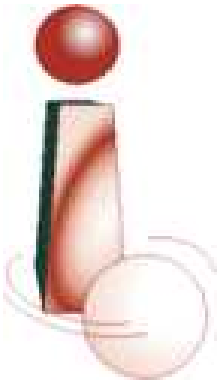
Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2009

Luís Alberto
Capote Ribeiro

Portal de Gestão de Competições Robóticas Simuladas

Simulated Robotic Competitions Management Portal



Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2009

Luís Alberto Capote Ribeiro

Portal de Gestão de Competições Robóticas Simuladas

Simulated Robotic Competitions Management Portal

Dissertação apresentada à Universidade de Aveiro, para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica de Prof. Doutor Artur Pereira e Prof. Doutor Nuno Lau, professores auxiliares do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri

presidente	**Doutor Tomás António Mendes Oliveira e Silva** professor associado da Universidade de Aveiro
1º vogal	**Doutor José Manuel Castro Torres** professor auxiliar da Universidade Fernando Pessoa
orientador	**Doutor Artur José Carneiro Pereira** professor auxiliar da Universidade de Aveiro
co-orientador	**Doutor José Nuno Panelas Nunes Lau** professor auxiliar da Universidade de Aveiro

acknowledgements / agradecimentos

À minha família, em especial aos meus pais, Carlos e Conceição, e à minha irmã, Ana, por todo o apoio, carinho e força ao longo da minha vida e por todas as oportunidades que me proporcionaram para chegar onde estou, sem eles tudo isto não passaria de um sonho.
À Nancy, por todo o carinho, amizade, força e paciência, obrigado pela pessoa que és.
Aos meus orientadores, Artur Pereira e Nuno Lau, por toda a disponilidade, apoio e conselhos ao longo do desenvolvimento deste projecto. Foram essenciais no decorrer da dissertação.
A todos os meus amigos pela amizade e companhia, especialmente ao Alex e Ricardo, pelos momentos relaxados e pela ajuda nos momentos mais complicados.

Abstract

Robotic competitions' popularity, both hardware and software based types, has been growing over the years due to the continuous evolution of the robotic and computer technologies.
The University of Aveiro, more specifically DETI, has played a relevant part in the propagation and development of these robotic competitions, since it introduced the Micro-Rato robotic competition to motivate the students to learn more about robotic environments.
These competitions' initial stage was at a physical level, that is, the robots had to be built, the developed agents had to be applied to the robots and the actual competition would have to occur in a physical unique location.
Later, these robotic competitions' responsible teachers decided to provide new competition platforms to reach a higher number of interested people, so a new modality called Ciber-Rato was launched restricting the competition to a simulated environment where the developed agents would run in an application created to simulate the physical environment and relative variable components.
Since then, Ciber-Rato has been more disseminated, expanding into high-schools and into worldwide robotic events, being nowadays associated with the international RTSS event. Unfortunately it still requires the gathering of the participants in a unique physical location in order to compete.
Evidently, it is time to advance to the Internet field, which would allow reaching more participants, internationally promote the competition and also provide a new stage of competitions, since remote participations on physical and online events would be a near reality.
This project aims the development of an autonomous simulated robotic competitions management portal, which would be extremely useful as an auxiliary tool for physical events management and also empower the realization of online events with full autonomous management (except in critical situations).
With this in mind, this thesis formalizes and describes the simulated robotic competitions management portal structure, development techniques and resulting interfaces, as well as discuss the current technologies and how they are used to connect the various components of the simulation.

Resumo

A popularidade das competições robóticas simuladas, em ambos os tipos baseados em hardware e software, tem vindo a crescer com o passar dos anos devido à contínua evolução das tecnologias da robótica e da informática.

A Universidade de Aveiro, mais especificamente o DETI, desempenhou um papel relevante na propagação e desenvolvimento destas competições robóticas, visto que introduziu a competição robótica Micro-Rato para motivar os estudantes a aprender mais sobre ambientes robóticos.

A etapa inicial destas competições era a um nível físico, isto é, os robôs tinham que ser construídos, os agentes tinham que ser introduzidos nos robôs e a competição teria que decorrer num local físico único.

Mais tarde, os professores responsáveis pelas competições robóticas decidiram possibilitar novas plataformas de competição para atingir um maior número de interessados, assim uma nova modalidade denominada Ciber-Rato foi lançada restringindo a competição a um ambiente simulado, onde os agentes seriam executados numa aplicação criada para simular o ambiente físico e as variáveis dos componentes relativos.

Desde então, o Ciber-Rato tem sido mais propagado, expandindo-se para escolas secundárias e eventos de robótica pelo mundo, sendo actualmente associado com o evento internacional RTSS. Infelizmente, a competição ainda requer a reunião dos participantes num único local físico para competir.

Evidentemente, está na altura de avançar para o campo da Internet, que possibilitaria alcançar mais participantes, promover a competição internacionalmente e também disponibilizar uma nova etapa de competição, uma vez que participações remotas em eventos físicos e online seriam uma possível realidade.

Este projecto visa o desenvolvimento de um portal autónomo de gestão de competições robóticas simuladas, que seria extremamente útil como uma ferramenta auxiliar na gestão de eventos físicos e também potenciaria a realização de eventos online com uma gestão autónoma completa (excepto em situações críticas).

Com isto em mente, esta dissertação formaliza e descreve a estrutura, técnicas de desenvolvimento e interfaces resultantes do portal de gestão de competições robóticas simuladas, assim como discute as tecnologias Web actuais e de que forma são usadas para interligar as diversas componentes da simulação.

Contents

List of Figures

List of Tables

List of Acronyms

ABM	Automated Banking Machine
AJAX	Asynchronous JavaScript and XML
ANSI	American National Standards Institute
AOLServer	America OnLine Server
API	Application Programming Interface
ASP	Active Server Pages
ATM	Automated Teller Machine
BBCodes	Bulletin Board Codes
BSD	Berkeley Software Distribution
CGI	Common Gateway Interface
CMS	Content Management System
CPSS	Cyber-Physical Systems Simulator
CSS	Cascading StyleSheets
DBMS	DataBase Management System
DETI	Department of Electronics, Telecommunications and Informatics
DET	Department of Electronics and Telecommunications
DMS	Document Management System
DOM	Document Object Model
EMWAC	European Microsoft Windows NT Academic Center
FAQ	Frequently Asked Questions

FTP	File Transfer Protocol
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IBM	International Business Machines corporation
IDE	Integrated Development Environment
IIS	Former Internet Information Server, current Internet Information Services
IPv6	Internet Protocol version 6
ISO	International Standardization Organization
J2EE	Java 2 Enterprise Edition
JSON	JavaScript Object Notation
JSP	Java Server Pages
JVM	Java Virtual Machine
LAMP	Linux, Apache, MySQL and PHP
LAN	Large Area Network
LED	Light-Emitting Diode
MMORPG	Massively Multiplayer Online Role-Playing Game
MUD	Multi-User Dungeons
NCSA HTTPd	National Center for SuperComputing Applications HTTP Daemon
NNTP	Network News Transfer Protocol
ODBC	Open DataBase Connectivity
ORDBMS	Object-Relational DataBase Management System
OS/2	Operating System/2 created by Microsoft and IBM
OS	Operating System
PHP	Former Personal Home Page, current recursive PHP:Hypertext Preprocessor

PL/pgSQL	PL/postgreSQL
PSM	Persistent Stored Modules
PSQL	Procedural SQL
RDBMS	Relational DataBase Management System
RSS	Really Simple Syndication or, less commonly, Rich Site Summary
RTSS	Real-Time Systems Symposium
SMTP	Simple Mail Transfer Protocol
SQL PL	SQL Procedural Language
SQL	Structured Query Language
SSL	Secure Sockets Layer
T-SQL	Transact-SQL
TCL	Tool Command Language
TLS	Transport Layer Protocol
UA	University of Aveiro
UML	Unified Modeling Language
UPS	Uninterrupted Power Supply
URL	Uniform Resource Locator
VBScript	Visual Basic Scripting
W3C	World Wide Web Consortium
WCMS	Web Content Management System
WWW	World Wide Web
WYSIWYG	What You See Is What You Get
XAML	Former eXtensible Avalon Markup Language, current eXtensible Application Markup Language
XHTML	eXtensible HyperText Markup Language
XML	eXtensible Markup Language

XSLT	eXtensible Stylesheet Language Transformations
Y2K	Year 2000 bug, originated from the abbreviation of 4-digit years to 2-digit years

Chapter 1

Introduction

Since 1995 the UA, by initiative of a few teachers of the, at the time, DET and currently DETI, has been developing a physical robotic competition, with certain fixed rules, named Micro-Rato[1] as a mean of motivating and complementing the department students' skills.

As the years passed by, the number of competition participants increased and the competition's popularity growth was evident, and became, not only a students competition, but also an important method for developing the desire to work in vanguard technological areas, and for contributing to a better understanding and usage of the seized knowledge with the participations. A proof of this was the propagation of this new kind of competition all around the country, also having been developed some other robotic competitions nationally based in the same principles as Micro-Rato at various stages of education.

With the popularity growth mentioned above, the competition itself would have to be adapted in order to meet its defined goals. Although Micro-Rato had ran with the same rules in the first five editions, the rules had to be significantly altered in order to better objectify the main goal (promote the robot's learning ability). In this sixth edition, that occurred in 2001, a new competition modality was also released, named Ciber-Rato[2] which reflected, basically, the same rules and ideals as its predecessor with the difference that this new competition wasn't physical, it ran in a simulated environment with virtual agents competing with each other.

1.1 Simulation Environment

The CiberMouse Design Competition is a simulated robotic competition based on the Ciber-Rato's Simulation Environment running in a network of computers, whose purpose is to develop an agent or multi-agents which will participate in a mission similar to a regular rescue mission. Each agent, or several agents as a team, have to find a certain spot, during a given amount of time, in an unknown maze and return to the initial spot or gather at the final spot, depending on the kind and level of competition.

The simulation system is responsible for the virtual arena which is composed of a initial position grid, a target area signaled by a beacon and a number of walls and obstacles as

pictured in figure 1.1. The simulation also modulates the robot's body which will afterwards be controlled by the software agents developed by every participant team.

Every robot has the same composition, a cylindrical shape and a number of sensors, actuators and command buttons. The simulation system emulates these sensor measures sent to agents and the actions sent by them as a response.

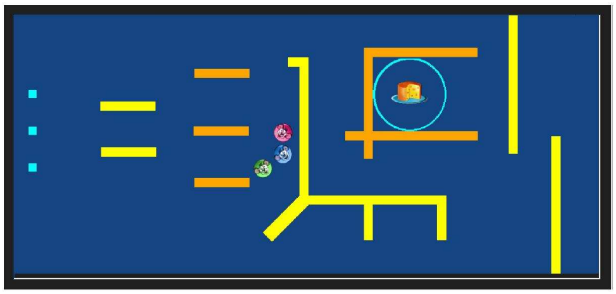


Figure 1.1: Virtual arena created by the simulation system, courtesy of the CiberMouse Organization.

The organizers main goal is to help improve the students ability to handle real-time challenges and to develop their overall robotic experience and knowledge through the competition.

The competition has several modalities within the above mentioned purpose, a high-school modality which is basically the regular Ciber-Rato competition but more simplified, a competitive modality which is the regular Ciber-Rato competition and a collaborative modality which was introduced recently, having an application available for download from the CPSS website which simulates this environment. The rules to each of these modalities will be explained in the following sections.

1.1.1 Robot Body

As mentioned before, the robot's body is composed of a cylindrical shape and various sensors, actuators and command buttons.

Each robot body includes up to four obstacle sensors, one beacon sensor, one compass, one bumper, one ground sensor and one GPS, as seen on figure 1.2. The bumper and the GPS are always available without the need to request them, but the other sensors are requestable sensors, which means they have to be requested in a previous cycle to be used, having a request limit for sensors of this kind per cycle. Since the competition is trying to simulate a real robotic environment, the measures given to the agents through the sensors are a bit noisy and suffer from a time latency, depending on the sensor requested, making the challenge more real.

The obstacle sensors receive the distance between the robot's body and an obstacle (walls or other robots), having each sensor sixty degrees of aperture angle. These

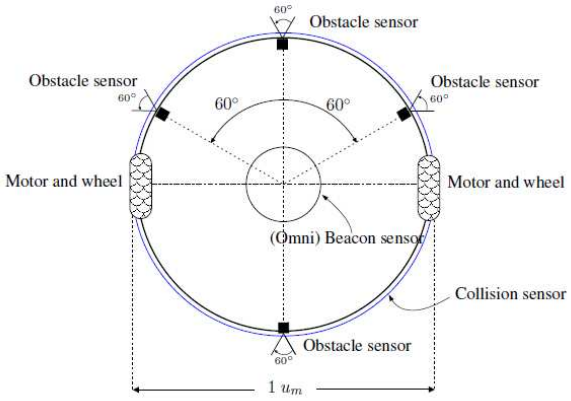


Figure 1.2: Virtual body of a CiberMouse agent, courtesy of the CiberMouse Organization.

sensors measurements are inversely proportional to the shortest distance to the obstacle, as the obstacle becomes closer the measures increase, with a range of [0,100] and a 0.1 resolution. These sensors are all movable around the robots periphery, being its default positions represented on figure 1.2.

The beacon sensor is omnidirectional and is positioned on the center of the robot's body. It measures the angle between the beacon's position and the robot's frontal axis, with a range of [-180,180] degrees and 1 degree resolution. A beacon may only be detected at a certain distance and if not behind a high wall through which it can't be seen.

The compass is also positioned on the center of the robot's body and it measures the angle between the robot's frontal axis and the simulation environment's virtual North, which is assumed as the X axis on the grid. This sensor also ranges in [-180,180] with 1 degree resolution.

The bumper or collision sensor acts as a ring around the robot's body, being activated whenever the robot collides with an obstacle. This sensor is always available with no time latency and its measures have no noise since it's basically a boolean variable.

The ground sensor simply detects whether a robot is completely over the targeted area, indicating either not over the area or completely over the area. This sensor is requestable, but has no time latency and no noise.

The GPS is a sensor located at the center of the robot's body and indicates its global position within the arena, also having a latency of 0 time units.

A summary of the above sensor characteristics can be seen in table 1.1.

Sensor	*Range*	*Resolution*	*Noise Type*	*Deviation*	*Latency*	*On Request*
Obstacle	*[0.0, 100.0]*	*0.1*	*Additive*	*0.1*	*0*	*Yes*
Beacon	*[-180, 180]*	*1*	*Additive*	*2.0*	*4*	*Yes*
Compass	*[-180, 180]*	*1*	*Additive*	*2.0*	*4*	*Yes*
GPS	-	*1*	*Additive*	*0.5*	*0*	*No*
Bumper	*Yes/No*	*- N/A -*			*0*	*No*
Ground	*Yes/No*	*- N/A -*			*0*	*Yes*

Table 1.1: *Robot's sensors characteristics summary.*

The buttons referred above are a Start button and a Stop button and are actuated uniquely by the simulator. The Start button is used to start or restart a previously interrupted competition and the Stop button is used to suspend the robots movement, being the robots duty to read these buttons' status and act according to it.

The actuators referred will be explained later on as the available actuators depend on the modality of the competition.

The game arena, depicted on figure 1.1, is a rectangular outer delimited maze with several inner walls, obstacles and one or more target areas. The maze is $14\mu_m$ high and $28\mu_m$ wide, all obstacles/inner walls are at least $0.4\mu_m$ wide, although they can be larger, corners' angles between inner walls must be ranged in [90;270] degrees, some inner walls may be higher than others and, when parallel, forming a corridor, must be, at least, $1.5\mu_m$ apart.

The target area's radius must be at least $2.0\mu_m$ wide with the beacon in its center, acting as a simple symbol in the maze and not an obstacle. Since this is a purely virtual environment, all measures of the arena and its objects are related to the robot's size which is $1\mu_m$ (one MicroMouse) and all time measures are related to the cycle time (varies with competition) which is μ_t.

As mentioned above, the actuators in the robot's body, although they are all present in every modality, each one of them uses only some of the actuators according to their actions' needs, namely:

The motors drive two independent wheels, one on each side of the robot as seen on figure 1.2, that depend on the power applied to them in order to move the robot around. The power applied to the wheels ranges in [-0.15;0.15] with a resolution of 0.001 units, although this isn't the actual power applied due to the inertia, and it is continuously effective until a new power is inserted. These actuators always have the same behaviour regardless of the modality.

The three LED's are a Beacon LED, a Return LED and an End LED. In the competitive modality the Beacon LED must be activated when a robot is visiting the targeted area and deactivated before the robot leaves, the Return LED must be activated before the robot leaves the targeted area and stay activated throughout the return

Actuator	*Range*	*Resolution*	*Noise Type*	*Deviation*
Motor	*[-0.15; 0.15]*	*0.001*	*Multiplicative*	*1.5%*
End LED	*On/Off*	*- N/A -*		
Return LED	*On/Off*	*- N/A -*		
Beacon LED	*On/Off*	*- N/A -*		

Table 1.2: *Competitive and collaborative modalities robot's actuators characteristics summary.*

trip to the initial position. Once the robot reaches its approximate initial position, the Return LED must be deactivated and the End LED must be activated, indicating that the robot has finished. In the collaborative modality merely the End LED is used and it must only be activated once the robot is completely parked inside the target area, signalling the end of its goal.

A summary of the competitive and collaborative modalities robot's actuators can be seen in table 1.2.

1.1.2 Simulation System

The CiberMouse's virtual environment is based on a distributed architecture, composed of one simulator, one or more visualizers and exactly three or five agents per trial, depending on the competition modality, as depicted on figure 1.3.

The simulator is responsible for implementation of the robots' bodies, estimating sensor measures with noise and sending them to the respective agents, moving an agent around the arena but taking in account the environment's restrictions, setting and updating the robot's score according to the completed goals, time used for completion and occurred penalties, sending all environment and competition related information to the visualizer(s), and also responsible for routing the messages sent by robots, having in consideration the communication constraints (message size and communication radius).

Each visualizer, on the other hand, is responsible for all the graphical representation of the competition on screen (robots, arena, scores, etc.) and making available a control panel to control the simulation (may be visible or not).

1.2 Competition Modalities

Such as was mentioned before, the CiberMouse competition is divided into several modalities, each with its own goal and specific parameters, but sharing common main characteristics, which allow the globalization of a simulation tool.

Since all these modalities share the same main features, only one application is needed to simulate them, so the CiberMouse Organization developed the CPSS' simulator, which, by defining the modality's challenge and its noise, latency and number of requestable sensors, can be applied to any of the CiberMouse's modalities.

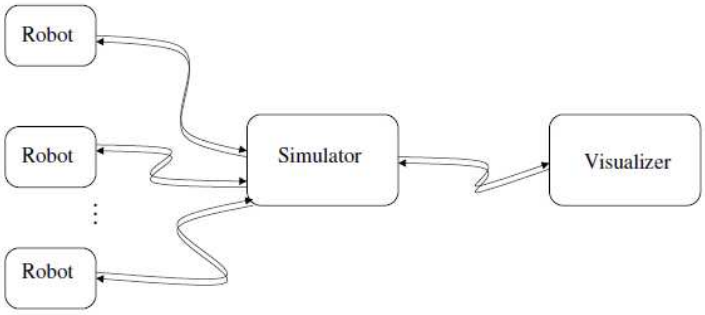


Figure 1.3: Competition's Simulation System Overview, courtesy of the CiberMouse Organization.

1.2.1 Competitive Modality

Continuing the previous rules explanation, in this modality each team is represented by one robot competing with two other teams simultaneously in the same arena.

Each robot has to accomplish two goals in order to conclude successfully this competition, find and visit the targeted area, and afterwards return to its initial position in the arena.

Challenge

As explained above, this competition has three teams competing simultaneously with one robot each and its challenge is to complete successfully two goals, visit the target area and return to the initial position.

Each robot must find the target area and visit it and once there the agent must activate the Beacon LED to indicate it has reached the target area and deactivate it before leaving. Afterwards the agent must activate the Return LED before exiting the target area and then begin the return journey to its initial position in the arena, having the Return LED activated throughout the trip. Once the robot reaches its approximate initial position, the agent must deactivate the Return LED and afterwards activate the End LED indicating the end of the challenge. The time limit in this competition varies in $[1800;3600]\mu_t$ depending on the scenario.

At the beginning, to each robot is assigned a score, that will be partly removed if the robot completes successfully some tasks. In addition, each collision or mistake the robot makes increases its score. At the end of each trial the robots are aligned by descending score, the best score being always the lower.

1.2.2 Collaborative Modality

Continuing the specific modalities' rules explanation, in this modality each team is represented by five robots working together to achieve one goal as a team. In this modality

there is only one team of robots per trial, meaning there is no direct confrontation in the arena.

The goal of the team is to organize itself, ideally communicate with each other (not mandatory), find the beacon and gather inside the target area. The presence of all robots inside the target area is also not mandatory to complete the goal, but more robots inside the area means better score.

Challenge

As mentioned, this competition has only one team per trial competing with five robots simultaneously and its challenge is to complete successfully only one goal, unlike the other modalities, which is to find the beacon and park inside the target area. Team collaboration with communication is ideal but not mandatory to complete the goal successfully.

Each robot starts in a different part of the map with their positions less than $8\mu_m$ apart so that connectivity between robots is present. Ideally, the robots should explore the maze by going into different areas, in order to minimize the time to find the beacon. Since communication is possible, they would be able to communicate whatever they wanted throughout the trial, as long as the communication radius isn't exceeded ($8\mu_m$).

In this modality there are two time limits to accomplish the goal, the key time and the total time. The key time is lower than the total time and is used to benefit the robots that signal their ending first, the total time is the time limit to conclude the trial and varies in $[1800;3600]\mu_t$ depending on the scenario.

As the other modalities, to each robot is assigned an initial score, that will be partly or fully removed if the robot completes the goal successfully (fully removed in case the robots completes within the key time, otherwise only part of the score is removed). In addition, each robot's collision increases its own score.

At the end of each trial each robot has a score and the team's score is the sum of the team's robots' scores. Afterwards the teams' score are descendingly aligned and, again, the best score is always the lower.

1.2.3 High-School Modality

This modality, although it is being referenced as a different modality of the competition, is actually just a particular of the other above presented modalities, since it can be applied to either one of them and is only separated to distinguish the different levels of the relative modality.

Since the above described modalities require a certain level of programming knowledge and skills, it would a bit difficult for people other than University students of a programming area to participate in these events with success. So, the High-School modality was created as a way to simplify the existent competitions, lowering their level of difficulty, by removing certain relevant aspects which would augment the complexity of the developed agents and their data.

In this manner, the usual CiberMouse modalities could be open to anyone who was interested in participating, whether they are University students from non-programming areas, high-school programming enthusiasts or any other non-schoolers who wanted to learn, improve or test their programming knowledge in a real-time simulation challenge.

At the moment this modality is only applied to the Competitive modality, inheriting all the robot body's specifications, the challenge's rules and the simulation system's environment definitions, excluding only the environment variables' noise, the sensors' data noise, deviation and latency, the actuators' noise and deviation, and also unlimiting the number of requestable sensors per cycle.

