) or binary/ XML formats (like Microsoft Word). The document paradigm allows the information from related objects to be seamlessly transferred with the document.

In the object-centric view, individual objects can be published as documents and collections of related objects can be published in documents, much the same way as a report generator provides a view on some information in a database. However, a report is not a substitute for a copy of the database. How will we support the interchange of information with rich underlying models? One solution is to create a representation of the objects' inter-relationships and store it along with the published documents.

A document represents a snapshot of the information at a point in time. As projects progress, revised versions of documents can be easily generated from the updated content. Like software releases, document releases will need to be tracked and verified before release to the intended audience (usually the client). It would seem natural to use standard version control techniques to track and control the release of documents [CVS 1998].

Supporting Online and Offline Updates

Objects can be easily updated when the user is online. However, users will not always be online. Moreover, snapshots of the objects' states, i.e., documents, will still be needed to exchange with other role players who are not yet in the system.

When documents are published for objects, they cannot be easily updated and then reflected back to the objects. Recall that a document is a snapshot the information (materialized view in database terminology). Once created, it is disconnected from the updates to the corresponding objects. If we want to or need to allow modifications to the information in a document or modification to model elements passed to another system, how do we coordinate modification? Do we allow concurrent modification with an optimistic locking policy? Do we use a strict locking policy?

If we disallow disconnected modification of project information, we introduce a complete dependency on network availability. Any network disruption would nearly halt work on the project, and adding another risk to project completion. If such a situation is unacceptable, it is better to plan for disconnected operation before it happens.

Harvesting Information

There exist large collections of documents whose content would be useful on future engagements. To make use of this information, it needs to be brought into imported into the model centric systems. Documents consist of headings, paragraphs, lists, tables, images, embedded objects, and arbitrary combinations of these elements (e.g., lists inside tables). A document's content is annotated with formatting/visual cues. A person can usually identify content of interest by reading the text and using the visual queues that distinguish particular kinds of content. However, two sections of content that look identical can have very different binary representations in a document. We are working on harvesting technologies to extract useful information from legacy documents and transform to a model centric representation.

In consulting methods, templates are often used to create work products. When work products have been created using templates, the extraction process can become more tractable. However, most projects modify standard templates, so a cross project template can not be relied on. When a collection of documents share a template, even if the template is unknown or none standard, we can still make use of common landmarks in the documents to help extract semantically meaningful content. IBM Content Harvester [Srivastava 2009] is an example of a tool designed to discover the template(s) from many instance documents. It can then extract, cleanse, tag and store template-directed content. The resulting structured information can be more easily imported into the web of information.

Integrating Information

The need to integrate information is omnipresent. In document-centric methods, projects integrate information by having a dedicated team for information integration. That team works with the business process teams to coordinate activities and force consistency across the document based workproducts. In an object-centric method, relationships between objects are already represented by the web of information. The user gets a current, integrated view of the information at all times and the need for manual coordination is reduced.

Beyond the bounds of a project, there are useful sources of information that should be made available to the team members. There is a need to exploit available taxonomies, ontologies and information repositories.

Conclusion

The service industry has traditionally relied on physical documents to communicate and manage projects and

operational activities. As the industry adopts object-centric Web 2.0 technologies in their productivity tools, consultants should become more productive and produce higher quality work. To achieve this result, consultants will have to adapt and will face new challenges. Software tools and methods will also need to adapt. To date, research in the semantic web and information retrieval has focused on building the web of information from loosely coordinated contributors or on analyzing existing webs of information. There is little work on coordinating the work of a large group of people to build an information web to satisfy the requirements of a services project. In this paper we have identified hand-offs as a major source of problems and discussed new technical challenges that must be addressed for making object-centric Web2.0 based tools successful for services.

Sample References

AMR 2008. AMR Research. "World-wide ERP market". In *AMR Report*, 2008.

Berners-Lee 2001. Berners-Lee, Tim; James Hendler and Ora Lassila (May 17, 2001). "The Semantic Web". Scientific American Magazine.

CVS 1998. Concurrent Version Control, http://www.nongnu.org/cvs/

IBM. 2008. Jazz.net.

Lu et al 2002. Lu, Shiyong, Dong, Ming and Fotouhi, Farshad (2002) "The Semantic Web: opportunities and challenges for next-generation Web applications." *Information Research* **7**(4), Available at: http://InformationR.net/ir/7-4/paper134..html

Matthes and Ziemer 1998. F.Matthes and S. Ziemer. "Understanding sap R/3: A tutorial for computer scientists." In *EDBT. Slides at http://www.sts.tuharburg. de/slides/1998/03-98-MaZi-SAP-EDBT98.pdf*, 1998.

McCool 2005. McCool, R. "Rethinking the semantic Web. Part I". Internet Computing, IEEE Volume 9, Issue 6, Nov.-Dec. 2005 Page(s): 88, 86 – 87

McCool 2006. "Rethinking the semantic Web. Part 2". Internet Computing, IEEE Volume 10, Issue 1, Jan.-Feb. 2006 Page(s): 93 - 96

OMG 1996. Model Driven Architecture, http://www.omg.org/mda/

Srivastava 2009. Srivastava, B. et al. IBM Content Harvester. At http://www.alphaworks.ibm.com/tech/contentharvester