CHALMERS

Collaborative web application design and development

Master of Science Thesis in the Programme Intelligent Systems Design

Albert TRIGANON

Chalmers University of Technology

University of Gothenburg

Department of Applied Information Technology
Göteborg, Sweden, June 2011

Collaborative web application design and development

ALBERT TRIGANON

Report number: 2011:075

ISSN: 1651-4769

Examiner: CLAES STRANNEGARD

Chalmers University of Technology

Abstract

Innovation is a key point for companies if they want to remain competitive, especially in sectors that are highly competitive. Just like many big companies, Total understood quickly its importance and is using many different technologies all through its hierarchy. Another key point that might be neglected as its implications are not straightforward is the information sharing process. Sharing efficiently the information is essential when an increasing number of people becomes involved in a process; indeed, the wasted time resulting from a bad information management will consistently increase because of knowledge loss.

To catalyze innovation and avoid knowledge and energy loss, Total has decided to create a PowerPoint document that communicates about the numerical technological orientation of the company and its different branches. The purpose of this document is to provide a good tool for branches to be aware of what is performed in other branches and how. Otherwise, they might be wasting time and money redoing things that has already been done.

Currently, Total branches are spending much time on this PowerPoint document whose format is outdated. Instead, the tool ought to be flexible and collaborative.

The thesis has consisted of creating a web application based on this document in order to make it easier to use, less time consuming, andin also in order to bring functionalities that were impossible to be performed by a PowerPoint file. The work has been divided in two phases:

- A *design* phase that has necessitated the use of methodic approaches in order to evaluate as accurately as possible the user needs and expectations

- A *development* phase based on an agile development method that ended up on a functional prototype of the web application

The report concludes by the fact that spending time on the design method is very necessary as it acts as the foundations of all the further work. With a good field study, the subject can be accurately framed. Afterwards, during the development phase, the use of an agile method was perhaps the reason why the product was a success in the end. A lot of changes had been made all along the development phase as people were realizing little by little that the specifications they had given in the beginning of the project did not fit with what they really wanted.

Index Terms: Web application, SharePoint, .NET, Agile development, User-interface design, Collaborative, Ergonomic

Acknowledgement

First of all, I would like to thank everyone that has been involved in my project.

I will specifically thank Claude Fauconnet for his great help through all the steps of the project, but also the Total DSIT members for their support and the way that I have been integrated. They have been here to support me whenever I needed it and they really gave me the possibility to put together a good project.

I would also like to thank Stephan Coquelet and vNext consulting company, for their help in the technical part especially around SharePoint programming where their expertise was really useful.

I will finish with a specific thank to Chalmers University that offered me the opportunity to carry this project on and to Claes Strannegard that helped me with external stuffs all along my master thesis.

Table of contents

1. Introduction

With no doubt, Internet is one of those big inventions that have absolutely changed the way people live and work. Before the expansion of Internet, information was localized and accessible to fewer people. Now, anybody, rich or poor, can access the same information wherever he or she is on the planet.

Progressively, it has changed everything in our daily life and in the way companies run. Nowadays, every company is connected to the Internet and has its own Intranet network to store its information safely and share it among its employees.

The last decade has witnessed the birth of new management styles whose final purpose was to facilitate information sharing between people. Google thought that letting employees do whatever they liked in the company would stimulate discussions and sharing. Oticon Company decided to become a paperless company in order to share the information more efficiently. NASA is also trying to find a way to prevent the loss of knowledge due to the retirement of most of its highly skilled employees.

Collaborating around a subject is also very important as the pertinence of a content increases with the number of people involved in it. The strength of Wikipedia comes mostly from its regulated collaborative aspect.

This thesis project takes place in the heart of the information sharing process as it aims at developing a collaborative web application to help Total branches to communicate and collaborate about their technological orientation and its consequences on their activity. This subject appeared consecutively to the problems they were having with the old system based on a PowerPoint document which was inducing a lot of wasted time as information coming from all branches had to be updated and merged each year manually prior to be sent back to all branches.

Furthermore, apart from the time and money gained with the better collaboration. This project will also help the innovation process itself: people will be able to share about technologies, which favor the appearance of new technologies and new ways to use them.

The project has been performed at Total headquarters, inside the eNovation department, which is responsible for the update of the document and whose part at Total is to perform a technological monitoring, being able to inform the branches of any interesting innovation and being the technological referent at Total when it is about redirecting properly people wanting to go deeper in a specific technological field.

1.1. Problem description

In such a big company as Total, multiple numerical technologies are used in various branches: RFID, Master Data Management, 3D printing and etc. Considering the time spent by isolated people on each technology, eNovation decided that to avoid loss of knowledge and wasted time, a system had to be found so that every branch could be aware of the projects, studies, made by other branches.

Such a system would avoid absurd situations where a branch would be working on a project that had already been performed somewhere else in the company or even paying an external service for a study that has already been carried out elsewhere at Total.

Therefore, Total and especially the eNovation department have initiated and developed a process based on PowerPoint files called the POT[1] that aims at communicating about Total technological orientation. The POT can be general, presenting the global orientation of the company, but it can also be specific to a branch as they all have different needs, thus using different technologies.

This solution has been used for several years now and has fulfilled its primary objective that was to communicate about the technologies, describing them, their examples of application and creating a kind of a dynamic around the Enovation department: they are warrantor of the technological orientation knowledge and considered as referent when it comes to knowing whether a technology has already been used in Total or even just to know if a specific use of a technology is judicious in a given context.

However, the system does not satisfy everybody and presents several issues because of its old fashioned format that slows the whole process down because of a lack of ergonomic and real data structure:

- Updating the parameters is a difficult task because of the PowerPoint format of the POT which is combined with an Excel file used as a database

- The process suffers from a lack of functionalities: data analysis tool, versioning, user rights management and etc.

Out of those technical issues, another big missing point of the actual system is the inexistence of a collaborative aspect: it is unfortunate that a process communicating about innovations does not provide any tool for users not directly involved in the process to bring their ideas.

All people working with the POT agreed to say that the actual solution was not meeting their needs and that a lot of time could be saved with using an application that could avoid most of the long manual operations.

1.2. Research purpose

The purpose of this master thesis is to design and build a collaborative web application that will help communicate about Total technological orientation like the old POT did. This web application will save a lot of time as all the tedious manual operations will be automated and it will also bring functionalities to the process. The thesis is comprised of two parts:

- A *design* part consisting of observational researches associated with interviews in order to define the functional and technical specifications of the application. The interface definition has also necessitated a thorough study, as a bad interface can be responsible of the failure of an application even though the functionalities are available.

- A *development* part that aimed at building a working prototype of the web application. This step has been made using an agile development method in order to fit at most with the real expectations of the end users and not only with the contract.

Formulation of the research purpose:

"How to design and develop a collaborative web-application?"

[1] Technological Orientation Plan

2. Background

2.1. Total in a few figures

- 5th largest oil and gas company in the world
- Biggest market capitalization of the euro zone: 105,7 billions Euros (12/2009)
- 96 387 employees
- Operates in more than 130 countries
- Exploration and production activities in over 40 countries
- Oil and gas production in over 30 countries
- Total assets: 143 billions Euros (2010)

2.2. Context at Total

In order to carry out this web application project, I have been integrated in the eNovation department.

As in every company whose core business does not belong to the IT field, it is hard to put a real focus on innovations in order to help them to develop. Therefore, ten years ago, Total decided to create the eNovation service inside the DSIT[2]. eNovation department is pretty original as its kingly purpose is to perform a technological watching on what is newly created in the field of numerical technologies that could be used at Total. They also provide ideas and support to branches, relating to one another concerning the technological area.

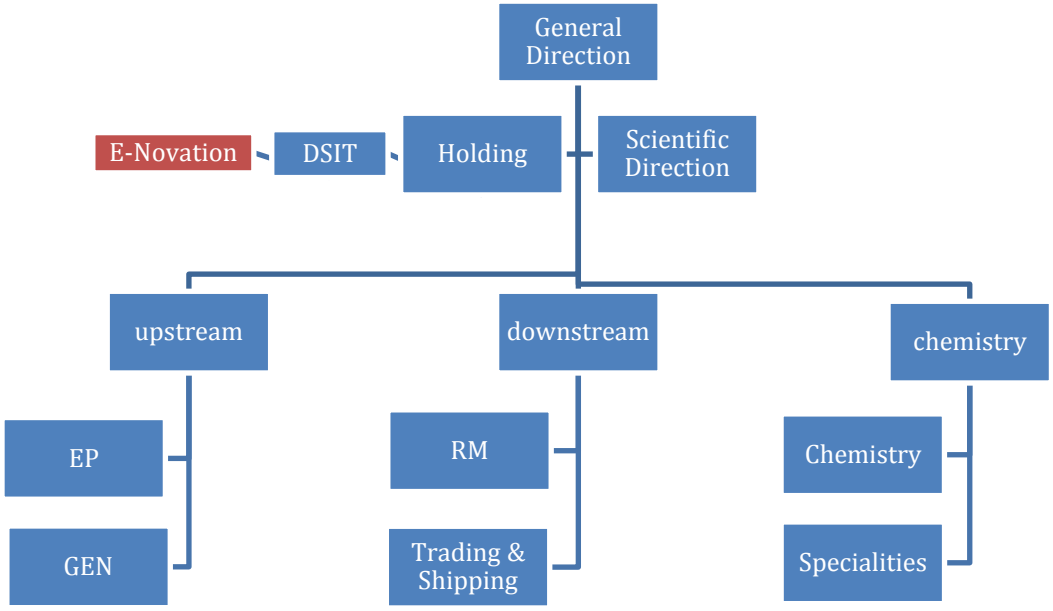
eNovation is aware of all the technologies that are used through the company and have a good knowledge of the market, therefore, they are considered as referent by each branch when it comes to knowing if a technology is used inside the group, or just to obtain some information about a given technology.

Its other purposes are to:

- Animate intra-branch networks to develop information, knowledge and best-practices sharing
- Anticipate branches needs
- Suggest solutions and/or innovative services
- Realize analysis and prototypes
- Promote solutions whose value has been proved
- Provoke creativity

Here is its position inside the group organization chart, which emphasizes the original position of eNovation as an independent entity inside the company:

[2] Telecommunication and information systems direction



Not surprisingly, eNovation is the entity responsible for the POT and they also have to take care of the whole merging phase of the branches POT (POT with a scope limited to a given branch) that aims at providing the general Radar. Thus, they were in the best position to realize how much time was wasted through the whole process while a web application could have done the job in a more efficient way.

2.3. Existing solution

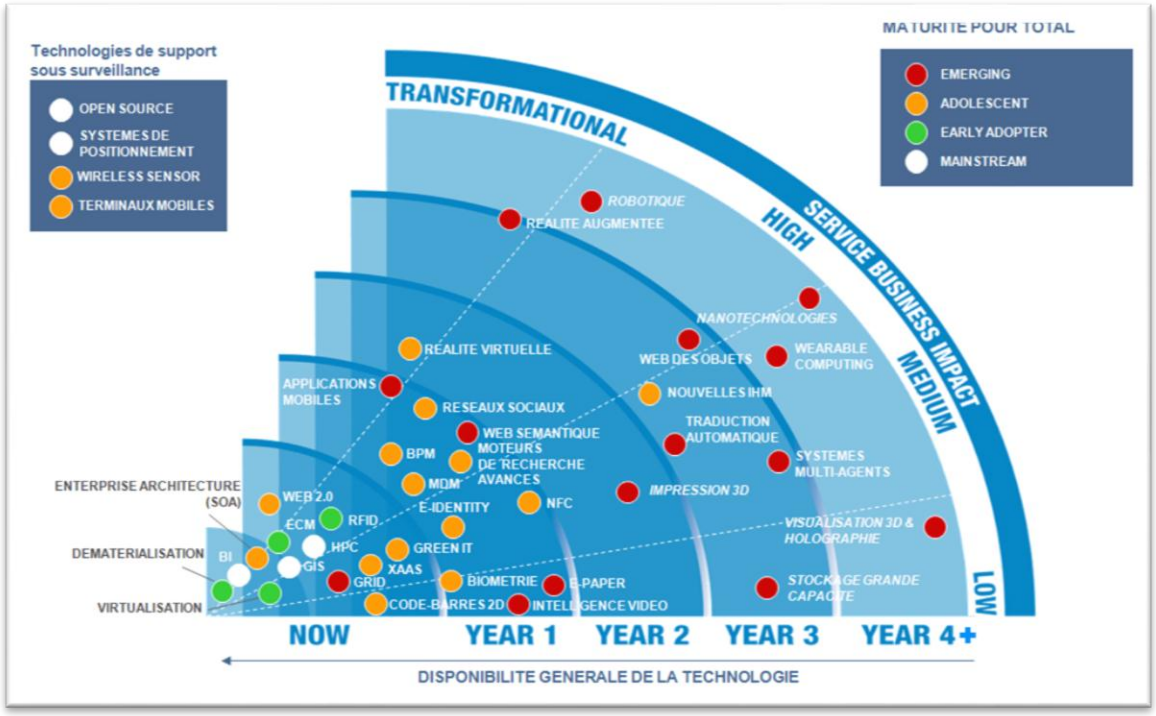
Total is subdivided into branches:

- **EP**: Exploration & Production
- **RM**: Refinery & Marketing
- **Holding**: General administration
- **PCH**: Petrochemical
- **TS**: Trading & Shipping
- **GEN**: Gas and New Energies

Each of those branches is using different technologies according to their needs –these needs cannot be the same on an oil platform and on the Paris headquarters- and the POT is in charge of the communication of those technological uses.

The original version of the POT is made of two major parts: the radar and the technology cards, while all the modifications are reported into an excel file. This POT can represent technologies in the context of a specific branch, but it can also represent the technologies in a global context, at Total (*general* POT).

2.3.1. Radar



The first component of the POT is the radar of the technologies. Each point on it represents a technology instance and it has several attributes:

- **Availability** (now, year1…): Delay for the technology to be operational in the company
- **Impact** (low, medium…): Impact of the technology on the company
- **Maturity** (mainstream, early adopters): Describes the development status of the technology and influences the color of the corresponding point
- **Priority** (low, medium…): Priority set on this technology by the branch

The points/technologies are positioned on the radar according to polar coordinates: the distance corresponds to the availability delay and the angle corresponds to the impact of the technology on the company.

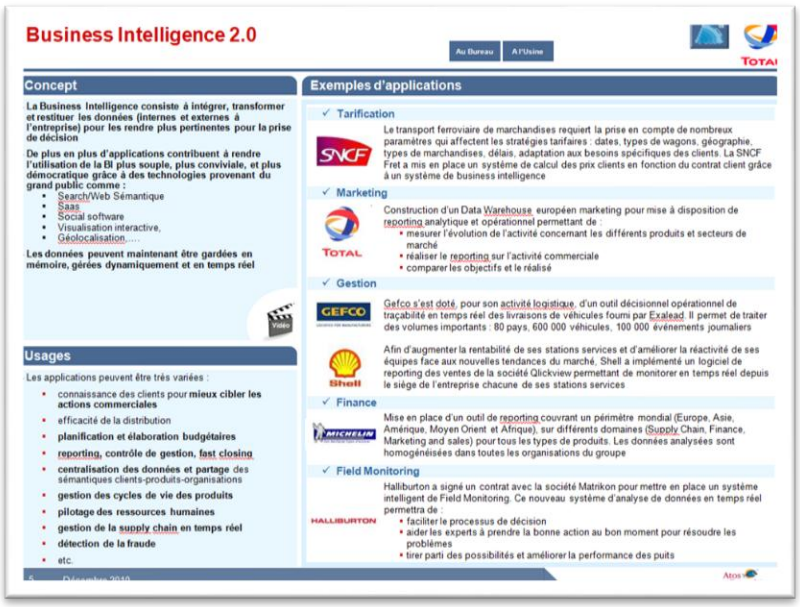
Technologies represented on this graph are called instances because for every technology existing at Total, several instances exist: an instance is defined by three parameters: the branch, the year and the version of the technology.

It makes sense to have several instances for a technology, as the parameters for a given technology for the EP in 2011 are probably different than the parameters for the same technology for the GEN branch in 2007.

Therefore, each radar represents a **set of technology instances**. For example, we can have the radar of the EP, 2010, version 1.

2.3.2. Cards

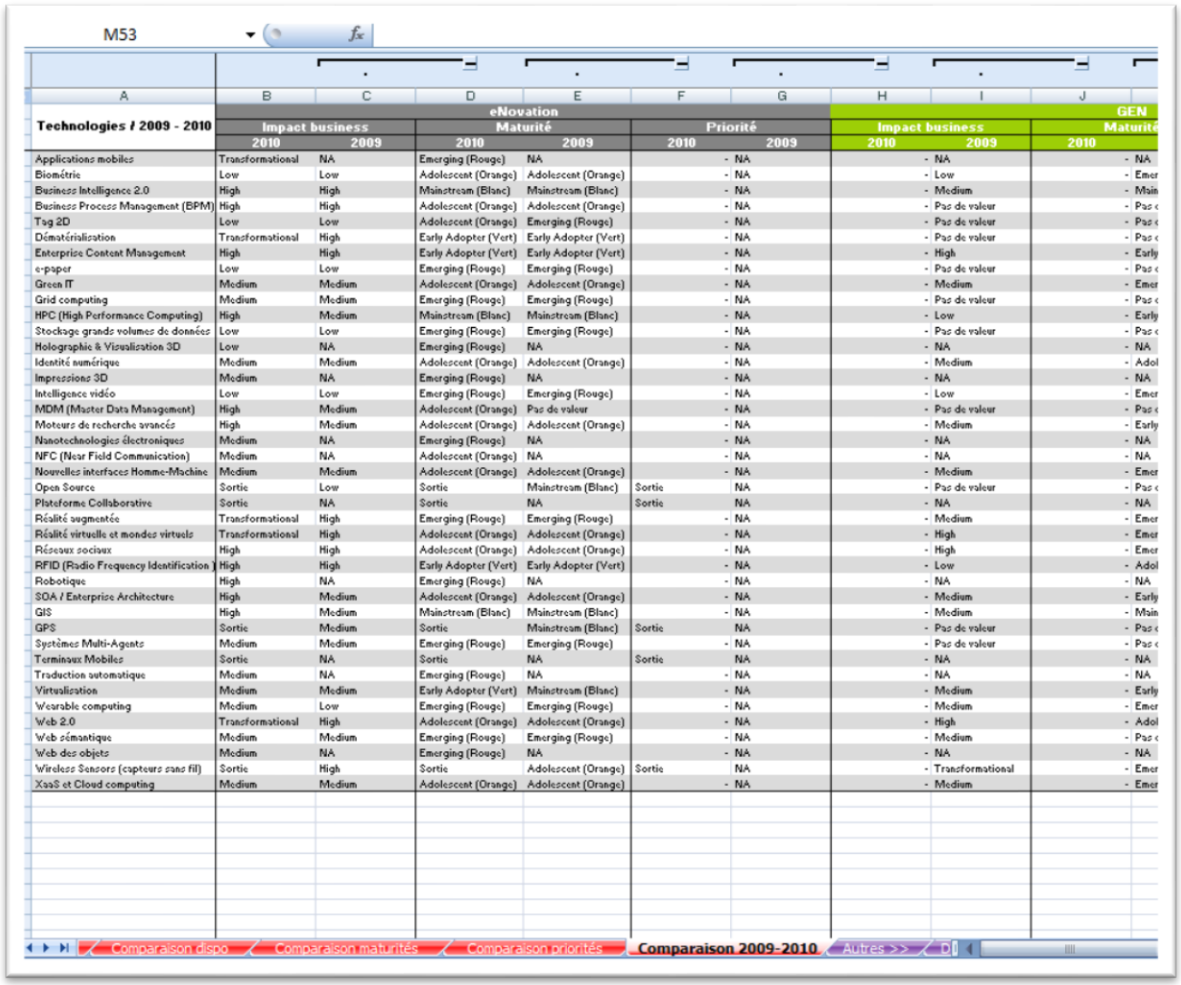
The second major component of the POT is the description of each technology, where the general concept and usage of the technology is described, as well as some examples of uses for the branches.



