

# Graduation Final Report

---

*Enterprise Content Management System*



Author: Denny Damara  
Student Number: 2185000  
Date: 10 June 2015  
Version number : 6



**GRADUATION REPORT**  
**FONTYS UNIVERSITY OF APPLIED SCIENCES**  
**HBO-ICT: English Stream**

<b>Data student:</b>	
Family name , initials:	<b>Damara, DD</b>
Student number:	<b>2185000</b>
project period: (from – till)	<b>02-02-2015 until 30-06-2015</b>
<b>Data company:</b>	
Name company/institution:	<b>Orcion B.V.</b>
Department:	
Address:	<b>Industrieweg 23c, 5066 XJ, Moergestel</b>
<b>Company tutor:</b>	
Family name, initials:	<b>Deijkers, RD</b>
Position:	<b>Technical Director</b>
University tutor:	
Family name , initials:	<b>Lahaije, PL</b>
<b>Final report:</b>	
Title:	<b>Graduation Final Report-Enterprise Content Management System</b>
Date:	<b>10-6-2015</b>

Approved and signed by the company tutor:

Date: 15-6-2015

Signature:



Approved and signed by the university tutor:

Date:

Signature:

<p><b>Lahaije, Paul P.D.M.E.</b></p>	<p>Digitaal ondertekend door  Lahaije, Paul P.D.M.E.  DN: c=NL, o=Stichting Fontys,  cn=Lahaije, Paul P.D.M.E.,  1.2.840.113549.1.9.2=p.lahaije@fontys.nl  Datum: 2015.06.14 13:30:25  +02'00'</p>
--------------------------------------	--



# Preface

This report was written to serve as my final report of my graduation project for my final year of studying at the Fontys University of Applied Science. This report is a proof that I have completed my graduation project successfully during the last 5 months.

My graduation internship took place at Orcion B.V., an IT company located in Moergestel, The Netherlands. My project was about a Document Management System (DMS), which allows the users to map an existing folder structure and the files inside, after which they can be managed through the application. The most important functionality of this application is the ability to edit documents online, which is also required to be integrated into the company's other web applications. This project will be the starting point of a bigger project to create an Enterprise Content Management System (ECMS). While there are some ECMS already established in the world, Orcion wanted to make one of their own, under their ownership that they can modify to suit their customers' need. They hope that one day, this ECMS that they are going to develop can compete with other ECMS like SharePoint or Sitecore.

Working on this project was quite an interesting and valuable experience for me. I had the chance to apply all the knowledge that I had been taught for the last three and a half years, including what I had learned during my last internship last year. I got to learn more new knowledge and programming techniques and write better documentation for the project. All of these will be very useful to prepare me for working world after graduation. Of course, nothing is ever without challenge, travelling to another town every day was quite a taxing activity for my body. I had to re-arrange all the activities so I could work on my project properly and still keep myself healthy. Nevertheless, I had good time working on my project.

I would like to take this opportunity of thank several parties. First of all, to God the Almighty, for always being with me throughout this entire time and for every one of His guidance He has given me. To Orcion B.V., for giving me the chance to have my internship there. To Mr. Ing. Ron Deijkers, my company mentor, for all the help and knowledge given during my time at the company. Mr. Paul Lahaije, my Fontys tutor, for helping me write a better documentation of the project. To my family, for every support they have given me up to the completion of this project. Last but most certainly not the least, to each and everyone else I couldn't mention here, for all the help and prayer for me to complete this project.

Eindhoven, 8-6-2015

Denny Damara



## Table of Contents

Preface.....	3
Summary .....	7
Glossary.....	9
1. Introduction .....	11
2. The Company .....	13
3. Assignment Overview .....	15
4. Research and Analysis Phase .....	17
4.1. Requirement planning .....	17
4.2. Technology Analysis.....	18
4.2.1. WebDAV.....	18
4.2.2. HTTP Request Handling .....	20
5. Design Phase .....	23
6. Implementation Phase .....	27
6.1. First Sprint.....	27
6.2. Second Sprint .....	28
6.3. Third Sprint .....	28
6.4. Fourth Sprint .....	29
6.5. Fifth and Sixth Sprint .....	30
7. Results .....	33
Conclusions and Recommendations.....	35
Evaluation.....	37
References .....	39
Appendix A: List of Implemented DMS Functionalities .....	41
Appendix B: Project Plan.....	42



## Summary

Orcion is an IT company whose current main focus is building custom software for pharmacovigilance authorities worldwide. In their biggest project at the moment, which is done for Lareb, a Pharmacovigilance Centre located in 's-Hertogenbosch, The Netherlands, users can upload documents through the web application to be managed afterwards. However, it was not possible to edit those documents without downloading them first. Thus, Orcion started this ECMS project to fix the problem. This project built a Document Management System (DMS) which is intended to be both a standalone application and to be integrated into their web applications which have been, are being, and will be developed.

Chosen for its versatility, Scrum was used in this project for the development, in which the company mentor acted as the customer during the sprint kick-off/evaluation meeting. There were six sprints in total, each of which lasted for two weeks.

The result of this project, the DMS, will later on be developed into a fully working, commercial ECMS under Orcion's ownership. Orcion hopes that this ECMS will be able to compete with other existing CMS like SharePoint or Sitecore CMS. This DMS was built on top of an existing file system, which can be managed through the application. Using it, users can modify their documents online, without the need of downloading them, which was the problem Orcion needed to solve. This online editing functionality will be integrated into their web applications in the future. Another benefit of this DMS is that it will allow users to structure the folders in their preferred way (each users may have different preference from each other) without actually modifying the underlying structure itself. There is still room for improvements in the project, but since the source code is under the company's ownership, they can extend it easily anytime it is needed.

The problem that Orcion had in the beginning has been solved. They now have the DMS to start building the ECMS, whose document editing functionality can be integrated into other applications they have. There were still some limitations of the resulting application that can be further improved such as the implementation of the versioning system and the speed and effectivity of the search engine. However, this DMS is ready to be the building block of the ECMS.



## Glossary

1. ECMS – Enterprise Content Management System
  - a. A formalized system for organizing, managing or publishing contents (documents, web pages, images, etc.) that relates to the organization's processes.
2. CMS – Content Management System
  - a. A system/computer application for organizing, managing for publishing content.
3. DMS – Document Management System
  - a. Part of ECMS/CMS pertaining to document and file management
4. API – Application Programming Interface
  - a. A set of routines, protocols, or tools for building software applications. It expresses software component in terms of input, output, operations, etc.
5. IIS – Internet Information Services
  - a. A web server developed by Microsoft
6. WebDAV – Web Distributed Authoring and Versioning
  - a. Internet based standards for editing over HTTP or HTTPS
7. UI – User Interface
  - a. The space where human-machine interactions occur.
8. MS SQL Database – Microsoft Structured Query Language Database
  - a. A database management system developed by Microsoft.
9. HTTP – Hypertext Transfer Protocol
  - a. An application protocol for distributed, collaborative, hypermedia information systems. Used for data communications in World Wide Web.
10. ASP .NET – Active Server Pages .NET
  - a. An open source, server side web-application framework for building dynamic web pages, developed by Microsoft.
11. MVC – Model View Controller
  - a. A software architectural pattern for implementing user interface. It divides the software into three interconnected parts to separate the presentation of information to the user from the internal representation of those information.
12. Pharmacovigilance
  - a. Pharmacovigilance is the pharmacological science relating to collection, detection, assessment, monitoring and prevention of adverse effects with pharmaceutical products. It focuses on adverse drug effects (ADR) which are defined as any response to a drug which is noxious and unintended.
13. WOT – Web of Trust
  - a. An online website reputation rating tool
14. BLOB – Binary Large Object
  - a. A collection binary data stored as single entity in database management system.



## 1. Introduction

Orcion B.V. is an IT company that helps other organizations to work more efficiently and effectively. They support their clients by developing and maintaining professional websites and custom web applications.

In one of the web applications they are developing, there is a need for managing documents, thus, requiring the functionalities of a Document Management System (DMS) to be integrated into this web application. This may also be true for any web applications they are going to develop in the future. The company wants to have this DMS under their ownership and later on sell it with their brand, instead of using an existing DMS.

They aim to develop a new advanced product that later on, will be able to compete with internationally known products in the industry, in this case ECMS (Enterprise Content Management System), such as Microsoft SharePoint or Sitecore CMS. The final product of this project will be the base of many web applications they will develop, as the online editing functionality is designed to work both in the standalone application developed in this project or to be integrated into other applications.

This project aims to fulfill that goal through the development of a DMS. This DMS is a web application that allows the users to map their existing folder structure and manage them all through one application. Once it is completed, this application is going to be developed into a fully working ECMS with more than just a document management system.

There are notable benefits from this project. The company will have the source code of the application, allowing them to modify it to suit their customer's needs. While there are some renowned ECMS which have DMS functionality already in existence, they may have some functionalities that the company and their customers do not need. This DMS will allow them to customize it to suit the need of their customers. Another advantage is that it provides the customers with an ability to better organize documents of an existing file system and allows them to try it out without disorganizing their original folder structure. Last but not least, the online editing functionality of this DMS can be integrated into existing applications that the company may have regardless of where the files are stored, which may differ from application to application.

This project was done in an agile manner, using Scrum. A Scrum meeting was held every 2 weeks (sprint duration) for progress meeting and discussion regarding bottleneck if any or sprint planning.

This report contains several chapters. Chapter 1 is the introduction about the assignment and report. Chapter 2 explores the history of the company from the moment it began up to this day. Chapter 3 gives a description about the project, why it was initiated, what problem it will solve, and what the final result is. Chapter 4 up to chapter 7 describes in detail the process (research, requirement gathering, design, implementation and result) of the project itself. At the end there will be a conclusion of this report and recommendations for improving the result of this project further. Finally, there will be an evaluation of the writer about the experience he gained during his time working on the graduation project.



## 2. The Company

Orcion B.V. <sup>[36]</sup> was started by Ing. Ron Deijkers and Sander Slingerland MSc. The company officially started in July 2013, in Asten. However, at that time they did not have an actual office yet. Therefore, the address registered for the company was Mr. Slingerland's home address, to which all letters and communication went. Finally, in January 2015, they got their own office in Industrieweg 23c, Moergestel. There are currently 4 people working for the company.

The name Orcion came up after the two founders bounced back and forth several names from the list they gathered. The list itself contained the possible names that are still available for their website with .nl and .com domain. While the name itself may not have a special meaning, their logo, which was designed based on pyramid symbolizes their mission and way of working. The pyramid was thought to be a structure that was almost impossible to be built, yet the ancient Egyptian managed to build them. So, Orcion's logo was conceived under the thought that they will build seemingly impossible solutions for their customers. Now it means building a solid solution that their customers need. The pyramid has an open door, which symbolizes their way of working, honestly, clearly, and transparent to their customer. Orcion is currently changing the logo to a pyramid with three sections, each floating above the other, which still symbolizes the very same ideal of honesty, clarity and transparency.

