# Information System Modeling for Effective Information Testing

**Michel Lanque**
Alcatel-Lucent
Centre de Villarceaux
Route de Villejust
91620 Nozay (France)
Tel : +33 1 3077 10 84 – email : Michel.Lanque@alcatel-lucent.com

**Philippe Larvet**
Independant Consultant
24520 Lamonzie-M. (France)
Tel : +33 6 1570 4510 – email : phlarvet@gmail.com

**Abstract**: This paper presents a process for an effective testing of a modeled information system, as well as a contribution to performance of using complex software systems. This approach allows performing technical and functional validation of a system, directly from the contents of the technical documentation for the system, such as use cases descriptions considered as input data.

**Key words**: Technical documentation, complex system documentation, information system, system test, system validation.

## 1. DOCUMENTING COMPLEX SYSTEMS

### 1.1 Context

Designing and developing industrial products generate more and more complexity in the management of technical information, especially if products are web-based applications. The reason for this increasing of complexity comes from internal interactions within the system itself and from interactions of the system with other systems, but it comes also from the fact that, more and more, systems have to cover various technical information within different domains.

Like software products integrating web-based applications, information is at the same time unidirectional and autonomous within the context of the concerned topic, and multidirectional if we consider its capacity to be associated with other information domains related to the concerned topic.

The volume of information also increases due to the necessity to take into account existing information (legacy). New data must be integrated with the legacy and have to be implemented according to the policy of the company.

Consequently, a software application, considered as an industrial product and including all the levels of operational functions (for example local operations, networking management and monitoring, hardware management in nodes management and end-to-end operation), demands complex subsets of technical information for a complete and consistent specification.

### 1.2 Product performance and user information

Most part of technology products consist in software products including a Graphical User Interface (GUI) as a key element for all user tasks (installation, administration, operations and maintenance).

That means the product can be essentially seen as an information system enabling user tasks though a GUI that exposes and embeds a lot of technical information.

Consequently, the performance of the product depends on the level of quality of the information provided to the end users. Technical information, its modeling and use to help the validation of the system, is the topic on which we focus in the present paper.

### 1.3 Exchanged data

As exposed before, the information provided to the user is an important part of the performance of the product. In software systems, the key element for providing information to the user is the GUI.

So, for most of software applications, the product ***is*** the Graphical User Interface.

Consequently, the functional information displayed to the user or referencing information components (documentation, on-line help, etc.) is part of product performance: quality of data themselves, understandability of actions, message interpretation, processed time according to results, knowledge skills vs provided information, etc.

Using the GUI, the priorities of end-users mainly concern the tasks for operations and maintenance, through the necessity of an immediate and efficient access to the right information ready to be used. However, the end-user also needs to be able to identify the information that requires specific tasks: emergency actions, corrective maintenance, monitoring for preventive actions, etc.

These points refer to the technical contents of documentation and/or information embedded in the software application such as on-line helps and system information displayed via the GUI.

In fact, the content of this documentation can be defined from the technical specifications of the system. The way the information is managed and provided to the end-user to help him to perform his tasks has a direct impact onto the good use and performance of the application.

### 1.4 Technical specifications and product documentation

Technical information is the reference for any R&D team who has information to update and create within the context of developing new functionalities of a system. New information is built from the existing information, which is mainly included in a technical documentation.

Technical documentation is developed for many kinds of users: R&D teams, for product development and testing; technical writers, for producing customer documentation; technical support: hot line, installation and maintenance teams, etc. ; training teams.

Given the mass of technical information more and more increasing in complexity and open mapping of documentation contents accessible everywhere, the user requirements go over the documentation topic. The necessity to access pertinent information and the consistency of this information involve the control and the management of the whole documentation contents. That requires skills (specific and full) on the technical domains covered by the documentation contents.

An increasing volume of information for complex systems requires more and more control for reliability, security and quality of treatment and processing. This requirement directly concerns the performance of systems delivered to operators, experts and final users – then, as exposed before, this concerns the reliability and the quality of technical information.

Consequently, a pertinent question can be asked: how to check the reliability and quality of the technical information and guarantee its content consistency, required by R&D teams in their work and by end-users in their operational tasks?

The approach we propose is to ***test*** the technical information, as if it was made of software components. This testing approach enables to target the pertinent information and helps to update it, keeping the technical consistency according to the legacy documentation.