Those technology cards are also related to an instance of a technology. The description of the technology in 2011 may not be the same as in 2010, even if changes are usually minor between two years.

2.3.3. Excel file

To store the information contained in the POT, an Excel file is used. All the modifications brought on the PowerPoint file are reported on the Excel file in order to store them and to have a more performing way to exchange the data rather than a simple PowerPoint file from which you cannot easily extract the essential data.



Technologies / 2009 - 2010	eNovation						GEN			
	Impact business		Maturité		Priorité		Impact business		Maturité	
	2010	2009	2010	2009	2010	2009	2010	2009	2010	
Applications mobiles	Transformational	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
Biométrie	Low	Low	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Low	-	Emer
Business Intelligence 2.0	High	High	Mainstream (Blanc)	Mainstream (Blanc)	-	NA	-	Medium	-	Main
Business Process Management (BPM)	High	High	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Pas de valeur	-	Pas d
Tag 2D	Low	Low	Adolescent (Orange)	Emerging (Rouge)	-	NA	-	Pas de valeur	-	Pas d
Dématérialisation	Transformational	High	Early Adopter (Vert)	Early Adopter (Vert)	-	NA	-	Pas de valeur	-	Pas d
Enterprise Content Management	High	High	Early Adopter (Vert)	Early Adopter (Vert)	-	NA	-	High	-	Early
e-paper	Low	Low	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Pas de valeur	-	Pas d
Green IT	Medium	Medium	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Medium	-	Emer
Grid computing	Medium	Medium	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Pas de valeur	-	Pas d
HPC (High Performance Computing)	High	Medium	Mainstream (Blanc)	Mainstream (Blanc)	-	NA	-	Low	-	Early
Stockage grands volumes de données	Low	Low	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Pas de valeur	-	Pas d
Holographie & Visualisation 3D	Low	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
Identité numérique	Medium	Medium	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Medium	-	Adol
Impressions 3D	Medium	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
Intelligence vidéo	Low	Low	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Low	-	Emer
MDM (Master Data Management)	High	Medium	Adolescent (Orange)	Pas de valeur	-	NA	-	Pas de valeur	-	Pas d
Moteurs de recherche avancés	High	Medium	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Medium	-	Early
Nanotechnologies électroniques	Medium	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
NFC (Near Field Communication)	Medium	NA	Adolescent (Orange)	NA	-	NA	-	NA	-	NA
Nouvelles interfaces Homme-Machine	Medium	Medium	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Medium	-	Emer
Open Source	Sortie	Low	Sortie	Mainstream (Blanc)	Sortie	NA	-	Pas de valeur	-	Pas d
Plateforme Collaborative	Sortie	NA	Sortie	NA	Sortie	NA	-	NA	-	NA
Réalité augmentée	Transformational	High	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Medium	-	Emer
Réalité virtuelle et mondes virtuels	Transformational	High	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	High	-	Emer
Réseaux sociaux	High	High	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	High	-	Emer
RFID (Radio Frequency Identification)	High	High	Early Adopter (Vert)	Early Adopter (Vert)	-	NA	-	Low	-	Adol
Robotique	High	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
SOA / Enterprise Architecture	High	Medium	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Medium	-	Early
GIS	High	Medium	Mainstream (Blanc)	Mainstream (Blanc)	-	NA	-	Medium	-	Main
GPS	Sortie	Medium	Sortie	Mainstream (Blanc)	Sortie	NA	-	Pas de valeur	-	Pas d
Systèmes Multi-Agents	Medium	Medium	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Pas de valeur	-	Pas d
Terminaux Mobiles	Sortie	NA	Sortie	NA	Sortie	NA	-	NA	-	NA
Traduction automatique	Medium	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
Virtualisation	Medium	Medium	Early Adopter (Vert)	Mainstream (Blanc)	-	NA	-	Medium	-	Early
Wearable computing	Medium	Low	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Medium	-	Emer
Web 2.0	Transformational	High	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	High	-	Adol
Web sémantique	Medium	Medium	Emerging (Rouge)	Emerging (Rouge)	-	NA	-	Medium	-	Pas d
Web des objets	Medium	NA	Emerging (Rouge)	NA	-	NA	-	NA	-	NA
Wireless Sensors (capteurs sans fil)	Sortie	High	Sortie	Adolescent (Orange)	Sortie	NA	-	Transformational	-	Emer
XaaS et Cloud computing	Medium	Medium	Adolescent (Orange)	Adolescent (Orange)	-	NA	-	Medium	-	Emer

2.3.4. General POT

As explained before, the POT represents either the technologies in a specific branch context or in a global context. The last one, called the *general* POT, is built as a synthesis of all the branches POT. This general POT has the objective to give an idea and to communicate about the **general orientation** of the company.

The general POT has to be updated each year because the technologies are evolving very quickly and without frequent updates it would quickly become outdated.

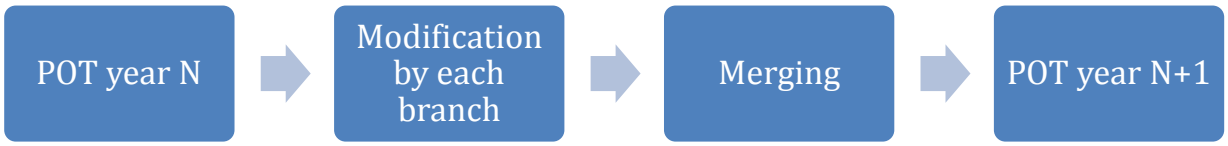
The updating process for a given yearly cycle is the following:

1. In the beginning of the year N, the general POT of the year N-1 is made available to all the branches, as a starting POT.

2. Through the year N, each branch will modify this general POT of the year N-1 according to its own activities. That means that for each technology, the responsible branch will position the point on the radar, bring some parameter modifications and, if necessary, modify the technology card.

3. At the end of the year N, all the branch POT of the year N will be gathered in order to build the general POT of the year N

4. In the beginning of the year N+1, the general POT of the year N is made available for all branches, as a starting POT.



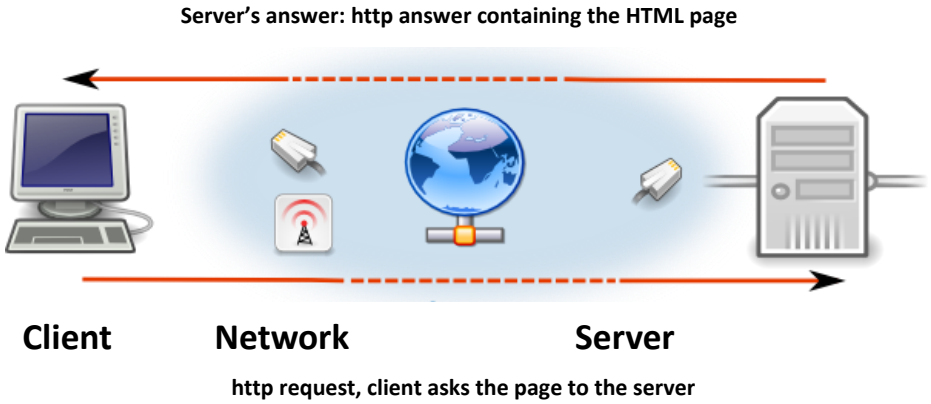
2.3.5. Conclusion about the existing solution

Through this description, it appears that in a mere communication purpose, no important functionality is missing. The POT is already fulfilling its requirements. The main problem, which however is an important one, is that a lot of time and energy are required to perform tasks that a web application could easily do through the automation brought by programmed features, especially concerning the data storage, which is currently done manually with the excel file. Moreover, an application could contain interesting features that are lacking in the actual version of the POT.

2.4. Web application

Considering the functionalities that will have to be supported by the application, when eNovation decided to get rid of the old POT process, they immediately considered the web application format as the most convenient, making it a project requirement.

A web application is any application that can be manipulated through a web browser. Generally, the application itself is stored on a distant server. A good example of a web application is the website gmail.com that contains multiple possibilities without any software that needs to be installed on the computer; you just need to have a compatible web browser, which is the case of most of them.



Here is a summary of the benefits and the drawbacks of the web application format:

Benefits	Drawbacks
• It is very easy to deploy: a web browser is enough	• With no internet access available, the application does not work
• It is very easy to upgrade as the only action to make is server-side	• The Web-Browser has to support the features one wants to build
• It is not space-consuming	• Factually, the user experience will never be as good in a web browser as in an equivalent "classic" software

In the case of the POT, there has not really been any discussion concerning the adoption of this support for the application as its advantages were quite straightforward :

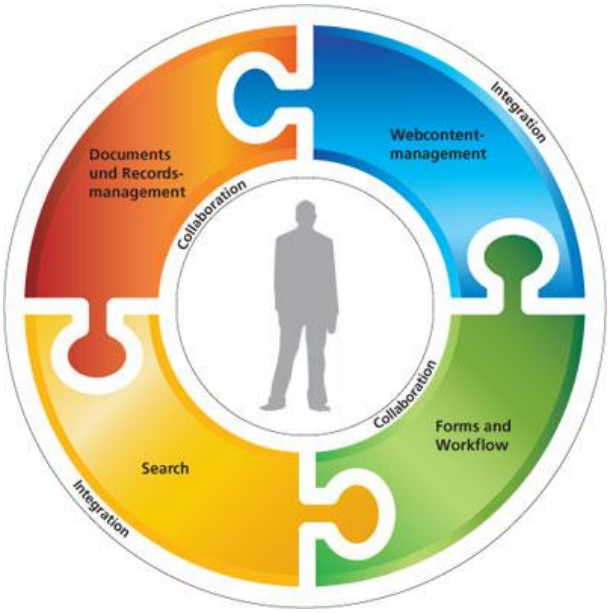
- Total has a reliable Intranet network
- The objectives of the application are not complex enough to necessitate a strong application and to make the internet browser limitation a real problem

2.5. SharePoint

When it comes to the way in which Internet has enabled a better information sharing process, SharePoint is one of those products totally dedicated to information sharing. Currently used in many big companies, Total has in turn decided to use it as its main platform to manage its intranet portal and to provide an efficient file sharing system.

"SharePoint's multi-purpose platform allows for managing and provisioning of intranet portals, extranets and websites, document management and file management, collaboration spaces, social networking tools, enterprise search, business intelligence tooling, process/information integration, and third-party developed solutions. SharePoint can also be used as a web application development platform".[3]

[3] Wikipedia, http://en.wikipedia.org/wiki/Microsoft_SharePoint, last visited on June 14th, 2011



SharePoint Server main functionalities[4]

When SharePoint appeared approximately ten years ago, it was only an electronic document manager made for users located at different places so that they could easily share their documents.

After ten years of development and upgrades, SharePoint is now a highly scalable solution that one can easily customize through its interface as well as programmatically by developing new Webparts[5] to add features that are not available initially.

The main interest of SharePoint is that it is a full solution that is easy to use by non-technical people through its interface while developers can still customize it through advanced development.

In order to fulfill Total requirements concerning its intranet platform, the web application had to be built according to the SharePoint platform. Concretely, data had to be stored under SharePoint lists and the SharePoint base website will be the frame of the web application, enabling it to benefit from the user rights structure of SharePoint, Webparts…

2.6. Context summary

The project took place at the eNovation department of Total, and the mission was to turn the old PowerPoint based POT into a web application based on a SharePoint platform. This application is meant to bring additional features to the POT and avoid much wasted time due to the many manual steps required by the old process.

---

[4] Laboratoire Microsoft, Microsoft Office SharePoint Server 2007, http://www.labomicrosoft.org/articles/Microsoft_Office_SharePoint_Server_2007_Interaction_Groove/0/ , last visited on June 13th 2011

[5] A webpart is an ASP.NET control that can be implemented in any SharePoint page to provide additional functionalities that are not already available in SharePoint initially

3. Method

"The design of good houses requires an understanding of both the construction materials and the behavior of real humans" *(Peter Morville)*

During many of years, in the computers industry, constructors were keeping aside the computer aesthetic aspect but also the ergonomic of the user interface. They were focused on the practical benefit of the feature, which turned out to be a big mistake as people care about the finality of a feature as much as the way the feature is used.

The key point of Apple success is not located inside the functionalities that their products offer, but on the interface that makes people feel like using it. It does few things, but it does them well. Their computers and iPods do not have better technical characteristics than the other products but in the United States, they manage to get 70% of the market for all types of music players[6] because of their really intuitive interface.

The definition of the preliminary requirements of an application is a long process that should not be neglected; spending enough time on these steps can avoid much wasted time for the rest of the development. Indeed, a misunderstanding of the real need of the users can quickly occur if the requirements are not properly framed.

Before everything and as a preliminary work, a thorough study of the existing solution was necessary.

Then, the design phase was comprised of different mandatory steps. At first, there had to be a field study in order to understand where the issues came from in the original process, on which task time was wasted, which features were lacking… This part has consisted of observational researches and interviews given with people involved in the POT process, namely the technologies responsible that were manipulating the PowerPoint files.

When the field study has been performed, with the help of the existing solution study, the specification phase could be done in order to determine which functionalities to implement in the POT web application and how to do it.

These specifications were divided into three distinct steps:

- **Functional specifications**: this part was directly inspired by all the information that had been gathered during the field study: it enumerates all the functionalities that the application has to be able to do

- **Interface definition**: the interface elaboration has consisted of several trials and confrontations with the final users in order to determine the most relevant interface for the application. Particular attention was dedicated to this step because a feature is never used as well as with a good graphical user interface, no matter how great and performing it is on paper

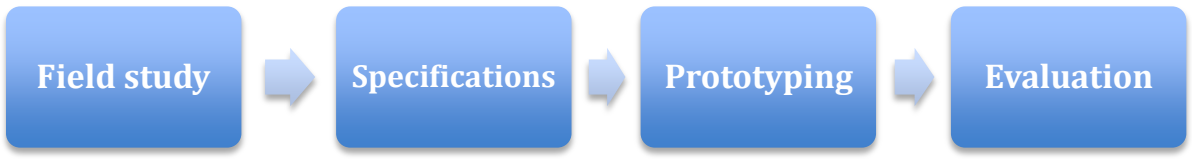
---

[6] Marsal, K., *iPod: how big can it get?* http://www.appleinsider.com/articles/06/05/24/ipod_how_big_can_it_get.html . 2006, last visited on June 13th, 2011

- **Technical specifications**: the technical requirements were elaborated according to the functional requirements, choosing which technology was the best to offer a good user experience in terms of rapidity but also to facilitate the development effort.

When those specifications were made, the development of the prototype could start. At first, the prototype has been developed in a test environment and was used by no one, but after a few weeks, some people started using it in order to help the development by bringing an external point of view.

Summary of the method:



3.1. Field study

The field study is a very important step in an application design. A good field study will give the project its solid basis as well as the directions to follow all along the development phase. User needs have to be perfectly circled before starting programming; developing features and creating a data model are difficult operations closely related to each other. Any modification that has to be brought later in the development will be really time consuming, therefore, this step must not be neglected.

3.1.1. Observational research

The first step that has been performed during the field study is the observational research phase. According to Brown[7], "*observational research findings are considered to be strong [in terms of validity]*". Observations made can be of two kinds: direct observations or unobtrusive observations. The first kind consists of making an observation of people knowing that you are watching them while the second kind involves methods where individuals do not know they are being observed.

The first kind is far easier to do, unfortunately, it is not appropriate for all kinds of situations; the main aspect of the POT project that makes the direct observations relevant is that the topic is all but sensitive. People using the POT will not change their behavior or change their attitude because they are being observed. There is nothing you can be ashamed of while using the POT and nothing that you should hide from somebody, in opposition to more sensible questions. Therefore, the work has consisted of being next to the final users of the POT while they were using it, both individually and in group sessions when they were building the general POT.

[7]Brown, Laura, *Observational Field Research*, http://www.socialresearchmethods.net/tutorial/Brown/lauratp.htm, last visited on June 13th, 2011

Using the observational research has necessitated keeping some points in mind:

- **Positive points**: observations are flexible, no prior hypothesis is required and you do not need to structure your action. This is called a descriptive research. In our case this is an adapted method, even if it would not be suited to all kinds of research.

- **Negative points**: the reality of the *researcher bias*[8], which states that a researcher may "see what he wants to see".

To avoid the researcher bias, some observations were made by people not directly involved in the project, people that had not worked on the subject enough to already have a solid opinion concerning the problems of the POT.

The observations were not stated in any predefined support, observers just had to write what seemed interesting for the study. The absence of any answer grid went in the direction of *non-influencing* the researcher.

3.1.2. Interviews

They were the second part of the field study. While the observational research enabled to deduce a lot of things concerning the POT problems, it was really important to give to POT users the real opportunity to express themselves about their own experience of the tool.

Several kinds of interviews are possible but considering the fact that a lot of the final users of the POT were located at the same place at Total headquarters, face-to-face interviews were given. The purposes of the interviews were the following:

- What are the steps that cause the most wasted time
- What functionalities are lacking on the actual version of the POT

Those interviews were guided with the help of a questionnaire that was followed during the whole interview. In the end the person interviewed was invited to express his or her self about his or her experience of the POT.

3.2. Specifications

This step also required the use of face-to-face interviews that were made with different people, from DSIT final users to technical people in order to define the functional and technical specifications of the future web application.

