A Case-Based Engine to Create Dynamic Content
Adapting Users’ and Context Profiles

Daniel Almirón Fuentes
Didac Gil de la Iglesia
Abstract

Nowadays, we can find many interactive applications and mobile services accessible everywhere. Usually, these services have been designed to serve a unique target population, independent for instance; of the kind of devices the users have or the particular situation the users are encountered.

In many cases, these services do not provide exactly the type of information people are expecting and looking for. Another related problem in this particular domain concerns the way information is presented, both from a content and a presentation perspective.

Many times, although the content may fit our expectations, the way this information is presented does not fit the proper device, neither for our actual situation. Those are the particular problems we want to tackle in this work, the development of more flexible applications and services that could adapt to a particular user in different environments.

The scope of this thesis relates to the field of adaptive hypermedia. Our main goal is to explore, develop and implement an approach that will let adapt a number of mobile services to new situations, depending both on the user’s and context’s profiles. In this report we present the ideas and results of the work we have been conducting during the last four months.

Keywords: Adaptive Hypermedia, Case-Based Reasoning, Artificial Intelligence, Context Awareness, CC/PP, UAProf, Mobility, Apache Cocoon, XML, Information Retrieval, Human Computer Interaction.
12.c Selection Engine
1 Introduction

1.1 Motivation

The authors of this thesis are Daniel Almirón Fuentes and Didac Gil de la Iglesia, both students in the last year of Computer Science Engineering at the Universitat Autònoma of Barcelona (UAB) in Spain. We are taking part of the Erasmus program in the Växjö University, in Sweden.

The reason to work within the field of adaptive hypermedia is because we believe that actual technology has developed in a way that allow us as to explore new ways to support users in their daily tasks. Our vision is that the user has to be always in the first place, and products and services should be designed having in mind this viewpoint.

On the other hand, we would like to explore ways in which artificial intelligence techniques can be used in order to see how the user’s vision of the world could be stored and represented inside the computer, and how could be possible to use this information in order to bring a better solution to him/her.

1.2 Identifying the problem

The aim of our work is to explore new ways in which techniques and knowledge within the fields of artificial intelligence, information retrieval and adaptive hypermedia can be combined in order to allow us to develop an approach that will customize a service to the user, considering his/her preferences, his/her characteristics and also the environment’s and device’s characteristics. Moreover, we will illustrate this approach by implementing a demonstrator; these aspects will be described later in this document.

Here we pose three questions which are the basis for our research:

1. "Can we obtain good results using a case based reasoning in order to adapt the service to the user, from the feedback gathered by the system in previous situations?"
2. “Which are suitable technologies to support the task of adapting the way to present the services to the users?”
3. “Which formalisms will be used in order to get a better understanding of the user and his/her environment?”

Through this report we are going to expose several techniques related to these topics, and in the conclusions’ section we will discuss the results we have achieved.

1.3 Technical background

Due to the fast growing of the Web and the increasing number of users, many organizations and companies are exploring new ways to publish their services in many different ways.

Despite this situation, the way of how the Web could interact with the user and the way how content is presented does not change considerably during time. Due the
heterogeneity of users that can access to the Web and its actual limitations of adaptability, many users are not satisfied and maybe would not use the same service again.

From this starting point, it has become more important that the user and his/her features should be considered early in the design process. Some of these new fields of research related to the user are the Human-Computer Interaction (HCI) and the Adaptive Hypermedia (AH).

Human-Computer Interaction is “the study of how people interact with computers and to what extent computers are or are not developed for successful interaction with human beings” [54]. Historically, computer system developers have not paid much attention to computer ease-of-use. Many computer users today would argue that computer makers are still not paying enough attention to making their products "user-friendly." This discipline tries to change the way of how the design of the products are built, always making the design process with the user in mind.

Furthermore, not only making the interface user-friendly is enough to have a good interaction with the system. It would be good if the user could have a particular interaction, according to his preferences. According to [Whitehead, 2004] hypermedia can be defined as an associative relationships among information contained within multiple media data for the purpose of facilitating access to, and manipulation of, the information encapsulated by the data. Through this definition, the way of how the user can perceive this information can be done by an adaptable hypermedia, where the user has to customize the application by explicit commands, or by an adaptive hypermedia, where the customization is automatic, it happens by observing the user and his environment.

As the adaptive hypermedia needs to customize the service automatically from the user and his behavior, the system must interpret the information that arrives from the user, and store it. User Modelling (UM) is that part of an interactive system whose function is to incrementally construct a user model. The user model is defined as “an information structure containing knowledge about the user, which is exploited by the system in order to increase the quality of the interaction” [2]. In this kind of systems, Artificial Intelligence (AI) plays a very important role. It provides techniques to retrieve the data (data mining [Giudici, 2003]) from the databases and from this data give a satisfying solution for the user.

As we already mentioned, nowadays not only content is an important ingredient to have a good service. We need also to have to look at the presentation. If a user wants to use a service from his cell phone, but this service only works for desktops, don’t mind how good the service is, the user will not use it. Due to this reason, especially with the increase of mobile devices, it becomes more important the development of applications and services for any kind of device. The problem now is how can the system know which is the device which is asking for the service, and how has to send the data.

Under the sentence “Access to a Unified Web from Any Device in Any Context by Anyone” [56] the Device Independent work group (World Wide Web Consortium) has the mission of avoid fragmentation of the Web into spaces that are accessible only from subsets of devices. Nowadays, this work group has to continue the work of the CC/PP.

CC/PP stands for Composite Capabilities/Preferences Profile [42], and is a system for expressing device capabilities and user preferences by using the Resource Description Framework (RDF), build on URI and XML. With CC/PP, a user with a specific preference or disability-related need can clarify that even though their browser
handles millions of colors, they personally can only distinguish certain colors. This specification allows to have detailed information about the device capability, and adapt the presentation to this device in concrete.

For mobile devices there is a specification called User Agent Profile (UAProf) [41], developed by the Open Mobile Alliance. It is a CC/PP vocabulary specially developed to describe cell phone. For the moment, there are millions of mobile phones using this specification.

All this new technologies are based in the eXtensible Markup Language (XML), also a W3 recommendation. XML allows separate content and layout to offer flexibility to different representation from the same data.

Under the same philosophy, around 1998, appeared Apache Cocoon [14]. It is a “web development framework built around the concepts of separation of concerns and component-based web development”. Cocoon implements these concepts around the notion of 'component pipelines', each component on the pipeline specializing on a particular operation. This makes it possible to generate depending on the device one presentation or another one from the same content by applying different templates.

1.4 Adaptation

In the next chapters we will explain in more detail which elements may be adapted in the application. In this section we will basically give a brief introduction to this elements and we will explain the way we will combine them to obtain a final solution.

In our vision of the problem, the adaptation of the system may be made from three clue elements: the user, the device being used in that moment and the environment where the user is.

A user is the person that uses the application. His principal characteristics may be his likes, skills, knowledge, etc. This information may be picked in different ways, as we will see furthermore.

The device refers to the media how the customer is using the application: basically the principal properties are the hardware and the software.

Finally, as the environment consist of all the aspects that surround the user in the application running moment, it is important for the adaptation. It can be relevant for the system some details as the light, noise and location, etc.

From user, device and environment information, it is possible to make several combinations to obtain a result to get adapted in the content as much as they way to present it. The most complete manner to use this information would be that all the data influence in all the decisions. This way we will have:

**User, device and environment → adaptation of the content**

**User, device and environment → adaptation of the presentation**

In our solution we have adopted in a reduced manner that trilogy, since in the same way it let us a high degree of adaptation and besides simplifies the global process. The combination of the elements is the next:

**User → adaptation of the content**

**Device and environment → adaptation of the presentation**
We believe that in this way we will take advantage of the most important characteristics of each profile, and we let the possibility of a future extension where the three profiles were considered in each adaptation process.

1.5 Disposition

In chapter 2 we will find State of the art. All the theoretical information needed to create an adaptive hypermedia application, including new concepts definitions and scopes where we can apply them.

In chapter 3, we will introduce the new ideas seen in chapter 2 to our application. A Software Engineering Approach is explained in detail before going into the implementation.

The developed application is analyzed in chapter 4, having a general idea about which items are needed in it and why are they needed.

Chapter 5 contains information about how the application has been done. How the database is buildt, how to navigate across the website, some important lines in the code as well as a user manual to use the website.

Tests are shown in chapter 6, showing some screen shots with different devices and environments.

After creating the implementation and doing all the required studies, some conclusions are found in chapter 7, seeing what have we learnt from this project and what can we reuse for other ones.

Finally future development and evolutions is presented in chapter 8. We will find some possible modifications that will improve the application and also what it is going to appear in next years.
2 State of the art

We have been working on an application which adapts a presentation according to some user information.

In this chapter we will see what can be adapted and when it’s is useful to adapt a presentation. In order to create a proper adaptation, be need to gather some information from the user, his/her device and the environment where the user is located, so we will explain how can this information be adquired. Related with it, be find Ubituitous computing, which will be explained in the mobility head. Finally, we will talk about how all this presentation has to be done. How can we create different presentations having the same information, introducing to the XML world.

2.1 Adaptive Hypermedia

Adaptive hypermedia is a new direction of research within the area of adaptive model-based interfaces. Those systems build a model of the individual user an apply it for adaptation to that user. [Brusilovsky et al. 1998]

A "classic" hypermedia application serves the same pages and the same set of links to all users. This is true even for most applications that are built on top of systems that are capable of presenting different views to different users.

Adaptive Hypermedia Systems (AHS) make it possible to deliver "personalized" views or versions of a hypermedia document (or hyperdocument for short) without requiring any kind of programming by the author(s). Also, although it is possible to offer users a way to initialize the user model through a questionnaire, an AHS can do all the adaptation automatically, simply by observing the browsing behaviour of the user. (There are many adaptable systems that allow "personalized" views based on user-selected stereotypes like "beginner" and "expert", or based on interface and style preferences. [4]

For example, it could be possible to adapt the user presentation and content by changing the links kind and distribution for each interaction, depending on the user’s goal, knowledge about the topic, technical background and his environment.

In the next Illustration (see fig. 1) we can see the classic loop “user modeling-adaptation” adaptive systems frameworks.

![Figure 1: Adaptation process passing by user data gathering and user model creation.](image)
The user data can be gathered directly from the user, by asking typical questions like personal data and demographic characteristics, questions that cannot be inferred by the system. Also it is possible collect user data from the observation of the user behavior. The application watches how the user interacts with the system and infers some useful information from his behavior. This gathered data is processed (User Modelling) by the system and represented as the user model. Having the user model the system must do the transformation in order to adapt the presentation in the proper format for the concrete user.

2.1.1 Where can adaptive hypermedia be used

However, not all the applications need to be adaptable to the user. Then, where adaptive hypermedia can be helpful?

[Brusilovsky et al., 1998] enumerates six different kind of hypermedia systems where it can be useful to. Those six areas are educational hypermedia, on-line information systems, on-line help systems, informational retrieval hypermedia systems, institutional information systems and systems for managing personalized views.

- Educational Hypermedia

It is the most common application domain within the adaptive hypermedia research field. The aim of this kind of areas is to assure the user learns most of the site lessons.

Not all the people studying on a Web School has the same knowledge level; a novice user, due his knowledge, will find a lesson page difficult to follow while an expert user will get bored about it, and in that way the adaptive hypermedia can be used. It can be useful also for giving some help to a novice user while the navigation by the web site, as it all will be new for him.

This area has been astonishingly growing, and that has given motivation to invest in it.

- On-line Informational Systems

This one is another popular type of application. This area has changed a lot during the last years, and now it can be divided in different subgroups. [Alexandre et al., 2003]

- Electronic encyclopedias and information kiosks

The aim is to provide reference access to information, instead of having a educational hypermedia using systematic introduction.

Those specialized systems can follow the user acknowledge and give information using comparisons with aspects learnt before (E-Encyclopedies) or can learn about the user preferences following his movements and use this information to offer the most interesting articles on a user point of view.

- Virtual museums and museum hand guides

Knowing where the objects are located, let the user explore a virtual or real museum giving an hyperspace adapted guide, with comments adapted to the environment. Museum hand guides are able to determine where the user is located and his behaviour, and it uses this information to stop or start the information streaming. If a user is passing near a picture, but his movement is quick, the hand guide will not give any data about it, but if he is walking slowly, it will start its presentation.
• **E-Commerce systems and activity support**

Those systems support what the user is doing, and save information about the user goals and the context where the user is having his activities. Those aspects all together able the system to create a more detailed user model and consequently, a higher adaptation level.

• **On-line help systems**

On-line help systems are used to help user to find information about the program they are using. So, the difference to the On-line Informational Systems is the program dependence and the hyperspace used in them is much smaller. As the hyperspace is smaller, we can have a better knowledge about the user's goals and we can give him the most relevant results.

• **Information Retrieval hypermedia systems.**

This new area of information retrieval is combined with a content document comparison. It's know that for any user it's very difficult to make a concrete formal query, so the system will help the user by giving him some query answers which have similar contents.

This structure is done by a system, using some measurements like a similarity one could be. A current example of it is the IR systems on the World Wide Web, where the hyperspace has no limits and it can be very difficult to find what a user is really looking for.

We can distinguish two groups of IRHS:

• Search oriented IRHS

After the user has made a query asking for some links, it's given to him a list of those taking in account the words written by the user inside the query but also his preferences and interests, in other words, using his profile. Inside this group, we can find two more subgroups which are:

1. Classic IR Systems- Used in closed information domains. The hyperspace is tuned to give for providing a quick browsing.

2. Searching filters – Used in open information domains. It extends the power of the searcher motor engines by using different browsing support adaptations. One example of it is **SysKill & Webert**. This software agent learns from the user selections when he makes a query in a motor engine. After the results are given, the user is asked to give a mark to the given result. The agent analyzes the content of the page to create a user profile, and it will use this profile in next queries to give a more adapted result and will also give some pages that may be interesting for the user. [3]

• Browsing oriented IRHS

They try to help the user to explore the web by using some browsing adaptation.

We have here 3 subgroups:

1. Adaptive Orientation – The user is given some additional information to have a easier navigation.

   One example of this is **WebWatcher**. After the user tells what kind of information he seeks, it will accompany him from page to page highlighting
hyperlinks that it believes will be of interest. The decision to highlight a link and not another one is learnt by previous tours done with this agent.

2. Adaptive Annotation – Gives some images next to the links to help the user to take the selecting decision.

3. Adaptive Recommendation – It's objective is to deduce the user goals and interests looking at his movements by the net. The user is helped, by his browsing, been showed a way to reach a page he can be interested in and which is impossible to reach directly from the current page or following it's links. The agent tries to learn what the user is looking for, and gives him a list of links to pages having that content. This group can be divided into two more subgroups depending on the openness of the hyperspace. (see fig. 2)

When the hyperspace is closed, the agent can look for all the pages in the hyperspace and give the results, but when the hyperspace is open, the agent only studies part of it.

Institutional Information Systems.
In some institutions, we can find this kind of adaptive hypermedia. Before changing, the system was a institutional data base, where the employees could find the

![Diagram](image)
information they were looking for. But now, they have the possibility to combine their DB with a single hyperspace. When an employee is seeking information, this one will be related with his current work, so he will want to get that information that is more adapted to him.