### 1.5 Approaches around documentation testing [1]

The ways for controlling and validating the technical information could be based on several techniques. The use cases that come from the technical specifications could be verified by an end-user through a manual testing of operations [2]. The entire product could be tested with the operator documentation. Customers and end-users could be involved within the product life cycle (first off, etc.). A system able to target the operational information needed by the end-user could be built, this system processing this information as a software code enabling a testing process.

This last point is the solution we have selected according to our objective to guarantee the technical information consistency compliant with the use of the product.

## 2. CONTEXT OF OUR STUDY

### 2.1 An "information module" is an "object"

The study consists in a new approach for technical system testing based on a rational set of functional information. In other terms, pieces of information are first defined by an information type (functional, descriptive, etc.) and metadata (summary, pre-conditions, operation, result, etc.). This enables us to manage them as "information modules" or objects as it is used in software development.

The study focused on ***operational information.*** First, this category of information is the most important part the end-user requires for his tasks (daily tasks, specific tasks, and emergency ones). Second, all technical information delivered to the customers relies on users operations (Use Cases) and maintenance tasks in priority [3]. Third, software products are mainly used through GUI that display operational information from embedded information systems.

With the proposed approach, our objective is to improve the quality of operational information from technical documentation, and mainly the followings: customer-specific technical documentation, product or user documentation, product training documentation, extranet for customers, on-line help and documentation embedded in the product, and technical support. The technical information contained in these documents (product data sources) represents the starting point of an automated process for building an information/product operational system. That means the operational information is the key element to test completely in order to insure the technical consistency with the use of the product.

### 2.2 Use cases as main source of information

Our main source of information is Use cases, that represent the user tasks – product / system use [4]. Use cases are formally expressed in operator's documentation as "procedures" [5]. A procedure is a user task description, expressed step by step, for operations actions.

To face the user product requirements about the operational performance and the high quality level of the completeness on all tested functions before the final product delivery, it is interesting to have a solution vouching the user operations without additional costs during the development and after it.

## 3. "DOCUMENTATION ENGINEERING"

The process we have selected is based on three main ideas. First, a ***single sourcing process*** of technical information development enables the reuse of information and the automated application of a semantic analysis approach [6]. Second, every ***elementary information*** is seen as a "module of information", similar to an object within the context of object-oriented development. XML format has been selected for elementary information management and indexation [7]. Third, these ***information modules are managed as software code*** for building associated combinations in rational operations such as product information architecture, task-oriented information model, etc.

Through this process, we insure the compliance with information development standards [8] with usual tools for writing, editing, checking and accessing documentation contents. Such as XML for the source of the format of "information module" files, we used standards for structured documentation (DITA [9], S1000D [10], etc.), data bases (CMS [11]), writing and editing environments (XML Editor, Wiki, MS-Word®, Framemaker, etc.).

### 3.1 The detailed process

Our process of "documentation engineering" (see Fig. 1) leans on the following steps:

1. A ***semantic analysis*** of the product specifications [12] (or the equivalent technical documentation) allows identifying Use Cases structures, even in the content of informal documents.
2. A dedicated ***extraction of the Use Cases*** from technical documents allows expressing them in a compliant, structured format.
3. A specific construction allows producing ***formal procedures*** [5] with the content of structured use cases. These procedures, containing user information modules, are saved into XML format.
4. ***Storing the modules*** in a database enables indexing them from their metadata.
5. Finally building the ***functional architecture of the product*** through the assembly of the formal procedures gives a virtual representation of the future functioning of the system, of the functional consistency of data and gives a status report of operational information modules to be reviewed (missing or duplicated data, inconsistent task processing, compliance or non-compliance, etc.)

