DOCUMENTS AS AN ENABLING MECHANISM FOR CONCURRENT ENGINEERING IN THE CONSTRUCTION INDUSTRY

R. Amor[1] and M. Clift[1]

ABSTRACT

In current engineering practice, be it concurrent or otherwise, documents are the central mechanism for communicating, informing and instructing. Any attempt to engender a greater uptake of concurrent engineering in the industry has to recognise the central role of documents. To date, the various aspects of IT applied to engineering have developed independently, leading to stand-alone product, process and document management systems. This development path, though productive in each individual area, misses the major gains that can be achieved from integration of all aspects of IT usage. This paper shows that the management of documents provides information about all aspects of a project. It is clear that, through careful management, documents can provide the means to co-ordinate work on the activities required to complete a project, and to determine how processes can be managed to greatest effect using concurrent engineering frameworks.

Keywords - Document Model, Document Management System, Concurrent Engineering

1. INTRODUCTION

Documents are the central mechanism for managing, communicating and contracting in an engineering environment, in particular when considering it within the construction industry which is made up of virtual companies formed for a short duration when compared to the life of the product. Manufacturing industries are more commonly formed of long established organisations producing relatively short life products, in contrast with the expected life of many buildings. However their approach to documentation and its purpose is more sophisticated and better supported by IT than that generally adopted by the construction industry. Such IT support that is available has been developed independently leading to

[1] Building Research Establishment, Bucknalls Lane, Garston, Watford, WD2 7JR, UK

company or project stand alone product, process and document management systems, lacking the capability for the immediate integration essential when quickly forming new temporary organisations. These often comprise thirty to forty SMEs (small to medium enterprises have less than 500 employees) on an average construction project costing around £2m.

This paper presents the work, led by the Building Research Establishment, being undertaken as part of the EU-ESPRIT funded project 'Towards a Concurrent Engineering Environment in the Building and Engineering Structures Industry' (ToCEE) [1], that examines models of documents and their interaction with the engineering process within the construction industry. As the project runs from January 1996 to December 1998, the paper represents work in progress and the project team would welcome comment.

2. THE KEY OBJECTIVES OF TOCEE

There are many definitions of concurrent engineering, but the view adopted by ToCEE is that it represents co-operative or collaborative working. The main goal of the ToCEE project is the development of a conceptual framework and a prototype environment to support and encourage concurrent engineering by a wide range of organisations in the architectural, engineering and construction domain. To achieve this goal ToCEE is building upon the results from many previous EU funded projects:

- in product model based integration, by STEP [2], with parts 41-45, 103, 106, and APs 225, 230 and 231 and by several EU projects, e.g., ATLAS [3], CIMsteel [4], COMBI [5], and COMBINE [6]
- in process modelling, e.g., by the CALS [7] initiative
- in workflow developments and electronic document management, by the WfMC [8] as well as in several commercial [9, 10] and research [11, 12, 13, 14, 15] based EDM systems

However ToCEE extends the scope of these projects to include several important concurrency enabling aspects with their mutual interrelationships. The focus of ToCEE is on the holistic approach to design, the management of the information flows in the multidisciplinary collaborative work processes, and the legal aspects of electronic communication and information exchange.

The primary objectives of the ToCEE project are the development of an overall conceptual framework along with specific software tools for concurrent engineering support. Some of these tools are intended to be applied directly within current software implemented by end-users on completion of the project with others being developed and tested as prototypes in the longer term. This short term exploitation and long term research is driving the approach of the ToCEE consortium, allowing it to focus both on the application of sophisticated AI-methods for the solution of complex tasks and on the rapid introduction of advanced state-of-the-art information technology in everyday practice.

Key issues for a successful concurrent engineering approach being addressed are:

- distributed process, product, document and regulation requirements modelling with special focus on intra- and inter-model operability
- inter-discipline conflict management
- legal aspects related to the product data and the electronic documentation

- information logistics and communication management
- monitoring and forecasting
- cost control

Models being developed by the project under separate work packages cover process, product, logistics and documents, with cross-cutting themes of legal issues, conflict management and standards and regulations.