Usually, employees get distracted looking in some results they are not interested in according to their job, and this could be solved giving him fewer but more work-oriented results.

- **System for managing personalized views.**

When a telecommunication system offers as much information as the WWW can offer, this information can be stressing if we get all. A normal user will want to set some view for the whole hyperspace and get only those entries that match with his profile, containing his work interests and goals. It tries to solve the same problem that Institutional Information System has, but in this case, the hyperspace is increasing and evolving, having to check each time its content (searching for disappeared items and adding the new ones).

As we can see, the separation between those areas is not well defined. Some of them are pair-problem-related and also share some aspects of some of those six areas. Those pairs are: IR hypermedia and On-line information systems, on-line systems and educational hypermedia, educational hypermedia and institutional hypermedia, institutional hypermedia and information space management systems.

Taking this similarities into account, we can draw a graph (fig. 3) where all those areas are next to those ones that share some features. In the middle we can find those areas that have all classic hypermedia characteristic, as contextual links, indexes, guided tours, etc. due to have a small size of hyperspace. An the ends of the line, the hyperspace size gets bigger, and it's more difficult to get all those features from classics hypermedia, so they only use some of them.

IR Hypermedia and Personalized Views in Information Spaces also have some similarities related to large hyperspaces.

2.1.2 *What can be adapted?*

If we want to adapt a system, first of all we should know what can be adapted on it. Finding which features can be different for each user is our answer. We don't have a
large set of features that can be adapted. Hypermedia consist of hyperdocuments (content) or WebPages, called nodes in hypermedia terminology, and links relating ones to others. So, what it is possible to adapt is the content of the pages and, for the second group, the links, index pages and maps.

According to [Brusilovsky et al., 1998], there are those two classes, the first one is called Adaptive Presentation and the second one Adaptive Navigation Support.

- **Adaptive Presentation (content adaptation)**
  It's tried to adapt the presentation to agree the user preferences, goals and knowledge (an expert user will be looking for more detailed information than a novice user can afford, instead of that, a novice user would prefer to get some help).
  “The adaptation can range from a simple (automatic) selection between different versions of some information pages to the completely dynamic generation of all pages from atomic information units and the automatic generation of all hypertext links” [5]
  In this area, we find two domains, the adaptive text presentation and the adaptive multimedia presentation.
  The most common of those is adaptive text presentation, because even there are some systems that have non-textual items, they don't really adapt them. Canned text fragments are manipulated to adapt the presentation of information to the user preferences.
  In the text domain, it's usually adapted by adding/deleting information, altering text fragments (changing the content of a page to be sure that the most important items inside, in the user point of view, are seen by the user), sorting text fragments, summarizing text fragments or hiding them (MS-Word, for example has the capability to add hidden comments that are later shown if the user wants so).
  So, having a canned text, with additional information, using *conditional inclusion of fragments*, will decide if this supplement has to be displayed. That will depend on the user model and the concept relationships in the domain model.
  There is another technique that uses *stretch*. When a block of interesting information is not visible enough, it is stretched. Also, the redundant information is shrunken to give more importance to the other texts and let the user find the desired information much faster. The user can undo this adaptations by clicking on the modified areas.

- **Adaptive Navigation Support**
  Adaptive Navigation support changes links to help the user to find their path in hyperspace. It hides, sorts, annotates links generates new ones, direct guidance and map adaptation.
  For example, adaptive navigation support can be done by changing the link anchor depending on the link relevance. If a link is supposed to be very useful for the user, it will be coloured in green, if it's not relevant, it will be displayed in red.
  When a link is leading to inappropriate or non relevant information are hidden by presenting them as normal text, instead of being links (ISIS Tutor [Brusilovsky]).
• **Direct guidance**: The user will find a “next” or “follow” button next to the most appropriate node (page) that the AHS has selected for him.
• **Sorting of links**: Links are sorted ordered by its relevance (from most relevant to less relevant). This technique has been used in information retrieval and in goal oriented educational systems.
• **Link annotation**: Different anchors presentation of a link are used here to show the content relevance of the destination. It can also add some annotation items, like colored arrows, to give this information. Commonly they are displayed green if their content is interesting and red means that it is inappropriate. Some software let the user configure that colors.
• **Link hiding**: Those links that lead the user to inappropriate content pages, are hidden. The way to hide them could be like presenting them as normal text (setting the link colour in black, so it seems normal text).
• **Link disabling**: The link entry is deleted. That means the user will not be able to follow the link (for example, the browser will delete the `<a href=""></a>` tags. This technique is usually used in combination with link hiding or link annotation.
• **Link removal**: Links and anchors are removed. This can be useful in lists of links, but doesn't work for anchors that appear in running text.
• **Map adaptation**: For those hypermedia systems that offer graphical presentation of the link structure, are able to have those maps adapted.

This graphic (fig. 4) can summarize where be can apply the adaptive methods.

![Figure 4: Adaptation Technologies Summary](image)

2.2 Gathering information: The user

The most important characteristics related with the user are the following ones [Alexandre et al., 2003]:

- **Demographic characteristics**: those objective information about the user as his personal data (name, surname, address, etc), geographical information (location,
country, etc), characteristic information as user age or sex, psychological
information (life style) and information as a buyer (how much does he spend
buying, how often).

• The user level acknowledge in a subject represented in the hyperspace. This
aspect has been one of the most important ones in the user study. Each user has
his own acknowledge, and it's changing while he is learning. So the Adaptive
Hypermedia System has to follow his apprenticeship and update this user
information.

  The user acknowledge in one subject is represented by a revetment
model. This model is a net of concepts related between them, and there is a value
for each of them estimating the user acknowledge in that concept.

  By this reason, the revetment model can be saved as a set of concept-
value tuples.

• Goals and user plans: Is more related with the user work than in the user itself.
It tries to answer why the user is using the hypermedia system and what does he
wants to reach with it.

  The user goal can change quite often, maybe once per session, or maybe
more.

  It can be considered as the most important user characteristics.

  It's representation is similar to the acknowledge one. The nodes here are
some predefined goals, in an usually small set of goals. Not all the goals are
related between them.

  It can also be represented as a tree of goals.

  It has been demonstrated that knowing which is the user goal, speeds up
the user and computer interaction once the system has some idea about what the
user is looking for.

• User baggage and user experience in the hyperspace: Quite similar with the
user acknowledge in a subject. The difference between them is:
  1. User Baggage : All the user experience before going inside the
hypermedia. That's the user occupation, experience in similar scopes, etc.
  2. User experience : How easily can the user browse by the hyperspace.

• User preferences: This information has to be given by the user in some way.
Sometimes a user prefers one page instead of another one or some links instead
of other ones.

  This preferences can be absolute preferences or relative preferences. That
depends on the current node, the goal, and the current context.

  It's representation is always done in a numeric way instead of a symbolic
one. Using this methodology, we can combine several user models and create a
group model for all those users.

• User interests: They are combined with the user goals to filter or recommend
items.

  The user is asked to mark some items. After this interest has been
established, the system will look for other user who have given the same marks
to the items. The user is given those items that other users with his same
preferences have marked as interesting ones.
• *User personality*: (introvert, extrovert, etc). Those information is not going to change easily, and has to be extracted from the user using specialized psychological tests.

2.3 Gathering information: The environment

Environment can be any information used to detail the entities location (where a person or object is) useful for the user and application interaction.

Context categories usually are the location, people identity and state, groups and physical objects and computerized objects.

What it's needed to define in the context is:

1. Environment
   - User location
   - Interesting items in the user point of view
   - Presentation content
   - Session establishment

2. Entities
   - Places: geographical regions, rooms, offices, building and streets
   - People: single person or group, joined groups or distributed groups
   - Objects: physical objects or software components and artifacts

3. Categories
   - Identity: set a unique identifier and telephone number or home address.
   - Location: 2D position information, such as elevation, orientation, proximity between objects, inclusion. It's applied also to places, where we can define it's location, relative position and characteristics.
   - Status (activity): intrinsic characteristics of an entity that are able to be registered, such us temperature, light, noise, psychological facts, vital signs, or the activity a person is developing.

The 3 first are user for places, while the other ones are for people.

- Time: Used for the history. It provides more information about the situation. With it we can know when the information has relevance.

All this information is used in the reasoning process. It's needed to decide a way to get the context information and decide how this information is going to be shown and how user services are going to be affected.

So we have to think about:

1. How an application is going to realize what's happening in the context
2. How are we going to get the information and how the distributed communications are going to be used (it's not supposed to have all the sensors connected to the same device, instead of that, there will be distributed sensors.)
3. How the acquired information is going to be stored
4. How the stored information is going to be searched
2.4 Gathering information: The devices

In the previous sections, we have seen how it is possible to gather information from the user and his/her environment. The last step in our adaptation process is to know what kind of device is the customer using.

Due the wide range of devices we can find, there are many different properties we have to take into account depending on the device. For example, the structure of a web page cannot be the same for a desktop with a 1024x768 screen size and for a cell phone with a screen size of 120x160 pixels. However, not only the resolution becomes important, many characteristics like the capability of sound, how many colors the device have, which is the operating system or which kind of input it have affect the way the user can perceive and interact with the system, and our goal is adapt the application to any particular device according to all this features.

The next question we can do is, how can the application knows which is the device the customer uses? In a first approximation, we can look at the header that any request for a web page in internet have. There is a field inside the header called User-Agent. A User-Agent is a means through which the user communicates with the application, for example a wap browser [48]. This field carries information about the browser that the device has. If we know the browser, it becomes easier to infer the device. For example, if we find the word “Windows CE” in the user agent field, we can deduce that the device runs the browser in a Windows CE operating system, which is only found in Pocket PC. As well, we can infer when we found the word “nokia”, that the device might be a cell phone, or if we find the word “Blaze”, a browser for Palm OS (also a PDA).

However, this approximation is not enough to know all the properties. Besides, exists too many different browsers, and it is difficult to know all the “key words” for each browser and infer from them the device. Then, there must be another way to gather the device specification.

Recently, it has been developed a new standard to define the capabilities of the devices. This new standard is called CC/PP. According to the World Wide Consortium (W3C) [56], CC/PP [42] is a RDF-based framework (in XML) for describing and managing software and hardware profiles that include information on the user agent's capabilities. The user's specified preferences within the user agent's set of options; and specific qualities about the user agent that can affect content processing and display, such as physical location.

CC/PP is designed to work with a wide variety of web-enabled devices, from PDA's to desktop machines to laptops to WAP phones to phone browsers to web television units to specialized browsers for users with disabilities. For example, although the device of a customer could play sound, the user might be unable to hear, and this characteristic could be included inside the CC/PP profile.

The CC/PP framework provides a way to describe generic profiles accessible via the web, for example from the hardware or software vendor, reducing the amount of information that must be directly sent from the user agent itself, an important consideration for limited-bandwidth cellular modem technologies. It would be necessary link where the CC/PP profile is, and the application only has to read the capabilities from this link. If there is any capability or preference that differs from the standard profile, this characteristic must been sent from the device to the application, but not all the user specification.

At present, there is a standard that implements what CC/PP says. This standard is called User Agent Profile (UAProf) and describes the capabilities of the
device/browser for mobile devices. This information is communicated in an XML document and covers the following attributes [43b]:

- Hardware Platform. For example: screen size, audio capability color capability.
- Software Platform. For example: operating system, content types etc.
- Network Characteristics. For example. GSM/GPRS capable, WTLS capable
- Browser name and version, xHTML version, JavaScript Support
- WAP Characteristics: WML version, deck size, OMA download
- Push Characteristics: Push content types, application.

The information is quite comprehensive and structured in a hierarchy in the XML file. These profiles might are more than 10 or 20 k in size. Since it is not feasible for a device to send this information over the air, as we said before for CC/PP, the profiles are stored on a publicly accessible repository so that all the device needs to do is to state the URL of its UAProf.

Example of these profiles is the UAProf for Nokia 6600. It is available at the URL [43]. This URL is contained within the http header of requests from the device, within the field “x-wap-profile”.

The device accesses the application, the application uses the information in the http header to gather the device's attributes and then uses this information to adapt and optimize the way that it presents to the user.

Regrettably, not all the actual devices offer this helpful information when they connect with the applications. Only the new cell phones are presenting the UAProf (already millions of cell phones do that). Hopefully in a near future all devices like desktop, laptop, PDA, television with Internet and any new device that can appear will bring a complete description of its capabilities.

2.5 Mobility

If we take a look at the situation when the first computers appeared, we can see at that time existed the mainframes: one computer, many people. The last decades the situation changed and appeared the PC: one person, one computer. At the moment, the situation is changing again, and we walk through the ubiquitous computing: one person, many computers.

Besides, all computers or devices that one person has are not with the same characteristics and capabilities, and many of them are not in the same place, they have mobility.

From this reality, a new discipline appeared: ubiquitous computing (Concept introduced in 1988 by Mark Weiser at Xerox PARC). This new field of investigation is exactly the opposite of virtual reality. Where virtual reality puts people inside a computer generated world, ubiquitous computing forces the computer to live out here in the world with people. Ubiquitous computing is a very difficult integration of human factors, computer science, engineering, and social sciences.

According to [Schaefter, 2002] one major problem in that field considers seamless and flexible obtainment of information about the context in which computing of a mobile devices takes place.

There are starting services in mobile phone localization services, but they have not been well adapted to other applications yet. Then, nowadays the user has to configure the applications manually when they move from one environment to another.
Schaefer, 2002] in his paper presents profile-based concept for multi-modal interaction in intelligent environments. He proposes four groups to fully describe the environment:

- Devices: hardware and software information of the device, shape, normal position and so on.
- Persons: focuses on people’s preferences and abilities and also basic data as name, age, gender, etc.
- Situation: location of persons and devices, and other important objects in the scene.
- Modalities: are used to model the input and output channels. Define what data can be accepted and how, and how can be mapped to devices or persons.

Ubiquitous computing environments offer a wide range of devices in many different shapes and sizes. In the future, collaboration between user and environments with multiple interconnected devices will determine work and daily activities to a large degree.

Computer Supported Cooperative Work (CSCW) [52] is the use of technology to facilitate the work of groups. It may be used to communicate, cooperate, coordinate, solve problems, compete, or negotiate. A large part of the activities supported by ubiquitous computing environments involve collaboration.

The reasons to use computer supported cooperative work as a way of communicating are the following:

- to facilitate communication: make it faster, clearer, more persuasive
- to enable communication where it wouldn't otherwise be possible
- to enable telecommuting
- to cut down on travel costs
- to bring together multiple perspectives and expertise
- to form groups with common interests where it wouldn't be possible to gather a sufficient number of people face-to-face
- to save time and cost in coordinating group work
- to facilitate group problem-solving
- to enable new modes of communication, such as anonymous interchanges or structured interactions

One example of these two fields of investigation is EasyLiving [53], a ubiquitous computing project of the Vision Group at Microsoft Research. EasyLiving is developing prototype architecture and technologies for building intelligent environments. The key features of this project include:

- Computer vision for person-tracking and visual user interaction.
- Multiple sensor modalities combined.
- Use of a geometric model of the world to provide context.
- Automatic or semi-automatic sensor calibration and model building.
- Fine-grained events and adaptation of the user interface.
- Device-independent communication and data protocols.
- Ability to extend the system in many ways.
2.6 Enabling techniques: AI–XML–Java-Cocoon

In this section we will explain the technologies that will be used together for the problem solution.

