INFORMATION SOCIETIES TECHNOLOGY (IST) PROGRAMME

STATUS

"Software Architecture for Usability"

WORKPACKAGE: 3 Study of the usability/software architecture relationship

ANNEX E: PHASE 2 Validation with Practitioners in a Real Project

DELIVERABLE 3.4. Techniques, patterns and styles for architecture-level usability improvement

Version: 1.0.

Submission Date: 4/2/02

Authors: Camilo Clanderón

Partners: UPM

Stage:	Confidentiality:
[ ] Draft	[ ] Public - for public use
[ ] To be reviewed by WP participants	[ ] IST – for IST programme participants
[ ] Pending of approval by next consortium meeting	[X] Restricted – for STATUS consortium and PO
[X] Final / Released to CEC	

DOCUMENT CONTROL

Registration of Changes

Date	Version	Author of Changes	Comments

List of STATUS Related Documents

Document Name	Version
D.3.4.	1.0

TABLE OF CONTENTS

ANNEX E: PHASE 2 VALIDATION WITH PRACTITIONERS IN A REAL PROJECT

This annex contains the results of the Master Thesis of an UPM student who has applied the usability patterns defined in D.3.4. in order to contribute to the validation of the desing solutions proposed in such deliverable.

E.1 BUSINESS AREA

E.1.1 Introduction

The business of the company PUBLIGLOB is the rental of sign advertising space for a period of time, counted in weeks. The advertising spaces can be divided into three categories: public and private land and transport.

There are five main areas that support the company's business model, which are: operations, finance, liabilities management, sales and human resources. There is a sixth area, which does not fall directly within the business model, but does support most of the activities, which is the computing area.

E.1.2 Departments

- **Operations department:** The more technical business activities are grouped in what is known as the operations area. This area is responsible for the operational management of the urban furniture, maintenance, stock management, advertising sign preparation, distribution and affixation. The main activities are illustrated in Figure 2-1.

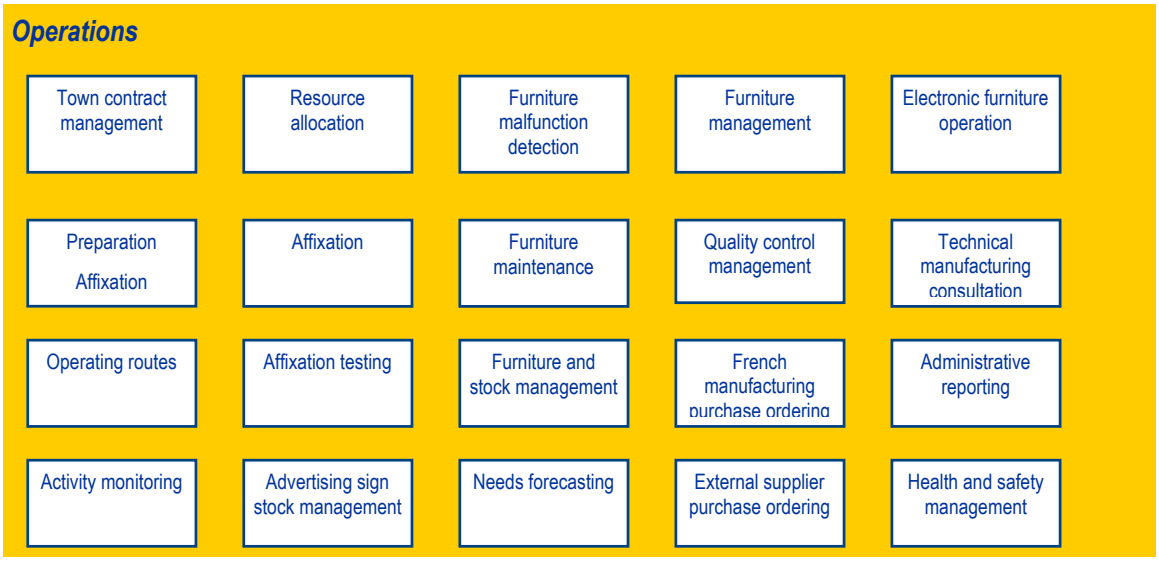

Figure 1 Operations department activities

- **Finance department:** This area is responsible for the financial management of the company and reports all movements to the head office based in France. The main activities are illustrated in Figure 2-2.

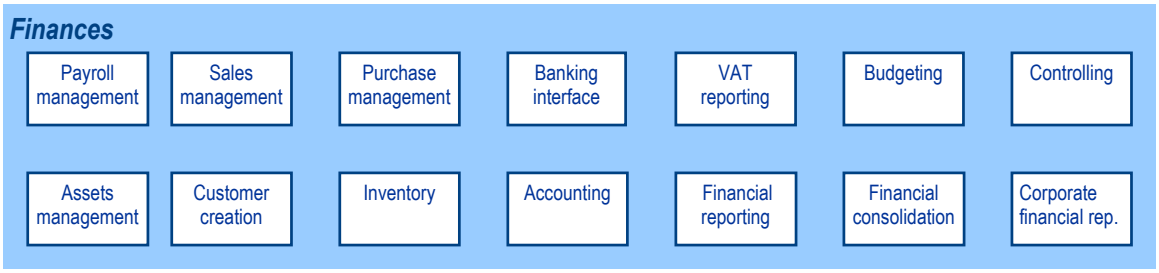

Figure 2 Finance department activities

- **Administration department:** This area is responsible for company liabilities management, issuing periodic reports for the financial department. Its main activities are illustrated in Figure 2-3.

Figure 3 Administration department activities

- **Sales department:** This area is responsible for selling the advertising spaces and dealings with customers. Its main activities are illustrated in 2-4.

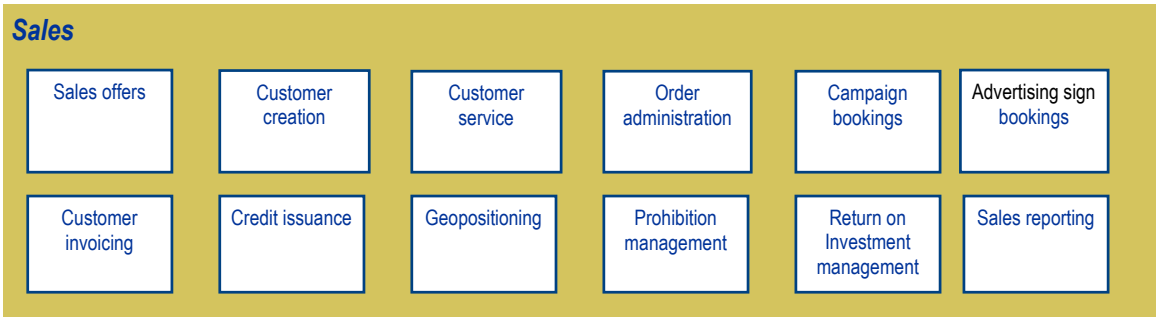

Figure 4 Sales department activities

- **Human resources department:** This area is responsible for managing the personnel employed by the company. Its main activities are illustrated in Figure 2.5.

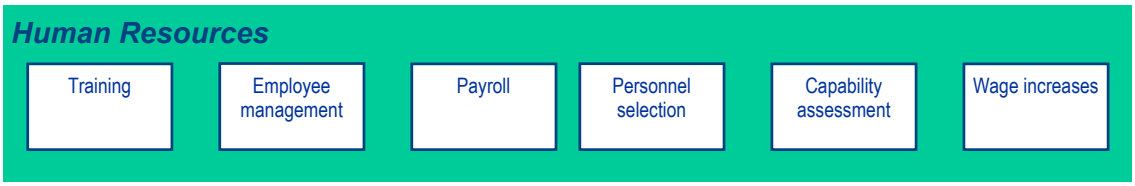

Figure 5 Human resources department activities

- **Computing Department:** This department does not directly fall within the company's business model, but cooperates with the different areas by providing support for the computer applications used in these areas. A brief description of these computer applications is given below.