The idea to build Orcion came from the founder's belief that out there in the world, there are many ICT companies who deliver mediocre quality results, yet they charge their customers with high prices. This can happen because of the lack of communication and because they don't always think with and for their customers. Therefore, Orcion was built to change that. Orcion believes that with effective communication, custom built software can be more affordable for the customers, and can match the customer's requirements better.

Their current focus is building custom software for pharmacovigilance authorities world-wide, by providing them the products and services that can be adopted to their needs.

There are three main selling points of Orcion compared to other companies of similar/same line of work. Firstly, they are thinking with their customers. They have a strong focus on consultancy to ensure that the company will build and eventually have software that their customers need rather than the software that they thought they need. Secondly, clear communication. They always try to be as clear as possible to prevent any surprises afterwards by explaining all functionality and, if desired, technology as well in clear layman's terms, as not everyone have the advanced knowledge of technical terms. Lastly, honesty and integrity. All their communications are done in such a way that it is understandable for everyone. There is no vague communication or double meaning in everything they said or wrote, and what is invoiced is based on the actual hours they work on.



### 3. Assignment Overview

Orcion B.V. specializes in making custom made applications to fulfill their clients' needs, which is the strong point of the organization. These custom made applications are in the form of web applications, written in C#, ASP .NET MVC. In one of the web applications they are currently developing and maintaining, an option for uploading documents is available. However, that was as far as it goes up to the moment this graduation project was started. To view and edit the uploaded documents, their clients had to download them, edit them offline, and then upload them again. Therefore, the company needed a web application that allows the users to manage documents more efficiently, without having to download them at all.

The project was initiated because of the need of having an Enterprise Content Management System whose online editing can be integrated into existing web applications, and this ECMS has to be under the company's ownership. The final product of the project needs to be designed in such a way that it can work as a standalone application, or have its document editing functionality integrated into another application. Since building the complete ECMS in 5 months is impossible, this graduation assignment focused on building a part of the complete ECMS. This will then be the base of ultimately many web applications they will develop.

The project's SMART goal is described as follows:

By 30th of June 2015, implement a web application in ASP .NET MVC which will allow users to manage their folder structures, and edit their Microsoft Word documents without the need of downloading them. The editing functionality will be designed such that it can be integrated to other web application. Whether or not it is ready for integration will be decided by company mentor.

The final product of this project is a DMS (Document Management System), which later on (outside of this project) will be developed into a fully working ECMS, with more than just a document repository. This DMS will be developed both as a stand-alone application, and integrated into their custom-made web applications which have been, are being, and will be developed by the company. Thus, the core functionality of the DMS (online editing) must be decoupled so that it can be used both as a standalone DMS and integrated in a web application. The actual integration of the DMS into a commercial application is not part of the project itself. However, it is required that a clear manual on how the integration should be done is written so that the company can integrate it into their web applications easily.

The functionalities of this DMS consist of the requirements defined in the deliverables section of project plan, chapter 2<sup>[1]</sup>. The most important requirements are the document online editing, which allows users to modify the content of a file without the need of downloading it, and online folders (and documents) management (modify, create, delete, etc.). The online editing functionality of this DMS will be designed in such a way that any application that it is integrated into can use their own file handling mechanisms (Retrieval of file, locks, etc.), and still use the same server implementation to handle the request. This design will then be verified by company mentor, who will decide whether or not it is ready for integration.

This project requires several research to be conducted with DOT Framework, with the questions defined as follows:

1. What functionalities should this DMS have?
2. How can the application read/write the documents without having to download them first? What kind of external library is needed?
3. How can integration to other web application be done?
4. What are the pros and cons of existing CMS/DMS solutions?

This project requires the application to be implemented with the same technologies as other applications of Orcion B.V., which is C# ASP .NET MVC. Visual Studio is used as the IDE for development.

The development methodology used for this project is Scrum<sup>[38]</sup>. Scrum is chosen for its versatility. Using Scrum, all requirements are broken down to atomic level, which are going to be the work items/sprint back logs. This way, executor can easily focus on the items defined for each sprint. Also, Scrum is versatile in the sense that it can adapt to requirements changes midway project easily. Through the sprint meeting, based on the last sprint, the customer and executor can decide which items from the original list defined should be changed or added to be worked on in the next sprint. Since the project executor have had experience with Scrum prior to the project as well, it was then agreed to use this methodology.

While scrum, an agile methodology, was intended to be used for the project, it was not strictly followed. The project started off with requirement planning and research phases, in which the initial requirements of the project were gathered. Most of the researches for the aforementioned questions were also done in this phase. It was followed by design phase, in which the initial designs of the UI, database, and application structure were constructed. The implementation phase was divided into 6 sprints of 2 weeks, in which detailed designs, analysis, implementation and testing took place. The intensive testing was done in the last sprints.

At the conclusion of this project, the following products are going to be delivered:

1. The DMS application with at least all mandatory functionalities agreed in the project plan.
2. User manual for using the application.
3. Technical manual for setting up the server.
4. Integration manual for WebDAV capability.
5. Project Report.
6. Demo for company.
7. Final presentation for Fontys.

## 4. Research and Analysis Phase

### 4.1. Requirement planning

This phase took place before sprint 1. It was decided that Scrum was chosen for the methodology to be used for the project instead of waterfall, to allow the project to adapt to changes easily. With scrum, requirements can be modified as necessary at the beginning of the sprint. Suppose that in the previous sprint it was discovered that certain functionalities were impossible to be implemented within the timeframe, it can either be changed to what the research deemed possible, or scrapped altogether for the next sprint. Research for most of the research questions were conducted in this phase. Referring to research question 1 (see chapter 3) regarding the functionality of the DMS, during the interview prior to this project, a discussion regarding the assignment took place (FIELD strategy), where the initial “big picture” of the requirements were acquired, some of which were the online editing of the documents, network folder mapping and the need for integration. Further requirements were acquired by interviewing/discussion with the customer, in this case represented by the company mentor. Interview was selected as the requirements gathering method because by talking face to face, it is easier to gain what exactly the customer needs, not what he wants. Any unclear answer could be cleared immediately through follow up questions. From this discussion, it was decided that the DMS would have the functionalities described in appendix A. Versioning system for the documents was an important functionality to be implemented, as there are more than one way to accomplish this, a research on “Which method of storing old versions of documents are more effective?” was conducted.

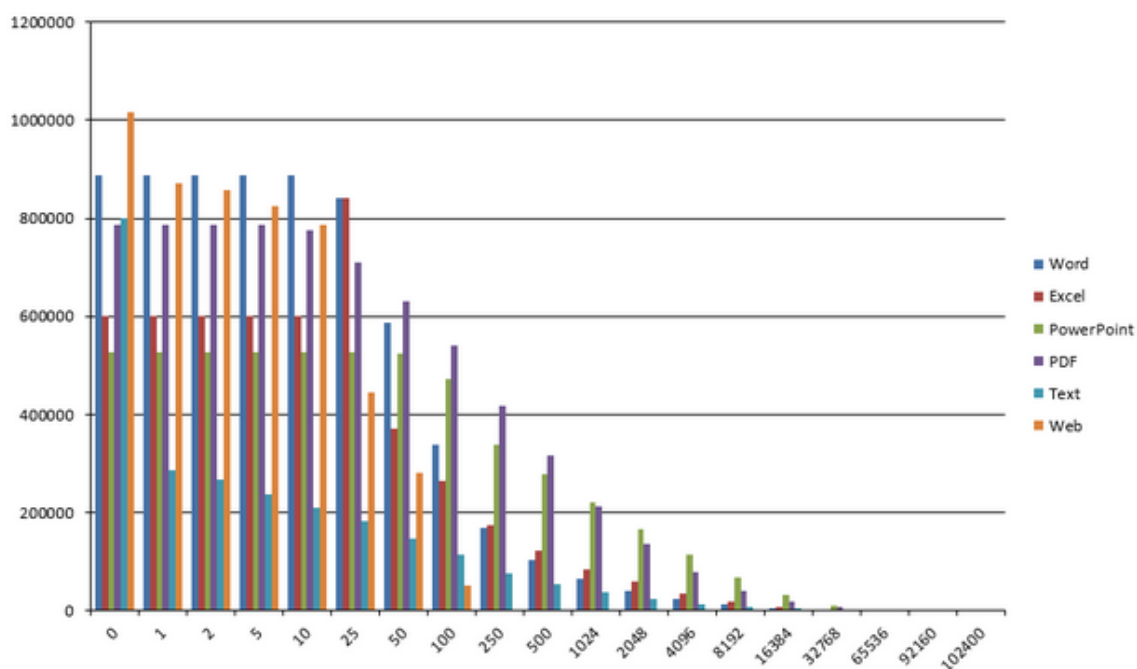
A discussion was held on how the old versions of documents should be kept. 2 options were considered:

- i. As a file in the file system.
- ii. As BLOB in database, distinguished into:
  1. Whole file
  2. Binary difference between latest version and a restore point (a complete version of the document)

There were some things to be taken into account, the most important one was integrity of the old version files. After doing online research, and through discussion with the client on how versioning was to be implemented, the following decisions were made:

- i. Old versions would be stored as BLOB in database, BLOBs with 256Kb up to 1 Mb size in average are more efficiently handled in database <sup>[2]</sup>. As the average of the files to be edited, (Office documents) are around this size <sup>[3]</sup> (also look at figure 1), storing it this way is better. This was confirmed also by company mentor that this application is intended to be used for documents whose size is around this average size. The Microsoft article this decision was based on <sup>[2]</sup> stated that, over time, read throughput will decrease as the files are overwritten. But in this case, that does not apply. As the files stored in database are the old versions of documents, they are not meant to be edited.
- ii. Integrity is kept better using BLOB <sup>[4]</sup>. As users do not have direct access to the database, drastically decreasing the risk of anyone accidentally editing or removing it. Not to mention that BLOBs are basically lines of bytes,

- which is unreadable for normal people. If anyone edits it, there is a high chance of corrupting the file.
- iii. For the prototype, version as a whole file will be stored as BLOB in database. Later on, if possible, storing only binary differences between versions was considered (The idea was based on video compression IPB frames<sup>[34]</sup>). This way even if the size stored in database remains the same, integrity is better secured. The individual row in the database does not contain the whole file (with several exceptions described below), thus reducing the risk of users reading or editing the files improperly.
  - iv. Complete versions are stored every 5 versions to improve the performance and integrity. The application will then calculate the difference between the current version and the latest complete version. Should one of the complete version is broken, users still have other restore point available.
  - v. In the application, metadata is not versioned, meaning that any change to the metadata will not be recorded.



**Conclusion:** The average size of an office document is 321 kB.

Figure 1. Research on the average size of documents [5]

## 4.2. Technology Analysis

There were two main technology analysis that were done for the project. The first one was about the online editing technology WebDAV, and the second was about the HTTP request handlers.