Adaptive hypermedia has a strong dynamic component, since the system has to evolve searching to adapt the presentation to the user and also to the environment. As the cases that the system will have to adapt may lead to infinite, it is not viable to have all the possible solutions stored in memory. Therefore, this evolution requires learning, because somehow from an initial point the system has to be able to recognize new cases and infer its solution. Here is where the techniques of artificial intelligence appear [Vitrià, 2001].

There are many techniques of AI that give support to this learning, such as Bayesian Classifier, Statistic Models, Decision Tree or Case Based Reasoning [Batlle, 2003]. In our solution we have preferred to use this last technique (CBR) as the learning mechanism, because is the one which is more alike to human reasoning and because the reasons we will see along this chapter.

Kolodner proposed the Case Based Reasoning (CBR) in 1983. This model bases the solution of the new incoming cases from the solutions of the similar cases that are already stored in memory. It is a system that simulates the human behavior, which is that new problems are solved from the experience that we have of the ones we have lived.

The learning process of the Case Based Reasoning is based in the 4-process cycle [49]:

- RETRIEVE the most similar cases;
- REUSE the cases to attempt to solve the problem;
- REVISE the proposed solution if necessary, and
- RETAIN the new solution as a part of a new case.

We can see this cycle represented in the next schema (fig. 5):

![Figure 5: The CBR Cycle [49]](image-url)
The stage of retrieve is made from the cases that we have stored in memory, and the cases retrieve are the ones that look like the new case. By intuition it is supposed that if two problems are similar, the solutions should be similar too. The algorithm most frequently used to infer the most similar cases is the one of the nearest neighbor. Is an algorithm of similarity that has the next equation:

\[
\sum_{i=1}^{n} w_i \times \text{sim}(f_i^r, f_i^p) \\
\sum_{i=1}^{n} w_i
\]

Where,
- \( w \) → importance of the weight of an attribute
- \( \text{sim} \) → function of similarity of the attributes
- \( f \) → values for the attribute \( i \) in the old case and in the new one

Once the system has retrieved a set of cases that are approximated enough to the new case, it has to be decide if the solution of the retrieved cases serve to the actual case. If one of the retrieved cases contains a solution that is completely correct for the actual case, it will simply give that solution. If not, there will be necessary to adapt the extracted solution (or make a mix of the different solutions) to find the correct one to our problem. This is known as the reuse stage.

When we already have an inferred solution, there comes the revise stage. The ideal is to revise if the solution is correct in the real life, but if it is no possible, the revision has to be done in a simulation. If the solution fails during the revision, it is necessary to repeat the previous stages adapting the solution in a different way.

If the solution founded is correct, we go the retain stage. In this step, it is stored the new case as well as its solution. That way concludes the learning process.

The machine of the Case Based Reasoning that is used in our system solution is written in Java language [Appendix C]. Being written in Java, it can be ran in any platform having the Java Virtual Machine (JVM), and so, we assure the portability of the

---

**New Package Query**

![Diagram](image.png)

Figure 6: Sample of a algorism followed by our borrowed Case Base Reasoning machine. Looks for existing cases and looks for similarities.
application. Besides this language is oriented to objects, thus we don’t loose any of the advantages that this type of languages offers.

In our solution, the access to the service and the results obtained will be made through the Web. That way we avoid that the customer has to install himself any application in concrete, because the majority of the new devices that can be connected to Internet are equipped with a browser. Besides, all the logic of the application is made in the server, so that the devices that have low power processor, such as mobiles, will not have any problem, they will only need to make web requests and receive the results.

The web server used is Apache Cocoon [14]. This server is framed into the Apache Project and it is web publishing framework written in Java and based in the XML technology.

The big difference of Cocoon with respect to the rest of the web servers is the separation between the content and presentation. Usually in a normal file that is in HTML in a server, the content as well as the way it will be visualized in the navigator are in the same file. Cocoon uses the XML concepts: the content is represented in a set of files and the way to represent them in others. Thus, once the stylesheets are generated the transformation (XSLT) from the same origin of data we can obtain different formats: HTML, PDF, SVG, WML, etc. [31]

This separation is perfect for our solution because it let us have all the information in the database, and then depending on the device apply a stylesheet or some other.

Aside from the advantage of the separation between content and presentation, while being functioning on a servlet container de Java, Cocoon also let us execute Java code in its Java Scripts and its eXtensible Server Pages (XSP). XSP is used to build dynamic XML content. Besides, also it is possible to access directly to the data base where the information is stored, so that from the Cocoon we have access to all the needed components in the application: the data base, the reasoning machine in Java and access to the stylesheets of transformation.
In this illustration (see fig. 7) we can observe how the client-server interacts. In the server side, we have a DB Server which will contain:

- the gathered information from users, devices and environments,
- and also another DB containing the Case Base (those cases solved in the past, having the query and its answer)

This DBServer is connected to the Java Case Base Reasoning Machine which will be asked by Apache Cocoon to give answers to new queries.

2.7 Summary

In this chapter we have got a deeper sight of the adaptive hypermedia, studying the six different fields we can divide adaptive hypermedia in and the objective each of
them has. Also we saw we can adapt the presentation and the content and how can we take this tasks.

Related to profiles, we saw which profiles we can use to create an adaptation. Those were, user’s, device’s and environment profiles.

After dividing the gathered information I fields in each of those three scopes, we saw how can this information be acquired.

Finally, we got an introduction to ubiquitous computing and which technologies have we used in our application implementation and why.
3 Software engineering approach

As it was said before in chapter two, our application will be an adaptive service, showing some information to the user, having in mind his profile and also his device and environment profile. In this chapter we will get an overview of our web application and also a software engineering study.

3.1 Service description

As we already have explained before, the application we will use to exemplify the problem and our solution, consists of a service that is accessible through the web. The service content as itself will be adapted to the user profile, and the way to present it to his device and his environment.

The service at issue is a planner of activities to be done during a whole day. That means, that the user (who already has a defined profile stored in the database) asks to the system to show a set of activities to do among a starting hour and a finish hour. It also indicates how many persons will do the activities and the approximated budget per person. This kind of service is included inside the on-line informational systems, explained in the State of Art section. We have chosen this sort of service because can offer good solutions, and we don’t need a huge knowledge about the hyperspace and the hyperspace is not a big one.

The system, from the user’s profile and the requirements given, has to give back a pack of activities that could be adjusted as much as possible to the user and its request.

Once the user chooses one of the packs presented, the system will search for the multimedia information related with each one of the options showed in the pack and will do a presentation related to the device and the environment where the user is.

3.2 Viability and utility

The content of this project has a high theoretic frame and contains a wide range of fields of study, besides many of the theories take as reference new technologic advances. Thus for instance, we can talk about new mobile devices with multimedia capacity, or the sensors in any point of a room that indicates the local position as much as the global of a person, and infinity of new technologies that even there are already in the market are not extended enough by the moment.

For this reason, because of the lack of resources and the goal of our project, it is necessary to delimitate the reach of the project from the beginning and keep in mind where the implementation will arrive and where not.

The utility of the project itself is to show a possible solution to the problem of contents adaptation to the user, using new tools and interrelating concepts of many disciplines. Therefore, the implementation is not a definitive solution, but an approximation.

The functionality of the chosen service is trivial for our purpose. It is not important the content itself, but the way it is adapted. It only works to exemplify the global behavior, being applicable to any other service. In the same way, the fact of have chosen one type of presentations or any other does not mean that it would not be extended to any type of format, since the architecture allows that new stylesheets can be developed without too much effort.
3.3 Analysis of requirements

In this section we will focus on which requirements are needed for the system and we will see which are the limitations that this offers.

Focusing on the service, there are two basic requirements for the application to be filled: the information in the database about the different activities that can be made should be inserted; the multimedia objects that will be used to make the presentation have to be also loaded in a repository, as well as its information in the database with the URL of its physic location.

Another basic requirement to make the system work properly is that the devices have to give real information about their capacities. Due that is possible to modify the header [48] of the web requests, navigators can induce the system to believe that they belong to a device when actually they belong to another, as well as it may happen in the UAProf, that indicate that they have capacity to reproduce one type of file and in fact is not truth.

We want to highlight some limitations. It is not technically possible to dispose of locating sensors, light, movement, etc., so, all this information has to be provided manually by the user to our prototype, choosing the type of environment where he is. If all this technology were reachable, the system would be able to adapt the presentation fully without the need of the user intervention.

In the same way, since the majority of the devices do not give information of its capacities, the user has to be in charge of providing it. In a close future, where all the devices could be identified and all their capacities be acquired, this step will also be done automatic, and the user will not have to take care of his device configuration.

3.4 Sequence diagrams

Plan a day

The user, having seen the option on his display (Interface), selects “Plan a Day”. The system will change the interface for a new one asking the user for logging in. This sequence will be repeated until the user enters a valid login-password tuple. The system will know that the tuple is valid checking it in a DB (MySQL).

In case the user has been correctly validated, the system will access again to the DB to check which devices and environments this user has declared. This is needed to create another new page where the user will have to select one environment, one device, and some relevant information to make a service search (budget, nº of people, starting hour, ending hour).

The system sends this information to our reasoning machine and this last will search similar cases in the Case Base (MySQL).

The system creates 3 new packages proper to the user and creates a list with the 3 packages given by the reasoning machine and the 3 ones it has just created and sends this list to the interface.

The user selects one of them and after the system knows which one has been selected, it makes a query to the multimedia DB asking for the identifiers of the needed files in order to create a presentation (taking into account the device and environment preferences).

The system creates a new entry in the DB located in the “Query Presentation” table with the need multimedia identifiers, the appearance order, etc.

Finally, the system gets the entry information and creates a presentation in the user display.
We can see the sequence diagram for the “Plan a day” option in figure 8.

Creating a Presentation
When a presentation is needed, this is the process done:

The XSP file will be executed, will get all the required multimedia files and from them, will create a structure and will be transformed to an XML file. This file will be processed by the XSL stylesheet and will get the proper format, which will be displayed in the interface. (fig. 9)
User registration

When a user wants to be registered in our service, he will have to fill a JXForm [15] (a Cocoon approximation to the new standard XForms [12]). The system, after having received this information, will insert his information in the DB and send a feedback presentation to the user display. (fig. 10)

Other Kind of Registrations

This sequence diagram is alike the one before. The only difference resides in the user authentication before any insert is done into the DB. (fig. 11)
3.5 Interface Design

The interface for the application has been thought to be as much adaptable as possible to any device, by this way we will keep the homogeneity of the web apart from any device. [Cooper, 2003]

As long as all the interface is designed from the same stylesheet, same structure is also kept from one section to the next one, thus once the customer is familiarized with the design, it is easier for him to navigate through the website, since he quickly knows where he has to search for the information in each moment.

3.6 Scenarios

3.6.1 Scenario “group scheduling”

A group of friends decide to join together at Juan’s place on Friday at the afternoon to decide what they can do next day during the whole journey. They are tired of going always to the same places, and doing the same things, they all are young people, around 20 and 25 years, they all study computer science, they like to visit night clubs and bars, they like sports, specially football, they like having adventure and as they are all students, they do have a big budget but they agree to spend 80 € each. They have a computer at Juan’s place, so they decide to use the activities planner with the profile of Juan, which was already registered and as they all have almost the same likes and interests, they don’t mind to let him do the research.

The system realized that the starting hour is at 8:00 so it suggests to start having breakfast in a small coffee shop near an amusement park that they didn’t know that was rebuilt, then they will go to spend 4 hours in the park, the system realizes that they could also go to play football before going to the park while de breakfast comes down near a club right in front of the coffee shop. In the profile of Juan there is a preference to oriental food, so it suggests to go and have lunch at 15:00 in a Chinese restaurant in a mall where there is an exposition of technology, these activities during the afternoon and at night there is the option to go to a very fancy night club that was in the other side of the city and they didn’t know.
3.6.2 Scenario “mobile tourist information”

Ali is having a business traveling in a city where she had never been before. Her flight to go back home is at 8 in the morning, but when she was just about boarding the plane, she was told that her flight was canceled for technique problems and it will be postponed until the next they at 7:00 pm. Ali feels happy because she thinks that it is a good chance to know the city. However the firm will pay. As she has no idea of what she could like to see in the city, she decides to use the service of the activities planner. She has used it previously in her home city, so she does not have to specify the preferences. She does not have a Laptop, but she has a high technology mobile with Internet access. She gets connected to the webpage that offers the service, and she picks the city, introduces her user name and password. She will to spend 100 euros, the starting hour is at 10:00 and the finishing hour at 18:00, so she may have time to take her flight.

The service, has stored the information of the activities that may be done in the city, it realizes that Ali likes art a lot, and the there is a very famous art gallery in that city. As the schedule matches, it decides to send Ali to see the exposition from 10 to 13. It also notices that when she goes out from the exposition it will be time to have lunch, and that she will have to eat, thus it suggests going to an Italian restaurant, because she doesn’t like fast food. When she finishes her lunchtime, it could be 16:00 hrs, so considering that Ali likes literature it suggests to go the National Library, the biggest in that country, and recognized as one of the oldest in the world.

Once the system finishes the offer of the pack, as we have seen before, it shows it to Ali, with some other packs with different options, and when she picks one of it, she will be able to see a multimedia presentation of the tour she will have and with color because its mobile let her do that, but without sound, because she is still at the airport and she doesn’t want that everybody get aware of what she is doing.
4 Analyzing the application

4.1 Architecture

The proposed system will be developed on the basis of a client-server architecture.

To help the devices that have low calculation capacity, the server is in charge to make all the operations.

All the information related to the profiles, services, multimedia objects, stylesheets and so on, is stored into a MySQL database. We have chosen this type of database server because of its great potential and also because it is free of charge, as well as all the tools used in this project.

As it has been mentioned before, the interacting with the user side is done in a website. So, the user only needs a browser installed in his device to be able to start to use the application, and some other additional software if he wants to watch the multimedia presentations, for example an SMIL display. Anyway, if he does not have this reproducer, the presentation can be done directly in the web.

The server in charge to collect the web requests is Apache Cocoon. This service works on Java, and it needs a Servlets container to be executed. This Java servlet container is Apache Tomcat, also included in the Apache project.

We can see a representation of the system’s architecture in the next illustration (fig. 12):

Figure 12: Architecture Diagram in detail
4.2 The reasoning machine

The case based reasoning machine used in our system comes from an open source project named The Selection Engine [50]. It is written in Java language, so it lets us execute it in any platform that supports the Java Virtual Machine, and as it is an open source, it has been possible to modify it to make it adaptable to our needs.

In the Appendix C more information about the selection engine will be detailed, here we will explain the most relevant ideas.

The selection algorithm used in our application is the most common of all of the case based reasoning algorithms, already explained before: the nearest neighbor algorithm.