There were three kinds of sessions:

- *Functional* sessions to determine which functionality was lacking on the past system and which functionalities were the most important to keep among those that were already available. There was also a debate about the new features that could be implemented in the application. Then, a hierarchy had been made between all the features that had to be implemented in the web application of the POT to facilitate the choice of a technology over another

[8] Experiment resources, http://www.experiment-resources.com/research-bias.html, last visited on June 13th, 2011

- *Interface definition* sessions to define the best interface considering the functional needs

- *Technical* sessions to determine at first the feasibility of the functions, but also which technical structure fits the most with the web application

The results obtained during these sessions represented the initial basis of the web application. Then, all along the development of the prototype, new sessions were held in order to bring some modifications resulting from issues that could not have been predicted before, appearing during the development stage.

3.2.1. Functional specifications

"Functional specifications, in the end, are the blueprint for how you want a particular web project or application to look and work. It details what the finished product will do, how a user will interact with it, and what it will look like." (Allen Smith)

Even if the size of the POT web application project is quite small, its functional specification stage has been inspired by the *functional specifications tutorial* by Allen Smith[9].

The very first thing that has been done during this specification process was to bring answers to some basic questions in order to frame the project once and for all:

What is the application supposed to be?

A collaborative web application dedicated to the sharing of information concerning the technological orientation of Total and its branches

What is the application supposed to do?

The main purpose of this application is to facilitate collaboration between the employees. No branch should pay an external service to get information that is already present in the company

Who is going to be using the application?

People in charge of the technologies branch, in the edition mode, and also everybody in a branch that is involved in the innovation process for the reading mode

What are the metrics?

Around 10 000 people will be using the application, but just a few in edition mode.

After having defined those basic answers and with the help of the field study results, the requirements in terms of functionalities were defined during sessions organized with several people involved in the POT process. Some people not directly involved in the POT process were also consulted in order to have non-influenced points of view concerning the improvements that could be made on the actual product.

[9] Smith, Allen, *Tutorial: Functional Specification*, http://www.mojofat.com/tutorial/, last visited on June 13th 2011

Indeed, the main inconvenience for people that are really used to a product or a problem is that they become less and less able to get out of the frames they are in and to go beyond what they already know so well. Therefore, involving those people was a good solution to get new ideas, concerning the interface as well as the functionalities.

3.2.2. Interface specifications

ISO defines usability as *"The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use."* This is exactly what user interfaces are made for.

A good user interface can absolutely change the way people are using a product. Too many people believe that only the technical aspect of an application counts and that what the application does is most important. This is absolutely wrong. A functionality that is hard to use or even hard to learn will never be used by most of the people. Users do not want to read hundred pages of documentation before being able to use it.

The first step of the specification of the interface has been to perform brainstorming sessions concerning the global design of the web application. Should we keep the old one or should we decide to build something completely different?

Once the general aspect of the interface was chosen, the next step has consisted of finding the best way to arrange the functionalities with each other, to position them on the interface. In a nutshell, to find a way to make the functionalities of the application available to the user. To accomplish this task, all the information coming from the results of the field study had to be gathered and put in relation with the essential components of a good user interface as stated in Constantine and Lockwood collection of principles[10] for improving the quality of user interface design.

- *The feedback principle:* the user must be able to have a feedback of his o her actions at any moment. Especially in the case of the POT where the user is making a lot of modifications on sensible data

- *The structure & simplicity principle:* the application must contain only essential information and arrange it in an intuitive way so that the user does not have to think of how the function has to be performed. Furthermore, navigation between major user interface items should be facilitated. People must never struggle to find the window they were on a few seconds before

- *The visibility principle:* the user must not be overwhelmed with too many information and for a given task, all the menus, buttons and labels needed must remain visible

- *The consistency principle*: from a window to another, same functionalities have to be at the same position. Same buttons have to perform the same action. Same colors have to mean the same thing. If a double click is needed to perform some action, the same action on another place has to be called by a double click too. It is absolutely necessary to be consistent in order to prevent the user from being lost and wasting time wandering around

- *The tolerance principle:* when using an application, inevitably, the user will make a mistake and click on something that he did not want to. The user should *not* waste hours and hours

[10] Constantine, L. Lockwood, L. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design.* Addison-Wesley Professional. 1999

of work because of a human mistake, therefore, the functionalities have to be built thinking of this aspect

Those principles were the guiding track of the real interface designing part that has been accomplished with the help of several Photoshop-based prototypes that were made according to those principles. From this point, several sequence diagrams were made so that flow problems could be quickly identified. In parallel, they have been submitted to several people, including an ergonomic specialist that could easily detect interface dysfunctions.

3.2.3. Technical specifications

The technical specification part did not necessitate as many efforts as the other steps. The main reason is that there are not so many ways to structure a web application and there is less room for creativity in this field.

Therefore, after having defined accurately the functional and interface specifications, several sessions with technical experts were organized in order to establish the best technical structure for the application, considering the SharePoint constraint.

Some aspects were put ahead while discussing the technical specificities of the application:

- Displaying the radar and the technologies cards must be done quickly. If the user has to wait for too long he will never want to use the application
- As much as possible, the information displayed on the web application have to be upgraded without any page refresh
- Everyone should be able to use the application on their web browser without downloading any further component
- Mobile devices such as the iPhone and the iPad have to be able to run the application

3.3. Prototyping with an agile development method

Once the specifications were made, the development of the prototype started. The prototype of the application has been developed on a test server to make sure that it was working before deploying it on the production environment. The main interest of using a test environment is that modifications of the infrastructure (SharePoint version, database…) are flexible. This differs from the production environment that cannot be modified as everything has to be validated by a long internal process at Total. Therefore, the application will be deployed on the test environment when everything will be working well.

To focus the development effort on the final user needs and to be as flexible as possible when it comes to the need for change, it has been decided to base the working approach on *agile methodologies*.

A few years ago, while computer programs development was performed according to the same linear process almost everywhere, Ward Cunningham, the creator of the Wiki concept, started to formalize a new kind of working method, more pragmatic than traditional methods and emphasizing the final client satisfaction more than the contract terms, by implicating the user through the whole development effort. This method is called *agile*, in opposition with *disciplined* or *plan-driven* methods. In a scale going from predictive to adaptive, while the last ones are predictive, the agile method stands in the adaptive side.

Here is the introduction of the manifesto for agile software development[11]:

> *We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:*
>
> ***Individuals and interactions*** *over processes and tools.*
> ***Working software*** *over comprehensive documentation.*
> ***Customer collaboration*** *over contract negotiation.*
> ***Responding to change*** *over following a plan.*
>
> *That is, while there is value in the items on the right, we value the items on the left more.*

A major drawback of predictive methods is their lack of adaptability when confronted with a need of change. The agile method is more flexible and enables the possibility of quick changes. It favors face-to-face communication instead of written communication based on multiple specification documents, and takes a working application as its main measure of progress.

The agile method is iterative. In the case of the POT web application project, it will consist of:

- Functionally: a systematic adaptation to the final user needs, as the latter will be involved in all the phases of the development to validate them. This procedure avoids the deception of the client while receiving the application when the contract requirements were not depicting the real final user needs, which is a really common situation as it is rather hard to have an accurate representation of what a project will look like after several weeks of development.

- Technically: a regular refactoring of what has already been developed.

The agile method will also have an influence on the testing sessions. Contrary to other approaches that would leave the testing phase to the end of the projects, after the completion of the application, the agile method will bring the testing part all along the development so that no major modifications will have to be brought after the whole development, thus consuming a lot of time.

Before starting production, while still in the development part, the application will progressively be used by some ENovation members that will give their impressions through the usual sessions so that modifications in the structure will not happen too late in the development phase when it will be really hard to modify the structure of the web application.

3.4. User evaluation

[11] *Manifesto for Agile Software Development*, http://agilemanifesto.org/, last visited on June 13th 2011

In order to finalize the prototyping part, a user evaluation phase has been held. Basically, this step has consisted of a light version of the field study made concerning the old POT:

- **Observational researches** in order to know whether the prototype satisfies or not the user needs and to get some improvement tracks
- **Interviews** to determine accurately the problems encountered while using the application

In fact, this step completes the constant evaluation made during the development phase with the agile method but in a more formal way as it is the last step before deploying the application in the production environment. Therefore, nothing can be left aside at this stage.

4. Results

The completion of the field study provided information about problems that were encountered along the initial POT process but also about missing functionalities. Those points were synthesized and brought in specifications sessions in order to formalize them.

4.1. Problems with the initial process

4.1.1. Technology updates

Not surprisingly, the field study emphasized the problem of the technologies update process, which is very tedious. If a technology is moved on the radar, its coordinates have to be modified also on the excel file. In the same way, if one wants to modify the availability or the maturity of a technology, the modification has to be made in parallel on the PowerPoint and the Excel files. Special care has to be taken in order to keep the two files synchronized. Furthermore, sharing those files is complicated as two users will never be able to bring their own modifications at the same time.

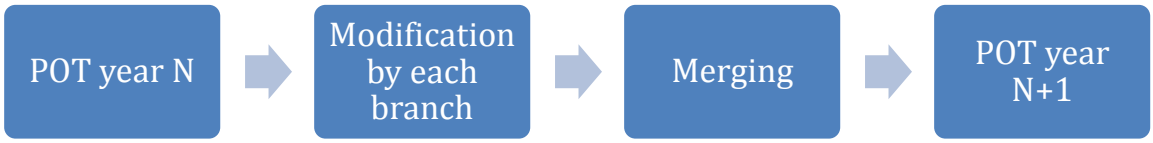
4.1.2. Data analysis tools

The lack of any data analysis tool is very constraining. The most recurrent task that has to be handmade is the comparison between two radars (sets of technology instances). Depending on the parameters that you want to compare, it can turn out to be really hard to do. For instance, if you want to compare the maturity of the *high priority* technologies, you have to use some detours with the Excel files, copying information from one, pasting into another…

4.1.3. General POT generation

The generation of the general POT is the document presenting the whole technological orientation for the group at the end of each year. It is a very important process at Total. To get a better understanding of this complicated process that is surely one of the biggest motivations to convert the POT into a web application, a lot of interviews have been carried out to clearly identify the generation mechanism.

The process has been explained in the existing solution study but here is a summary of the generation task:



Several problems have been identified during this process:

- A lot of time is wasted because of the inefficiency of the PowerPoint format that does not favor cooperation: After each modification brought by each person in charge of a branch, the POT has to be sent to another person in charge, who will bring his or her own modifications.

- The Excel database is difficult to deal with: it is impossible to work only with the excel file because there is no way to have a radar view of the modifications. It is impossible to work only with the radar, because only one set of technologies instances is represented on each. Therefore, back and forth operations between the Excel file and the PowerPoint files have to be made in order to define the *general* parameters of each technology based on each branch values.

- Versioning is hard to make considering the lack of real database structure. For instance, the situation where the POT is being modified at two different places has to be avoided because it would become impossible to know which PowerPoint file corresponds to the latest version. It is also very inconvenient to work with several versions.

4.2. Missing functionalities

In addition to the problems encountered while using the functionalities, the field study also put an accent on functionalities that were lacking on the PowerPoint based POT and that people would like to have in the future web application. Those ideas have all been studied in order to determine whether it was technically feasible and if it was worthwhile to integrate it to the specifications of the application.

4.2.1. Collaborative tools

The POT is a communication tool that deals with the field of innovations and that is particularly rigid and retrograde when it comes to the way in which information is treated. It would be interesting to give to more people the possibility to express themselves about technologies in order to share their experience about it through the use examples for instance.

4.2.2. Comments

The upgrade of the POT requires many modifications; as mentioned before, modifications were very tedious with the old POT. Moreover, another point has been mentioned several times: the possibility to leave a comment after each significant modifications of the POT. Such a functionality will make it easier for somebody that did not participate in the updating session to know why a given technology has been moved from its former position on the radar.

4.2.3. Selective display

Something very complicated to do with a PowerPoint file and that would be easy to program in a web application is the possibility to display the technologies on the radar according to certain criteria. For instance, one could want to display only the *emerging* technologies used *at home*. This would act as a basic data analysis tool in order to compare the technologies according to various criteria.

4.2.4. Radar comparison

This lack of functionality goes further than the simple selective display: people would like to be able to compare radars from two different years or two different branches in order to see the evolution of a technology. With the PowerPoint file, there was no other solution than opening both files and comparing them manually.

4.3. Specifications

The information taken from the field study gave an excellent base to perform efficient sessions to define the specifications of the application.

4.3.1. Functional specifications

Here is the specifications list that came out of the sessions:

- Possibility to move a point (technology) directly on the Radar (its impact and availability will change according to the place it will be)
- Possibility to modify the maturity and the priority of a technology directly on the Radar with an adequate menu
- Possibility to display selectively the technologies according to their maturities, priorities etc… by clicking on the corresponding area of the radar
- Search functionality to look for a technology when a lot of them are displayed on the radar

- Cross-Radar display option that corresponds to a comparison feature between radars from different branches, years and versions
- Wiki functionality that will act as a parallel unofficial technology database that will influence the building of the next year version
- Video functionality as many technology cards contain video documents
- Access to the Total and General card through the technology on the Radar
- Possibility to store a comment after some modifications such as a technology or a maturity change
- All the data has to be stored on a server database that will be updated after each modification brought on by the client
- User rights management: according to the branch to which a user belongs, the functionalities will have to be restricted
- PDF conversion tool
- Auto generation of the final Radar

The main idea that has to be remembered from this feature list is that the POT application had to be absolutely interactive and all the modifications had to be stored in a database in real time. This functional session was not realized with an exhaustive concern, it was used to give a direction to the project, to reason about the right problematic while thinking of the technical specifications and to get a track to follow.

In addition to those functionalities, another specification harder to classify that arose during the sessions, was the idea of web 3.0 compatibility. Which means in our case to bring a semantic dimension to the application. A concrete application to semantic web for the application would be the search feature that could be really efficient using semantic, which would help to establish pertinent relationships between the technologies. Beyond this practical interest, considering that the application's main purpose is to favor innovation, integrating a semantic dimension to the project was really interesting and relevant.

4.3.2. Interface definition

After having determined the functional specifications, the interface had to be defined.

Other than the fact that the interface had to respect the *good user interface principles* of Constantine and Lockwood, several other constraints closely related to the project were to be considered:

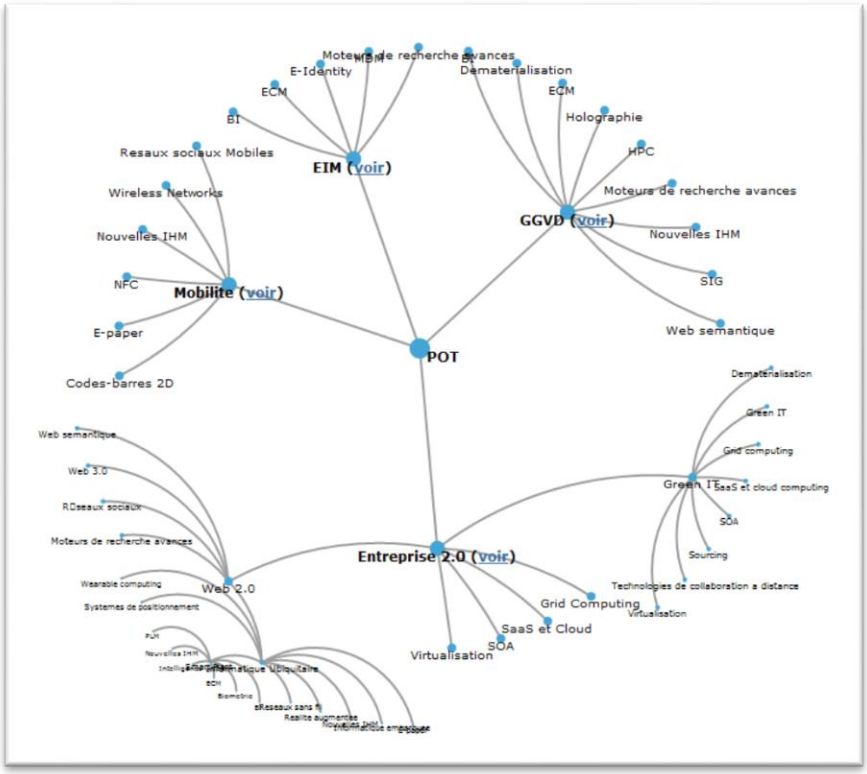
- The application must be usable by non-technical users
- People that will use the web application are used to working with the **same** process but with a different format. Therefore, the habits that they have acquired from their past experience

have to be taken into consideration: one cannot assume that the users are ready to work on an entirely new tool.

The interface contains two distinct parts, the first is the place where the technologies are displayed and the other one is the technologies cards. While the second point is quite rigid in terms of interface, there exist multiple ways to represent technologies on a graph.

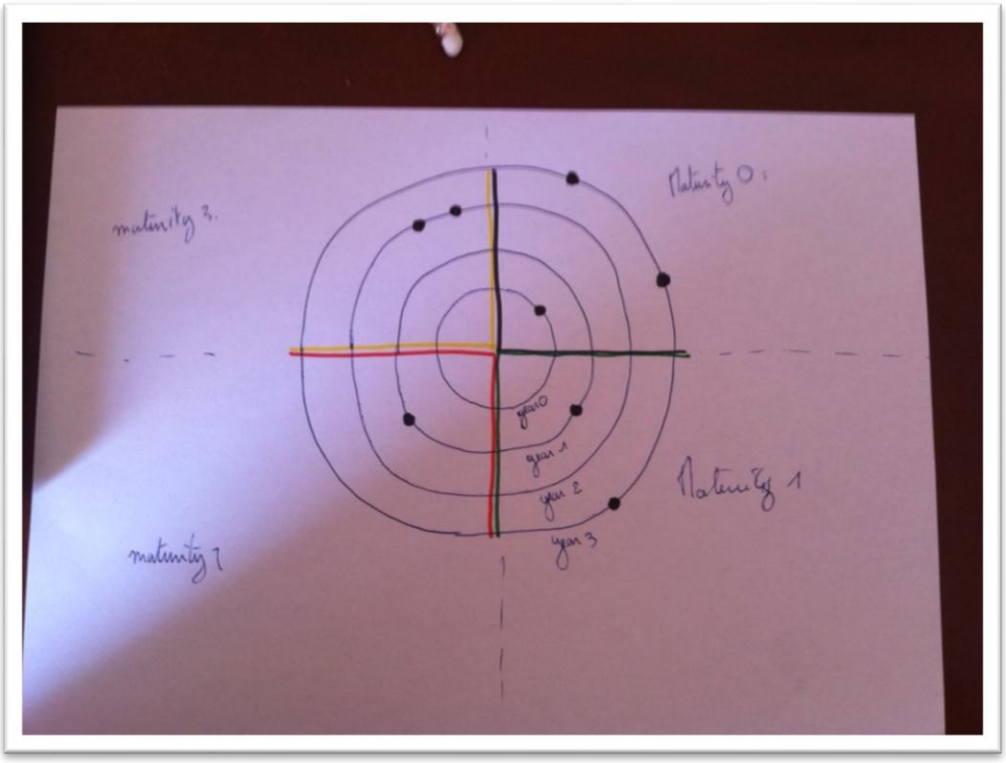
Before thinking of the details of the interface, such as how to communicate between a technology and its card, or how to modify the parameters and which, the global interface had to be determined. In order to perform this task, a lot of paper based or computed interface prototypes have been built. Here are some of the most interesting ones.