- SIGES : Accounting.
- HYDRA & CARAT : Consolidation and group accounting reports.
- MONITOR : Company A bookings and assets management.
- PHOENIX : Company B bookings, assets and invoicing management
- SAIB : Company B rental supplier payment.
- TMD: Electronic furniture message sending and control.
- A3NOM: Payroll management.
- SPECIAL-PURPOSE DEVELOPMENTS: Some customised applications have been developed to meet some specific needs in the business areas.

- EXCEL / ACCESS: Some reports and small applications are generated using these tools.

Figure 2-6 shows a map of the specific business areas in which these applications are operational. The departments are located on the left-hand side of the figure and positioned so as to illustrate their interaction. The human resources department, which is responsible for managing the personnel in all areas, is located at the bottom. The operations department, which is the centre of the business, is placed in the centre of the figure. Above, but working in cooperation, are the administration and sales department and above these, the financial department, which is responsible for analysing the results and taking corrective measures to make the business more productive. Finally, the computer department is responsible for supporting the applications upon which the business operates. It is not shown as an independent department in this diagram, as, in terms of organisational structure, this department depends on and is part of the administration department. The applications it manages are included, however, as they are the central point of this research.

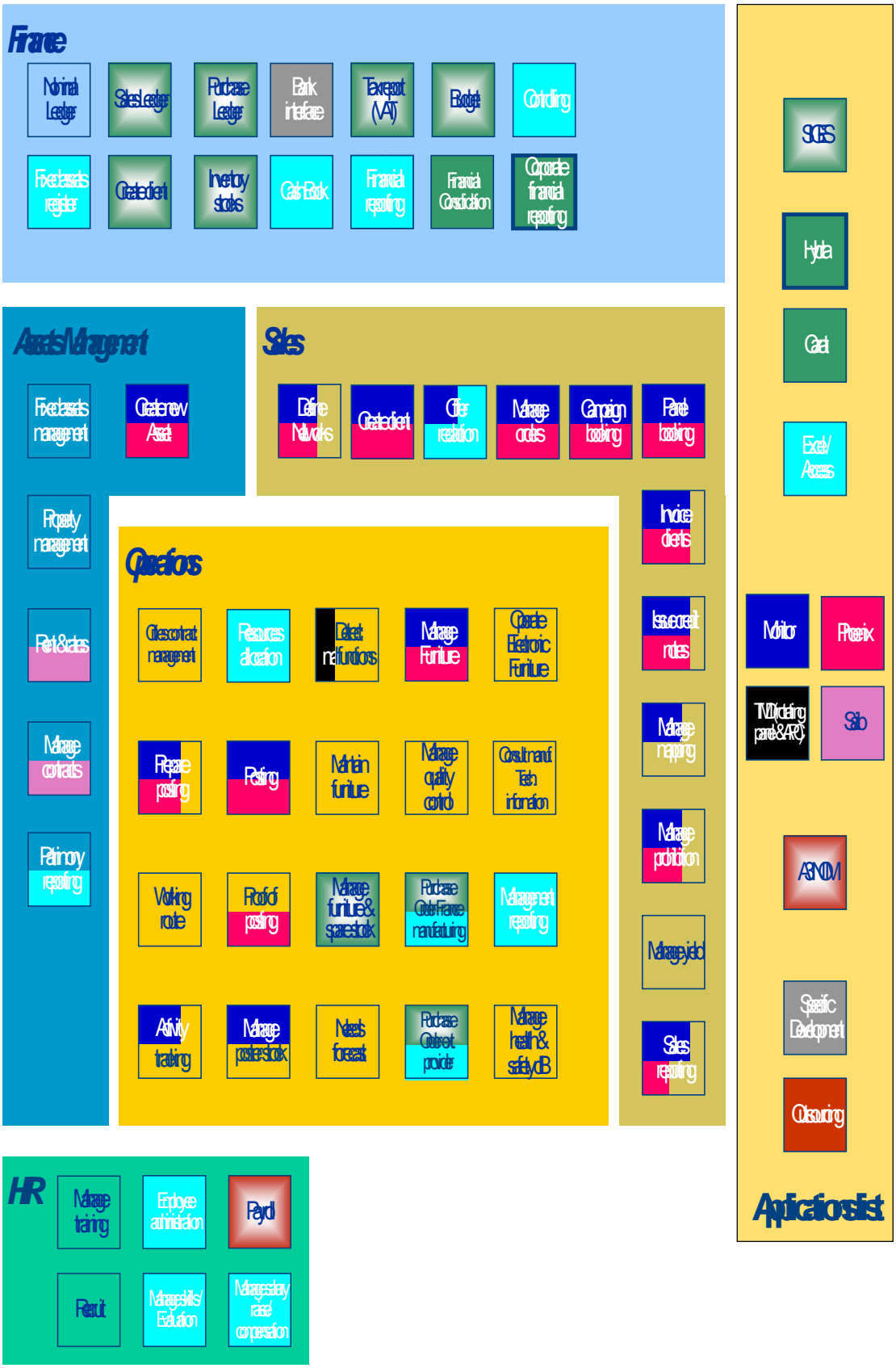

Figure 6 Map of applications

The information flows through interfaces between these applications, which run on heterogeneous technological platforms. Figure 2-7 shows the key data flows.

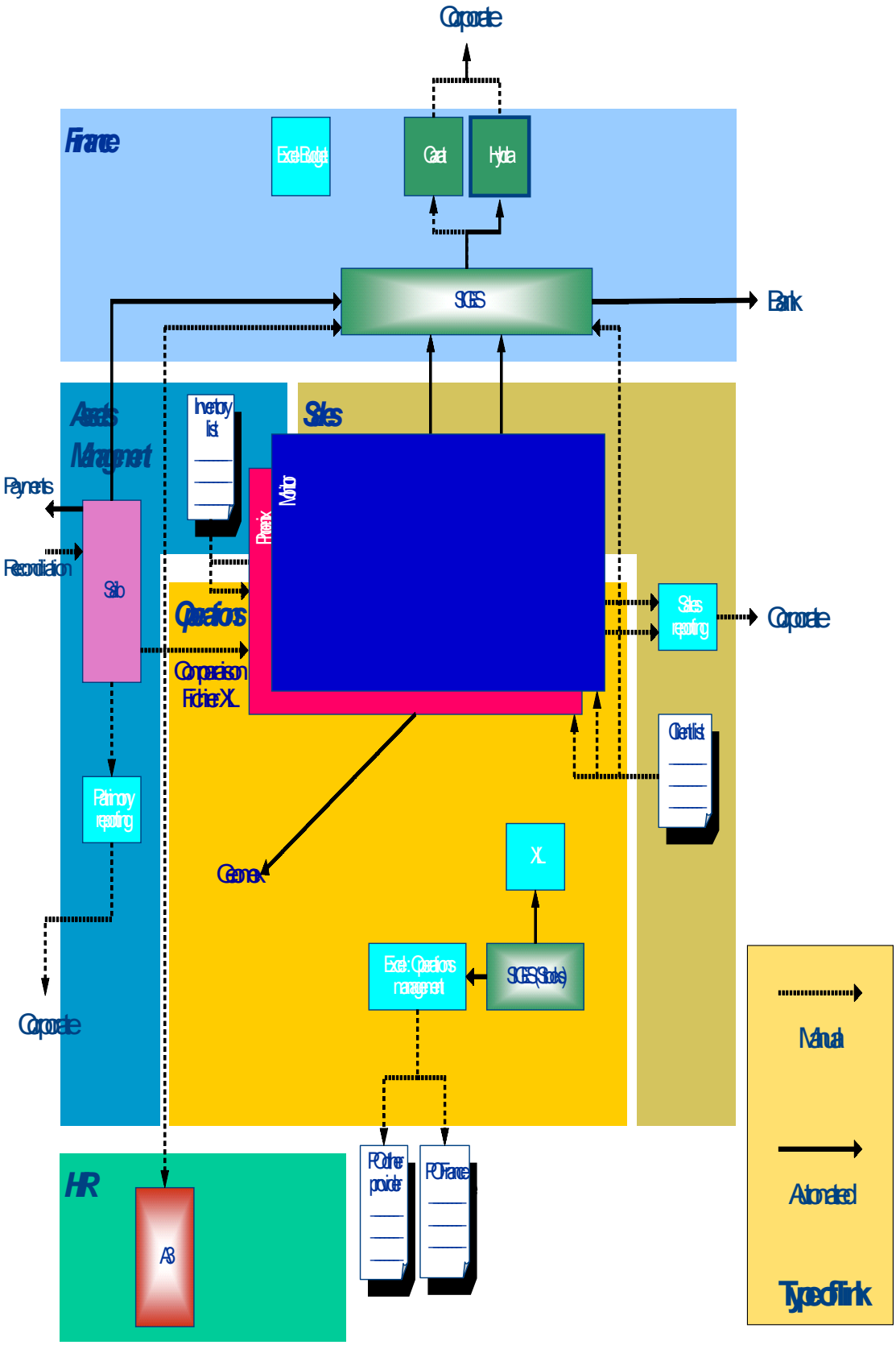

Figure 7 Map of applications

E.1.3 Control flow

Figure 2-8 shows a simplified summary of the business process and interaction between the departments.