Even though this modality is only implemented for the above mentioned modality, it can easily be applied to the Collaborative modality or to any other kind of future CiberMouse competition modalities, since it inherits all the competition's relative specification and definition data from its *parent modality*, only removing the noise, deviation and latency from the measured data, and also its number of requests limit per cycle.

1.3 Objectives

These competitions have an increasingly higher level of work to organize and diffuse, making it continuously difficult to carry them out more frequently, and since Ciber-Rato competition is already spreading internationally, the need to have a centralized management portal which would embrace these competitions is extremely elevated.

The objective of this project is to develop a centralized management portal in order to run simulated competitions more frequently and completely autonomously, but also in order to be an important support tool for international and localized events, such as RTSS's Students Design Competition named CiberMouse, which occurs once a year, and UA's own Ciber-Rato Competition, which also occurs once a year on university grounds.

The portal would need to play two roles in the competitions. The role of active entity which regulates the players, teams and competitions, and is responsible for the presentation of these competitions' results, whatever level of difficulty or frequency of the competitions. But also the role of passive entity which functions as tool for players and teams to take part of the localized competitions, and to help organizers and judges with the presentation of results.

1.4 Thesis Structure

This thesis has the following structure:

In chapter 2 is presented an overview of the state of the art in Web development, namely methods, technologies and some specific applications (Content Management Systems, discussion applications and virtualization applications).

In chapter 3 is described the structural definition of the management portal solution. There are requirements explanations, actors and actions descriptions, and some applica-

tions integrated in the portal.

Chapter 4 analyzes the business logic layer and the portal's interface regarding some development aspects and implementations overview.

Lastly, chapter 5 discusses the conclusions achieved at the end of this project and presents some suggestions for future work on the portal.

Chapter 2

Web Development and Publishing

In this chapter an overview of the Web development technologies, methodologies and tools is presented. Several current technologies are discussed as well as some development methodologies used regularly nowadays. Some applications used as web development or publishing tools are also presented, namely CMS and other online discussion methods/applications.

2.1 Current Technologies

Ever since the beginning of the World Wide Web up until now, the WWW and the technologies around it have been constantly changing and extended from the simplest way of communication and information display to a new complex way of interacting and doing business, affecting every area in the information technology industry.

With this evident evolution, it quickly becomes critical to the companies' survival the implementation of a new breed of web applications. Consequently, these applications demand a platform providing production-quality tools for content management, application development and integration, so developers had to constantly adapt their knowledge in order to meet the additional requirements of this new generation of internet websites and applications.

An overview of some of the most important technologies and designs that are affecting the current web development market is hereafter presented in the following subsections, such as web servers, database management systems, dynamic webpages technologies for server and client sides scripting, content management systems, online discussion methods and some development methodologies and concepts.

2.1.1 Web 2.0

The Web 2.0 (figure 2.1) term was used for the first time by Darcy DiNucci on her article "Fragmented Future" in 1999[6] and later again in 2003[7][8], but the concept was only fully introduced with a conference in 2004[9], by the companies *O'Reilly Media*[10]

and *MediaLive*, and reflects the basic ideas of web development and web design as a form of interactive sharing, interoperability and also design and collaboration centered on the user.

In fact, although the name connotes a new version of Web 1.0, it doesn't refer to any kind of technical specifications' update, it's a series of changes on how software developers and users interact with the Web. Still, this concept is not generally accepted by some technology experts such as Tim Berners-Lee, inventor or the *World Wide Web*, who defends[11] that most of this newly created Web 2.0's technological components had been created even before the appearance of the Web 1.0. Some critics also call this concept a kind of marketing move or a "piece of jargon".

Figure 2.1: Web 2.0 cloud-map picture, courtesy of Luca Cremonini.

Characteristics

The main idea of Web 2.0 is to allow users to do more than just retrieve information. It takes advantage of the Web 1.0's capabilities to provide a "Network as a platform" concept, enabling the users to run software applications entirely through a browser window.

Normally, the websites built following this concept have a *participation architecture* that invites users to, not only retrieve information, as said before, but also add information to the application in order to increasingly enrich it through an AJAX interface and similar client-side interface frameworks, or even through full client-server application frameworks.

The essential attributes within the Web 2.0 concept, according to David Best from the University of Eindhoven[12], are its rich user experience and participation, dynamic content, web standards, scalability, openness and freedom and collective intelligence through user contribution. It also has the downside that not all users contribute to enrich the information provided throughout the web, but rather withhold their contribution of effort and free-ride on the others' contributions. This, of course, is a step back towards the Web 2.0 ideal and, sometimes, forces website managers to apply a *Radical Trust* policy, which consists on crediting the information according to the confidence level an organization or individual has in an online community.

The Web 2.0 also has some development rules which should be followed[13]:

Eternal Beta meaning software shouldn't be treated as a completed artifact, but rather as a commitment process with its users, providing the software as a service that is constantly being improved without the need to install any software on the end-user's equipment.

Loosely bonded modular programming which empowers the sharing of own data and services and also the reuse of others' data and services.

Device independent software in order to implement web applications that aren't limited to a single client or a single server environment, using them both as a team to improve the applications' interface and information exchange, bringing the *Web as Platform* concept more real and closer to desktop applications.

Data is the new "Intel Inside" as a analogy to, probably, the most important feature in Web 2.0, comparing the CPU as a computer's heart to website data as an application's heart, pointing to applications as simply tools to manipulate its data.

2.1.2 Servers

Web Servers can be described as software that is installed on a machine and whose purpose is to accept HTTP requests from clients, such as web browsers, and serve them their respective HTTP responses and some optional data contents, such as HTML webpages and linked objects (images and other media). A Web Server commonly also refers to, not only the software, but also the machine in which it is installed and running (a sample operation of this second notion of web server can be seen in figure 2.2, where a computer, considered as Web Server Machine that includes the respective web server software, serves the web users' computers and the server administrators' computers through the public internet network).

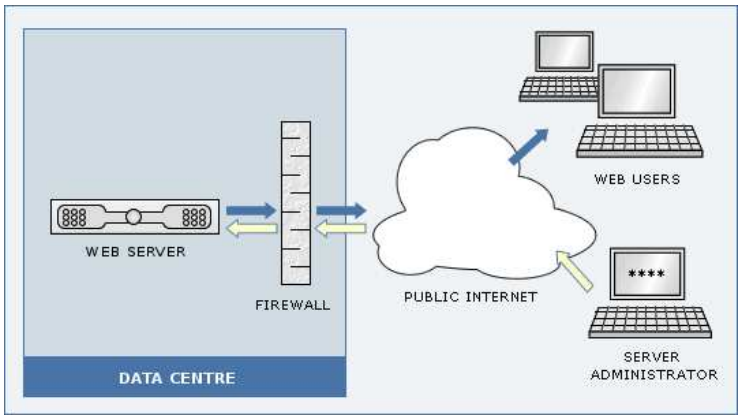


Figure 2.2: A Web Server's sample operation with its users and administrators, courtesy of the iServe Corporation.

Web Server	*Percentage*
Apache HTTP Server	*54.32%*
Microsoft IIS	*20.05%*
Google Web Server	*14.53%*
Nginx Web Server	*5.22%*
LightTPD Server	*0.34%*
Others	*5.54%*

Table 2.1: *NetCraft's August 2009 Web Server survey results, courtesy of the NetCraft Organization.*

In the context of this thesis, the notion of Web Server only concerns the software area of this field, so the future references to the notion of Web Server are always connected to the first notion described above, describing Web Server as a software application.

According to a Web Server survey in August 2009 led by NetCraft[14], the top servers with active pages are *Apache HTTP Server*[15], *Microsoft IIS*[16],*Google Web Server*[17][18], *Nginx Web Server*[19] and *LightTPD Server*[20]. Table 2.1 presents the survey's results in percentage and it shows that, currently, *Apache HTTP Server*, *Microsoft IIS* and *Google Web Server* are the most popular web servers. Since *Google Web Server* has been confirmed as a new compilation of *Apache HTTP Server* with some minor changes, it will be dropped from discussion and only *Apache HTTP Server* and *Microsoft IIS* will be better discussed ahead.

Apache HTTP Server

Apache HTTP Server[15] (figure 2.3), or more commonly just Apache, was initially created by Robert McCool, who had been involved in NCSA HTTPd, and a number of other contributers that developed some patches for the server, and later joined and founded the Apache Software Foundation.

Figure 2.3: Apache HTTP Server logo, courtesy of the Apache Software Foundation.

In fact, one of the two theories for the naming of this server derives from the patches made for the server, since the group called it *a patchy web server*. The other theory is that the name was chosen as a tribute and sign of respect for the Native American tribe of Apache. Both theories have been backed up by the Apache Software Foundation, so, although they've both been stated true by the group, there isn't a conclusive idea of which is the inspiration.

The web server is currently at a 2.2 version, which seems low, but there is a lot of history behind its development and the server is still being improved, developed and maintained

by the Apache Software Foundation. Its main characteristics are the fact that it is an open source software and is completely OS independent.

This web server has the ability to extend its core functionality by adding some compiled modules, such as server-side programming language support (PHP, Perl, CGI, Java, C++, ASP - still at its beginning but improving its support, etc.), authentication schemes, SSL and TLS support, a proxy, a URL rewriter, custom log files and filtering support. There are also some other popular features which can be integrated in Apache, like compression methods modules and intrusion detection and prevention for web applications[1].

Apache HTTP Server is mainly used for creating dynamic pages in PHP language and, since it is free to use, it is often used with the popular environment LAMP, although it supports other languages and environments, for example, this server is integrated into Mac OS X as its built-in web server.

Microsoft IIS

IIS[16] (figure 2.4) was created by Microsoft as a response to the launch of the Apache HTTP Server by the Apache Software Foundation. It has been distributed as an optional application for use solely in Microsoft's Windows environment, since Microsoft Windows NT 3.51 up to Microsoft Windows Server 2008 R2 and Windows 7, providing FTP, SMTP, NNTP, HTTP services and some others.

Figure 2.4: Microsoft IIS logo, courtesy of the Microsoft Corporation.

This project was initially researched by EMWAC and distributed as freeware, however, Microsoft was forced to develop its own server due to the increasing volume of traffic going to microsoft.com. At the beginning the web server was just an additional set of Internet services, but, as it became popular, Microsoft introduced the ASP dynamic scripting environment with IIS's version 3.

This server's latest version is 7.0 which is distributed for use in Windows Vista, Windows 7 and Windows Server 2008, not working on any of the other windows versions. In order to allow the former Windows' users to keep their OS and still use IIS, versions 6.0 and 5.1 are still available and being maintained. Version 6.0 already supports IPv6 and is only for use with Windows Server 2003 and Windows XP Professional x64 Edition; version 5.1 lacks IPv6 support and also only supports 10 simultaneous connections and one single website, this version is only for use with other distributions of Windows XP.

[1]ModSecurity is an example of an open source module of this type

Although this server is highly limited in its OS compatibility (only works in Windows environments), it still supports a wide variety of programming languages, such as VBScript, Java, C++, Python, ASP, CGI, PHP, JSP, Perl, among others. In concern to its security features, since it is highly integrated with its base OS, it is as simple and as secure as the OS.

IIS 7.0 has the advantage of allowing an unlimited number of simultaneous connections, only limiting the concurrent requests up to 10 (depending on the Windows distribution it is installed in), unlike IIS 5.1 on Windows XP, that limited the number of connections to 10, rejecting all others. Since IIS 7.0 and 6.0 have their own HTTP.SYS kernel driver they can be much faster than its predecessor IIS 5.1, yet, some benchmarks show that they are still slower than other servers that run in user-mode.

2.1.3 DataBase Management Systems (DBMS)

DataBase Management System (DBMS) is an application or applications whose objective is to control the creation, maintenance and use of an organization's database by its own applications or end-users.

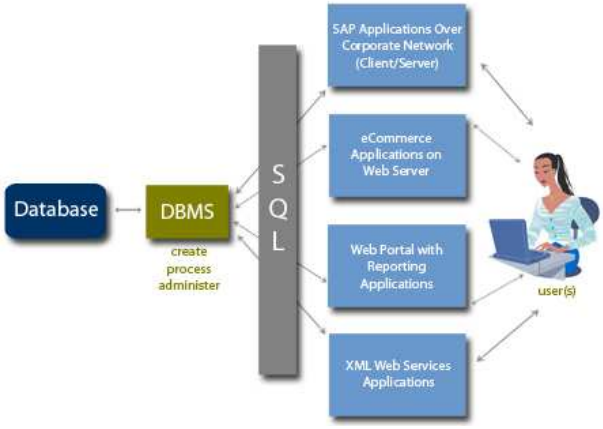


Figure 2.5: A database management system's operations scheme, courtesy of the SystemsView Website.

An organization's database is a collection of the its information organized into logically related records or files consolidated in a common pool of data records in order to be easily accessed, managed and updated. These data records are persistent (meaning that they are stored regardless of terminating the users' sessions, terminating the management applications or shutting down the databases' computers) and organized according to a database model, which can be the flat model, the hierarchical model, the network model, the relational model (the most common currently), the dimensional model or other objectional database models.

As seen on figure 2.5, the database holds the organization's data persistently and is managed by a DBMS which, by communicating with several possible applications through

a certain language (databases and applications must communicate through the same language in order to exchange data successfully), receives information and provides information to the end-users of those applications.

Although there are several database query languages, the most common worldwide and the most widely used in relational databases is SQL, as it is a set-based, declarative query language which provides an easy method of communication with relational databases to perform several tasks, such as data query and update, schema creation and alteration, and also data access control. However, in order to add a procedural programming language functionality to SQL, some extensions where developed by several organizations to better adapt to their needs, such as:

- PL/PSM by PostgreSQL, which implements the SQL/PSM from ANSI/ISO Standard;
- PL/SQL by Oracle which resembles Ada, a structured, statically typed, imperative and object-oriented high-level programming language extended from several other languages, mainly Pascal;
- PL/pgSQL by PostgreSQL, which is based on the above PL/SQL from Oracle;
- PSQL by Interbase/Firebird;
- SQL PL by IBM which implements the SQL/PSM from ANSI/ISO Standard;
- SQL/PSM by ANSI/ISO Standard;
- SQL/PSM by MySQL which also implements the SQL/PSM from ANSI/ISO Standard mentioned above;
- T-SQL by Microsoft/Sybase;

The usage of these DBMS's has as main capabilities:

- The increase of the data storage's efficiency by enabling the access, process and altering of large data efficiently and orderly;
- Persistence feature, allowing to maintain the data on a physical storage indefinitely, regardless of the number of applications that use it and regardless of the system's availability (the system could be powered down and the data would remain untouched);
- Increased robustness, so in case of a hardware or software failure, the data remains consistent;
- Improved access control in order to allow access to multiple users with possible multiple access rules, raising the security level and consistent data access.

Along with these main capabilities, there are a lot of other features which empower these DBMS's even more, such as:

- The ability to perform queries to retrieve, update or remove information from the database;
- A backup and replication feature which enables the database administrator or similar privileged user to make copies of the database's structure and information in case of a a system failure or in case of a server migration or extension, keeping the consistency of the saved data;
- Rule enforcement techniques which allows the database users with appropriate privileges to apply rules to attributes in order to ensure that they are always cleanly inserted or updated and that they are reliable, allowing the modification or removal of those same rules later with ease;
- The capability to perform computations on the data or query results (like counting, summing and averaging, among others), relieving the client's application form implementing those calculations;
- An alteration and access logging capacity in order to keep a record of all actions performed by and through the DBMS;
- An automated optimization feature which monitors occurring patterns or requests and adjusts the DBMS providing a speed increase on those interactions or informs a database administrator of the statistics allowing him to perform the necessary adjustments.

Since the most common DBMS's used are Microsoft SQL Server, MySQL, Oracle and PostgreSQL, they will be more specifically discussed ahead. A small reference to DBLite will be made ahead too as a light weight DBMS solution.

DBLite

The DBLite is a light weight Relational DataBase Management System (RDBMS), since its main objective is not to compete with other real world RDBMS, but rather provide a simple standalone and single-user DBMS with relational features that don't have to be installed separately and enable its use without requiring an individual complex RDBMS.

In fact, DBLite was written in 1999 with the purpose of being included within other projects with no other dependencies, being able to reside in memory or disk for embedding with other resources.

This DBMS allows the application's developer to manipulate tables consisting of rows of data which are serialized vectors of objects. These tables may be designed in an administrative application, used in a desktop or web application and queried back in the administrative application, or any other combination of the above order. As features it

provides a SQL-like syntax, an GUI table designer and administration application, a test suite, some pre-compiled statements and the possibility to perform transactions with roll-back.

On the upside, this DBMS doesn't have a separate install and is distributed embedded within the application, has important relational features for single-user applications and provides the ability to share compatible binary data files with other applications. On the downside, the SQL-like syntax is very basic and weak, it doesn't provide sorting or grouping capabilities, it has no support for aggregations, and at the moment its syntax parser is flawed and the management GUI is a bit slow.

Microsoft SQL Server

The SQL Server[21] (figure 2.6) was initially developed by Microsoft in 1989 and its code base was originated in Sybase's SQL Server. It was Microsoft's entry to the enterprise-level database market and was launched for OS/2[2], being later launched, in 1993, for the Windows NT 3.1.

Figure 2.6: Microsoft SQL Server 2008 logo, courtesy of the Microsoft Corporation.

In 1995, Microsoft abandoned its former Sybase design and redesigned a new SQL Server from scratch, launching the version 6 for Windows NT (the first version specifically for NT). Ever since then, Microsoft has been continuously improving the SQL Server and also creating and improving some other complementary applications which were packaged with the server, and still are with the new SQL Server 2008 R2.

As all of Microsoft's applications, SQL Server also suffers from a lack of compatibility with OS's other than Microsoft Windows, only working under Windows environment.

The SQL Server 2008 R2 version (codenamed SQL Server "Kilimanjaro") claims to be a self-tuning, self-organizing and self-maintaining data management solution due to the development of Microsoft's *SQL Server Always On technologies*, in order to reduce the percentage of failures and downtime in this field.

It supports a new variety of support for structured and semi-structured data, such as digital media, allowing an easy access to this data through the use of newly developed data types in the DB. It also supports stored procedures, triggers, cursors and updatable views, SSL support and a GUI administration tool which is, as expected, limited to Windows environments and can get a bit heavy on some machines.

[2]Operating System/2 created by Microsoft and IBM

This SQL Server is a very powerful application and also includes in its package a large variety of other applications that, used along side with the DBMS, empower it even more and make it very easy to use without much database knowledge. The downside of this application is the fact that it must be used under Windows environment and its functionality is very restrict in terms of language.

It supports a lot of server-side programming languages, open-source and proprietary, but in order to take full advantage of its power and its complementary applications, it must be used with Microsoft's server-side programming language, ASP. In concern to hardware, it is very demanding due to the number of features it lays at the users disposal, but allows a great scalability power with a lot of ease, in terms of database growth and hardware replication.

MySQL

MySQL[22] (figure 2.7) is currently one of the most popular and successful RDBMS's in the world, as it has over 6 million installations. It stands for *My Structured Query Language* and has an open-source distribution (the most common) along with some other proprietary agreements.

Figure 2.7: MySQL logo, courtesy of the Sun Microsystems Corporation.

This server was initially developed by a single for-profit Swedish company, MySQL AB, and was recently acquired by Sun Microsystems in 2008, which now hold the copyright to the application's codebase. It is commonly used under the LAMP software stack which includes the famous PHP server-side language.

On the contrary of its rival, Microsoft SQL Server, this application is platform independent, which means it works on many different OS's, such as FreeBSD, Linux, MacOS X, SunOS and also Microsoft Windows. It also allows its use with a wide variety of server-side programming languages through language specific API's and an ODBC interface named *MyODBC* which allows additional programming languages that support the ODBC interface to communicate with the MySQL database, such as ASP and ColdFusion.

Another great benefit of this server's open-source policy is the number of commercial and non-commercial tools available developed by MySQL AB and other companies or users around the Web.

This DBMS also has a lot of history on its back; it was developed by Michael Widenius and David Axmark in C and C++ languages beginning in 1994, launching its initial release around May 1995. It's constantly being updated as every other DBMS, but as fallen a bit behind when most of the other DBMS developed the support for stored procedures, views

and triggers, recovering with the stable release of MySQL version 5 which introduced, among other technologies, the support for stored procedures, views and triggers.

Its main features, in addition to those referenced by most DBMS, are the following:

- Cross-platform support;
- Major languages support for accessing MySQL databases and an ODBC called MyODBC for languages that lack the MySQL standard support;
- Support for stored procedures, triggers, cursors and updatable views;
- A true VARCHAR support;
- Independent storage engines (MyISAM for read speed, Oracle's InnoDB for transactions and referential integrity, MySQL Archive for storing);
- SSL support;
- Embedded database library;
- Replication support (that is, Master-Master replication and Master-Slave replication);
- Included administration command-line tool;
- Various GUI administration tools, such as MySQL Administrator, MySQL Migration Toolkit, MySQL Query Browser and phpMyAdmin.

among others, and it also has some features that distinguish it from other DBMS, such as multiple storage engines, allowing its choice for each table in the application, having a wide variety of native, partner-developed, community-developed and custom storage engines; and such as commit grouping, allowing an increase of the number of commits per second.

Oracle

The Oracle Database[23] (figure 2.8) is, as the above RDBMS, an open-source DB application, including also proprietary distributions, developed by the RSIAcronym for Relational Software, Inc. company in 1979. Their first version of the RDBMS was named Oracle V2, although they never released a version 1, being considered as a marketing gimmick. Later in 1982, RSI changed its name to Oracle Corporation and they've been improving their RDBMS under that name ever since.

The Oracle Database was the first company to develop a commercially available SQL-based database in 1979, and the first to support symmetric multiprocessing in 1983. In 1986 the company introduced the concept of a distributed database system and in 1995 introduced the first 64-bit database. Later, in 1998, the company claims to be the first

Figure 2.8: Oracle logo, courtesy of the Oracle Corporation.

to incorporate a native JRE in its package and, in the same year, the first to develop a RDBMS available in Linux OS, also introducing in 1999 the first XML supporting database.

As evident in the above paragraph, the Oracle Database company was a pioneer in the database software market, being responsible for the introduction of several major technologies which, in today's software development techniques, are a must-have and a must-use for any kind of software applications and that facilitate and empower every database's use.

This RDBMS is also a very powerful database application and it remains one of the major presences in the market of database computing. It is characterized as a platform independent application, which supports most of the available OS's such as Linux, Microsoft Windows, Mac OS X and Sun Solaris, among others, just like its competitor MySQL RDBMS.

It is also very well known for its integration with the Java programming language, although it supports several other programming languages, as its rivals. The first Oracle Database versions, except version 2, were written using C programming language, and later in 1997, C++ programming language, but since 1999, the Oracle Corporation incorporated a native JVM in Oracle 8i version, which empowered its functionalities for use with the Java server-side programming language. Ever since then, the Oracle Database has been closely paired with the Java programming language for Internet and Desktop software development.

