Architecting an Enterprise Wide Document Management Platform

Jacob Willig

EMC Proven Professional Knowledge Sharing 2009

Jacob Willig
Senior Documentum Consultant
documentum.consultant@lazymen.eu
# Table of Contents

Table of Contents .............................................................................................................. 2  

Introduction ....................................................................................................................... 4  
  Target Audience ......................................................................................................... 5  
  Reading Instructions .................................................................................................. 5  
  Note to reader ............................................................................................................ 5

1. Information Management ........................................................................................... 6  
   Challenge Information Storage .................................................................................. 6  
   Challenge Retention ................................................................................................. 6  
   Challenge Central Information Storage ...................................................................... 7  
   Challenge Metadata Management ............................................................................... 7  
   Solution Standardized Retention ............................................................................... 7  
   Solution Standardized Document Types ..................................................................... 8  
   Solution Standardized Information Storage Architecture ........................................... 8  
   Solution MetaData management ............................................................................... 9

2. Object Model ............................................................................................................ 10  
   Challenge Duplication .............................................................................................. 10  
   Challenge Object Hierarchy Architecture .................................................................. 10  
   Challenge Object Hierarchy Depth ........................................................................... 11  
   Solution Object Hierarchy Architecture and Depth .................................................... 11  
   Solution Attribute Naming Convention ..................................................................... 13  
   Solution Attribute Redundancy ................................................................................ 14

3. Security .................................................................................................................... 15  
   Challenge User Management .................................................................................... 15  
   Solution Complexity ................................................................................................. 16  
   Solution User Management ...................................................................................... 19
4. Workflows................................................................................................................20
   Challenge Loss of Overview ....................................................................................20
   Solution Loss of Overview .......................................................................................20
   Considerations Workflow Security ...........................................................................21

Support Organization ......................................................................................................22
   Challenge Logging ...................................................................................................22
   Challenge Release Management .............................................................................22
   Solution Logging ......................................................................................................22
   Solution Release Management ................................................................................23

5. Final Remarks..........................................................................................................24

Biography ......................................................................................................................24

Disclaimer: The views, processes, or methodologies published in this compilation are those of the authors. They do not necessarily reflect EMC Corporation’s views, processes, or methodologies.
Introduction

EMC Documentum® provides a Content Management suite with a rich set of features. Increased complexity of configuration and support are the consequences. This article discusses some of the best practices I’ve discovered while implementing Documentum within large organizations using a central Platform approach as opposed to a fragmented, Point Solution approach.

Documentum is typically implemented as a dedicated host for document management and storage. This type of implementation is tailored to the specific set of functional requirements and does not necessarily consider future expansion into different functional areas within the organization. If the need arises, a new Point Solution will most often be implemented. It will be tailored to the new requirements with fresh design and configuration approaches.

Although this approach fulfills short-term business needs, it will become increasingly more expensive to maintain and operate long-term (with increasing number of Point Solutions). This minimizes the effectiveness of a content management solution since overhead costs re-occur.

Large organizations should consider an Enterprise view of content management. This will involve implementing Document Management as a strategic Platform able to host all kinds of document flows within the organization consistently. Adding a new document flow can be a relatively easy and cost efficient, as everything in the platform is setup and includes proper procedures.

This approach will have an additional startup cost to enable future growth in controlled manner. Several challenges that can be left unresolved in a Point Solution need to be addressed beforehand in a Platform approach. This article discusses several of the challenges faced while implementing Document Management in a platform architecture. It will explain the challenges and offer solutions. It is based on experiences gained in the last few years while assisting several customers implementing Documentum in a platform architecture.
The following topics will be discussed:

- Information Management
- Object Model
- Security
- Workflows
- Functional, Application and Technical Support

**Target Audience**

This paper is written for Documentum Experts who are technically responsible for or architecting Documentum implementation projects. The article discusses technical solutions that require knowledge of Documentum’s core Object Model and functionalities. Especially those who have several years of experience in this field will acknowledge many of these challenges and hopefully be inspired by the solutions.

**Reading Instructions**

Several challenges are discussed and explained, followed by several solutions.