### 4.2.1. WebDAV

After the requirements were defined and approved by Orcion, the next step was to investigate the required technology for enabling document online editing, which is WebDAV<sup>[6]</sup>, short for Web Distribution Authoring and Versioning, to answer research question 2. This is the technology that the company required for the application to use, therefore the research was more into the way WebDAV works. As this was a new technology for the project executor,

research was required to understand how the WebDAV could be implemented and support the online editing. For this, several online sources were used (LIBRARY strategy), Wikipedia <sup>[6]</sup> for basic understanding of WebDAV, and more details in WebDAV's own website <sup>[7]</sup>. From these sources, conclusions were drawn, as follows:

- a. There is a certain standard of the request-response life cycle that all WebDAV compliant servers must have in order for online editing to work. Thus, any WebDAV compliant server will have common parts that can be re-used, erasing the necessity to create it from scratch. This reduces the chance of bugs in the code, as creating code from scratch may result in very unstable code.
- b. This does not mean that any implementation of WebDAV servers can be used as is. As mentioned before, they have some common parts, which corresponds to the minimum http response that a WebDAV capable server must have. Where they get the data from (in this case the documents to be edited) may differ from server to server. This is further shown in the example implementations that was found afterwards (details are in next paragraph) that the implementation of file handling is located on a different project of the same solution and at least two of the server implementations use the same terminology for the classes used to make the HTTP response.

With both points in mind, the project executor began to search for existing, reusable WebDAV compliant servers that could be used as the base of this project. IIS <sup>[35]</sup> itself, in which this project will be hosted, is actually capable of being a WebDAV server. However, there were some limitations, as IIS WebDAV implementation cannot be modified. If the application is hosted somewhere else, it may not work. The server needed to be modified to allow the application to inject the file manager which will get the file data, which is not possible with IIS WebDAV. Also, IIS WebDAV repository must be defined by adding a virtual directory from management studio, which will be the location of the files. Since the application is meant for the users to map repository by themselves through the application, then it is not possible to use IIS WebDAV. Therefore, a decision was made to use a self-implemented WebDAV server, as this will allow modifications to be made as necessary, and it is not IIS dependent. There were several options that could be found such as ITHitWebDAV <sup>[8]</sup>, an implementation of WebDAV server by Sphorium found in SourceForge <sup>[9]</sup>, and a WebDAV application shared on CodePlex.com <sup>[10]</sup>. Out of this three options, only the solution in CodePlex provided an example implementation with the source code. ITHitWebDAV is a commercial product, therefore cannot be used unless it is purchased. Sphorium only has the library that contains the implementation, but no code, and lacking in documentation. Furthermore, according to the review on SourceForge, Sphorium's implementation contains bugs that are not corrected yet. As the implementation provided in CodePlex was felt more than enough to be chosen as the base of this application, it was decided not to search for any other WebDAV server implementation, and use this as the base for the research.

For around 2-3 weeks from this point, a study of the example application previously acquired from CodePlex <sup>[10]</sup> was conducted (WORKSHOP strategy). It was run and observed to see how WebDAV requests work, what the sequences when a document editing request is made contained, etc. This 2-3 weeks' time were used to replicate the functionality of the example in the DMS. This was quite a challenging step, as the executor did not have extensive web communication knowledge prior to this. So, a study on what HTTP methods were required <sup>[7]</sup> <sup>[30]</sup>, http status codes <sup>[31]</sup>, etc. were taken so that the executor gained better understanding to work

on the project. Fiddler <sup>[11]</sup> was used as a tool to observe the underlying communication when a request to the application is made. It was chosen as the executor had prior knowledge of using the application, and therefore it required short time to learn how to use it. At the end of the third week, the executor had better understanding of how WebDAV works, enough to work on the project, thereby answering research question 2.

#### 4.2.2. HTTP Request Handling

Every request made to an ASP .NET application follows the same lifecycle <sup>[12]</sup> (See figure 2 and 3). When the request is made, ASP .NET automatically creates http objects representing the http context, containing the http request and response. Once this is done, the request is intercepted by one or more modules before going to the handler for actual processing, after which it may go through one or more modules. The handler is where the actual call to WebDAV will be processed. To illustrate, in a railway system, modules will be the stations in the middle of the journey, while handler is the final destination. The train may stop at one or more stations, to be modified (people getting on or off the train, etc.) before finally reaching the destination where it will unload all the people inside.

In ASP .NET, these modules and handlers are `HttpModule` objects and `HttpHandler` objects respectively <sup>[12]</sup> <sup>[13]</sup>. Two options were originally considered, using only an http-handler or using both. Http-handler is an extension based preprocessor <sup>[13]</sup>, and thus will respond to any request ending with certain extension or a certain part of the request path (e.g. Request made .... /some/thing/else.doc can be handled by handler mapped to \*.doc or some/thing). The application has no control over the lifecycle of the request. Another drawback is that all handlers must be registered via config file, thus the handler registered will always be the handler used for a certain request. As editing the config file in run time will cause the server to restart.

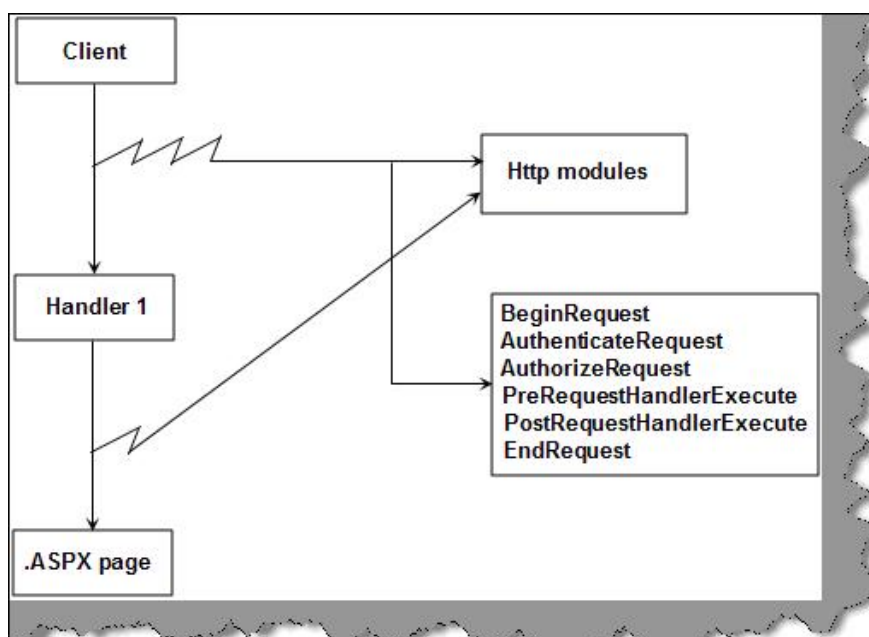


Figure 2 Http Request Life line in ASP .NET<sup>[13]</sup>

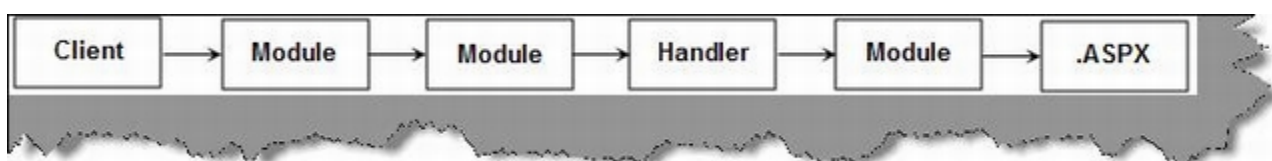


Figure 3. Http Request Life line in ASP .NET<sup>[13]</sup>

Http-module is an event based preprocessor<sup>[13]</sup>, all requests made to the web application will always go through it. With an http-module, the application can handle all the events raised in the lifecycle of the request (begin, authentication, end, etc.) and therefore have complete control over the request. To put it shortly, an http-handler can only have control over a certain type of request depending on the extension or path registered, while an http-module has control over all requests directed at the application.

Another reason for choosing http module is that it allows the application to inject pre-processing logic, like authentication before actually requesting the resources, and handling them using the handler. This way, the user can be authenticated first using a desired authentication method before he/she actually gets the resource. While it is actually possible to authenticate the user using an http-handler, if done so, all the codes (authentication, processing, etc.) have to be written in one long page, which can be hard to understand and manage. By using an http-module, the authentication logic is separated from the actual processing, and therefore easier to manage, the developer can see which part handles authentication and which part actually handles the file. Thus, option 2, which is using both Http-module and Http-handler is chosen.

Table 1 provides the summary of the characteristics of Http-handler and Http-module:

Http-module	Http-handler
1. Event based pre-processor	1. Extension based pre-processor
2. Clear separation between request handling and pre-processing.	2. One method for handling and pre-processing.
3. Can participate or modify each individual request.	3. Participates in only requests with pre-defined URLs
4. Each request can have multiple modules.	4. One handler per request

*Table 1 Comparison between Http-Module and Http-Handler*



## 5. Design Phase

With the initial requirements completed, the basic framework of editing functionality was almost complete. Two more things were left to be designed, the database and the UI for the standalone application. The UI prototype was made using one of the online wireframe designer, MockFlow<sup>[14]</sup> (There was no particular reason for choosing this one, any mock wireframe tool could work). Figure 4 shows the original UI Design made by the executor, as approved by company mentor. This design was made with the consideration that since the users of this application will be windows users, they are used to how windows explorer looks like. Therefore, it was made as close as possible to Windows Explorer's UI. As for the database, MS SQL Server was chosen, as it is the database used by the company.

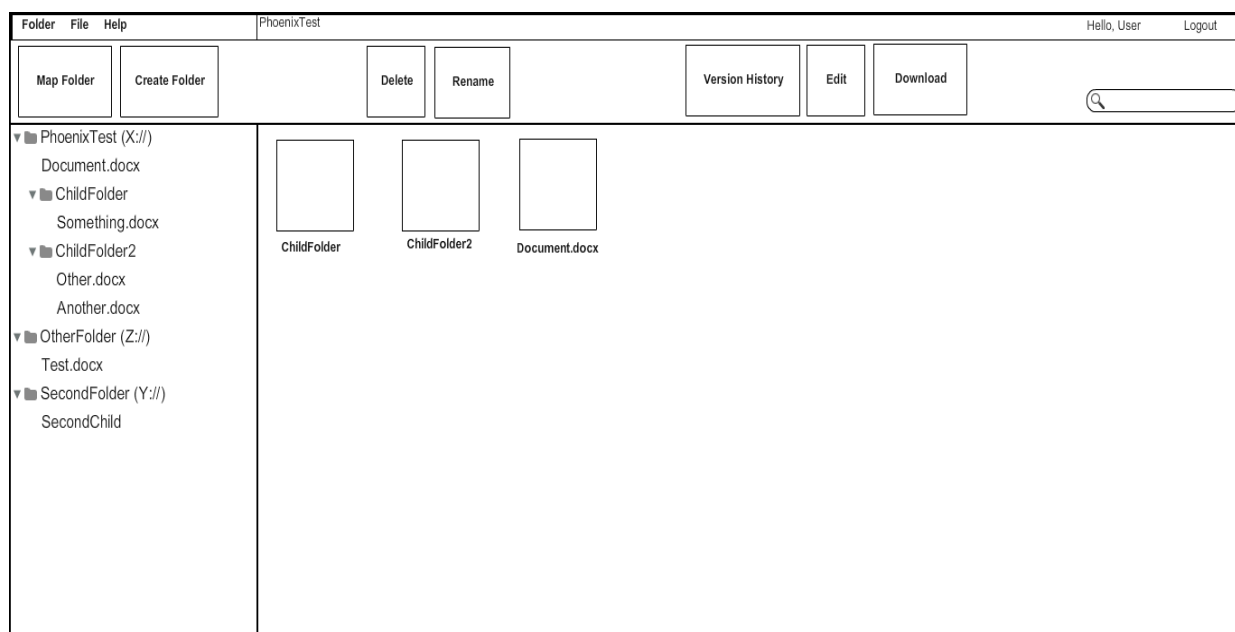


Figure 4 Original UI Wireframe

As the result of this project will be the base of a fully working, commercial ECMS to rival other ECMS, some research were performed to investigate how existing applications work to answer research question 4. For this purpose, several ECMS/DMS solutions were chosen for the following reasons:

1. MS SharePoint
2. Sitecore CMS
3. Kentico CMS
4. Umbraco
5. Alfresco

The first two solutions were the original benchmark for this project, while Kentico was requested to be investigated as an additional benchmark. Umbraco was chosen as it is the CMS solution currently in use by Orcion, and Alfresco was listed as number 1 of top 20 DMS solutions in a survey by Capterra in 2014<sup>[16]</sup>.