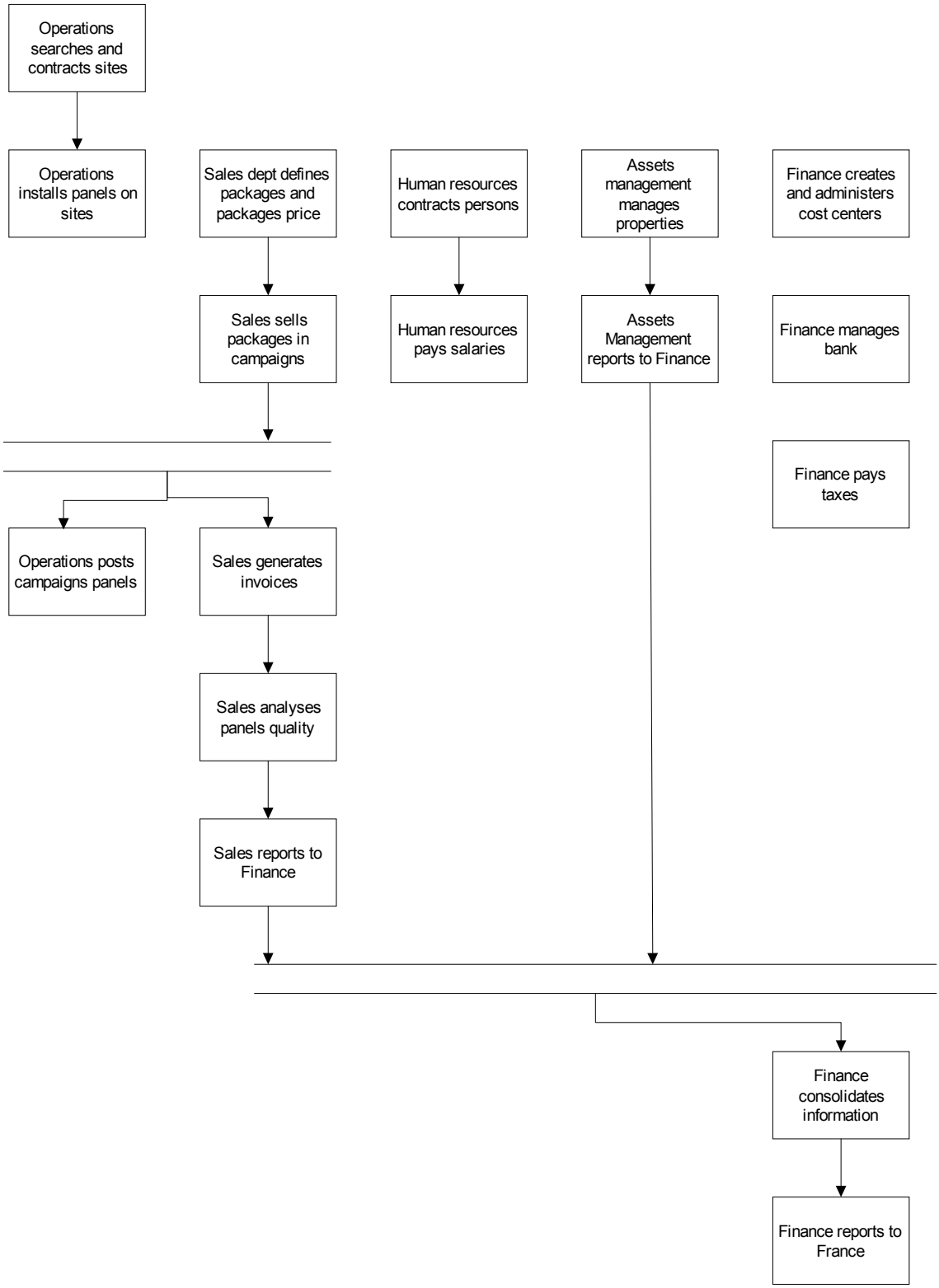

Figure 8 Control flow between departments

This is a very general overview. We will go into more detail as and when further information needs to be extracted for the analysis of the processes that the application under development is to support.

E.2 SOFTWARE REQUIREMENTS SPECIFICATION

E.2.1 Introduction

This document is a software requirements specification (SRS) for the system to be built. This specification has been drafted according to the guidelines established by "IEEE Recommended Practice for Software Requirements Specifications ANSI/IEEE 830 1998".

E.2.1.1 Purpose

The purpose of requirements specification is to clearly and precisely define the functionalities and constraints of the system to be developed. This document will serve as a basis for the team of developers, the quality group and the system end-users. This specification is subject to modification by the user group, which will be introduced by means of successive versions of the document until it is approved by the management, quality group and user group. Once approved, it will serve as a basis for the development team.

E.2.1.2 System scope

The future system will be called Intranet Publiglob España (IPESP). System development is driven by the decision of the computing department to set up a computer platform that unifies and centralises the information flow between the different companies, departments and computer applications used.

The management applications software support team, which is familiar with the details of the operation and data handled in the different business areas, has been commended with the development of an application that provides support to the processes of report generation, interface generation and loading, functionality extension and documentation. This application is not intended to replace and/or modify either the management tools or the implemented processes. It will operate in parallel to existing systems, providing support or enhancing the functionalities of these tools or the tasks performed by users.

The situation to be addressed by this system is that there are a series of applications implemented in the different group companies, which support the different business areas, but there are a series of problems concerning the integration of the information that should flow between these areas.

This project does not require any modification of the company's functional structure. The company's current technological infrastructure as regards both hardware and software will be used to take this development to all users whether they are working from the local Madrid network or from any of the 17 branches across Spain, which reduces the implementation and distribution costs of the final application.

E.2.1.3 Definitions

Agency	Company that designs advertising campaigns and rents radio, television and outdoor advertising spaces to broadcast the message.
Advertiser	Company interested in advertising a product.
Avenir	Company involved in the business of renting advertising spaces on private land.
Advertising face	In some cases, a piece of furniture may have one or more spaces for affixing advertisements. Each space is called a face.
Campaign	When a product is to be advertised, a campaign is created, which is consolidated by means of a booking.
Advertising sign	Advertising space.
Central	A company that rents radio, television, press and outdoor advertising space to broadcast a campaign.
Circuit	Group of advertising signs to be sold together.
Customer	Person to which the campaign is invoiced.
El Mobiliario Urbano	Company involved in the business of renting advertising spaces on public land.
Site	Place at which the hoardings are erected.
Geomex	Advertising sign geopositioning application.
Furniture	An object that fulfils a specific function to which advertising can be affixed, like a newspaper kiosk, bus stop or litter bin.
Monitor	Mobiliario Urbano company management application, used for bookings, affixation and assets maintenance.
Financial year	The year in which accounting operations are effected. A year is composed of 12 accounting periods.