**Note to reader**

*All solutions discussed in this paper are result of my personal attempts to find solutions for challenges experienced during my activities in the last few years. The value of several of these solutions is considered high by several experts in the Document Management field and some of them argued not to publish them at all. Several planned solutions have been left out, or only briefly discussed, for this reason.*

*The ideas that are discussed in this article are shared to inspire fellow Documentum Experts to join me (and others) in pursuing a more generic approach to implementing Documentum and to stop reinventing the wheel in Point Solution implementations.*
1. Information Management

Information Management is an often undervalued challenge when implementing a Document Management Platform solution. Decentralized paper based information management and storage is very different from centralized digital information management and storage. This sounds logical, but often goes unrecognized.

Challenge Information Storage

Storing information is complex; many decisions need to be made. Local storage of specific information is relatively easily to design, but designing to store all relevant information is more challenging. The possibilities of digital storage change our thinking related to physical storage becoming obsolete or insufficient. Digital systems typically cannot compete with human creativity when organizational procedures do not properly work for specific situations. The question is whether we want this flexibility and creative workarounds. These workarounds make matters more complex, are not properly measured, and often are more costly in the long run. There is a balance that should provide enough flexibility in the digital system while preventing too much human creativity.

Here is a summary of the most important decisions:

- What information should be stored
- How should it be stored and maintained
- Who should have access to it, how and when
- How do we enable sharing of information
- How do we properly classify information
- How do we enable future adaptation of new types of documents

Challenge Retention

Documents often have a legal status and might be required as evidence in future lawsuits, customer disputes or other forms of investigation. In the United States, this has already been a hot topic for several years now. In Europe, these practices are gaining momentum.
To properly apply Retention Management on documents entering the enterprise, it is important to secure the content early to prevent tampering and accidental loss. Content under Retention is more expensive to manage, hence it will be profitable to ensure only relevant documents are put under a Retention Policy.

**Challenge Central Information Storage**

An enterprise wide Document Management solution offers potential cost savings as documents can be stored centrally, only once, and are instantly accessible. But how can we ensure that only one copy of a document is stored? How can we ensure that all relevant persons are able to access it and no-one else? How can we deal with existing paper copies? Evolving from current paper based Information Storage to new centrally Digital Information Storage will be a challenge.

**Challenge Metadata Management**

When storing documents, it is important to be able to retrieve them in several different ways. The most obvious solution is to store all relevant metadata with the document so it can be searched. This is an obvious and simple solution, however it also leads to bad design and metadata quality issues. Storing volatile customer data as document metadata is one of the common pitfalls. This works perfectly as long as this data is unchanged, but problems start as soon as the customer changes name or moves to a new postal code. All of a sudden documents cannot be found, or you need to make updates to all related documents to process updated customer data. Support teams typically make these updates; unfortunately, human error can sometimes occur.

**Solution Standardized Retention**

Here are some basic rules when implementing Retention Management:

- Filter out all informal documents as early as possible
- Or prevent informal documents as much as possible → no informal email communication for specific email addresses
- Limit the number of Retention Policies and aim for enterprise wide Retention Management guidelines and policies
- Double check applied retentions properly to ensure documents really get retention applied. Too often, design flaws are undetected and documents fail to have retention applied without anyone noticing.
Solution Standardized Document Types

It is important to properly classify documents to predefined Document Types in order to store documents centrally and ensure single copies. Properly classify any document entering the system. With proper classification it is easy to:

- Validate the document’s required metadata
- properly store the document
- apply the correct Retention Policy
- discover and handle duplication
- easily retrieve the document when needed

It is important to distinguish between the true Document Type and its use by all related processes. For instance, a Passport can be an Identification Document. It can have multiple uses, like Customer identification, Business representative identification (the customer can, be a retail customer and also a Financial Director at a corporate client).

The uses of a Document Type should be kept separate and could / should be managed using dm_relation objects. Relation objects can specify any type of usage without restructuring the folder structure, using linking, or even duplicating content. As linking in folder structures forces a 2 dimensional view on content, relations can add the other dimensions easily. It does require user interface customizations to properly utilize these relations, but if used, they will help prevent the creation of a 3d-wannabe folder structure, which is bound to self destruct at some point in the future.

Solution Standardized Information Storage Architecture

It can be very beneficial to use a folder structure for Information Storage:

- Folder based Retention Management
- Browsing a folder structure is more intuitive
- Documentum is not yet tuned to property based browsing
- Most out of the box modules in WDK assume a folder structure is used
However Folder Structures are dangerous:

- They often introduce redundancy and ambiguity if metadata is not properly maintained.
- If folders need to be retrievable, they require metadata to be stored, which requires management.