This research was done by reading online sources (LIBRARY strategy), such as software review websites, customer reviews, or self-comparison for the available software (WORKSHOP strategy, by installing the product on the pc). The validity of the source website was checked in several ways, some websites are confirmed by Wikipedia (CUBRID<sup>[39]</sup>, MSDN

<sup>[40]</sup>) that they are legitimate websites, some other information exists in several websites, thus increasing the reliability. Some other websites have good reviews from WOT Service, a trusted website reputation rating tool <sup>[17]</sup>. Below is the result of the comparison of the previous 5 ECMS/DMS Application:

#### 1. Microsoft SharePoint<sup>[18][19][32][33]</sup>

- The document size is limited to 2GB, due to database limitation, as files are stored as BLOBs in SQL Server which has an upper limit of 2GB. This is not exactly a problem, as office documents rarely ever exceed 100 MB, they are very small compared to limit.
- Pros:
  - Have tighter integration with Office products than most of the competitors, since both are Microsoft product.
  - Social networking, SharePoint has a Twitter/Facebook-like social media capabilities, in which users can create profile, post “status” or “tweet” and follow other people to see or comment on their activities on SharePoint. (Like Facebook’s or Twitter’s newsfeed)
  - They are easy to use.
  - Newer version of SharePoint has enhanced mobile access experience.
  - It has a fast and easy internal search engine.
  - From developer point of view, when compared to previous versions of SP, SP2013 provides easier means to brand the website with the Design Manager. Developer only needs to edit HTML files of the custom master page and then convert it to ASP.NET master page with Design Manager, unlike in previous versions in which a specific technical expertise is needed to determine what content place holders are needed, etc.
- Cons:
  - Versioning is implemented as storing the whole file, not just differences, thus minor differences will actually occupy  $\pm 2$  times original space needed for the document in the server. (The document itself and the version)
  - Versions are created if either a document or its metadata is modified, so if the metadata is changed but the document itself stays the same, a new version is created. Therefore, it leads to the existence of versions of the document that may not have visible change in the content itself.
  - Not a pure CMS in a sense that SharePoint is built for more team collaboration and community (hence the Social Media), for managing documents or as intranet site. (Sitecore for example, is a pure CMS for building outward facing website).
  - Can only be installed in windows server.

#### 2. Sitecore CMS<sup>[22][23]</sup>

- Pros:
  - It is a .NET based solution, possible to build website from ground up (templates, layouts, and more)
  - It has an easy to use UI, ribbon based on Microsoft Office.
  - New versions have WebDAV support.

- Cons:
  - It may not be affordable for smaller scale projects.
  - It is not optimized for document management, as Sitecore is built to serve as a Web Content Management System, unlike SharePoint which is based on Document Management. Sitecore uses SPIF (SharePoint Integration Framework) to access SharePoint document management functionality. But in itself, Sitecore is better suited for publishing web contents.

### 3. Kentico CMS<sup>[26][28][29]</sup>

- Pros:
  - It thrives on many built-in features, like blogs forums, intranet site, content rating.
  - It has WebDAV integrated into the solution, therefore it can be used to edit documents online. (not in free version)
  - Easy to use from developers standpoint.
- Cons:
  - It is not suitable for small scale projects, as even the basic version is very expensive. The free one requires Kentico's logo in the website.
  - WebDAV is currently supported only in internet explorer, and is not compatible with IIS WebDAV.

### 4. Umbraco CMS<sup>[20][21]</sup>

- Pros:
  - It is an open source solutions.
  - Users have full control of site's appearance.
  - It was designed with developers in mind, thus easy to use for web developers.
- Cons:
  - It requires more time for anyone else to use it.
  - It is not optimized for document management.
  - Many must have features have to be purchased.
  - It does not provide support unless purchased.

### 5. Alfresco CMS<sup>[20][24]</sup>

- Pros
  - It is WebDAV compliant, integrated with Microsoft office and open office.
  - It is suitable for advanced websites, as technical support is not free.
  - It is available on mobile device.
  - It is an open source solution.
  - It is platform independent.
  - The files are stored with different file type and unreadable file name in file system, reducing the risk of corrupting the underlying file.
- Cons
  - It has limited technical support.
  - It has no backup capabilities.

To summarize the differences, refer to the table 2 below. ('v' means the corresponding CMS/DMS application has the criteria on the left side).

	Microsoft Share Point	Sitecore CMS	Umbraco	Alfresco	Kentico
Open source			v	v	
WebDAV compliant	v	v		v	v(only in paid versions)
DMS capability	v			v	v
Ease of use	v	v	v(for developers)	v	v(for developers)
Platform independent				v	v
Optimized for CMS		v	v	v	v
Suitable for intranet	v			v	v
Support	v	v			v
Full control of site's appearance		v	v	v	v
File integrity	v			v	v(if in database)

*Table 2 CMS/DMS Comparison*

The research on existing CMS/DMS solutions resulted in the following recommendations for the development of Orcion's ECMS:

1. Since the application is going to be a commercial application, it is not open source.
2. It has to be WebDAV compliant, so it can be used for editing documents online.
3. It has to be easy to use, this is done by making the UI based on Windows Explorer, since the users are windows users.
4. It was made for Windows, as it is currently set up to work with windows authentication, for intranet. If more authentication methods are planned, making it platform independent will be a good idea.
5. File integrity must be kept by storing the old version files in database and using differential algorithm, to avoid users modifying them directly.
6. Users should be given freedom in their website's appearance, so that they can build websites suitable for their usage easily. If a free version is to be made, then certain limitations can be added (mandatory logo, etc.).

## 6. Implementation Phase

As stated in the project plan, the development was done using Scrum as the chosen methodology. Originally, a scrum meeting was to be held every day, but after a discussion with the company mentor, it was decided that the meeting will be held only for the sprint evaluation/kick-off meeting every two weeks. If there were any problems mid-way sprint, then a meeting could be arranged to talk about them

### 6.1. First Sprint

The first sprint was focused on the most important functionality, the online document editing and the realization of the UI itself. In the beginning of the first sprint, the UI was designed based on the wireframe of the design phase, with some adjustments. As the application is currently set for windows authentication, there is no need to have a login-logout mechanism in the application itself. This will be done when the user logs in to the windows itself. Thus, the log out link was removed completely. The folder content display was changed to a simple list with icons, instead of large icons as in the wireframe. The realized UI is shown in Figure 5.

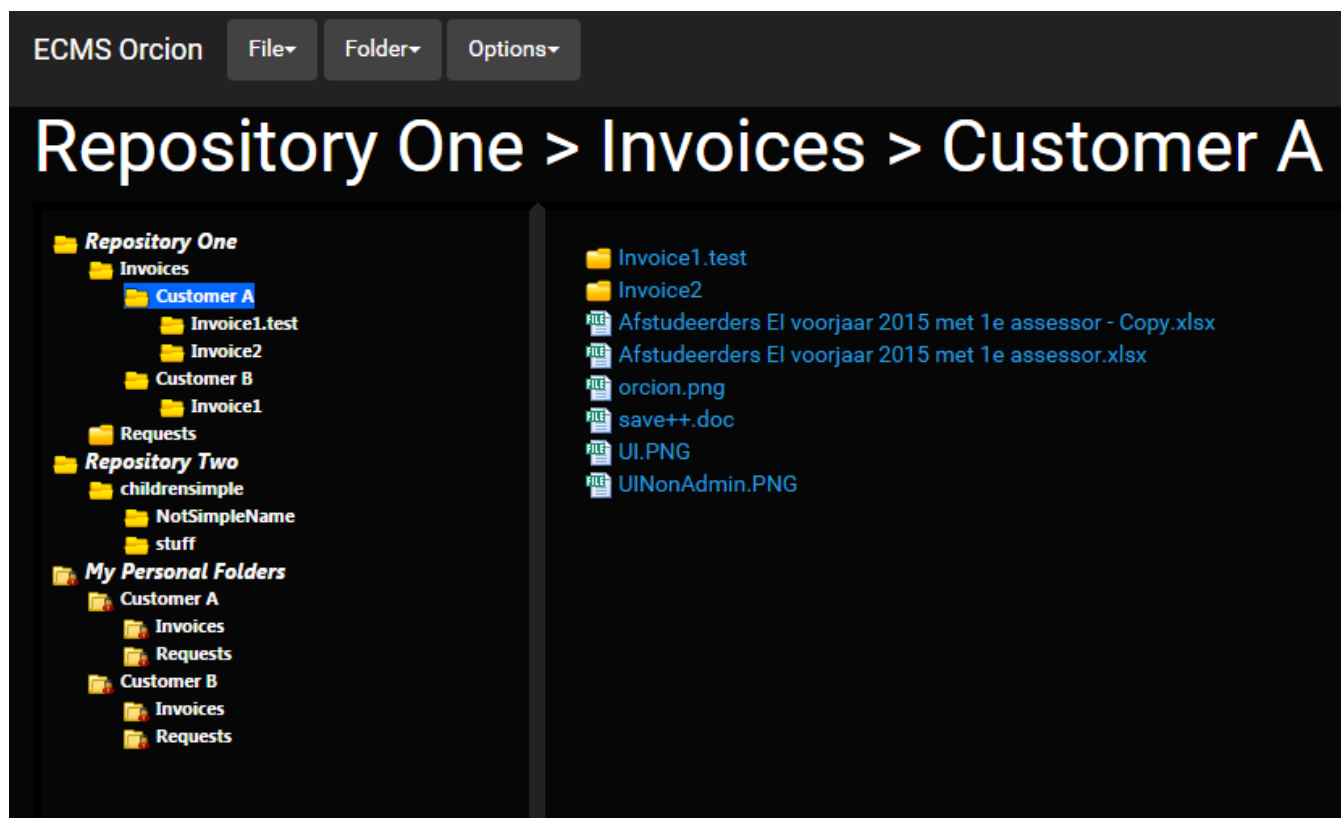


Figure 5 Realized UI

Since the research on how to edit the document using WebDAV was done previously (refer to chapter 4.2), this functionality did not take long to implement. At first, the display name of the document (on the title bar of Microsoft Office) was not user-readable, due to the fact that the URL used for the request did not contain the file name. For the WebDAV request to be made, the URL has to end with a file extension, and since there may be more than one file with the same name, it is not possible to use the file name for the server to find the file. Thus the unique id of the document as it is stored in database was used, which led to that id being displayed as file name, as the URL was something like: <http://localhost/EditDoc/4567asdf1234->

[12we-34rt-56yu-0987l jmn.docx](#). To avoid this, the URL was then changed to include both the file name and the id, so that the file name is displayed instead of the ID. So, the new URL format is: <http://localhost/EditDoc/4567asdf1234-12we-34rt-56yu-0987l jmn/READABLE.docx>. This way, the server can search for the correct file while the user still gets a readable file name displayed.