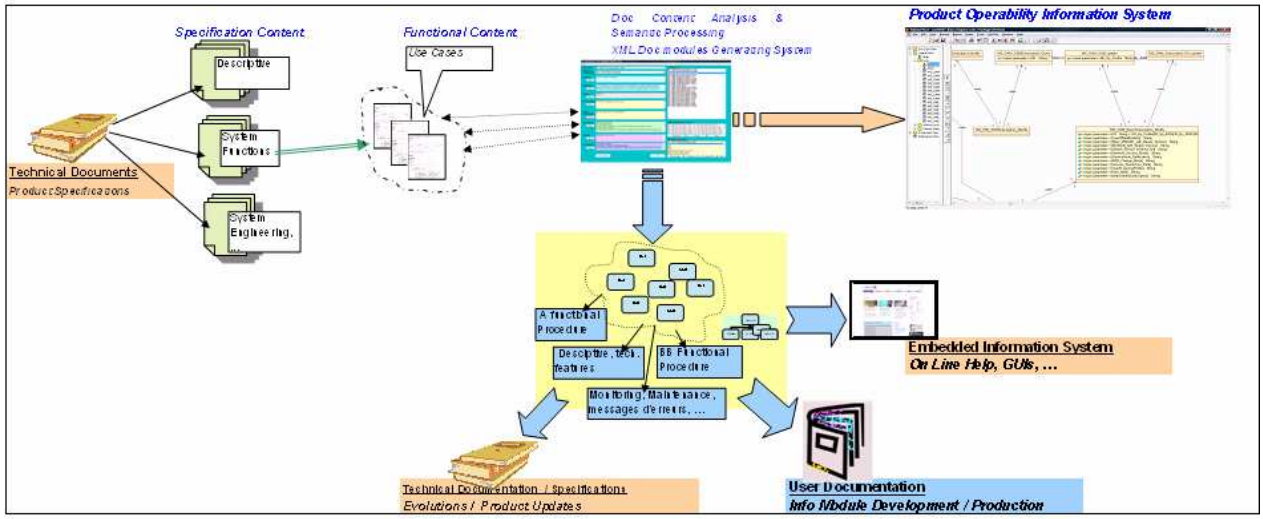


Figure 1. Documentation engineering

Notice that the "functional architecture" of the last step above can be expressed under the form of a UML model, which is a way for testing the consistency of modeling functions and data within the information architecture model.

### 3.2 Main concepts as results

The main idea is that a documentation (= a set of documents) is seen as a software.

An interesting consequence is that modular techniques and object-oriented techniques – valid for software – are also valid for documentation.

Like software, technical documents contain reusable information modules that exchange data one to each other (see Fig. 2).

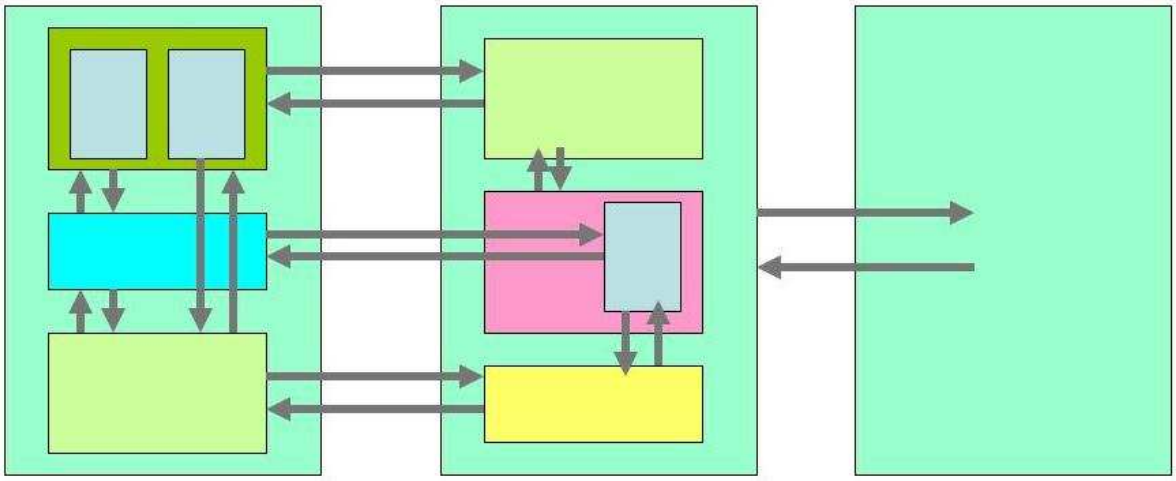

Figure 2. Information modules, like software objects, exchange data one to each other

## 4. INFORMATION MODELING

From the previous description of the proposed process, we can clearly expose the key steps of Information Modeling (see Fig. 3):

1. The technical documentation for a product, made of pieces of information, is seen as software.
2. Modules of this "software" are "information modules".
3. Each module can be seen as a kind of programmable component.
4. All these components can be assembled in a "functional architecture".