3. OBJECTIVES OF THE DOCUMENT MODEL

The document modelling work package within ToCEE defines the requirements for an electronic document management system (EDMS) able to satisfy the requirements of all the players involved in a construction project for formal documents. Typical formal documents currently include information covering:

- briefing
- design decisions
- progress records
- specifications
- schedules
- bills of quantities
- drawings
- contracts
- site diaries
- design
- calculation records
- operation and maintenance

These documents represent a series of views of the common product information, the instantiated product model, fulfilling legal requirements of the product and the production process as required by both contract and common law.

In current practice documents are usually distributed and held in paper form requiring a costly resource-intensive filing, retrieval and issue system. The ToCEE document model addresses the enormous administration costs associated with document handling (estimated at 30-40% of an engineer’s work effort). The scope and layout of documents are determined by practice and may be laid down in standards which differ from company to company and, of course, country to country. The document model must be sufficiently generic to handle the variables required by different practice.

A conceptual model of an electronic system to replace and enhance the traditional procedures must be produced. This system will allow the integration of all documentary information as well as creating audit trails to be used for quality assurance and legal purposes. The document model will be closely tied to the product and process models in ToCEE to help facilitate the concurrency aspects of the project and in recognition of the vital link documents play to the relationships between these models in a running project.

Particular emphasis is paid to how such a system can keep track of document version numbers, the interrelationships between them, and their auditability. This is managed whilst retaining the document character of electronically stored data. The model has to have an open architecture which can be easily adapted by users to meet requirements particular to them. It must address current and emerging standards for project information, making use of existing systems where appropriate and seeking to influence their future development so that they facilitate the future requirements of electronic concurrent (collaborative) engineering. Although the document model is specified independently from the product and process models there must be very close links between the models to help maintain the legal and auditing requirements of the emerging concurrent engineering environment. Appendix A describes the top level schema for the envisaged document model.

4. SCOPE OF AN ELECTRONIC DOCUMENT MANAGEMENT SYSTEM (EDMS)

The meaning and scope of the word ‘document’ in the traditional design and construction process was always clear. A document was any collection of paper that related to a project. Several distinct styles of data layout were used to present information in these documents, as indicated in Table 1. The physical nature of paper helped define its (contractual/legal) status as a document, but as we move into an age where an electronic representation of all traditional paper documentation is possible, the definition of a document becomes blurred. Some paper documents may not currently be able to be sent electronically (e.g., standards) and some forms of information that are sent electronically were not considered documents in traditional practice (e.g., e-mail or telephoned orders).

To define the scope of electronic documents we use a modification of the paper document definition to specify that:

‘An electronic document records any transfer of information which occurs during a project and it must:

- provide views of the product model
- support all processes in a project.’

This greatly broadens what can be considered as a document (we prefer the term 'information container'), it still covers everything which has a paper form in Table 1, but also extends it to information transfers such as:

- phone calls
- a colleague's advice
- project discussions
- data files used for a design tool
- video clips of the construction site, etc.

Type	**Form**	**Author**	**Authority**	**Legal Status**
Brief	paper	client/owner	signature	high
Contract	paper	client	signature	high
Drawing	paper	designer/ contractor	signature	high
Specifications	paper	designer	signature	high
Bill of quantities	paper	quantity surveyor	signature	medium
Program/ Schedules	paper	contractor	signature	high

Calculations	paper	designer/ contractor	signature	medium
Site diaries	paper	contractor	signature	medium
Variation orders	paper	designer	signature	high
Progress records	paper	contractor	signature	medium
Claims	paper	contractor	signature	high
Letters	paper	all	signature	high
E-mail	electronic	all		low
Fax	paper	all		low
Phone	verbal	all		low
Verbal order	verbal	designer		low
Request for information	paper	contractor	signature	medium
Confirmation of instructions	paper	designer	signature	medium
Notices	paper	client	signature	high
Tender documents	paper	designer	signature	high
Valuations	paper	quantity surveyor	signature	medium
Payment certificates	paper	designer	signature	high
Purchase orders	paper	contractor	signature	high

Table 1. Documents in current construction practice

Although an electronic document records any transfer of information it need not contain the full content of the information transfer. For example, an electronic document representing a verbal order is likely to contain the essence of the order rather than the whole audio capture of the conversation (though this may be recorded if required). In many cases the electronic document may capture a reference to existing paper documents which were utilised during the project, e.g., a firm's collection of printed codes and standards.