The Oracle Database application includes, among others, the following features within their standard package:

Active Session History (ASH) which stores recent database activity for monitoring purposes;

Automatic Workload Repository (AWR) which provides monitoring services to Oracle Database installations;

Data Aggregation and Consolidation to improve the quality of the stored data;

Data Guard to ensure a high database availability;

Generic Connectivity in order to easily connect to non-Oracle systems;

Data Pump Utilities to aid in the import and export of data between databases;

Database Resource Manager to control the use of computational resources;

Flashback to selectively recover and reconstruct data;

iSQL*Plus which is a web-browser based GUI for database manipulation;

Oracle Data Access Components which consist in tools to help access database information;

Fine-Grained Auditing which supplements the database's standard security-auditing features;

SQL*Plus which is a command-line application to interact with the database via SQL and PL/SQL commands.

PostgreSQL

PostgreSQL[24] (figure 2.9) is an Object-Relational DataBase Management System (ORDBMS), distributed under a BSD license and, therefore, free to use and open-source application. Unlike the above DBMS's described, the PostgreSQL application was and still is completely developed by a community of software developers and companies, not belonging to any specific company or person except the MySQL DBMS (which complies with both community and proprietary development techniques).

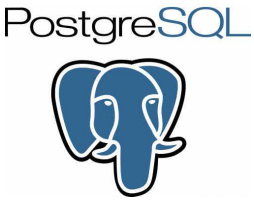


Figure 2.9: PostgreSQL logo, courtesy of the PostgreSQL Organization.

This DBMS has evolved from the Ingres database project started at the University of California, Berkeley, and later, in 1985, Michael Stonebraker who was the Ingres project leader began working on a post-Ingres project, named Postgres (from which the current PostgreSQL name was originated) that addressed the problems of the various database systems which became increasingly evident in the previous years. Although this new project used several ideas provided by its predecessor Ingres, its codebase was entirely developed from scratch.

As mentioned earlier, this project was early released under a BSD license by Berkeley, so, in 1994, graduate students Andrew Yu and Jolly Chen replaced the standard Ingres QUEL query language interpreter with the much more popular SQL query language interpreter, distributing this project and its codebase freely on the web as Postgres95.

Like the above graduate students, several others developed their own open-source or proprietary database applications based on the PostgreSQL project (named, in 1996, to PostgreSQL in order to reflect its standard support for SQL), such as Paula Hawthorn and Michael Stonebraker, who had worked in the Ingres project, the company Great Bridges,

which was created by some former Red Hat investors, the Command Prompt, Inc., and others. Even despite this, the main force of development of this project is still a group of database developers and volunteers around the world, via the internet.

As for this DBMS's features, it includes a wide variety of capabilities that place it on the top of the most used RDBMS's in the world, such as:

Functions which allow the database programmer to develop blocks of code to be executed by the server in several programming languages, like PL/pgSQL, PL/Lua, PL/Perl, plPHP, PL/Python, PL/Ruby, C/C++, PL/Java, among others;

Indexes which support scanning indexes backwards eliminating the need for a separate index, usage of expression indexes for indexing results of an expression or function, usage of partial indexes for indexing only parts of tables and usage of multiple indexes together to satisfy complex queries;

Triggers which are, as normal triggers, events triggered by the action of SQL statements, but these triggers can also invoke functions within it that can be written in several languages, as referred above;

Multi-Version Concurrency Control (MVCC) which is a system that manages database access concurrency, giving the users a snapshot of the database allowing them to make changes invisibly to others until the changes are committed, eliminating read locks;

A wide variety of data types which includes variable length arrays of up to 1GB, IPv4 and IPv6 addresses, CIDR blocks, MAC addresses, UUID, geometric primitives and arbitrary precision numerics, among others, and also custom data types created by the users;

User-defined objects which include new casts, conversions, data types, domains, functions, indexes, operators and procedural languages, among others;

Inheritance which allows to set tables to inherit their characteristics from a parent table;

Add-ons which allows the database administrators to add new functionalities and objects, created either by them or by others that share them openly.

2.1.4 Dynamic Pages

There are several ways to create webpages, some are completely static in which the user requests a webpage from a webserver and the server simply responds by sending the pre-written HTML content to the users browser, and some are dynamic in which it interacts with the user.

Static webpages, as can be seen in figure 2.10, are files which content is already determined before the client requests the webpage to the server, they are created by the webdesigner through HTML and CSS commands and stored in the webserver. These webpages are easy to construct, but, since its aspect and content will always be the same

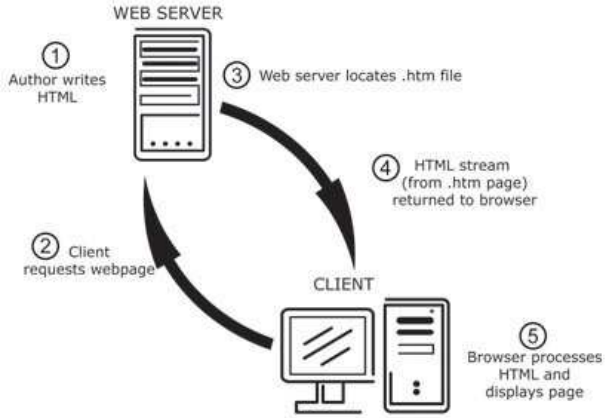


Figure 2.10: Interaction between client and server for static webpages, courtesy of the WebHosting Website.

independently of the viewing user, they are mostly used in websites with valid information that are rarely altered, like institutional webpages.

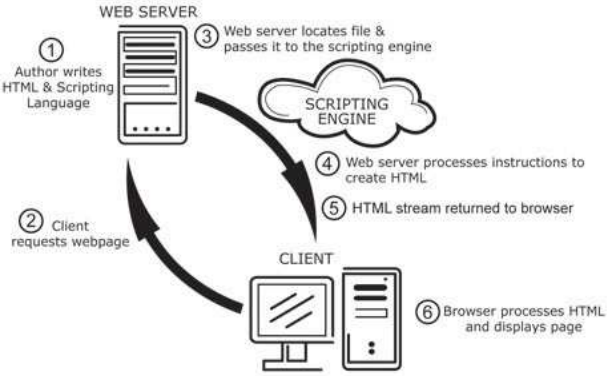


Figure 2.11: Interaction between client and server for dynamic webpages, courtesy of the WebHosting Website.

Dynamic webpages, as depicted in figure 2.11, are basically statically coded webpages which include, not only static HTML and CSS content, but also some other elements that interact with the user through several ways, like rollover events, popup windows, database information retrieval and presentation, or data operations' results.

Although these webpages are initially created in a static manner (every bit of code is created before the files' deployment to the server), the content viewed by the user is generated, through the interactive elements pre-defined at webpage creation, by the server before it sends the content to user or by the user's computer when it receives the content from the server.

These two ways of creating interactive webpages are called client-side scripting and server-side scripting. Throughout the years many technologies were used to include this kind of interactivity within the webpages sent to the users, such as Java programming language and ActiveX objects and components. Since these technologies require reasonably complex programming skills and excessive memory and processing capacity in the client's side, they were traded over the years by some other technologies which enabled the same interactiveness without the excessive requirements and programming skills, such as JavaScript/JScript, Adobe Flash, Microsoft SilverLight, among others that will be described ahead.

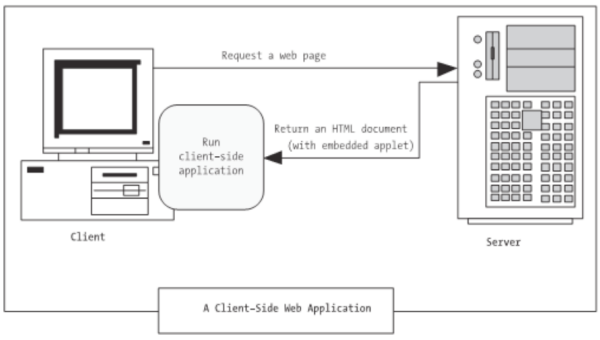


Figure 2.12: An example of a client-side scripting operation, from an unknown source.

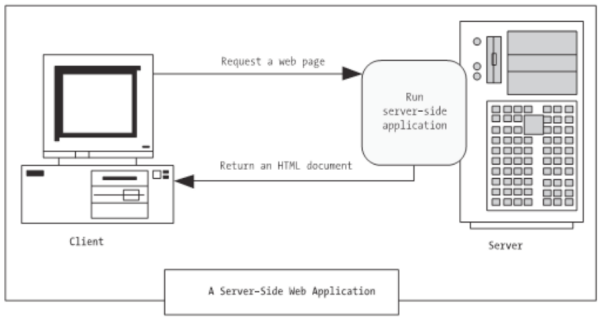


Figure 2.13: An example of a server-side scripting operation, from an unknown source.

As mentioned above, there are two ways of developing dynamic webpages, through client-side scripting and server-side scripting. The earlier written technologies belong to the client-side scripting techniques, as for the server-side scripting technique uses mainly web programming languages such as the more common ASP, PHP or JSP, and also the falling CGI, which will be described next. Sometimes, as a way to disguise their choice of programming languages for security reasons, some websites still use the common .htm or .html extension while including scripting commands within the code.

Common Gateway Interface (CGI)

The Common Gateway Interface[25], or more commonly CGI, is a standard protocol used as an interface between external application software and web servers. This protocol's purpose is to define a standard way of requesting the execution of a command to the server and afterwards return its output, by establishing how the information related to the server and the request is sent to the command as arguments and environment variables, and how the command can send back the output's related information in the form of HTML headers.

CGI was one of the first dynamic webpage presentation techniques approached, since it appeared in 1993, the early beginning of the WWW. It uses the web browser's URL as a path to a program that is executed via CGI, and when its request is received by the server the program indicated by the URL is executed. As mentioned above, data is normally sent to the program as environment variables, but in case HTTP PUT or POST techniques are used, that data is sent to the program in the form of arguments through the standard input. The result of this program's execution is, as stated above, sent back via the standard output prefixed by a HTTP header and a blank line.

Since this technology was becoming a bit underachieving due to its excessive memory needs and time requirements, and also due to an increasingly higher traffic load on the web servers, a different solution would have to followed in order to meet the higher demands in current networks, so, FastCGI was developed as a variation of the former CGI, whose main objective is to reduce the CGI's overhead associated with interfacing the web servers and CGI executed programs, enabling the web server to handle more page requests at once, consequently responding better to the higher traffic load and allowing a better system scalability.

The FastCGI protocol allows a single long-running process to handle more than one request at a time, still following closely the CGI's programming model, retaining its simplicity and remaining independent of the web server, but eliminating most of the overhead incurred by CGI, which created a new process for each request. Although this concept is not as used nowadays as it was before, it is still partially used and implemented by other very common server-side scripting languages, being available for standalone use or integrated with them.

As mentioned above, this technique's basic purpose is to be the most simple it can be and to provide the crucial communication basis between the client's web browser and the web server, but remaining independent of each other. So, its advantages and disadvantages vary from developer to developer, since, for some developers, FastCGI serves their every purposes and intentions, as for example the implementation of a wiki (the request references a name entry and the resulting HTML from the command executed is sent back to the browser), and for other developers a closer integration with the core web server is a highly important part of their application's implementation, ultimately using other tools for their development.

Active Server Pages (ASP)

Active Server Pages[26], also known as Classic ASP, is Microsoft's first server-side script engine for dynamically generated webpages and was, initially, released as a IIS add-on, later became included as a free component of Windows Server, and is currently superseded by ASP.NET (figure 2.14) . These files' extensions can be .asp for ASP webpages, or .aspx for ASP.NET webpages which are based on Microsoft's .NET framework and, therefore need another languages as base backend code which must be pre-compiled, making it faster and more robust than regular ASP, which is interpreted at runtime.

Figure 2.14: ASP.NET logo, courtesy of the Microsoft Corporation.

The Classic ASP was based on the dbWeb and iBasic tools, which were created by Aspect Software Engineering, and was one of the first development environments to integrate the execution of web applications directly in the client's browser. Most ASP webpages are written in VBScript, but Classic ASP supports development in several other Active Scripting engines, such as JScript or PerlScript.

The ASP.NET is the new version of the former Classic ASP, released in January 2002 with version 1.0 of Microsoft's .NET Framework, and is built on the Common Language Runtime, which allows the development of code to be made in a wide variety of programming languages, as much as the languages supported by the .NET framework itself.

These Classic ASP and ASP.NET are very powerful web application frameworks for creation of dynamic websites, web applications and web services, but, when Microsoft developed these frameworks had the intention that they would be restricted, as most of Microsoft's software projects, to usage in Web Servers running Windows environments. Another restriction in the development of web pages with Classic ASP or ASP.NET is the need to pay Microsoft's IDE license fees to develop.

Fortunately, although Microsoft developed these frameworks for Windows environment only, a lot of web servers, besides Microsoft's IIS, are introducing new features which allows to include support for Classic ASP and ASP.NET dynamic webpages, such as Apache HTTP Server, FastCGI and XSP. A lot of other application frameworks, such as the Mono Project[27] support Classic ASP and ASP.NET development, and, used with the web servers described above, allow web developers to built dynamic webpages independent of the runtime environment with little or no cost at all.

PHP:HyperText Preprocessor

Hypertext Preprocessor[28] or, more commonly PHP (figure 2.15) , is a server-side scripting language created by Rasmus Lerdorf in 1995 and has been continually developed

ever since, being, at the moment, developed by The PHP Group and released as a free software under the PHP License (a special license due to some restrictions in the general GNU General Public License).

Figure 2.15: PHP logo, courtesy of The PHP Group Organization.

It is a widely used and general-purposed scripting language, which was originally developed bearing has main objective the construction of dynamic webpages embedded into an HTML page, running on almost any web server available, on almost every operating system in the world and with a lot of development tools which support this language, with the possibility of developing a project absolutely free of charge (the developer may also choose to use a proprietary licensed platform, web server or IDE, in which case the costs are associated directly to the license fees).

In fact, this project first started as a personal project in 1994 as Common Gateway Interfaces written in C programming language in order to replace a set of Perl scripts he was using to maintain his personal webpage. The PHP acronym actually came from Personal Home Page and the tools he created were called Personal Home Page Tools, initially used to display his résumé and record the amount of traffic on the webpage.

Later, this personal project evolved into PHP/FI, which included his own Form Interpreter binaries, allowing the communication with databases and, thus, enabling the building of simple and simultaneously dynamic, webpages. This was the version released in 1995 as version 2 of the PHP programming language as we know it.

In 1997 the PHP parser was rewritten by Zeev Suraski and Andi Gutmans, and formed the PHP version 3 with a change of the PHP's acronym meaning (now it stands for PHP:Hypertext Preprocessor, as described above). These two former developers at the Technion IIT founded, in 1999, the Zend Technologies company with the release of the new PHP's core, called Zend engine.

The PHP4 and PHP5 were already released under the new Zend engine and are the main currently used distributions of the PHP interpreter, which currently includes an independent command line interface capability and may also be used as addition to standalone graphical applications. Also, a new version PHP6 is, at the moment, being developed by The PHP Group, but the release date is not yet set.

This server-side scripting language is very powerful, since it is extremely easy to learn and use, it has an excellent performance due to its reduced use of the system's resources and because there is no need to compile the PHP code in runtime (it is just interpreted when a webpage is requested), and, since it is completely free and portable, the developers are given a wide variety of choices of web servers and operating systems, allowing them to build webpages with absolutely no cost.

Java Server Pages (JSP)

JSP[29] (figure 2.16) is a server-side Java technology for building dynamic webpages in response to a client's request to a Java Application container (in this context corresponds to a web server). The JSP webpages are a simplified abstraction of Java servlets, they are deployed to the server and then operated by a J2EE normally packaged as .war or .ear file.

Figure 2.16: JSP logo, courtesy of the Sun Microsystems Corporation.

This technology allows Java code to be embedded into static HTML content and compiled at runtime, being operated by a virtual machine that is integrated with the client's operating system, in this case called Java Virtual Machine (JVM), although it still needs backend server-side Java code as a base support.

The first JSP version (JSP 1.0 specification) was released in 1999 as a response to ASP and PHP programming languages, by Sun Microsystems who developed both servlets and JSP. When the JSP 1.2 was released, the development became a responsibility of all interested third party developers under the Java Community Process, which regulated and described the developing process.

As mentioned before, this programming language is operated by a virtual machine integrated into the client's operating system, which enables it to be platform independent, running without problems on any operating systems that integrate a Java Virtual Machine, such as Microsoft Windows, Linux, MacOS X, among others.

On the other hand, although the OS is variable according to how much the developers intend to spend on licenses, the web servers may not allow the same liberty. However, this programming language is already supported by many of the available web servers mentioned, such as Microsoft IIS. There are also some web servers which were specifically designed and developed for use with JSP, like Apache Tomcat[30] by the Apache Software Foundation, Glassfish[31] by Sun Microsystems and JBoss[32] by JBoss Inc., which was bought by Red Hat in 2006 and is being developed by the JBoss Community.

As far as IDE goes, this programming language is also supported by a lot of development tools that range from some license cost to absolutely free, such as NetBeans[33], Eclipse[34] or MyEclipse[35], IntelliJ IDEA[36], among others, also giving the choice of whatever cost the developer wants.

Similarly to what was mentioned for the above languages, this framework has some advantages and disadvantages, depending on the developers point of view (some developers'

advantages may be others' disadvantages and vice-versa). That said, this programming language has a good performance, since, like ASP.NET, it pre-compiles the base server-side code before deployment in the server, meaning there is only simple runtime interpreting needed, being more laborious at the time of development. However, although Java introduced the term JavaBeans and J2EE introduced Enterprise JavaBeans as a way of reusing components and developing modular enterprise applications, unskilled Java, J2EE or JSP developers might find these technologies a bit hard to learn, unlike PHP.

ECMAScript

ECMAScript[37] is a client-side scripting language regulation standardized by ECMA International (figure 2.17) in the ECMA-262 specification[38]. This language standard is widely used on the web and is best known as JavaScript, ActionScript and JScript (the current most popular dialects of ECMAScript).

Figure 2.17: ECMA International logo, responsible for the ECMAScript standard, courtesy of the ECMA International Organization.

ActionScript is a scripting language used on the client's side, which is integrated in the Adobe Flash platform (in the form of SWF files embedded in webpages) for the creation of dynamic websites and is also used in some database applications such as Alpha Five. Since this scripting language is mostly used with Adobe Flash and it needs the Adobe Flash Player plugin to be used, it will be discussed later in following sections.

JavaScript is a client-scripting language and, since it is implemented as an integrated component of the web browser, it enables the development of enhanced graphical user interfaces and dynamic webpages. As referenced above, JavaScript is a dialect of the ECMAScript standard ECMA-262 and was influenced by many languages, being designed to be similar to Java, but easier to learn and operate for not very skilled programmers to work with.

JavaScript was initially developed by Brendan Eich for Netscape and was introduced and deployed in the Netscape Navigator web browser in December 1995 which coincided with adding support for the Java technology in this browser's version. Although the JavaScript scripting language and the Java programming language share several similarities, the JavaScript is essentially unrelated to Java and was named as JavaScript as a result of a marketing deal between Netscape and Sun Microsystems, in exchange for the addition of the Java technology into the, at the time, market dominant web browser.

Due to the great success of the JavaScript as a client-side scripting language for dynamic webpages and since Netscape refused to license JavaScript for use with Microsoft's Internet

Explorer web browser, Microsoft was compelled to create its own version of the JavaScript scripting language based on the ECMA-262 specification, and so, Microsoft's new scripting language was born as another ECMAScript dialect, named JScript to avoid trademark issues and used only in the Internet Explorer web browser, since Microsoft also denied licensing it to others.

The launch of Microsoft's JScript also added and corrected a few features present in JavaScript, namely new date methods to fix the Y2K bug[3] which was still present in Netscape's scripting language. Still, with the advance of these scripting engines throughout the years, JScript became increasingly more a kind of Microsoft standard instead of an ECMA standard, being, in many points, non ECMA-compliant.

Netscape, however, in order to transform JavaScript into a universal scripting language, submitted it to ECMA International for standardization. Although many professional programmers denigrated JavaScript because it targeted web authors and amateur programmers, with the integration of this fully standardized version of JavaScript into many technologies, such as AJAX which will be discussed ahead, it became one of the most popular programming languages on the web and is nowadays used in conjunction with others server-side scripting engines and is also expanding as a server-side JavaScript platform.

Asynchronous JavaScript and XML (AJAX)

AJAX[39] (figure 2.18) stands for Asynchronous JavaScript and XML, and, although it sometimes can be mistaken for a client-side scripting technology, it is, in fact, a group of technologies combined together to be used on the client-side to create interactive web applications, with the advantage of these technologies already being standardly compatible with any browser and platform, since no additional plugins are normally needed.

Figure 2.18: AJAX logo, courtesy of the AJAX Organization.

AJAX's constituents are XHTML/HTML and CSS for marking up and styling information; DOM which, used through JavaScript, can dynamically display and interact with the information presented to the client; XMLHTTPRequest (XHR) objects, IFrame objects or dynamically added <script> tags for exchanging data asynchronously between the

[3]Year 2000 bug originated from the abbreviation of 4 digit years to 2 digit years

server and the client's browser, avoiding constant page reloads; XML, XSLT, pre-formatted HTML, plain text or JSON as a format for the data sent to the browser, this data can also be created by server-side scripting.

In the traditional webpages developed in an either static or dynamic technique, each time the client performs an action on the webpage, the information relative to that action is sent to the server and, according to the scripting technology used (client-side, server-side or both), the response is calculated and transformed into HTML and CSS content to be presented in the client's browser.

This, obviously, can be a very sluggish task, specially at times of high network traffic or in the case of high volumes of information being exchanged between client and server. Also these times can be even more augmented if there is the need to connect to various systems to perform a certain operation (either simultaneously or individually).

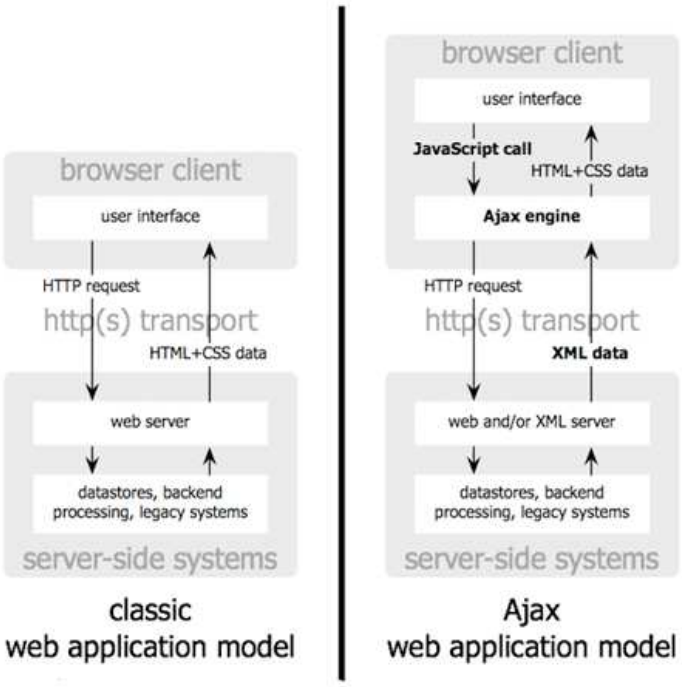


Figure 2.19: Interactions between client and server using traditional methods and AJAX, courtesy of Jesse James Garrett from Adaptive Path Website.