The first thing that the reasoning machine does is to read the case base values. In our project, these values are in the database. Once all the characteristics and the old cases values have been charged, it reads the new case to study also from the database.

There are two types of characteristics: those which are restrictive and those that aren’t. An example of a restrictive characteristic may be needed age to get in a place, if you are not old enough you will not be able to get in, even it is very close to you. A non-restrictive characteristic, as can be the preference of an sport, as more similar it is, more importance it has, but not being the same does not mean that this case is not valid.

By this way, once charged the old cases and the new one, they will be passed through a filter to eliminate all the old cases that do not fit with some of the restrictive characteristics. Then from this result, it goes to the similarity filter. There is where the remaining cases are ordered by the percentage of similarity to the current case. The more similar old case to the current one, will appear in first place.

In our application we use the reasoning machine in two different actions: when a new user is registered and when a registered user plans a day.

In order to split the users into different groups, we need to know how is the user similar to each group. To do that, the reasoning machine gives us this functionality. The case base contains information of each group. In concrete, the relevant information is about the preferences of the users. The values stored are the average of each group member preferences. When a new user is inserted, the reasoning machine looks for the group that is more similar to the user. If the similarity is over than 90%, the user will belong to this group, and the average will be recalculated taking into account the new values. If there is no group similar enough to the user, a new group is created in the case base. (See fig. 13)

Figure 13: Algorithm followed by the reasoning machine for new users

The reasoning machine is also used in order to retrieve the cases more similar that the actual case requested for planning a day. (See fig. 6)
When a user requests for an advice of a new pack for a day, it looks into the database from the cases that are similar to the request of such user, having in consideration his preferences and the marked options.

4.3 The intelligence in the system

First of all, we have to look at what can we understand as artificial intelligence or simply intelligence. After reading several books and papers related to this topic, it is possible to have many different definitions. For example, [51] defines artificial intelligence as “the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable”. Through this definition, the next question is about what is intelligence itself. He also defines intelligence as “the computational part of the ability to achieve goals in the world. Varying kinds and degrees of intelligence occur in people, many animals and some machines”.

In other words, we can say that a computer program has intelligence if this program has any ability to achieve goals in the world. Many different techniques are used in order to reach this purpose. One of them is trying to simulate the human’s behaviour within the computer. For example, we can try to simulate the human learning and make a program learn and take decisions from previous experiences, like humans do.

In our system, we have decided to implement a program with this philosophy. The actions that in the future the program will take depend on the actions that the user does now. When a user with a concrete profile asks for a new plan for a day, first of all the system looks for the cases more similar to the actual query, in order to give him

![Figure 13: Algorithm followed by the reasoning machine for new users](image-url)
some packages that other similar users have chosen. The criteria to calculate the
distance between the actual case and the old ones takes into account the importance of
the different properties that the profile has. This importance is called the weight of the
property.

Initially these weighs have the same value. As users are separated in groups,
depending on their profiles, the value of these weighs are stored in the group
information. When one user of the group chooses one plan rather than other one, all the
weighs of the services included in the plan are incremented in their values. By this way,
when in the future a user from the same group asks for a new plan, not all the properties
will have the same value for him. The system will believe that the characteristics that
other users in the same group have chosen are more important. The system will modify
the results presented to the user according to the actions that other users do.

Moreover, not only the most similar cases are presented. The system also
generates a new solution that fits completely into the user and his request. We have
created an algorithm that searches all the activities available for the user in any moment
and, from his likes and dislikes, chooses one. This algorithm is able to create a complete
plan for a day, according to the opening hours, the age, the number of people and the
time spent in each service. For more detail about the algorithm, see Appendix A.

The last “intelligent” component in our system is the dynamic generation of the
multimedia presentations. There is another algorithm that from the number and type of
multimedia objects, and taking into account the user’s device and environment, makes a
dynamic presentation in SMIL or HTML. It is possible to get more information about
this algorithm in the Appendix B.
5 Implementation in use

We will find some relevant information about the application implementation here. Since the implementation is too much large to be given here, we will only find the most important aspects. For more information about the implementation, see Appendix or look at the application source code.

5.1 The Data Base

In this section we are going to show how is the database structured and the properties of the different fields.

User Profile

This MySQL table is called “User_Profile”. This table contains information about the user, his/her demographic data and his/her preferences. A user is recognized by his nickname. The preferences of each field are a number between one and five:
1 the user hates this category
2 the user doesn’t like this category
3 the user likes a little this category
4 the user likes a lot this category
5 the user loves this category

Each user has a “group_identifier” and shares some values with the other people in the same group. One user belongs to one group if his profile is near enough (90%) to the average of the profile in this group. Then, the average is recalculated with the new data.

Group profile

Here is the information shared by users in the same group. The most important information is the categories weights, which will be used to refine the queries in the case base when a user asks for a new service.
**Device**

This table stores information of each device that a user has. After a user is registered in the system he is able to register all his devices with different properties. The user can choose the name of each device.

<table>
<thead>
<tr>
<th><code>device</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name</code> varchar(20)</td>
</tr>
<tr>
<td><code>user_nick</code> varchar(20)</td>
</tr>
<tr>
<td><code>screen_size_x</code> int(10)</td>
</tr>
<tr>
<td><code>screen_size_y</code> int(10)</td>
</tr>
<tr>
<td><code>sound</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>color</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>image</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>video</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>smil</code> int(1) unsigned</td>
</tr>
</tbody>
</table>

Figure 16: DB Description (Device)

**Environment**

Like the device table, the user can also register his own environments and give to them an easy name to remember them.

<table>
<thead>
<tr>
<th><code>environment</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name</code> varchar(20)</td>
</tr>
<tr>
<td><code>user_nick</code> varchar(20)</td>
</tr>
<tr>
<td><code>sound</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>lighting</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>completed</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>speed</code> int(1) unsigned</td>
</tr>
<tr>
<td><code>bandwidth</code> int(11) unsigned</td>
</tr>
</tbody>
</table>

Figure 17: DB Description (Environment)

**Service**

It includes information of all the services registered in the system. Another company, web services or any other entity should fill this information.

<table>
<thead>
<tr>
<th><code>service</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>identifier</code> int(11) &lt;pk&gt;</td>
</tr>
<tr>
<td><code>name</code> varchar(20)</td>
</tr>
<tr>
<td><code>sector</code> varchar(20)</td>
</tr>
<tr>
<td><code>subclass</code> varchar(20)</td>
</tr>
<tr>
<td><code>age_min</code> int(2) unsigned</td>
</tr>
<tr>
<td><code>age_max</code> int(2) unsigned</td>
</tr>
<tr>
<td><code>price</code> int(10)</td>
</tr>
<tr>
<td><code>start_hour</code> float(9,3) unsigned</td>
</tr>
<tr>
<td><code>finish_hour</code> float(9,3) unsigned</td>
</tr>
<tr>
<td><code>people_min</code> int(2) unsigned</td>
</tr>
<tr>
<td><code>people_max</code> int(2) unsigned</td>
</tr>
<tr>
<td><code>dist_x</code> float(4,2)</td>
</tr>
<tr>
<td><code>dist_y</code> float(4,2)</td>
</tr>
<tr>
<td><code>time_spent</code> float(9,3) unsigned</td>
</tr>
</tbody>
</table>

Figure 18: DB Description (Service)

**Multimedia Object**

As well the service information, in the database there is information about the multimedia objects of each service and its properties. This information must also be provided by other entity.

<table>
<thead>
<tr>
<th><code>multimedia_object</code></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>identifier</code> int(11) unsigned &lt;pk&gt;</td>
</tr>
<tr>
<td><code>service_identifier</code> int(11) unsigned</td>
</tr>
<tr>
<td><code>type</code> int(2) unsigned</td>
</tr>
<tr>
<td><code>size</code> int(11) unsigned</td>
</tr>
<tr>
<td><code>duration</code> int(11) unsigned</td>
</tr>
<tr>
<td><code>url</code> varchar(100)</td>
</tr>
</tbody>
</table>

Figure 19: DB Description (Multimedia Object)
**Base Case Content**

This table is used for the reasoning machine. It contains information of all the petitions that have been done in the past. When a user selects a new package of activities, the query is stored in this table, and the results in some entries in the table “packs”.

**Packs**

<table>
<thead>
<tr>
<th>packs</th>
<th>content_identifier</th>
<th>int(11) unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>order</td>
<td>int(11) unsigned</td>
</tr>
<tr>
<td></td>
<td>service_identifier</td>
<td>int(11)</td>
</tr>
<tr>
<td></td>
<td>start_hour</td>
<td>float(9,3) unsigned</td>
</tr>
<tr>
<td></td>
<td>finish_hour</td>
<td>float(9,3) unsigned</td>
</tr>
</tbody>
</table>

Figure 21: DB Description (Packs)

Here he can find all the activities we can do in a concrete plan of a day. The field “order” orders the activities. Also it has information about the start and the finish hour of each activity as well as the service identifier.
5.2 Interaction with the user

One of the most important parts of a project is what the user sees. The user and computer interaction has to be taken into account when an application is under development, so the user will find all the functionalities according his model [Cooper, 2003].

However, that wasn't our main goal, so we only have created a basic interface so the user can find the most important features in the web application.

User Welcome

When the user reaches our web address, he will be asked for choosing one of the following 4 options:
- Plan a day Go out!!!
- Register new users
- Register new devices
- Define a new environmental

The first selection will lead the user to find some services packages and select the one he likes most.

The second option is the user registration process. That's the first step a new user has to do.

In the third option, the user will have the possibility to register new devices (Computers, PDA's, Mobiles...)

The fourth one is similar to the one before, but the user will be able to define new environments selection some presentation preferences related to each environment.

What you can do

Plan a day Go out!!!
Register new user
Register new device
Define a new environment

Copyright © 2004 Didac and Dani project
Figure 22: Application Screen Show (Main Window, options in the service)
Plan a day Go Out!!!

In this section, the user will have to log in, so he can enter to the intelligent services search.

This log in is not used, in our application, for security reasons, it's only to get the user profile.

Afterwards, the user will have to select the device, the environment, how much money is he going to spend, how many people is going to join the activities and the starting and ending hours.

The two first parameters will be avoided in a future, when all the devices are able to send its UAProf and it will also be possible to get the environmental information using the device and environmental sensors.

With all this information inserted, the user will get the three most adapted-to-him packages. Moreover, three new packages will be created to the user, in case he would prefer a one it fits better to him.
When the user has taken his decision, he will click on the one he prefers and a presentation will be displayed. This presentation will be adapted to the device he has selected and also thinking in the environment where he is. 

Figure 25: SMIL Presentation Screen Shot.
The possible outputs are SMIL, HTML.

In an SMIL presentation, there will be from 1 to 4 regions, each one containing a multimedia file or some text. Also it will be always a bottom region containing the service information. As a video can contain audio also, it's never going to be played audio at the same time as video. Each slide will spend the needed time for the most large file, that is, if we have two videos, one of 12 seconds and another of 4, the slide will last 12 seconds.

In case there are only pictures, it will take 5 seconds in normal mode, or 8 seconds in slow mode.

In HTML, videos aren't embedded. That's because an HTML presentation can have several videos, and it would take a lot of time to load them all. So, an html presentation will have links leading to the video files.

**Register new user**

That's the first step a new user should do.

This application is user oriented, so we need to have some information about the user preferences. Here is where it's going to be done.

The user has to fill in his personal information such as nickname, password, age, birthday and e-mail.

Also he is asked for some preferences in different scopes. This value is select between 5 levels of preference: *I hate it, I don't like it, Just a little, I like it* and *I love it*.

All this information is needed for the service selection.

The user will be identified by his nick-name and password. The scope preferences will be used to select the most enjoyable service for the user when he ask for a planed day, but also the sex and the age are required to select the services, the reason is because there are some services (or there could be) services oriented to a gender and we can find some services which are not allowed to people under a certain age.

The e-mail is asked for future developments. It could be used to send a confirmation to the user once the service reservations are done (that's not contemplated in our implementation) and for contacting with other users in our same group when we wanted to create a group activity or meet them just for knowing people like them.
Register new device

In this section, the user will create a new entry for a device he owns.

This information is used to create the package presentation. Knowing the device capabilities, it will be possible to create the presentation which fits the best to each device.

The user will have to specify screen size and some other capabilities used to select the multimedia files. Those other capabilities will be color, sound, image, video and SMIL.

Also, it's asked to identify the device with a name, so the user will be able to select his device in the Go out section.

With UAProf arrival in all the devices, this section could be removed, because device information will be acquired from the device itself.

By now, if the user is browsing this webpage with a device using UAProf (new mobile phones), the form will be auto filled.
Register new environment

As there is no way yet to acquire the environment information from the device, we have done a formulary where the user can create so environments. Those environments will be used in the service package presentation, although, in a future, they can be also used to select some services in the showed packages (just for avoiding some outdoors activities in a rainy day, for example).

The user, will have to identify himself to relate the device with his profile, give a name identifier for the environment and fill the information required.

This information is:

<table>
<thead>
<tr>
<th>Fields</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>Yes / No</td>
</tr>
<tr>
<td>Light Completeness</td>
<td>Normal / Little Light</td>
</tr>
<tr>
<td>Speed</td>
<td>Normal / Slow</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>Modem or less / ADSL 256 / More than 1 Mbps</td>
</tr>
</tbody>
</table>
After having an environmental analysis, and thinking on the most important ones, we defined the five fields above as the discriminator for each environment. Below we can see an example of each environment that we can find.

<table>
<thead>
<tr>
<th>Env. Name</th>
<th>Audio</th>
<th>Contrast</th>
<th>Completeness</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>yes</td>
<td>normal</td>
<td>yes</td>
<td>normal</td>
</tr>
<tr>
<td>Meeting</td>
<td>no</td>
<td>normal</td>
<td>yes</td>
<td>normal</td>
</tr>
<tr>
<td>Night</td>
<td>yes</td>
<td>high</td>
<td>yes</td>
<td>normal</td>
</tr>
<tr>
<td>Movement (for ex. train)</td>
<td>yes</td>
<td>high</td>
<td>yes</td>
<td>slow</td>
</tr>
<tr>
<td>Summary</td>
<td>no</td>
<td>normal</td>
<td>no</td>
<td>normal</td>
</tr>
</tbody>
</table>

Home: There is no special handicap.
Meeting: It's required to be quiet in a meeting, in order not to disturb the rest of the employees.
Night: In darkness it's difficult to read, so it will be a help to use high contrast.
Movement: Joggle is present. If we use an slow presentation, the user will more probably be able to read / watch what is displayed.
Summary: Is we don't want to spend much time in a presentation, because we have had just a little break in our job, we would like to see only a summary of the asked information.

Also, we decided to add the bandwidth field in the environment scope because, even it can be limited by the device, most of times it's done by the connection we have in each environment. If we think on a laptop, it's simply to realize that the bandwidth is always depending where we are connected, at home we can have an ADSL connection instead of a broadband used in our office.