The types of document that can be recorded in electronic form could be maintained in a plain relational database system, which would be managed by all those people concerned with a particular project. However, if a domain specific system is implemented, with knowledge about documents and their usage, a much higher level of functionality can be supplied to the users. This can be seen in the commercial arena by the plethora of EDMS which are available for managing documents, over a hundred of which are aimed at engineering documents. Knowing the nature of the objects that it is dealing with, an EDMS can automate many document management tasks, such as logging document creation and modification times or automatically generating version numbers for modified documents. If we assume that every transfer of information to an actor is accomplished through a set of documents, then the following tasks can be managed by an EDMS:

- ascertaining the relationship between a product and the documents in which it is referenced
- ascertaining the relationship between a document and the products referenced in it
- ascertaining the relationship between an activity and the documents utilised during its execution
- ascertaining the relationship between a document and the activities it is referenced, or used, in
- determining the reason for particular design decisions from related documents

- representing the history of document revisions
- tracking the documents which feed into an aggregated document
- representing the relationship between a document and the current state of the product model
- representing the working status of a document
- representing the legal status of a document
- representing the current actor(s) working on a document
- representing the modifications made to a document by an actor

ToCEE envisages that the developed document management system should offer the following services.

4.1 General Document Handling

Check-in

Place a document in the EDMS. This may either be a new document, a new version of an old document, or the collation of several documents to create a new document type. Dependent upon the parameters passed with the document, an automatic version number is assigned. The checked-in document is expected to be accompanied by several parameters which, dependent upon the type of document, define amongst other items the:

- originator or modifier
- time stamp
- document type
- working status
- distribution event
- setting of access rights
- information on signing of the document or signatures in the document

Retrieve

Returns the latest version of the requested document and sets the document's working status to that indicated in the retrieve command. Dependent upon parameters in the command, the document can also be accompanied with information on:

- legal status
- access rights
- version information
- last modification information
- last distribution event
- referenced documents

Locate

Performs the same function as *Retrieve*, except that it does not return the actual document, just the associated meta-data about the requested document and its actual location.

Search

Looks through the document database identifying a set of documents which meet the criteria specified in the search. *Search* can operate on two levels, either it can search through the meta-data about a document or, in future versions, search the actual text contained in all document types, or in named document sections. In a basic search the latest version of each document is searched, though all versions of all documents can be searched if desired. The return of a search is a set of document descriptors of the same form as returned by the *Locate* function.

Index of documents
Returns an index of the latest version of all documents in the named document set. The index comprises the meta-data about each of the documents as in the *Locate* function.

Access to related model
Returns the set of product references associated with the specified document. This is the full set of views of the product model, with versions where these are applicable. The product model data is not directly accessible from the EDMS. The product model data must be retrieved from the product data server, but the returned references are all that is required to identify and retrieve the specified product model view.

Specify links between product and document
Allows an association to be defined between a document and a view of the product model data, along with its associated version. This will be useful where the document is created independently of the product model, but is found to have a bearing on various aspects of the model.

Referenced documents
Returns the set of documents which are referenced by the specified document. The return of the referenced documents query is a set of document descriptors of the same form as returned by the *Locate* function. The inverse function is also supported to find all documents which reference the specified document.

Document set
Returns the latest version of the whole set of documents which comprise the named document set. The return of the document set query is a set of document descriptors of the same form as returned by the *Locate* function. Documents can be added to, or removed from, a document set. The whole document set can be frozen due to a certain process completing, or duplicated in its entirety for associated usage.

Denote documents as views of each other
For some work items it is useful to develop documents which are views of the same document, but for different use functions (e.g., an HPGL document and the original CAD file as a view of each other), participants, or audiences. In many cases they present the same ideas, results or orders, but are written in different terms for the different audiences. This function allows a document to be named as a view of an existing document in the same document set.