Once the editing functionality was done, the versioning was implemented. As said before, it was first implemented by storing the whole old version as BLOB in database. Once this was working properly and reverting was possible, it was then changed to storing the differentials as explained in chapter 4.

During the sprint meeting, the company mentor requested (if possible), for the application to open the document directly for editing, as at this point, they all opened in protected view, in which an option to enable editing mode is available for user to choose. This may cause the users to believe that the files are malicious, therefore a sprint item for researching how to open documents in non-protected view was added to the back log to be conducted in later sprint.

Sprint result: The DMS was able to edit the documents online and old versions were stored in the database as differentials.

## 6.2. Second Sprint

Second sprint was focused on implementation of more functionalities to make the application a fully working DMS, starting from folder mapping, adding personal folders, searching and managing folders and documents. Since making sure the functionalities work was priority, all these functionalities were still shown in different views in MVC, meaning that they all had different pages, which can be inconvenient. The web application would jump from page to page to execute certain request. It was then changed to use JQuery modal dialog to display the forms, to avoid reloading the page, instead it would reload only parts of the page that needs to be reloaded when submitting.

Important items in sprint 2 are Themed UI and non-destructive setting. The UI was set to use bootstrap classes so it is easier to change them by simply changing the css files. The most difficult subject of this sprint is the nondestructive setting. This setting, which is the default setting of the application, will make all changes made through the applications visible only through the application. Underlying files and folders will not be affected. There are three exceptions to this rule, which are document editing, folder creation and empty folder deletion, which will always be applied. The users are given the ability to apply all these changes at once whenever they intended it. Since users may have done a lot of soft changes, it can take a long time to go through the all the changes. Thus the order of the operations applied when checking the items are important, if a folder/file is meant to be deleted, there will be no point in renaming or moving it.

The second sprint was completed successfully and earlier than the expected deadline so some work items were added to be completed in this sprint. Resulting functionality was approved by mentor, and there were no additional comments from mentor.

Sprint result: The DMS was able to map existing network folder and manage the folders and files inside.

## 6.3. Third Sprint

The third sprint was focused on the need for integration of document editing (Research question 3), which as stated in chapter 3, requires decoupling of the file handling from request handling. At the beginning, one option was to make the WebDAV server as a plugin, which is

a way of integrating functionality to another application. This will require the company's application to be built to support plugins, which will require the host application to be modified to support plugins. In which case, this plugin, and therefore the online editing ability must be installed individually and is removable by users. This was not the intention of the integration. Another problem, after researching the application that this integration was intended for by asking for demonstration from colleague (FIELD strategy) it was clear that future web applications that may need this integration may have different ways of storing the files, so the server must not implement the file handling. Thus, it was decided to use another option, which is making an API. A C# (the language used) interface was provided at the server implementation which is published as a .dll file.

All that the host application needs to do is reference this dll, and then implement the interface according to its file handling mechanism, then inject the implementation object to the server. After which the server will provide the http response with the result from the file manager to the host application.

Research to open the files in non-protected view was conducted in this sprint. (LIBRARY, WORKSHOP strategy) Originally it was thought that since the files are locally hosted, it was possible to somehow bypass the protected view to allow editing. However, only files whose size is 0 byte can bypass it, every other file is protected, tested with online WebDAV server <sup>[41]</sup>. One way to accomplish this is by turning off protected view of internet-origin file in each MS Office application, in which case, all other files from the internet will also bypass protected view. Since this obviously may cause security breach easily, it is not a recommended idea, but possible to be done. However, every single solution found either suggest turning it off completely as explained earlier, or simply ignore it and just click on the button to enable editing. Even Microsoft themselves, <sup>[25]</sup> recommends to turn it off completely, as local hosted servers are considered part of internet zone, therefore unsafe. Thus the research result was that this is simply not possible at the moment, unless Microsoft changes their implementation of protected view in MS Office application themselves. To conclude, users who want to edit the document will either have to click 'enable editing', or turn it off (per application) for all documents.

During the feedback session, the mentor examined the code required for the integration. The interface provided was required to be as general as possible, meaning that it must not have any application specific code that cannot be reused. For example, it cannot use a type defined only in this application, because it will prevent the integration into their web applications. Most of the functionalities defined in the interface were made to return C# defined types like bool, int, etc. Some exceptions return the type defined in the server, which will be referenced in the integration, and therefore they were acceptable. These exceptions were made to make implementation simpler, because if they are implemented using C# defined types, it will make the codes unnecessarily long. Thus, the implementation was deemed enough for the integration. The protected view from Microsoft Office cannot be bypassed, therefore this requirement was dropped completely, and no further research was required.

Sprint result: The DMS was ready for integration into existing web applications, and documents will open in protected view unless disabled completely.

## 6.4. Fourth Sprint

As all main functionalities have been implemented in the first three sprints, this sprint was focused on the implementation of the extended requirements (See project plan chapter 2.4<sup>[1]</sup>), low priority requirements and improvements. As mentioned before, additional application settings (non-destructive, version storage duration, etc.) were stored in a registry. This was then changed to use an xml config file to store the value, this way, if the application

is moved to different server, it will retain the settings. And in case the registry is corrupted, the application settings will be safe. Also, Ajax was used to replace several page reload to change only the necessary parts of the web page.

By the end of sprint 4, most agreed functionalities (See appendix A) were implemented, the only exception was opening in non-protected view which is not possible, as explained in chapter 5.3. Thus in the feedback session, it was decided that sprint 5 and 6 would be dedicated to testing the application to find and fix bugs, making improvements and writing the documentation (manual, report, etc.).

Sprint result: The DMS had all the possible agreed functionality implemented.

## 6.5. Fifth and Sixth Sprint

The last two sprints were focused on testing the application for finding bugs and improvements. All manuals were written alongside these sprints as well.

The tests were done in for each of the functionalities defined in appendix A. This was done in accordance to the test plan made. One by one, each functionalities were run to see whether they have the intended result. Visual Studio (the IDE used for development) was used for debugging whenever a bug was found. Once the root cause was found, it was then fixed first before testing the next functionality. (LAB strategy)

As this DMS is a web application, a test run was also conducted for the 5 major browsers to ensure that it works properly, regardless of which browser is used. These 5 browsers were Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Safari, and Opera.

Up until this point, documents were opened using OpenDocument Control (for IE) or FFWinPlugin (for the other browsers) in their corresponding MS Office application. This approach has one little problem, Google had announced that it will disable the support for NPAPI <sup>[27]</sup>, which is used by FFWinPlugin, starting from September 2015. Thus, rendering Chrome users unable to use the editing document function. Around this point, while testing for backward compatibility with office 2010, a problem occurred. For the testing, Office 2010 was installed side-by-side with Office 2013. After testing was successful, the 2010 version was uninstalled, unfortunately, this action broke the Office 2013, Excel and PowerPoint could no longer open the documents hosted locally for editing properly. There were several possibilities:

1. The codes were incorrect.
2. The Office 2013 installation was corrupted.
3. The executor's PC was corrupted.

Tests were conducted to eliminate the possibilities. The application was run on another machine, and the result was that no problem occurred, eliminating possibility 1, because if the code was broken, then it would not work there as well. An online WebDAV server was used to test the 2<sup>nd</sup> possibility, and once again, no problem was detected, thereby leaving possibility 3 as the cause. Thus, a research was done on finding the root cause to this online. While doing this research, the executor came across another way to implement this functionality. After testing the functionality using the newly found way, using Office's URI Scheme <sup>[37]</sup> (Refer to table 3 for comparison), it was observed that the problem did not occur using this way, so it was implemented as a temporary workaround while research continued to find the fix. However, no clear solution could be found to fix this problem.

After running this implementation, it was found that this implementation actually solved two problems previously encountered. The Office 2013 that could not open documents properly, and the fact that Chrome will not support NPAPI in the near future. Since this

implementation does not rely on NPAPI, Chrome users will still be able to use the functionality even after NPAPI support is disabled later on September.

This implementation requires that the URLs used to open the document formed in such a way that they contain the corresponding application's unique name. Thus, the links of the document will vary, depending on the extension that is set in the config file. They are now stored in the same config file for the settings to make it easier to add more extension. Previously, supported extensions were listed in an enumeration, requiring the developer to recompile to add more extension(s). The way it is now, there is no need to recompile the application just to add new supported extension.

To summarize:

<b>FFWinPlugin</b>	<b>Using Office's URI Scheme directly</b>
<b>1. Chrome users will not be able to use it after NPAPI support is disabled in September 2015.</b>	<b>1. Chrome users can still use the application even after NPAPI support is disabled in September 2015.</b>
<b>2. Document URI does not require unique commands.</b>	<b>2. Document URI requires unique commands.</b>

*Table 3 Online editing method comparison*

After considering the options, it was decided that the implementation is going to be using Office's URI Scheme directly in the call, since the addition of unique commands does not really add visible burden on the application.

Three manuals were written. User manual, technical manual, and integration manual. The user manual was written as clearly as possible, by providing images alongside the description, to ensure the user understands how to execute certain operations. In case of a complicated procedure, an example was given to provide user with a case where certain functionality occurs. Technical manual was written for the company to be able to set up the machine where the application is running. The technical manual describes the set up that needs to be taken care of when preparing to use this application.

The integration manual was written for the company so that they can integrate the editing functionality to their web applications. It contains the description of the interface that needs to be implemented (parameter, operation, return value, etc.) and how to inject it to the server before the integration can actually work.

Sprint result: Most found bugs were fixed, improvements to some functionalities were made, and developer will be able to add more file extension without recompiling. Manual for integration, user manual and technical manual were written.



## 7. Results

At the end of the project, all functionalities agreed to be implemented were correctly implemented, with non-protected view being the only exception. There were some limitations that can still be improved when this DMS is developed into ECMS, such as the storage of the old versions of documents.

