Structured and Fragmented Content in Collaborative XML Publishing Chains

Stéphane Crozat
Université de Technologie de Compiègne
Unité Ingénierie des Contenus et Savoirs
60200 Compiègne, France
stephane.crozat@utc.fr

ABSTRACT

In this paper, we present the main results of the C2M project through one of its operational deliverable: the Scenari4 collaborative editing and publishing system for XML content. The purpose of the C2M project was to design a system able to manage structured and fragmented contents - as XML *publishing chains* do - while providing collaborative possibilities - as Enterprise Content Management systems (*ECM*) do. The main issue is related to transclusion relationships which are massively used in XML publishing chains, in order to support repurposing without copying. This approach is not compatible with the classical way ECMs manage content, especially in terms of propagation of modifications, rights or transactions management. We propose two complementary solutions to manage two different levels of collaboration. The *workspace* is designed as a highly dynamic place able to deal with live fragments, linked together in a network, that can be easily updated at any time by any user. The *library* is a more static and more classical way to manage content, dedicated to *folder-documents*, which are XML frozen versions of sub-networks extracted from workspaces. While workspaces are dedicated to content elaboration and maintenance, libraries are places to store, to read, or to exchange stable documents. Scenari4 is released under FLOSS license and has been being used in several experimental and commercial contexts since the beginning of 2012.

Categories and Subject Descriptors

I.7.1 [**Document and Text Processing**]: Document and Text Editing – *Document Management, Version control*

General Terms

Design, Reliability, Experimentation.

Keywords

XML Publishing Chain, Structured Document, Fragmented Document, Transclusion, Repurposing, ECM.

DocEng'12, September 4–7, 2012, Paris, France.

1. INTRODUCTION

The C2M project (Multimedia Collaborative publishing Chains[1]) is a French research project funded by National Research Agency (ANR). It began in September 2009 and ended in March 2012. It was coordinated by the Université de Technologie de Compiègne (UTC) and gathered the companies Kelis and Amexio, the laboratories UMR-CNRS 7253 Heudiasyc and INRIA Rhône-Alpes, and the French National Audiovisual Institute (Ina). The C2M project addresses two important aspects of present mutations in the documentary field: the XML publishing chains and the Enterprise Content Management (ECM).

A *publishing chain* is a technology oriented toward the creation and publication of structured documents [1], *i.e.* documents described through their logical structure rather than their physical presentation. Early implementations in the 80s with LaTeX and SGML, addressed contexts with huge and strategic documentary issues (aeronautics, scientific publication...). Since 1998, XML and associated software progressively democratized the use of publishing chains in less specific areas. The interest of such an approach is to enhance automatic manipulation of digital document, in order to surpass the classical word processors, with writing control, polymorphic publication, reuse without copy (transclusion) or multimedia integration [3].

An *ECM* is a collaborative system dedicated to document management, born in the 80s as Document Management Systems (DMS), evolving in the 90s as Web CMS, and in the 2000s as ECM in companies, and "Web 2.0" in the mass market. The strength of these tools is to democratize digital content creation and circulation, anyone can now easily write and publish on line.

The aim of the C2M project is to articulate publishing chains and ECM in order to be able to produce highly qualitative documents, as expected by professional contexts (such as technical documentation, training...); along with collaborative practices organized though new cycles of information. The project is scientifically related to research in the field of document engineering, meaning systems designed to optimize technical manipulation and human interpretation of digital documents [2]. The project is based on the system Scenari[2], invented at UTC in 1999 and now edited by Kelis, and the main result of the project is the new version Scenari4, released in 2012.

[1] http://www.utc.fr/ics/c2m

[2] http://scenari-platform.org

2. REPURPOSING

2.1 Repurposing using transclusion

Repurposing is a documentary process consisting in building a new document with archives. Whereas in non digital approaches repurposing is more or less similar to an original creation, computer systems brought the possibility to *clone* a document fragment. Cloning helps in automating repurposing, and has become involved in most of document elaborations. But cloning engenders *redundancy* of information, and redundancy engenders lowering of information quality (by introducing inconsistencies). An alternative to cloning is *transclusion* [6], *i.e.* the multiple referencing of a single instance of a fragment via an address. This concept is poorly mobilized in ordinary writing tools and practices, whereas standards like XLink and Xpointer [9] exist and some technologies like HTML allow it in part (*iframe*). Transclusion is only used in some specific areas, such as technical documentation [4] where maintenance stakes are fundamental, or audiovisual where the cost of copy is high [5].

