KNOWLEDGE MANAGEMENT IN INFORMATION SYSTEM DESIGN AND DELIVERY PROCESS

An Application to the Design of a Legal Information System

Ovidiu Vasutiu[a,b], Youssef Amghar[a], David Jouve[b] and Jean-Marie Pinon[a]

[a]*Laboratoire d'InfoRmatique en Images et Systèmes d'information – INSA de Lyon, France*

[b]*Caisse Nationale des Allocations Familiales – Cnedi Rhône-Alpes, France*

Keywords: System design process, legal document, document retrieval, document consistency, UML, XML.

Abstract: Nowadays information systems are more and more important for all types of organizations. To deal with the complex technologies available, IT specialists use proven best practices inspired from more comprehensive process frameworks for software and systems delivery or implementation and for effective project management. Methods developed to support theses processes produce many different heterogeneous resources (design documents and models, planning, project prototypes, etc...). Due to the continuously changing reality, designers will always have to consider new user and stakeholders requirements and go back to the starting design case for an update. In this paper we present an organizational and technical infrastructure for a collaborative design process management system which automates mechanisms to assure the coherence and consistency of these heterogeneous and continuously updated resources..

1 INTRODUCTION

Organizations are turning out to be more and more dependent on Information Systems. At the moment where IT technologies are becoming more complex, where many frameworks and architectures models are available, the need of tools to support the Information System design, delivery and implementation process is very important. This design process take in consideration large teams and deal with many different skills and profiles: users and stakeholders define the functional requirements, software and technical architects elaborate the system architecture, developers build it, designer package it, etc… Each one of these actors provides a different view and knowledge on the system. Their functional, technical and organisational knowledge, while representing different aspects of the same system, will be expressed by heterogeneous resources: documents and models in different formats, and, finally, by applications' source code.

Methods such as RUP (IBM, 2007), MSF (Microsoft, 2007) or Extreme Programming (Larman, 2003), have been developed to support this process using different phases (i.e. inception, elaboration, construction, transition) each one producing different deliverables: documents, models. Due to the complexity of the systems, incremental and iterative approaches are used. The system is then delivered into sets of iterations producing or updating deliverables (design resources, documents, models). Information Systems need to follow, to adapt to a continuously changing reality. Therefore the system's knowledge is building all the way during the design and implementation phases making it very important to assure the coherence of the design resources. Furthermore, in many domains (legal organisations, banking, e-government, etc.) it is also very important to guarantee the traceability all the way through the process; each component, each update has to be related to a user or functional requirement.

In this paper we propose a system based on knowledge management to control and to guaranty the coherence and the consistency of theses heterogeneous resources (documents and models) and the information system.

The work reported in this paper has been carried out in the context of the Caisse Nationale des Allocations Familiales (*Cnaf*), the French Family Benefits Office, whose main mission is applying the law in the process of evaluating each family's allowances. Therefore, our proposition was implemented in this particular domain and organization. However, it is not restricted to the legal domain. It contains generic elements that can be either directly reused or extended so as to represent specifics of various domains of interest.

2 MOTIVATIONS

The legislation of our organization's domain is particularly complex. The organization has to deal with a massive amount of legal documents (more than 100,000 references) and numerous legal updates (over two legal updates every week). Considering the important number of beneficiaries and the large amount of information to be processed, the workload of this organization overwhelms human capacities forcing it to use the computer's assistance, a Legal Information System.

For any legal sub-domain, the set of relevant legal documents is partially structured (Jouve *et al*, 2001), documents can be laid out in a pyramidal structure according to their legal importance, *Figure 1*. The highly operational degree documents are at the lowest level of the pyramid. They are used for the Legal Information System design.

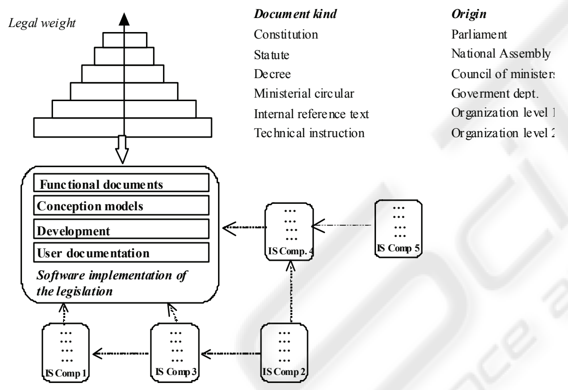

Figure 1: Legal and design resources hierarchy.