With the use of AJAX in the development of the webpages, the operations described above act a bit differently. Each time the client performs an action on the webpage, the information relative to that action is sent to the AJAX module developed by the author of the webpage and operating in the client's platform (the exchange of information between the client and the AJAX module is completely transparent), in case the response is already available in the module the resulting HTML and CSS content is generated and displayed in the browser, in case the response is unavailable, a request is made to the server only for

the needed information and the needed information is sent to the AJAX module where it will then form the response to be presented in the browser.

A simple interaction sketch can be seen in figure 2.19. The integration of the AJAX technologies in the development of dynamic webpages can effectively decrease the amount of unnecessary traffic in a network, since only the crucial information to the performed actions are exchanged, avoiding the need to resend information already sent earlier to the client; and also a decrease of the time the user has to wait for the server's response, since this method is asynchronous, the user can continue interacting with the webpage until the AJAX module receives the information to present in the browser.

Adobe Flash

Formerly known as Macromedia Flash until it was acquired by Adobe Systems, Adobe Flash[40] (figure 2.20) is a multimedia platform for adding animation and interactivity to webpages. It is currently being developed and distributed by the Adobe Systems company, and is integrated into the webpages through the use of a plugin installed on the client's computer, being presented in a normal browser.

Figure 2.20: Adobe Flash CS4 logo, courtesy of the Adobe Corporation.

Adobe Flash is a platform to manipulate vector and raster graphics, also supporting bidirectional streaming of audio and video. It is normally used to create animations, advertisements and various other flash components which can, for example, integrate video and audio into a webpage, also being used to develop rich internet applications.

As mentioned earlier, this platform includes a dialect of the ECMAScript standard named ActionScript, which was initially designed for controlling basic two-dimensional vector animations made in Adobe Flash, evolving, eventually through the addition of other functionalities, to the development of web based games and web applications integrating streaming media.

As written in the beginning of this chapter, Adobe Flash requires a plugin installed in the client's computer to operate, which is called Adobe Flash Player available free for most of web browsers and operating systems. Some mobile phones and electronic devices also support this platform's format, through the use of Flash Lite, which is also free, but isn't yet supported by every device of this kind. A standalone Flash Player may also be used to visualize Adobe Flash content through an .swf file (supported by all major

operating system's since it's cross-platform) and it may also be incorporated into an .exe, for Windows environments, or .hqx, for MacOS environments, called a Projector, which consists of a self-executing Flash movie containing an integrated Flash Player into the Projector (supported only by the earlier mentioned environments due to its self-executing ability, which hasn't yet been investigated by Adobe under other environments).

In what IDE is concerned, this platform requires the use of the Adobe Flash Professional[41] multimedia authoring application in order to have full access to all the platform's functionalities for development of content for the Adobe Engagement Platform, which consists in the web applications, games and movies mentioned earlier, as well as the mobile phone's and other embedded device's content. Several other Flash authoring tools have been released over the years by third-party developers, but could never integrate the full capabilities of the multimedia authoring application provided by Adobe, since the company hid the files' specifications very closely, only releasing part of them recently with its Open Source Project[42].

Unfortunately, although the Flash Players, both standalone and web browser integrated Flash Player, are free to download and use in any client's computer, the multimedia authoring application that allows the development with the full functionality provided by the platform, is a proprietary development framework, available for both Mac OS X and Microsoft Windows, which is absolutely necessary for the creation of the web applications, games and movies mentioned above with full freedom.

This platform started off as just a drawing application for pen computers running the PenPoint OS named SmartSketch, which aimed to make the creation of computer graphics as easy as drawing on a piece of paper, being later transported to Microsoft Windows and Mac OS. Later in 1996, FutureWave, which had been developing SmartSketch since its beginning, modified their application by adding frame-by-frame animation features, re-releasing it as FutureSplash Animator for both Microsoft Windows and Mac OS.

As the product evolved, other features were also developed for it, such as the authoring tool user interface, the graphics renderer, curve and shape math code, and also the browser plugin. Since the product was bought by Macromedia in 1996, its focus was mainly the designers point-of-view, seeing that its interface has a time-oriented line of action (a frame-by-frame interface has mentioned above) and a wide variety of 2D and 3D vectoring and designing tools.

After the product was sold to Adobe, and since, by the moment it was sold in 2005, its usage was very high, there was an emerging need to take this application to the next level by developing its programming ability, through the evolution of the ActionScript already available in the previous versions of Flash. So, in 2007, the new ActionScript 3.0 was then presented for the client's logic creation and new programming features and a new MXML modeling language were introduced to describe interfaces' behaviours and appearance.

Microsoft SilverLight

Microsoft SilverLight[43] (figure 2.21), much like the above described Adobe Flash, is also a web application framework which integrates multimedia, graphics, animations and

interactivity into a runtime environment. In fact, Microsoft SilverLight was released to compete with Flash since, up until SilverLight's appearance, it didn't have any direct competitor, dominating the web design market.

Figure 2.21: Microsoft Silverlight logo, courtesy of the Microsoft Corporation.

Although many similarities can be found between Microsoft SilverLight and Adobe Flash, and although their main objective is the same, Microsoft still claims that SilverLight is not Microsoft's own version of Flash. Nevertheless, since the products are very much alike, SilverLight may be also be used to create web applications integrating multimedia streaming, develop web games and build interactive interfaces and animations, like its competitor Flash.

Like the previous product, this framework also lacks standard support for its developed applications, requiring a web browser plugin to operate the animations or interfaces. Unlike other Microsoft products, this actually doesn't require a Microsoft Windows environment to operate and can be installed in a wide variety of operating systems, such as Microsoft Windows and Mac OS X (web browser plugin developed by Microsoft), and also many open source platforms (through the use of Moonlight[44], a web browser plugin developed by Novell Netware in cooperation with Microsoft). Added support for mobile devices operating Windows Mobile 6 and Symbian OS is expected in 2010, but is still, at the moment, unavailable.

Unlike the Flash product, Microsoft didn't release a standalone version of the web browser plugin nor does SilverLight support a single file publication like Projector (Flash file format), restricting SilverLight's use for web browsing only, since its format is always subdivided into several types of files. Despite of this fact, SilverLight can still be used outside the web browser concept, as it may also be used to develop Windows Sidebar gadgets for Microsoft Windows Vista.

As far as IDE goes, SilverLight's development follows the rules of most of Microsoft's products, requiring a .NET framework compatible authoring application, allowing, as Microsoft has accustomed its developers, to develop code for SilverLight in any .NET supported language, provided that they can target the SilverLight CoreCLR for hosting the application, instead of the usual .NET Framework CLR. For this objective, Microsoft has released two authoring applications, the well known Visual Studio[45] (now in its version 2008) and the recently developed Expression Blend Studio[46] (now in its early version 2

SP1).

With the release of SilverLight version 2, the also very well known IDE Eclipse[34], an open-source application, was added as an alternative open-source development tool for developing SilverLight web applications. The emerging development tool Mono, which is known for its versatile development languages and frameworks support, is also currently developing and evolving support for the above mentioned Moonlight, which will eventually support the development of SilverLight applications as well.

As was partly mentioned before, in SilverLight applications user interfaces are programmed using a subset of the .NET Framework languages and declared in XAML. The XAML was recently introduced by Microsoft in its .NET Framework 3.0 and is used in SilverLight to markup the vector graphics and animations, allowing textual content to be searched and indexed by search bots, since it is not compiled, being represented as text in the XAML format.

2.2 Content Management Systems (CMS)

Content Management Systems are basically computer applications used to aid in the management of companies' workflows, important for the creation, edition, publication and archiving of several types of digital media and text with no or little knowledge of what's behind it. CMS's are commonly compared to Document Management Systems (DMS's), although, in reality, despite the common basic objective, the CMS's are a bit more powerful.

These systems' greatest characteristic is its large amount of available tools and functions integrated in the applications and used to store, control and publish various company specific documents, which may include manuals, guides and other important information in the form of image, audio, video, electronic documents or web content.

2.2.1 Types

There are three main types of CMS's, Enterprise CMS's, Web CMS's and Component CMS's.

The Enterprise CMS's are specially indicated for managing content and documents related to organizational processes, including several specific applications that allow the management of an enterprise level organization's information.

The Component CMS's are used to manage content at a granular level, as opposed to the Enterprise CMS that operates at document level. They are organized in components and each one represents a single topic, concept or asset, such as an image, table or product description, and are later assembled and can be viewed as components or traditional documents.

The Web CMS's are content management systems normally implemented as web applications used to create, manage and publish HTML content in a fast, easy and automated manner with no programming skills necessary, including many tools to simplify this process and to add functionalities to an already existing web portal. This type of content

management system will be discussed again ahead.

2.2.2 Web Content Management Systems

As mentioned before, this type of CMS enables the creation of web pages in several programming languages with little or no knowledge of them at all. The majority of these system's uses data bases to store content, metadata or other required objects. XML is used very frequently to facilitate the reuse of the data and allow flexibility.

The management is normally carried out through web interfaces on a browser, but it may, at times, be necessary, in some systems, the usage of a *fat client*. A *fat client* is basically a computer client in a client-server architecture, which provides all the essential functionality, independently of the central server. It simply requires a periodical connection (non constant) with the server, allowing it to carry out all the necessary actions without that connection.

The presentation layer of the Web CMS's content is generated through a set of pre-defined templates. Unlike the other website creation applications (Microsoft FrontPage, Adobe Dreamweaver, ZendStudio, PHPEdit, etc.), the Web CMSs can be used by any person without the need of any special training or previous technical knowledge, requiring some technical experience configuring and adding new functionalities to the system, but still being mainly a management tool for non technical users.

A WCMS lays at the users disposal a wide set of automated templates to facilitate their application both on new content to be generated and already existent content, making the transition automatically. It also allows simple and fast content editing, module addition to extend the systems' functionalities, automatic upgrades, workflows management with ease, the creation and management of user groups with different access levels, document management and content visualization.

Types of WCMS's

These WCMS's can be sorted into three main types, the *Offline processing systems*, the *Online processing systems* and the *Hybrid systems*.

Offline processing systems are, as the name implies, WCMS's that pre-process all the content before its publication (such as *Sagar Vignette CMS* and *Brigolage*). Since these systems don't require a server to process actions, apply templates or generate content, they are often used simply as *design-time tools* like *Adobe Contribute*.

The Online processing systems are CMS's which process actions and apply templates *on demand*, generating the content when a user visits a webpage or when a webpage is retrieved from a cache memory such as:

- Hosted CMS's that belong to companies and are only available on company servers, like *Aspire CMS*, *Bravenet*, *UcosZ*, *Freewebs*;

- Open Source CMS's are free to use and can be installed in any server, like *Mambo*, *Joomla*, *Drupal*, *TYPO3*, *Zikula*, *Plone*;
- Web Application Frameworks that can be Hosted or Open Source, but differ from normal CMS since they follow strict online models and don't allow workflows, like *Wikis*, *MediaWiki*, *Twiki*.

The Hybrid systems are CMS's which combine both *Offline processing systems* characteristics and *Online processing systems* characteristics (like *Blosxom*).

A small description of the some of the above mentioned CMS is presented in the next section, along with some other popular CMS.

2.2.3 Existent Web Content Management Systems Examples and Descriptions

There are several Content Management Systems, proprietary and open source, that can be used for many purposes without any programming knowledge or knowledge of how it works, but they are often more specific to serve a certain objective or, in some cases, include the ability to add new extensions and resources to support a specific purpose, allowing them to be more universal.

The following list presents a few of them along with their categories and respective description[47][48]:

- Open Source WCMS's:
  - **Joomla** is based on PHP language and fits in the Portal category which consists in a common website that may be extended with a FAQ manager or a forum, like mentioned above;
  - **Mambo** is a very popular CMS based on the PHP language too and that also fits in the Portal category as Joomla;
  - **B2Evolution** is also based on PHP language and is in the Blog category which consists in a basic online diary;
  - **TikiWiki** is a PHP based Blog-Wiki or, more commonly, Bliki and consists of a basic Blog with user contributing abilities;
  - **Drigg** is PHP based as well and belongs to the category Digg-like which allows the contribution and marking of news;
  - **PHPMyFAQ** is, as the name suggests, built over PHP and fits into the FAQ category which consists in a manager for questions from users and answers from webmasters;
  - **Current CMS** is a Java based Groupware application which consists, basically, in a collaborative work website;

  - **Xiawe** is BBCodes based and fits in the Templates category, consists in a static CMS where pages are defined by codes;
  - **PHPMotion** has PHP as root language and is in the Media Sharing category and is used to display and manage media content such as video and audio;
  - **Coppermine** also has PHP as root and belongs to the Image Gallery category and is used to, as the name implies, create and manage image galleries;
  - **OpenACS** is TCL based over AOLServer and fits into the Web Application category and is a toolkit for building community-oriented web applications, functions like a sort of an extended CMS;
  - **KWiki** is a multi-language based Wiki CMS, which is formed mainly through user contributions;
  - **Slash** is a Perl based News CMS and is used to display and manage news' archives;
  - **SMF** is, as most of the above, PHP based and fits into the Forum category, consisting in a community based discussion website;
  - **PrestaShop** is also PHP based and belongs to the e-Shop category and is, basically, a toolkit for creating an electronic shop for online shopping;
  - **Graffito** is Java based and is a Universal CMS, meaning it's a toolkit for building any kind of CMS;
  - **Elgg** is a PHP based Social Network CMS, used uniquely to create and manage social environment networks;
  - **Freeglobes** is also PHP based and fits into the Directory category, since its function is to manage directory listings;
  - **IntraLibre** is PHP based as well and sits in the Intranet category, being used to create and manage workflows and collaborative work within an intranet.

- Proprietary WCMS's:
  - **SurgeBlog** is multi-language based and belongs to the Hub-Blog category, which consists, basically, in a server of blogs used to create and manage several blogs;
  - **ForBrains** is a PHP based Web Portal, used to create and structure a full website, also having the ability to extend its functionality through ForBrains' Company developed modules;
  - **Elixon WCMS** is PHP, AJAX and XUL based, and has the ability to create and manage a full website, dividing the client's and administrator's interface into different technologies, also allowing the addition of supplementary modules;
  - **Microsoft Office SharePoint Server** is a .NET and ASP based Web Portal WCMS, but a bit different from the above since it is integrated with Microsoft's

Office Applications to organize and built webpages, being able to create webpages from simple Microsoft Word, Microsoft Excel or Microsoft PowerPoint documents, also having the ability to maintain the created website easily.

2.3 Online Discussion Methods

With the introduction of the Web 2.0 concept, discussed in the beginning of this chapter, the web changed from an information presentation platform to a rich-application web browser environment platform, allowing the users, not only to retrieve information, but also to interact with the webpages in many ways.

It is in the same Web 2.0 concept that the online discussion methods fit, since through these discussion methods everyone may interact with a webpage and also interact with many other users all around the world, bringing to life the *Network as a Platform* concept to connect people, sharing ideas, discussing topics and displaying all kinds of multimedia all around the world.

There are a lot of online discussing methods presently available, each one with a main objective and different capabilities, according to the goal of the method, but they all share the same common idea to connect people and share information dynamically and with a lot of ease. Several examples of these methods are news boards, web forums, web chat interfaces, message boards, web shoutboxes, social networking platforms and blog-wikis, among others, which will be discussed ahead.

2.3.1 News Boards

A News Board, as the name implies, has the same purpose as a traditional cork bulletin board, with the difference that this news board is actually an online virtual bulletin board scripted into a webpage, where people can browse news set either by any visitor browsing the website or only by the website's administrators as a form of displaying relevant news of a certain subject, controlling news insertions and their couriers.

There are several open-source and proprietary scripts and CMS available on the internet that implement the News Boards with a lot of ease and with no programming skills necessary at all, but a common situation nowadays is to have the News Boards integrated into other platforms, allowing users to have an "all-in-one" dynamic platform with all their needs.

2.3.2 Forum

The original Forum has its beginnings in the Roman Empire and was, at the time, a gathering place with an enormous social significance to the roman civilization ("Forum Magnum" as the citizens called it), being often used as center for diverse activities, such as political discussions and meetings, among others.

The Internet Forum inherited the same basic idea implemented by the Roman Forums in 600 BC, but directing this old idea into the new internet fields. In this way, the basic ideas of an ordinary forum were maintained and the fixed gathering location evolved into a multipoint gathering place available to anyone using the internet, with the Web 2.0 *Network as a Platform* concept.

In this way, an Internet Forum is a web application that manages user-generated content, more specifically, it is an online discussion website, where people can participate in topics or create fresh own topics, explaining and expressing their ideas, exchanging knowledge of a certain subject and cultivating social bonds and interest groups of a discussed topic.

There are several open-source and proprietary forum implementations available for download on the internet and they have evolved a lot from their first sketch, since nowadays they include a high number of features which improve the discussions' quality, introduce a wide variety of posting control and effectively implement roles to regulate the forum.

The most common implementations of internet forums used around the web are vBulletin, PHPBB, SMF and Kunena (formerly known as FireBoard and integrated into the Joomla CMS), which are already implemented by an elevated number of websites and companies, such as the UbuntuForums for Linux/Ubuntu related discussions, MacRumorsForums for Apple related discussions, mozillaZine Forums for Mozilla's Foundation discussions, Gamedev.net for game programming discussions and Overclock.net for overclocking discussions, among others with other specific discussion themes or various discussions themes simultaneously.

2.3.3 Chat

The chat room or chat line has existed for many years and is a form of synchronous (occasionally asynchronous) conferencing, enabling the people using this service to, as the name implies, chat in a virtual room over the internet with people all around the world or even chat in a private room, normally with one single person or in a conference with several people, with a difference from before of being a closed entry room which allows the entry of authorized users only.

This online discussion method has been around since the 1980's through a chat system called Talkers and is still currently best known for its standalone applications used all around the internet by people of a wide variety of ages and backgrounds. The most common tools available for this kind of chatting are instant messengers, such as Windows Live Messenger and Yahoo Messenger, internet relay chats, such as mIRC and XChat, talkers, available in MMORPG's and other virtual worlds, and possibly some MUD's which are multi-user real-time virtual worlds described entirely in text.

More recently these forms of chatting have been moving towards new fields, being integrated into websites as a form of member chatting, enabling the use of the above implementations without the need to install standalone applications in the client's computer, being already available several scripts to support these methods. Although these discussion methods have the basic objective of communication between users all around the world,

they are also currently very used by minors, and if not correctly controlled may facilitate illegal sexual contacts.

2.3.4 Message Boards

Message Boards are a lot like News Boards as they share a wide variety of features, and even considering they both have the main goal of propagating information throughout the community of board browsers, the main interest is a bit different, since the News Boards are systems which contain news and important facts about real incidents and the Message Boards are systems which can contain any kind of messages a visitor or administrator wishes to present.

So, it can be concluded that Message Boards are also a virtual version of the traditional cork bulletin boards, in which, according to the rules set by the Message Boards' administrators, either any visitor or just members with permissions, can publish any kind of messages, from simple job opportunity ads to even new product releases with optional authors contacts for possible discussions, with or without control over the published content, depending on the Message Board's rules.

The latest Message Boards implemented and available on the internet, not only allow the publishing of various messages from users, but also allow the division of the messages according to their subject, enabling the browsing of messages through sections according to the subject of the user's interest.

Although these Message Boards may be mistaken for Internet Forums due to this subject division feature, they are not the same method, since Internet Forums allow a real discussion between members within the same topic, keeping a continuous topic throughout the discussion, and the Message Boards don't allow a real discussion between board browsers, allowing the user to create a reply which contains the previous message and so on, thus creating the illusion of a discussion.

This is the reason why people often confuse the two methods and even use the Message Boards as if they were Internet Forums. Some websites are also implementing Internet Forums and using them instead of Message Boards although their use in the website would be actually the same, since the purpose of those websites is just to publish messages for users instead of allowing real discussions between them.

2.3.5 ShoutBoxes

ShoutBoxes are very simple chat-like systems integrated as features into some websites. They share the basic idea of the chat systems described above, but, since the main goal of these systems is to keep it simple, the number of features implemented on the ShoutBoxes are normally a lot lower than the chat systems' features.

ShoutBoxes basically allow people to quickly leave messages on the website, normally without the need to previously register on the website, acting as simple lists of short messages, possibly containing website information or message authors' information, which will be visible to other people browsing the website.

The websites containing these ShoutBoxes generally implement a refresh method that allows to keep new messages visible, either by an automatic webpage refresh after a certain interval or through dynamic poll of the messages' storage space source. Since the objective of this method is to simply present short immediate messages, the older messages are normally deleted once the total number of messages reaches a certain limit, in order to preserve space on the server.

2.3.6 Social Networking

Social Networking is a system that focuses on building online communities of people sharing the same interests and activities or who are simply inspired to explore the interests and activities of other members of the community.

These online social networks are normally web based systems which provide a wide collection of features that allow the interaction between community members, such as e-mail and instant messaging services, as well as some other methods mentioned above and others non-mentioned, such as game and multimedia sharing.

Basically, these social networks are, in some way, a combination of many online available discussion methods and multimedia applications integrated into one single website, allowing the members of the community represented by the social network to interact in a high number of ways without the need to resort to numerous websites to achieve the same objective. In fact, the social networks have encouraged the elaboration of new forms of communication and information sharing on the internet and, as a result, proprietary encapsulated services have been gaining popularity.

Although these social networks normally aim for a more casual kind of community, like a friends community, they are currently also roaming towards more specific communities, such as government agencies' groups trying to get in touch with the public, business entrepreneurs and small businesses looking to expand their contact bases, dating groups to exchange personal information for dating purposes, educational school board associations as a way of discussing school related topics and exchange schoolwork knowledge, and also medical networks updated by healthcare professionals as a means to manage institutional knowledge.

2.3.7 Blog-Wikis

Blogs are websites, normally maintained by an individual or a small group of individuals with regular entries of commentary, descriptions of events or other multimedia material, being usually displayed in reverse chronological order (from the last to the first), with subject specific news or commentaries, or even as personal online diaries, combining text, multimedia and links to different blogs or websites related to a certain topic, and allowing visitors to leave comments in an interactive form.

Wikis are websites which allow the creation and editing of large number of interconnected webpages with ease, using a simplified markup language or a WYSIWYG text editor within the web browser, being often employed to create collaborative websites, to

empower community websites, to enable personal note-taking and to improve corporate intranets and knowledge management systems. These Wikis, as many other online discussion methods, may implement several publishing rules over user permissions and added content.

Blog-Wikis are groupware platforms that fuse the collaborative editing features of Wikis with the user friendly publishing characteristics of blogs, enabling the use of one single website that allows users to share their knowledge or even discuss certain presented subjects in a wide variety of forms, such as pure text of multimedia content, also having a lot of CMS and scripts available on the internet, both open-source and proprietary.

2.4 Development Methods

With the increasingly more demanding web users, the progressively higher demand for effective traffic organization for data exchange and the continuously elevated raise of the number of users browsing the internet, the web developers have to face the challenge of developing websites which are able to provide an acceptable quality of service and workaround the evident current web technologies limitations.