Transclusion and repurposing are one of the main basis of Scenari publishing chains : any content in the system is natively a network of fragments, some of these fragments being reused in several distinct documents.

2.2 Principles of transclusion

The document d1 contains an information i1 at the address &1, which is willing to be reused in a document d2 (see Figure 1). In the first case (cloning), i1 is copied in document d2 at the address &2, becoming a different instance i1' (even if identical to i1 at initialization, it will freely diverge from it in the future). In the second case (transclusion), d2' stores the address &1 of i1. A dedicated software will be able to resolve this reference, in order to integrate i1 in d2' when needed. So, logically d2 and d2' represent exactly the same content at the initialization. But the way information is stored is different, with deep impact on the *documentary nature* of the content. Cloning drives to two separate instances, which will follow their own separate evolutions: as traditional documents they can be separately transported, updated, destroyed... In the other hand, transclusion drives to a single digital instance acting as a network of dependent fragments. As a typical direct consequence, updating i1 in d1 will also modify d2'.

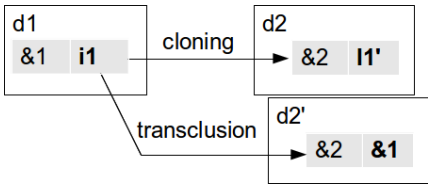

Figure 1. Example of repurposing by cloning or transclusion

2.3 Fragmented content and collaboration

Digitization intrinsically disturbs classical document definitions [7], since the document is a calculated reconstruction of a binary resource: what we read is *not* what is written on the support. Structured document reinforces this disruption by proposing several documentary forms from one single resource. We switch from a 1:1 to a 1:N relationship. Fragmented document reinforces it further more: document is now a reconstruction from a network of several resources, switching to a N:M relationship. The main problem is that the modification of a fragment in the network could be relevant for one document, but not for another sharing the same resource. The system has in charge to help the author in maintaining the consistency of the whole, with reasonable cognitive effort, so that not to handicap his writing process.

In a collaborative environment the management of the network can not be anymore held by one single person, each author is only aware of a part of the dependencies between fragments, it has to be the distributed responsibility of a group. The aim of the system is finally to be able to maintain the coherence of several documents represented by a single network of *live* fragments, in an environment in which several users are working at the same time. The difficulty of such a system is that it cannot be built on the classical solutions provided by ECM systems, since they were designed for non fragmented (and mainly not structured) documents.

3. MAIN ISSUES

3.1 Modification propagation

The first issue is related to the propagation of a modification one user does on one fragment at one moment. In Figure 2, Al changes fragment A in A'. But Bob uses also this fragment in B by transclusion, in a different context. The problem is that Al is not, in the general case, able to decide if A' is still consistent with B, moreover he may not be aware at all of the existence of this use of A by someone else. Only Bob can decide whether to use the live version A'; to remain linked to the dead version A (knowing he will not profit anymore from future updates); to copy A in A" in his context in order to manage it himself in the future; to merge A and A' in A" in order to use a part only of Al's modifications... Such a choice depends on the nature of the modification, a mistake correction is always to be propagated, a technical modification will always be context dependent. For instance if Al manages the technical documentation of a software and Bob the documentation of its installation in a specific company, the modification Al generates when the software evolves in a new version is to be propagated only when this version is installed.