Find views
Retrieves the set of documents which have been named as views of the specified document. The return of the find view query is a set of document descriptors of the same form as returned by the *Locate* function.

4.2 Revisions

Retrieve version

The general *Retrieve* command returns the latest version of the named document. This command allows a specified previous version of a named document to be retrieved from the EDMS. As with the *Retrieve* command, the document can also be accompanied with information on:

- legal status
- access rights
- version information
- last modification information
- last distribution event
- referenced documents

Locate version
Performs the same function as *Retrieve version*, except that it does not return the actual document, just the associated meta-data about the requested document and its actual location. This command can always return a result immediately, whereas the *Retrieve version* command may need to access backup storage to access the specified document. In some cases this could take a significant amount of time, measured in days.

Version tree
Return the version history for the specified document. This tree-like structure shows all merges that went towards the current document as well as the usual chain of versions over time. The information for each document comprises the meta-data associated with documents, including:

- modification time
- modifier
- status
- legality codes of the various versions

Get modifications
By default this returns the set of modifications which denote the difference between the current version of the document and its most recent ancestor. Dependent upon parameters this command can return an aggregated set of modifications going back several versions, or if required the modifications between any two specified versions of a document.

4.3 Status

Document status
Returns the status of the named document. As well as indicating whether it is checked out or free, it provides information on the working status of the document and the server upon which it actually resides. If the document is checked out, the actor who has it checked out and any comments specified as to why the document is being worked upon is returned.

4.4 Validation

Authenticate
Checks whether the proffered document is identical to the named document in the EDMS. This allows an outside source to validate that what they have been passed has not been modified from what is known to be the authentic version of the document (i.e., what is in the EDMS).

Audit trail
Provides an audit trail for the named document. The information returned provides the version history, along with authorising agent for each of the significant stages the document migrates through. This has a very similar structure to the result of the *Version tree* command.

Set access
Allows the access rights to the current version of a document to be specified or modified. This allows the individual actors who have any rights to be specified, as well as the rights they have on the document.

Check access
Determine what rights the named actor has for the specified document. This command returns the set of access rights enjoyed by the named actor.

4.5 Signature

Retrieve
Returns the signature of the named document. Dependent upon the type of signature applied to the document, a different type of answer is returned. For documents which only have a physical signature then the location of the signature is returned, and if the document was scanned then a facsimile of the signature as well. If the document has a digital signature then that signature is returned.

Validate
Checks whether a document with a digital signature was signed by the named actor. This command has no meaning for documents with only a physical signature.

4.6 Process

Routing to actors
Determines which actors need to be notified of a modification to a document due to a named event. This command returns a list of actors to be notified of the event and the modified document. No action is carried out to actually notify the actors, this is assumed to be handled by a lower level support system.

Set of modified documents
This command identifies a set of documents which were modified or affected by the named event. The return of the modified documents query is a set of document descriptors of the same form as returned by the *Locate* function. This command assumes that activities undertaken in a single process can modify multiple documents.

Specify event
Allows an event to be defined and a set of actors associated with the particular event to be defined. This set of actors are the actors who will be notified of a modification to a document associated with the named event. This step must be carried out before change or event notification can be supported for documents which are created or modified.