The first one is a "graph" that you can navigate through, and then click on a point in order to display its options, cards included.



Advantages	Drawbacks
The representation is clear and interactive The links between the technologies and their appurtenance to a domain appears clearly	It does not respect the availability & impact representation and the parameters are not easy to represent in the view of the technologies. Furthermore, if a technology belongs to several domains, the representation can quickly become confusing

The next one is a more conventional interface that differs from the original POT interface, as the position of a point does not depend on its impact and availability but on its maturity (depending on the quarter) and availability (depending on which concentric circle). The more a point is available, the closer it is from the center of the radar.



Advantages	Drawbacks
The representation is interesting as it emphasizes the value of the maturity by influencing the position of the technology	It can be unclear to anyone that has been used to the past representations and it necessitates a small adaptation period
The availability representation is really intuitive	

Even if the last representation was more conventional than the first one, the common point between them was that they did not display the technologies according the same axis as the original POT. After having submitted those representations to different people involved in the process, the same conclusion appeared: the availability/impact axis had to be kept, as it appeared to be the most relevant criterion. It was important that people should have an idea of the value of those parameters at first glance. From this point, the problem had to be thought out differently, the possibilities were narrowed. In fact, the real problem was that even if the availability/impact representation was not necessarily the most convenient way to represent the technologies, it was the way it had been represented for a long time and people were reluctant to use another representation as they would lose the references they were used to.

At this point, it appeared more and more that the best solution to satisfy everybody was to keep the old representation. People were used to it, and they would quickly become operational on this new platform. The application would visually produce the same result as what was provided before, but in a really more efficient way.

This decision took some time to be definitive as some people unhappy with the original radar were hoping that the web application would provide an opportunity to redefine the whole POT but all things considered, the original representation was satisfying the majority and it has been decided that the focus would be put on the features and the way they would be used.

From this point, the elements responsible of the interaction with the user had to be defined. During all that part there had been a permanent concern of the user interface principles.

How to display the parameters of a given technology?

This aspect is really important as a technology is defined by its parameters and an efficient way has to be found in order to make it accessible. The most straightforward solution consisted of displaying a window with those parameters, a window accessible through some menu linked to the technology. The drawback of that solution is that it really does not facilitate the comparison between technologies, which is something important. Furthermore, if the parameters were displayed in such a way, there would be too much information. The solution that had been found was to provide a direct access to the most important parameters directly through the interface:

Some menus located around the radar would provide a direct access to the priority, domain, maturity, etc… clicking on a given value would only display the corresponding technologies on the radar.

Considering the importance of the maturity, it has been decided that this parameter would be directly accessible through the color of the point representing the technology so that at first glance it would be possible to know the maturity of a technology.

Concerning all the remaining parameters that were not of prime importance but that had to be available, the best solution was to access directly the technology on the SharePoint lists interface, which would also be possible through the radar.

How to access and/or edit the information related to a given technology?

This question was an important point. That information can be of several different kinds: videos, Wiki, technology card, a parameter modification... In fact, all that concerns a specific technology.

Two different approaches were possible:

- Access the technology through the parameter
- Access the parameter through the technology

Does the user have to click on the parameter, and then select the technology? Does he have to click on the technology and then select the parameter? After some discussions it has been decided that it would be really clearer to get to the parameter value through the technology. An intermediate interface will be used, accessible through any technology and in charge of the relation between the technology and its attached pieces, parameters…

(Technology *palette* that has been developed)

This interface contains all the links to the information related to a technology and will be accessible through a click on any technology of the radar.

How should the application give the user a feedback on his actions?

In such an application involving a lot of actions from the users, it is very important to keep them informed about whether their modification has been recorder on the server or not, or even about the state of their manipulation. For instance, it could be interesting to mention somewhere the technology you have just clicked on (when on the *palette*), or even to mention the parameters that are currently selected for the selective display.

In a simplicity concern, it has been decided that the best way (and least intrusive) to display this information was to have some kind of a status bar in the bottom of the radar. Constantly during the utilization of the application, the feedback information will be displayed in the bottom of the screen, so that the user could not be lost.

Should the radar be the central menu of the application?

The radar is the key point of the application, but it is not obvious that it should be the menu of the application. There are several reasons that corroborate this point of view:

- The options and other administrative tools have to be somewhere, it might overwhelm the radar if they are available on it whereas the user watching the radar should be focused on the technologies

- The application will be more scalable if there is an external menu on which a lot of things can be added

As the web application is based on SharePoint and as all the content will be stored in SharePoint lists, it has been decided that the central menu will be the welcome page of SharePoint that can easily be customized and through which there is an easy access to all the content of the web application.

At the end of the interface design procedure, a final session has been held around the six Constantine and Lockwood principles, asking whether the application as designed was respecting or not those points.

- *The feedback principle:* the status bar will keep the user informed of the current state of what he is performing
- *The structure & simplicity principle:* only the essential information is displayed on the radar and on the menus
- *The visibility principle:* all that is linked to a specific technology is displayed on the *palette* of the technology, so that the user is not bothered with potentially useless information
- *The consistency principle*: the structure of the cards is the same from a technology to another (it is built on the same template), the palette is the same no matter what technology you click on
- *The tolerance principle:* confirmation is asked for every critical decision, and it will always be possible to perform a full backup/restore of the application to get the data back as they were before

As one of the hardest things while developing an application is to take some distance concerning the application, an ergonomic-designer will follow the project in order to bring its expertise concerning the interface building.

4.3.3. Technical specifications

The technical specifications describe which technology to use in order to minimize the development effort and to ensure a good flexibility in case of making some modifications to make. A differentiation has to be made between technologies used server side and technologies used client side.

Total's computer infrastructure is all based on a SharePoint platform, which is very convenient to manage content and documents for non-experimented users. Therefore, there was no question about the fact that the application had to be compatible with this platform. There are several obvious motivations for this choice, but one of them is less obvious: SharePoint offers a unified interface to handle data and users are used to it. Thus, the web application will be easier to use, as they already know how to store data inside SharePoint.

However, the use of SharePoint includes several constraints:

- Server side, a .NET language has to be used to develop webparts (components that can be included in any SharePoint page to bring functionalities)
- SharePoint lists are not as easy to manipulate as a real database management system like SQL Server

The major problem concerns the second point as SharePoint lists are not built according to a relational model, therefore you cannot make jointures easily and you have to torn SharePoint to use them as you wish. A possibility could have been to use a separate SQL Server database but all thing

considered, keeping SharePoint lists is the more convenient option, as the SharePoint Object Model facilitates the querying.

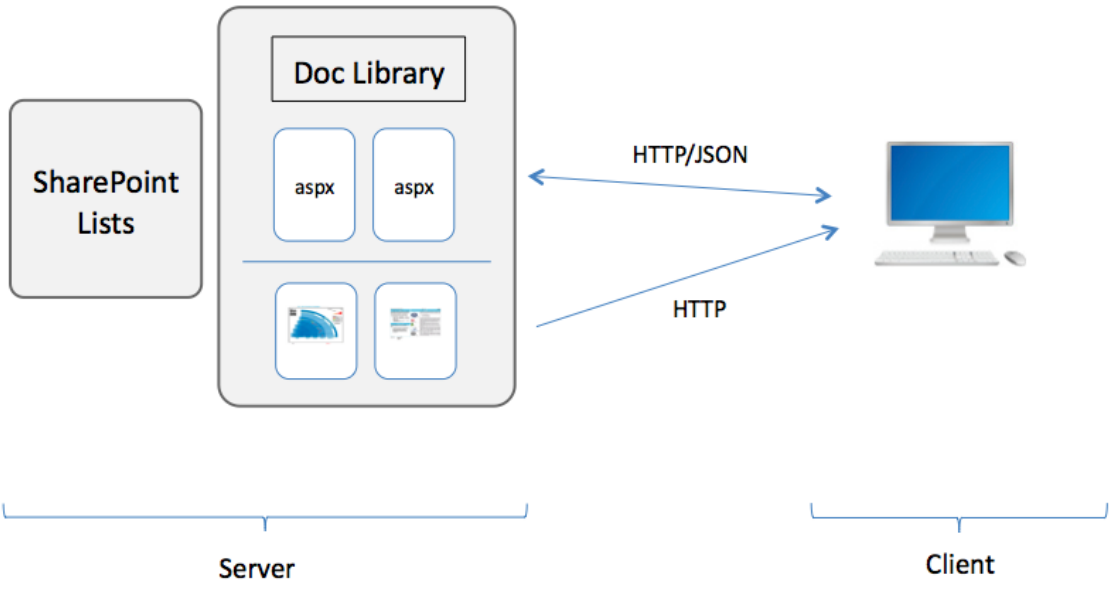
Therefore, server side data will be stored on SharePoint lists, and the programming part will be done in a .NET language like C#.

On the client side, it will consist of HTML combined with AJAX to make the page dynamic.

AJAX (Asynchronous JavaScript and XML) is a way of building web applications combining several technologies:

- DOM and JavaScript to modify the information shown by the navigator programmatically
- The XMLHttpRequest object to communicate asynchronously with the server
- XML or JSON to structure information running between client and server

The main interest of an AJAX development is located in the "asynchronous" part: it means that you can update the information displayed on the web browser without having to refresh the whole page which makes the user experience far better than with classic HTML or JavaScript: The user will be able to move a point on the interface, and **with no refresh**, the information in the lists server-side will be updated and parameters client-side as well.



Using SharePoint as the frame of the application also provides a solid right handling solution: SharePoint access can be connected to the Active Directory and therefore, with the help of the SharePoint object model, the web application rights management will be far easier to make than without SharePoint, and it will be reliable.

5. Prototyping

5.1. Technical context, plan

When the specifications were made, the development part could start. The material used to store it was a server running on Windows 2008 R2 associated with a Visual Studio 2010 and a SharePoint 2010 platform.

Here is the development plan that had been defined before the beginning of the development:

January 2010	Build the radar view page
January/February 2011	Start working with SharePoint. Define the data model and establish the communication between the view and the data contained in the SharePoint lists
February 2011	Build the technologies cards view page template
Mars/April 2011	Develop additional features like duplication and deletion of POT versions, wikis, and videos… and think about the semantic dimension of the application
May/June 2011	Develop the generation algorithm for the general radar and deploy the prototype of the application on the production environment after positive user evaluation

5.2. Development

All the development has been made using an agile method. This method is based on a high implication of the final user of the application during the development phase.

Factually, it has consisted of very frequent sessions held with eNovation team with two purposes:

- A description of the work performed since the last session associated with a practical demonstration of the current version of the software that enables an early detection of a divergence between the software and the ENovation expectations

- A discussion concerning the immediate plans concerning the development of the prototype to ensure that the development goes in the right direction

5.2.1. Radar view prototype (client side)

This step was the very first thing to perform and had the double purpose to get used to the POT while starting to develop the application. As we decided to keep the radar view of the PowerPoint, I have built an HTML page with JavaScript functions that were retrieving contents from a text file that contained all the information of the Excel technologies files.

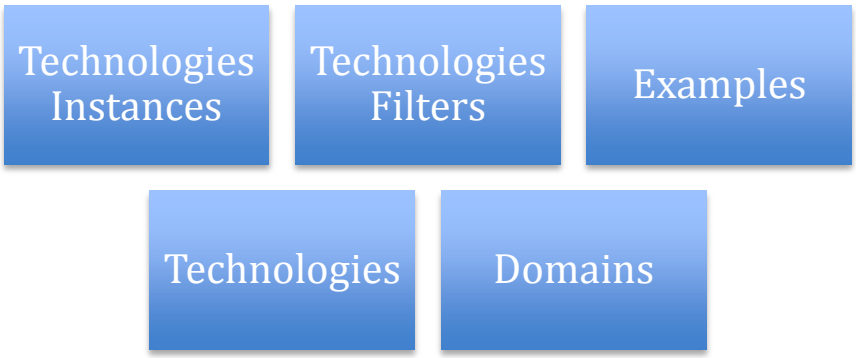
This HTML page was displaying the technologies on the screen using their coordinates.

This page is in fact the skeleton of the future application that will have to retrieve the information concerning the technologies directly from the SharePoint lists. For the moment, the page was only running client side, as JavaScript is a client side script, which is directly interpreted by the web browser.

In a further development step, the information received by the client will not be a simple text, but an XML file directly sent from the server, containing all the information concerning the technologies.

5.2.2. Definition of the data model

SharePoint structures all its information with lists. In SharePoint, everything is stored in lists. As it had been decided to avoid the use of an external database management system like SQL server, an appropriate data model had to be found, considering the functions that had to be supported by the web application. Here is the simplified data model, each square representing a list.



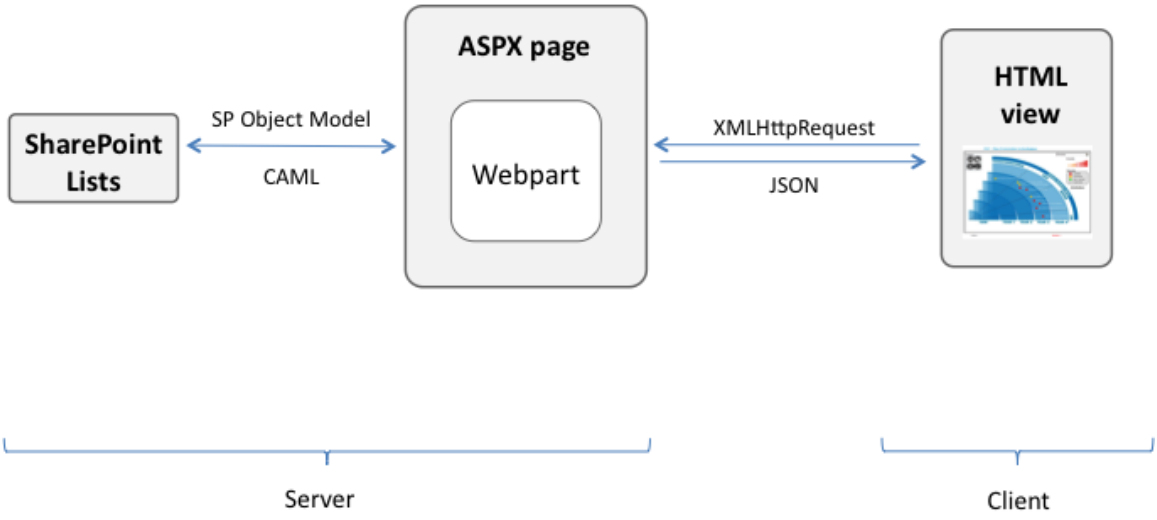
Technologies	Contains all the existing technologies at Total
Domains	Contains all the domains
Technologies Instances	This is the real place where technologies are stored: it will contain each technology instance with its position, maturity… For a given technology from *technologies* list, several instances can exist in *technologies instances* under different branch, year or version

	number
Technologies Filters	Contains all the parameters concerning the year, branch and version number existing in the web application
Examples	Contains all the examples of use of the technologies by each branch

A detailed data model is available in the annex, representing the link between the lists.

5.2.3. Connection between SharePoint lists and the HTML view

The next step is maybe the most important: How to establish an efficient communication between the client and the server?



To make the navigation in the application as comfortable as possible, an AJAX development based on the XMLHttpRequest object has been used.

This object is used for two major scenarios:

- The first one is to retrieve the SharePoint lists data to display it on the HTML page. Practically, what happens is that the HTML page will send an HTTP request to a webpart that has been developed on purpose and that is stored on the server. Then, this webpart will query the SharePoint lists using the SharePoint Object Model and it will send back to the client browser an XML file containing all the data asked. Once the client receives the XML file, a JavaScript function will parse it and add it to the DOM where all the nodes represent technologies and will have a maturity, priority… attribute.

- The second case happens when an update is made on the client browser, in order to update the SharePoint list. Practically, when an object is moved or modified on the client browser, a request is sent over HTTP using REST (the information is contained in the url), which will be

treated by a webpart that will update the SharePoint list using the SharePoint Object Model. Once the update has been made, a XML message is sent from the server to the client to confirm that the modification has been made.

The utilization of the XMLHttpRequest object enables the user to get a very interesting user experience, as all the communications that happened between the server and the client are asynchronous. Concretely, when a user does a modification, it is transmitted to the server that will update the SharePoint list and the DOM without requiring any refresh of the client page.

5.2.4. Technologies cards view

At this moment, the radar view prototype was done, and the development of the view of the technologies cards had to be done. Several possibilities were explored:

- Use a layout modified with SharePoint Designer[12]
- Develop an ASPX page that will directly query the lists in C# using ASP forms
- Develop a specific HTML page that will use the XMLHttpRequest object to query the lists through a webpart like what has been done previously for the radar view

The layout solution has been tried at first but it quickly appeared that there were too many limitations inherent with the fact that SharePoint Designer is designed for basic manipulations that does not require any programming effort.

At this point, there were two solutions left. One using the ASPX and direct server queries, the other using HTML and the XMLHttpRequest object. The HTML format was really efficient in the case of the radar view as many objects had to be manipulated, which is something the DOM can handle pretty well. In the case of the technologies cards, there is no need of direct modifications, requests sent to the server… in fact, the cards are just a more user-friendly view of the SharePoint lists, therefore, using ASPX pages that will query directly the lists in order to display their content appeared to be the best solution.

5.2.5. Additional features

Once the basic features of the POT were developed, additional features that were asked by the POT users had to be thought of and developed:

At first, a function that would enable the duplication and deletion of technologies instances was really necessary. For example, when the general POT is made available for every branch, they have to be able to duplicate this information before bringing their own branch modifications. To perform this function a webpart using the SOM[13] had to be developed.

The video functionality was not hard to develop, as it is a simple *object* tag inside the HTML page that point to the video location in SharePoint.

A point that turned out to be really time consuming was the specification of the Wiki. There were a lot of characteristics to define, both functionally and technically:

---

[12] Software used to bring modifications to SharePoint objects without any programming
[13] SharePoint Object Model

- Functionally, after several sessions, it has been determined that the Wiki will be open to every branch and everybody will be able to bring their modifications with no restriction. Then, at the end of the year, while creating the general POT of the year, eNovation members will take the Wiki content in consideration.

- Technically it has to be determined whether we would use the SharePoint integrated Wiki or a MediaWiki one which is open source. After several discussions, it has been decided to develop the Wiki with MediaWiki as it had far more functionalities than the SharePoint one.

The Wiki will be an exchange platform where everyone at Total will feel free to talk about technologies and their utilization. This tool is here to bring the community aspect that was missing on the PowerPoint POT. The Wiki will generate a dynamic around the POT which will benefit innovation.

