Integration of External Data in Document Workflows via Web Services

Davide Gazzè[1], Alessio Bechini[2], Marco Avvenuti[2], Maurizio Tesconi[1] and Andrea Marchetti[1]
[1]*IIT-CNR, Via Moruzzi, 1, I-56124, Pisa, Italy*
[2]*Dep. of Information Engineering, University of Pisa, Largo L. Lazzarino, I-56122 Pisa, Italy*

Keywords: Document Workflows, Content Management Systems, Web Services, Office Automation.

Abstract: In recent years the adoption of systems to automatically manage information assets in networked, collaborative ways has become vital for enterprises. In this work we focus on collaborative document management systems, where actual document development relies on accessing structured data in remote repositories. Forms within documents must be filled in with proper data items: An automated support to gather the required remote data may substantially improve the overall document management process. The Web Service technology is of fundamental help in this setting, because it lets remote data be exposed to potential clients according to standard interfaces. On the other hand, document management systems must be modified to provide this new kind of data access. This paper introduces a novel approach to support remote data integration in documents handled by Content Management Systems. According to the reported experience, such a data integration can be regarded as an enabling feature for a better exploitation of document workflow systems.

1 INTRODUCTION

Document management systems (DMS) are primary tools for enterprises to deal with information involved in business procedures (Baresi et al., 1999). Often the enterprise knowledge base can be practically identified with a collection of documents, to be handled from creation up to final archiving (Jones, 2007).

A significant portion of business procedures is carried out by operating upon specific documents, whose life cycle is determined by the progression of the business task. The automation of this kind of activities requires setting up both document templates and a precise definition of document flow (Rockley et al., 2002). From this viewpoint, documents related to a particular business process can be created as instances of a procedure-specific template, and their content experiences different and successive forms of *compilation*. Thus, according to (Marchetti et al., 2005), in the following we shall refer to Document Workflow (DW) as to a particular workflow in which all activities turn out to documents compilation.

Several actions in a document workflow require the end user to insert either non-structured or structured information. Although automated support to the former can be hardly provided, the latter could be more easily and proficiently dealt with, as in the case of exploitation of ontological knowledge bases (Bechini and Giannini, 2011). Most of the times the values to be inserted in a document are stored in an enterprise repository, or generally on a system that can be accessed via Internet. In the following, such kind of data will be referred to as *external data*.

In the last years, software tools dedicated to document management and Web Information Systems (WIS) have evolved side by side, and the advantages of their merging have been recognized very early (Balasubramanian and Bashian, 1998). A further evolution has led to embed DMSs directly within larger web application frameworks, namely Content Management Systems (CMS) and Enterprise Content Management (ECM) systems (Cameron, 2011). Surprisingly enough, the web infrastructure has been extensively used for easily connecting users to the DMS, but not to make external data directly available from the DMS.

In CMS/ECM frameworks, access to external data should be provided as a basic means to speed up document compilation. This can be achieved by *integrating* a specific functionality in the framework, and by accordingly modifying/enriching the standard document handling approach. Examples on how to seamlessly operate on DMSs to obtain an integration of this kind at the document repository level are reported in (Bechini et al., 2008) and in (Bechini et al., 2007). Anyway, external data are not always exposed to the

Gazzè D., Bechini A., Avvenuti M., Tesconi M. and Marchetti A..
Integration of External Data in Document Workflows via Web Services.
DOI: 10.5220/0004143203460351
In *Proceedings of the International Conference on Knowledge Management and Information Sharing* (RDBPM-2012), pages 346-351
ISBN: 978-989-8565-31-0

Internet or, if they actually are, not necessarily according to a standard interface. For this reason, we must specify how to implement data access on the repository side as well. To this aim, Web Services (WSs) provide a flexible solution that keeps the systems completely decoupled. Moreover, past integration experiences on DMS, based on WSs, have shown to be both effective and efficient (Bechini et al., 2008). In particular, (Chieu and Zeng, 2008) proposed to build up a whole Content Management System according to the SOA paradigm, using WSs as founding components. In a slightly different perspective, we advocate the use of WSs for data integration as a means to provide flexibility in the exploitation of document workflows, with no stringent need to reorganize the complete DMS according to a distributed, SOA-compliant design. The outlined ideas led to the implementation of a dedicated module in Drupal[1], one of the most popular CMS currently used: This system will be described to show how the proposed approach can be practically applied.