In order to overcome this challenge web developers must elaborate methods which allow them to develop websites considering the situations above described, being the most common development methods preferred by the web developers the Three Layer Web Development model for universal user access and the Three-Tier Application Development methodology for a better traffic organization, which will be discussed below.

2.4.1 Three Layer Web Development

It must be a general assumption that all users are different and, as so, all users like and want to browse through the internet in their own specific way, searching for the most convenient method for them to retrieve the information they seek or view the data they look for.

With that thought in mind, the website developers must create webpages susceptible of accommodating the highest number of users possible, regardless of how the users wish to view or retrieve information from the webpages, and regardless of how the users access the developed websites.

A common way to achieve this goal is to develop websites according to the Three Layer Web Development model recommended by the W3C, which is the entity responsible for most regulations that manage the internet. This Three Layer Web Development model is distributed into a Content Layer, which consists of HTML, XHTML and XML scripts, among others; a Presentation Layer, which integrates CSS, XSL and XSLT scripts, among others; and finally a Behaviour Layer, comprising JavaScript/JScript, DOM, Flash and SilverLight scripts, among others. These layers' representation can be seen in figure 2.22 and will be better described ahead.

Figure 2.22: The Three Layer Web Development model's graphical representation, courtesy of Kevin Yank for SitePoint.

Content Layer

The Content Layer is the first layer to be developed and is where the user inserts the webpage's content and structures the webpage according to his needs through the use of a markup language, like HTML, among others.

The web developers must not implement any visual aspect scripts, concentrating on the content and its structure, since this is the layer that every visitor will have access to and will retrieve information from, regardless of the web browsing technique he chooses to use.

Presentation Layer

The Presentation Layer is the second layer to be implemented by the web developers and, assuming the Content Layer is already complete, it is the layer responsible for the presentation's visual information, being normally implemented through CSS scripts.

The CSS scripts allow the web developers to define every visual aspect of the webpage's presentation, from letter type, size, alignment and color to many other properties of the existing objects in the webpage, such as tables' formatting and images' manipulation, among others.

Behaviour Layer

The Behaviour Layer is the third and final layer to be developed and is responsible for all the interactivity and dynamic behaviour between the webpages' objects and the user actions performed in the webpage.

This layer is normally implemented through JavaScript or JScript, which enable simple and fast methods for user to webpage interaction and vice-versa, with very little programming knowledge, such as changing images or text on mouseover events or clicks.

Nowadays, with the evolution of the technology and web developers knowledge, this layer is becoming more frequently implemented through the use of other client-side scripting languages and frameworks, such as AJAX, Adobe Flash and Microsoft SilverLight, which

allow a higher lever of interactivity between the users and the webpage and a improvement on the visual effects.

2.4.2 Three-Tier Application Development

Nowadays, with the continuous development of a higher number of increasingly more elaborate websites, the level of internet traffic is becoming eventually a bigger concern all around the world.

So, in order to improve the quality of the services provided by the websites and, at the same time, reduce the very elevated level of internet traffic, web developers are advised to create their web applications using the Three-Tier Application Development methodology, implementing a client-server architecture where the Presentation Tier, Logic Tier and Data Tier are developed and maintained as independent modules, most of the times even in distinct physical machines.

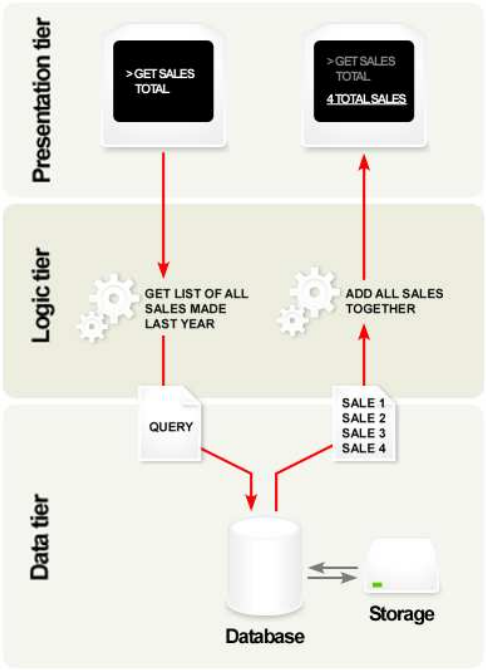


Figure 2.23: The Three-Tier Application Development methodology's graphical representation, courtesy of Bartledan for the Wikipedia Project.

A graphical representation of the Three-Tier Application Development methodology is depicted in figure 2.23 and each tier will be described next.

Presentation Tier

The Presentation Tier consists of a user interface which translates tasks and results into a more user-friendly information presentation structure, in order for the users to understand

the information easily.

This user interface is typically executed in the user's computer through the use of a web browser and a standard graphical interface integrated into the operating system, such as Microsoft Windows, Linux or MacOS X graphical environments.

Logic Tier

The Logic Tier is responsible for the coordination of the application, processing of commands, making evaluation and logical decisions, and calculating operation results, being also responsible for the exchange and processing of data between the Presentation Tier described above and the Data Tier described below.

The processes' functional logic may consist of one or more separated modules which are executed in a server (either physical or virtual) and may also be sub-divided into inner layers within the Logic Tier which is known as an n-Tier architecture.

Data Tier

The Data Tier is where all the information relative to the system is stored to and extracted from either a database or the filesystem. The data is transfered to the Logic Tier to be processed and afterwards, if necessary, that processed data is passed to the Presentation Tier where it will be translated to user-friendly information.

The storage and access to data and its associated logic are placed in a database server, which may be a physical or virtual server, being managed by a DataBase Management System, such as the systems described in the DBMS section.

2.5 Summary

This chapter presented a general view of the state of the art in Web developing techniques. The, increasingly more commonly used, CMS applications were described as well as presented the most known tools. The main current technologies in web developing were discussed, namely the Web 2.0 concept, web servers and DBMS's and also client-side and server-side scripting languages and technologies. There is also a brief overview of the currently most common online discussion methods and web development methods.

Chapter 3

Structural Definition

The management portal's structural definition will be presented in this chapter, including the system requirements description, an overview of the solution's architecture and a presentation of the model's definition. This information is crucial to the developer in order for him to understand the needs of the project and also to start implementing the actual application.

The system's actors, as seen by the application, will also be described and their respective allowed actions in the system, as well as some step-by-step task presentations, which also have a very relevant importance in the implementation of the project, since they allow the developer to know who will be the final users of the application, how it should be used by them and what they need from it.

Finally, some model definitions of the application's information management layer will be shown and discussed in the end of this chapter. This section is the most important part of the project's development, as it is the base structure of the application and it's from these models that the developer will build-up the rest of the application.

3.1 System Requirements

In this section are presented all the system's requirements, from the physical requirements to successfully operate with the application to the functional requirements which are the applications' most important actions for the users.

3.1.1 Functional Requirements

The functional requirements are used to describe the main functionalities which are a *must have* in the application's user allowed actions and the respective system's descriptive responses to those functionalities, that is, how the system interacts with the user in order to provide the feature.

Since the main objective of this portal is the diffusion of the robotic events to a community of interested players and sympathizers, it is extremely important that the application

allows the addition of events and also the registration of the players in these events, participating through the internet without the need to relocate themselves to attend the event.

Another important feature would be a way to communicate with the portal's guests and members, and also to discuss among them several topics of their interest, so these discussion features should also be available for anyone to view and every member, moderator and administrator to use.

A way to navigate through the past events and view their respective media (photographs, videos and real-time arena screens) and logs from the simulations is also a relevant section of the portal, so the application's users may view other teams' simulation results and some images from the events environment and participants.

All the above features, although they are must be implemented in the application, there should also be performance and scalability guarantees from the developer's solution, so the system as a minimum acceptable response time, high online availability and also a good system's growth adaptability.

Summarizing, table 3.1 displays the application's main desired functionalities and their respective system's responses which were described above.

3.1.2 Usability Requirements

The usability requirements' description's main purpose is to specify a certain set of rules which the application's developer must follow in the implementation of the project in order to allow the future system users a clean and simple usage of the web application without confusion and also an easy way of interacting with the application without much previous knowledge of it.

The usability requirements for this type of applications are not so different from one application to another, so, the main usability requirements for the system are:

- The developer must use fonts and colors which facilitate information legibility for any kind of user;
- The application's design must have a clear interface to simplify the navigation throughout the system;
- The developer must apply the three-layer-model implementation advised by the W3C for a better interpretation by any users independently of the browsing techniques used, such as RSS readers and screen readers, also providing an acceptable visualization of the application's interface on older browsers which don't support all current web technologies. The application's implementation must, therefore, be divided into three development layers:
  - The structure layer (through the use of HTML/XHTML);
  - The presentation layer (through the use of CSS);
  - The behaviour layer (through the use of JavaScript);

Functionality	***System's Response***
Diffusion of robotic events through a central and unique portal.	*The application will allow administrators to add one or more events in a simple and fast way (also allowing the closure of invalidly added events).*
Allow members to participate remotely in existing CiberMouse events.	*The application will allow the registration of any person for possible participation in the tournaments (creating a new team or simply joining an existent team in order to participate). The team's binary code will be sent through the portal and stored locally in order to be run at the respective competition.*
Diffusion of website and event related news.	*The application will have a news panel, easily updated by any administrator or moderator through the use of a simple news editing tool. The portal will also display a right side panel with the latest news constantly updated.*
Navigation through a media and log gallery and visualization of these media files and logs.	*The application will allow any person (either member, moderator, administrator or even just a guest) to navigate through a gallery of media files and logs from past events, and visualize them online or download them for storage and offline visualization purposes.*
Scoreboard presentation feature.	*The application will have a constantly updated scoreboard according to the results of the finished events, divided by levels and competitions (each level must have its own scoreboard to avoid unfair score positions and each different competition must have its unique scoreboard avoid confusions).*
Participation in online discussion forums.	*The application will have a discussion forum with several threads and categories, allowing the members, moderators and administrators to participate in discussions or open new threads. The system's guests will only be allowed to view the forum and not to participate in any way.*
Chat feature with online system users.	*The application will include an in-site chat feature in order to exchange ideas directly between currently online users, allowing a real-time conversation method of discussion.*
Performance guarantees.	*The application should guarantee that the pages load fast enough (the ideal would be less than thirty seconds) and that the performance of the event's simulations is not affected by the usage of the website and vice versa.*
Scalability guarantees.	*The application's development must guarantee that the final application has a high scalability in order to allow a satisfying system growth without damaging the performance. It must also guarantee that the various parts of the application may be divided into different physical locations if necessary.*

Table 3.1: *The application's main functional requirements.*

- The developer must implement the system based on the three-tier-application development model in order to ensure a minimum quality of the system's functionality even in cases of greater network traffic, without breaking down the server.

3.1.3 Hardware Requirements

This section defines the hardware requirements to develop, maintain and run the application in order to provide a good performance of the system.

For now, since the system is at its beginning and therefore may be considered as a small application project, the required hardware is as follows:

- One powerful computer which acts as a web server that responds to the users' requests, as an application server that runs active events' simulations and as database server that stores and retrieves the application's data, all on one machine;

- One Ethernet 10/100 Mbit capable network board with internet access in order receive and respond to the users' HTTP requests;

- One UPS in order to prevent computer powerdowns when electric failures occur;

- One backup system with its own data storage in order to backup the system's data, preventing data loss.

At a later time, with the application's activity growth due to the increased popularity of the website and its number of accesses, the system requirements may be as follows:

- Three powerful computers, one acting as a web server to respond to the users' requests, another acting as an application server to run the active events' simulations and the other computer acting as a database server to store and retrieve the relevant application's data;

- One Switch/Router capable of at least 3 Ethernet 10/100 Mbit connections to connect the three above described computers and one connection to the internet in order to receive and respond to HTTP requests from the users;

- Three Ethernet 10/100 Mbit capable network boards, one for each computer, in order to connect each computer to the above described Switch/Router, to create a LAN;

- Three UPS, one for each computer, in order to prevent computer powerdowns when electric failures occur;

- One large backup system with its own data storage capable of backing up the three systems' data to prevent data loss.

3.1.4 External Systems Interface Requirements

This section describes the external systems which are required for the implementation and future use of this application, and they normally vary depending on the developers choice or the application's needs. For this application the requirements chosen were:

- The utilization of the Linux OS, since it is an open-source operating system, completely free of charge, with great security measurements and which grants a high level of power to the OS administrator;
- The utilization of the MySQL DBMS, since it is a platform independent open-source DBMS, also free of charge and which has a great power of usage and a great simplicity;
- The utilization of Apache webserver was chosen, also due to it being a platform independent open-source web server and free of charge, also having great functionalities;
- The utilization of the PHP interpreter, because it is also a completely free to use and develop server-side scripting language due to its needs in terms of IDE and some external systems mentioned above, and because it is a very simple language and very easy to learn and develop in;
- The possible utilization of an external authentication system through the UU's (Universal User) used around the University of Aveiro's Campus. This feature is still being evaluated and isn't part of the main development plans.

3.2 Architecture Description

The architecture description is a conceptual design which defines the system's components and provides an idea of how these components interact with each other, helping the developers in the implementation of the overall system and the system planners understand the business needs. Below are presented and further described this system's applicational and installational architecture.

3.2.1 Applicational

In this section the applicational architecture is presented, which describes the interaction between the different components of the system's implementation, helping the developers to understand better the system's distribution and identify faster any integration problems in the system. As can be seen in figure 3.1, this application is divided into three layers, in accordance with the Three-Tier Application Development methodology advised by the W3C, the presentation layer which consists of the application's webpage operating in the user's browser, the logic layer which performs all the actions requested by the user on the server, and the data layer which comprises all the necessary data to present to the user of perform the requested operations.

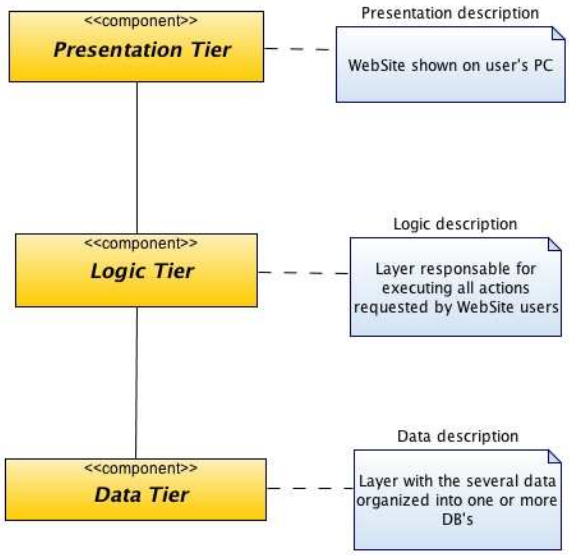


Figure 3.1: The system's applicational architecture diagram.

3.2.2 Installation

This section presents the installation architecture, which describes the physical distribution of the application, defining the several components that constitute the system and the ways they use to interact with each other in order to perform the needed actions. In figure 3.2 are displayed four components, the DBServer which is responsible for all the data storing and gathering, the AppServer which is responsible for the execution of robotic competitions' events, the WebServer which comprises the web application and is responsible for the interaction between the users and the rest of the system, and the User's PC which is responsible for the presentation of the web application to the users through their web browser. In this figure is also defined how the different components interact with each other, the server-side components communicating via a LAN connection (they can separated into different physical machines or gathered all into one) and those components communicating via an internet connection to the user's terminal.

3.3 Actors Description

This section describes the users and their respective roles in the system, including their allowed actions and permitted access locations. These descriptions are used to aid the developer in the implementation of the application as they specify the system's intervenients, their access permissions and their required functionalities.

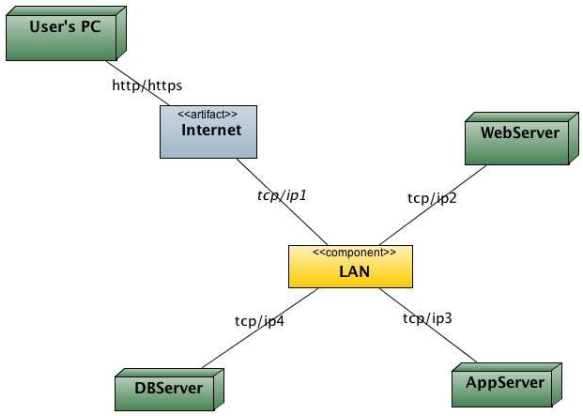


Figure 3.2: The system's installation architecture diagram.

3.3.1 Guest

This actor is not a real member of the application, since it consists of every single person around the world that can access the application through the internet, but it still can visualize the competition's rules and each events' scoreboards, navigate through the media and log gallery, download the competition tools to run single simulations locally and view the system's latest news. This actor can register anytime it wants in order to have access to more functionalities, such as participate in an event.

3.3.2 Member

This actor is the registered user with the lowest access permissions in the system, inheriting all the actions and access permissions from the guest, it is also allowed to register a new team and add other members to it, join one of the existing teams in the system (in case of acceptance by the team's leader), participate in events, submit team code for participation purposes, join or start forum discussions and use the application's chat feature to have real-time conversations with other online actors (except with guests).

3.3.3 Moderator

This actor is not a simple user as the above mentioned, but still doesn't have the highest level of access permissions in the system. It inherits the actions from the guest and member actor classes, but, unlike the members, it can't participate in the events or create teams. It may, however, manage the news board, validate legs and trials, moderate forum discussions when necessary and also validate event entries (when a payment registration is needed to participate in an event).

3.3.4 Administrator

This actor is responsible for all the administrative functions in the application and has the highest access permissions in the system, inheriting all the actions from the inferior class actors. Although this type of actor can't participate in any event or create teams, it is responsible for all the management tools (events management, news management, forum management, users management, teams management, etc.) and it must intervene whenever human supervision is necessary, such as leg and trial validations, scoreboards corrections, among others.

3.4 Actions Description

This section presents the actions that can be performed in the application and the respective actors with sufficient permissions to perform these actions. This description helps the developers to understand the various sections necessary in the system and their division depending on the class of user that accesses the application.

3.4.1 Action Distribution Among Actors

Here the connections among the different users and their respective permitted actions organized into packages are presented, where each package contains all the actions that a user is allowed to perform, helping to understand how the users of the application may be related to each other. Figure 3.3 shows the different packages containing their respective actions and the connection between those packages (the solid line arrows indicate an inheritance relation and the dotted arrows indicate an include relation). Each packages' actions will be presented ahead.

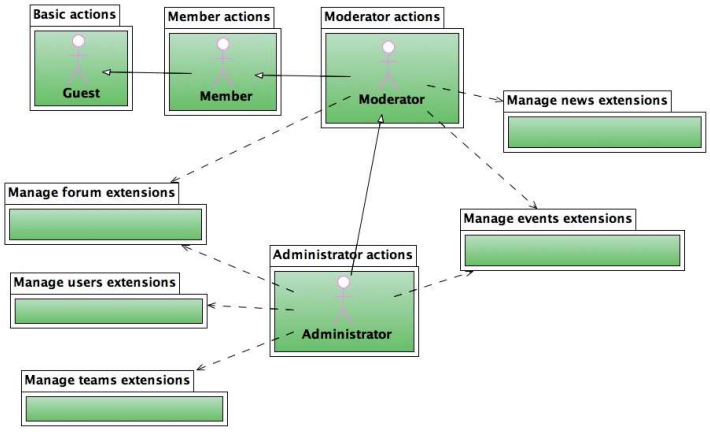


Figure 3.3: The application's actions distributed between actors as a package diagram.

3.4.2 Use Cases

This section, as mentioned above, will present each packages' actions from each user. These diagrams are useful for the developer to identify what to implement and how to do it. It is a powerful aid in understanding both small and large projects, since it provides a full view of the application's functionalities and respective permissions.

Figure 3.4(a) presents the basic actions which can be performed by anyone that visits the web application and which is not registered or logged in. These actions can also be performed by the other registered and consequently logged in users.

Figure 3.4(b) shows all the registered member allowed actions, providing that those users are logged in at the time of navigation, otherwise they will be considered simple guests and won't have access to these functionalities.

In figure 3.4(c) the moderator allowed actions are displayed. Although this actor inherits from member and therefore is able to perform any actions allowed to members, it can't participate in events, having some other minor responsibilities in the system.

Figure 3.5(a) depicts all the administrative actions of the web application. This actor inherits all the allowed actions the other classes own, plus all the administrative responsibilities of the system. This actor has the highest level of access in the system.

A full diagram with the complete actions and their respective actors can be seen in figure 3.5(b), which consists, basically, in an integration of all the above displayed actors' actions with the respective relations between the different system intervenients.

3.4.3 Some Task Descriptions

Since not all of the application's operations are completely straightforward, the activity diagrams help a lot, not only the user but also the developer. These diagrams help the user since with them he can visualize the steps to go through in order to perform a certain operation, and also help the developer since he can use them as a guide to develop that certain operation step by step.

Below some activity diagrams with step-by-step guides of some more complex operations in the system are displayed, namely the operation of viewing the media gallery in figure 3.6(c), the operation of registering a new team in the system in figure 3.6(a) and the operation of entering a team into an active event in figure 3.6(b).