Due to the legal context of each country, legislation is constantly added, deleted or modified, like an informational flow in perpetual evolution. One change at a given level of the legal pyramid will have repercussions to all beneath levels until the lowest level is reached, affecting the information systems components that are directly and strongly connected to the legal domain.

Information Technology specialists use comprehensive process frameworks, providing then with proven best practices for software and systems delivery and effective project management. For example the RUP - Rational Unified Process proposes a project design life cycle in 4 phases: inception, elaboration, construction, transition. In each phase different actors (users, designer, architects and trainers) take part producing heterogeneous resources (analysis documents, UML (OMG, 2006) models). This resources are interconnected and they express the same knowledge but on different points of view.

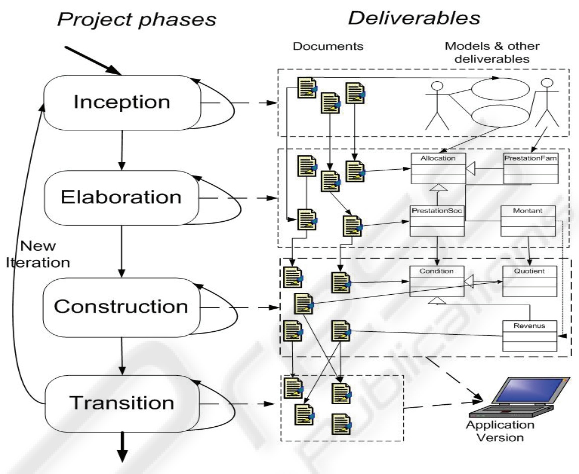

Figure 2: Phases and heterogeneous deliverables.

When user requirements change, the update has repercussions to all others design documents. There are solutions for document management systems and collaborative work; others solutions for UML modelling; and even for requirements management, but not many solutions control the coherence and consistency of both kinds of resources. In this paper, we propose a logical and technical architecture for the design resources management and production environment of our organization.

3 STATE OF THE ART

3.1 Structuring Documents

XML(W3C, 2006) describes the logical structure of the documents using mark-up, enabling both an explicit representation of the given information and a description of relations between modelled information and documents sub-structures. Furthermore the benefit of mark-up languages is that they allow a clear separation between data and its representation.

XML documents are stored in a native XML databases which uses the XML document as the fundamental unit of storage having better performance on most of the rich content document queries (Runapongsa *et al.*, 2006).

3.2 Document Integrity

Research projects in literature address document integrity in a document repository based on link integrity check focusing their work on the intranet environment (Amghar & Chbeir, 2003), others (Ashman, 2000) summed up all measures in two categories. (1) *preventive solutions* : forbidding changes, versioning, embedded links, external links, link attributes and (2) *correctives solutions* : forward mechanism, Universal resource names, relative references, utilization of the paradigm of agents.

3.3 Knowledge Representation

Many scientific works managed to accomplish models for the representation and manipulation of legal documents (Wright, 1963). David Jouve (Jouve *et al.*, 2003) addressed the issues caused by the constantly changing legislation proposing a solution to the need to efficiently control a legal domain: impact studies, incoherence and conflicts detection.

Others related works interested on the management of human knowledge within the design process, the collaborative work and knowledge sharing (Rodríguez *et al.,* 2004) and the organizational memory (Rus and Lindvall, 2002). We propose to use knowledge management for consistency and coherence control mechanisms within heterogeneous resources.

4 DESIGN PROCESS MANAGEMENT ENVIRONMENT

Our answer to the issues presented here is an XML based solution for a production and management environment for information system design resources. The environment developed at the *Cnaf* has an N-layer type of architecture.

4.1 Meta-Model of the DMIS

The design document referential embeds a set of *Document collections*. Each collection is provided with a set of descriptors defining all the types of contained resource: document types (i.e. legal document, analysis, design, etc.), asset types, reference types… The collection meta-description contains *taxonomies, docflow definitions* and *references types.*

4.2 Basic Features

a) Document format and structure

To avoid redundancy, two structures are used: *the document persistence structure* - content stored at only one place, enabling document archiving, indexing, and retrieval -, and the *document presentation structure* (user ended documents containing richer and more detailed information, an aggregation from different persistent documents). All documents respect a generic structure having two top elements: the *meta* element (all document meta-data: document type, ID, version, revision and edition identifiers, origin, state and docflow history) and the *content* element (contains the information). The system uses a native XML database to store persistence structures. For backward or external interoperability raisons it can also wrap in an XML structure any other content.

b) Version and revision management

The stable reference documentation is stored with the "reference" version in the reference space. When a new user requirement occurs a new editing workspace is created containing documents to be added, modified or deleted (*Figure 3*) in order to respond to that requirement. The new workspace is created starting from the reference space therefore in each workspace we find new versions of the reference documents. This new versions follow theirs document editing life-cycle leading to different revisions. At any time, editors can synchronize, using a merge tool, their working version of the document with the reference version of the document (the public version of the document).