5. CONCLUSION

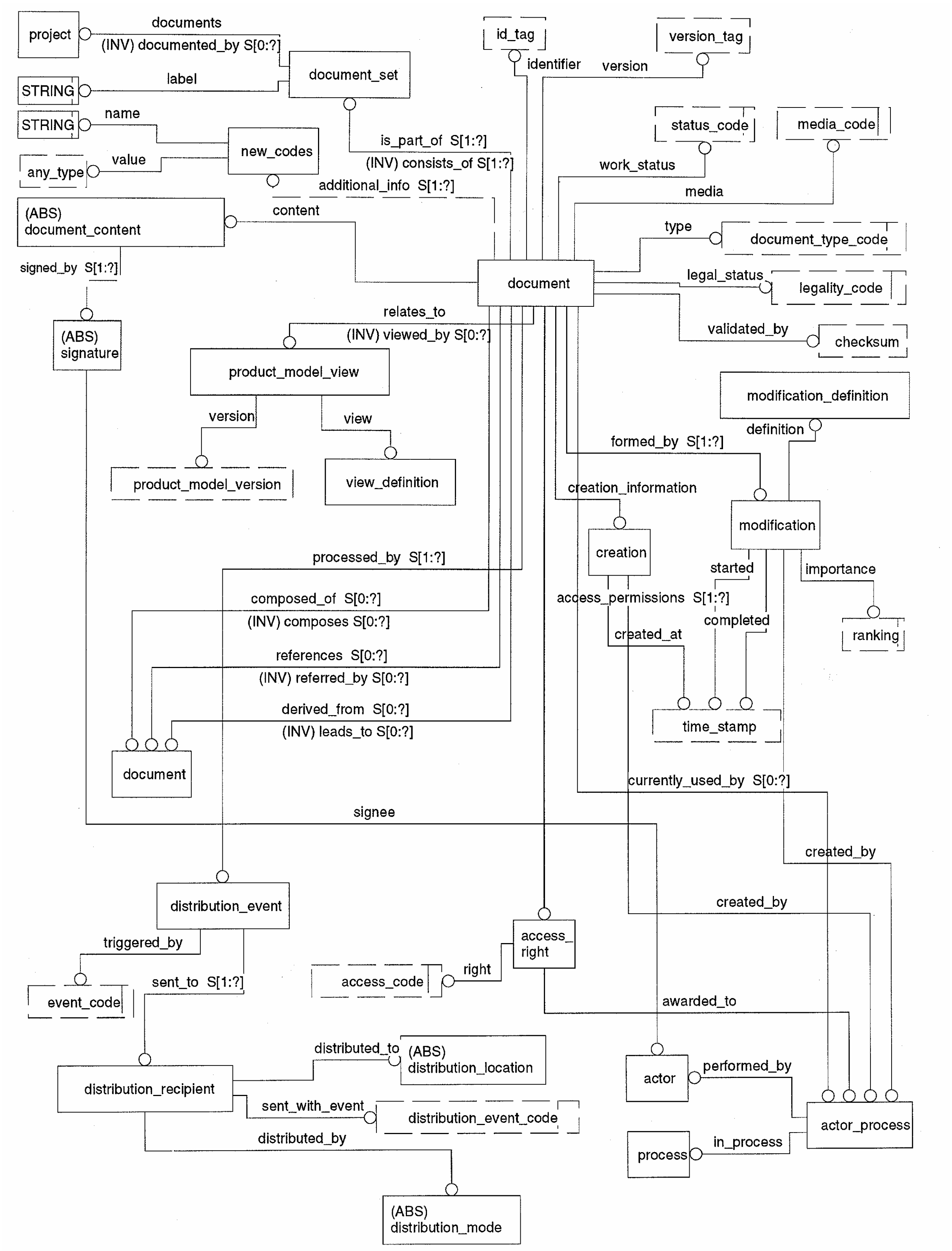
An electronic document management system will have to provide tangible benefits over the traditional paper system. Speed is an obvious advantage, saving resource costs and storage space, but the industry will need to overcome its habit of waiting for an ‘official’ paper version before committing itself to an action. This habit is partly to do with lack of trust, a barrier to concurrent or collaborative engineering, and it is vital that EDMS achieves legal/contractual recognition if it is to help towards concurrent engineering. An industry accepted standard for a document model, a key aim of ToCEE, would enable project teams, once created, to be able immediately to communicate securely and with confidence, no matter where members are located and work with the speed and co-ordination of co-located multi-disciplinary organisations. The document model, in conjunction with other ToCEE concepts, e.g., conflict management, will reduce undetected errors and the need for rework throughout the whole procurement process, which adds considerably to the cost of a project.