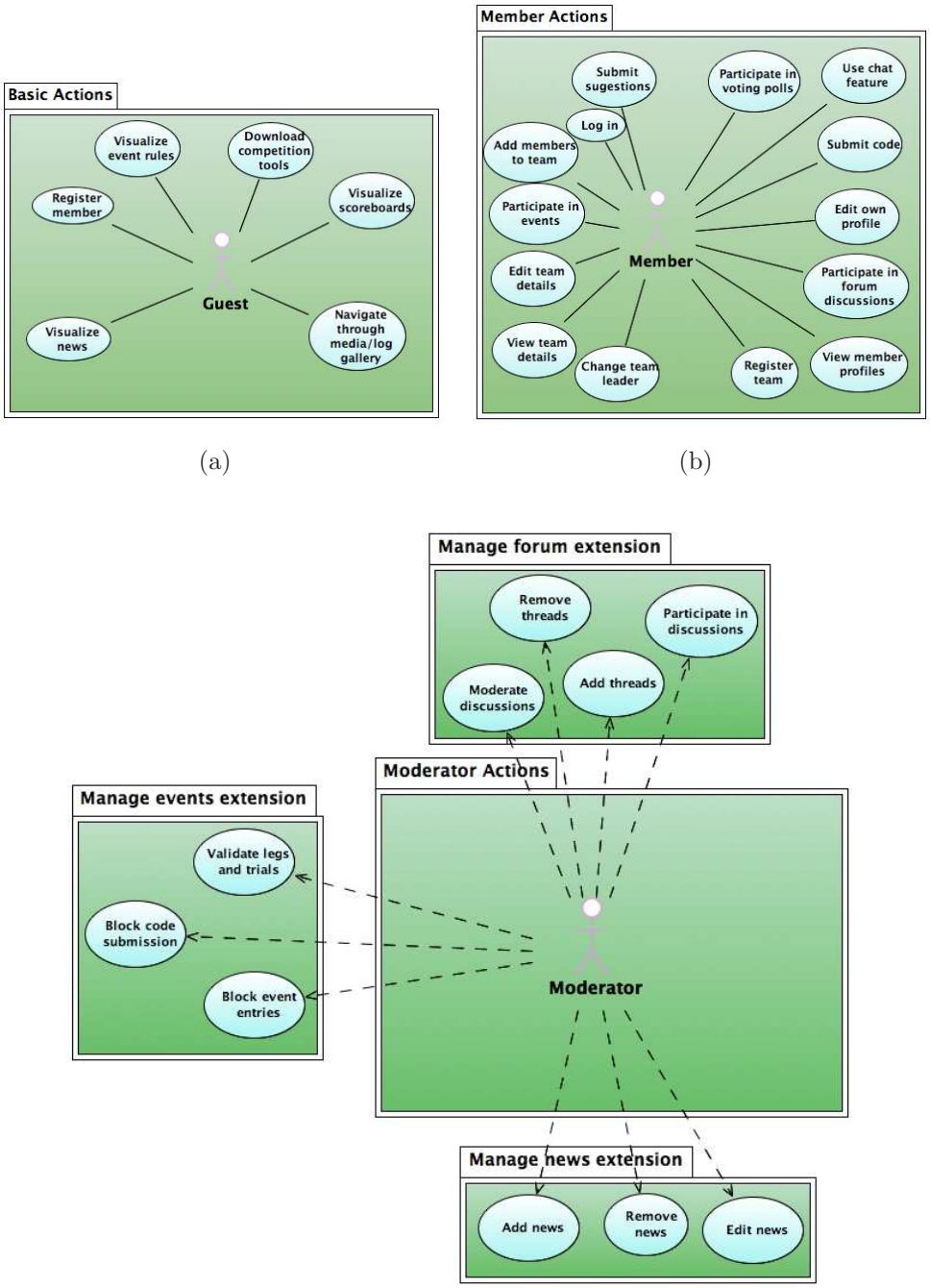
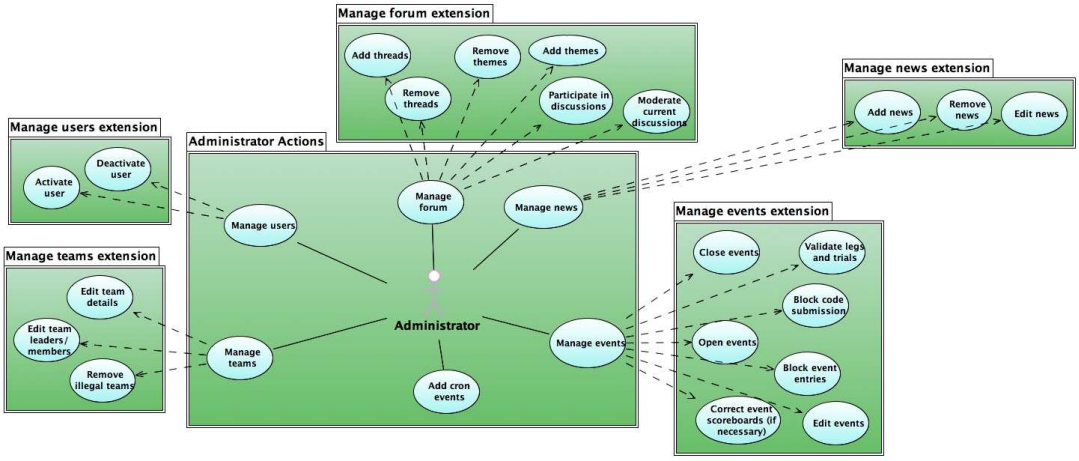
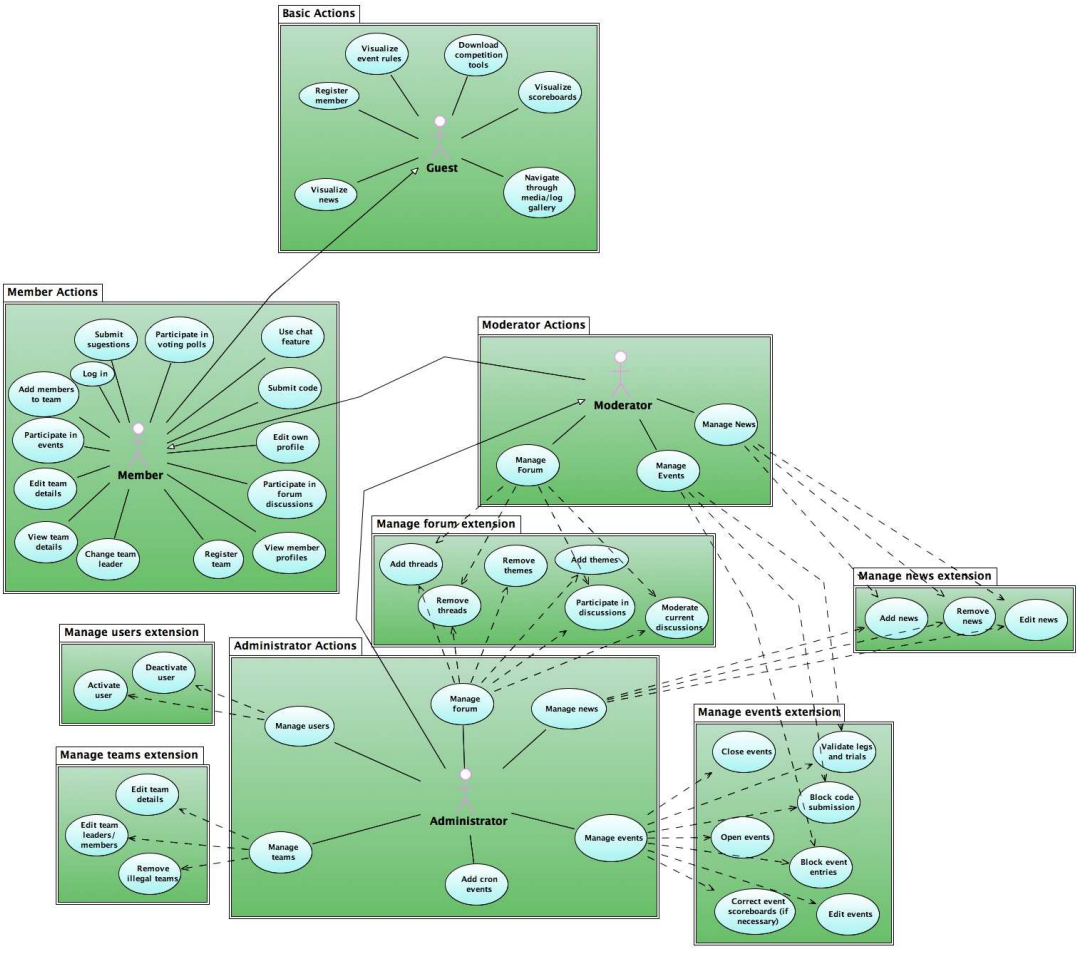


Figure 3.4: Use Case Diagrams divided by actors: figure (a) displays the guests' use cases diagram; figure (b) displays the members' use cases diagram; finally, figure (c) displays the moderators' use cases diagram.

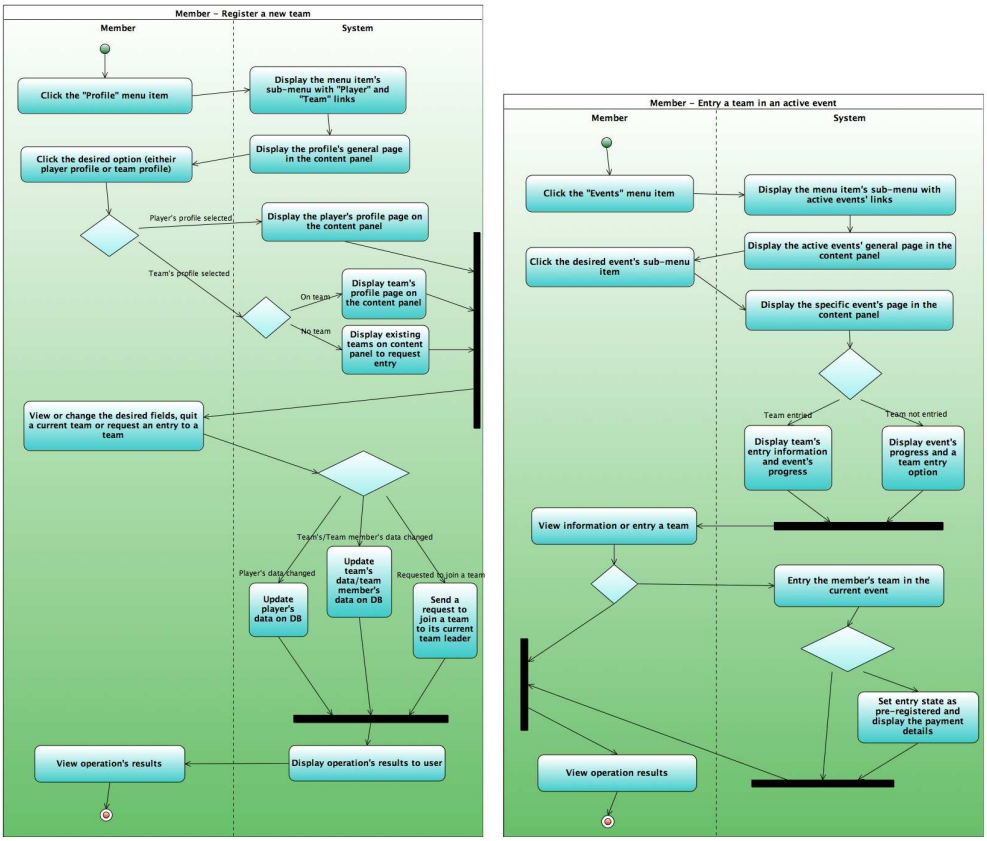


(a)



(b)

Figure 3.5: Use Case Diagrams divided by actors: figure (a) displays the administrators' use cases diagram and figure (b) displays a full view of all the system's use cases and their respective actors as a diagram.



(a) (b)

(c)

Figure 3.6: Some tasks descriptions' diagrams: figure (a) presents an activity diagram of the operation of registering a new team in the system; figure (b) presents an activity diagram of the operation of entering a team into an active event; figure (c) presents an activity diagram of the operation of viewing the media gallery.

3.5 Model Definition

This section presents the model definitions of the system structure, which were used as a basic building block and developing guide throughout the application's implementation procedures. These models' purpose is actually to aid in development of a certain application or even in understanding a certain application, as they describe the system's structure. Below are described the domain and class models for this application.

3.5.1 Domain Model

A domain model is just a plain conceptual model representation of the system which describes the various entities involved in the development and also their respective relationships with each other. This model's purpose is just to give the developers a general idea of the system's components and their connections, also describing each entities main attributes for the implementation step. Below, in figure 3.7, is displayed a diagram of this application's domain model integrating the main entities of the system.

As is depicted, each user may be associated to a team or not, eitheir as a team member or as a team leader; the users with administrative priviledges may add news and events to the system; the teams may enter the active events in the system; the running events will have associated legs and trials according to the teams' distributions when an event is launched; the team's score and each robot score are associated with their own participation in a trial.

3.5.2 Class Model

A class model is a static structure diagram which describes the system's basic structure, presenting the application's data classes, respective attributes and the relationships between them. This model acts as a complement of the above explained model, since the basic purpose is the same, except this model introduces some other important information for the development of the application, making this model crucial during the system's implementation step. Figure 3.8 shows a diagram of the system's class model which presents the data structures as they are known to the application.

Here are depicted the extra enumeration classes that represent the users' class and state, the events' level and type, and the event entries' state. Some more information about the linkage between the classes is also represented in each entity, such as the primary keys.

3.6 Summary

In this chapter the robotic competitions management portal's full structure was described. It presents the application's functional, usability, hardware and external system's requirements, consisting of the basic system's requirements. The architecture models for

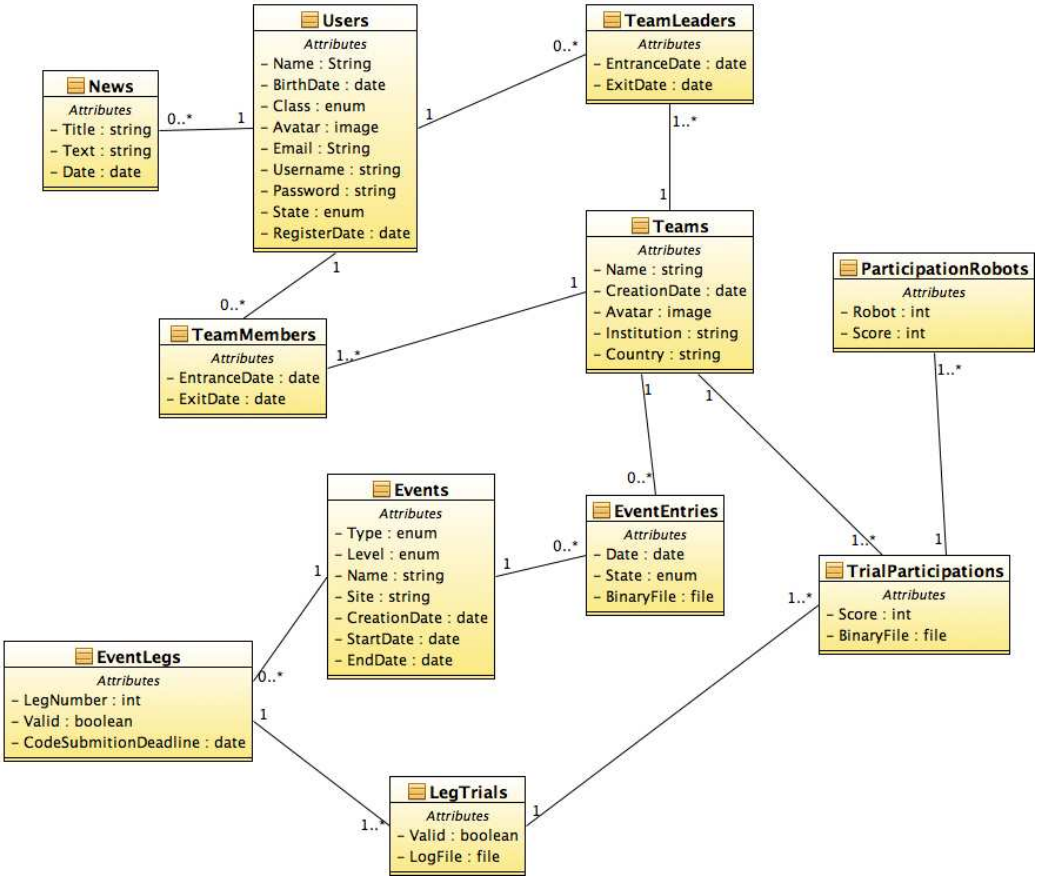


Figure 3.7: The system's domain model diagram.

the project were also explained as well as a brief description of the type of actors which will interact with the application.

After this introduction of the actors, their respective possible actions with the application were specified and some of the more complex actions were explained step-by-step. Finally the basic data structure models were described, that is, the domain and class models for the application.

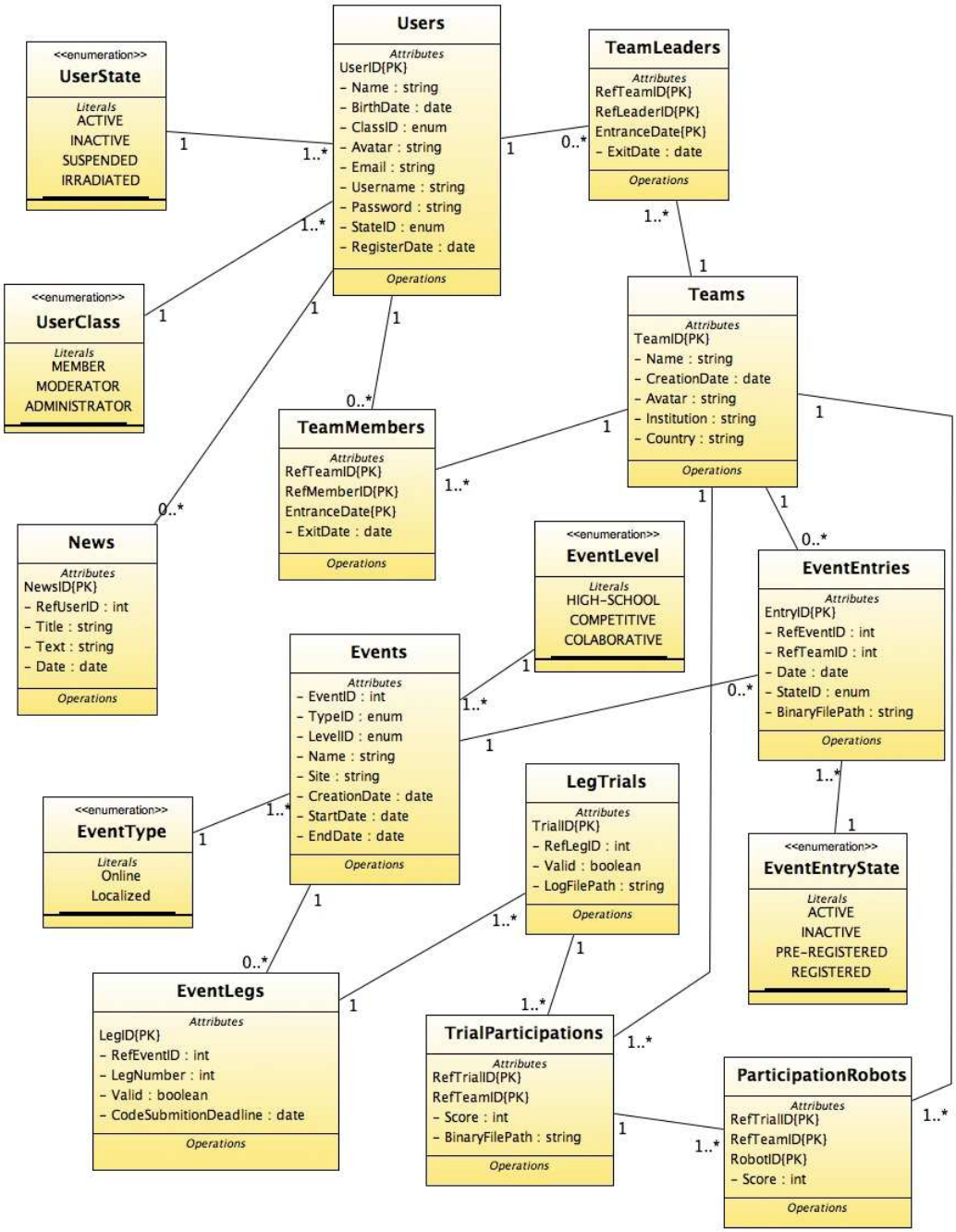


Figure 3.8: The application's class model diagram.

Chapter 4

Development and Implementation

After all the base structure is defined, the next step is to start the business layer and respective interface's development and implementation, so, this chapter will present some issues relative to the development of the application and will also describe an overview of the implementation of the different sections of the application, according to its actor permissions.

This is an extremely important part of an application's development since it the business layer is responsible of performing all the system's actions, and the interface is the only form of interaction between the actors and the system's business implementation, which, combined, allow the actors to perform their functions in the application.

The system's business layer's base structure is already defined in the models presented earlier, but these models' definitions must be followed from the beginning to the end, so that the created code is clean and can easily be modified or updated by the developer, or continued by other developers, and also to ensure a good system's performance and satisfactory behaviour.

In order to create a fully functional interface, it is crucial to keep in mind some implementation aspects and restrictions, such as the availability of visual space for the interface, always give feedback of all the actions to the actor, allow the actor to control the application but always be prepared to prevent error prone situations, and others which will be discussed ahead.

Afterwards, the online discussion methods' interfaces included in the application will be presented and its implementation's processes will be described, along with some simple uses of their allowed actions with various actors. Also included in this chapter is a brief description and walkthrough of some basic actions as they are performed by their respective actors in the web application, and also a small discussion of the implemented security measures and how they aid in protecting the application and its users' data.

4.1 Business Layer Logic

The business layer logic is a crucial part of the development of this project, since its the part of the application that actually performs all the necessary actions in the system. This layer is responsible for the interaction with the interface, which will present the actions' results to the user through the web browser.

In order to provide these functionalities to the actors, the business layer implements some methods that, interacting with the interface to request information to the administrator, perform the mentioned actions in the system and store the handled results in the application's database. Whenever, in this process, the system must return feedback of the actions to the actor, the business layer implements methods that interact with the interface again to display the actions' results to the actor.

This section describes the different parts of the business layer logic implementation and explains some aspects of how it was implemented.

4.1.1 Database Implementation

This part of the project's development is the most important for the application's implementation, since it is the base point of the system. It is responsible for the application's data storage and retrieval, so, as considered by the Web 2.0, it is the heart of this application.

For this part of the system's implementation, the models specified in the previous chapter were crucial and were strictly followed to create the required data entities and their relations in the DataBase Management System.

In order to guarantee these models specifications' competent behaviour in a real life situation, the database implementation was tested against the information provided by past events from different level categories, having a good overall performance and satisfactorily serving the application's needs in concern to the data handling.

4.1.2 Competitions Management

This section of the application includes all the competitions' related actions, from the users' point of view for event participations, to the moderators and administrators point of view for the management of the event related informations.

As is described in the previous chapter in concern to the competitions, the administrator has full power to open, close or edit events' informations, validate leg and trial simulations to ensure that valid resulting data is used in the competition, block the submission of code when the deadline for each delivery is reached and block entries into events to ensure that the teams' can't entry during a running event.

The moderators don't have the same access permissions for this section as the administrators, but they are allowed to validate leg and trial simulations, block the submission of code and block the event entries, as mentioned above for the administrators, but only

if they are not participating in the specific event, otherwise their allowed behaviour is the same as the users.

The users, which act on the other side of the competitions, are allowed to view event details and, in case the user's state is set to "ACTIVE", are allowed to participate in events by entering a team into an event (only teams may compete, so every single user that wishes to participate must register a team even if it only has one member).

When a team wishes to participate in an event, it must pass some conditions in order to be able to entry the event, namely the team is only allowed to entry a high-school level event if it isn't already registered for participation in higher level events, whenever an entry fee is required for an event the team must pay it before the payment deadline, the team can participate if it has delivered the binary code before the delivery deadline otherwise its entry is invalidated, among others.

For the actual events, the business layer provides methods that function as tools for the events' organizers to perform all the actions needed before running the simulations, such as retrieve the participating teams' data and each team's users' data, gather all the submitted binary code for the simulations and distribute the teams' into legs and trials, and provides methods to display the simulations' results to the application's users, such as the scoreboard tools and the media and log gallery discussed below, storing the resulting data in the system.

4.1.3 Users and Teams Management

The business layer is also responsible for providing the user management and team management tools to the users and administrators of the application.

The users are allowed to view and edit their own profile details, to register new teams if they are not part of a team already and having a limit of one team per user (when a user register a team he is, by default, the team's leader), change the current team leader (only the team leader is allowed to perform this alteration, or an administrator in exceptional cases with reasonable justifications), to view and edit their own team's profile details, to add new members to the team or accept current offers and view other users' profiles.