It is important to choose the right classification structure when storing documents in a folder structure; one that does not mix up the Documents’ base value with their potential uses.

For example, it is better to store Identification Documents (like passports) centrally in a Person based structure. An Identification Document identifies a Person, and this Person can have multiple roles within the enterprise that will lead to problems when it is stored any other way.

**Solution MetaData management**

It is important to distinguish between true content related metadata and related metadata to properly manage it in Documentum. All metadata related to content is static and represents the context of the specific piece of content at the moment of creation. All other metadata is potential dangerous and should be avoided. See discussion on Object Model Attribute Naming Convention for solutions on how to deal with these issues.
2. **Object Model**

Creating an object model for a dedicated Point Solution is relatively easy. All object types can have specific names and do not have to prepare for future adaptation of new document flows.

Designing a reusable object model for a Central Platform, hosting many Document Management solutions, however, will pose some challenges:

- Generic Object Hierarchy
- Standard Object Naming Convention
- Generic Attribute Usage
- Standard Attribute Naming Convention
- Prevent Attribute Redundancy

**Challenge Duplication**

In the physical world, information is typically stored in paper format with cabinets used for indexing and security. If a customer holds more than one product with the organization, it is possible for the same information to be duplicated across each product the customer holds. If the organization is global, it is also possible for the information to be duplicated across different geographies, or because it was easier to do so. Access to information stored this way has its limits and many are convinced digital storage would bring many advantages. However, it might sound simpler then it really is.

**Challenge Object Hierarchy Architecture**

Another challenge is the approach to take when defining object types. When designing an object model, it is important to use one approach. As with many IT designs, the object model allows a tree like inheritance. This demands a 2d view and does not allow inheritances from multiple parents. The introduction of Aspects should take away the need for one parent inheritance and does allow individual inheritance where required. However, this article is based on experiences gained while working with Documentum 4.2, 5.3SP5 and 6.0SP1 and does not take into account the options enabled by the introduction of Aspects. Even with the use of Aspects, it is important to properly architect the Object Hierarchy or Aspect usage.
Challenge Object Hierarchy Depth

There is a technical limit to the depth of the object hierarchy. Every object type is represented by 2 tables in the database. One table is for the repeating attribute, another is for the single attributes. Determine the size and complexity of the backend SQL query when executing a DQL query. It is best to minimize the depth of object type inheritance to 4 levels maximum.

For instance: dm_sysobject, dm_document, base_document, client_document

The challenge is to find the balance between defining the right number of inheritance levels for optimum use of attributes against the drive to minimize the depth for performance reasons.

Solution Object Hierarchy Architecture and Depth

Documentation can be organized into three main category types:

- Entity related
- Product/Service related
- Process related

Note: This approach will automatically reduce the required depth and minimize it to a comfortable three levels deep.

A document containing a client acceptance form would be related to a customer (entity) as all documents directly related to the customer and representing information about the customer. The same goes for employee related files, where the employee is the entity. Typically, a customer is related to several products/services he has purchased, like insurance or a bank account. These products/services have documentation related to them, like contracts or correspondence.

Finally, there are process-related documents. For instance, damage or health insurance will have claim forms related to it, or an internal audit process that reviews a client. Its result will be a client related document, its work documents and investigation documents are process related.
It is very likely that several object types will have common attributes that would be better off on a base object type as the parent for those specific object types.

<table>
<thead>
<tr>
<th>Context group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OOTB types</td>
<td>The default OutOfTheBox object types provided in the Documentum suite. Most used are: dm_document for Documents and dm_folder for Folders. Configuration specific object types typically will be extending dm_sysobject directly</td>
</tr>
<tr>
<td>Basis types</td>
<td>Basis object types containing all common generic attributes. These should not contain any dedicated attributes that cannot be reused by any lower level type</td>
</tr>
<tr>
<td>Entity types</td>
<td>Content about an entity. e.g.: passport, signature card, abstract Chamber of Commerce. Example of possible entities: Customer / Personnel / Suppliers</td>
</tr>
<tr>
<td>Product(group) types</td>
<td>Content about a certain product or product group. e.g.: Service contract, Contract Terms, Product related correspondence</td>
</tr>
<tr>
<td>Process types</td>
<td>Content about a certain process or function. e.g.: Audit Report, Damage Claim form, Expense sheet, Performance Review Report</td>
</tr>
</tbody>
</table>
Solution Attribute Naming Convention

It all starts with a generic and intuitive naming convention.