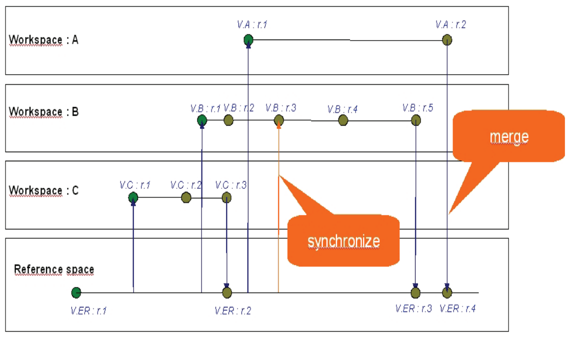

Figure 3: Multiple parallel document editing workspaces.

Once a working version of the application is validated all the modifications will be integrated in the reference space, updating it. This situation sometimes leads to multiple parallel versions of the application having parallel versions of the application's documentation.

c) Resource identification

Knowledge is scattered all over the document corpus thus preserving link integrity of the document repository is a crucial problem. A solution to precisely addressing a resource is the use of Universal Resource Names or URNs. A revision of a document is precisely identified by the unique document identifier, its version and revision number. It is referenced by a storage independent URN formed as follows:

```
   collection-name : resource-kind :
resource-type : resource-id
[: version : revision]
```

Where the *resource-kind* describe the kind (document, image, an external resource), *resource-type* (data description document, class description documents, legal document, etc…); *resource-id*: is the unique identifier given by the database.

This identification allows us to have a unique reference model where a resource can be moved without affecting the references pointing it. URN resolution rules will identify the physical document, taking in consideration if the version and/or revision are specified or not. Documents with the current or reference version and those in the last revision are privileged. The advantages of this link resolution algorithm are that a link can be defined to always point out to the last revision of a document even if the target document evolves, or it can be strictly forced to point out to a specific revision number, even if obsolete.

d) Document Addressing Model

A reference is encoded as an XML element with a set of attributes that is mapped to the URN of the target:

```
   <ref col="sair-daf" kind="docobject"
type="tache" res="00000003"
[ver="REF" rev="0001" nodes="id1 id2"
ref-type="originContent"]/>
```

The *col*, *kind*, *type*, *res* define the URN of the referenced resource. The optional nodes attribute can address a specific sub-fragment of the document. Referenced document's label, description, type are retrieved. Sub-fragments identified by the *nodes* attribute, content defined by the *ref-type* are included in the presentation of the referencing document. All the information concerning the target document is stored within so that any update is automatically available. We can identify two categories of reference types: (1) *simple reference*: a document points out to another document implying a simple link, (2) *specialized reference*: attribute *ref-type* is present defining the semantic of the link. This semantic defines the transformations to be made during loading to build the presentation document.

e) Docflow

Each document at a given moment is associated to a state in its lifecycle (specific to each document type). User contextual menu functions are defined according to user's authorizations and document state. An internal docflow engine listens to external (function activation, service call, etc...) or internal (document transitions) events that might (if all conditions are validated) launch a transition to another state. Actions can be executed on in/out transitions.

f) Information retrieval

The retrieval mechanism is very flexible allowing users to specify a search expression (containing multiple keywords or phrases) and/or document expected metadata (identifier, document type, current state, date).

4.3 Design Document Drafting Functionalities

a) Unique referential

The resource addressing feature and link resolution algorithm allow us to reduce redundancy without losing any valuable performance. A link can either point to the last revision of a document, or it can be strictly forced to point out to a specific revision number, even if obsolete. So documents in parallel editions workspaces (different versions) can be referenced without any interference. Using the "ref-type" attribute, we can define meaningful links between two documents (i.e. origin of the document, knowledge dependency or similarity) or how content from a document is included in the presentation structure of another document.

b) Drafting process management

Responsibility in the design process is often shared between many actors. Docflow mechanisms allow us to define the automation of the drafting process, the roles of each participant, the flow of documents and information, in whole or part, from one decisional instance or state to another for action.