In this context, the moderators don't have any power of management except for the same access permissions as the users, and the administrators are also allowed to perform the same actions as the users, being restricted in actions involving teams and event participations (the administrators are not allowed to register a team nor are allowed to participate in events).

However, the administrators have the power to manage the users, in order to deactivate and later activate them again if needed, suspend or irradiate users that don't follow the organization's rules and edit some user details if strictly necessary. For the management of the users' teams, the administrators are allowed to change team leader's in exceptional cases, as mentioned above, remove team members from teams when they don't follow the rules, edit team details if absolutely necessary and remove illegally registered teams from the system.

4.1.4 Scoreboards Generation

This section of the application, in concern to the business logic only the administrators are allowed to perform actions, since this is a delicate part of the events where little mistake should happen.

So, the business layer interacts with the interface to request the log files from the simulation to the administrators, either all together in a compressed format or one by one as regular log files, and afterwards parses each log file to retrieve the relevant informations from the simulation, storing the results in the database.

When this process is completed, the users, moderators and administrators are allowed to visualize the scoreboards by accessing the specific event and entering the scores section. In order to minimize the possible flaws that could emerge with a manual introduction of the information, the whole process is completely autonomous, only requesting the log files, however, if erroneous situations occur, the business layer provides a tool to correct the scoreboards (only by administrators).

4.1.5 Media and Log Gallery Generation

Such as the above described section for scoreboards generation, this gallery generation is also restricted to use by the administrators, whenever an event terminates or during a running event, in case the administrator wishes to display event related media.

This tool is also completely autonomous, since the business layer interacts with the interface only to request the media and/or log files to the administrators, either in a compressed format or one by one, building the gallery automatically according to the files sent to the application.

After this process ends, the users, moderators and administrators may navigate through the gallery, being able to view and download the media for storing and offline visualization purposes, and the log files for storing and simulation reproduction purposes. Once again, another tool is also provided that allows only the administrators to change some aspects of the gallery, such as certain media and/or log removal, or a correction of the galleries' items in case of generation errors.

4.1.6 Online Discussion Tools' Integration

The online discussion tools, as already mentioned in previous chapters, are a crucial part of the application in order to allow the communication between the users of the website, so, the initial requirement was a news board to diffuse news, a forum to allow the discussion of topics related to events or others and a chat tool which allowed the real-time communication between online users.

The news board is implemented from scratch, and therefore, is completely integrated with the application in all aspects. The moderators and administrators have full management power over the news board and, thus, are allowed to add news items, remove news

items and edit the parts of the existing news items through the use of a tool provided by the business layer.

For the forum, since the complexity of this tool is a lot greater than the news board and the security and permission issues that apply to it are a bit different from the rest of the application, I decided to use a forum CMS which already implements the forum related features. After a bit of investigation on the forum CMS area, I found the phpBB3 CMS to be very simple to use, highly configurable and having great integration possibilities.

However, this integration is only partly implemented yet, since some problems have arose during the implementation of the business layer's tools which allow the full integration with the forum. However, the forum is now integrated with the application, in concern to the users' and teams' data, only remaining unlinked for the login feature, which is not yet synchronous with the application's login feature.

In concern to the chat tool, although some research has been done in this area and some implementation attempts have been tried, a complete release of a chat tool has not yet been completed successfully, so, for now, the integration of a chat feature with the application is still unavailable for the moment.

4.2 Interface's Development

Before and during the development of the actor interfaces and respective business layer actions, some key points have arose, which dictated the limitations and possible accomplishments of both the server-side and client-side of the application.

These points are not only relevant in the presentation layer of the interface, but also in the underlying processes in the client-side's implementation, since it is a very limited environment. On the other hand, on the application's server-side business layer, there are also some limitations to what can be done for the actors and how certain actions are allowed to be performed by the OS behind the application.

4.2.1 Restrictions

Since a part of the web application is intended to run on the client-side, some restrictions to the development of the system must be imposed in order to maintain good level of user's satisfaction with the overall application and a good level of performance even during the highest network traffic times.

First of all, the developer must have in mind that not all users are the same, that is, each user has different ways to access the website, browse it and visualize it on the screen, so, it is extremely important to define some parameters before starting the implementation of the application in order to try to meet most of the users expectations, needs and demands.

As mentioned before in previous sections, the users methods of browsing the web and navigating through a website are not always the same and, sometimes, not even identical, so it is important to conceive an implementation plan divided into layers with content, presentation formating and behavioural definitions, as described earlier in the Three Layer

Date	*Higher*	*1024x768*	*800x600*	*640x480*	*Unknown*
January 2009	*57%*	*36%*	*4%*	*0%*	*3%*
January 2008	*38%*	*48%*	*8%*	*0%*	*6%*
January 2007	*26%*	*54%*	*14%*	*0%*	*6%*
January 2006	*17%*	*57%*	*20%*	*0%*	*6%*
January 2005	*12%*	*53%*	*30%*	*0%*	*5%*
January 2004	*10%*	*47%*	*37%*	*1%*	*5%*
January 2003	*6%*	*40%*	*47%*	*2%*	*5%*
January 2002	*6%*	*34%*	*52%*	*3%*	*5%*
January 2001	*5%*	*29%*	*55%*	*6%*	*5%*
January 2000	*4%*	*25%*	*56%*	*11%*	*4%*

Table 4.1: *W3Schools web survey[55] results concerning the display resolutions used in 2000-2009.*

Web Development method, allowing any user with any browsing technique to navigate through the website without experiencing a distortion of the interface.

Another very relevant parameter is the screen configurations, since not all users have the same computers and use the same screen resolution and screen color and depth definitions. So it is crucial to define a minimum screen resolution, color and depth which the client's computer should have to visualize the interface without having any size or color abnormalities. In order to establish these values, the developer must take into account the most used screen definitions among the web users as a way to meet the highest number of satisfied users, remembering that a solution to everyone is still quite impossible.

In table 4.1 are displayed a W3Schools web survey results, from the year 2000 up to the year 2009[55], which present the percentage of inquired web users that use the specified display resolution on their computers, showing that currently most of the web users have a resolution of 1024x768 or even higher, meaning that the developer should define a minimum screen resolution of 1024x768 for the visualization of the interface and design it to fit the defined resolution.

As for color depth, table 4.2 displays a W3Schools web survey, taken from year 2000 up to year 2009[55], which presents the percentage of inquired web users that have a color depth definition capable of displaying the specified number of colors on their computer screens, showing that most web users have a color depth of at least 24 or 32 bits, capable of displaying perfectly 16,777,216 colors on the screen, giving the developer a vast range of colors which can be used on the interface.

One last restriction when implementing web client-side interfaces is the amount of processing power which is available and the web browser's accepted behaviour language, not only due to the fact that each web user has different computers with different processing powers and different web browsers, but also the fact that the web browsers architecture is quite limited in concern to the computer's resources usage and its interpreted behaviour language, that, although it must be normalized, in some points differ from its standard definition.

Date	*16,777,216*	*65,536*	*256*
January 2009	*95%*	*4%*	*1%*
January 2008	*90%*	*8%*	*2%*
January 2007	*86%*	*11%*	*2%*
January 2006	*81%*	*16%*	*3%*
January 2005	*72%*	*25%*	*3%*
January 2004	*65%*	*31%*	*4%*
January 2003	*51%*	*44%*	*5%*
January 2002	*43%*	*50%*	*7%*
January 2001	*37%*	*55%*	*8%*
January 2000	*34%*	*54%*	*12%*

Table 4.2: *W3Schools web survey results concerning the color depth used in 2000-2009[55].*

Year 2009	*IE8*	*IE7*	*IE6*	*Firefox*	*Chrome*	*Safari*	*Opera*
October	*12.8%*	*14.1%*	*10.6%*	*47.5%*	*8.0%*	*3.8%*	*2.3%*
September	*12.2%*	*15.3%*	*12.1%*	*46.6%*	*7.1%*	*3.6%*	*2.2%*
August	*10.6%*	*15.1%*	*13.6%*	*47.4%*	*7.0%*	*3.3%*	*2.1%*
July	*9.1%*	*15.9%*	*14.4%*	*47.9%*	*6.5%*	*3.3%*	*2.1%*
June	*7.1%*	*18.7%*	*14.9%*	*47.3%*	*6.0%*	*3.1%*	*2.1%*
May	*5.2%*	*21.3%*	*14.5%*	*47.7%*	*5.5%*	*3.0%*	*2.2%*
April	*3.5%*	*23.2%*	*15.4%*	*47.1%*	*4.9%*	*3.0%*	*2.2%*
March	*1.4%*	*24.9%*	*17.0%*	*46.5%*	*4.2%*	*3.1%*	*2.3%*
February	*0.8%*	*25.4%*	*17.4%*	*46.4%*	*4.0%*	*3.0%*	*2.2%*
January	*0.6%*	*25.7%*	*18.5%*	*45.5%*	*3.9%*	*3.0%*	*2.3%*

Table 4.3: *W3Schools web survey results concerning the web browsers used in year 2009[56].*

In regard to the client-side processing power available, the developer must always have that in mind and realize that every web browser limits its resource usages and that no user would want to wait endlessly for an action to occur if the process takes too long, so the developer must try to build simple and light behaviours for the interface to lower its processing needs. Involving the definition of the behaviour language's dialect, as can be seen in table 4.3 which presents the results of a W3Schools web survey results of the web browser's usage in the year 2009[56], the most used web browser is Firefox, allowing the developer to implement the interface's behaviours in the normalized JavaScript standard dialect, since Firefox is fully compatible with it and follows its language rules.

4.2.2 Workspace Description

The application's workspace has a great importance since it is the users only way of interacting with the underlying business processes and performing actions in the application, so it is crucial not only that the user finds it visually satisfying, but also that it has a good performance and meets the users' needs.

In order to accomplish these goals, the workspace's design must be clean and simple, but also allowing the user to perform all the major needed actions fast and easily, summarizing, everything must be in near reach without crowding too much the visual interface, so the users don't feel smothered by the application.

Figure 4.1 presents the proposed design for the interface's workspace, dividing the interface into 5 essential blocks, the *header*, the *menu*, the *content panel*, an *extra panel* and the *footer*. The header will be used simply to display to the user the application's banner and the current system date; the menu is the main website's navigation form and is used to navigate through the various sections of the website; the content panel is used to display the requested information by the user and, occasionally, an options menu relative to the requested information when needed; the extra panel will be used to display relevant events or website information, and possibly include a poll on a certain subject; and finally, the footer will be used simply to display copyright information, authors' information or even to include a small website map-like menu to navigate through the most important sections.

Since the adopted development method was a modular development, this workspace can be easily changed later or even have other divisions or components inserted with only a few or no changes at all to the workspace definitions.

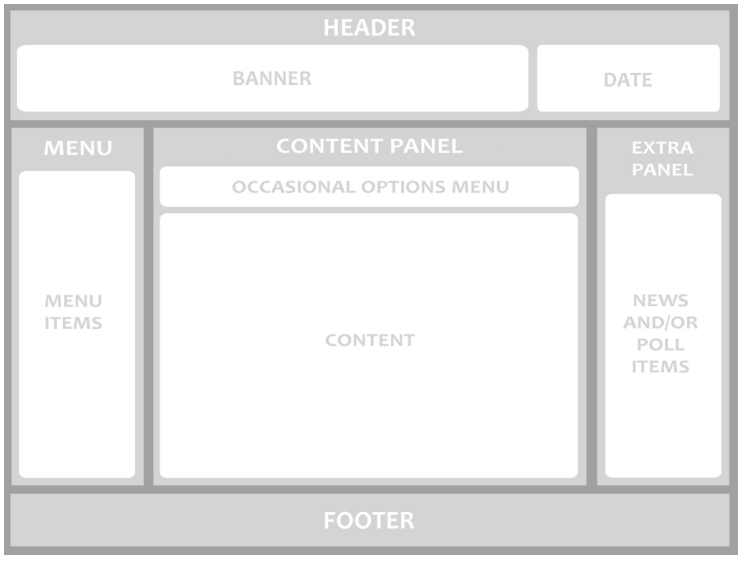


Figure 4.1: The application's workspace.

4.2.3 Some Relevant Aspects

With all applications, big or small, there are always some points in their development which can pass us by, if they are not taken into account while designing the application. So, ahead will be presented some aspects which, as small or insignificant as they may seem to be, have some importance on the users experience of the application and also on the system's protection for unwanted actions.

DBMS abstraction layer for the communication between the business logic layer and the data layer, so the developer doesn't depend on the used DBMS to implement the business logic, also allowing the change of the used DBMS without having to change the rest of the application's implementation;

Submitted data to the underlying business logic is always concealed from nearby snoopers in order to protect the application's users' information;

Action verification is always performed twice during a submission process, once in the client's side through JavaScript and once on the server's side through PHP, as a way to protect the system against unwanted or misused accesses, and also to remind the user in case he forgot to input some requested data;

Data validation on the client's side to diminish the time spent by the user waiting for a result in case of a validation failure, and on the server's side in case the client-side scripting is disabled, so the invalid data won't pass through the business logic;

The news panel displays the last news (the number of news is configurable) and enables the user to simply click the desired news item in order to read the full news item, being automatically redirected to the news section on the selected news item;

Concealable menu items according to the section being viewed by the user as a way to simplify the user's menu;

Remember me feature that enables the system to identify a user on his next visit to the application without the need to perform the login again;

News listing with several options for listing the latest news by days, by number of simply to list all the current system's news;

Online discussion methods are embedded into the application, so the user doesn't need to navigate away from the application or have several different windows for discussion purposes.

4.3 Interface's Implementation Overview

After all the above described considerations were defined and taken into account, it is time to start implementing the actual interface, so the goal of this section is to provide an overview of the interface and explain how it was conceived theoretically and how that theory was accomplished.

First of all, the starting base point of the implementation is the base structure of the application. With the workspace already defined it becomes easier to draw a sketch of the interface, since the components disposition is already set. Figure 4.2 shows the base application's interface, which corresponds to the website's homepage that is displayed to every web user that requests the website.

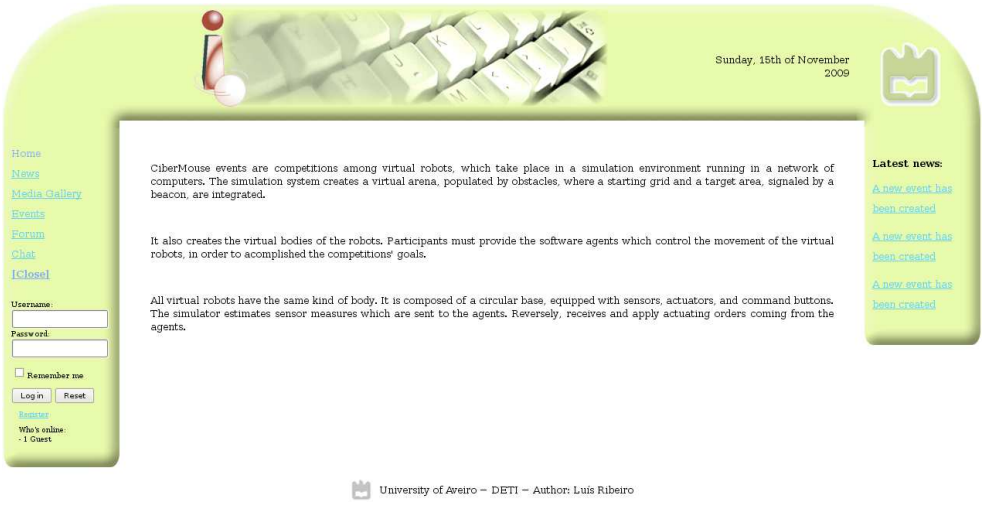


Figure 4.2: An overview of the complete interface implementation.

As described in the workspace definition above, the different components of the application's interface can easily be seen when visiting the website. Starting off with the header, which can be seen in every page of the interface, including figure 4.2, since it is a common component of all the sections of the application.

By examining figure 4.3(a), which displays the application interface's header, the inner component divisions are clear, at the left the application's banner and at the right the system's current date. Looking at figure 4.3(b) we can see the application's navigation menu as it is seen by the guests visiting the website (certain items also unfold when inner sections exist). Figure 4.3(c) displays the application interface's extra panel, which is primarily used to display the latest news to every guest that requests the website.

In the middle section is the application's content panel, which can be seen above in figure 4.2 and where the information requested by the user is displayed and possible options menus in case of a need for an inner section navigation. At the bottom of the application's interface is a small footer, depicted in figure 4.3(d), that describes the organization involved and the people responsible for the application's administration.

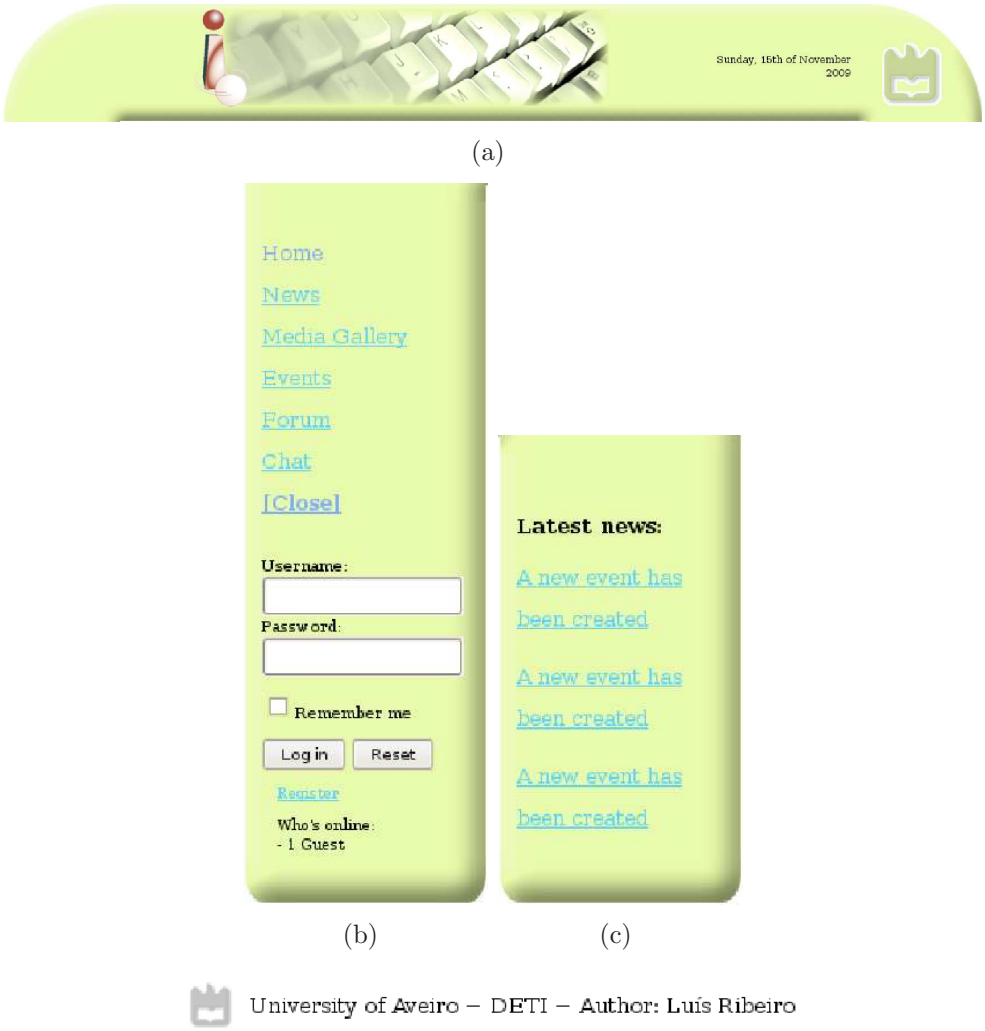


Figure 4.3: A close-up of the application interface's components: figure (a) displays a close-up of the application interface's header; figure (b) displays a close-up of the application interface's menu; figure (c) displays a close-up of the application interface's extra panel; figure (d) displays a close-up of the application interface's footer.

4.3.1 Member Implementation

Since the users of the interface inherit all the actions from the guest actor, the previously described implementation sections also apply to the member actor of the application, but obviously with some differences.

As mentioned in earlier sections, this actor is allowed, not only to view past event's media gallery, but also to view the current events, register a team and access its information, and view or edit his profile. Figure 4.4 displays the menu as it is seen by a member who is navigating the website. As is depicted, the member has access to a new section named

Profile which incorporates the player's data and the team's data, in case the member has an associated team, otherwise a team registration or join page. The member also has access to the inner sections of the *Events* section, which includes the entry in active events and other info.

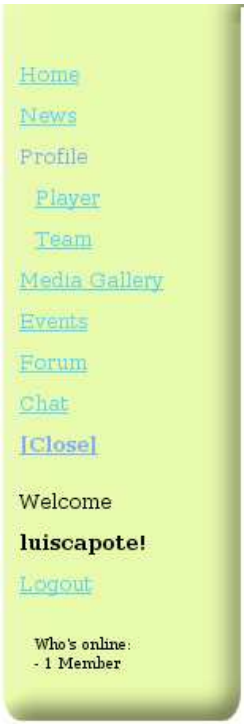


Figure 4.4: A close-up of the application interface's menu as seen by a member.

Another relevant point shown in this figure that has been already mentioned before is the differentiation between the section's main links and its inner links, separated by level (level 1 is considered as the main section and has a lower indentation, level 2 is considered a section's inner link and has a higher indentation), and the unfolding of the inner links whenever the main section's link is clicked (in this case the main section selected is *Profile*).

4.3.2 Moderator Implementation

As mentioned above, since every user inherits from the earlier, the moderator actor would not be different, having access to everything the member has, but also having a new section named *Admin Panel*. This new section is an administration panel with the administrative options the user has access to. In this case, since the actor is a moderator, and, according to what was described before, can only perform some management actions on the event's, news' and forum's data.

Figure 4.5 displays an example of the moderator's administration interface, which shows, in the menu, the administration link, and, in the content panel, the occasional

options menu described in the workspace that lead to the respective administrative subsection when clicked.

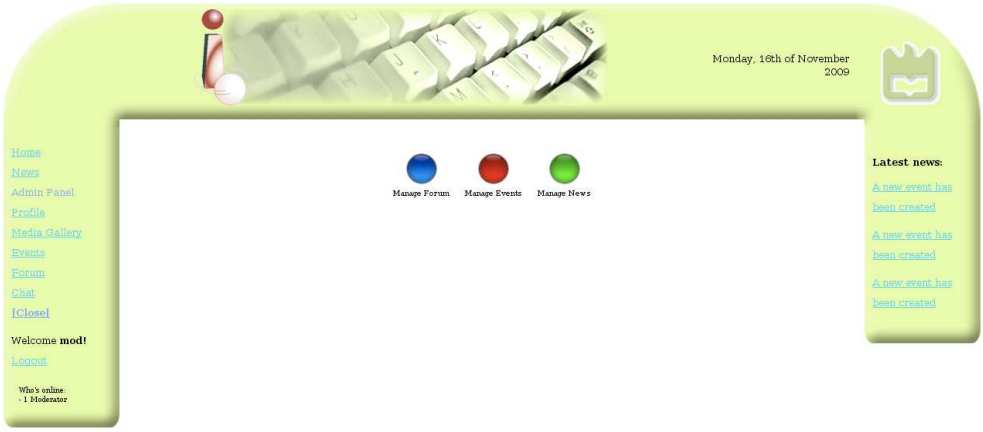


Figure 4.5: An overview of the application interface's administration panel for moderators and its inner options.