Because of simplicity, we avoided to add this field in the device formulary, because almost never we are going to find a device having less speed capabilities than the environment.
NOTE: It's possible that the first hit to our webpage takes some time. That's because Tomcat, the servlet engine. This servlet doesn't know which java services are going to be used, so it doesn't load Cocoon until it's called. So, the first hit will have to wait Tomcat create all the .class files for Cocoon. This could be solved making a pre-load of Cocoon adding these lines in the web.xml file located in /CATALINA_HOME/webapps/webdav/WEB-INF

```xml
<servlet>
    <servlet-name>Cocoon</servlet-name>
    <display-name>Cocoon</display-name>
    <description>Cocoon</description>
    <servlet-class>org.apache.cocoon.Cocoon</servlet-class>
    <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
    <servlet-name>Cocoon</servlet-name>
    <url-pattern>/servlet/ourCocoon.html</url-pattern>
</servlet-mapping>
```

The url_pattern is used to access to the servlet by the browser path. The number used in load-on-startup will tell Tomcat in which order has to load the servlet in case we have more than one.
5.3 Processing the information

Within this section and the next one we are going to expose how the information from the user is gathered and processed.

Cocoon works with the idea of pipeline. When a new request is done to the web server, the URL has to match with an entry in the sitemap file. This entry in the file tells Cocoon which is the path that the request must follow. For example, one path could be to process a XSP file and then go through the specified XSL, or simply read a image file.

Also it is possible give the control to any script file (control flow).

The data flow in some requests is controlled basically by this Cocoon control flow[36]. It has the ability to describe the order of Web pages that have to be sent to the client. The control flow is made in JavaScript language, but it is possible to make Java calls to object methods inside it.

Cocoon also allows an advanced control in the forms. The technology used is called JXForms[15] and pretends to follow the XForms[12] standard from the W3C. XForms “defines a device-neutral, platform-independent set of form controls suitable for general-purpose use”[56]. JXForms supports a subset of these controls. Doing that we can kept the separation required between the data and the presentation.

To assure the information is correct, JXForms provides declarative form validation using the Schematron[46-47] assertion language. In a simple XML file we can define all the rules for different forms and link this file in the Cocoon pipeline.

Inside the flow, it is necessary to define a data model to be filled in the form page. Once the information is submitted, the flow script receives this data model. In order to generate dynamic pages, we use XSP. This page processor allows to execute Java code and the results are transformed into a XML file. The XML resulted is transformed with the appropriate XSL and presented to the user.

In order to demonstrate better this process we can see a concrete example. We want to focus in the “Plan a new day” flow because is the most relevant in the application. Below we can see the pipeline in charge of process the “newservice” request.

```
<map:match pattern="forms/newservice/">
  <map:call function="jxform">
    <map:parameter name="function" value="newservice"/>
    <map:parameter name="id" value="newservice"/>
    <map:parameter name="validatorNamespace" value="http://www.ascc.net/xml/schematron"/>
    <map:parameter name="validatorSchema" value="forms/validation/schematron.xml"/>
  </map:call>
</map:match>
```

This pipeline notifies Cocoon that the request will be processed with JXForms, and the control flow will be made by the function newservice. Also informs the validator that will be used, in this case Schematron, and where is the file with the rules. The pipeline is processed and the function newservice takes the control.

First of all, the user must log in. To do that, once the flow script has created the data model (the user name and password), the web page asking for the log is sent:

```
form.sendView("forms/newservice/login.xml");
```
This XML is the JXForms file, but is transformed by the XSL to allow the browsers to understand it. When the user submits the form, the information is validated by Schematron[46-47]. If all the fields are correct, the control is returned to the flow. There, the flow accesses to the data base and checks if the user and password are corrects. If they not are correct, the forms is sent again with an advice.

We suppose the user is valid. In this case, the flow script must access again to the data base, and retrieve the information about devices and environments that the user has registered. Then, another form is sent to the user in order to complete the request.

```
form.sendView("forms/newservice/newservice.xml");
```

In this form, the user must select his actual device and environment as well as the options required in the service. When it is filled, the data goes to the flow script again. With the options selected, the system calls (Java) the reasoning machine to retrieve the most similar cases and also generates new solutions. The results are sent to a XSP file in order to be presented to the user.

```
cocoon.sendPageAndWait("forms/newservice/showpacks.xsp");
```

The XSP file processes the data and presents a list with all the possible plannings. Once the user chooses one, the flow scripts takes the control again and prepares the multimedia objects to be presented, according with the user's device and environment. Then, the flowscript calls to the appropriate pipeline.

```
cocoon.redirectTo("http://goout.servehttp.com:8080/cocoon/project/presentationsmil?id="+id_query_presentation+"&id_content="+id_content+"");
```

From this point on, a new pipeline takes the control of the application and is the one in charge of construct the appropriate presentation.

### 5.4 Presenting the service

There are two pipelines in charge of doing the presentation: one for the SMIL and another one for the HTML presentation. As an example, below we can find the one used for the SMIL presentation.

```
<map:match pattern="presentationsmil">
  <map:generate type="serverpages" src="presentation/presentation.xsp">
    </map:generate>
  <map:transform src="stylesheets/smil/stylesheetsmil.xsl"/>
  <map:serialize type="smil"/>
</map:match>
```

The process in this pipeline is done by the presentation.xsp file. The parameters identifying the presentation and the content are sent to the pipeline and requested inside the XSP file. It has to access to the data base and retrieve the information generated
before by the script file. Dynamically goes through all the multimedia files that have to be presented and schedules a layout for the SMIL presentation.

The presentation will show sequentially the services. At maximum, it can only be 4 multimedia objects at the same moment in the screen. If one service contains more than 4 multimedia objects, from 5 to above will be showed in the next slides.

In this step, the system takes into account the user device and environment. If the sound is not activated, the presentation will not play the audio in the videos. If the user wants a slow presentation, here is where the system control the time. Also the screen, text and pictures size are defined in this process.

After this process, is generated a XML file with all the information needed to construct a SMIL file. The stylesheet file is the ones in charge of transforms this XML in a correct SMIL file.

The resulting SMIL file is sent to the client’s browser. If the presentation contains any text, it will be created in RealText format. The link with the properties of the text is included inside the presentation. When the SMIL player plays the file, the link with the realtext will be called an the Realtext sent to the client.

There is one pipeline in the sitemap definition in charge of the RealText generation. We can see the code below.

```xml
<map:match pattern="realtext">
  <map:generate type="$serverpages" src="presentation/rt.xsp">
  </map:generate>
  <map:transform src="stylesheets/smil/rt.xsl"/>
  <map:serialize type="rt"/>
</map:match>
```

This pipeline receives parameters from the request: the identification of the text in the data base and the screen size. This parameters are sent to the XSP, processed and the XML result is transformed according to the stylesheet. The RealText generated is returned to the client SMIL player.

The HTML presentation is quite similar to the SMIL one, but it has another pipeline, with different XSP and XSL files. Instead of putting each service in one slide, the HTML presentation split the services in different sections. Videos and audios are not directly played, but the user must select which ones wants to play.

### 5.5 Running the application

Juan saw there is a new service on the net which offers you a plan for a day thinking on the user preferences. As he is a computer scientist, he enjoys web services, so he decided to register in.

First of all, he went to [http://goout.servehttp.com:8080/cocoon/project](http://goout.servehttp.com:8080/cocoon/project) and he clicked on *Register new user*. Once there, he entered *Juan* as his nickname and *piano* as his password. His mail is *juan@juanserver.cat*, he is a male person, born *1980-04-12*.

Then he filled his preferences according to his thoughts:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Value</th>
<th>Topic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sports</td>
<td>I love it</td>
<td>Shopping</td>
<td>I like it</td>
</tr>
</tbody>
</table>
After that, he clicks on Register a new device and creates a new device profile. He wants to register his personal computer at home, which has all the new technologies you could imagine. He writes his nickname and password, identifies the device as Computer and fills the formulary with the following data:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>1024</td>
</tr>
<tr>
<td>High</td>
<td>768</td>
</tr>
<tr>
<td>Color Capable</td>
<td>Yes</td>
</tr>
<tr>
<td>Sound Capable</td>
<td>Yes</td>
</tr>
</tbody>
</table>

And also creates a new environment describing his bedroom by going in the Register new environment section. This new environment will be called Bedroom, and has this information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio?</td>
<td>Yes</td>
</tr>
<tr>
<td>Light</td>
<td>Normal</td>
</tr>
<tr>
<td>Completeness</td>
<td>Completed</td>
</tr>
</tbody>
</table>

Two days after, 3 friends of Juan come to meet him and they decide what are they going to do for the following day. Juan wants to try the new service, and his friends let him do it. He goes direct to the Go Out webpage and clicks on Plan a day Go Out!!! After writing his login and password he gets into the service specification site. As he only has defined 1 device and 1 environment, the selection done by the agent is the correct one, and he doesn't have to change anything there. They are 4 people, and they have decided to spend 80€ per person. That's what he writes in the form:
The system, after studying his query, gives him 3 possible packages they will enjoy. This system is very new and there aren't many users registered. So, his group is empty (he is the only one in his group) and also this is his first query in the system, so, the reasoning machine hasn't found any existing package suitable for him. Hence, Juan will only have the three new packages in his computer display. Juan has decided to choose the first package. The three ones are enjoyable on his point of view, because the activities have been selected thinking on his preferences.

There aren't existing packages suitable for you

Still you can choose a personalized packet:

1. Los Mañanitas -- Sector: gastronomy
   Start hour: 8:00 Finish hour: 10:00
2. Albatros -- Sector: sports
   Start hour: 9:30 Finish hour: 11:30
3. Technoav -- Sector: tech_science
   Start hour: 12:00 Finish hour: 14:00
4. La Mexicana -- Sector: gastronomy
   Start hour: 14:30 Finish hour: 16:30
5. Cinepolis -- Sector: movies
   Start hour: 17:00 Finish hour: 19:00
6. McDonald -- Sector: gastronomy
   Start hour: 20:00 Finish hour: 20:30
7. El Cubo -- Sector: night_life
   Start hour: 23:00 Finish hour: 24:00

Figure 29: Application Screen Shot (Package sample)

Now, he wants to see the package presentation. This presentation is an SMIL file that takes 5 seconds for each slide, using colors, videos, and some descriptions for each activity. Those proprieties has been chosen because his computer can afford them. When Juan registered his device, he specified his computer could play SMIL, was color capable and so on. Some other preferences has been controlled in an environment point of view. Each slide takes 5 second because his environment doesn't says he wants an slow presentation. Here we have some screenshots:
Figure 30: First SMIL Screen Shot (2 Regions consisting on text and an image + the informational region with hours and place name. SMIL file created dynamically)

Figure 31: Second SMIL Screen Shot (2 Regions consisting on text and an image + the informational region with hours and place name. SMIL file created dynamically)
6 Tests

After finishing the implementation, we made some tests with different devices, environments and users.

We should have a lot of users registered in our application to check if the reasoning machine is useful or not, but we could still see that the given packages were what the users expected.

As we couldn’t check the reasoning machine in its totality, we decided to check the adaptive presentation. So we used a Laptop, a PDA emulator and a phone emulator as devices, and we used distinctive environments to show important characteristics of each one.

In this first picture (fig. 32) we have selected a low resolution laptop able to display pictures, and videos, but unable to play SMIL files. That’s why we are seeing an HTML presentation. The environment used for this presentation had the best characteristics: good light, not summary, with audio, not slow speed needed and using a broadband connection.

As we can see, images has been added to the presentation, we have all the possible information for each service and text has a normal format. Even the device and the environment enabled us, this presentation didn’t have audio or video. That was only because of the selected services; those didn’t have any of those kind of multimedia files.

Figure 32: Using a Laptop in our application
Using a PDA with a meeting environment we got the following samples (fig. 33). We can see is has been acquired the head in que web query to know the used device was a PDA (Go out!! For PDA). The screen size is gotten for the device profile and used to resize the images to a proper dimension. According to the environment, text has been set in bold, because the user is having poor light in meetings.

Figure 33: Browsing Go Out on a PDA. Using a meeting environment.
Using a mobile (fig. 34) in a train we get this presentation. Images has been resized by Cocoon to fit to the mobile’s screen, videos has been avoided due to the long bandwidh the phone has in a train. Because of the clatter in a train, text has been set in bold. It’s not appreciated in those images, but information has been summarized according to the environment profile, so there as maximum, one text information and one image per service offered.

Different to the images before, be can see this other capture (fig. 35) using the same device, asking for the same information but beeing used in a different environment. The environment has been decisive to select text to be in a normal format.

We also did some tests with SMIL capable devices, as we can see in figure 25, 30 and 31.

Figure 34: Nokia emulator in a train environment
Figure 35: Mobile phone in an office environment
7 Conclusions

When we started to think about this thesis, we didn’t know almost anything about adaptation and its related fields of study. Our initial idea was only to adapt one service to the user from his profile and present it to him in a standard presentation. But when we started to investigate within the profiles and users we discovered the field of adaptive hypermedia. Reading some papers about this topic, we thought that they just were theories for the future, but not currently applicable. Especially if we are talking about the information related to the environment, where are needed many sensors and locators. Anyway, we liked the idea of adapting the services: the content as well as the presentation of it. Then we opted to do the thesis within these fields but without use advanced technologies; we wanted to make a general architecture as a possible framework, showing the different technologies that can be used to solve the problem.

Once we decided the goal of the project, we had to think about which technologies we were going to use. Initially our idea was to make all the project in Java, because we thought it was enough stable and almost all the devices can execute Java code. But we changed our mind because the application would need to be loaded and ran in the devices. We didn’t like this idea because some devices, like cell phones, have a very limited memory and process capability.

After this reasoning we thought about making the interface with the user through a web browser. Nowadays, almost all the devices are sold with a web browser. We looked for a web server and we chose one that we used in other project with very good results: Apache Cocoon. Also we wanted to adapt the content by using a case base reasoning, but the ones we found were written in Java. Fortunately Cocoon allows us to add this Java code inside the application. This is another reason why we chose work with Apache Cocoon.

Finally, we were conscious of we couldn’t use any new technology like a GPS or movement detector. But we wanted to explore new technologies and that’s why we decided to read the UAProf from the cell phones, to see what we could expect in the next years about devices specification.

After deciding all this points, we made the three questions presented at the beginning of the report. These questions were the basis for our research, and now is the moment to answer them.

The first question is: "Can we obtain good results using a case based reasoning in order to adapt the service to the user, from the feedback gathered by the system in previous situations?"

As we have explained along all the thesis, the case base reasoning is used in our project in two different actions: to insert the user in a appropriate group and for retrieving the similar cases to the actual.

In the first case, we can say that the reasoning machine is a very good tool because we can easily know how near is a new user to each group and take a decision according with the result. And this separation within groups is a good help in order to get a better adaptation to the user, because we can take into account the group information.

If we look into the other functionality, it is a little bit difficult to answer with a quite objective vision. The problem is that if we want to know if the options selected by some users are good options also for other users, we need to do some exhaustive tests with a wide diversity of users and it has not been possible by the time we had. Anyway,
according to the test done by the authors of the thesis, we could say that the results returned by the reasoning machine are quite good. This value is incremented if we change the service and the new one requires solutions that are difficult to get, and can be easily deduced by previous cases. In this kind of services, the case base reasoning gets a extra value.