The paper is organized as follows: section 2 presents the typical way document workflows can be specified and handled in CMSs, taking as a reference the Drupal framework; in section 3, the proposed approach to integrate external data access in the standard document management procedures is discussed; case studies with practical workflows are shown in section 4. Conclusions are finally drawn.

2 DOCUMENTS AND WORKFLOWS IN CMSs

Nowadays Content Management Systems let users organize whole knowledge bases. To this aim, the specific mechanisms put in place may be different across different CMS platforms. We are interested in investigating how document workflow can be addressed; in the panorama of open-source applications, this particular feature is present in popular CMS/ECM platforms like Drupal and Alfresco[2], which adopt an event-based workflow management mechanism.

Drupal is a very popular open-source content management platform which has been used so far for the development of millions of websites and applications (Miles and Miles, 2011). It makes use of an abstraction of the document type, known as *content type*, which can be enhanced to accommodate also the access to external data. On the other hand, Alfresco handles document workflow using *content rules*. In

[1]http://drupal.org

[2]http://www.alfresco.org

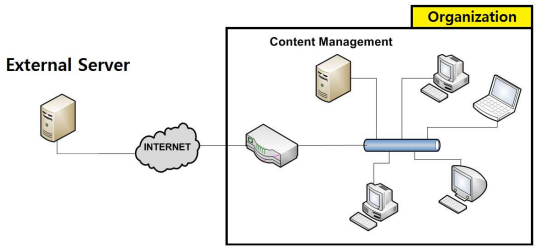

Figure 1: Access to structured information beyond the enterprise boundaries.

practice, the overall workflow can be considered as a sort of a collection of possible paths along dedicated spaces belonging to the involved actors. The movements of a single document across spaces are automatically driven by the system, according to rules defined at the space level.

In both platforms, the access to external data can be implemented with no substantial modification to their core software, exploiting the respective standard extension mechanisms and the underlying functionalities/modules. The Drupal's *content type* abstraction allows an elegant integration of remote data access, which is independent from the workflow progression mechanisms: This is the main rationale for focusing on an actual implementation of the remote data access system on this CMS platform.

In Drupal, each content item is represented as a *node*, whose structure and behavior are defined through the specific *content type* it belongs to. Hence, in a typical enterprise setting, it is a common practice to setup a content type for each document type required in business processes. In particular, content types might be defined to formally support ordinary administrative forms.

The overall framework is organized into *modules*, a sort of functional plugins. Custom modules can be developed and freely provided by members of the Drupal community. Modules can both rely on the CMS core functionalities, and exploit features of other modules. They can also be shaped to permit the customization of fields within node types. One particular basic module is crucial for supporting remote data integration: The *Content Construction Kit (CCK)*, which let the CMS administrator add *custom fields* just operating via a web graphical interface. Custom fields are intended to store additional structured information, respect to the system defaults.

Workflow management is available through the *workflow module*. Arbitrary workflows can be created and assigned to node types. Moreover, the *rules module* provides the ability to define actions that are conditionally triggered by occurring events. Thanks to such modules, the DMS administrator can design

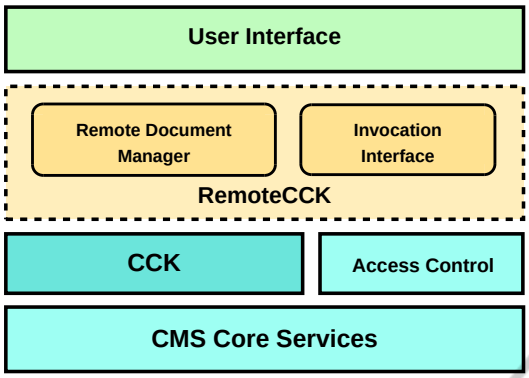

Figure 2: Architecture of the proposed solution. The RemoteCCK module is accommodated in a middle layer.

a document workflow for some of the content types created with CCK.