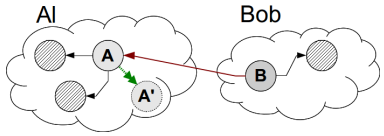

Figure 2. Example of modification propagation issue

3.2 Rights consistency

The second issue is related to the consistency of reading rights. There is no logical difference between a document composed with fragments integrated all together by copy, and fragments linked together by transclusion. In the example 1 of Figure 3, if B1 integrated B2 instead of referencing it, it would represent strictly the same content, the only difference being the future modification management, as seen before (in a totally static system, there would be no difference at all). Consequently it has no sense for Bob to give reading rights on B1 and not on B2. Similarly, in the example 2, if Al gives reading rights to Bob on his fragments, and Bob to Charlie on his owns, then Charlie's fragment C can reference B which can reference A, whereas Charlie has no right to read Al's fragments. These two cases show that classical document management can not directly apply to fragmented content for reading rights (there is no similar problem for writing rights management).

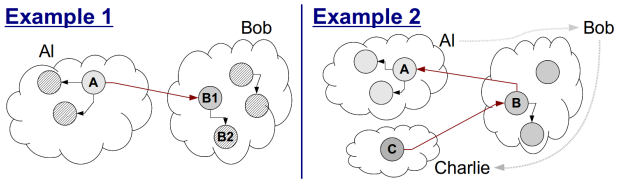

Figure 3. Example of reading rights consistency issues

3.3 Transactions management

A transaction is the way the system manages two (or more) concurrent accesses on one single peace of information. Inherited from databases management, we study here the main strategies exploited in ECM.

The *locking* makes unavailable (for writing and eventually for reading) a piece of information while it is under modification by a user. It can be either automatic or manual. It works well for short transactions on small pieces of information. In a fragmented network, the lock of a fragment should recursively lock every referenced fragment: the problem is that each modification in each fragment would potentially freeze the whole system, by propagation of locks.

A *working copy* is a temporary modifiable copy of a content destined to a single user, in order to isolate it and allow its modification independently of other user actions in the system. Once the modification is done, the working copy is released in the system, most of the time as a new version of the content it was originally copied from. If two users are allowed to ask for a working copy of the same content, a problem will occur when copies return to the system, since they will compete for replacing the original content. Solutions are mainly: *overwriting* of one version on the other, typically the first release is set as version x and the second as version x+1, eclipsing the previous; *merging* (automatic and/or manual) of the two copies in a single one representing changes of both; *forking*, each version is released as a distinct content. In fragmented context, we have to consider the working copies of networks rather than only isolated fragments.

The *check-in/check-out* process is a combination of both. It imposes one user to check-out a working copy, generally locking the original content at the same time, before being able to modify anything, and then check-in it when modifications are done, to make it available and unlock it.

4. PROPOSITIONS

4.1 The workspace and the live network of fragments for intensive collaborative

To manage the issues raised by fragmentation, we define the concept of *workspace* as the only place in which transclusion is authorized. Its default behavior is: every modification is always propagated; every user of the workspace has reading rights on all the fragments of the workspace (not necessarily writing rights); and very permissive transactions are allowed, mainly based on forking and post managing of concurrent accesses. A workspace manages a single large *live network of fragments* created for multiple documents' publications. Collaboration and reuse outside the workspace will base on more classical approaches: management of consistent and isolated set of fragments, cloning and exchange of dead versions of content, traditional rights management (see extensive collaboration below).

So a workspace is *open* as everyone can see each others' work. In order to introduce privacy in the workspace, private fragments can be authorized, but they will *not* be allowed to be referenced until they are publicly opened to reading. The workspace is also *alive* since the network of fragments is constantly evolving thought each modification of each user. A workspace is to be considered as a highly collaborative space, dedicated to a team for a common project, only that way it is able to manage highly fragmented *and* dynamic content.