The second question formulated is the next one: “Which are suitable technologies to support the task of adapting the way to present the services to the users?”

Our personal answer to this question is that saying Apache Cocoon and all its related technologies are suitable to our project goal is an outright affirmation. After being working with Cocoon during the last 8 months, we have a great opinion about this platform. We think they are running in the correct way, and they want to adopt all the new standards, like XForms. It is possible to have access to many different functions that make any task easy, and also allows to execute complex Java code inside the application.

With the philosophy of separation the content (XML) and the presentation (XSL), Cocoon makes easier the idea of adaptive hypermedia. Also is a good tool to create XML file dynamically, by using the XSP file, technology very useful because allows access to data bases and present the retrieved results.

The problem with this kind of applications open source and under development is that some functionalities are not completely developed, and may don’t work properly. Also the documentation is often hard to find, and we have to look into many forums to find a good solution to do some issues.

Finally, the last question posted is: “Which formalisms will be used in order to get a better understanding of the user and his/her environment?”

After using the UAProf, a CC/PP implementation for mobile devices, we have good expectations about this technology. We have been able to read the UAProf from a Nokia Emulator and the feedback has been positive. We have found many relevant characteristics, but for our goal only we needed a few properties.

It is necessary to make an exhaustive study of all the fields contained in this files, because there is a wide variety of properties. Doing that and parsing the file properly, it is possible to get a detailed description of the device.

Nowadays, the problem is that there are not many devices using CC/PP. We only have found some cell phones using UAProf. In the future, when all the devices use this kind of information files, it will be easier to make adaptive presentations, as well as contents, because it is possible to add information about the user inside the XML file.

As a last comment, only to say that we believe our initial goal has been accomplished. We have developed a general framework and architecture capable to adapt content and presentation, and we have demonstrated how useful can be to use the technologies presented in order to realize these kind of tasks.
8 The Future

The developed application is only a demonstration that we can use to answers proposed at the Identifying the problem section. It's only a prototype that can be improved a lot and have several features added. Also, this implementation can be modified to be used in other projects.

8.1 Nearer development

1. Be allowed to publish services and multimedia information using webservices

The current application creates a package of services oriented to the user. After that, it gives a presentation with some information about each service. This information is composed by images, audio data, texts, videos, multimedia data in general.

As this software agent is intended to increase its usability, hence it will increase the offered services and also all the multimedia files in the data base, a possible improvement to be done in this application would be to add a webservice. Those companies that would like to offer their services using our agent, could connect to the data base and upload all the information needed using a webservice.

Another possibility could be to create a webservice that looks for standarized information for each service it's going to offer. Using this reusability, we wouldn't need a large Data Base (an alike methodology as SCORM does for educational information). This technology is also better than having symbolic references to foreign web pages, it also don't force you to have a large Data Base system, but not internal pages can be changed, hence, those links could be broken. It's supposed that connected web servers will give us the correct multimedia, so we would not have this problem with it.

2. Use the groups to allow users to send messages to group partners

Having defined the user groups, consisting on users that have very similar preferences, it can be done another service consisting on an advising system.

The current agent has been developed to offer just a package of services to a user according to the user query (hours, number of people, etc). Sometimes, people join to activities that some friends are going to do.

Scenario 1: Paul has met his friend John. His friend tells him that all their colleagues are going to a famous fairground and asks him to join them. Of course, Paul accepts the invitation.

Scenario 2: A new Swedish flight company is selling tickets for seeing Växjö from a helicopter point of view. As this company is just starting, it sells them ten times cheaper that it usually could be. One constrain is that it's needed at least 10 people per flight.
This two scenarios are not taken into account in the current system. I could be added another service to advice people in the same group about group activities even they haven't asked for this activity, so users could post messages to invite people in the same group and meet people with similar preferences. Moreover, the last activity wouldn't appear in our system, because it wouldn't meet the number of people constrain.

3. **Deeper study in devices UAProf**
   We know UAProf informs us about the characteristics of a device, but this information is not always what we expect from it. For example, we can find in some cellular devices UAProf they are able to read SMIL files, although they don't have the needed software to play them. So, another improvement to be done is to have a better study of this information files to be assure what they contain.

4. **Multimedia instances for a same multimedia file**
   We have seen that depending on the user device, the stylesheet used will be different from another stylesheet used for another kind of device.
   Also, each stylesheet ask for different kinds of multimedia files. An stylesheet for a medium level phone will create a presentation with images and text, while a computer stylesheet could present the same information using an SMIL file with movies embedded.
   Even if we have the webservice to connect to the service provider, it's not supposed to have an instance of each multimedia item adapted to each stylesheet.
   A multimedia adapter application could be used in the system to convert movies to pictures, and colour pictures to B&W pictures. Thus, we only would need the most sophisticated multimedia file.

5. **Use the environment information in the service selection**
   In our example, the packages has been done, in a profile point of view, just looking the user preferences. That's only a prototype, so we haven't taken other information into account why selecting the services to present to the user. Some other aspects has to be studied before giving a service, like the weather could be.
   Environment has a relevant importance to select the activity the user can do. Of course a user can prefer playing golf instead going to the cinema, but if we know that it's not been a shiny day, it would be better to take another decision, because the user will not enjoy playing in those conditions.
   We can think about the distance to the starting point also as an environment aspect. So, if we have two activities equally important to the user, the one nearer to the user could be the selected one.
   Lots of other environmental sides have to be seen before taking a decision. The importance of each one is a designer question to think.

6. **Connect our service agent with the service provider to book the services selected by a user.**
If this software is going to be used, the first addition that has to be done is to connect it with the service providers. It's needed a webservice to connect our agent with all of the companies offering a service, so the booking could be done.

By now, we have an application that shows the user some possible activities combinations, just this. The user is given some ideas to know what he can do, but it would be much more useful if it were possible to book those activities once the user has selected the one he likes.

By now, this selection is only used to improve the reasoning machine decision, but, of course, it's more user oriented to believe that this decision is going to be used to have a reserve.

7. **Improve the layout**

The layout has been done to be similar for all kind of devices. The goal of this project wasn't to have a user friendly layout, but combining different kind of profiles with reasoning machines and XML.

So, the layout can be improved to be more pleasant to the user. It can be much improved for a computer device, but that's a designer scope.

8.2 **Further development**

1. **Extend this platform to be adapted to other services**

The idea of combining user profiles, device profiles and environment profiles can be used in lots of scopes.

The modules which would have to be adapted would be the reasoning machine, the final presentation (it doesn't have to be a multimedia presentation, it could also be a reaction in the physical world), the use methodology with the profiles information.

Scenario 1: Entertainment S.A. is going to use our platform to create an intelligent room for business people. This kind of people owe a PDA given by their companies. In their PDA they have their profiles which are sent by bluetooth to a receiver inside the room. The room also has some sensors to take some environmental information.

This room should take the user profiles from all the people inside, look for an appropriate service to offer them, between displaying a movie, playing some quiet music, or converting the room in a discotheque.

Here, the environmental profile will have more importance than in our application. Firstly the reasoning machine will study which of those services will the people enjoy most, and after that, it will use the sensors inside to make some changes to the result.

For example, it could see that all the people is speaking one to each other, and in their profiles it's seen that they like music. As they are just talking, it will start the session with quiet music, but if it perceives that it starts to be more lively, it will change the result to the discotheque one.
In our new scenario, it's seen we have to change the reasoning machine, the decision to take doesn't come from the same input information, also the solution is not the same as the one we use. Here, the answer is a modification of the environment, while ours is a multimedia presentation.

Here, the device profile can be used also, but just to know what can each item in the room do (displaying movies, changing colours for each disco light, etc.)

2. **Use of future computer UAProf (hardware and software)**

It's on development a UAProf for personal computers. This UAProf will have some default information about the hardware used in the computer, and also some default information about the software installed. Moreover, all the differences between the default system and the real one will be added to the computer UAProf.

When this new item is stable, it could be used to know also if a computer is able to display different presentations, so we could use the best stylesheet for it.

For example, we will know if the computer has Real One to be able to play SMIL presentation.

3. **Have a better knowledge of the environment to be able to create more suitable presentations**

Our environmental scope is very small by now. As we were doing only an small demonstration, we didn't get lots of different environment states.

Having a deeper study about the environment will help us to create more adapted presentations and a more useful application.

4. **Automatic environmental gathering using sensors**

In the future, it will be possible to access to environmental data using sensors around. We suppose our world will have some public sensors to gather information about the environment, and there will be an standarized protocol to send and receive information from them. So one device will be able to ask for those data and use them in the way it wants.

In our implementation, we ask the user for those information, but in future, that will be done automatically.
9 References


E-Links

Adaptive Hypermedia


Java


XML and WML


[12] http://www.infoescena.es/achuter/web/w3cdocs/xforms-for-html-authors_es.html#FormsControls (XForm)


Cocoon Documentation


**CC/PP and UAProf**


[43] http://nds.nokia.com/uaprof/N6100r100.xml (Nokia UAProf sample)


**SMIL**


[45] http://www.w3.org/AudioVideo/

**Schematron**


**User Agent**


**Reasoning Machine**


**Mobile**

[52] http://ppo.viha.ca/strategy/informatics/computer_support.htm (Computer Supported Cooperative Work)
[53] http://www.research.microsoft.com/easyliving/ and 
http://www.research.microsoft.com/vision/ (Ubiquitous computing project)

**HCI**

[54] http://iroi.seu.edu.cn/books/ee_dic/whatis/hci.htm

**Emulators**


**Definitions**

10 Index of images

Figure 1: Adaptation process passing by user data gathering and user model creation. 12

Figure 2: Graphical representation of an Adaptive Recommendation. Watching user browsing, the agent tries to deduce his/her goals, looks for possible user interesting links and shows......................................................... 16

Figure 3: Kinds of adaptive areas in a continuous. 
Closed areas have similarities............................................................ 17

Figure 4: Adaptation Technologies Summary........................................ 21

Figure 5: The CBR Cycle .................................................................... 31

Figure 6: Sample of a algorithm followed by our borrowed Case Base Reasoning machine. Looks for existing cases and looks for similarities............... 31

Figure 7: Simplified Architecture View .................................................. 34

Figure 8: Sequence Diagram - Plan a Day ............................................. 40

Figure 9: Sequence Diagram – Creating a Presentation .......................... 40

Figure 10: Sequence Diagram – User Registration ................................. 41

Figure 11: Sequence Diagram- Other Kind of Registrations ...................... 41

Figure 12: Architecture Diagram in detail ............................................. 46

Figure 13: Algorithm followed by the reasoning machine for new users ........... 48

Figure 14: DB Description (User Profile) ............................................... 51

Figure 15: DB Description (Group profile) ........................................... 52

Figure 16: DB Description (Device) ..................................................... 52

Figure 17: DB Description (Environment) ............................................. 53

Figure 18: DB Description (Service) .................................................... 53

Figure 19: DB Description (Multimedia Object) .................................... 53

Figure 20: DB Description (Base Case Content) .................................... 54

Figure 21: DB Description (Packs) ...................................................... 54

Figure 22: Application Screen Show (Main Window, options in the service) .... 56
Figure 23: Application Screen Shot (Left: Fields for user login. Right: Service options needed to choose services in the package ) ..................................................57

Figure 24: Application Screen Shot (List of packages suitable to the user according to his preferences and the selected service options) ...........................................58

Figure 25: SMIL Presentation Screen Shot. ........................................................................58

Figure 26: Application Screen Shot (User Registration. Formulary to be filled by the user and have a better understanding about his/her preferences) .................................................................60

Figure 27: Application Screen Shot (Device Registration. Asking for some device information to create a more suitable presentation. This section would be erased in a future) .................................................................61

Figure 28: Application Screen Shot (Environment Registration. Information required to create a more suitable presentation. This section will be erased in a future, when sensors are accesible from any device ) ..................................................64

Figure 29: Application Screen Shot (Package sample) ..........................................................72

Figure 30: First SMIL Screen Shot (2 Regions consisting on text and an image + the informational region with hours and place name. SMIL file created dynamicaly) ........................................................................73

Figure 31: Second SMIL Screen Shot (2 Regions consisting on text and an image + the informational region with hours and place name. SMIL file created dynamicaly) ........................................................................73

Figure 32: Using a Laptop in our application .......................................................................74

Figure 33: Browsing Go Out on a PDA. Using a meeting environment .................................75

Figure 34: Nokia emulator in a train environment .................................................................76

Figure 35: Mobile phone in an office environment ...............................................................77
## 11 Glossary of terms

### A

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptable hypermedia</td>
<td>7</td>
</tr>
<tr>
<td>Adaptive hypermedia</td>
<td>2, 5, 6, 7, 10, 11, 12, 13, 16, 21, 35, 78, 80, 91</td>
</tr>
<tr>
<td>ADSL</td>
<td>62, 63, 70</td>
</tr>
<tr>
<td>&quot;Asymmetric Digital Subscriber Line&quot; Internet Connection</td>
<td></td>
</tr>
<tr>
<td>Apache Cocoon</td>
<td>8, 33, 45, 78</td>
</tr>
<tr>
<td>Artificial Intelligence</td>
<td>2, 7, 96</td>
</tr>
</tbody>
</table>

### C

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case Based Reasoning</td>
<td>30, 32, 91</td>
</tr>
<tr>
<td>CC/PP</td>
<td>2, 8, 25, 26, 80, 81, 95</td>
</tr>
</tbody>
</table>

### G

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPRS</td>
<td>26</td>
</tr>
<tr>
<td>GSM</td>
<td>26</td>
</tr>
</tbody>
</table>
Human-Computer Interaction

The study of how people interact with computers and to what extent computers are or are not developed for successful interaction with human beings

Hypermedia

Associative relationships among information contained within multiple media data for the purpose of facilitating access to, and manipulation of, the information encapsulated by the data

Java

Java technology is a portfolio of products that are based on the power of networks and the idea that the same software should run on many different kinds of systems and devices. For more information: http://java.sun.com/

Java Virtual Machine

The Java Virtual Machine is only one aspect of Java software that is involved in web interaction. The Java Virtual Machine is built right into your Java software download, and helps the Sun JRE run Java applications.

JXForm

Cocoon approximation to the new standard XForms

MySQL

Open Source database Server http://www.mysql.com/

OMA

OMA is the leading industry forum for developing market driven, interoperable mobile service enablers

PDA

Personal Digital Assistant. A small digital device that is used to store information such as phone numbers, addresses, schedules, calendars, etc.

RDF


General framework for describing a Web site's metadata, or the information about the information on the site. It provides interoperability between applications that exchange machine-understandable information on the Web. RDF details information such as a site's sitemap, the dates of when updates were made, keywords that search engines look for and the Web page's intellectual property rights.

RealText
RealSystem product for streaming text from files or live sources

S

Schematron
A language for making assertions about patterns found in XML documents www.schematron.com

SCORM
SCORM is a suite of technical standards that enable web-based learning systems to find, import, share, reuse, and export learning content in a standardized way.