Accounting period	The months of the year, numbered from 01 to 12. It is defined as the minimum time unit in which accounting movements are effected.
Phoenix	Avenir management application, used for bookings, affixation, assets maintenance.
Planning	Chart that indicates the distribution of bookings by time and by circuits.
Proprietor	Owner of a site.
Booking	Contract that defines the rental of advertising spaces for a period of time and on one or more circuits.
Saib	Avenir site supplier payment management application
Siges	Group accounting application.

E.2.1.4 Acronyms

SRS	Software Requirements Specification
IPESP	Intranet Publiglob España
DSI	Information Systems Department

E.3 References

- IEEE Recommended Practice for Software Requirements Specification. ANSI/IEEE std.830, 1998.
- Phoenix User's Manual.

E.1.1 Overview of the Document

This document is composed of three sections. The first section is an introduction to requirements specification. Section 2 briefly describes the system, and its main functions, constraints, assumptions and dependencies. Section 3 details the requirements to be met by the system.

E.3.1 General Description

This section presents a general description of the system. It defines the requirements it is to satisfy and the functions it is to perform, as well as the constraints and other factors influencing system development.

E.3.1.1 Product perspective

The system will not directly interact with any other software system, it will take the interfaces of other systems, which it will process to generate new information or modify their status.

E.3.1.2 System functions

In general terms, the system will have to perform the functions described in the following sections:

- *System user management*
- *Application management*
- *Report generation*
- *Interface management*
- *Extension of application functions*
- *Documentation management*

E.3.1.2.1 System user management

The system will have two basic user types:

- System administrators: Given certain privileges thanks to which they can access and modify application parameters. As several people may share the tasks of administration, there may be two or three administrators.
- Users: This user type will be divided into groups, which may or may not match the organisational units in which they work, and the system functionalities will be restricted depending on the group to which they belong. They have fewer privileges than an administrator and may, in no case, access system administration or documentation management.

E.3.1.2.2 Applications and menu management

The system will gradually incorporate a series of functionalities, which should be maintained in a menu that will be displayed to users depending on their permits. This application will be responsible for maintaining this menu.

E.3.1.2.3 Report generation

This category includes the applications described in the sections below.

E.3.1.2.3.1 Geomex additions, disposals and assets status

An application is to be developed that can be used to consult which hoardings of companies A and B entered in Geomex assets have been modified, added and disposed of.

Additionally, it should be able to show the differences between the Geomex database and the Monitor and Phoenix applications.

E.3.1.2.3.2 Phoenix manual invoicing report

A report showing an itemisation of the manual invoices made using the Phoenix application should be generated every month.

E.3.1.2.3.3 Outdoor assets report

The purpose of this report is to show the number of outdoor hoardings there are in a particular week in the Phoenix application.

E.3.1.2.3.4 Exclusives assets report

The purpose of this report is to show the number of exclusives hoardings there are in a particular week in the Phoenix application.

E.3.1.2.3.5 Average painting price

This query will output the average prices of company B's painted hoardings to be able to control Phoenix application errors.

E.3.1.2.3.6 Objectives report

This report will display what deviation there is between the number of advertising signs there are in each of the circuits in each town and the previously established objectives.

E.3.1.2.3.7 Query prohibitions

This report should show which hoardings have prohibitions.

E.3.1.2.3.8 Audience profitability report

This report should calculate the sales information on each hoarding, the average hoarding rate and cross-reference this information with audience information to measure the profitability and quality of each hoarding.

E.3.1.2.3.9 Campaign hoarding report

This report should show the hoardings booked for a particular campaign in a given period.

E.3.1.2.4 Interface management

Some programs generate data that should be processed by other applications. This requires some processing, validation and modification by the user responsible before outputting the final format required by the applications. An application is to be implemented to support these tasks and automate this process.

E.3.1.2.4.1 Phoenix-Siges Interface

A program should be developed that takes both the Phoenix application manual and automatic invoicing data and generates a flat file with the format required by the Siges program for later loading in the daybook.

E.3.1.2.4.2 SAIB-Siges Interface

A program is to be developed that takes the data on rental supplier payments and generates a flat file with the format required by the Siges program for later insertion into the SIGES DB daybook table.

E.3.1.2.5 Extension of application functionality

E.3.1.2.5.1 Monitor Planning

This application should interactively show the bookings and booking type for a given week in the Monitor application, resembling the existing wall chart as closely as possible.

E.3.1.2.5.2 Phoenix bookings change of status

This program should permit the status of bookings to be changed even after they have been invoiced, as the Phoenix application does not allow this variable to be altered once the accounting period has lapsed, and the accounts department needs to change the status of the booking during the invoicing process.

E.3.1.2.5.3 Classification of bookings by marketing category

This development will classify bookings and show more information by means of a new categorisation of the bookings associated with the advertiser, product and category.

E.3.1.2.5.4 Affixation control

This development will permit the booking status to be entered for a specific week. The Phoenix application actually does manage this, but the process is very complex and takes too long, which means that it is not done.

E.3.1.2.6 Application manual and documentation querying

Within the scope of the application, a module will be generated that will allow users to easily, clearly and quickly search and query the most routine tasks of the applications used within the organisation on the basis of existing manuals.

E.3.1.3 User characteristics

The users will be clients who which to access the system functionalities by means of the local area network using the Microsoft Internet Explorer 5.0 or higher web navigator. Therefore, this system should have an easy-to-use interface. It is assumed that the users of this

application are accustomed to using Windows and Internet Explorer. It is also assumed that each user understands and uses the applications that support the business area in which they work.

E.3.1.4 Constraints

The platform selected to develop this application is Microsoft.Net.

This application is not considered critical. However, it is considered that it should be available from 5 am to 8 pm from Monday to Friday with service interruptions of a maximum of 2 hours.

It is essential for users to be registered as corporate network users, which will enable easy user maintenance and the use of the Windows 2000 network security system.

E.3.1.5 Assumptions and dependencies

E.3.1.5.1 Assumptions

It is assumed that the requirements described in this document are subject to modification. Any request for a change in the specification must be approved by all the parties concerned and managed by the configuration management group.

E.3.1.5.2 Dependencies

The system needs to communicate with other external systems, either by means of interface file or proprietary solutions for user validation or database access tasks (ODBC).

System availability will depend on the connection between the client machines (on which the client program will be resident) and the server.

E.3.2 Specific requirements

This section presents the functional requirements to be met by the IPESP system. These requirements have been specified to assure that they can be easily be demonstrated.

Some of these requirements are related to usability patterns. These requirements will be highlighted to makes them easily identifiable and be able to trace the changes that these patterns introduce in the design and analysis phases.

E.3.2.1 Functional requirements

E.3.2.1.1 System user management

Req A1	The system will have 2 user types: ▪ System administrators, who will have all the privileges to access the system and all data. Additionally, they will be able to perform application, menu and user control parameterisation tasks. ▪ Users will only have access to the functions they have been assigned by the administrators, and the menu will only list the options to which they have access.

Req A2	To be able to access system data, the users will have to be previously validated in the corporate network. And these validation credentials will be the means by which the system will be able to identify what permits they have.

Req A3	The system must control user access, irrespective of whether they are administrators or normal users. Every time users want to access the system, their credentials should be validated. If their credentials cannot be located, they should enter their user name and password. If these do not match any name stored in system, the user will not be able to access the system.

Req A4	Passwords will only be able to be changed through Windows or by means of application to the network administrator.

Req A5	The system should allow administrators, depending on their privileges, to query all the details of the following: ▪ User and group permits.

Req A6	The system should provide maintenance functions that will help administrators with queries and modifications.

E.3.2.1.2 Geomex assets additions, disposals, modifications and status

Req B1	It should be possible to select the start and end of the period to be queried.

Req B2	It should specify what assets have been added, disposed of and modified from the start to the end of the period.

Req B3	These changes should only be specified for Phoenix assets that match the Geomex classification and belongs to associate 01.

Req B4	The assets at the end of the period should be specified.

Req B5	The information to be viewed in this report is described in the Geomex assets additions, disposals, modifications and status data document.

Req B6	The results of the query should be able to be exported to Excel.

E.3.2.1.3 Phoenix manual invoicing report

Req C1	It should be possible to query what manual invoices have been issued for a given accounting period.