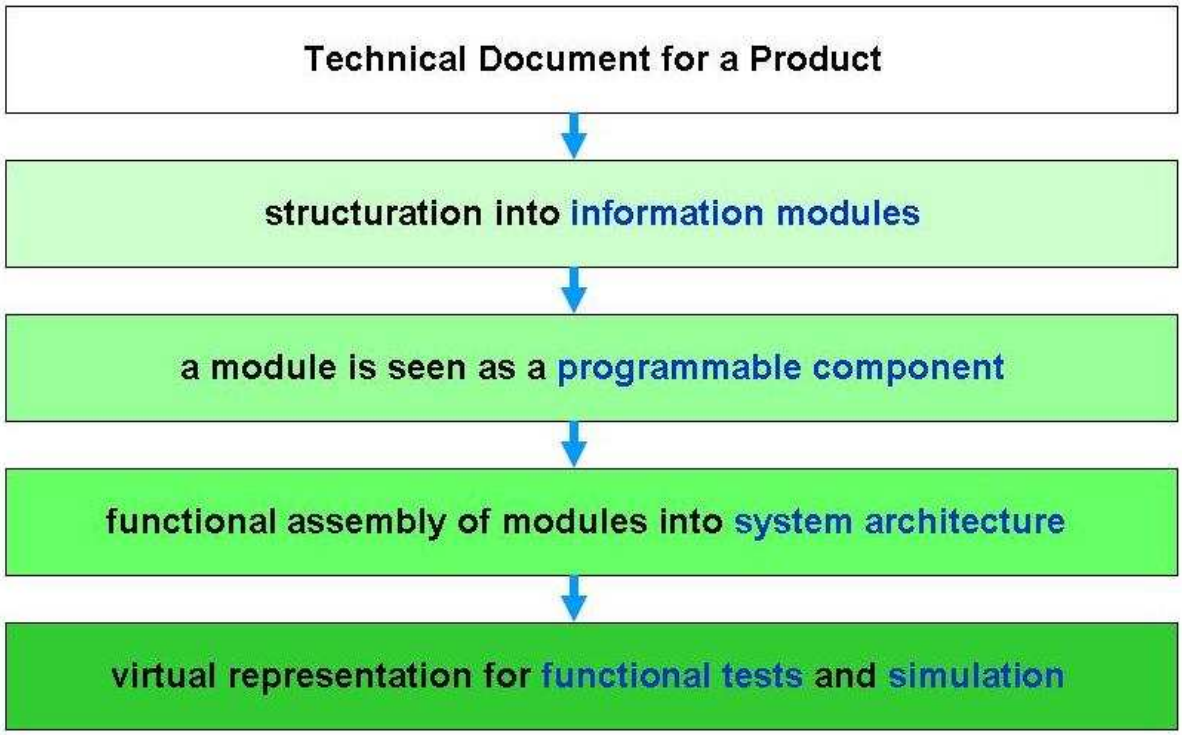


Figure 3. The key steps of Information Modeling

5. This architecture is a virtual representation of the future system, and this representation can be used for functional tests, verification, validation, and simulation.

### 4.1 Functional information module

Within our approach, a functional information module is used as an active object, defined through some precise characteristics: pre-requisite, input parameters, description of functioning steps (procedure), expected results, exceptions due to input data values.

From these items, the active information module acts as a functional module, i.e. taking into account the input parameters, performing step by step, action by action, its internal procedure and generating, for instance, logical operators. Such operators can take values like, for example, "true result" = compliant vs expected result, or in contrary, "false result" = not compliant. In other cases, these operators can give information about the values of input parameters or the status of pre-requisite, etc.

So, as the information module directly comes from the textual content of the documentation, it means that the functioning of the module, as exposed above, is able to test the information procedure, its data and process. Consequently, the functional architecture being an assembly of information modules, simulating this architecture and checking the simulation results is a big help for the verification and validation of the virtual system behavior.

### 4.2 Module management

The module management relates to some precise criteria, to be used in a global user information system (internal to a company) before delivering the modules to customers and end-users. Access to this information system could be done via the extranet for example.