Even with proper names for attributes, it is possible to misinterpret its usage and value related to time. For instance, an attribute’s value can be related to a point in history (customer address at creation of document) or it could be an up-to-date value that is related to the present at all times (e.g., current customer phone number).

It is wise to introduce some intuitive prefixes that are mandatory for every custom attribute to minimize misinterpretation.

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc</td>
<td>Document specific property. The value is historic and should be immutable.</td>
</tr>
<tr>
<td>fdr</td>
<td>Folder specific property. The value is historic and should be immutable.</td>
</tr>
<tr>
<td>sys</td>
<td>System property. The system manages this value and could be made read-only available to end users; if not, it should be hidden.</td>
</tr>
<tr>
<td>ref</td>
<td>(External) Reference property. The value references a (external) key value and should be immutable. (see also ‘sync’)</td>
</tr>
<tr>
<td>sync</td>
<td>Synchronized property. The value is maintained by regular synchronization with the (external) source system. It depends on one or more ‘ref’ properties to update this value during synchronisation. Note: A synchronization process should be created and activated to actually manage these values.</td>
</tr>
<tr>
<td>proc</td>
<td>Object specific process related property. The value is Managed by (a) process(es) related to this object. It is better to use Process Variables in most cases. The most useful practice of this usage is to maintain shared (object specific) values between multiple processes on the same object. Thread safety should be acknowledged.</td>
</tr>
</tbody>
</table>
Solution Attribute Redundancy

Often, attributes are created directly from the Functional Design document without much thought. Even in a Point Solution design, this can and likely will lead to future problems.

For example, registering the source of the document could lead to three different attributes if they are too directly derived to the physical document. An email will have a SMTPFrom attribute, a fax will have a SenderNumber, and a CustomerLetter will have a SenderAddress.

It might be obvious when presented like this, but one single attribute called doc_senderaddress would fit all cases and is generic enough to apply to other document media potentially processed in the future.

Applying this logic will reduce the number of redundant attributes to a bare minimum.
3. Security

Security is a challenge when converting from paper based into digital context. Where filing cabinets and buildings pose a physical security barrier, these do not apply to the digital storage on servers. Who should have access to what data, how do we enable the sharing of data, or should we? How do we maintain users and their roles?

There are some technical challenges as well. Too many levels of nesting will reduce the performance of the system. Groups belonging to many groups will cause performance degradation.

Challenge User Management

User Management is a specialist task that should not be the responsibility of any application that is not dedicated to it. Many applications provide a form of user management, which should only be used for small dedicated solutions, and only when no global user management solution is available. An LDAP based solution is the most commonly used. Managing security centrally for most or all applications limits the number of people responsible for assigning access rights. Although these individuals become more powerful, they also are more easily controlled and monitored.

Documentum also has user management capabilities, but these have many of the same restrictions as other application in this field:

- It is not possible to give someone access to manage users without preventing that person to gain full control over the contents in the Documentum Repository
- It requires security personnel to be trained in the use of the Documentum product, or Documentum trained personnel that need to adhere to security rules. Either option is expensive and will lead to errors.

Fortunately, it is possible to avoid internal user management. If the organization already has an LDAP based security service, it is possible to perform all user management from within this service. Creating the right groups and populating them is sufficient for all user management that Documentum requires.
Solution Complexity

Security is complex so we will keep it as straightforward as possible.

To improve an overview of security, it is wise to introduce categories of groups that have a specific security function:

- **Content Roles** → Controls access to content objects
- **Function Roles** → Controls access to GUI functions and may not be considered security, just cosmetics
- **Business Roles** → Represents a business role relevant to the system and populates both Content Roles and Function Roles
- **LDAP Roles** → Required to loosely couple the LDAP synchronization from the internal security setup, every LDAP Role will have an equivalent Business Role

This diagram shows the relationships between these Role categories:

![Diagram showing the relationships between End User, LDAP Role, Business Role, Content Role, Function Role, ACL, User Interface, Work Queue, and LDAP Synchronization]

Content Roles are defined for a specific set of content (folders and documents) that share the same security regime. The set shares something in common, such as:

- Client Identification documents
- Insurance Policy Claim documents
- Personnel files
The name of the Content Role should describe the function of the content set and should not reference the users dealing with the content!