Req C2	The information should be grouped and ordered by towns.

Req C3	The information to be viewed in this report is described in the manual invoicing data document.

Req C4	The results of a query should be able to be exported to Excel.

E.3.2.1.4 Outdoor assets report

Req D1	It should be possible to query how many outdoor advertising signs there are, ordered by town and circuit for a given Phoenix period.

Req D2	The major groups of advertising signs to be identified are circuits, unit sale, local and unexploitable modules.

Req D3	The advertising signs of all the associates should be included and they should be able to be filtered by this criterion.

Req D4	The advertising signs should be grouped by format within each group. See outdoor assets report data to identify the information to be displayed.

Req D5	The results of a query should be able to be exported to Excel.

E.3.2.1.5 Exclusives assets report

Req E1	It should be possible to query how many exclusives outdoor advertising signs there are, ordered by town and circuit, for a given Phoenix period.

Req E2	The different exclusives should be separated.

Req E3	Each circuit of each exclusive should be separated.

Req E4	The results of a query should be able to be exported to Excel.

Req E5	The advertising signs should be grouped by format within each group. See exclusives assets data to identify the information to be displayed.

E.3.2.1.6 Average painting price report

Req F1	The average price of painted hoardings for the current period should be specified by municipality.

Req F2	Only own assets should be included.

Req F3	The results of a query should be able to be exported to Excel.

Req F4	For a description of the required fields, see average painting price data.

E.3.2.1.7 Objectives report

Req H1	This report should show the current status of the assets as compared with the defined objectives.

Req H2	Only own assets should be included.

Req H3	The user should be able to view and modify the objectives currently defined for each of the circuits.

Req H4	The information should be ordered by towns. See objectives report data to identify the information that should be displayed.

Req H5	It only includes information on circuit hoardings.

Req H6	The results of a query should be able to be exported to Excel.

E.3.2.1.8 Prohibitions report

Req I1	It should be possible to query the hoardings that have prohibitions for a given period.

Req I2	It should be possible to make a query by period, associate, prohibition type, municipality and site.

Req I3	This report should include own and associates' assets.

Req I4	The information should be ordered and grouped by town, municipality and site. See prohibitions report data to identify the information to be displayed.

Req I5	The results of a query should be able to be exported to Excel.

E.3.2.1.9 Campaign hoardings report

Req J1	It should be possible to query the hoardings booked for a specific campaign and for a given period.

Req J2	Which hoardings are owned by the company and which by associates should be specified.

Req J3	For a description of the required fields, see campaign hoardings report data.

Req J4	The results of a query should be able to be exported to Excel.

E.3.2.1.10 Phoenix-Siges interface

Req L1	The Phoenix invoicing data should be taken and translated to a flat file formatted as defined by the Siges application for later loading.

Req L2	It should check that all the invoiced customers are registered and their data are correct.

Req L3	It should provide the option of checking that the amount of the transferred data matches the data given by Phoenix.

Req L4	It should check that the sum of the invoicing details matches the heading and, if not, they should be able to be adjusted at the users' discretion.

Req L5	For a description of the required fields, see Phoenix-Siges interface data.

E.3.2.1.11 SAIB-Siges interface

Req M1	The file containing the SAIB rental payment data should be taken and the data inserted into the SIGES tables.

Req M2	The process should be automatic by means of file selection.

Req M3	It should be possible to check the transferred amounts before they are inserted in the database.

E.3.2.1.12 Monitor Planning

Req N1	This application should visualise the status of the bookings by region, town, circuit and week in a manner that resembles as closely as possible the current wall chart.

Req N2	It should be possible to run searches by region, town, circuit, starting and ending period.

Req N3	If a circuit has more than one booking in a week, this should be specified visually.

Req N4	If a circuit has more than one booking in a week, the higher category booking should be automatically displayed.

Req N5	It should be possible to view the details of each circuit booking for a given week.

Req N6	The data should be as recent as possible.

Req N7	For a description of the fields required and the wall chart, see Monitor planning data.

E.3.2.1.13 Change of Phoenix bookings status

Req O1	An easy-to-use interface should be generated to easily modify the booking status from B to I and vice versa.

Req O2	Its use should be restricted to an authorised user group.

E.3.2.1.14 Bookings classification by marketing category

Req P1	The contracts report should be able to be generated for any accounting period.

Req P2	Each campaign should be associated with preliminary information supplied by the contract, and this contract should be able to be queried and updated automatically. See bookings classification by category data for the description of the required fields.

Req P3	The total invoiced per booking should be included, and if further invoice details are required, the invoices should be able to be queried.

Req P4	If booking payments have been made, these should be included, and if further payment details are required, these should be able to be queried.

Req P5	A new classification of Advertiser, Campaign, Product, Sector should be created separate from the Phoenix classification, which should match the Monitor classification.

Req P6	The number of campaign advertising signs should be given as the weekly average of advertising signs over bookings.

Req P7	The starting date should be the first date of advertising sign affixation and not the date on which the booking is made.

Req P8	If the booking refers to an exclusive, the name of the exclusive should be stated (e.g., Barcelona Metro, Bilbao Metro).

Req P9	Only the Blocked bookings will be analysed.

Req P10	The application should allow maintenance of the tables concerned, save any, which for reasons of consistency, can only be modified from the Phoenix application.

Req P11	The system should detect the client operating system language and, depending on this, show the messages and labels in the detected language. If the detected language is not one of the parameterised languages or cannot be detected, the messages and labels will be shown in English.
Us Pattern	Different languages

Req P12	The system should allow information access and visualisation by means of a standard web page or data access by means of a web service for use by a remote application.
Us Pattern	Different access methods

Req P13	If an application error occurs, the application should inform the user to take action. Alerts should be issued by means of visible messages the user can see.
Us Pattern	Alert

Req P14	To speed up the process of data entry, the system should allow the use of abbreviated methods for data deletion, entry or modification functions.
Us Pattern	Shortcuts

Req P15	If the user makes a mistake when entering any of the data or wants to go back to the default values during a contracts report request, the system should allow the user to go back to the values entered previously.
Us Pattern	Undo

Req P16	If a new contract is to be entered, the system should allow users to use an existing contract as a model, that is, to get all the contract data so as not have to rewrite all the information.
Us Pattern	Reuse information

Req P17	The query parameters should be validated.
Us Pattern	Form/Field validation

Req P18	When requesting a report, it should be possible to ask for a preview of the result before giving the order to print.
Us Pattern	Preview

Req P19	A historical record should be kept of each of the queries executed in remote mode, including information on the query date and time, the user who made the query and the query parameters.
Us Pattern	History logging

Req P20	During the data search process, the system should report its status so that the user does not try to run a query in the same window.
Us Pattern	Status indication

Req P21	The user should be able to cancel report generation while it is being processed.
Us Pattern	Cancel

Req P22	The user should be able to open two program instances and run two queries at the same time.
Us Pattern	Multitasking

Req P23	The system should be able to export the generated report to an Excel format file via the route defined by the actor.

E.3.2.1.15 Application manuals and documentation management

Req Q1	The most important topics for each of the organisation's in-house applications should be included.

Req Q2	It should allow subject-related search.

Req Q3	It should be possible to send a mail with the text of a selected subject.

Req Q4	It should be possible to include remarks on a given subject to enhance the content or ask the administrator for a correction.

E.3.2.1.16 Affixation control

Req R1	It should be possible to change the status of a booking's affixation tickets to any of the 7 possible statuses and the date on which the incident takes place.

Req R2	An observations field should be included for a specific affixation job, which users will be able to use to note down their remarks on a job.

Req R3	Each depot should be able to manage its tickets.

Req R4	It should be possible to show a report of the incidents by town grouped by type for days 1 to 5, specifying the percentage of affixation completed each day and the number of advertising signs not affixed each day.

Req R5	It should be possible to view the details of the advertising signs not affixed and the type of incident that prevented affixation in an affixation job report.

Req R6	See affixation incidents report data for the description of the input files.