3 SEAMLESS INTEGRATION OF EXTERNAL DATA

The central problem tackled in this paper can be represented as shown in Figure 1. The access to structured data beyond the boundary of the enterprise information system must necessarily comply with constraints set by the external data server. Often the data hosting organization is interested in exposing part of its knowledge base via data access services. Furthermore, the easiest the access method, the larger the benefits that can be obtained. Thus it is not surprising finding out that nowadays many organizations choose to offer different services though the Web Services (WS) technology (Chappell and Jewell, 2002).

The widespread adoption of the WS technology is also determined by the modest coding effort to put into operation a data access service. In fact, modern IDEs for several languages are able to automate most of the required coding steps. On the other hand, the development of a WS "client" code is likewise really simple. These characteristics make WS a key technology for decoupling a complex service into small, manageable sub-services. WS have become pervasive in information systems of any kind, and their adoption in the proposed approach let us target the largest possible DMS users community.

Apart the basic choice of exploiting WS to connect data access providers and a DMS/CMS client, it is important to properly design the system architecture to take advantage of the existing functionalities in Drupal and to easily integrate the new modules in the standard software platform.

3.1 System Architecture

The overall architecture for the proposed integration system is organized according to a layered scheme, as shown in Figure 2. In the depicted scheme, the two lowest layers represent the core CMS and the *Content Construction Kit (CCK)* module[3]. They offer the basic functionalities to create document templates, manage the life cycle of each single node, perform versioning, create and manage users, carry out logging procedures, etc. CCK is devoted to handle the customization of document templates.

The remote data access function is embedded in *RemoteCCK*, the third layer of the system architecture. It contains the basic services of our enhanced data access system, which is able to exploit external WSs. It encompasses two distinct functional blocks, aimed at managing web services and users. In particular, the *Remote Document Manager* (RDM) block extends the CMS core functionalities by supporting the creation of document types, making them able to access and process data via WSs. We designed the RDM to let the user create a manual mapping between an existing content type and an "operation" (i.e. a data request towards WS), or a composition or WS operations. The *Invocation Interface* block has been expressly created to support the invocation of remote WSs from the classical web interface of a document.

The uppermost layer, the *User Interface*, offers the CMS administrator and ordinary users a web GUI to properly interact with the system. In its basic form, it is provided by the CMS, and it can be possibly modified and enhanced according to the needs of new custom modules.

3.2 RemoteCCK

In the remote data access sytem, two different user roles are defined: The *manager*, who is in charge of setting up the binding between content types and WSs, and the *document user*, who must be presented the outcomes of the WS invocations in a straightforward way.

Inside RemoteCCK, the RDM block supports:

- Analysis of WS specifications.
- Operation-driven content type creation and update.
- Mapping of operations to content types.
- Creation of WS templates.

[3]http://drupal.org/project/cck - In recent versions, the CCK functionalities have been moved to the CMS core.

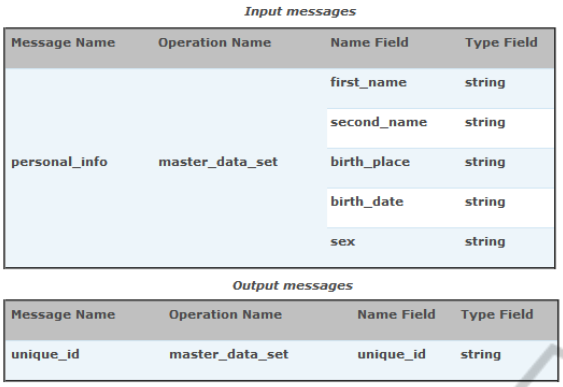

Figure 3: WS are inspected via their WSDL specification.