All research questions, including the two additional questions were answered:

- i. What functionalities should this DMS have? (See appendix A for list of functionalities)
- ii. How can the application read/write the documents without having to download them first? What kind of external library is needed?
  - a. As this was required to be WebDAV, this research was changed. The conducted research was about the way WebDAV works, from the moment a request was made until the saving of the documents.
  - b. WebDAV request has a certain minimum life-cycle that must be followed by all WebDAV compliant server, which means that all server will have a common response in regard to this life-cycle. The only difference found was the way the server acquired the files, as it may be stored anywhere. Thus it is impossible to have the same implementation for file manager, this has to be implemented by application that uses the server and let the server use it.
- iii. How can integration to other web application be done?
  - a. Integration was done by creating an API for the file retrieval. The host application (that needs the integration) must supply the server with an implementation of this interface and inject it to the server. The server will then use this implementation to retrieve the file and return it to the host along with the other mandatory response for WebDAV requests.
- iv. What are the pros and cons of existing CMS/DMS solutions?
  - a. See chapter 5 for the result.
- v. Which method of storing old versions of documents are more effective?
  - a. Storing them as BLOB in database is more effective than in the file system. This way integrity is better kept as users do not have direct access to it. They are also stored as binary differences (between current version and a previous checkpoint/complete version) to have better integrity.
- vi. How can document be opened in non-protected view?
  - a. This is currently impossible to be done due to the way it is implemented by Microsoft.

The browser testing in implementation phase yielded the following results:

1. Internet Explorer, Firefox, Google Chrome, and Opera:
  - a. Correctly had all functionality working properly.
2. Safari:
  - a. Display problem was encountered (The resulting layout of the page was not the same as the values provided in the code), as Safari for Windows is no longer being developed, and the latest version for Windows, (v 5.1.7) has some issue with the measurement units.

The WebDAV implementation of this application is incompatible with IIS WebDAV, thus, if IIS WebDAV is enabled for this site, it will not function properly. It is currently unknown which part causes the incompatibility. However, since the self-implemented WebDAV server was not meant to depend on IIS, this issue was left unsolved.

The user manual document has been handed over to the company as well to give to the users, and is also available to be downloaded from the application itself. The technical manual for the company/users to set up the environment for the application to work optimally has also been delivered. Lastly, the integration manual containing the guideline for integrating the online document editing has also been delivered so the company can integrate the online editing to their web application as it was originally intended.

## 8. Conclusions and Recommendations

Below are the conclusion reached after the completion of this project and the recommendations for future development.

### 8.1. Conclusions

The original goal of this project, which was creating a Document Management System that can be used on top of existing file system and integrated into other web applications had been completed successfully.

Referring to the SMART goal, the project was completed before the 30<sup>th</sup> of June. It was developed using ASP .NET and MVC structure. The resulting DMS solution is capable of managing (create, delete, rename, etc.) folders and files that exists in the file system. Users can edit not only Microsoft Word documents, but also Excel and PowerPoint online without downloading. The WebDAV server was implemented with an API, which allows any application to use it with their own file handling mechanism.

Some advantages of this solution in comparison to existing solutions are the ownership of the application, document management on top of existing file system, and non-destructive settings of the application. One major drawback is that it will take a while before the DMS (later on ECMS) can compete with other ECMS, as more development is still needed. Using existing ECMS provides the users with supports from the developer, which for this ECMS will have to be taken care by Orcion themselves as this ECMS is theirs.

### 8.2. Recommendations

Currently, document versions are stored as differentials, however each of them still occupy the same amount of space as the complete versions. This idea was started thinking that this can save some disk space in addition to integrity. Thus, if there is a way to compress the difference without too much burden on the performance, it will be a good algorithm to implement.

Searching files takes a really long time proportionate to the amount of files to be searched. Since searching on content requires all files to be checked one by one, and results aren't displayed immediately after found, but rather at the end after search is finished. It may be a good idea to implement a way of searching that is faster and probably display result immediately after found.

Lastly, if other methods of authentication are to be implemented. The application can be tested on another operating system, as this is a web application which is accessed from internet browser, most of which are not operating system dependent.



## Evaluation

I am very grateful for all the time I had while working on the graduation assignment in Orcion. The colleagues were very friendly and ready to help if I needed it. My supervisor was very understanding and really helpful in guiding me throughout the entire project. There were disagreements during the first few weeks of my time here, mostly about working time, in which I learned that there are some things I had to compromise so that everything works well. The company have rules, as everything else, and I cannot just force my own schedule to fit theirs, as I am just a mere intern after all. In the end, we managed to reach an agreement that worked for both of us without breaking any rules, and I managed to finish my tasks successfully.

There were some things I could have done better. This project was intended to use agile method, in which all phases (analysis, design, implementation, and testing) are repeated every sprint. While the implementation followed this pattern, the testing was not, the intensive testing was done near the end of the implementation, and this could have resulted in disaster if any major bug couldn't be fixed, as there weren't that much time left at the end.

I get to work with Scrum in a company for the first time after several times using it in school. The non-protected view problem was one time that using Scrum was beneficial. In waterfall approach, the client will not know this is not possible until the completion of the product, which may cause disagreements. With scrum, the client was involved frequently in the sprint meeting, where he was informed of this impossibility and a discussion could be made that resulted in the requirement being dropped.

Another thing I need to improve is my language skill, which is really important especially in writing documentation. I was rushing quite a lot in writing this report that I made quite a lot of language mistakes, some of which may still be in this report. I need to learn to be patient with myself, and really read what I wrote carefully, because otherwise I may miss a lot of things.



## References

- [1] Damara, D. (March 2<sup>nd</sup>, 2015). Chapter 2.4. Project Deliverables and Non-Deliverables in Graduation Project Plan-Enterprise Content Management System, (pp 9-10)
- [2] Sears, R., van Ingen, C. Gray J. (April 2006), To BLOB or Not To BLOB: Large Object Storage in a Database or a File system
- [3] Performance and Capacity Planning (Fast Search Server 2010 for SharePoint), (August 8<sup>th</sup> 2013), Retrieved from : <https://technet.microsoft.com/en-us/library/gg604780.aspx>
- [4] (April 29<sup>th</sup>, 2011) Files – in the Database or Not? Retrieved from : <http://dba.stackexchange.com/questions/2445/files-in-the-database-or-not>
- [5] G. Lunde, D., (October 18<sup>th</sup> 2012), What is the average size of an office document, Retrieved from: <http://blogs.technet.com/b/dangl/archive/2013/09/25/3527368.aspx>
- [6] WebDAV, Retrieved from: <http://en.wikipedia.org/wiki/WebDAV>
- [7] Goland, Y., Whitehead, E., Faizi, A., Carter, S.R., Jensen, D. (February 1999) HTTP Extension for Distributed Authoring – WEBDAV, Retrieved from: <http://www.webdav.org/specs/rfc2518.html>
- [8] IT Hit WebDAV System, Retrieved from: [www.webdavsystem.com](http://www.webdavsystem.com)
- [9] gibarolla, (May 28<sup>th</sup> 2013), WebDAV.NET Server Framework, Retrieved from: [www.sourceforge.net/projects/webdav/?source=typ\\_redirect](http://www.sourceforge.net/projects/webdav/?source=typ_redirect)
- [10] dotsad, (August 23<sup>rd</sup> 2013), MVC 4 + WebDav For Office Editing, Retrieved from: <https://mvc4webdav.codeplex.com>
- [11] Fiddler, Web Debugging Proxy, Retrieved from: [www.telerik.com/fiddler](http://www.telerik.com/fiddler)
- [12] ASP .NET Application Life Cycle Overview for IIS 7.0, Retrieved from: <https://msdn.microsoft.com/en-us/library/bb70252%28v=vs.140%29.aspx>
- [13] Koirala, S. (March 4<sup>th</sup> 2009), The Two Interceptors: HttpModule and HttpHandlers, Retrieved from: [www.codeproject.com/Articles/30907/The-Two-Interceptors-HttpModule-and-HttpHandlers](http://www.codeproject.com/Articles/30907/The-Two-Interceptors-HttpModule-and-HttpHandlers)
- [14] MockFlow Wireframe Pro, Retrieved from: <http://app.mockflow.com>
- [15] Umbraco CMS, Retrieved from: [www.orcion.com](http://www.orcion.com)
- [16] Roe, D. (September 17<sup>th</sup>, 2014), Are You a Top 20 Document Management Vendor? Retrieved from: [www.cmswire.com/cms/document-management/are-you-a-top-20-document-management-vendor-infographic-026534.php](http://www.cmswire.com/cms/document-management/are-you-a-top-20-document-management-vendor-infographic-026534.php)
- [17] WOT Services, Retrieved from: [http://en.wikipedia.org/wiki/WOT\\_Services](http://en.wikipedia.org/wiki/WOT_Services)
- [18] Holme, D. (September 21<sup>st</sup>, 2011), BLOBs and SharePoint Storage Capacity, Retrieved from: [www.sharepointpromag.com/sharepoint/blobs-and-sharepoint-storage-capacity](http://www.sharepointpromag.com/sharepoint/blobs-and-sharepoint-storage-capacity)
- [19] Ward, T. (September 25, 2013), The SharePoint 2013 Intranet: Pros and Cons, Retrieved from: [www.cmswire.com/cms/social-business/the-sharepoint-2013-intranet-pros-and-cons-022605.php](http://www.cmswire.com/cms/social-business/the-sharepoint-2013-intranet-pros-and-cons-022605.php)
- [20] Sagynov, E. (2011), Comprehensive Overview of Top 2014 Content Management Systems, Retrieved from [www.cubrid.org/blog/dev-platform/comprehensive-overview-of-top-14-content-management-systems](http://www.cubrid.org/blog/dev-platform/comprehensive-overview-of-top-14-content-management-systems)
- [21] Zinck, B.M. (May 28<sup>th</sup> 2009), Quick Take Review: Umbraco Web Content Management System, Retrieved from: [www.cmswire.com/cms/web-cms/quick-take-review-umbraco-web-content-management-system-002296.php](http://www.cmswire.com/cms/web-cms/quick-take-review-umbraco-web-content-management-system-002296.php)
- [22] Sitecore CMS Review, Retrieved from: [www.business-software.com/product/sitecore-cms](http://www.business-software.com/product/sitecore-cms)