Req R7	It should be possible to change the status of multiple tickets.
Us Pattern	Action for multiple objects

Req R8	It should only be possible to change the status of the tickets that match the user's depot.
Us Pattern	User profile

E.3.2.1.17 Applications and menu administration

Req S1	It should be possible to enter a new application.

Req S2	It should be possible to modify the information on an application.

Req S3	It should be possible to delete an application and its information.

Req S4	The application menu should be generated from the applications and their classification.

Req S5	A menu element can be the principal leaf or node.

Req S6	The applications are menu leaves.

E.3.2.1.18 Audience profitability report

Req T1	The income generated by each advertising sign should be calculated and cross-referenced with its cost and audience data.

Req T2	An accounting period should be selected in which to compare the rental payment data and respective audience file.

Req T3	See audience profitability report data for a description of the input files.

Req T4	See audience profitability report data for a description of the output results.

E.3.2.2 External interface requirements

E.3.2.2.1 User interfaces

The user interface should take the shape of web pages or programs integrated into these web pages and based on web interface standards. The system will be managed by a web navigator.

E.3.2.2.2 Hardware interfaces

None have been defined.

E.3.2.2.3 Software interfaces

As the characteristics of the system are mainly based on web content, it will be related to programs like web navigators and servers.

E.3.2.2.4 Communication interfaces

The connection to the network will be established in the traditional manner, that is, an Internet connection through a network protocol. The protocol is TCP/IP (Transmission Control Protocol/Internet Protocol), and it will be possible to transmit the information via the http protocol or activex controls, both of which are supported by the web navigators.

The communication with other databases will be assured by the proprietary OleDB drivers of each database supplier.

E.3.2.3 Performance requirements

Taking into account the operating environment, the system should accept a lot of accesses simultaneously and should respond instantaneously. Performance will depend above all on the installed hardware requirements.

In some cases where more intensive processing is required, the action may be executed in asynchronous mode, and the user will be informed when the process ends. However, the system should not freeze while waiting for the response.

E.3.2.4 Development requirements

An incremental life cycle has been chosen to development the system, which will be based on use cases so that changes and new functionalities can be incorporated.

E.3.2.5 Technology requirements

The system should be implemented on existing platforms and will be able to run on Windows 9.x or higher and Internet Explorer 4 or higher. It will be divided into two parts: client and server.

The server application will be resident on a server connected to the corporate network. Any of the following *drivers* will be used for database access*:* ODBC, ORACLE and SQL Server.

The client application will run on a PC connected to the corporate network. A web navigator and the Microsoft Office suite will be essential for viewing the generated data.

E.3.2.6 Attributes

E.3.2.6.1 Security

Each time any user whatever tries to access the system data, the user credentials will be checked against the corporate network. If it cannot access this information, the system will request validation. If the data entered do not match the stored data, it will display an error message.

Depending on the user type, access will be granted to the information and functions to which the user is entitled. The user types are mentioned in the functional requirements, to which users will have access through a personalised menu.

E.3.2.6.2 Portability

The system will be implemented using the ".Net" platform, which means that the portability requirements will be as permitted by this platform.

E.3.2.6.3 Usability

The system should ease user interaction with the interface, simplifying information search and giving responses rapidly.

E.3.3 System use case identification

E.3.3.1 Introduction

This document describes the IPESP use case diagrams using UML (Unified Modeling Language) notation. First, they are illustrated by a use case diagram, then they are briefly described to give a rough understanding of the complexity and functionality of the system [Larman, 1999]. A conceptual model of the system concepts is also built.

E.3.3.2 Use case diagram

The high-level use case diagram shown in Figure 3-1 illustrates all the IPESP system use cases. This diagram establishes the external actors and their actions with respect to the system.

Figure 9 High-level use case diagram

E.3.3.2.1 Identification of actors participating in the system

An actor is an external entity that participates in the use cases definition. The actors participating in the IPESP system are defined below.

- Most of the system users are employees who can access the system by means of a PC connected to the corporate network, for which purpose they must have been registered in the domain. These users are classed depending on the department for which they work as follows: finance, assets management, sales (SOS-sales operation support), operations (technical), human resources.
- The administrator is responsible for administering the IPESP configuration and permits.
- The content manager is responsible for including, classing and updating the information on the manuals of the different applications.

E.3.3.2.2 Use case description in high level format

E.3.3.2.2.1 Use case: Administer users

Actor	Administrator
Type	Primary
Description	The actor selects the corporate network user. The system displays the different applications that the user can access. The administrator should assign the applications that the user is entitled to operate.

E.3.3.2.2.2 Use case: Administer menu

Actor	Administrator
Type	Primary
Description	The system displays the menu structure graphically to the actor. The actor will be able to add, delete or modify the menu applications.

E.3.3.2.2.3 Use case: Administer manuals

Actor	Content manager
Type	Primary
Description	The actor will be able to insert, delete or modify the manuals added to the system. First, the applications for which the manuals are to be managed should be entered, then the main functionalities of each application will be inserted.

	Having entered the above information, a manual can be inserted, for which purpose the application to which the manual in question refers, the functionality to which this manual belongs, a descriptive text, a list of keywords separated by ";"and the file containing the manual in the actor's hard disk will be selected. Having entered this information, the system will save the information and transfer the local file to the server.

E.3.3.2.2.4 Use case: Query manuals

Actor	Administrator, content manager, assets management, SOS, marketing, invoicing, sales.
Type	Primary
Description	The actor will be able to query the manuals by any of the following criteria: application, functionality or keywords. The system will return a list of the manuals that match these criteria containing the following information: the application to which the manual in question refers, the functionality to which this manual belongs, a descriptive text and a link to the manual file for downloading.

E.3.3.2.2.5 Use case: Update data

Actor	Administrator, sales, invoicing.
Type	Primary
Description	Some of the applications developed require data from external applications like Geomex, Phoenix, Siges or SAIB. In some cases, these data can be directly extracted from the databases by means of ODBC connections or must be inserted by means of a flat file generated previously in the external applications. The user will be able to load data from an external database directly by means of a button, unless they are already being loaded by another user at the time, in which case a warning message will be given. To load data from a flat file, the actor will have to select this file and press a button to start loading the file to the system. In either case, the system should indicate the last date on which these data were loaded and by which user.

E.3.3.2.2.6 Use case: Generate outdoor assets report

Actor	Assets management

Type	Primary
Description	The actor enters the application. The system displays the list of generated reports. The user selects one of the generated reports and the system returns the file in MS Excel format. If the report has not been generated, the user selects the period for which the report is to be generated. The system generates the report and the report appears in the reports list. The user will be informed of whether the report has already been generated or is in the process of generation.

E.3.3.2.2.7 Use case: Generate exclusives assets report

Actor	Assets management
Type	Primary
Description	The actor enters the application. The system displays the list of generated reports. The user selects one of the generated reports and the system returns the file in MS Excel format. If the report has not been generated, the user selects the period for which the report is to be generated. The system generates the report and the report appears in the reports list. The user will be informed of whether the report has already been generated or is in the process of generation

E.3.3.2.2.8 Use case: Generate objectives report

Actor	Assets management
Type	Primary
Description	The actor enters the application and requests the report for the current period. The system displays the report. The user asks for the report in MS Excel and the system returns the requested file. The user asks for the currently defined objectives. The system displays the objectives.

E.3.3.2.2.9 Use case: Generate average painting price report

Actor	SOS
Type	Primary
Description	The actor enters the application and selects the period for which he wishes to view the report. The system displays the information for the

	wishes to view the report. The system displays the information for the selected period. The user asks for the report in MS Excel and the system returns the requested file.

E.3.3.2.2.10 Use case: Generate Geomex assets status report

Actor	Marketing
Type	Primary
Description	The actor selects the Phoenix accounting period against which he wishes to compare the Geomex assets. The user verifies the Geomex application route. The system returns the requested information. The user can export the data to MS Excel.