5.2.6. General POT radar generation algorithm

The generation of the general POT is probably the essential purpose of the POT. All the efforts made during the year to complete the branches POT are focused on one aim: concluding about the global orientation of the group by building the general POT. This step was also the most time consuming steps of the previous POT. After each branch had built its own radar with its technologies, they had to find an agreement concerning the way they would build the general radar and the priority radar which are described as the following:

- **General radar**: This radar is a synthesis of all the branches radars, for each technology existing on the branch radars, a *general* technology will be created based on all the branches parameters

- **Priority radar**: This one is the same as the general radar but with a selection made on the technologies based on their priority. In addition to those technologies of utmost importance taken from the general radar, the ENovation department will be able to add a few technologies that they consider as being important to follow.

In order to write the algorithm to automatically generate those radars, ENovation explained the procedure they were using to define the general POT. This task was not that easy as they were not using any predefined rule during their discussion.

As the technologies cards are roughly the same between the different branches, the major part of the general POT generation concerns the radar. For some parameters like the maturity of a technology the procedure is easy to turn into an algorithm: they simply take as the *general* technology maturity the most advanced maturity among all branches.

```
generalMaturity = Max(maturityBranch(i)) for all i
```

The maturity is supposed to reflect the progress made on a technology in the company therefore it is natural that the overall progress corresponds to the progress made in the most advanced branch.

In fact, the conversion into an algorithm is easy as long as the parameter does not influence the position of the technology. To establish the position of the *general* technology, ENovation takes all the radar of all the branches, they compare them and determine an *average* position, depending on the importance of the branch but also on what they think about the importance of the technology (less easy to define).

This procedure had to be formalized and put into an algorithm.

How to position a given technology on the final radar programmatically?

The simplest way to do it was to perform a mere Cartesian average balanced by the importance of each branch. Obviously, this method is irrelevant as technologies are positioned on a polar scale. A mere balanced polar average was not relevant either: the radar is not a perfect circle.

Another issue that had to be considered while determining the coordinates of the final point was the following: The most important thing while thinking about the position of a point on the radar, after all, is not about its distance and angle coordinates, it is more about its impact and availability delay.

For example, a technology located at the frontier of two impacts in a branch, and at the frontier of two other impacts on another branch, may be wrongly positioned with an algorithm that would determine its exact average position, even considering the elliptic radar.

In the end, after some thinking, considering *only* the availability/impact discrete scale or *only* the distance/angle continuous scale led to mistakes as each one brings important information about the technology state. Moreover, no matter which deterministic way would be used, an ENovation member is never using such a mathematical rule to define the position of a technology on the final radar. Therefore, the solution that has been adopted was a compromise between the algorithm (using both scales) and the manual operation divided in four steps:

1. A rough approximation of the final coordinates is made using polar coordinates:

```
tempFinalDistance = [distanceBranch(i)*coeffBranch(i)]/sumCoeffs

tempFinalAngle = [distanceAngle(i)*coeffBranch(i)]/sumCoeffs
```

2. Considering all the branches impact coefficients and the impacts and availabilities of the technology in all the radars, the expected impact and availability of the *general* technology is calculated

```
expectedImpact = [impactBranch(i)*CoeffImpactBranch(i)]/sumCoeffs

expectedAvailability = [availabilityBranch(i)*CoeffAvailabilityBranch(i)]/sumCoeffs
```

3. An algorithm will iteratively move the technology positioned in step 1 until it belongs to the right dial. For instance, for the impact:

```
if (GetImpact(tempFinalDistance , tempFinalAngle ) > expectedImpact )
{
    tetaFinal = tetaFinal * 0.85;
}
else
{
    tetaFinal = tetaFinal * 1.15;
}
count++;
```

At first sight, this method seems to be very inaccurate, as the point will not necessarily be placed at the very average location. The fact is that this tool is not expected to build the final version of the general radar. Whatever deterministic method would be used, ENovation members would have restudied each point location because it is not an exact science, it is also a matter of feeling and intuition. Therefore, concerning the location, after those steps, the ENovation team will have the possibility to move manually any technology to make the global arrangement coherent with what they think of the technologic landscape at Total.

Priority is not taken into consideration in this final radar. All technologies, no matter what their priorities are, will be positioned on this general radar. The priority concern will be satiated in the priority radar. Again, a method had to be found in order to get roughly the same results as the ones obtained with the manual job performed on the old POT. After some discussions, it appeared that ENovation members were only selecting the technologies that were, according to them, the most present at a high degree in the other branches radars. To go in the same direction, two different algorithms performing that selection were studied:

- For a given technology, the algorithm counts the number of "high priority" iterations, then the N technologies having the highest number of "high priority" iterations would be displayed on the priority radar

- For a given technology, the algorithm will sum the priorities (high = 4, average = 3, middle = 2…) and then, the N technologies having the highest sum will be kept to appear on the priority radar

In both cases, a weighting can be used to rank the technologies according to their priorities and the branches they belong to.

In order to define the one to choose, a very pragmatic method has been used. A session has been organized with several branches radars that had to be merged manually by ENovation members into a general one, like they do every end of a year. In parallel, the two algorithms were run. The second algorithm was the one that presented the mosr similarities with the manual radar generated by ENovation members, therefore it is the algorithm that has been kept for the web application generation of the priority radar.

5.2.7. Semantic dimension

"The Semantic Web is not about links between web pages. The Semantic Web describes the relationships between things (like A is a part of B and Y is a member of Z) and the properties of things (like size, weight, age, and price)"[14]

The RDF[15] format is a way to describe those relationships on the web. In the case of the POT, here is what it looks like for the RFID:

```
<?xml version="1.0"?>

<rdf:RDF

<rdf:Description
rdf:about="http://81.255.213.111/sites/pot/Radar/fichesTechnologies.aspx?ID=RFID">
  <si:title>RFID</si:title>
  <si:maturity>Emerging</si:maturity>
</rdf:Description>

</rdf:RDF>
```

This format enables to build the relationships between the objects themselves, and not only between the pages.

Because of a lack of time, this aspect has been put aside and will be completed after as it sounds very promising.

5.2.8. User evaluation

When the prototype development was over, even if the agile development method ensured that the final user was satisfied with the application features and the way they were built, it was important to proceed to a real user evaluation period before deploying the application on the production environment. This evaluation has been divided in sessions were technical and non-technical people were trying out the application and using the features over and over while observational researches were made, followed by small interviews. It has been useful to discover some small problems in the interface and to help the documentation building as it is sometimes hard to take enough distance during the real development part.

During this step, it was clear that using an agile method all along the project has been a good idea: no major bug was experienced, and no one found a gap between what they were expecting concerning the application and what it was actually doing.

5.2.9. Deployment of the application in the production environment

When everybody approved the application, the application had to be deployed on the production environment.

This part has been divided in two different steps.

---

[14] *Semantic Web Tutorial*, http://www.w3schools.com/semweb/default.asp, last visited on June 13th, 2011

[15] The Resource Description Framework (RDF) is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model.

- The first one was the deployment on the SharePoint 2010 server under a different site collection so that the production environment was separated from the test environment and so that branches could start working with the POT web application even if it was not a definitive solution.

- The second one was the real deployment on the production environment. Finally, the last SharePoint version was not available on the production server; the only possibility was to run the application on a SharePoint 2007 server. This complicated the task a lot, as some modifications had to be brought to the initial program but also because deploying with SharePoint 2010 is far easier and automated than deploying with SharePoint 2007. This solution is in constant evolution and what was done in ten steps in the 2007 version is now done in a single one.

The first step was pretty quick to perform. A new site collection has been created, and then the program was deployed on the production environment. Then, the difficulty was to maintain both versions updated.

The second step was far harder and it induced some modifications in the program. Most of them were only laborious. Lists definitions, fields definitions… One of the differences existing between SharePoint 2010 and SharePoint 2007 concerns the deployment part and the development of SharePoint objects, which were mostly automatic with the last version whereas a lot of manual interventions had to be made with the previous one. For instance, to deploy, the CAB file has to be built manually by declaring all the features and files included inside, and then, placed at the right place on the server:

- In the BIN of the application, to restrict it to a single web application
- In the GAC, so that all the web applications can use it

Moreover, the user rights management was hard to convert with the 2007 interface: SharePoint 2010 folders are pretty useful to group the technology instances and to give them specific rights. With SharePoint 2007 list definitions, it is not that easy to put items into folders; therefore event receivers had to be built in order to give the rights out to each technology.

5.3. Problems encountered

During the development, as SharePoint is not an easy solution to manipulate, especially if the use goes further the borders of the file management solution, many solutions had to be found in order to make everything working together efficiently. Originally, SharePoint was a file management system. Even if the solution broaden a lot its possibilities, it still remains complicated to do something really different than mere file management with it, and the web application has necessitated some compromises and bypasses.

5.3.1. Non relational list model

The first thing to perform while using SharePoint for the application was to create the data model. The data structure of the web application is pretty simple, as not so many data has to be handled by the software. Unfortunately, SharePoint lists are not built according to a relational model, which means that the lists are not really made to have links between them.

In order to circle this, a SharePoint kind of field can be used: the *lookup* field. In a given list, you can add a lookup field that will be able to take the value of a given field in another list. Therefore, in the list **technology instances**, a lookup field will take the value of the *name* field of **technologies** to associate the instance with a technology.

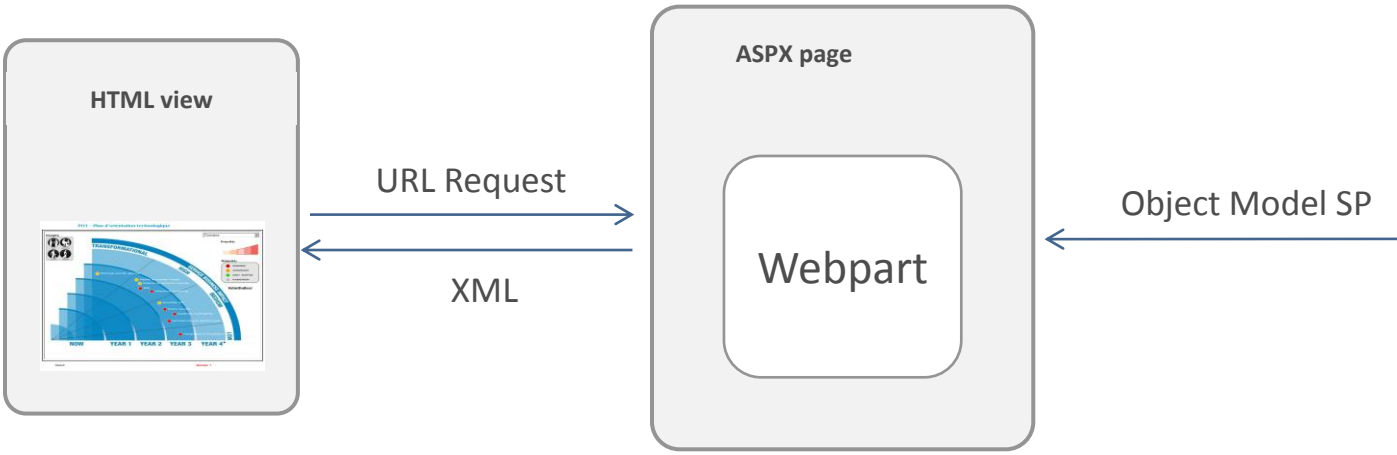
This trick does not create a real jointure, as you will not be able to request on the jointure. First, you will have to get the value of the field in one list and then, you will make the request with that value on the second list in order to get the other parameter value of the second list that you are interested in.

In the end, the only possibility that avoids these additional steps was be to use a real database management system like SQL server, but considering the little complexity of the data model, it is really not worthwhile.

5.3.2. Slowness of the web service REST

The link between SharePoint lists and the client browser is important as it is used at each parameter update. Microsoft provides an interesting tool, which is a web service REST[16] that enables to retrieve an XML file from a URL call. Unfortunately, this tool did not produce satisfying results as the request was pretty slow.

To get better results, a webpart has been developed to perform the exact same result:



The webpart receives the request from URL parameters sent by the client, then it requests to the list using the SharePoint Object Model before sending everything back to the client browser using an XML file. In fact the web service has been recreated.

Task/Solution	Web service REST	Webpart developed
Load 1 radar	984 ms	356 ms
Load 2 radars	1235 ms	495 ms

[16] Representational State Transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web

5.3.3. User rights management

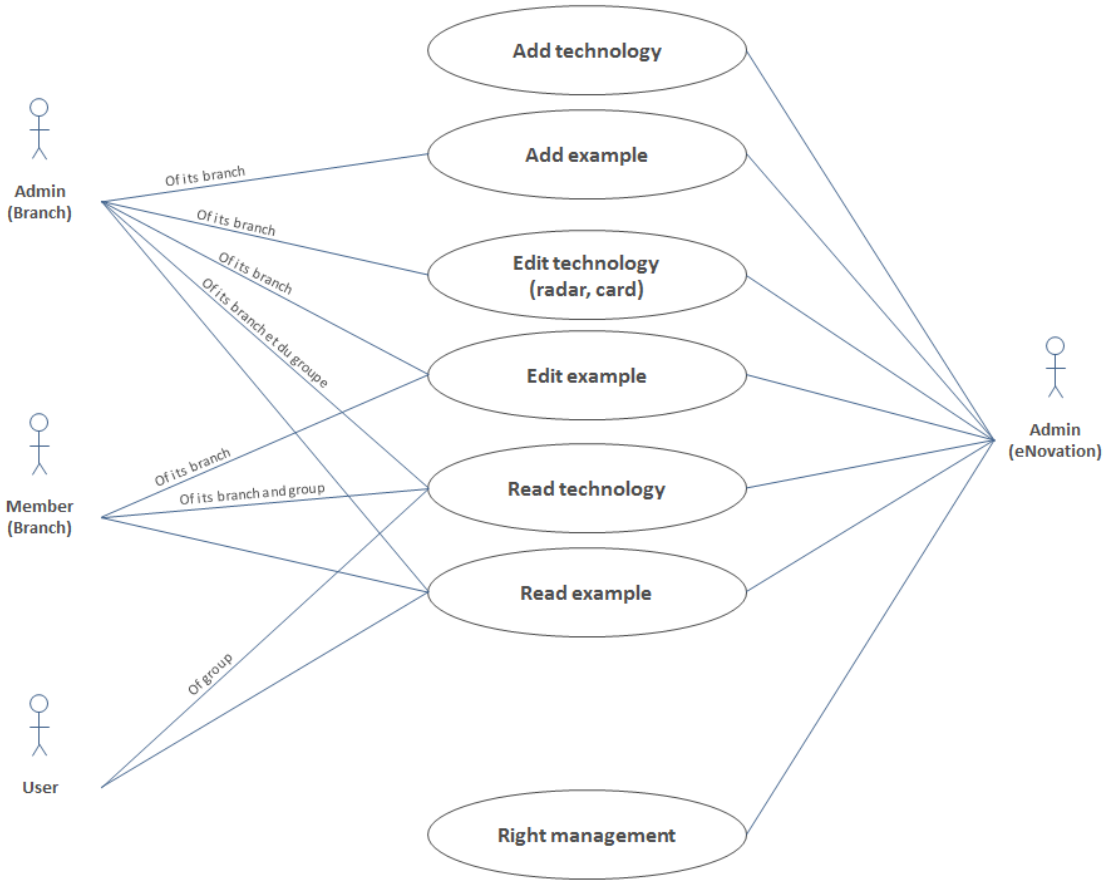
SharePoint is not made to handle item rights individually. The structure of a SharePoint web site allows you to give specific rights to each individual list, but different rights cannot be given to different items in the list. Fortunately, a bypass exists as specific rights can always be granted to folders. Therefore, the solution consists to position folders at the root of the list which will have different rights, and then to put the items in the folder corresponding to the right you want to give. In the case of the application the folders were named after each branch name, so that specific rights could be given according to the branch of the user.

6. Final prototype study

After a few months of development, the application was running on the production environment. Most of the functional requirements were fulfilled except certain functionalities such as the PDF exportation feature and the semantic search, which was hard to build because of a lack of time.

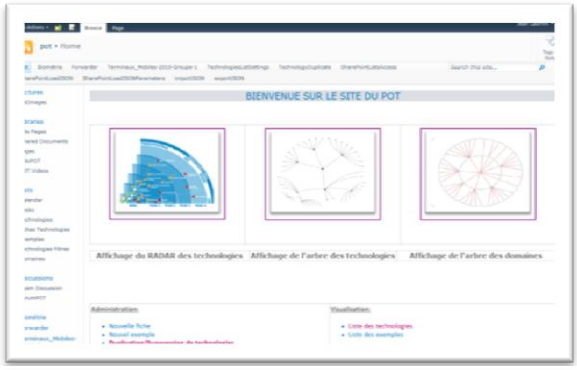
6.1. Use cases

Here is a quick overview of the web application use cases.