Content Roles loosely couple access to the content from the users (Business Roles) handling the content. Regardless of who deals with the content, the names of the Content Roles will never need to change.

In order to prepare for any likely type of usage, Content Roles are created in sets of related roles, each representing a specific type of usage throughout the lifecycle of the content. This sounds complex, but is surprisingly simple!

Note: For this to work, it is important to also create generic Lifecycles with predefined Lifecycle States, clear names, and transparent descriptions.

Consider the following generic Lifecycle: Draft → Review → Active → Archived

If you think about it, there are not that many types of content usage that can be defined in relation to the content lifecycle:

- Consumer → requires Read access in the Active (or Published) state.
  Many Business Roles can populate one single Consumer Content Role. All Business Roles requiring access to Passport documents would populate the Consumer Content Role of Identification Documents in their Active State.
- Reviewer → Must like consumer, but Read (maybe Write) access in earlier stage (Review) as well
- Indexer → Write access in earliest stages of lifecycle
- Specialist → special access, like Write access while content is in Active state

<table>
<thead>
<tr>
<th>Role</th>
<th>Draft</th>
<th>Review</th>
<th>Active</th>
<th>Archived</th>
</tr>
</thead>
<tbody>
<tr>
<td>abc_indexer_crol</td>
<td>Write</td>
<td>Read</td>
<td>Read</td>
<td>Read/None</td>
</tr>
<tr>
<td>abc_reviewer_crol</td>
<td></td>
<td>Read/(Write)</td>
<td>Read</td>
<td>Read/None</td>
</tr>
<tr>
<td>abc_consumer_crol</td>
<td></td>
<td></td>
<td>Read</td>
<td>Read/None</td>
</tr>
<tr>
<td>abc_specialist_crol</td>
<td></td>
<td></td>
<td>Write</td>
<td>Read/None</td>
</tr>
<tr>
<td>abc_support_crol</td>
<td>Delete</td>
<td>Delete</td>
<td>Delete</td>
<td>Delete</td>
</tr>
</tbody>
</table>
It is important to limit the terminology used for Content Roles to a limited set that is intuitive and transparent:

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indexer</td>
<td>Complete Metadata on new documents (Write access at Draft state)</td>
</tr>
<tr>
<td>Reviewer</td>
<td>Perform review on document (default Read access at InReview state)</td>
</tr>
<tr>
<td>Consumer</td>
<td>User of document (Read acces at Active status)</td>
</tr>
<tr>
<td>Specialist</td>
<td>Specialist with additional access, like Write access in Active state</td>
</tr>
<tr>
<td>Support</td>
<td>Support activities, like corrections at any time in the lifecycle of the document.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Additional terms can always be added, however they should add a unique set of access rights throughout the lifecycle.

Function Roles are defined for a set of functions within the user interface. They do not provide any security and should never be used as such. Function Roles can be used to limit or tune the functions in the user interface to the ones required by a Business Role.

Taskspace and Work queues can be controlled by the use of Function Roles.
Solution User Management

The above sketched security design and helps us to loosely decouple user management. If the right security roles have been created during the design phase, all is in place to couple the Business Roles with End Users. There are two ways of coupling the groups to the external LDAP service:

- Direct synchronization of the Business Roles with the LDAP Service
- Indirect synchronization, using LDAP Roles placing each member in their own related Business Role. After initial synchronization, all LDAP Roles will exist in the Repository and a simple DQL script can finalize the correct coupling between LDAP Roles and their Business Role.

Direct synchronization gives best performance as one less level of group nesting.

Indirect synchronization gives more flexibility at the cost of one extra level of group nesting. The following is possible with indirect synchronization:

- Business Roles can be part of the deployment package and are not influenced by external conditions
- Multiple LDAP Servers can be used, each having there own set of LDAP Roles, resulting in several LDAP Roles per Business Role
- Independence of LDAP naming conventions that might differ between deployment environments
4. Workflows

The use of Workflows is a challenge. A whole new way of thinking is required when mapping the old physical ways of working into the digital world. This is a challenge in a Point Solution, but an even bigger challenge when aiming for a generic approach to implementing workflows.

Challenge Loss of Overview

Working with multiple workflow definitions in Documentum can quickly lead to loss of overview due to the volume of activity and the number of workflow names. The default naming mechanism does not help in later support challenges. I have seen different developers working (separately) on the same Repository, all implementing their own naming conventions, or using ad hoc names. As Documentum does not enforce unique names for Process Definitions, and only limits uniqueness of Activity names to its own Process Definition, a loss of overview is imminent.