SMIL
The Synchronized Multimedia Integration Language (SMIL, pronounced "smile") enables simple authoring of interactive audiovisual presentations

SVG
Scalable Vector Graphics. W3C Recomendation

U

UAPProf
WAP Forum specification that is designed to allow wireless mobile devices to declare their capabilities to data servers and other network components

Ubiquitous computing
Computers everywhere. Making many computers available throughout the physical environment, while making them effectively invisible to the user. Ubiquitous computing is held by some to be the Third Wave of computing. The First Wave was many people per computer, the Second Wave was one person per computer. The Third Wave will be many computers per person. Three key technical issues are: power consumption, user interface, and wireless connectivity

URI
Universal Resource Identifier - (URI, originally "UDI" in some WWW documents) The generic set of all names and addresses which are short strings which refer to objects (typically on the Internet). The most common kinds of URI are URLs and relative URLs

URL
Uniform Resource Locator - (URL, previously "Universal") A standard way of specifying the location of an object, typically a web page, on the Internet. Other types of object are described below. URLs are the form of address used on the World-Wide Web. They are used in
HTML documents to specify the target of a hyperlink which is often another HTML document (possibly stored on another computer).

User model 7, 11, 12, 14, 19
Structure containing information about the user, exploited by the system in order to increase the quality of the interaction.

User Modelling 7, 12
Part of an interactive system, whose function is to create an user model.

W

WAP 25, 26, 95
Wireless Application Protocol - (WAP) An open international standard for applications that use wireless communication, e.g. Internet access from a mobile phone.

WTLS 26
Wireless Transport Layer Security - (WTLS) The WAP standard related to security. WTLS is based upon its TCP/IP counterpart, Secure Sockets Layer.

WML 26, 33, 92
Wireless Markup Language is the primary content format for devices that implement the WAP Wireless Application Protocol (WAP) is an open international standard for applications that use wireless communication

X

XForms 41, 65, 80
The Next Generation of Web Forms.
"XForms" is W3C's name for a specification of Web forms that can be used with a wide variety of platforms including desktop computers, hand helds, information appliances, and even paper. XForms started life as a subgroup of the HTML Working Group, but has now been spun off as an independent Activity

xHTML 26
Extensible Hypertext Markup Language (xHTML), or simply XML, is a markup language that has the same expressive possibilities as HTML, but conforms to the XML standard which is more strict.

XML 2, 8, 25, 26, 30, 33, 39, 65, 66, 68, 69, 80, 81, 85, 92, 95
eXtensible Markup Language is a W3C recommendation for creating special-purpose markup languages. It is a simplified subset of SGML of describing many different kinds of data. Its primary purpose is to facilitate the sharing of structured text and information across the Internet. Languages based on XML (for example, RDF, SMIL, MathML, XSIL and SVG) are themselves described in a formal way, allowing programs to modify and validate documents in these languages without prior knowledge of their form.

XSL 39, 65, 66, 69, 80, 95
Extensible Stylesheet Language (XSL) is a language which allows one to describe how files encoded in the XML standard are to be
The language has two parts: XSLT and XSL-FO (Formatting Objects) to describe additional formatting.

**XSLT**

XSLT is a XML Transformation Language which transforms documents in XML format.

**XSP**

eXtensible Server Pages. XSP is used to build dynamic XML content in Apache Cocoon.
12 Appendix

12.a Source code to plan a new day

```java
/**
 * This class creates a set of packs, every pack with one day route
 */
public class Pack {
    static String bd = "server"
    static String login = "root"
    static String password = "root"
    static String url = "jdbc:mysql://localhost/"+bd;
    static int numberOptions = 1;
    static int numberOptionsTotals = 10;
    Connection conn = null;
    private TreeSet allPacks;
    int age;
    int budget;
    int people;
    float start_hour;
    float finish_hour;
    float rest_hours;
    private TraitDescriptors traitDescriptors;
    public Pack(TraitDescriptors traitDescriptors, int id)
    {
        this.traitDescriptors = traitDescriptors;
        allPacks = new TreeSet(new CompareDistance());
        NewPack(id);
    }
    public TreeSet getAllPacks()
    {
        return allPacks;
    }
    public void NewPack(int id)
    {
        try {
            LinkedList cand = new LinkedList();
            cand.add(new LinkedList());
            cand.add(new LinkedList());
            cand.add(new LinkedList());
            cand.add(new LinkedList());
            cand.add(new LinkedList());
            LinkedList candGastronomy = new LinkedList();
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            conn = DriverManager.getConnection(url, login, password);
            if (conn != null)
            {
                System.out.println("Conexión a base de datos "+url+" ... Ok");
                Statement stmt = conn.createStatement();
                Statement stmt2 = conn.createStatement();
                ResultSet query = stmt.executeQuery("select * from query_content where identifier='"+id+'"");
                ResultSetMetaData metadata = query.getMetaData();
                int ncolumn = metadata.getColumnCount();
                query.next();
                age = query.getInt("age");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

budget = query.getInt("budget");
people = query.getInt("people");
start_hour = query.getFloat("start_hour");
finish_hour = query.getFloat("finish_hour");
rest_hours = finish_hour - start_hour;
for(int i=7; i <= ncolumn; i++)
{
    String column = metadata.getColumnName(i);
    int value = query.getInt(i);
    ResultSet service = stmt2.executeQuery("select * from service where sector=" + column + "");
    if(!column.equals("gastronomy"))
    {
        while(service.next())
        {
            Service servi = new Service();
            System.out.println("Name: " + service.getString("name"));
            servi.setIdentifier(service.getInt("identifier"));
            servi.setName(service.getString("name"));
            servi.setSector(service.getString("sector"));
            servi.setSubclass(service.getString("subclass"));
            servi.setAge_min(service.getInt("age_min"));
            servi.setAge_max(service.getInt("age_max"));
            servi.setPrice(service.getInt("price"));
            servi.setStart_hour(service.getFloat("start_hour"));
            servi.setFinish_hour(service.getFloat("finish_hour"));
            servi.setPeople_min(service.getInt("people_min"));
            servi.setPeople_max(service.getInt("people_max"));
            servi.setDist_x(service.getFloat("dist_x"));
            servi.setDist_y(service.getFloat("dist_y"));
            servi.setTime_spent(service.getFloat("time_spent"));
            LinkedList lis = (LinkedList) cand.get(value);
            lis.add(servi);
            cand.set(value,lis);
        }
        System.out.println("Prefer * + value= " + value+ " ----------------------");
        service.close();
    }
    else
    {
        while(service.next())
        {
            Service servi = new Service();
            System.out.println("Gastronomy: " + service.getString("name"));
            servi.setIdentifier(service.getInt("identifier"));
            servi.setName(service.getString("name"));
            servi.setSector(service.getString("sector"));
            servi.setSubclass(service.getString("subclass"));
            servi.setAge_min(service.getInt("age_min"));
            servi.setAge_max(service.getInt("age_max"));
            servi.setPrice(service.getInt("price"));
            servi.setStart_hour(service.getFloat("start_hour"));
            servi.setFinish_hour(service.getFloat("finish_hour"));
            servi.setPeople_min(service.getInt("people_min"));
            servi.setPeople_max(service.getInt("people_max"));
            servi.setDist_x(service.getFloat("dist_x"));
            servi.setDist_y(service.getFloat("dist_y"));
            servi.setTime_spent(service.getFloat("time_spent"));
            candGastronomy.add(servi);
        }
        service.close();
    }
}
boolean breakfast = false;
boolean lunch = false;
boolean dinner = false;
if(start_hour>10.5)
{
    breakfast=true;
}
if(start_hour>15.5)
{
    lunch=true;
}
if(start_hour>22.5)
{
    dinner=true;
}
float actual_hour = start_hour;
boolean find;
OnePacket pack = new OnePacket();
for(int i=0; i < numberOptionsTotals;i++)
{
    newService(actual_hour,5, cand, candGastronomy, pack, breakfast, lunch, dinner);
}
Iterator iter = allPacks.iterator();
System.out.println("OK");
while(iter.hasNext())
{
    OnePacket pack2 = (OnePacket) iter.next();
    System.out.println("-------------------
    NEW PACK
    -------------------------");
    for(int i=0; i<pack2.size(); i++)
    {
        System.out.println("Option: " + ((Service) pack2.getService(i)).getName());
        System.out.println("Budget: " + pack2.getPrice());
        System.out.println("Dist: " + pack2.getDistance());
        System.out.println("Points: " + pack2.getPoints());
    }
    System.out.println("Number of different packs: " + allPacks.size());
    query.close();
    stmt.close();
    conn.close();
}
catch(SQLException ex)
{
    System.out.println(ex);
} catch(ClassNotFoundException ex)
{
    System.out.println(ex);
} catch (Exception e) {
    String methodName = "Pack";
    System.out.println(methodName + " error: " + e );
} //--- catch

public void newService(float actual_hour, int pref, LinkedList cand, LinkedList candGastronomy, OnePacket pack, boolean breakfast, boolean lunch, boolean dinner)
{
try {
    if (actual_hour <= finish_hour) {
        Statement stmt = conn.createStatement();
        ResultSet hour = stmt.executeQuery("select * from service where start_hour<='" + actual_hour + "; and (" + actual_hour + "+ time_spent) <= finish_hour and (" + actual_hour + "+ time_spent) <=" + finish_hour + " and age_min <= " + age + " and age_max >= " + age + " and people_min <= " + people + " and people_max >= " + people + " or people_max is NULL")

        LinkedList identTime = new LinkedList();
        // candidatos por tiempo
        while (hour.next()) {
            Service servi = new Service();
            int ide = hour.getInt("identifier");
            servi.setIdentifier(hour.getInt("identifier");
            servi.setName(hour.getString("name");
            servi.setSector(hour.getString("sector");
            servi.setSubclass(hour.getString("subclass");
            servi.setAge_min(hour.getInt("age_min");
            servi.setAge_max(hour.getInt("age_max");
            servi.setPrice(hour.getInt("price");
            servi.setStart_hour(hour.getFloat("start_hour");
            servi.setFinish_hour(hour.getFloat("finish_hour");
            servi.setPeople_min(hour.getInt("people_min");
            servi.setPeople_max(hour.getInt("people_max");
            servi.setDist_x(hour.getFloat("dist_x");
            servi.setDist_y(hour.getFloat("dist_y");
            servi.setTime_spent(hour.getFloat("time_spent");

            identTime.add(new Integer(hour.getInt("identifier");)
        }
        hour.close();
        stmt.close();
    } else {
        boolean find = false;
        int counter = 0;
        Random random = new Random();
        if(8 <= actual_hour && (actual_hour <= 10.5) && (!breakfast)) {
            int size = candGastronomy.size();
            LinkedList candGastronomy2 = (LinkedList) candGastronomy.clone();
            int index = size;
            int ran;
            for (int i = 0; i < size; i++) {
                if (index != 0) {
                    ran = random.nextInt(index);
                } else {
                    ran = 0;
                }
                Service option = (Service) candGastronomy2.get(ran);  
                candGastronomy2.remove(option); 
                index--; 
                if (identTime.contains(new Integer(option.getIdentifier()))) && (option.getSubclass().equals("coffee")))
                    
                
        }
    }
}
LinkedList candGastronomyx = new LinkedList();
candGastronomyx = (LinkedList) candGastronomy.clone();
candGastronomyx.remove(option);
OnePacket packx = pack.clone();
Service optionChoice = clon(option);
optionChoice.setStart_hour(actual_hour);
packx.addService(optionChoice,pref);
breakfast = true;
newService((actual_hour + option.getTime_spent() + (float) 0.5), 5, cand,
candGastronomyx, packx, breakfast, lunch, dinner);

find = true;
counter++;
if(counter==numberOptions)
{
    break;
}
}
}

else if((13<=actual_hour)&&(actual_hour<=15.5)&&(lunch))
{
    int size = candGastronomy.size();
    LinkedList candGastronomy2 = (LinkedList) candGastronomy.clone();
    int index = size;
    int ran;
    for(int i=0; i < size; i++)
    {
        if(index!=0)
        {
            ran = random.nextInt(index);
        } else
        {
            ran = 0;
        }
        Service option = (Service) candGastronomy2.get(ran);
candGastronomy2.remove(option);
index--;
if((identTime.contains(new Integer(option.getIdentifier())))&&(option.getSubclass().equals("coffee")))
{
    LinkedList candGastronomyx = new LinkedList();
candGastronomyx = (LinkedList) candGastronomy.clone();
candGastronomyx.remove(option);
OnePacket packx = pack.clone();
Service optionChoice = clon(option);
optionChoice.setStart_hour(actual_hour);
packx.addService(optionChoice,pref);
lunch = true;
newService((actual_hour + option.getTime_spent() + (float) 0.5), 5, cand,
candGastronomyx, packx, breakfast, lunch, dinner);
find = true;
counter++;
if(counter==numberOptions)
{
    break;
}
}
else if((20<=actual_hour)&&(actual_hour<=22.5)&&(dinner)) {
    int size = candGastronomy.size();
    LinkedList candGastronomy2 = (LinkedList) candGastronomy.clone();
    int index = size;
    int ran;
    for(int i=0; i < size; i++)
    {
        if(index!=0)
        {
            ran = random.nextInt(index);
        }
        else
        {
            ran = 0;
        }
        Service option = (Service) candGastronomy2.get(ran);
        candGastronomy2.remove(option);
        index--;
        if((identTime.contains(new Integer(option.getIdentifier())))&&(!option.getSubclass().equals("coffee")))
        {
            LinkedList candGastronomyx = new LinkedList();
            candGastronomyx = (LinkedList) candGastronomy.clone();
            candGastronomyx.remove(option);
            OnePacket packx = pack.clone();
            Service optionChoice = clon(option);
            optionChoice.setStart_hour(actual_hour);
            packx.addService(optionChoice,pref);
            dinner = true;
            newService((actual_hour + option.getTime_spent() + (float) 0.5), 5, cand,
            candGastronomyx, packx, breakfast, lunch, dinner);
            find = true;
            counter++;    
            if(counter==numberOptions)
            {
                break;
            }
        }
    }
}
else {
    LinkedList candi =(LinkedList) cand.get(pref-1);
    int size = candi.size();
    LinkedList candi2 = (LinkedList) candi.clone();
    int index = size;
    int ran;
    for(int i=0; i < size; i++)
    {
        if(index!=0)
        {
            ran = random.nextInt(index);
        }
        else
        {
            ran = 0;
        }
        Service option = (Service) candi2.get(ran);
candi2.remove(option);
index--;
if(!identTime.contains(new Integer(option.getIdentifier())))
{
    LinkedList candx = new LinkedList();
candx = (LinkedList) candi.clone();
candx.remove(option);
int sizeb = candx.size();
for(int m = 0; m < sizeb; m++)
{
    Service serv2 = (Service) candx.get(m);
    if(option.getSector().equals(serv2.getSector()))
    {
        candx.remove(serv2);
        sizeb--;
        m--;
    }
}
LinkedList candix = new LinkedList();
candix = (LinkedList) cand.clone();
candix.set(pref-1, candx);
OnePacket packx = pack.clone();
Service optionChoice = clon(option);
optionChoice.setStart_hour(actual_hour);
packx.addService(optionChoice, pref);
newService((actual_hour + option.getTime_spent()) + (float)0.5), 5, candix, (LinkedList) candGastronomy.clone(), packx, breakfast, lunch, dinner);
find = true;
counter++;
if(counter==numberOfOptions)
{
    break;
}
}
}
}
if(!find)
{
    if(pref!=1)
    {
        newService(actual_hour, pref - 1, cand, candGastronomy, pack, breakfast, lunch, dinner);
    }
    else
    {
        newService(actual_hour + ((float)0.5), 5, cand, candGastronomy, pack, breakfast, lunch, dinner);
    }
}
else
{
    allPacks.add(pack);
}
}
}
catch(SQLException ex)
{
    System.out.println(ex);
}
}
catch (Exception e) {

String methodName = "Pack";
System.out.println(methodName + " error: " + e);

public Service clone(Service original) {
    Service copy = new Service();
    copy.setIdentifier(original.getIdentifier());
    copy.setName(original.getName());
    copy.setSector(original.getSector());
    copy.setSubclass(original.getSubclass());
    copy.setAge_min(original.getAge_min());
    copy.setAge_max(original.getAge_max());
    copy.setPrice(original.getPrice());
    copy.setStart_hour(original.getStart_hour());
    copy.setFinish_hour(original.getFinish_hour());
    copy.setPeople_min(original.getPeople_min());
    copy.setPeople_max(original.getPeople_max());
    copy.setDist_x(original.getDist_x());
    copy.setDist_y(original.getDist_y());
    copy.setTime_spent(original.getTime_spent());
    return copy;
}

12.b Source code to create a dynamic presentation

<?xml version="1.0" ?>

<xsp:page language="java"
xmns:xsp="http://apache.org/xsp"
xmns:xsp-require="http://apache.org/xsp/request/2.0"
xmns:xsp-hello="http://apache.org/xsp/hello/1.0"
xmns:jpath="http://apache.org/xsp/jpath/1.0"
xmns:xsp-session="http://apache.org/xsp/session/2.0"
xmns:esql="http://apache.org/cocoon/SQL/v2"
create-session="true">
  <xsp:structure>
    <xsp:include>net.goout.selectionengine.*</xsp:include>
    <xsp:include>java.util.*</xsp:include>
    <xsp:include>java.lang.*</xsp:include>
  </xsp:structure>
  <xsp:logicsheet location="logicsheets/hello.xsl"/>

  <document>
    <xsp:logic>
      int screen_size_x = 0;
      int screen_size_y = 0;
      <esql:connection>
        <esql:pool>databaseserver</esql:pool>
    </xsp:logic>
  </document>
</xsp:page>
<esql:execute-query>
<esql:query>select * from query_presentation_order where presentation_identifier=<esql:parameter><xsp:request:get-parameter name="id"/></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
int iden= <esql:get-int column="multimedia_object_identifier"/>;  
types.add(new Integer(<esql:get-int column="type"/>));
if(<esql:get-int column="type"/> == 0)  
{  
<esql:execute-query>  
<esql:query>select * from text_object where identifier=<esql:parameter><xsp:expr>identif</xsp:expr></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
if(<esql:get-int column="service_identifer"/)!="service")  
{
  if(service!=0)  
  {  
groups.add(new Integer(groups));  
grups = 1;  
service_identifier.add(new Integer(service));  
service = <esql:get-int column="service_identifer"/>>;  
  }  
else  
  {
    
  
}  
</xsp:logic>
</esql:row-results>
</esql:results>
</esql:execute-query>

<esql:execute-query>
<esql:query>select * from query_presentation_order where presentation_identifier=<esql:parameter><xsp:request:get-parameter name="id"/></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
im in = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int groups=0;  
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:row-results>
</esql:results>
</esql:execute-query>

<body>
<xsp:logic>
im = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:execute-query>

<esql:execute-query>
<esql:query>select * from query_presentation_order where presentation_identifier=<esql:parameter><xsp:request:get-parameter name="id"/></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
im = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:row-results>
</esql:results>
</esql:execute-query>

<esql:execute-query>
<esql:query>select * from query_presentation_order where presentation_identifier=<esql:parameter><xsp:request:get-parameter name="id"/></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
im = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:row-results>
</esql:results>
</esql:execute-query>

<esql:execute-query>
<esql:query>select * from query_presentation_order where presentation_identifier=<esql:parameter><xsp:request:get-parameter name="id"/></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
im = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:row-results>
</esql:results>
</esql:execute-query>

<body>
<xsp:logic>
im = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:execute-query>

<esql:execute-query>
<esql:query>select * from query_presentation_order where presentation_identifier=<esql:parameter><xsp:request:get-parameter name="id"/></esql:parameter></esql:query>
<esql:results>
<esql:row-results>
<xsp:logic>
im = 0;  
Vector urls = new Vector();  
Vector types = new Vector();  
Vector groups = new Vector();  
Vector service_identifier = new Vector();
int service = 0;  
String region;  
int serv_id=0;  
int info_size_y = (int) ((float) screen_size_y * 0.2);
if(sound==1)  
{  
  <sound>100%</sound>  
}  
else  
{  
  <sound>0%</sound>  
}  
</xsp:logic>
</esql:row-results>
</esql:results>
</esql:execute-query>
service = <esql:get int column="service_identifier"/>
    grups++;

    }
    }
else
    {
    grups++;
    }
if(grups==4)
    {
    groups.add(new Integer(grups));
    grups = 0;
    service_identifier.add(new Integer(service));
    service = <esql:get int column="service_identifier"/>
    }
</xsp:logic>

</esql:row-results>
</esql:results>
</esql:execute-query>

urls.add(<esql:get string column="multimedia_object_identifier"/>);
in++;
}
else
{
<esql:execute-query>
    <esql:query>select * from multimedia_object where identifier=<esql:parameter><xsp:expr>identif</xsp:expr></esql:parameter></esql:query>
</esql:execute-query>
    <esql:results>
    <esql:row-results>
        <xsp:logic>
            urls.add(<esql:get string column="url"/>);
            in++;
            if(<esql:get int column="service_identifier"/!="service")
            {
            if(service!=0)
            {
            groups.add(new Integer(grups));
            grups = 1;
            service_identifier.add(new Integer(service));
            service = <esql:get int column="service_identifier"/>
            }
            else
            {
            service = <esql:get int column="service_identifier"/>
            grups++;
            }
            }
            else
            {
            groups++;
            }
            if(grups==4)
            {
            groups.add(new Integer(grups));
            grups = 0;
            service_identifier.add(new Integer(service));
            service = <esql:get int column="service_identifier"/>
            }
        </xsp:logic>
    </esql:row-results>
</esql:execute-query>
</esql:row-results>
</esql:execute-query>

urls.add(<esql:get string column="url"/});
in++;
if(<esql:get int column="service_identifier"/!="service")
{
if(service!=0)
{
groups.add(new Integer(grups));
grups = 1;
service_identifier.add(new Integer(service));
service = <esql:get int column="service_identifier"/>
}
else
{
service = <esql:get int column="service_identifier"/>
grups++;
}
}
else
{
grups++;
}
if(grups==4)
{
groups.add(new Integer(grups));
grups = 0;
service_identifier.add(new Integer(service));
service = <esql:get int column="service_identifier"/>
}
}
if(grups!=0) {
    groups.add(new Integer(grups));
    service_identifier.add(new Integer(service));
}

int size = 0;
for(int k=0; k < groups.size(); k++) {
    Integer temp = (Integer) service_identifier.get(k);
    serv_id = temp.intValue();

    Integer nmembersI = (Integer) groups.get(k);
    int nmembers = nmembersI.intValue();
    if(nmembers==1) {
        Integer val = (Integer) <xsp:expr>types.get(size)</xsp:expr>;
        int value = val.intValue();
        region = new String("one");
        if(value == 0) {
            <text>
                <xsp:logic>
                    <region><xsp:expr>region</xsp:expr></region>
                    <url><xsp:expr>urls.get(size)</xsp:expr></url>
                    int size_y = (int) ((float) screen_size_y * 0.8);
                    <screen_size_x><xsp:expr>screen_size_x</xsp:expr></screen_size_x>
                    <screen_size_y><xsp:expr>size_y</xsp:expr></screen_size_y>
                </xsp:logic>
            </text>
        }
        else if(value == 1) {
            <image>
                <xsp:logic>
                    <region><xsp:expr>region</xsp:expr></region>
                    <url><xsp:expr>urls.get(size)</xsp:expr></url>
                </xsp:logic>
            </image>
        }
        else if(value == 2) {
            <audio>
                <xsp:logic>
                    <region><xsp:expr>region</xsp:expr></region>
                </xsp:logic>
            </audio>
        }
    }
}

if(nmembers==1) {
    Integer val = (Integer) <xsp:expr>types.get(size)</xsp:expr>;
    int value = val.intValue();
    region = new String("one");
    if(value == 0) {
        <text>
            <xsp:logic>
                <region><xsp:expr>region</xsp:expr></region>
                <url><xsp:expr>urls.get(size)</xsp:expr></url>
                int size_y = (int) ((float) screen_size_y * 0.8);
                <screen_size_x><xsp:expr>screen_size_x</xsp:expr></screen_size_x>
                <screen_size_y><xsp:expr>size_y</xsp:expr></screen_size_y>
            </xsp:logic>
        </text>
    }
    else if(value == 1) {
        <image>
            <xsp:logic>
                <region><xsp:expr>region</xsp:expr></region>
                <url><xsp:expr>urls.get(size)</xsp:expr></url>
            </xsp:logic>
        </image>
    }
    else if(value == 2) {
        <audio>
            <xsp:logic>
                <region><xsp:expr>region</xsp:expr></region>
            </xsp:logic>
        </audio>
    }
}
87