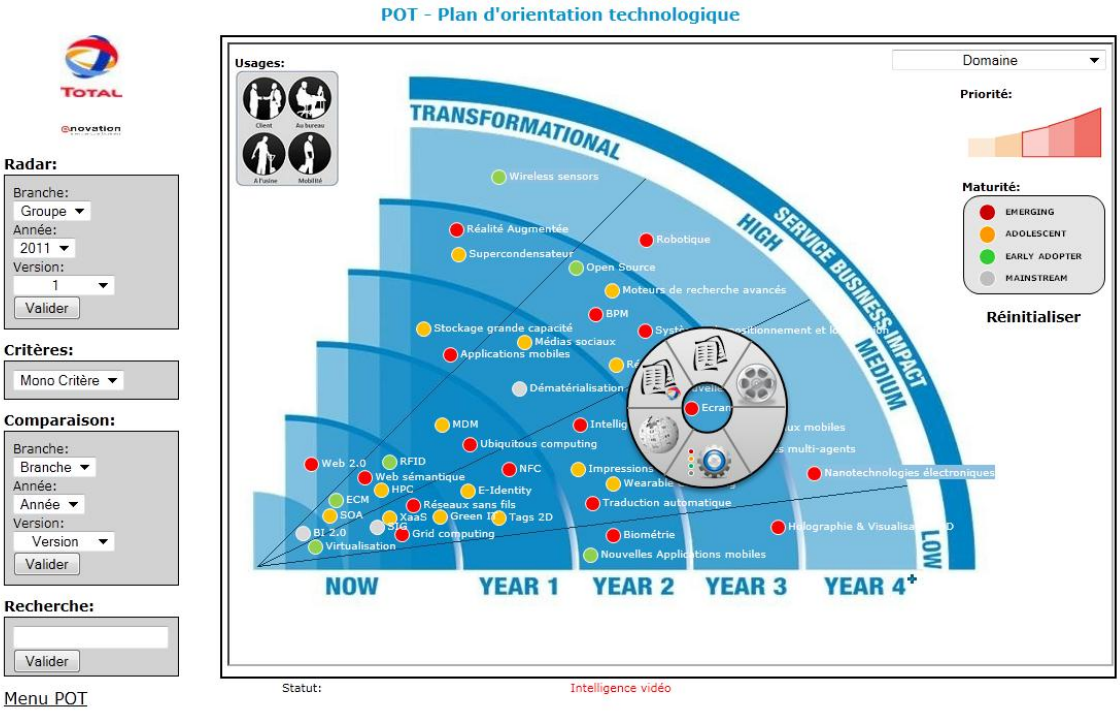
6.2. Interface

The web application is divided into a central menu (giving access to the SharePoint settings, lists and administration pages developed during the project) and the radar (giving access to the technology cards)



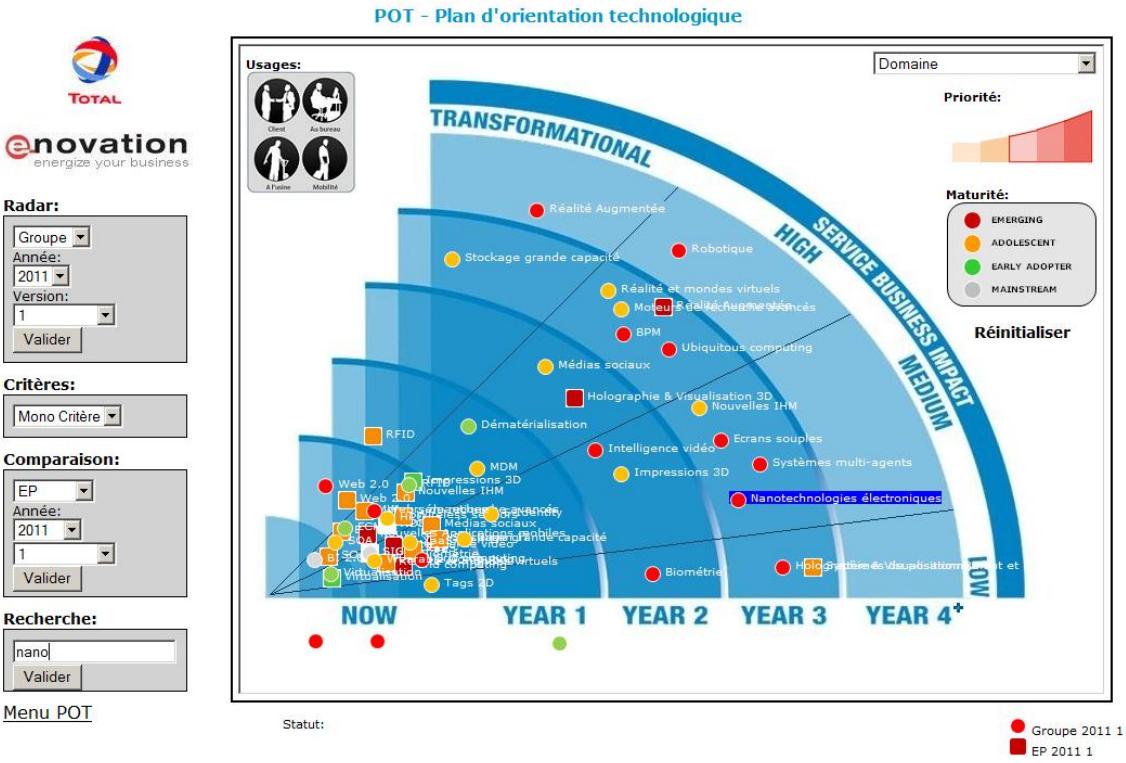
Central menu from where an easy access is provided to the radar

The radar of the technologies displaying a set of technology instances



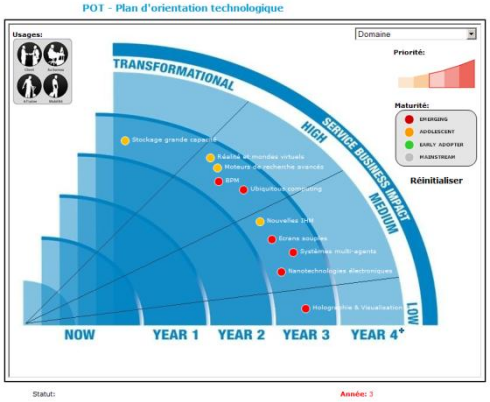
On the left-hand side of the radar, a menu enables the user to select the set of instances that he or she wants to display. Furthermore, there is an access to the radar comparison mode and to the search feature.
Here, the user has clicked on *intelligence video* thus displaying the palette giving access to the technology information/modifications.

6.3. Comparison feature

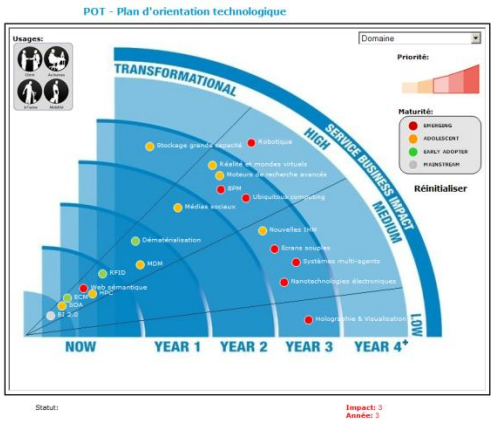


This is an example of the comparison feature; here there a comparison is made between the **Group 2011 vers 1** instance and the **EP 2011 vers 1** instance. A search has also been performed on the *nano* keyword.

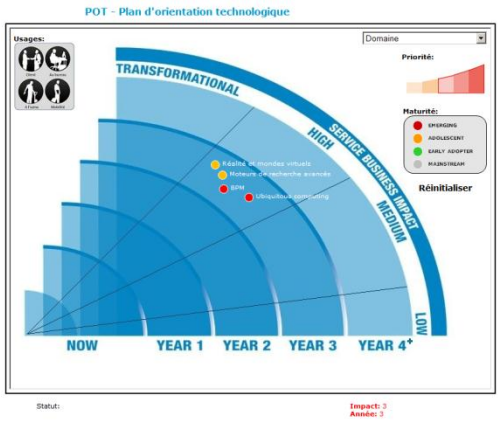
6.4. Selective display



Clicking on *year 3* on the radar will display only the technologies whose availability is year 3.

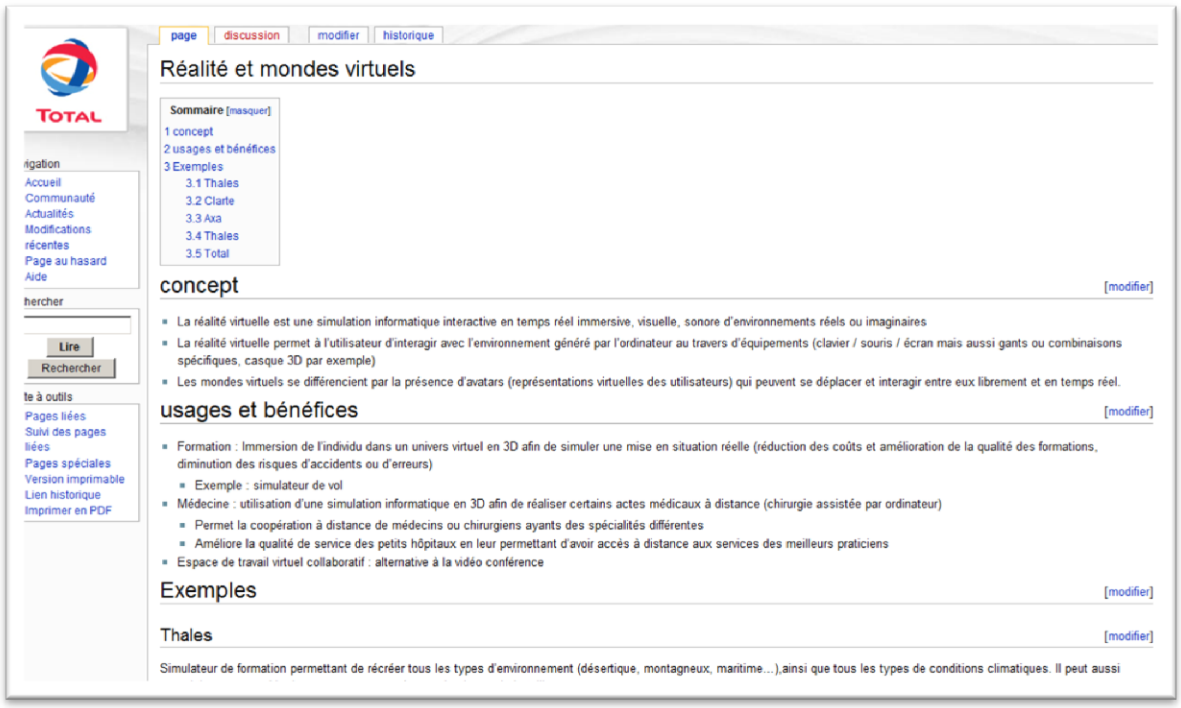


Here, the display mode is OR, therefore, by clicking on *year 3* and then *impact high*, technologies presenting one of those parameters are displayed



Here, the display mode is AND, therefore, by clicking on *year 3* and then *impact high*, technologies presenting both parameters are displayed

6.5. Wiki platform



This is the wiki side of the POT. All the technologies displayed on the radar have their wiki page on which everyone can add content. This page will be used by eNovation members to update the card description at the end of each year.

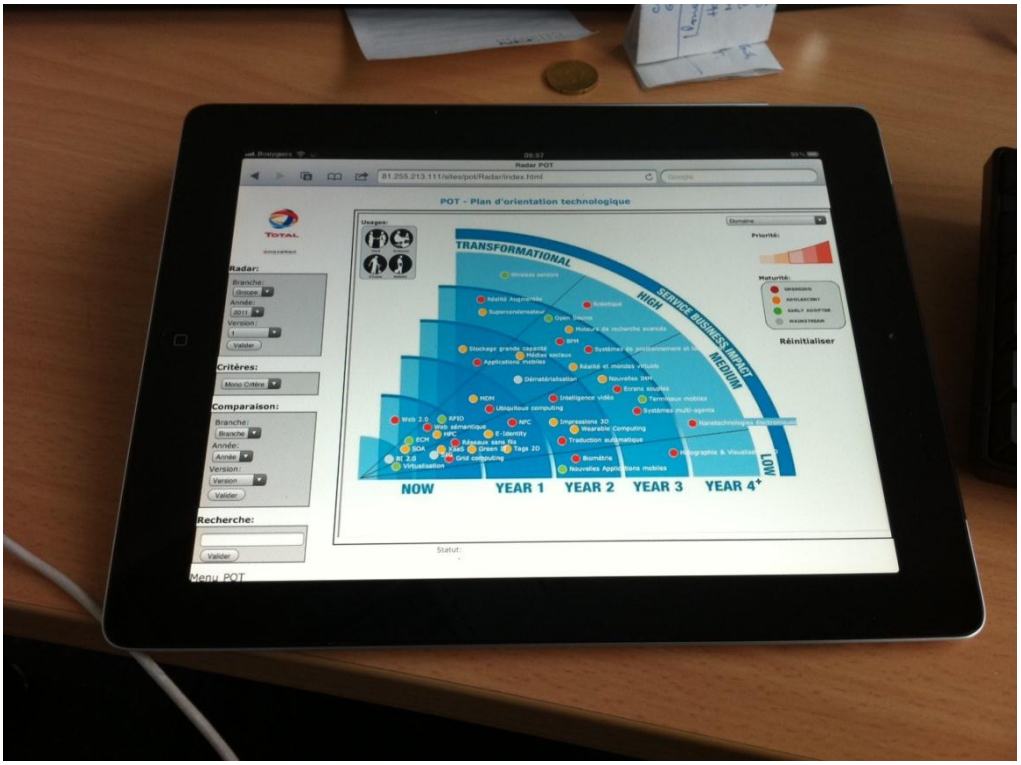
6.6. User rights management

Concerning the right management, according to the SharePoint group a user belongs to, the interface will be modified. First of all, the dropdown list on the right menu will adapt to the radar the user can read. Subsequently, the edition possibility will be displayed only if the user has the rights to do it.

In fact, those interface modifications are not made to restrict the move, but to give a better user experience: as SharePoint handles the rights, it is anyway impossible for the user to go over it. Therefore, if the user interface was not adapted, a non-allowed modification will just lead to an error message.

6.7. Mobile version

The web application has been built with a permanent compatibility concern, especially with mobile devices such as the iPhone or the iPad as they are used a lot at Total and because of the many applications that can be done especially to communicate about the technologies radar.



iPad running the web application

6.8. Conclusion about the prototype

To conclude with the prototype of the application, the main idea is that the application is already working on the production environment; all the features that have been proven to be safe have been kept. Unfortunately, some of them are lacking like the PDF exportation and semantic features, as they are not fully functional.

The main purposes have been fulfilled as the application provides administrative, collaborative and analysis features. Another thing that can be mentioned is that the application is really intuitive and no manual is required to use it.

7. Discussion and conclusion

Numerical technologies are now everywhere and it is becoming more and more important to share about them to gain efficiency in big companies. Thus, dynamic action can be generated around a given technology, encouraging people involved in a technology to talk with each other.

The application has changed many things in the way eNovation was communicating about Total technological orientation. They now spend far less time on manual steps or technical details than before and they can focus on the real subject of the technologies. The benefit of the application is most obvious concerning the general POT generation where people will appreciate to gather around an iPad, discussing the automatically generated technologies of the general radar.

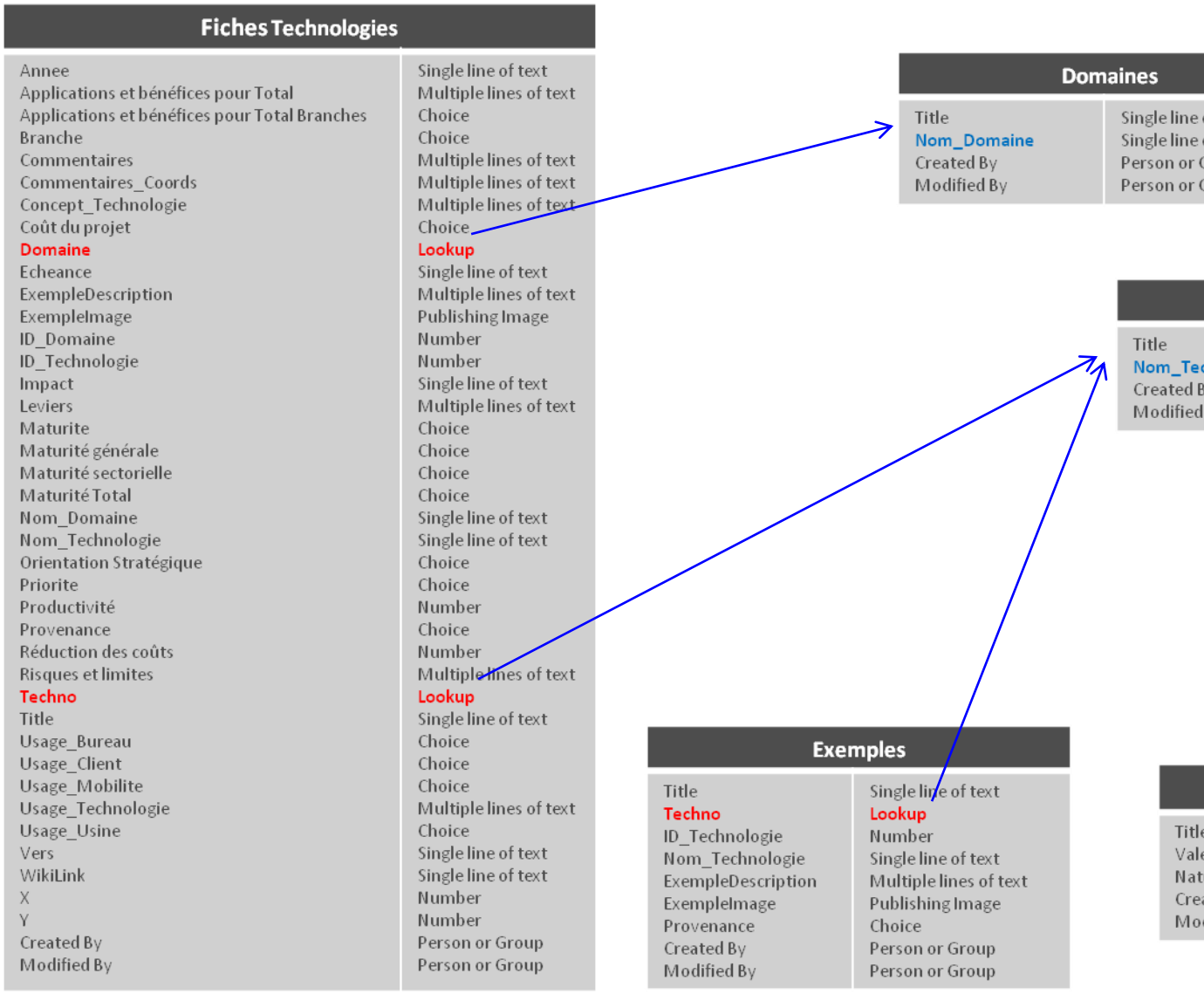
This is something really interesting, which was not really anticipated while starting to develop the web application. The main objective of the application when the development started was to provide a collaborative tool with advanced functionalities that would enable automation of some manual steps in order to avoid a lot of wasted time. The compatibility with a mobile platform was a simple requirement. Now that the application is running on computers and mobile devices, all the benefit of the mobile utilization is revealing itself: in a meeting, nothing is more efficient than turning the tablet on, opening the POT and starting to use the application, showing the orientation of the branch and making some comparisons through the interface directly in front of people that can also manipulate.

This master thesis also enforced the idea that a good method was necessary to perform a project in good conditions and to avoid bad consequences of last minute changes as during the development the initial specifications have changed many times. It is impossible to avoid the fact that prior to have the application in their hands, people may give specifications that does not fit with what they really want, just because one cannot always figure out accurately what a specification will look like in practice. In that sense, using agile methods appear to be really necessary, and it saved a lot of time in the case of the web application project.

Total is going to pursue the development of the web application as they now have the certitude that this tool is going to save them a lot of time and that the communication possibilities, especially through the use of mobile devices, are huge. The collaborative area around the POT has to be explored, for the moment the Wiki is at an experimental state but if it grows enough it can represent an interesting subsidiary platform to the POT where people will be able to express themselves in an open environment where everyone can bring their knowledge.