E.3.3.2.2.11 Use case: Classify bookings by marketing category

Actor	Marketing
Type	Primary
Description	The actor selects the accounting period that he intends to query and requests the information. The system returns the requested information. The user asks for the report in MS Excel. The system returns the requested file

E.3.3.2.2.12 Use case: Generate audience profitability report

Actor	Marketing
Type	Primary
Description	The actor selects the Phoenix accounting period against which he intends to compare the income data. The user selects the Excel file containing the SAIB (supplier payment) Excel file. The user selects the flat file containing audience information. The system returns the requested information. The user asks for the report in MS Excel. The system returns the requested file.

E.3.3.2.2.13 Use case: Query Monitor planning

Actor	Sales
Type	Primary
Description	The actor selects the region, towns, circuits, starting period and ending period. The system returns the requested information. The user selects a

	period. The system returns the requested information. The user selects a week on which he would like to view further details. The system displays the details of the bookings associated with this period, town and circuit.

E.3.3.2.2.14 Use case: Generate campaign advertising signs report

Actor	Sales
Type	Primary
Description	The actor selects the booking and the period for which he would like to view the advertising signs. The system returns the requested information. The user asks for the report in MS Excel. The system returns the requested file.

E.3.3.2.2.15 Use case: Generate Phoenix manual invoicing report

Actor	Invoicing
Type	Primary
Description	The actor selects the accounting period for which he would like to view invoicing. The system returns the requested information. The user asks for the report in MS Excel. The system returns the requested file.

E.3.3.2.2.16 Use case: Generate Phoenix-SIGES interface

Actor	Invoicing
Type	Primary
Description	The actor selects the accounting period for which he would like to import the file. The system validates the information and generates the requested file and e-mails it to the user who requested the file.

E.3.3.2.2.17 Use case: Load SAIB-SIGES interface

Actor	Invoicing
Type	Primary
Description	The actor selects a file to be loaded, the system checks the file and loads the information in Siges.

E.3.3.2.2.18 Use case: Change bookings status

Actor	Invoicing

Type	Primary
Description	The actor selects the bookings whose status is to be changed and then asks the system to change their status. The system changes the status of the booking.

E.3.3.2.2.19 Use case: Control affixation

Actor	Technical
Type	Primary
Description	The actor queries and modifies the status of the booking affixation tickets.

E.3.3.3 System conceptual model

The conceptual model shown in Figure 2-3 shows all the concepts and associations identified for the IPESP system.

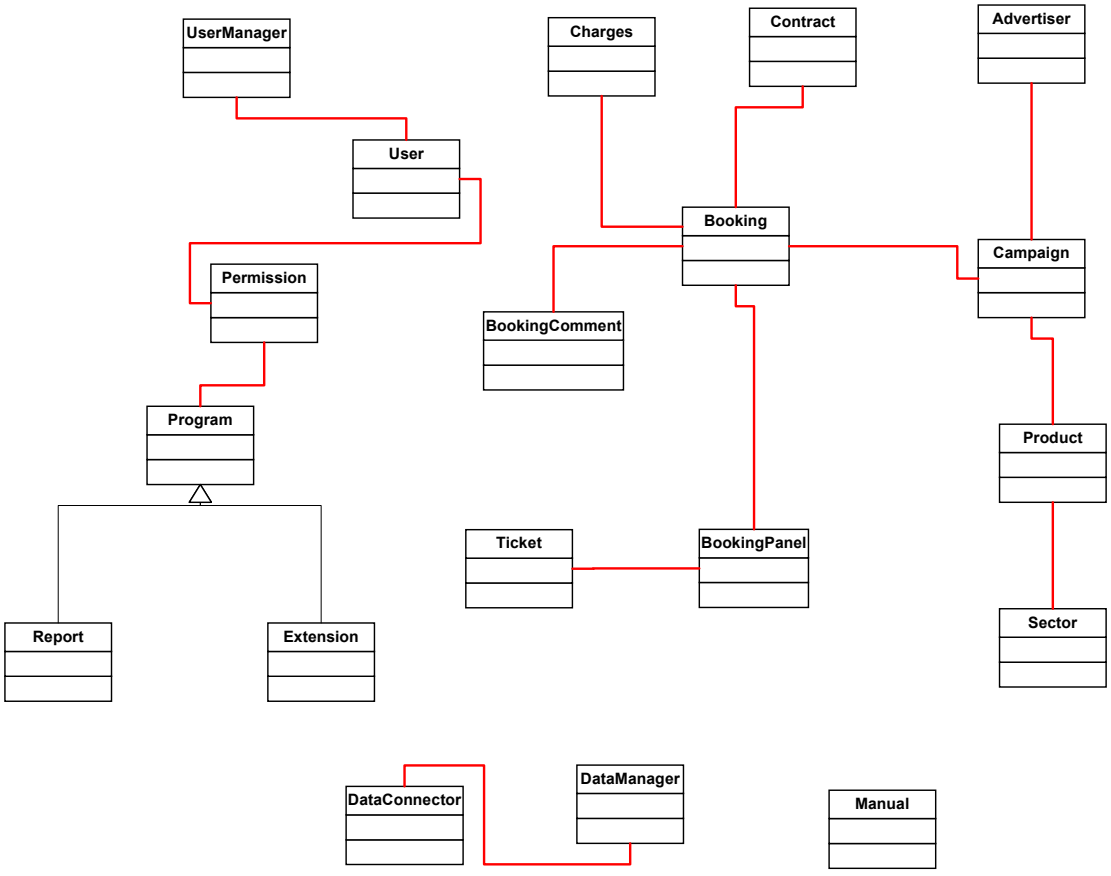

Figure 10 IPESP domain conceptual model

The above conceptual model is a useful representation of concepts in the domain of this project. Its goal is to improve the understanding of the problem within the requirements context.

E.4 DEVELOPMENT CYCLE 1 (WITHOUT USABILITY PATTERNS)

E.4.1 Cycle 1 analysis

This cycle addresses the analysis of the IPESP system. The steps taken are as follows: description of the use cases in expanded format, construction of sequence diagrams and preparation of operations contracts. The requirements related to usability patterns will not be included in this first analysis.

E.4.1.1 Use cases in low level format

In the following, we present the low-level use cases for the development cycle 1 applications for which insufficient detail was provided by the first approximation.