But even so, such a dynamic system must ensure the authors that, whatever happens, they will be able to find again a content in a state they had identified before, for instance in the state one had left it before other users modified it. Scenari4 propose two main functions to protect content through dynamic modification: *automatic historization of each separate fragment*, and *manual versioning of network of fragments*. Every times a fragment is modified and saved, a new version is automatically created and the old one is kept as older version (as most wikis work). It ensures that any state of any fragment can be found again at any time by any user. Nevertheless, in highly fragmented context, finding back the state of a whole network looking for the right fragments one by one can be very difficult. The manual versioning of a network of fragments answers that: any user can at any moment ask the system to store a *snapshot* of a fragment along with all the fragments it references (recursively). The result is a dead (read only) sub-network (more precisely, a tree) of fragments, that can be consulted independently of the living network. On Figure 4, we visualize: on the left side some of the fragments that exist in the system; in the middle the current edition of the fragment "anomalies.chapter", which references the fragment "cfNoticesIndividuelles.chunk"; in the upper right frame the historization of changes that occur since the creation of fragment "anomalies.chapter", any previous state is so available for consultation; and in the lower right frame the versions "v1" and "v2" of the network composed with "anomalies.chapter" and its referenced fragments (for instance, "cfNoticesIndividuelles .chunk" will be available in the state it had when versions "v1" or "v2" of "anomalies.chapter" were set up).

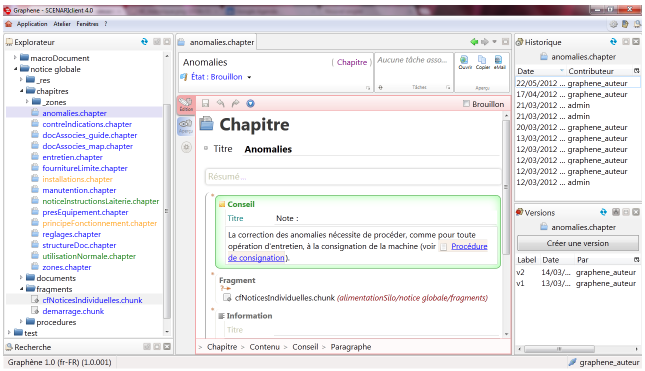

Figure 4. Scenari4 structured and fragmented content editor

4.2 The library and the folder-document for extensive collaboration

The workspace is a place for a local and intensive collaboration, dedicated to a coherent team for a common project. Whereas the workspace is very dynamic, the *library* is on the contrary a place for exchanging stable versions of content. We propose the concept of *folder-document* (foldoc) as a frozen extract of the live network (a sub-network with a root fragment, *i.e.* a tree) packaged with some of its readable views. A foldoc is composed with (see Figure 5): a single generative form (GF), XML fragments linked together extracted from the workspace; the model (M), *i.e.* the formal code necessary for GF manipulation; eventually some pre-calculated transformed forms of the GF (TFi), for instance HTML, PDF, other XML format...; and metadata (MD) allowing standard identification of the foldoc (Dublin Core description for instance). Technically a foldoc is a ZIP package with an XML manifest file describing its content.

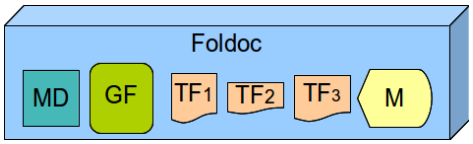

Figure 5. The folder-document (foldoc) structure

Libraries of foldocs are stable places to store and exchange content. GF and model make possible the future manipulation of the content by computers (to transform the content, to reintegrate it in a new workspace...), whereas metadata and TFs make the content searchable and readable by human beings. Foldocs are the mean to get back to a more classical grasp of documents, freezing and integrating parts of the networks that are living in workspaces. Libraries can so be implemented with classical ECMs (Alfreso, Documentum...), adding the function for introspecting foldoc packages. Integration of a native library in Scenari4 is also planned.

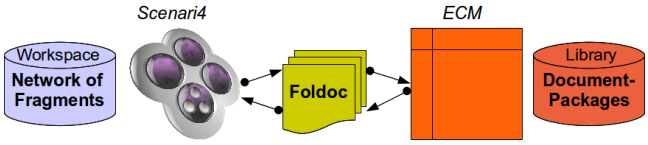

Figure 6. Workspaces and libraries

4.3 Other functions to manage transactions

Other functions have been designed and implemented to manage the issues related to transclusion, in order to maintain both the dynamism of the network and the consistency of each document represented in it. Their use is to be activated and parametrized depending on the context: Scenari4 is not a general ready-to-use system providing systematic functions, but mostly a generic framework to be specialized for specific organizational processes.