It is the Invocation Interface that actually calls WS operations, according to three different modes: i) Before, ii) During, and iii) After. In the "Before" mode, the WS is invoked at document creation time, i.e. before compilation start. In the "During" mode, the invocation is manually triggered from the web interface by the end user at compilation time. The "After" mode must be used whenever the WS has to be invoked at the end of the document compilation/submission. Regardless of the chosen mode, the operations to be carried out are: i) specification of the parameter values for the service request; ii) service invocation; iii) insertion of the returned values into the proper document fields. The WS operation parameters must correspond to values in specific document fields, and the returned values have to be inserted in other specific document fields.

3.3 Enhanced Content Type Specification

The binding of external WSs to document fields is set up by the manager following a two-step procedure: i) registration and analysis of a new WS, and ii) creation of a link between a content type and operation(s) of a registered WS.

In the first step, the URL for the target WSDL must be provided in input, and the corresponding WS is then analyzed in terms of Operation, Message, field name and type as described in (Chappell and Jewell, 2002). During the WS registration, additional information about the WS itself can be specified; typical examples are the login username, the password, and possible encodings. The graphical outcome of this process is shown in Figure 3.

The second step involves the *mapping* of a WS operation onto a content type field. RemoteCCK provides three alternative procedures to accomplish this task: 1) creation of a new content type, 2) update of an existing content type, or 3) manual mapping.

The first two procedures let the manager carry out the mapping automatically. This way, the manager is only asked to select a WS operation among the registered ones. In particular, the first procedure creates a new content type skeleton, whose structure mimics the WS interface. The second one behaves in a similar way, by adding new fields to an existing content type.

Conversely, in the manual mapping point-to-point links must be drawn on a graphical interface (see Figure 4). A list of the content type fields is depicted against a similar representation of the parameters of the WS operation; the manager can then easily draw lines to connect endpoints of the opposite sides.

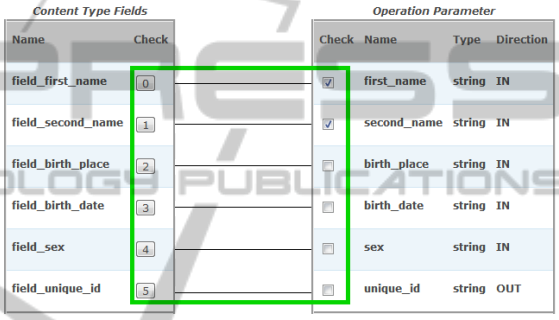

Figure 4: Mapping of operations onto content type fields.

3.4 Composition of WS Outcomes

To make the proposed data access and integration mechanism as general and flexible as possible, it must be possible to combine the outcomes from multiple (homogeneous) WSs. A straightforward way to perform this task can be based on the application of *aggregate functions*, which semantically correspond to the typical SQL-like ones like maximum, minimum, average, count and so on. How to employ an aggregate value depends on the content type semantics, and it is up to the manager to carefully specify how to compose and use it. In our system, the definition of WS composition is supported by *aggregation templates*, which make available the calculation in the form of a single reference, to be subsequently used by RemoteCCK for content type updates.

The User Interface for managing an aggregation template is quite complex and let the manager choose different operations from many WSs. For the sake of simplicity, the whole process can be divided in two phases: A preliminary selection of the involved WS/operations and the aggregate function, and the subsequent mapping between fields and operation parameters. This last task can be graphically organized as shown in Figure 5.

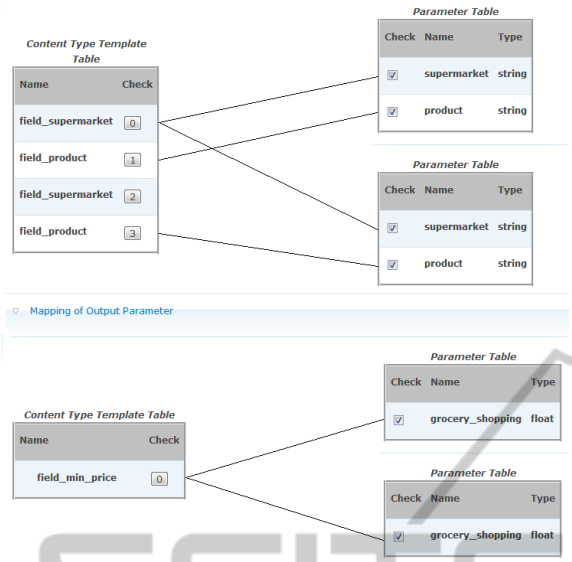

Figure 5: Graphical specification for aggregations.