- [23] Review: Sitecore CMS: when platform extensibility is a must, (February 20<sup>th</sup>, 2014), Retrieved from: <https://www.trustradius.com/reviews/sitecore-cms-web-content-management-2014-02-20-12-11-09>
- [24] Shah, V. , Alfresco ECM Review, Retrieved from: <https://www.trustradius.com/reviews/alfresco-2014-05-27-12-30-49>
- [25] Files on local hosted servers are opened in Protected View in Excel 2010, in Power Point 2010, or in Word 2010 (August 26<sup>th</sup>, 2011), Retrieved from: <https://support.microsoft.com/en-us/kb/983201>
- [26] Kentico 8 Review: Kentico CMS, your .NET solution, (March 18<sup>th</sup>, 2014), Retrieved from: <https://www.trustradius.com/reviews/kentico-2014-03-17-13-23-59>
- [27] NPAPI deprecation: developer's guide, Retrieved from: <https://www.chromium.org/developers/npapi-deprecation>
- [28] WebDAV Requirements and limitations, (April 15<sup>th</sup>, 2014), Retrieved from: <https://docs.kentico.com/display/K82/WebDAV+requirements+and+limitations>
- [29] Kentico 8: "Kentico Content Management System Review", (December 9<sup>th</sup>, 2014), Retrieved from: <https://www.trustradius.com/reviews/kentico-2014-04-07-12-09-28>
- [30] HTTP Methods definitions, Retrieved from <http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html>
- [31] List of HTTP status codes, Retrieved from: [http://en.wikipedia.org/wiki/List\\_of\\_HTTP\\_status\\_codes](http://en.wikipedia.org/wiki/List_of_HTTP_status_codes)
- [32] Mason, J. (February 14<sup>th</sup> , 2013), SharePoint 2013: Social Features Highlights, Retrieved from: <http://www.cmswire.com/cms/social-business/sharepoint-2013-social-features-highlights-019624.php>
- [33] What's new with SharePoint 2013 site development, Retrieved from : <https://msdn.microsoft.com/en-us/library/office/jj163942>
- [34] Video Compression Picture Types, Retrieved from: [http://en.wikipedia.org/wiki/Video\\_compression\\_picture\\_types](http://en.wikipedia.org/wiki/Video_compression_picture_types)
- [35] Internet Information Services, Retrieved from: [http://en.wikipedia.org/wiki/Internet\\_Information\\_Services](http://en.wikipedia.org/wiki/Internet_Information_Services)
- [36] Wie zijn wij, Retrieved from: <http://www.orcion.com/nl/over-ons/wie-zijn-wij/>
- [37] Office URI Scheme, Retrieved from: <https://msdn.microsoft.com/en-us/library/office/dn906146.aspx>
- [38] Scrum, Retrieved from: [http://en.wikipedia.org/wiki/Scrum\\_%28software\\_development%29](http://en.wikipedia.org/wiki/Scrum_%28software_development%29)
- [39] CUBRID, Retrieved from: <http://en.wikipedia.org/wiki/CUBRID>
- [40] Microsoft Developer Network, Retrieved from: [http://en.wikipedia.org/wiki/Microsoft\\_Developer\\_Network](http://en.wikipedia.org/wiki/Microsoft_Developer_Network)
- [41] Ajax File Browser Demo, Retrieved from: <http://ajaxbrowser.com/>

## Appendix A: List of Implemented DMS Functionalities

Below is the list of the functionalities that were implemented in the resulting DMS application:

- a. Online document editing, including concurrency management.
- b. Addition of repositories to the DMS application.
- c. Themed UI, allowing the company to change the styling by simply replacing the file with another bootstrap templates.
- d. Document editing integration, this is about the way that the application is designed, not the actual integration.
- e. Personal folders for each users (Facilitating the groups for the documents)
- f. Create, Rename, and Delete folders.
- g. Search document by file name.
- h. Delete document.
- i. Versioning system for the documents.
- j. Search tags for document, and search using them.
- k. Downloadable documents.
- l. Adding groups for the personal folders, according to folder names of each of their ancestors up to themselves (if they exist as an actual folder).
- m. Non-protected view for the documents when editing online.
  - a. Cannot be implemented due to how the protected view is designed.
- n. Separation of the view of folders and documents.
- o. Downward compatibility with previous versions of Microsoft Office.
- p. Applying underlying changes to the repository in the web application
- q. Move and rename documents.
- r. Move folders.
- s. Non-destructive setting and means to apply those changes, changes to folders and documents are not applied to their underlying items with the exception of few things. This is the default state of the application, and when the user decided to use it permanently, it must be made possible to apply all the “soft changes” previously made to the underlying structure.
- t. Upload documents
- u. Version expiration
- v. Search on content

## **Appendix B: Project Plan**

# Graduation Project Plan

---

Enterprise Content Management System



**Author : Denny Damara**  
**Fontys student number : 2185000**  
**Date : 2/3/2015**



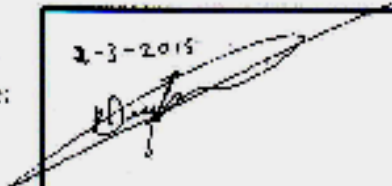
<b>Data student:</b>	
Family name, initials:	Damara, DD
Student number:	2815000
project period: (from – till)	2-2-2015 until 30-6-2015
<b>Data company:</b>	
Name company/institution:	Orcion B.V.
Department:	-
Address:	Industrieweg 23c, 5066 XJ, Moergestel
<b>Company tutor:</b>	
Family name, initials:	Deijkers, RD
Position:	Technical Director
<b>University tutor:</b>	
Family name, initials:	Lahaije, PL
<b>Final project plan:</b>	
Title:	Graduation Project Plan-Enterprise Content Management System
Date:	2-3-2015

Approved and signed by the company tutor:

Date:

2-3-2015

Signature:



Approved and signed by the university tutor:

Date:

Signature:

Lahaije, Paul  
P.D.M.E.

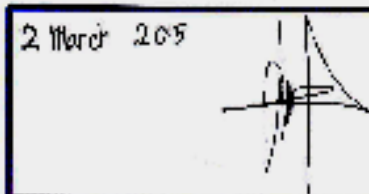
Digitaal ondertekend door Lahaije, Paul  
P.D.M.E.  
DN: c=NL, o=Stichting Fontys, cn=Lahaije, Paul  
P.D.M.E.,  
1.2.840.113549.1.9.2=p.lahaije@fontys.nl  
Datum: 2015.03.03 14:40:29 +01'00'

Agreed and signed by the student:

Date:

2 March 2015

Signature:





# Table of Contents

1. Introduction .....	7
2. Project Statement .....	9
2.1. Formal Client .....	9
2.2. Project Leader .....	9
2.3. Project Goals .....	9
2.4. Project Deliverables and Non-Deliverables .....	9
2.5. Project Constraints .....	10
2.6. Project Risks .....	10
2.7. Communications Plan .....	11
2.7.1. Project Progress .....	11
2.7.2. Document Review .....	11
3. Methodology .....	13
4. Project Phases .....	15
4.1. Initiation and Planning .....	15
4.2. Development .....	15
4.3. Closure .....	16
5. Project Management .....	17
5.1. Money .....	17
5.2. Skills .....	17
5.3. Quality .....	17
5.4. Information .....	18
5.5. Project Timeline .....	18
5.6. Organization Structure .....	19



# 1. Introduction

Orcion B.V. is an IT company that helps other organization to work more efficiently and effectively. They provide their clients by developing and maintaining professional websites and custom web applications.

In these web application they develop, there is a need for managing the documents, thus needing functionalities of a Content Management System (CMS) to be integrated into those web applications already developed, or any future development. The company wants to have this CMS under their ownership and later on sell it with their brand, instead of using an existing CMS.

They aim to develop a new advanced product that can later on engage with internationally known product in the industry, in this case (Enterprise) Content Management System, such as Microsoft Sharepoint or Sitecore CMS. The final product of this project will then be the base of many web application they developed or will develop, as the functionalities are designed to work both as standalone application or to be integrated into another applications.

This project aims to fulfill that goal through ECMS <sup>[1]</sup>. This ECMS is a web application that allows the users to map their existing network folder structure and manage them all through one application. It allows the users to create, edit, and manage folders or documents. This project aims to only develop the Document Management System part of the entire ECMS.

There are notable benefits from this project. The company will have the source code of the application, and therefore allowing them to modify it to suit their customer's needs. While there are some renowned CMS already in existence, they may have some functionalities that the company and their customers does not need. Therefore this ECMS will allow them to customize it to suit the need of their customers. Another advantage is that it provides the customers with an ability to better organize documents of an existing network structure and allows them to try it out without actually mess up their original folder structure. Lastly, the functionality of this ECMS can be integrated into existing system.

This document will describe the project overview, methodology and phasing as well as management plan.



## **2. Project Statement**

### **2.1. Formal Client**

On behalf of Orcion B.V., Ron Deijkers, the Technical Director will be the formal client of the project.

### **2.2. Project Leader**

Denny Damara, a fourth year ICT and software student of Fontys University of Applied Science, Eindhoven and intern at Orcion B.V.

### **2.3. Project Goals**

The project was initiated because of the need of having a Content Management System that can be integrated into existing web application, and this CMS has to be under the company's ownership, not an existing one. The final product of the project is designed so that it can work as a standalone application, or have its functionality integrated into another application. This will then be the base of ultimately many web application they will develop.

The project's **SMART** goal is described as the following:

By 30<sup>th</sup> of June 2015, implement a web application in ASP .NET MVC which will allow the users to manage their folder structures, and edit their Microsoft Word documents without the need of downloading them. The editing functionality will be designed such that it can be integrated to other web application. This integration will be measured by company mentor's decision.

### **2.4. Project Deliverables and Non-Deliverables**

This section describes the final deliverables of the project (The application, documents, etc.) along with the non-deliverables.

Project Deliverables:

1. ECMS, Enterprise Content Management System, with at least the following functionality:
  - a. Mapping multiple folders to the application.
  - b. Create, edit, and delete folders and documents from the application.
  - c. Editing or renaming the document directly without downloading.
  - d. Versioning system for the document.
  - e. Search tags for the documents.

The application must be designed in such a way that it does not modify the underlying folder structure.

2. Possible extension of the assignment
  - a. Adding group for the documents.
  - b. References to the same file from 2 different links in the application.

3. Non-functional deliverables:
  - a. Application must be easy to use.
  - b. Maintainability, company should be able to maintain the code easily.
  - c. Extendibility, easy to add more functionalities.
  - d. Ability to integrate the functionalities of this ECMS to other web application.
  - e. Concurrency management.
4. User stories document, consisting of the requirements of the application.
5. User Manual, describing clearly the usage of the ECMS.
6. Final report of the project (one for Fontys, and one for Orcion)
7. Final presentation of the project, once in the company and once in the school for final defense.

Non-Deliverables:

1. Changes to underlying folder structure reflected in the application.
2. Repository for other types of data (images, etc.)
3. The integration to company's web application.

## 2.5. Project Constraints

1. The ECMS must be written in ASP.NET MVC, C#.
2. Minimum deliverables must be completed within 5 months of internship duration.

## 2.6. Project Risks

Risks	Level	Possible Solution
<b>Lack of knowledge of the programming to do certain tasks</b>	Medium	Make a list of required knowledge and do pre-research of this knowledge to fill the gaps.
<b>Project requirements inflations</b>	Low	Agile methodology is selected to be the way of working.
<b>Delayed delivery of sprint result/wrong hour estimations</b>	Medium	Adjust the work items to be done next sprint to keep up with the target.
<b>Miscommunications, causing differences in project results</b>	Low	Communication plan must be defined clearly. Make a prototype. (UI Mockups) Feedback sessions at the end of every sprint.
<b>Final product cannot be used effectively</b>	Low	User Manual provided must describe how to use the ECMS in such manner that it is easily understandable. Testing and review of the usage, including the manual must be conducted prior to delivery date.

## **2.7. Communications Plan**

### **2.7.1. Project Progress**

Daily meeting with company supervisor will be held at 10:00 am, to discuss progress and problems of the project. Sprint meeting at the beginning of every sprint will be conducted to determine new set of requirements to be implemented.