```xml
<url><xsp:expr>urls.get(size)</xsp:expr></url>
</xsp:logic>
</audio>
}  
else if(value == 3)  
{  
<video>
<xsp:logic>
<region><xsp:expr>region</xsp:expr></region>
<url><xsp:expr>urls.get(size)</xsp:expr></url>
</xsp:logic>
</video>
}
size++;
}
else if(nmembers==2)  
{
int part=1;  
for (int j=0; j < nmembers; j++)  
{
Integer val = (Integer) <xsp:expr>types.get(size)</xsp:expr>;  
int value = val.intValue();  
if(part==1)  
{  
region = new String("one_two");  
}  
else  
{  
region = new String("two_two");  
}
part++;  
if(value == 0)  
{
<text>
<xsp:logic>
<region><xsp:expr>region</xsp:expr></region>
<url><xsp:expr>urls.get(size)</xsp:expr></url>
int size_x = (int) screen_size_x / 2;  
int size_y = (int) ((float) screen_size_y * 0.8);  
<screen_size_x><xsp:expr>size_x</xsp:expr></screen_size_x>
<screen_size_y><xsp:expr>screen_size_y</xsp:expr></screen_size_y>
</xsp:logic>
</text>
}  
else if(value == 1)  
{
<image>
<xsp:logic>
<region><xsp:expr>region</xsp:expr></region>
<url><xsp:expr>urls.get(size)</xsp:expr></url>
</xsp:logic>
</image>
}  
else if(value == 2)  
{
<audio>
<xsp:logic>
<region><xsp:expr>region</xsp:expr></region>
<url><xsp:expr>urls.get(size)</xsp:expr></url>
</xsp:logic>
</audio>
}  
else if(value == 3)  
{
<video>
<xsp:logic>
<region><xsp:expr>region</xsp:expr></region>
<url><xsp:expr>urls.get(size)</xsp:expr></url>
</xsp:logic>
</video>
}
```
size++;

else if(nmembers==3)
{
    int part=1;
    for (int j=0; j<nmembers; j++) {
        if(part==1)
        {
            region = new String("one_three");
        }
        else if(part==2)
        {
            region = new String("two_three");
        }
        else
        {
            region = new String("three_three");
        }
        part++;
        Integer val = (Integer) <xsp:expr>types.get(size)</xsp:expr>;
        int value = val.intValue();
        if( value == 0)
        {
            <text>
            <xsp:logic>
            <region><xsp:expr>region</xsp:expr></region>
            <url><xsp:expr>urls.get(size)</xsp:expr></url>
            int size_x = (int) screen_size_x / 3;
            int size_y = (int) ((float) screen_size_y * 0.8);
            <screen_size_x><xsp:expr>size_x</xsp:expr></screen_size_x>
            <screen_size_y><xsp:expr>screen_size_y</xsp:expr></screen_size_y>
            </xsp:logic>
            </text>
        }
        else if(value == 1)
        {
            <image>
            <xsp:logic>
            <region><xsp:expr>region</xsp:expr></region>
            <url><xsp:expr>urls.get(size)</xsp:expr></url>
            </xsp:logic>
            </image>
        }
        else if(value == 2)
        {
            <audio>
            <xsp:logic>
            <region><xsp:expr>region</xsp:expr></region>
            <url><xsp:expr>urls.get(size)</xsp:expr></url>
            </xsp:logic>
            </audio>
        }
        else if(value == 3)
        {
            <video>
            <xsp:logic>
            <region><xsp:expr>region</xsp:expr></region>
            <url><xsp:expr>urls.get(size)</xsp:expr></url>
            </xsp:logic>
            </video>
        }
    }
    size++;
}

else
int part=1;
for (int j=0; j < nmembers; j++) {

    if(part==1) {
        region = new String("one_four");
    } else if(part==2) {
        region = new String("two_four");
    } else if(part==3) {
        region = new String("three_four");
    } else {
        region = new String("four_four");
    }
    part++;
    Integer val = (Integer) <xsp:expr>types.get(size)</xsp:expr>;
    int value = val.intValue();
    if(value == 0) {
        <text>
        <xsp:logic>
        <region><xsp:expr>region</xsp:expr></region>
        <url><xsp:expr>urls.get(size)</xsp:expr></url>
        int size_x = (int) screen_size_x / 2;
        int size_y = (int) ((float) screen_size_y * 0.8) / 2;
        <screen_size_x><xsp:expr>size_x</xsp:expr></screen_size_x>
        <screen_size_y><xsp:expr>size_y</xsp:expr></screen_size_y>
        </xsp:logic>
        </text>
    } else if(value == 1) {
        <image>
        <xsp:logic>
        <region><xsp:expr>region</xsp:expr></region>
        <url><xsp:expr>urls.get(size)</xsp:expr></url>
        </xsp:logic>
        </image>
    } else if(value == 2) {
        <audio>
        <xsp:logic>
        <region><xsp:expr>region</xsp:expr></region>
        <url><xsp:expr>urls.get(size)</xsp:expr></url>
        </xsp:logic>
        </audio>
    } else if(value == 3) {
        <video>
        <xsp:logic>
        <region><xsp:expr>region</xsp:expr></region>
        <url><xsp:expr>urls.get(size)</xsp:expr></url>
        </xsp:logic>
        </video>
    }
    size++;
}
</xsp:logic>
</par>
</new>
12.c Selection Engine

Here we can see the Selection Engine class diagram. To know more about this reasoning machine, please visit the author’s web site:

http://ihatebaylor.com/technical/computer/ai/selection_engine/

Figure 31: Graphical representation of the Selection Engine