4.3.1 Automatic fragment locking

The system can automatically lock a fragment while it is edited in order to prevent concurrent modifications and unwilling overwriting or forking. Since it reduces dynamism, it is only available at the fragment level, to prevent from freezing huge parts of the network and locking many users' activities.

4.3.2 User information on system

The system can provide information on what other users do on fragments. It can help in monitoring what is occurring in the system. One simple example is an eye appearing in the XML editor when somebody else is reading the fragment, or a pen when he is editing it.

4.3.3 Diff and merge tools

Whereas it has not been integrated to the system yet, the C2M project permitted to research and prototype solutions to visualize differences between XML fragments and networks, and interactively merge them [8].

4.3.4 Planning and organizing of modifications

The system proposes the possibility to organize *a priori* the sequence of modifications by different users, and to program associated writing rights related to it. It is a more structured alternative, or complement, to permissive writing transaction processes.

4.3.5 The workspace derivation

To end with, the systems also provide a powerful mechanism of *derivation*. It is possible to create a virtual copy of a workspace, in order to be able to overwrite some fragments. The non modified fragments stay in the original workspace (and evolve synchronously with it), whereas overwritten ones (modified copies) are *informed* when their source has been modified (in order to be updated for instance). Derivation was originally used for content specialization in complex organizations (to complement transclusion, by re-introducing "under control cloning"), and for language translation purposes. Associated with a committing possibility - the derivative fragment is allowed to replace the original one by a specific user commit action - workspace derivation can also be used to work on draft copies of the content (without locking).

5. CONCLUSION

The C2M project permits to achieve Scenari4 software, made available under FLOSS license. Scenari4 is a unique solution for collaborative edition of structured and fragmented documents, independently of any specific model. Scenari4 is already used by Kelis with his customers, and helped this company to win the project of re-factoring of the back office of *service-public.fr*.

R&D is also continued through several experiments running: with Ina for the republishing of radio archives; with Quick restaurants for managing a common base of documentation serving several services; with 2IE Burkina Faso water and environment institute for training contents; or with Costech, the human sciences laboratory of UTC, for a scientific journal. We also designed two demonstrators: Graphene for technical documentation, and Webradio2 for multimedia publication (conferences, documentaries...).

Theoretical research is also pursued through: formalization of collaborative processes and context adaptation of publishing chains; elaboration of philological tools to help authors of fragmented and structured content to take decisions while working in highly dynamic workspaces; and document conceptual and logical modeling formalisms, tools and patterns for publishing chains design.

6. ACKNOWLEDGMENTS

Our thanks to the ANR for funding C2M project, to C2M partners for their collaboration, and to Sylvain Spinelli for his great technical and conceptual work on Scenari4.

7. REFERENCES

[1] André J., Furuta R., Quint V. (1989). *Structured documents*. Cambridge University Press.

[2] Bachimont B. (2007). *Ingénierie des connaissances et des contenus : le numérique entre ontologies et documents*. Lavoisier. Hermès.

[3] Crozat S. (2007). *Scenari, la chaîne éditoriale libre*. Eyrolles.

[4] DITA (2010). *Darwin Information Typing Architecture (DITA) Version 1.2*. OASIS standard. DOI=http://docs.oasis-open.org/dita/v1.2/spec/DITA1.2-spec.html.

[5] Gaillard L. (2010). *Modélisation rhétorique pour la publication de discours multimédias : applications audiovisuelles*. Thèse de doctorat de l'UTC.

[6] Nelson, T. H. (1981). *Literary Machines*. Mindful Press.

[7] Pédauque R. T. (2005). *Le texte en jeu : Permanence et transformations du document*. DOI=http://archivesic.ccsd.cnrs.fr/docs/00/06/26/01/PDF/sic_00001401.pdf.

[8] Vu X. T., Morizet-Mahoudeaux P., Geurts J., Crozat S. (2011). Extension d'un algorithme Diff & Merge au Merge Interactif de documents structurés. *Proceedings of CIDE.14*, Rabat, Maroc.

[9] Wilde E., Lowe D. (2002). *XPath, XLink, XPointer, and XML: A practical guide to Web hyperlinking and transclusion*. Pearson Education.