c) Knowledge and information traceability

Meaningful links allow us to trace the usage of the knowledge and its propagation while meta-data allow us to trace evolutions, contributors, contributions and updates of the documents. Software engineers use a specific process to create the application components needed to respond to each user or functional requirement. The history of each document can give us the information needed to retrace back all the interventions on the document to the linked document fragment expressing that requirement. Therefore we can define a new specialization of the reference relation, the "user requirement implementation relation".

d) Dependency and impact analysis

Due to traceability in our system we can identify the documents that are in a way related to a requirement text and that need to be updated with each evolution, giving an estimate of the impact (Vasutiu *et al., 2006)*. In our case as Legal Information System design documents are related to the law, it allows us to link legal requirements back to corresponding design artefacts and code.

5 DISCUSSION AND CONCLUSIONS

The document and model management system presented here and implemented at the *Cnaf,* while adapted to the particularities of the legal domain, responds to the generic demands expressed.

For instance coherence is assured by the reference and traceability mechanisms. References can allow us to avoid redundancy, often present especially in legal documents. Knowledge equivalence reference allow us to identify documents, although different but expressing the same knowledge. The complexity of the knowledge transformation within the design process is managed by the docflow and version/revision control features.

To specifically develop this infrastructure took over one year. The system was tested on a 30,000 design document collection which represents almost 30% of the document collection. All the feed-back received, after a user validation phase, shows that even if the new mechanisms implied more constraints in the work process, they help to reduce the risk of error. Traceability helped users to precisely identify the documents that need to be updated and version/revision control allowed then to constantly be able to control changes and evolutions.

Therefore the next perspective of our research is to estimate the whole area of impact of a user, functional or legal requirement using a dependency and impact analysis on both documentary area and on the information system.

REFERENCES

Amghar Y., Chbeir R., (2003). Advanced Management of Documents Integrity in an Intranet Environment, *ACS/IEEE Conference: International Conference on Computer Systems and Applications*, Tunisia, 2003

Ashman, H., (2000). Electronic Document Addressing: Dealing with Change. *ACM Computing Surveys*, Vol. 32, No.3, September 2000

IBM International Business Machines (2007) *Rational Unified Process*, Retrieved September 15, from http://www-306.ibm.com/software/awdtools/rup/

Jouve D., Chabbat B., Amghar Y., Pinon J-M. (2001). L'archivage des documents réglementaires : maîtrise d'une matière première spécifique. *Document Numérique*, Vol. 4, (pp. 315-329).

Jouve D., Amghar Y., Chabbat B., Pinon J-M. (2003). Conceptual Framework for Document Semantic Modelling: an Application to Document and Knowledge Management in the Legal Domain. *Data & Knowledge Engineering*, Vol. 46, No. 3, (pp. 345-375).

Larman, C., V. Basili (2003), *Iterative and Incremental Development: A Brief History*, Computer (IEEE Computer Society) 36 (6) pp. 47-56.

Microsoft (2007) *Microsoft Software Fondation.* Retrieved September 15th from: http://www.microsoft.com /technet/solutionaccelerators/msf/default.mspx

OMG, Object Management Group, *Unified Modeling Language Specification* version 1.5, formal/2003-03-01 Retrieved September 15th from: http://www.omg.org/

O.M. Rodríguez, A.I. Martínez, J. Favela, A. Vizcaíno, and M. Piattini, 2004, Understanding and Supporting Knowledge Flows in a Community of Software Developers, LNCS 3198: 52-66.

Runapongsa K., Pate, J.M., Jagadish, H.V., Chen, Y., Al-Khalifa, S., (2006). The Michigan benchmark: towards XML query performance diagnostics, *Information Systems* No. 31, (pp. 73-97)

Rus, I., Lindvall, M., 2002, Knowledge Management in Software Engineering, IEEE Software, 19(): 26-38.

Vasutiu O., Amghar Y., Jouve D. (2006) Gestion des changements et étude d'impact dans un système d'information réglementaire». *XXIVème Congrès INFormatique des Organisations et Systèmes d'Information et de Décision,* pp. 1007-1022, (2006).

von Wright, G.H. (1963). Norm and Action: A Logical Enquiry, *International Library of Philosophy and Scientific Method*, Routledge and Kegan Paul, London.

W3C World Wide Web Consortium, (2006) *Extensible Markup Language (XML) 1.0 (Fourth Edition) W3C Recommendation* 16 August 2006. Retrieved April 10, 2007, from http://www.w3.org/TR/REC-xml/