6. REFERENCES

1. ESPRIT IV-20587 ToCEE: EU ESPRIT IV Project 20587 ‘ToCEE - Project Programme’, EU/CEC, Directorate Generale III, Brussels, 1995.
2. ISO 10303-1: ‘Product Data Representation and Exchange - Part 1: Overview and Fundamental Principles’, ISO TC 184/SC4, Geneva, 1994, 17p.
3. Böhms M. and Storer G.: ‘ATLAS - Architecture, methodology and Tools for computer-integrated Large Scale engineering’, in Proc. JSPE-IFIP WG 5.3 Workshop, DIISM'93, Tokyo, 1994.
4. Watson A. and Crowley A.: ‘CIMsteel Integration Standards’, in R.J.Scherer (ed) ‘Product and Process Modelling in the Building Industry’, Proc. of ECPPM'94, Dresden, Balkema, 1995, pp 491-494.
5. Scherer R.J. and Sparacello H.-M. (eds): ‘COMBI Final Report’, EU/CEC ESPRIT III Project No. 6609, TU Dresden, 1996.
6. Augenbroe G. (ed): ‘COMBINE 2 Final Report’, EU/CEC Joule Programme, Project JOU2-CT92-0196, TU Delft, 1995.
7. CALS: ‘Computer Aided Acquisition and Logistic Support’, The Executive Guide, Department of Trade and Industry, London, UK, 1990.
8. Jablonski S.: ‘Workflow-Management-Systeme: Motivation, Modellierung, Architektur’ (in German), *Informatik Spektrum*, Vol. 18, No. 1, http://www.aiai.ed.ac.uk/WfMC/, 1995.
9. IIC Consulting and Cimtech Limited: ‘1996 Engineering Document Management and Product Data Management Guide, An Introduction to EDMS and PDM and the Suppliers of Products and Services’, ISBN 0-900458-71-2, 1996.
10. BTC: ‘Document Management Resource Guide’, A BTC - Document Manager Publication, 24 High Street, Beckenham, Kent BR3 1AY, UK, ISSN 1360-9238, 1996.
11. Turk Z.: ‘Construction Design Document Management Schema and Prototype’, *The International Journal of Construction Information Technology*, Vol. 2, No. 4, 1994, pp 63-80.
12. Björk B-C.: ‘Conceptual Models of Product, Project and Document Data; Essential Ingredients of CIC’, *ASCE First Congress on Computing in Civil Engineering*, Wasington D.C., USA, 20-22 June, 1994.
13. Mokhtar A. and Bedard C.: ‘Towards integrated construction technical documents - A new approach through product modelling’, *Proc. of ECPPM’94*, The First European Conference on Product and Process Modelling in the Building Industry, Dresden, Germany, 5-7 October, 1994, pp 3-10.

14. Armstrong S.G. and Lockley S.R.: Modelling of generic document structures and the development of an integrated document data environment', *Proc. of ECPPM'94*, The First European Conference on Product and Process Modelling in the Building Industry, Dresden, Germany, 5-7 October, 1994, pp 19-28.
15. Finch E.F., Flanagan R. and Marsh L.E.: 'Electronic document management in construction using auto-ID', *Automation in Construction*, No. 5, 1996, pp 313-321.

APPENDIX A. DOCUMENT MODEL SCHEMA



project
documents
(INV) documented_by S[0:?]
document_set
STRING
label
STRING
name
new_codes
any_type
value
is_part_of S[1:?]
(INV) consists_of S[1:?]
additional_info S[1:?]
(ABS)
document_content
content
signed_by S[1:?]
(ABS)
signature
id_tag
identifier
version_tag
version
status_code
media_code
work_status
media
type
document_type_code
document
legal_status
legality_code
validated_by
checksum
relates_to
(INV) viewed_by S[0:?]
product_model_view
version
view
product_model_version
view_definition
modification_definition
definition
formed_by S[1:?]
creation_information
creation
modification
started
importance
access_permissions S[1:?]
completed
created_at
ranking
time_stamp
processed_by S[1:?]
composed_of S[0:?]
(INV) composes S[0:?]
references S[0:?]
(INV) referred_by S[0:?]
derived_from S[0:?]
(INV) leads_to S[0:?]
document
currently_used_by S[0:?]
signee
created_by
created_by
distribution_event
access_
right
triggered_by
event_code
sent_to S[1:?]
access_code
right
awarded_to
distributed_to
(ABS)
distribution_location
actor
performed_by
distribution_recipient
sent_with_event
distribution_event_code
actor_process
distributed_by
in_process
process
(ABS)
distribution_mode