Within the context of our study, we have selected three main criteria for the module management:

1. Information modules are stored in a Content Management System (CMS).
2. After a semantic processing, each module becomes a typed component – as if it was a piece of software.
3. Each module has a structure containing:
    a. a unique identifier for module management
    b. several metadata for characterization:
        i. type of information (descriptive, functional, alarm, etc.)
        ii. language (French, English, Spanish, etc.)
        iii. domain / product family (hardware, software, etc.)
        iv. etc.
    c. a precise structure according to its type (DTD, XML Schema, HTML, etc.)
    d. an informational content
    e. links, modular references and associations.

We consider that DITA [9] for example, could be a good choice for structuring information modules [13] and helping their storage in a CMS.

## 5. INFORMATION TESTING = ARCHITECTURE VERIFICATION & VALIDATION

The assets of Information Testing go over the documentation compliance. As seen before, a functional architecture can be built by assembling a set of information modules that are directly extracted, built and stored from documentation content (see Fig. 4) – this documentation being for instance the specification of a future product.

As a logical continuation of our study, we propose to consider an automated process in order to simulate the functional architecture, doing by this way a kind of processing of the virtual product. This simulation process of "information testing" opens wide possibilities to a real architecture verification and validation.

Capabilities of this simulation process are numerous: validation of operational procedures; functional consistency checking, notably through the test of product evolutions (new functions); technical validation of document contents

before their diffusion to R&D teams and customers; operational validation of test/integration plans; test of interoperability (consistency of links), i.e. automated validation of modular architectures of functional information; automated reliability i.e. test of end-users documents (installation, administration, maintenance, .) vs product lifecycle; functional simulation, notably within the context of a new modular architecture in order to help the design of a new product.

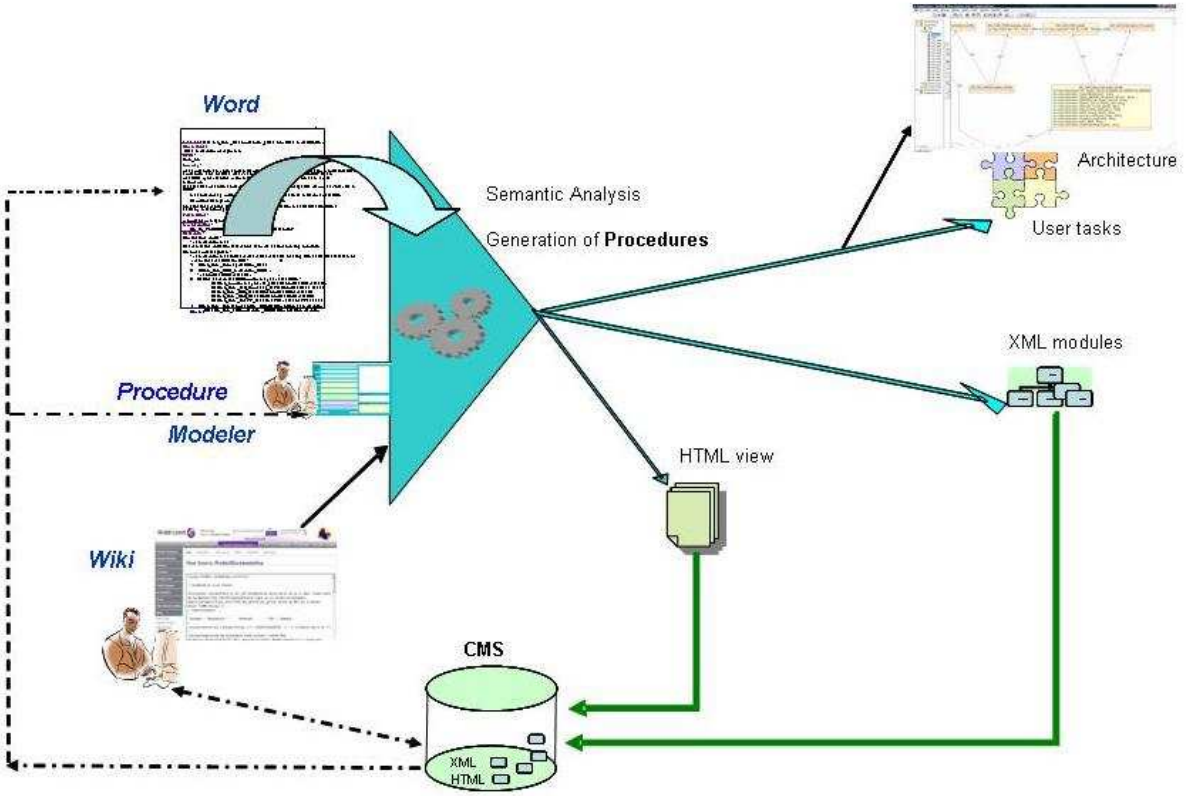


Figure 4. Building a functional architecture from information modules extracted from documentation content