Figure 6: WS-based access to a remote master data set.

4 CASE STUDIES

The validation of the proposed mechanism has been organized in two parts: In the former, remote data access and composition is tested; In the latter, a proficient integration within document workflow systems must be achieved. In a preliminary experimentation, WSs have been developed to access a master data set. In such a data model, every employee has a personal and unique ID. In document workflows each involved employee, as document user, is asked to insert personal data. Thus, a specific WS can be developed to obtain such information, given an employee ID. On the DMS side, different content types can actually insert the required data into the proper document fields, upon the related WS invocation (as shown in Figure 6). We can complete the first validation part by defining a report with the current cheapest price for a product sold in a given list of malls. To this aim, an aggregator can be used to determine the lowest value out of the prices returned by different WSs; the speedup due to the automated retrieval of distributed information is here particularly evident.

In both the described cases, the proposed mechanism has shown to be effective and flexible.

4.1 Integration in Document Workflows

In the next example, we want to check the use of RemoteCCK within the ordinary Drupal Document Workflow System. Indeed, we can observe that RemoteCCK relies on CCK, whose functionality is typically exploited in a Document Workflow context, and thus a straightforward integration is expected.

An integration test was performed by implementing a typical example of use: A “vacation request form” to be issued to the administration by an employee. In the first place, we must underline that in Drupal workflows a specific field may be editable for one user, only readable for another, or even hidden for a third one. The workflow, which describes the steps the application must undergo before being approved or rejected, involves multiple actors and can be described as in Figure 7: The employee, using the proper form, in the first place applies to the *Administration* and to the *Manager*. Then, upon both approvals, the document passes to the *Director* who will forward the final outcome to the *Human Resource Office* for the final notification to the requester. Note that in case of rejection at any level, the employee must be always notified about it.

In this use case, the document progression has been correctly managed, taking into account user roles, assessment of conditions, and leveraging also the “document creator” concept to determine the notification recipient.

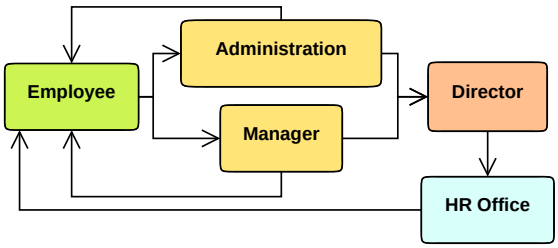

Figure 7: Workflow for a Vacation Request Form.

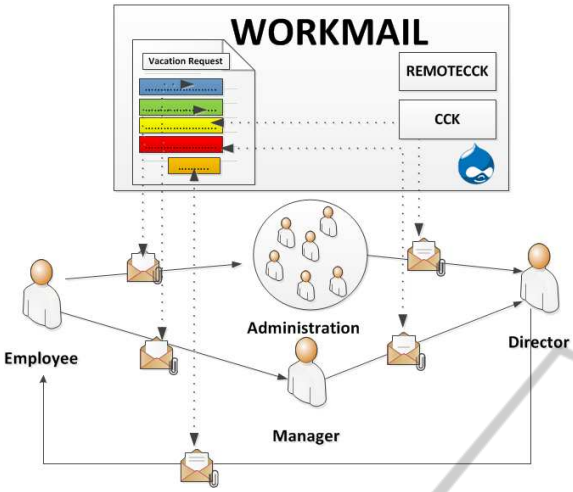

Figure 8: WorkMail can seamlessly integrate RemoteCCK.