Solution Loss of Overview

The solution is simple yet extremely powerful. You need a proper naming convention that is generic enough to be used by anyone designing processes in Documentum.

The basics:

- Unique names for all Process Definitions
- Unique names for all Activities (at least repository wide)
- Numbered prefixes for Process Definitions
- Numbered prefixes for Activities prefixed by the related Process Definition number
- Never reuse a number; create a new number:
  - activities that change function should get a new number
  - keep the number of activities that get bug fixed
- Do not use the numbers to stipulate order in the workflow.
- The numbers bring Repository-wide uniqueness, but do not express anything, so use practical and to the point names for both processes and activities
Considerations Workflow Security

Workflows can be started by client applications delivering documents to the system. It is important to ensure this application account does not become the Workflow Supervisor performing all automatic tasks.

This sounds logical, but application accounts can end up being Workflow Supervisors simply because they start the workflows.

Set a custom ACL on the dm_process object to ensure that only specific users are able to start them. where workflows are started manually, this would reduce the selection of workflows to the ones the user has access to. In a Point Solution, this might not make sense. However, in an Enterprise Wide solution with many different applications all with their own workflows, it starts to make sense quickly.
Support Organization

Every Documentum implementation requires skilled support resources. Supporting one Point Solution will require training in the specifics of this particular implementation. Supporting many Point Solutions will quickly make things more complex. Supporting a generic Platform can be as easy as supporting a single Point Solution if done correctly. This does require that, while solving the earlier mentioned challenges, future support is considered in every part of the design.

Challenge Logging

Documentum disperses their essential log files all over the system. This makes it difficult to find any useful information. It is also difficult to direct new support staff where to look. Also, the log settings differ for every application. Several changes need to be made to the OOTB settings to make them more useful.

Challenge Release Management

Release Management is a challenge which is not Documentum specific. However, it has been a problem on most projects I have worked on. Every organization aims for proper Release Management where DTAP (Development, Test, Acceptance, Production) environments, are preferably Production-like. These environments should be under strict Release Management procedures to eliminate issues caused by environment differences. Unfortunately, most of the time no-one can ensure the environments are totally alike or rule out environment influences.

Solution Logging

Use SMTPAppender for Error emails. Do not let any one error get by unnoticed!

Expect many emails in special situations; make sure you inbox is large enough.

Use of logging standards:

- central log location
- consistent log format settings
- proper rolling schemes
- proper cleanup/backup procedures
Solution Release Management

If possible make all deployment environments Production Like.

- Same OS version
- Same Documentum version
- Complete chains of services

Eliminate all possibility of unregistered changes.

Only allowed differences:

- Resources, number of CPU’s, amount of RAM, disk space
- Environment specific settings (reduce to minimum)

Do not deploy anything that is not a registered package.

ALWAYS create an Installation Report and use it to notify the groups involved.

Keep full records of all changes using a Release History.

As I cannot stress this point often enough I will repeat it once more:

ALWAYS create an Installation Report and use it to notify the groups involved
Keep full records of all changes using a Release History.

Never change an already deployed package, create a new minor release.
5. Final Remarks

All solutions discussed in this article are my current best practices and will likely be improved and further refined in the future. They are inspired by a broader vision to create easier and better ways of doing things in the new and mostly unexplored field of Document Management.

I am convinced we are just touching the surface of what is possible with Document Management and how to properly implement it as an Enterprise platform.

I encourage everyone inspired by this article to consider using more generic ways of implementing Documentum, to share ideas and to let me know where my Best Practices might contain flaws. All remarks will help to improve existing best practices and bring us closer to a truly Enterprise wide Document Management platform.

Biography

Jacob has been working with the Documentum suite daily since 2001. He has been involved in, and responsible for, many implementations for customers in the Banking and Insurance sector.

He understands the technical backend of many Documentum functions (to solve problems and reverse engineer and decompile code). Jacob has developed a thorough understanding of the inner workings of the product and has also been involved in more architectural responsibilities.

During his career, Jacob started building many Point Solutions trying to discover more generic ways of working. Two years ago, he had an opportunity to help implement Documentum on an Enterprise wide scale in a platform architecture. This project gave him an opportunity to start dealing with the challenges associated with designing a Document Management Platform.

He holds an EMC Content Management Foundations Associate level certification.