8. Appendix

8.1. Data model



8.2. Final radar generation webpart code

```
using System;
using System.Web.UI;
using System.Web.UI.WebControls;
```

```
using System.Web.UI.WebControls.WebParts;
using Microsoft.SharePoint;
using System.Collections.Generic;
using System.ComponentModel;
using System.Web;
using Microsoft.SharePoint.WebControls;
using System.Text.RegularExpressions;
using System.Text;
using System.Windows.Forms;

namespace Total.DSIT.POT.generateFinalRadar
{
    public partial class generateFinalRadarUserControl : System.Web.UI.UserControl
    {
        CheckBoxList checklist;
        DropDownList radarInitialBranch;
        DropDownList radarInitialYear;
        DropDownList radarInitialVers;
        DropDownList radarGeneralBranch;
        DropDownList radarGeneralYear;

        // Initialisation des coefficients normalement récupérés à partir de la liste
Sharepoint
        double coeffImpactHolding = 1;
        double coeffImpactEP = 1;
        double coeffImpactRM = 1;
        double coeffImpactPCH = 1;
        double coeffImpactTS = 1;
        double coeffImpactCH = 1;
        double coeffImpactGEN = 1;
        double coeffImpactGroupe = 1;
        double coeffPrioHolding = 1;
        double coeffPrioEP = 1;
        double coeffPrioRM = 1;
        double coeffPrioPCH = 1;
        double coeffPrioTS = 1;
        double coeffPrioCH = 1;
        double coeffPrioGEN = 1;
        double coeffPrioGroupe = 1;

        protected void Page_Load(object sender, EventArgs e)
        {

            this.Controls.Add(new LiteralControl("<font align=center face=Calibri
size=5>Bienvenue sur la page de génération du radar final et du radar des
priorités.</font>"));
            this.Controls.Add(new LiteralControl("<br/><br/>"));
            this.Controls.Add(new LiteralControl("<font face=Calibri size=3><b>1)
Selectionner les radars à utiliser pour la génération:</b></font>"));
            this.Controls.Add(new LiteralControl("<b></b>"));
            this.Controls.Add(new LiteralControl("<br/><br/>"));

            checklist = new CheckBoxList();
            this.Controls.Add(checklist);

            this.Controls.Add(new LiteralControl("<br/><br/><br/>"));

            this.Controls.Add(new LiteralControl("<font face=Calibri size=3><b>2) Générer
le radar:</b></font>"));
            this.Controls.Add(new LiteralControl("<br/><br/>"));
            this.Controls.Add(new LiteralControl("<b><u>FINAL</u></b>"));
            this.Controls.Add(new LiteralControl("<br/><br/>"));
            this.Controls.Add(new LiteralControl("Radar de départ:"));

            radarInitialBranch = new DropDownList();
            radarInitialYear = new DropDownList();
            radarInitialVers = new DropDownList();

            System.Web.UI.WebControls.Button submitRadars = new
System.Web.UI.WebControls.Button();
            submitRadars.Click += new EventHandler(submitRadars_click);
            submitRadars.Text = "Générer";

            this.Controls.Add(radarInitialBranch);
            this.Controls.Add(radarInitialYear);
            this.Controls.Add(radarInitialVers);
            this.Controls.Add(submitRadars);

            this.Controls.Add(new LiteralControl("<br/><br/><br/>"));
            this.Controls.Add(new LiteralControl("<b><u>PRIORITES</u></b>"));
```

```
            this.Controls.Add(new LiteralControl("<br/><br/>"));
            this.Controls.Add(new LiteralControl("Radar général de départ:"));

            radarGeneralBranch = new DropDownList();
            radarGeneralYear = new DropDownList();

            System.Web.UI.WebControls.Button submitRadarsPrios = new
System.Web.UI.WebControls.Button();
            submitRadarsPrios.Click += new EventHandler(submitRadarsPrios_click);
            submitRadarsPrios.Text = "Générer";

            this.Controls.Add(radarGeneralBranch);
            this.Controls.Add(radarGeneralYear);
            this.Controls.Add(submitRadarsPrios);

            this.Controls.Add(new LiteralControl("<br />"));

            // Remplit les listes de la page d'administration
            if (!IsPostBack)
            {
                Populate("Branch");
                Populate("Year");
                Populate("Version");

                List<string[]> radarList = generateRadarList();
                for (int i = 0; i < radarList.Count; i++)
                {
                    checklist.Items.Add(new ListItem((radarList[i])[0] + "-" +
(radarList[i])[1] + "-" + (radarList[i])[2]));
                }
            }

            // Définition des propriétés de la checkbox list des radars
            checklist.AutoPostBack = false;
            checklist.CellPadding = 5;
            checklist.CellSpacing = 5;
            checklist.RepeatDirection = RepeatDirection.Vertical;
            checklist.RepeatLayout = RepeatLayout.Flow;
            checklist.TextAlign = TextAlign.Right;
        }

        // Clic sur générer le radar final
        protected void submitRadars_click(object sender, EventArgs e)
        {
            populateCoeffs();
            ListItemCollection items = checklist.Items;
            List<string> radarToDisplay = new List<string>();
            foreach (ListItem item in items)
            {
                if (item.Selected)
                {
                    radarToDisplay.Add(item.Text);
                }
            }
            if (radarToDisplay.Count > 0)
            {
                generateFinalRadar(radarToDisplay);
            }
            else
            {
                //TODO: problème messagebox
                //MessageBox.Show("Sélectionnez au moins un radar pour générer le radar
final");
            }

        }

        // Clic sur générer le radar des priorités
        protected void submitRadarsPrios_click(object sender, EventArgs e)
        {
            populateCoeffs();
            ListItemCollection items = checklist.Items;
            List<string> radarToDisplay = new List<string>();
            foreach (ListItem item in items)
            {
                if (item.Selected)
                {
                    radarToDisplay.Add(item.Text);
                }
            }
```

```
            if (radarToDisplay.Count > 0)
            {
                generatePriorityRadar(radarToDisplay);
            }
            else
            {
                //MessageBox.Show("Sélectionnez au moins un radar pour générer le radar
des priorités");
            }

        }

        // Méthode qui alimente les coefficients avec leur valeur de la lsite
technologies filtres
        protected void populateCoeffs()
        {
            using (SPWeb web = new SPSite(SPContext.Current.Web.Url).OpenWeb())
            {
                SPList list = web.Lists[Const.TechnologiesFiltres];
                foreach (SPListItem item in list.Items)
                {
                    switch (item["Nature"].ToString())
                    {
                        case "pondImpactHolding":
                            coeffImpactHolding =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactEP":
                            coeffImpactEP =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactRM":
                            coeffImpactRM =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactPCH":
                            coeffImpactPCH =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactTS":
                            coeffImpactTS =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactCH":
                            coeffImpactCH =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactGEN":
                            coeffImpactGEN =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondImpactGroupe":
                            coeffImpactGroupe =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioHolding":
                            coeffPrioHolding =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioEP":
                            coeffPrioEP =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioRM":
                            coeffPrioRM =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioPCH":
                            coeffPrioPCH =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioTS":
                            coeffPrioTS =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioCH":
                            coeffPrioCH =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
```

```
                        case "pondPrioGEN":
                            coeffPrioGEN =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                        case "pondPrioGroupe":
                            coeffPrioGroupe =
Convert.ToDouble(item[Const.Valeur].ToString());
                            break;
                    }
                }
            }
        }

        // Méthode qui permet de générer le radar final à partir des radars cochés
        protected void generateFinalRadar(List<string> radarToDisplay)
        {
            using (SPWeb web = new SPSite(SPContext.Current.Web.Url).OpenWeb())
            {
                web.AllowUnsafeUpdates = true;
                web.Site.CatchAccessDeniedException = false;
                string liste = Const.FichesTechnologie;

                // Caracteristiques du radar de départ, sélectionné dans la liste
déroulante
                string branch = radarInitialBranch.Text;
                string year = radarInitialYear.Text;
                string vers = radarInitialVers.Text;

                // Radar d'arrivée
                string newYear = radarInitialYear.Text;
                string newBranch = "Groupe";
                string newVers = "general";

                DeleteItems(newYear, newVers, newBranch, web);

                SPList list = web.Lists[liste];
                SPQuery query = new SPQuery(list.Views[Const.allBranchesView]);
                StringBuilder viewFields = new StringBuilder();
                foreach (SPField field in list.Fields)
                    viewFields.AppendFormat("<FieldRef Name=\"{0}\"/>",
field.InternalName);

                query.ViewFields = viewFields.ToString();
                query.RowLimit = 1000;

                query.Query = "<Where><And><And><Eq><FieldRef Name='" +
Const.brancheTechnologie + "'/>" +
                    "<Value Type='Text'>" + branch + "</Value></Eq><Eq><FieldRef Name='"
+ Const.yearTechnologie + "'/>" +
                    "<Value Type='Text'>" + year + "</Value></Eq></And><Eq><FieldRef
Name='" + Const.versionTechnologie + "'/>" +
                    "<Value Type='Text'>" + vers + "</Value></Eq></And></Where>";

                SPListItemCollection items = list.GetItems(query);

                // Pour chaque item, on va d'abord faire une simple copie, puis on mettra
ses paramètres à jour
                foreach (SPListItem item in items)
                {
                    SPListItem myNewItem =
list.Items.Add(string.Format("{0}/Lists/{1}/{2}", web.Url, list.Title, newBranch),
SPFileSystemObjectType.File, null);

                    // Duplication des champs
                    foreach (SPField field in list.Fields)
                    {
                        if (field.CanBeDisplayedInEditForm)
                        {
                            myNewItem[field.InternalName] = item[field.InternalName];
                        }
                    }

                    myNewItem[Const.yearTechnologie] = newYear;
                    myNewItem[Const.brancheTechnologie] = newBranch;
                    myNewItem[Const.versionTechnologie] = newVers;
                    myNewItem[Const.commentairesTechnologie] = "";
                    myNewItem[Const.commentairesCoordsTechnologie] = "";

                    // Traitement des nouvelles coordonnées
                    myNewItem.Update();
```

```
                    updateParameters(web, list, myNewItem, year, vers, radarToDisplay);

                }
            }
        }

        // Méthode qui permet de générer le radar des priorités à partir des radars
cochés
        protected void generatePriorityRadar(List<string> radarToDisplay)
        {
            using (SPWeb web = new SPSite(SPContext.Current.Web.Url).OpenWeb())
            {
                web.AllowUnsafeUpdates = true;
                web.Site.CatchAccessDeniedException = false;
                string liste = Const.FichesTechnologie;
                string branch = radarGeneralBranch.Text;
                string year = radarGeneralYear.Text;
                //string vers = radarGeneralVers.Text;
                string vers = "general";

                string newYear = year;
                string newBranch = "Groupe";
                string newVers = "generalPrio";

                DeleteItems(newYear, newVers, newBranch, web);

                SPList list = web.Lists[liste];
                SPQuery query = new SPQuery(list.Views[Const.allBranchesView]);
                StringBuilder viewFields = new StringBuilder();
                foreach (SPField field in list.Fields)
                    viewFields.AppendFormat("<FieldRef Name=\"{0}\"/>",
field.InternalName);

                query.ViewFields = viewFields.ToString();
                query.RowLimit = 1000;

                query.Query = "<Where><And><And><Eq><FieldRef Name='" +
Const.brancheTechnologie + "'/>" +
                    "<Value Type='Text'>" + branch + "</Value></Eq><Eq><FieldRef Name='"
+ Const.yearTechnologie + "'/>" +
                    "<Value Type='Text'>" + year + "</Value></Eq></And><Eq><FieldRef
Name='" + Const.versionTechnologie + "'/>" +
                    "<Value Type='Text'>" + vers + "</Value></Eq></And></Where>";

                SPListItemCollection items = list.GetItems(query);

                foreach (SPListItem item in items)
                {
                    SPListItem myNewItem =
list.Items.Add(string.Format("{0}/Lists/{1}/{2}", web.Url, list.Title, newBranch),
SPFileSystemObjectType.File, null);

                    // Duplication des champs
                    foreach (SPField field in list.Fields)
                    {
                        if (field.CanBeDisplayedInEditForm)
                        {
                            myNewItem[field.InternalName] = item[field.InternalName];
                        }
                    }

                    myNewItem[Const.yearTechnologie] = newYear;
                    myNewItem[Const.brancheTechnologie] = newBranch;
                    myNewItem[Const.versionTechnologie] = newVers;
                    myNewItem[Const.commentairesTechnologie] = "";
                    myNewItem[Const.commentairesCoordsTechnologie] = "";

                    // Traitement des nouvelles coordonnées
                    myNewItem.Update();
                }

                // Après avoir mis à jour toutes les coordonnées et les paramètres, on ne
garde que les plus hautes priorités
                DeleteLeastPrioItems(newYear, newVers, newBranch, web, radarToDisplay);
            }
        }

        // Mise à jour des paramètres des technologies du radar final en fonction de
celles des radars sélectionnés
        protected void updateParameters(SPWeb web, SPList list, SPListItem newItem,
```

```
string year, string vers, List<string> radarToDisplay)
        {
            string nomTechno = newItem[Const.nomTechnologie].ToString();

            double rohFinal = 0;
            double tetaFinal = 0;
            double xFinal = 0;
            double yFinal = 0;
            double tetaSumPond = 0;
            double impactSumPond = 0;
            double availabilitySumPond = 0;
            double pondImpactSum = 0;
            double pondMaturitySum = 0;
            double pondPrioSum = 0;
            double prioriteSumPond = 0;
            double maturiteSumPond = 0;
            double impactDesired = 0;
            double availabilityDesired = 0;
            int prioriteFinalTemp = -5;
            string prioriteFinal = "";
            int maturiteFinalTemp = 1;
            string maturiteFinal = "Emerging";
            double rohSumPond = 0;

            // Listes qui vont stocker les paramètres des instances d'une technologie
            List<double> rohTable = new List<double>();
            List<double> tetaTable = new List<double>();
            List<double> pondPrioTable = new List<double>();
            List<double> pondMaturityTable = new List<double>();
            List<double> pondImpactTable = new List<double>();
            List<double> maturiteTable = new List<double>();
            List<double> prioriteTable = new List<double>();
            List<int> impactTable = new List<int>();
            List<int> availabilityTable = new List<int>();

            // Création de la query qui va utiliser la vue "allbranches"
            SPQuery query = new SPQuery(list.Views[Const.allBranchesView]);
            StringBuilder viewFields = new StringBuilder();
            foreach (SPField field in list.Fields)
                viewFields.AppendFormat("<FieldRef Name=\"{0}\"/>", field.InternalName);
            query.ViewFields = viewFields.ToString();
            query.RowLimit = 1000;

            query.Query = "<Where><Eq><FieldRef Name='" + Const.nomTechnologie + "'/>" +
                "<Value Type='Text'>" + nomTechno + "</Value></Eq></Where>";

            SPListItemCollection items = list.GetItems(query);

            // On parcourt chaque instance d'une technologie pour stocker les valeur de
paramètres
            foreach (SPListItem item in items)
            {
                string radar = item[Const.brancheTechnologie].ToString() + "-" +
item[Const.yearTechnologie].ToString() + "-" + item[Const.versionTechnologie].ToString();
                if (radarToDisplay.Contains(radar))
                {
                    string provenance = item[Const.provenance].ToString();

                    double x = (double)item[Const.xTechnologie];
                    double y = (double)item[Const.yTechnologie];
                    int impact = Int16.Parse(item[Const.impactTechnologie].ToString());
                    int availability =
Int16.Parse(item[Const.echeanceTechnologie].ToString());
                    string priorityTemp = item[Const.prioriteTechnologie].ToString();
                    string maturityTemp = item[Const.maturiteTechnologie].ToString();
                    int priority;
                    int maturity;

                    // Conversion de la priorité en int
                    switch (priorityTemp)
                    {
                        case "Sous surveillance":
                            priority = Const.pondPrioriteSousSurveillance;
                            break;
                        case "Autre":
                            priority = Const.pondPrioriteAutre;
                            break;
                        case "Non-prioritaire":
                            priority = Const.pondPrioriteNonPrio;
                            break;
                        case "Suiveur":
```

```
                            priority = Const.pondPrioriteSuiveur;
                            break;
                        case "Faible":
                            priority = Const.pondPrioriteBasse;
                            break;
                        case "Moyenne":
                            priority = Const.pondPrioriteMoyenne;
                            break;
                        case "Haute":
                            priority = Const.pondPrioriteHaute;
                            break;
                        default:
                            priority = -1;
                            break;
                    }

                    // Conversion de la maturité en int
                    switch (maturityTemp)
                    {
                        case "Mainstream":
                            maturity = Const.pondMaturiteMainstream;
                            break;
                        case "Early adopters":
                            maturity = Const.pondMaturiteEarlyAdopters;
                            break;
                        case "Adolescent":
                            maturity = Const.pondMaturiteAdolescent;
                            break;
                        case "Emerging":
                            maturity = Const.pondMaturiteEmerging;
                            break;
                        default:
                            maturity = Const.pondMaturiteEmerging;
                            break;
                    }

                    // Passage des coordonées cartésiennes aux polaires
                    double teta = Math.Atan((double)y / x);
                    double roh = (double)x / Math.Cos(teta);

                    // Si la technologie ne figure pas sous le radar, on la prend en
compte
                    if (y > -1)
                    {

                        // Coordonnées
                        rohTable.Add(roh);
                        tetaTable.Add(teta);

                        // Cadrans
                        impactTable.Add(impact);
                        availabilityTable.Add(availability);

                        // Priorité et maturité
                        prioriteTable.Add(priority);
                        maturiteTable.Add(maturity);

                        // On remplit le tableau des pondérations pour chaque paramètre
                        switch (provenance)
                        {
                            case "EP":
                                pondImpactTable.Add(coeffImpactEP * priority);
                                pondMaturityTable.Add(coeffImpactEP);
                                pondPrioTable.Add(coeffPrioEP);
                                break;
                            case "RM":
                                pondImpactTable.Add(coeffImpactRM * priority);
                                pondMaturityTable.Add(coeffImpactRM);
                                pondPrioTable.Add(coeffPrioRM);
                                break;
                            case "GEN":
                                pondImpactTable.Add(coeffImpactGEN * priority);
                                pondMaturityTable.Add(coeffImpactGEN);
                                pondPrioTable.Add(coeffPrioGEN);
                                break;
                            case "Holding":
                                pondImpactTable.Add(coeffImpactHolding * priority);
                                pondMaturityTable.Add(coeffImpactHolding);
                                pondPrioTable.Add(coeffPrioHolding);
                                break;
```

```
                            case "TS":
                                pondImpactTable.Add(coeffImpactTS * priority);
                                pondMaturityTable.Add(coeffImpactTS);
                                pondPrioTable.Add(coeffPrioTS);
                                break;
                            case "Groupe":
                                pondImpactTable.Add(coeffImpactGroupe * priority);
                                pondMaturityTable.Add(coeffImpactGroupe);
                                pondPrioTable.Add(coeffPrioGroupe);
                                break;
                            case "PCH":
                                pondImpactTable.Add(coeffImpactPCH * priority);
                                pondMaturityTable.Add(coeffImpactPCH);
                                pondPrioTable.Add(coeffPrioPCH);
                                break;
                        }
                    }
                }
            }

            // Parcours des tableaux afin de récupérer les valeurs calculées des
paramètres
            for (int i = 0; i < rohTable.Count; i++)
            {
                // Position en coordonnées polaires
                rohSumPond = rohSumPond + rohTable[i] * pondImpactTable[i];
                tetaSumPond = tetaSumPond + tetaTable[i] * pondImpactTable[i];

                // Position en termes de cadran
                availabilitySumPond = availabilitySumPond + availabilityTable[i] *
pondImpactTable[i];
                impactSumPond = impactSumPond + impactTable[i] * pondImpactTable[i];

                // Maturité et priorité
                prioriteSumPond = prioriteSumPond + prioriteTable[i] * pondPrioTable[i];
                maturiteSumPond = maturiteSumPond + maturiteTable[i] *
pondMaturityTable[i];


                // Somme des pondérations pour la moyenne qui sera calculée par la suite
                pondImpactSum = pondImpactSum + pondImpactTable[i];
                pondMaturitySum = pondMaturitySum + pondMaturityTable[i];
                pondPrioSum = pondPrioSum + pondPrioTable[i];
            }

            // Réajustements pour éviter les erreurs d'arrondi
            impactSumPond = impactSumPond + 0.01;
            rohSumPond = rohSumPond + 0.01;
            availabilitySumPond = availabilitySumPond + 0.01;
            prioriteSumPond = prioriteSumPond + 0.01;
            maturiteSumPond = maturiteSumPond - 0.01;

            // Détermination de l'angle et la position de la technologie du radar général
            tetaFinal = (double)tetaSumPond / pondImpactSum;
            rohFinal = (double)rohSumPond / pondImpactSum;

            // Détermination de la maturité et de la priorité de la technologie du radar
général
            prioriteFinalTemp = (int)(Math.Round((double)prioriteSumPond / pondPrioSum));
            switch (prioriteFinalTemp)
            {
                case -2:
                    prioriteFinal = "Sous surveillance";
                    break;
                case -1:
                    prioriteFinal = "Autre";
                    break;
                case 0:
                    prioriteFinal = "Non-prioritaire";
                    break;
                case 1:
                    prioriteFinal = "Suiveur";
                    break;
                case 2:
                    prioriteFinal = "Faible";
                    break;
                case 3:
                    prioriteFinal = "Moyenne";
                    break;
                case 4:
```

```
                    prioriteFinal = "Haute";
                    break;
                default:
                    prioriteFinal = "";
                    break;
            }

            maturiteFinalTemp = (int)(Math.Round((double)maturiteSumPond /
pondMaturitySum));
            switch (maturiteFinalTemp)
            {
                case 4:
                    maturiteFinal = "Mainstream";
                    break;
                case 3:
                    maturiteFinal = "Early adopters";
                    break;
                case 2:
                    maturiteFinal = "Adolescent";
                    break;
                case 1:
                    maturiteFinal = "Emerging";
                    break;
            }


            // Détermination des valeurs d'impact et de disponibilité attendues en vue de
les comparer avec celles obtenues par le calcul
            impactDesired = Math.Round((double)impactSumPond / pondImpactSum);
            availabilityDesired = Math.Round((double)availabilitySumPond /
pondImpactSum);

            // Version temporaire des coordonnées finales
            xFinal = rohFinal * Math.Cos(tetaFinal);
            yFinal = rohFinal * Math.Sin(tetaFinal);

            int count = 0;
            // Réajustement de l'angle en fonction de l'erreur d'impact
            while (GetImpact(xFinal, yFinal) != impactDesired && count <
Const.iterationStepLimit)
            {
                if (GetImpact(xFinal, yFinal) > impactDesired)
                {
                    tetaFinal = tetaFinal * (1 - Const.iterationStep);
                    xFinal = rohFinal * Math.Cos(tetaFinal);
                    yFinal = rohFinal * Math.Sin(tetaFinal);
                }
                else
                {
                    tetaFinal = tetaFinal * (1 + Const.iterationStep);
                    xFinal = rohFinal * Math.Cos(tetaFinal);
                    yFinal = rohFinal * Math.Sin(tetaFinal);
                }
                count++;
            }

            count = 0;
            // Réajustement de la distance en fonction de l'erreur d'impact
            while (GetYear(xFinal, yFinal) != availabilityDesired && count <
Const.iterationStepLimit)
            {
                if (GetYear(xFinal, yFinal) > availabilityDesired)
                {
                    rohFinal = rohFinal * (1 - Const.iterationStep);
                    xFinal = rohFinal * Math.Cos(tetaFinal);
                    yFinal = rohFinal * Math.Sin(tetaFinal);
                }
                else
                {
                    rohFinal = rohFinal * (1 + Const.iterationStep);
                    xFinal = rohFinal * Math.Cos(tetaFinal);
                    yFinal = rohFinal * Math.Sin(tetaFinal);
                }
                count++;
            }

            // Mise à jour des coordonnées et de la maturité après réctification
éventuelle
            newItem[Const.xTechnologie] = xFinal;
            newItem[Const.yTechnologie] = yFinal;
```

```
            newItem[Const.impactTechnologie] = GetImpact(xFinal, yFinal);
            newItem[Const.echeanceTechnologie] = GetYear(xFinal, yFinal);
            newItem[Const.maturiteTechnologie] = maturiteFinal;
            if (prioriteFinal != "")
                newItem[Const.prioriteTechnologie] = prioriteFinal;

            // Update de l'item
            try
            {
                newItem.Update();
            }
            catch { }

        }

        // Renvoie l'impact en fonction des coordonnées sur le radar
        private int GetImpact(double x, double y)
        {
            double C1 = Const.iC1; double C2 = Const.iC2; double C3 = Const.iC3; double
C4 = Const.iC4; double C5 = Const.iC5;
            int impact = -1;

            if (y > C1 * x && y < C2 * x)
            {
                impact = 1;
            }
            else if (y > C2 * x && y < C3 * x)
            {
                impact = 2;
            }
            else if (y > C3 * x && y < C4 * x)
            {
                impact = 3;
            }
            else if (y > C4 * x && y < C5 * x)
            {
                impact = 4;
            }

            return impact;
        }

        // Renvoie l'année de disponibilité en fonction des coordonnées sur le radar
        private int GetYear(double x, double y)
        {

            int annee = -1;

            if (y > 34 || x > 45 || y > 34 * (System.Math.Sqrt(1 - (x / 45) * (x / 45))))
            {
                annee = 5;
            }
            else if (y > 28.5 || x > 36 || y > 28.5 * (System.Math.Sqrt(1 - (x / 36) * (x
/ 36))))
            {
                annee = 4;
            }
            else if (y > 22 || x > 28 || y > 22 * (System.Math.Sqrt(1 - (x / 28) * (x /
28))))
            {
                annee = 3;
            }
            else if (y > 16 || x > 21 || y > 16 * (System.Math.Sqrt(1 - (x / 21) * (x /
21))))
            {
                annee = 2;
            }
            else if (y > 11 || x > 13 || y > 11 * (System.Math.Sqrt(1 - (x / 13) * (x /
13))))
            {
                annee = 1;
            }
            else if (x > 0 || y > 0)
            {
                annee = 0;
            }

            return annee;
        }
```

```
        // Remplit les listes déroulantes permettant de sélectionner le radar de départ
        public void Populate(String NatureValue)
        {

            using (SPWeb web = new SPSite(SPContext.Current.Web.Url).OpenWeb())
            {

                SPList list = web.Lists[Const.TechnologiesFiltres];
                SPListItemCollection items = list.Items;

                foreach (SPListItem item in items)
                {
                    String kind = (String)item["Nature"];
                    if (kind == NatureValue)
                    {
                        String temp = (String)item["Valeur"];
                        ListItem tempItem = new ListItem(temp, temp);
                        if (NatureValue == "Year")
                        {
                            radarInitialYear.Items.Add(tempItem);
                            radarGeneralYear.Items.Add(tempItem);
                        }
                        else if (NatureValue == "Branch")
                        {
                            radarInitialBranch.Items.Add(tempItem);
                            radarGeneralBranch.Items.Add(tempItem);
                        }
                        else if (NatureValue == "Version")
                        {
                            radarInitialVers.Items.Add(tempItem);
                            //radarGeneralVers.Items.Add(tempItem);
                        }
                    }
                }
            }

        }

        // Supprime les technologies répondant aux critères
        private void DeleteItems(String year, String vers, String branch, SPWeb web)
        {

                List<SPListItem> itemsToDelete = new List<SPListItem>();

                web.AllowUnsafeUpdates = true;
                web.Site.CatchAccessDeniedException = false;

                SPList list = web.Lists[Const.FichesTechnologie];
                SPQuery query = new SPQuery(list.Views[Const.allBranchesView]);
                StringBuilder viewFields = new StringBuilder();
                foreach (SPField field in list.Fields)
                    viewFields.AppendFormat("<FieldRef Name=\"{0}\"/>",
field.InternalName);

                query.ViewFields = viewFields.ToString();
                query.IncludeAllUserPermissions = true;
                query.IncludePermissions = true;
                query.RowLimit = 1000;

                query.Query = "<Where><And><And><Eq><FieldRef Name='" +
Const.brancheTechnologie + "'/>" +
                    "<Value Type='Text'>" + branch + "</Value></Eq><Eq><FieldRef Name='"
+ Const.yearTechnologie + "'/>" +
                    "<Value Type='Text'>" + year + "</Value></Eq></And><Eq><FieldRef
Name='" + Const.versionTechnologie + "'/>" +
                    "<Value Type='Text'>" + vers + "</Value></Eq></And></Where>";

                SPListItemCollection items = list.GetItems(query);

                foreach (SPListItem item in items)
                {
                    itemsToDelete.Add(item);
                }

                int count = 0;
                for (int i = 0; i < itemsToDelete.Count; i++)
                {
                    itemsToDelete[i].Delete();
                    count++;
```

```
                }
                count = 0;
                itemsToDelete.Clear();
        }

        // Supprime les technologies les moins prioritaires afin de former le radar des
priorités
        private void DeleteLeastPrioItems(String year, String vers, String branch, SPWeb
web, List<string> radarToDisplay)
        {

            List<SPListItem> itemsInTheList = new List<SPListItem>();
            List<double> prioSums = new List<double>();
            web.AllowUnsafeUpdates = true;
            web.Site.CatchAccessDeniedException = false;

            SPList list = web.Lists[Const.FichesTechnologie];
            SPQuery query = new SPQuery(list.Views[Const.allBranchesView]);
            StringBuilder viewFields = new StringBuilder();
            foreach (SPField field in list.Fields)
                viewFields.AppendFormat("<FieldRef Name=\"{0}\"/>", field.InternalName);

            query.ViewFields = viewFields.ToString();
            query.IncludeAllUserPermissions = true;
            query.IncludePermissions = true;
            query.RowLimit = 1000;

            query.Query = "<Where><And><And><Eq><FieldRef Name='" +
Const.brancheTechnologie + "'/>" +
                "<Value Type='Text'>" + branch + "</Value></Eq><Eq><FieldRef Name='" +
Const.yearTechnologie + "'/>" +
                "<Value Type='Text'>" + year + "</Value></Eq></And><Eq><FieldRef Name='"
+ Const.versionTechnologie + "'/>" +
                "<Value Type='Text'>" + vers + "</Value></Eq></And></Where>";

            SPListItemCollection items = list.GetItems(query);

            foreach (SPListItem item in items)
            {

                    string nomTechno = item[Const.nomTechnologie].ToString();
                    // Création de la query qui va utiliser la vue "allbranches"
                    query = new SPQuery(list.Views[Const.allBranchesView]);
                    viewFields = new StringBuilder();
                    foreach (SPField field in list.Fields)
                        viewFields.AppendFormat("<FieldRef Name=\"{0}\"/>",
field.InternalName);
                    query.ViewFields = viewFields.ToString();
                    query.RowLimit = 1000;

                    query.Query = "<Where><Eq><FieldRef Name='" + Const.nomTechnologie +
"'/>" +
                        "<Value Type='Text'>" + nomTechno + "</Value></Eq></Where>";

                    SPListItemCollection itemsToAnalyse = list.GetItems(query);

                    double prioSum = 0;
                    foreach (SPListItem itemToAnalyse in itemsToAnalyse)
                    {
                        string radar = itemToAnalyse[Const.brancheTechnologie].ToString()
+ "-" + itemToAnalyse[Const.yearTechnologie].ToString() + "-" +
itemToAnalyse[Const.versionTechnologie].ToString();
                        if (radarToDisplay.Contains(radar))
                        {
                                string provenance =
itemToAnalyse[Const.provenance].ToString();
                                string priorite =
itemToAnalyse[Const.prioriteTechnologie].ToString();
                                double coeff;

                                switch (provenance)
                                {
                                    case "EP":
                                        coeff = coeffPrioEP;
                                        break;
                                    case "RM":
                                        coeff = coeffPrioRM;
                                        break;
                                    case "GEN":
                                        coeff = coeffPrioGEN;
```

```
                                    break;
                                case "Holding":
                                    coeff = coeffPrioHolding;
                                    break;
                                case "TS":
                                    coeff = coeffPrioTS;
                                    break;
                                case "Groupe":
                                    coeff = coeffPrioGroupe;
                                    break;
                                case "PCH":
                                    coeff = coeffPrioPCH;
                                    break;
                                default:
                                    coeff = 1;
                                    break;
                            }

                            switch (priorite)
                            {
                                case "Haute":
                                    prioSum = prioSum + Const.pondPrioriteHaute *
coeff;
                                    break;
                                case "Moyenne":
                                    prioSum = prioSum + Const.pondPrioriteMoyenne *
coeff;
                                    break;
                                case "Faible":
                                    prioSum = prioSum + Const.pondPrioriteBasse *
coeff;
                                    break;
                                case "Suiveur":
                                    prioSum = prioSum + Const.pondPrioriteSuiveur *
coeff;
                                    break;
                                case "Non-prioritaire":
                                    prioSum = prioSum + Const.pondPrioriteNonPrio *
coeff;
                                    break;
                                case "Sous surveillance":
                                    prioSum = -1 * coeff;
                                    break;
                            }
                        }
                    }
                    prioSums.Add(prioSum);
                    itemsInTheList.Add(item);

            }

            List<double> prioSumsUnsorted = new List<double>(prioSums);
            prioSums.Sort();
            prioSums.Reverse();
            List<double> prioSumsSorted = prioSums;

            double threshold = prioSumsSorted[Const.threshold];

            for(int i = 0; i<itemsInTheList.Count;i++)
            {
                string nametech = itemsInTheList[i][Const.nomTechnologie].ToString();

                if (!(prioSumsUnsorted[i] >= threshold))
                {
                    itemsInTheList[i].Delete();
                }
            }

        }

        // Génère la liste des radars
        protected List<string[]> generateRadarList()
        {
            List<string[]> radarList = new List<string[]>();

            using (SPWeb web = new SPSite(SPContext.Current.Web.Url).OpenWeb())
            {
                web.AllowUnsafeUpdates = true;
                web.Site.CatchAccessDeniedException = false;
                string liste = Const.FichesTechnologie;
```

```
                SPList list = web.Lists[liste];
                SPQuery query = new SPQuery(list.Views[Const.allBranchesView]);
                StringBuilder viewFields = new StringBuilder();

                viewFields.AppendFormat("<FieldRef Name=\""+ Const.brancheTechnologie
+"\"/>");
                viewFields.AppendFormat("<FieldRef Name=\"" + Const.yearTechnologie +
"\"/>");
                viewFields.AppendFormat("<FieldRef Name=\"" + Const.versionTechnologie +
"\"/>");


                query.ViewFields = viewFields.ToString();
                query.RowLimit = 1000;

                query.Query = "";

                SPListItemCollection items = list.GetItems(query);

                foreach (SPListItem item in items)
                {
                    string[] temp = new string[] {
item[Const.brancheTechnologie].ToString(), item[Const.yearTechnologie].ToString(),
item[Const.versionTechnologie].ToString() };
                    if (containStringTable(radarList,temp)) { radarList.Add(temp); }
                }
            }

            return radarList;


        }

        // Retourne un booléen TRUE si radarList contient déjà le tableau de string
        private bool containStringTable(List<string[]> radarList, string[] stringTable){
            for (int i = 0; i < radarList.Count; i++)
            {
                string branch = stringTable[0];
                string year = stringTable[1];
                string vers = stringTable[2];

                if ((radarList[i])[0] == branch && (radarList[i])[1] == year &&
(radarList[i])[2] == vers)
                {
                    return false;
                }
            }
            return true;
        }

    }
}
```