4.2 Integration with WorkMail

WorkMail (Gazzè et al., 2012) is a custom, fully configurable Drupal module for documental workflow. It is based on using emails to drive the compilation flow of shared documents. Users can access a document according to their roles. A special content type, namely *WorkMail attachment*, is dedicated to support the workflow by including information about the recipients, the subject, the body, and status flags. The *WorkMail engine* is in charge of regulating the users' accesses to the document, according to their permissions. One of the main strenghts of the WorkMail system is its ability to resort to a delivering medium (emails) that is particularly familiar to the end users, thus making them easier collaborative jobs.

WorkMail is based on the CCK modules, and this makes the RemoteCCK integration automatic. Users that access the document via the WorkMail engine can benefit of all the RemoteCCK features, as shown in Figure 8.

5 CONCLUSIONS

The efficient employment of document workflows requires an integrated access to data residing beyond the boundaries of the enterprise information systems. The proposed solution can be organized in a module, called RemoteCCK[4], to be seamlessly placed in the Drupal framework; the new functionalities can be proficiently and flexibly used both in the ordinary DMS environment, and in additional custom modules. The possibility to graphically specify the inclusion in documents of plain/aggregated external data (accessed via WSs) makes the module easy to use.

In its current implementation, at runtime RemoteCCK does not deal with possible modifications of the registered WS operations; this problem could be overcome by carrying out a WS validation step, which may be included in future developments.

[4]Available under GPL2 license at https://drupal.org/sandbox/davide.gazze/1360152

REFERENCES

Balasubramanian, V. and Bashian, A. (1998). Document management and web technologies: Alice marries the mad hatter. *Commun. ACM*, 41:107–115.

Baresi, L., Casati, F., Castano, S., Fugini, M. G., Mirbel, I., and Pernici, B. (1999). WIDE workflow development methodology. *SIGSOFT Softw. Eng. Notes*, 24(2):19–28.

Bechini, A. and Giannini, R. (2011). Management of genotyping-related documents by integrated use of semantic tagging. *T. Large-Scale Data- and Knowledge-Centered Systems*, 4:15–39.

Bechini, A., Tomasi, A., and Viotto, J. (2007). Document management for collaborative e-business: Integrating ebxml environment and legacy dms. In *ICE-B 2007 - Proc. of the International Conference on e-Business*, pages 78–83. INSTICC Press.

Bechini, A., Tomasi, A., and Viotto, J. (2008). Collaborative e-business and document management: Integration of legacy DMSs with the ebXML environment. In *Interdisciplinary Aspects of Information Systems Studies*, pages 287–293. Physica-Verlag HD.

Cameron, S. (2011). *Enterprise Content Management: A Business and Technical Guide*. British Comp Society Series. British Computer Society.

Chappell, D. A. and Jewell, T. (2002). *Java Web Services - using Java in service-oriented architectures*. OŔeilly.

Chieu, T. and Zeng, L. (2008). Service-oriented approach for implementing an extensible content management system. In *Congress on Services Part II, 2008. SERVICES-2. IEEE*, pages 96 –103.

Gazzè, D., La Polla, M., Marchetti, A., Tesconi, M., and Vivaldi, A. (2012). WorkMail: Collaborative document workflow management by email. In *Proc. of 9th Int'l Conf. on Cooperative Design, Visualization, and Engineering*, LNCS. Springer.

Jones, J. I. (2007). *The Document Methodology: for Enterprise Analysis, Second Ed.* AuthorHouse.

Marchetti, A., Tesconi, M., and Minutoli, S. (2005). XFlow: An XML-based document-centric workflow. In *Proc. of WISE 2005*, volume 3806 of *Lecture Notes in Computer Science*, pages 290–303. Springer.

Miles, E. and Miles, L. (2011). *Drupal's Building Blocks*. Addison-Wesley Professional.

Rockley, A., Kostur, P., and Manning, S. (2002). *Managing Enterprise Content: A Unified Content Strategy*. Pearson Education.