## 6. CONCLUSION

Within the context of technical documentation or more precisely “user information systems from use cases”, defined for the specification of an industrial product, we have described an automatizable process to transform any textual technical information into a set of efficient information modules able to be assembled and executed as a software application.

The benefits of this process can be easily implemented within an R&D process with interesting possibilities. First, technical and functional information become programmable system modules. And second, this view allows an immediate optimization of modules operability according to functional contexts: anticipation of product evolutions; system test, Verification & Validation; reliability and consistency checking; simulation of system behavior.

We are currently implementing this process into an automated tool, in order to make it more concrete and to propose a good help for system verification & validation, directly derived from the technical documentation of the future system.

## *References*

[1] IEEE 829-1998, also known as the 829 Standard for Software Test Documentation, is an IEEE standard that specifies the form of a set of documents for use in eight defined stages of software testing, each stage potentially producing its own separate type of document. The standard specifies the format of these documents but does not stipulate whether they all must be produced, nor does it include any criteria regarding adequate content for these documents. These are a matter of judgment outside the purview of the standard.

[2] Alessandro Fantechi, Stefania Gnesi, Giuseppe Lami, Alessandro Maccari, *Application of Linguistic Techniques for Use Case Analysis*. In Proceedings of RE'2002. http://fmt.isti.cnr.it/WEBPAPER/RE02-revfin.PDF

[3] Jaroslav Drazan, Vladimir Mencl, *Improved Processing of Textual Use Cases: Deriving Behavior Specifications*. Organization: SOFSEM (2007)
http://d3s.mff.cuni.cz/publications/download/DrazanMencl-ImprovedUC-SOFSEM2007.pdf

[4] Kurt Bittner, Ian Spence, *Use case modelling,* Addison-Wesley (2003), p. xvi. ISBN 9780201709131

[5] Michel Lanque, Philippe Larvet, *Automatic Procedure Building from Use Case Detection within Informal Technical Documents,* CIDM (2010) http://www.infomanagementcenter.com/enewsletter/2010/201001/fourth.htm

[6] Samhaa R. El-Beltagy *Ontology-Based Annotation of Text Segments*, Computer Science Department (2007), Cairo University. Giza, Egypt http://www.claes.sci.eg/coe_wm/Sac07.pdf

[7] Jean-Pierre Chanod, Boris Chidlovskii, Hervé Dejean, Olivier Fambon, Jérôme Fuselier, Thierry Jacquin, Jean-Luc Meunier, *From legacy documents to XML: A conversion framework*, In Proceedings of European Conf. Digital Libraries (2005) http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.73.1502

[8] OASIS specification tools and guidelines. Providers of tools and guidelines [2002-07-06] Eve Maler, Norman Walsh (Sun Microsystems).

[9] DITA Version 1.1 Language Specification, OASIS Standard, (August 2007)
Chair: Don Day, Editors: Michael Priestley, Robert D. Anderson, JoAnn Hackos

[10] S1000D is an international specification for technical publications, using a Common Source Database
http://www.s1000d.net/

[11] A Content Management System is a family of tools providing a collection of procedures used to manage documents in a collaborative environment.

[12] Philippe Larvet, *Semantic Application Design,* Bell Labs Technical Journal, Aug.2008, Volume 13 Issue 2, Pages 75 – 91, http://portal.acm.org/citation.cfm?id=1405607 and
http://www3.interscience.wiley.com/journal/121376597/abstract?CRETRY=1&SRETRY=0

[13] Philippe Larvet, Michel Lanque, *Generation of XML-DITA Procedures from Use Case Detection in Informal Technical Documents,* DITA Europe 2009 Conference, Novembre 2009, München, Germany
http://www.infomanagementcenter.com/DITAeurope/2009/abstracts.htm