9. Glossary

Webpart	A webpart is a SharePoint component that is developed by a programmer, then the end user can integrate it to webpages in “webpart zones”
POT	Technological Orientation Plan, it contains information concerning the current state of each technology in every branch
DSIT	Telecommunications and Information Systems Direction
REST	Representational State Transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web
SOM	SharePoint Object Model
RDF	The Resource Description Framework (RDF) is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata data model.

10. Bibliography

Websites

Http://en.wikipedia.org/wiki/User_interface_design

Http://en.wikipedia.org/wiki/Usability

Http://en.wikipedia.org/wiki/Agile_software_development

Http://en.wikipedia.org/wiki/Microsoft_SharePoint

Principles for usable design, Http://www.usabilitybok.org/design/p287
Semantic Web Tutorial, Http://www.w3schools.com/semweb/

Tutorial: functional specification, Http://www.mojofat.com/tutorial/index.html

Manifesto for Agile Software Development, *Http://agilemanifesto.org/*

Brown, Laura, *Observational Field Research*,
http://www.socialresearchmethods.net/tutorial/Brown/lauratp.htm

Http://www.usabilitynet.org/home.htm

Http://www.useit.com/papers/heuristic/heuristic_list.html. 2003.

Rastello, M. Microsoft Office SharePoint Server 2007 and Office Groove 2007: Implémentation.
Http://www.labomicrosoft.org/articles/Microsoft_Office_SharePoint_Server_2007_Interaction_Groove/0/

Literature

Constantine, L. Lockwood, L. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design.* Addison-Wesley Professional. 1999

Lidwell, W., Holden, K., and Butler, J. *Universal principles of design.* Rockport Publishers. 2003.

Leffingwell, D. Agile Software Requirements. Addison-Wesley Professional. 2011

Nielsen, J. Ten usability heuristics.

Alan Dix, Janet Finlay, Gregory D. Abowd, Russell Beale, *Human-Computer Interaction*, 2005, available at http://cognac.ai.ru.nl/studie/MMI_summary.pdf