Weekly progress emails will be sent to Fontys tutor listing the activities done that week and planned activities for the next week.

### **2.7.2. Document Review**

Documents (Project plan, Report, etc) to be reviewed will be sent through email to supervisor to be read and given feedback via email/daily meeting.



### 3. Methodology

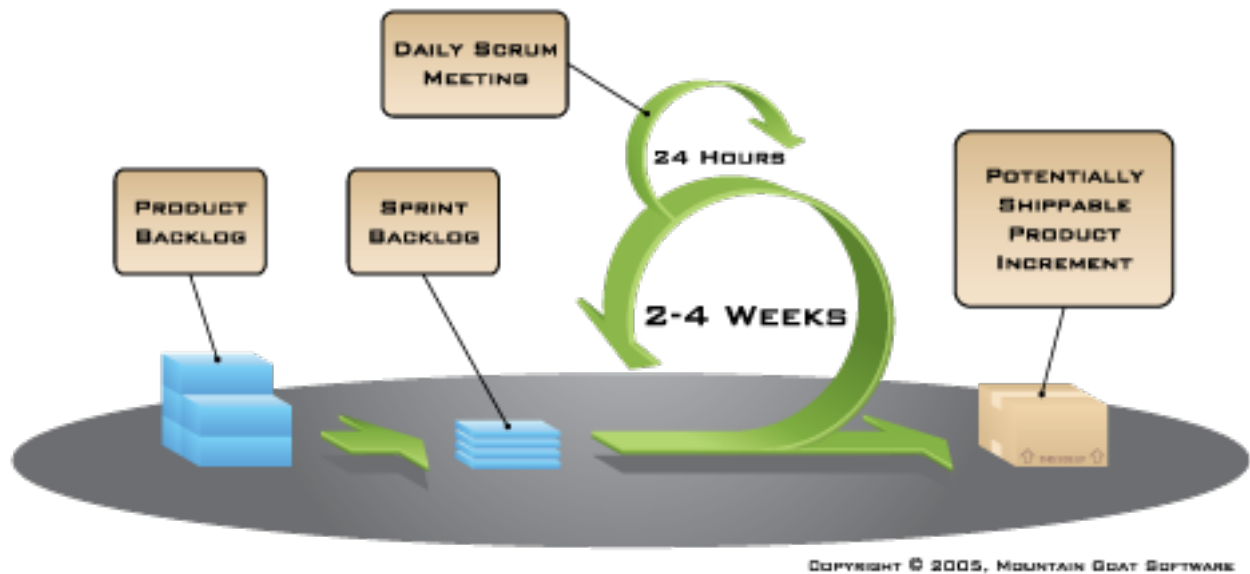


Figure 1 Agile Method: Scrum

This project will be done in 3 main phases. Initiation, development and closure. During initiation, requirements will be gathered through discussion (unstructured interview) with the client. Research for the technologies needed will be conducted also prior to development to ensure that the development runs smoothly. This will be done by FIELD or LIBRARY strategy, by reading sources or asking colleague. Testing will be done by unit testing, functional test and usability test.

In the development stage, this project will be done using agile methodology, to prevent inflations of requirements and to provide agility. In this case agility means that, depending on the feasibility of implementation of certain features, the requirements can be adapted to still have a functional product.

Scrum<sup>[2]</sup> is an iterative agile method that is simple to understand and implement. The team consists of 3 roles, Product Owner, The Development Team and Scrum Master.

Sprint is the fundamental part of Scrum, which consists of planning, daily meetings, development and evaluation. This allows the team to plan the activity for the sprint, and thus have a clear view of the development and a reflection of what have been done or what should be improved in the next sprint.

Scrum has several documents designed for transparency of the project. These documents are product backlog and sprint backlog, which represents the work needed to be done or have been done in the duration of the project.



## 4. Project Phases

### 4.1. Initiation and Planning

In the first phase, the initiation of the project takes place. The goals and scope of the project will be defined here. Therefore, analysis and research of the current situation of the project must be conducted, using LIBRARY (Reading online and offline sources) and/or FIELD (asking colleague) strategy. In this phase, the user stories are defined and the general structure of the application will be defined.

The main research question will be “How can this ECMS be implemented for Orcion such that it allows their customer to manage and edit their document through the network as a stand-alone application and integrated into other web application?” which can be translated into several sub research questions:

1. What functionalities should this ECMS have?
2. How can the application read/write the documents without having to download them first? What kind of external library is needed?
3. How can integration to other web application be done?
4. What are the pros and cons of existing CMS solutions?

Deliverables: Project plan, user stories, general architecture plan, test plan, prototype (UI Mockups).

### 4.2. Development

Development of the application (WORKSHOP strategy) will be done using SCRUM, with sprints of 2 weeks, resulting in total of 6 sprints. At the beginning of each sprint a sprint meeting will be conducted to determine the set of user stories to be implemented, changed, added or removed. User stories are the functionalities planned in the form of description of a user, what they want and why they want it.

Each sprint will consist of research with FIELD and/or LIBRARY strategy (Reading online source, asking colleagues, etc.) for the technologies or means to implement the user stories.

At the end of each sprint, a testing is done using LAB strategy to measure the performance (Usability, responsiveness, functionality) of the current version of the application. This consists of unit test and usability test. The last sprint will most likely use SHOWROOM strategy to compare the results of the development with existing solutions.

The result of each sprint will then be a functional application with the functionality agreed upon the beginning of the sprint. At the end of 6th sprint will be the final product.

Deliverables (for each sprint) : A functional product with the functionalities agreed in the beginning of a sprint.

### **4.3. Closure**

In the final phase of the project, the application is finalized. Project report is finished and the presentation of the final product will take place in this phase.

Deliverables: Project report, user manual, presentation and final product with at least minimum deliverables.

## 5. Project Management

### 5.1.Money

Description	Cost
Workstation and desk	€ 2000
Monthly allowance	(5 months) @ €300 = € 1500
Travel allowance	(5 months) @ €200 = € 1000
<b>Total :</b>	<b>€ 4500</b>

### 5.2.Skills

At the start (Initiation and planning) of the project:

1. Research, to be able to implement the needed functionalities, executor must be able to research all the needed technologies, and to gather requirements using interviews or literature studies.
2. Analysis, to foresee problems and/or risks that may arise during the project and to determine how to test the to-be-implemented features later at the end of the sprint.
3. Writing skill, to clearly state everything in the project plan.

At development phase:

1. Designing skills, to create an intuitive, user friendly design of the application and an effective back end implementation.
2. Programming in C# and ASP.NET, as it is the language that is going to be used in writing the software.
3. Debugging skill, to find and fix bugs in the code of the software.
4. Testing skill, to make sure that the application is working as intended.
5. Writing skill, for writing clear and precise user manuals.

At closure phase:

1. Presentation skill, to be able to explain in the final demo/presentation what he has achieved during the duration of internship.
2. Evaluation skill, to be able to reflect on all experience during the internship for personal and professional development.

### 5.3.Quality

Quality is controlled in daily meeting. First, it is determined by the submission of deliverables (final and for every sprint), whether or not it is according to deadline. Second quality measurement is the number of bugs found at the application. The last one is whether or not the application fulfills all the minimum requirements defined earlier in chapter 2.

## 5.4.Information

	Project Plan	User Stories Document	Test Plan	User Manual	Final Report
Company Supervisor/Client	D, A	D, A	D, A	D, A	D,A
University Tutor	D, A				D,A
Project Leader	W	W	W	W	W

Legend:

A = Approve

W= Write

D = Discuss

Deadlines for the documents:

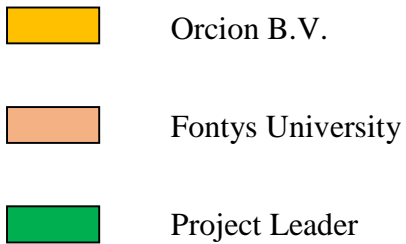
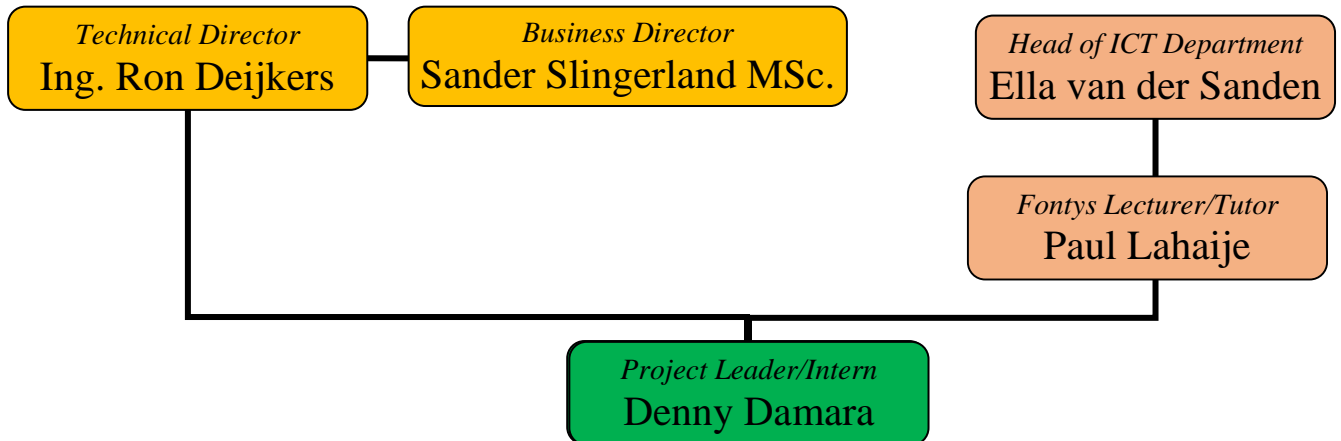
1. Project plan, user stories, test plan : 6<sup>th</sup> of March 2015
2. User manual : 30<sup>th</sup> of June 2015
3. Final report : 16<sup>th</sup> of June 2015

## 5.5.Project Timeline

Activities	Start Date	End Date	Number of Weeks
Initiation, planning, gathering general requirements	2-Feb-2015	6-Mar-2015	5
First sprint cycle	9-Mar-2015	20-Mar-2015	2
Second sprint cycle	23-Mar-2015	7-Apr-2015	2
Third sprint cycle	8-Apr-2015	21-Apr-2015	2
Fourth sprint cycle	22-Apr-2015	5-May-2015	2
Fifth sprint cycle	6-May-2015	20-May-2015	2
Sixth sprint cycle	21-May-2015	5-Jun-2015	2
Finalizing the report and project	8-Jun-2015	12-Jun-2015	1
Prepare presentation	15-Jun-2015	19-Jun-2015	1
Giving presentation at the company*	22-Jun-2015	30-Jun-2015	1
Giving presentation at Fontys*	2-Jul-2015	13-Jul-2015	2

\* Exact date currently undetermined, start date and end date given are the date range in which this activity may occur.

## 5.6. Organization Structure



## References

- [1] ECMS, Enterprise Content Management System
- [2] Scrum, [en.wikipedia.org/wiki/Scrum \(software development\)](https://en.wikipedia.org/wiki/Scrum_(software_development))