4.3.3 Administrator Implementation

The administrator actor, since, as referred previously, has the top level of power on the application, it is granted full access to all sections of the application's interface. The above mentioned *Admin Panel* that can be used by the moderator actor, is also used by the administrator with a few more allowed management actions, due to its high level of access.

So, as the moderator, this actor also has a new section added to its menu which includes the link to the administrative section. Figure 4.6 shows that both moderator and administrator share the same menu items, but it also shows, in the content panel, the occasional options menu which contemplates two more links, one for the management of users and another for the management of teams.

4.4 Online Discussion Implementation

In the previous chapter, in the functional requirement's definition section, some of the online discussion methods were described as being a very important part of the application for the users to communicate.

So, the implemented methods were an internet forum, as depicted in figure 4.7, through which the logged in users could discuss event related topics, help solve each others programming problems or even talk about any other subjects they desire, and also a news panel, as seen in 4.8, through which the administrators and moderators could diffuse event or website related news to all the website's community.

Figure 4.6: An overview of the application interface's administration panel for administrators and its inner options.

The displayed forum's interface, is implemented through the use of a Forum CMS named phpBB version 3, and it was chosen due to its great security features, its simplicity and the already built-in administrative, moderation and discussion functions available according to the users' level. Another very relevant fact that incited the use of this CMS was its easily customizable interface and its ability to be fully integrated into larger systems with few knowledge.

As for the displayed news' interface, it was completely implemented by scratch, although there are several News CMS which could provide the needed functionality. Since the goal was a simple news board for users to view and a simple backend news administration tool, it seemed the CMS solution would be a bit heavier for use on the desired features and, although, it required a bit more knowledge than the CMS, its interface would be much more moldable.

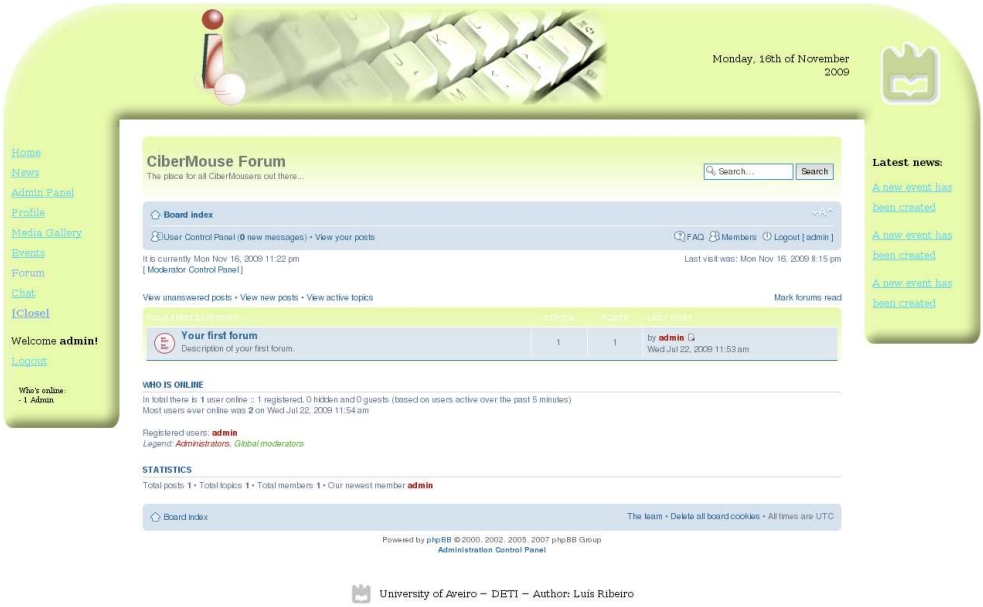


Figure 4.7: An overview of the application interface's forum feature implementation.

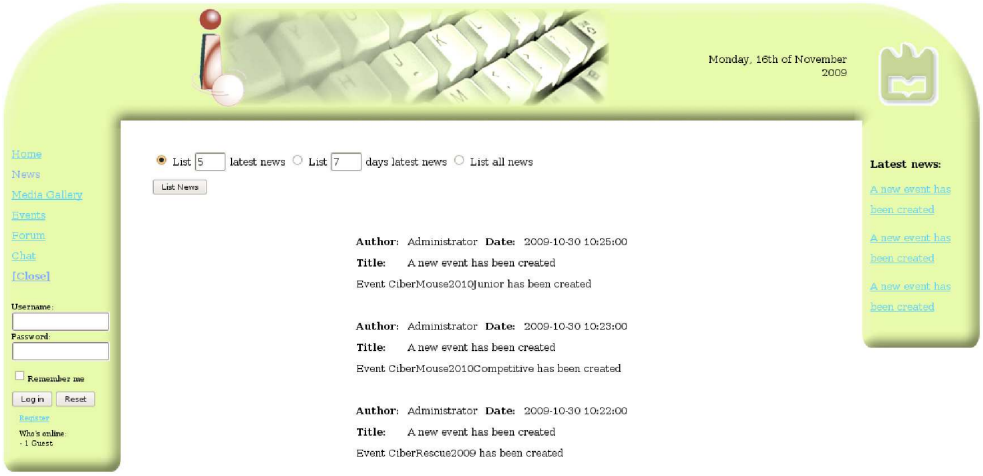


Figure 4.8: An overview of the application interface's news feature implementation.

4.5 Security Measures

In the current days, and with the evolution of the computer technologies, the computers are constantly being more used to access and store personal or confidential data, and the integration of these technologies with the internet was obviously a very important step for achieving the ideal of data sharing and remote usage of information of data structures.

However, this ideal is not as it was thought to be, and could, in fact, provide abusive accesses, such as mal-intended web users who roamed around the internet just looking for other users' crucial informations, but also to provide higher security features, since some web users actually cared about finding weak spots so the involved parties may fix them.

As many security measures concerning the web servers and other computers in general, already include several basic security features, there are still a lot of methods which help the developers increase their website's level of security at the range of the application's maneuver.

Although this application doesn't store any crucial data about the website's users, it is still very important to ensure a certain level of security, giving the web users a bit more safety and confidence when using the application.

So, the following measures were taken into account at the time of development:

- With the use of a login system, the application, not only knows who is navigating through the website, but is also capable of performing user related actions without the need to request identification constantly. As for the client-side, this login system also allows the users to have more confidence with the application's usage by always requesting some credentials in order to access its own data;

- Throughout all the development of the application, the possibility of others to upload content to the web server or execute some code on it was always considered, and this, obviously isn't the planned behaviour for the application's users, so every submition to the web server is always checked twice to ensure that there are no mistakes;

- Although the internet is logically used to connect people, sometimes it is also used by other people as a way to snoop around looking for unsecured transmitted data over the internet. Since this is a major concern on an web application, all the crucial data for the user is securely encrypted for transmission over the internet and even storage on a database;

- Through the implemented class differentiation within the application, another level of security was also achieved, since users must have certain permissions to perform delicate actions or access vital application's information, restricting this responsibility only to trustworthy members.

4.6 Summary

Over this chapter was described the development and implementation of the business layer logic and the portal's interface, presenting some relevant development considerations and definitions, and also a brief overview of the application's real interface, highlighting some particular sections of each actors view of the interface.

Afterwards, a small description of the actual online discussion methods used in the system was presented along with some real representations of each of them. Finally the importance of security measures in current web development was addressed and some simple cases of used security measures in the application were presented.

Chapter 5

Conclusion and Future Work

5.1 Conclusions

This section presents the project's objectives which were satisfactorily met with the development of the application and the personal evolution that I, as the application's developer, experienced throughout the various stages of this project.

5.1.1 Objectives

This thesis' main objective was to develop a centralized management portal which would be able to run simulated robotic competitions, giving support as a supplementary tool for use in localized events to aid the event organizers perform the simulations, and also allowing the organization of online events as a fully autonomous application.

At the beginning of this project's, with the raising of the application's requirements, it became clear that the defined objectives were a bit ambitious, since it was highly noticeable that the desired application would take a considerable amount of time, knowledge and experience in order to be fully implemented with all the needed functionalities.

So, at this point of the application's implementation, the state of development of the application may be considered satisfactory, since the defined main objective was partially accomplished with the implementation of the application for use as a supplementary tool to support the organization of localized robotic events.

As was mentioned in this thesis introduction, the application was expected to play two roles in the competitions, and the above described role, that was satisfactorily achieved, is the role of passive entity, whose purpose is to allow:

- The diffusion of localized robotic events, through the use of the news board to propagate the newly open event, the website's forum to discuss the opening of the event or other event related topics, or just by adding the event in the active events listing, making it available for participation by active teams;

- The registration of players on the web portal, through a simple application's form

which collects the users information, stores it in the system's database and simultaneously registers the player of the website's forum for immediate discussion purposes;

- The creation of teams with registered members as players, again through the use of a simple application's form which gathers the teams information, and allows the invitation of active website members or acceptance of other members' spontaneous candidature;

- The entry of active teams in active events, through a simple enrollment process that retrieves the necessary information for the team's participation in the event, setting the entry state according to the payment of a participation fee when required by the organizers;

- The submission of binary code for a specific event entry of a team, also giving the opportunity of submitting new binary code before the execution of a leg, enabling the remote participation of distant teams in localized events;

- The retrieval of all teams' binary code by the events' organizers for the actual execution of the event's simulations, arranging the binary files according their respective team and event;

- The diffusion of the events simulations' results online through the use of a scoreboard feature which enables the organizers to publish every simulations' results immediately after the simulation ends, being instantly available online;

- The elaboration of a media and log gallery of previously ended events, in order for guests to get familiar with the events' environment, view several media of the event and reproduce simulations locally through the use of the log files.

In concern to the chosen technologies, as is partly defined in chapter 3, the OS used as the application's base system is the LinuxOS, since it is very reliable, it has several different distributions to choose from, it is an open-source implementation, it is completely free to use and it gives the system's administrator a lot of power to operate.

As for the web server, the Apache HTTP Server was chosen due to its multi-platform feature, since it was supposed to be installed on a LinuxOS distribution, and also because it is also highly reliable, very powerful and free to use. The fact that this web server is extremely popular was another reason for this choice.

For the DBMS, MySQL was the first choice because it is also platform-independent, free of charge, has a very intuitive SQL syntax and works extremely well when integrated with the chosen web server. Although other choices also could be applied for this part, this was the best in terms of simplicity and computational power requirements, given the complexity of the application to be developed.

Finally, in concern to the base language for the application, the best choice, given the previous defined technologies, was PHP, since it is very easy to learn, it is completely

interpreted in runtime, it is extremely powerful when combined with the rest of the choices and has absolutely no cost associated with the development using it.

With the choices mentioned above, it is clear that the top priority when analyzing the possible candidates was to select the options which had a multi-platform or platform-independent implementation, and had little cost or absolutely no cost at all in licensing. This main priority was attained, since the defined environment was completely free to use with no charge, which is also called a LAMP environment.

In the implementation itself, as was presented before, a CMS could be used to build the web interface for the application, but that option was declined in favor of an implementation from scratch, since it allowed a higher freedom for development of the interface as it was thought by the developer and a better integration with the underlying business logic. However, for the implementation of the web forum, since it didn't require a deep connection to the underlying business logic relative to the competitions and several Forum CMS capable of providing the needed functionalities were already implemented, I chose to use a Forum CMS, namely, the phpBB3 CMS due to its great features and easy integration with external systems.

5.1.2 Personal Evolution

During the definition of the application's structure and its development, it was extremely necessary to acquire knowledge about the development and publishing of web applications using the chosen technologies described above.

In concern to the programming languages used to develop the application, I was required to obtain and improve my knowledge of HTML, CSS, XML, PHP, JavaScript and SQL/PSM (MySQL's version of SQL).

As for the used development tools and IDE's, vim and gEdit were used in LinuxOS for small alterations, but the major IDE used in the overall implementation of the application was Adobe DreamWeaver CS3 in MacOS, which required the improvement of my knowledge on using it. For the design of the UML diagrams the NetBeans IDE was also used due to its UML features, also requiring some improvement in that specific area.

For the DBMS, besides the SQL/PSM acquired knowledge, as stated above, some knowledge was also required to configure MySQL on the LinuxOS and to use the command-line tool. The web-based GUI phpMyAdmin was also used, but since it is highly intuitive, didn't require much knowledge or skills.

In the publishing of the application, I also developed some knowledge and experience on the configuration of the LinuxOS and the Apache HTTP Server.

Besides the technological knowledge acquired and improved, I also made an extremely important evolution in my programming and project organization skills, as I learned, first hand, the importance of following the structural definitions from bottom to top and the importance of the code organization with large projects, in order to develop an application with clean, reusable and easy to understand code, so I wouldn't get lost in my own implementation and so other developers could easily continue the project's development.

5.2 Future Work

Like all large implementation projects, this project required a lot of work from the start of the investigation to the development of the application. However, several other functionalities which were idealized in the beginning of the application's requirements definition weren't developed in the current release of the project and some development techniques used in the development may also be improved, such as:

Fully integrate the forum feature by developing a method to synchronously login on the web portal and the portal's forum;

The chat feature to allow real-time communication between online users was already investigated for development purposes, but, due to several failed implementation attempts, this tool has not yet been successfully;

Who is online feature as a way of informing the online users who else is online at the moment;

Automatic online event management to allow the creation of online events and handle the teams' entries in these events, distribute the teams' automatically into legs and trials and prepare each team's binary code for actual simulations without the need for human intervention;

Autonomously run simulations by preparing and launching each trial from each leg of an event in the simulator server, subsequently gathering the resulting log files, parsing them automatically and reuniting the information from the logs, and afterwards generating the scoreboard, the media and log gallery and storing the data in the application's database, completely autonomously without ever needing the intervention of the administration;

Automatic users and teams state management by calculating the users' and/or teams' state alteration conditions and automatically changing the state if the conditions are met;

Fee payment implementation in order to allow a payment through several online available methods, like PayPal for example, or provide methods for payment through *ATM/ABM* or bank transfers;

Online testing simulations to provide the application's users a platform for testing of the code when it is submitted to the server;

Improve security measures to provide a better and safer application for protected use by users worldwide;

URL rewriting in order to simplify the web portal's URL displayed in the users' web browsers, allowing the users to remember it more easily, increasing the website's security and easing the indexing by search bots;

Improve web portal's design to evolve the interface's design according to the current technological trends and user needs;

Implement AJAX features to improve the communication between the client's web browser and the server, diminishing the traffic in the network and improving the feedback techniques;

Automatic backup to provide the feature without the need to unleash it first through human interaction, saving all the application's data to prevent its loss in case of system failures;

Allow the introduction of new kinds of competitions by providing a platform for the execution of other robotic competitions other than CiberMouse, such as CAMBADA or ROTA;

Develop transfer methods to export data from this application into other applications easily and without losing any information;

among other functionalities and improvements that the application may need along its course of life.

Bibliography

[1] Micro-Rato Organization *Micro-Rato Competition Website* http://microrato.ua.pt/ [Online on November 2009].

[2] Micro-Rato Organization *Ciber-Rato's first edition website* http://microrato.ua.pt/main/Historia/historia-edic2001-CiberRato.htm [Online on November 2009].

[3] Nuno Lau, Artur Pereira, Andreia Melo, António Neves and João Figueiredo *Ciber-Rato: Um Ambiente de Simulação de Robots Móveis e Autónomos* DETUA Magazine, vol. 3, nr. 7, pages 647-650 [September 2002] http://microrato.ua.pt/main/docs/artigos/ciberRato2002.pdf [Online on November 2009].

[4] António Neves, João Figueiredo, Nuno lau, Artur Pereira and Andreia Melo *O Visualizador do Ambiente de Simulação Ciber-Rato* DETUA Magazine, vol. 3, nr. 7, pages 651-654 [September 2002] http://microrato.ua.pt/main/docs/artigos/VisualizadorDet.pdf [Online on November 2009].

[5] Tim O'Reilly *What Is Web 2.0* O'Reilly Network [September 2005] http://oreilly.com/web2/archive/what-is-web-20.html [Online on November 2009].

[6] Darcy DiNucci *Fragmented Future* Article for Print magazine, Design & New Media column [April 1999] http://www.cdinucci.com/Darcy2/articles/Print/Printarticle7.html [Online on November 2009].

[7] Kingsley Uyi Idehen *RSS: INJAN (It's not just about news), August 2003* and *Jeff Bezos Comments about Web Services, September 2003* Kingsley Idehen's Blog Data Space online http://www.openlinksw.com/weblog/kidehen@openlinksw.com/127/index.vspx [Online on November 2009].

[8] Eric Knorr *The Year of Web Services* CIO [December 2003].

[9] Tim O'Reilly and John Battelle *Opening Welcome: State of the Internet Industry* Conference in San Francisco, California [October 2004].

[10] O'Reilly Media, Inc. *O'Reilly – Spreading the Knowledge of Technology Innovators* http://oreilly.com/ [Online on November 2009].

[11] Tim Bernners-Lee *Interview* developerWorks Interviews [August 2006] http://www.ibm.com/developerworks/podcast/dwi/cm-int082206txt.html [Online on November 2009].

[12] David Best *Web 2.0 Next Big Thing or Next Big Internet Bubble?* Lecture Web Information Systems, Technical University of Eindhoven [2006].

[13] Tim O'Reilly *Web 2.0 Compact Definition: Trying Again* O'Reilly Network [October 2006] http://radar.oreilly.com/archives/2006/12/web-20-compact.html [Online on November 2009].

[14] NetCraft *August 2009 Web Server Survey* NetCraft Web Surveys' Archive http://news.netcraft.com/archives/2009/08/31/august_2009_web_server_survey.html [Online on November 2009].

[15] Apache Software Foundation *Apache HTTP Server Project official website* http://httpd.apache.org/ [Online on November 2009].

[16] Microsoft Corporation *Microsoft IIS official website* http://www.iis.net/ [Online on November 2009].

[17] Google Corporation *Google Web Toolkit* http://code.google.com/webtoolkit/ [Online on November 2009].

[18] Data Center Knowledge *Google's Custom Web Server, Revealed* http://www.datacenterknowledge.com/archives/2009/04/01/googles-custom-web-server-revealed/ [Online on November 2009].

[19] Nginx Corporation *Nginx WebServer* http://wiki.nginx.org/ [Online on November 2009].

[20] LightTPD Organization *LightTPD fly light* http://www.lighttpd.net/ [Online on November 2009].

[21] Microsoft Corporation *Microsoft SQL Server official website* http://www.microsoft.com/sqlserver/2008/en/us/default.aspx [Online on November 2009].

[22] Sun Microsystems *MySQL official website* http://www.mysql.com/ [Online on November 2009].

[23] Oracle Corporation *Oracle official website* http://www.oracle.com/technology/index.html [Online on November 2009].

[24] PostgreSQL Organization *PostgreSQL official website* http://www.postgresql.org/ [Online on November 2009].

[25] W3 Organization *CGI Specs and Documentation by W3* http://www.w3.org/CGI/ [Online on November 2009].

[26] Microsoft Corporation *ASP.NET official website* http://www.asp.net/ [Online on November 2009].

[27] Mono Organization *Mono Platform official website* http://mono-project.com/Start [Online on November 2009].

[28] PHP Group *PHP official website* http://php.net/index.php [Online on November 2009].

[29] Sun Microsystems *JSP official website* http://java.sun.com/products/jsp/ [Online on November 2009].

[30] The Apache Software Foundation *Apache Tomcat Webserver* http://tomcat.apache.org/ [Online on November 2009].

[31] Sun Microsystems *GlassFish – OpenSource Application Server* https://glassfish.dev.java.net/ [Online on November 2009].

[32] JBoss Community *JBoss OpenSource Middleware* http://www.jboss.org/ [Online on November 2009].

[33] Sun Microsystems *NetBeans IDE* http://netbeans.org/ [Online on November 2009].

[34] Eclipse Foundation *Eclipse IDE* http://www.eclipse.org/ [Online on November 2009].

[35] GenuiTec *MyEclipse – Eclipse Plugin Development Tools IDE* http://www.myeclipseide.com/ [Online on November 2009].

[36] JetBrains *IntelliJ IDEA IDE* http://www.jetbrains.com/idea/ [Online on November 2009].

[37] ECMA International *ECMAScript official website* http://www.ecmascript.org/ [Online on November 2009].

[38] ECMA International *Standard ECMA-262* ECMAScript Language Specification [December 1999] http://www.ecma-international.org/publications/standards/Ecma-262.htm [Online on November 2009].

[39] Jesse James Garrett *Ajax: A New Approach to Web Applications* http://www.adaptivepath.com/ideas/essays/archives/000385.php [Online on November 2009].

[40] Adobe Systems *Adobe Flash Platform official website* http://www.adobe.com/flashplatform/ [Online on November 2009].

[41] Adobe Systems *Adobe Flash Professional official website* http://www.adobe.com/products/flash/ [Online on November 2009].

[42] Adobe Systems *Adobe Open Source Project* http://opensource.adobe.com/ [Online on November 2009].

[43] Microsoft Corporation *Microsoft SilverLight official website* http://silverlight.net/ [Online on November 2009].

[44] Mono Organization *MoonLight official website* http://mono-project.com/Moonlight [Online on November 2009].

[45] Microsoft Corporation *Microsoft Visual Studio 2008 Professional Edition* http://www.microsoft.com/visualstudio/en-us/products/professional/ [Online on November 2009].

[46] Microsoft Corporation *Microsoft Expression Studio* http://www.microsoft.com/expression/ [Online on November 2009].

[47] opensourceCMS Organization *Open Source CMS directory website* http://php.opensourcecms.com/ [Online on November 2009].

[48] commercialCMS Organization *Proprietary CMS directory website* http://commercialcms.com/ [Online on November 2009].

[49] Kevin Yank *Simply JavaScript: The Three Layers of the Web* http://articles.sitepoint.com/article/simply-javascript [Online on November 2009].

[50] Triple-Networks *Three Tier Architecture* http://triple-networks.com/documents/three-tier-architecture/ [Online on November 2009].

[51] Carlos Serrão and Joaquim Marques *Programação com PHP5* 2^{nd} Edition, FCA − Editora de Informática, Lda..

[52] Pedro Remoaldo *O Guia Prático do DreamWeaver CS3 com PHP, JavaScript e Ajax* 1^{ft} Edition, Centro Atlântico, Lda..

[53] Refsnes Data Organization *W3Schools Online Web Tutorials* http://www.w3schools.com/ [Online on November 2009].

[54] Users Community *Wikipedia* http://www.wikipedia.org/ [Online on November 2009].

[55] Refsnes Data Organization *W3Schools Browser Display Statistics and Color Depth Surveys* http://www.w3schools.com/browsers/browsers_display.asp [Online on November 2009].

[56] Refsnes Data Organization *W3Schoos Browser Statistics Survey* http://www.w3schools.com/browsers/browsers_stats.asp [Online on November 2009].