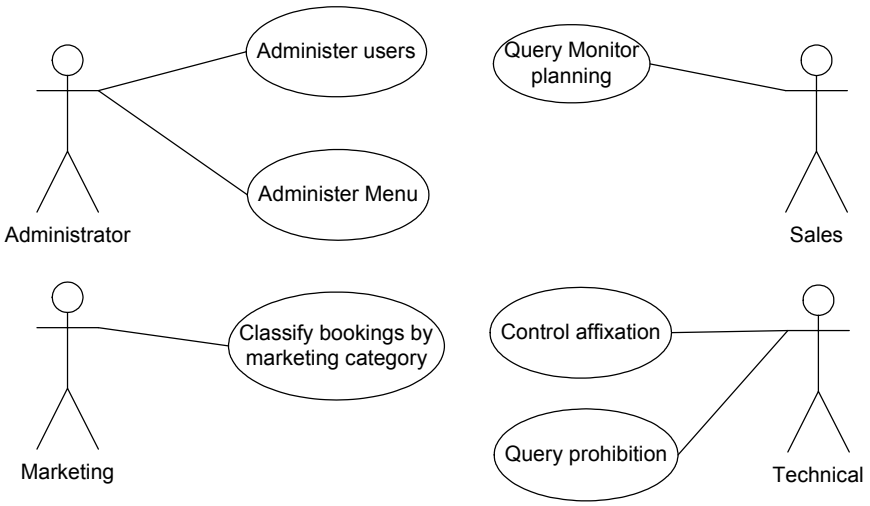

Figure 11 Marketing bookings classification use case diagram

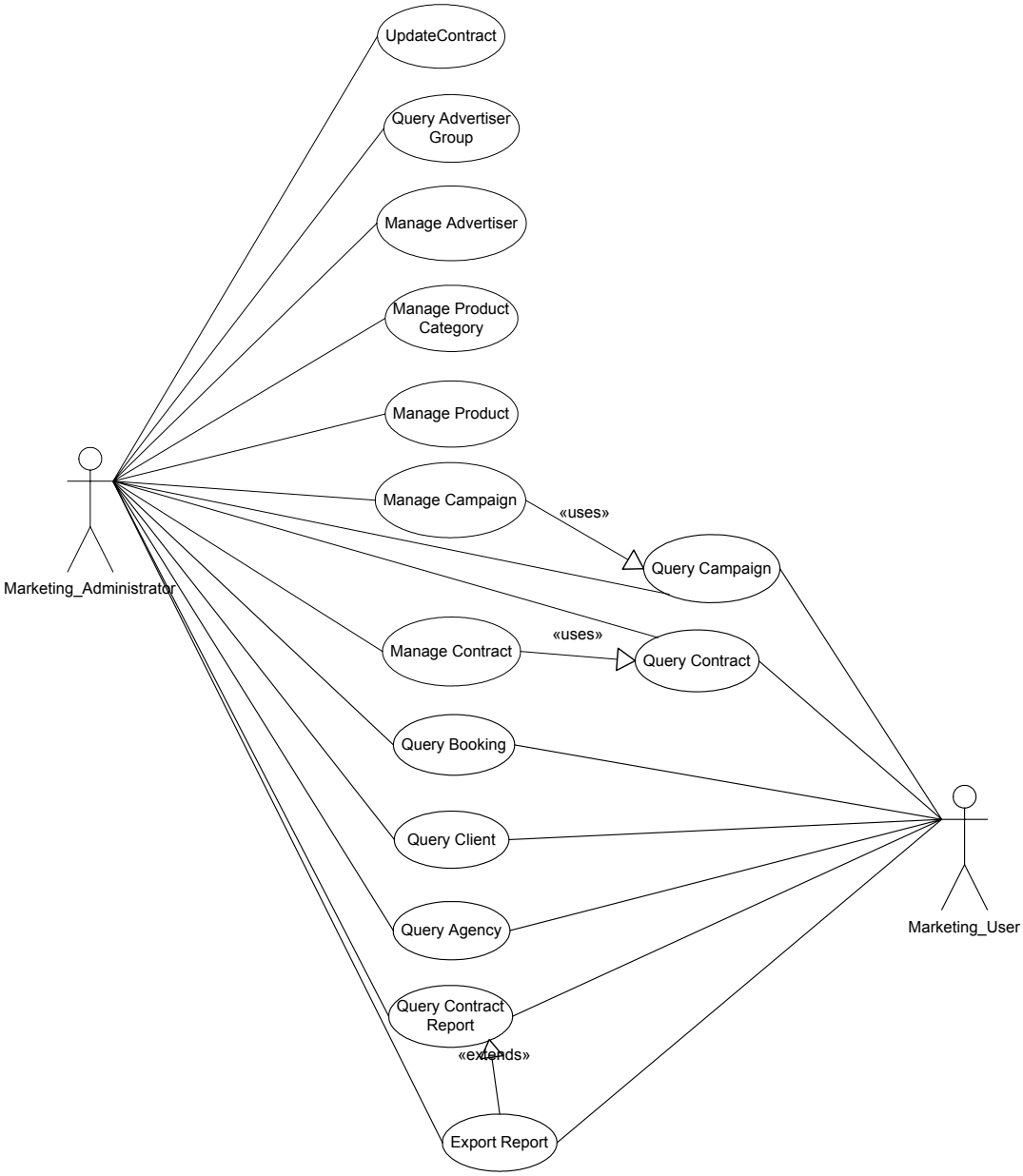

Figure 12 Marketing bookings classification use case diagram

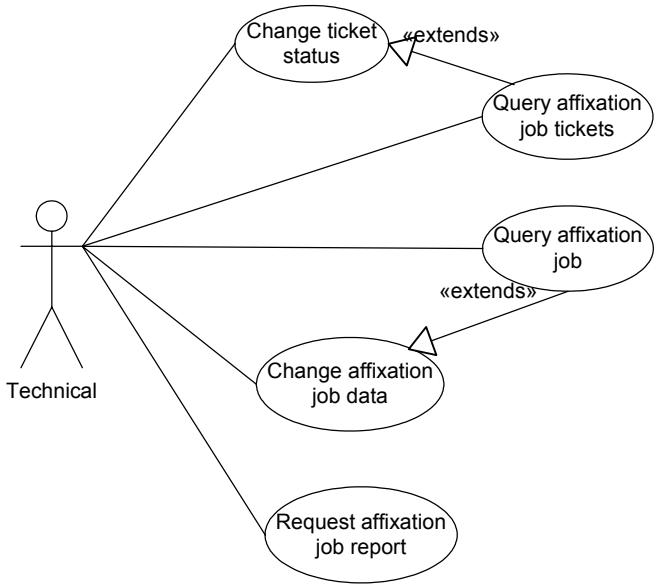

Figure 13 Control affixation use case diagram

E.4.1.2 Expanded format use cases

The use cases are described in expanded format in more detail below. This format describes the use case at more length than the high level format, which is of considerable help for building the sequence diagrams.

E.4.1.2.1 Use case: Administer users

Actor	Administrator	
Type	Primary and essential	
Purpose	Assign a permit to a user for the parts of the application this user can access.	
Overview:	The administrator selects a user name or user group in the corporate network, assigns the application modules which the user can access and requests registration.	
References:	*Req* A1, A5, A6	
Typical course	**Actor action**	**System response**

	1. The use case starts when the administrator queries information on a user. (AC1)	2. The system displays the information it has on the user, including groups and the list of applications that the user can operate.
	3. The administrator selects a new application that the user can operate and asks for it to be added to the list of permits. (AC2)	4. The system stores the information and updates the permits database.
Alternative course 1		
	1.1 The use case starts when the administrator queries information on a group.	2.1 The system displays the information it has on the group, including the list of applications that the group can operate.
	3.1 The administrator selects a new application that the group can operate and asks for it to be added to the list of permits. (AC3)	4.1 The system stores the information and updates the permits database.
Alternative course 2		
	3.2 The administrator selects a new application that the user can operate and asks for it to be deleted from the list of permits.	4.2 The system stores the information and updates the permits database.
Alternative course 3		
	3.3 The administrator selects a new application that the group can operate and asks for it to be deleted from the list of permits.	4.3 The system stores the information and updates the permits database.

E.4.1.2.2 Use Case: Administer applications and menu

Actor	Administrator
Type	Primary and essential

Purpose	Maintain the applications, application information and structure of the menu to be displayed to the user.	
Overview:	The administrator adds elements to the menu. These elements may be nodes or leaves. The leaves correspond to application information. The nodes organise the applications.	
References:	*Req* S1, S2, S3, S4, S5	
	Actor action	**System response**
Typical course of events	1 The use case starts when the administrator decides to enter an element in the menu.	2 The system displays the possible options: node or leaf.
	3 The administrator decides to enter a node. (AC1).	4 The system displays the fields to be filled and the list of possible parents for association.
	5 The administrator selects the parent, enters the information and asks to save the information.	6 The system saves the information.
Alternative course 1		
	3.1 The administrator decides to enter an application.	4.1 The course continues as in steps 4 to 6 of the typical course.

E.4.1.2.3 Use Case: Update contracts

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Search information on the new contracts entered in Phoenix	
Overview:	The actor asks the system to update the information on contracts, the system searches the related contract information in the Phoenix database and updates this information in the system database.	
References:	*Req* P2, P3, P4, P6, P7, P8, P9	
Typical course	**Actor action**	**System response**

	1 The user asks for the information on contracts to be updated.	2 The system retrieves the information on the contracts and updates the database.

E.4.1.2.4 Use Case: Manage advertising group

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify or delete advertising group.	
Overview:	The administrator inserts the data on the advertising group to be added, modified or deleted and asks the system to execute the action.	
References:	*Req*: P5, P10	
	Actor action	**System response**
Typical course of events	1 The administrator fills in the data on the new advertising group and asks the system to create this new advertising group.	2 The system creates the new advertising group.
Alternative course 1	1.1 The administrator fills in the data on a new advertising group and asks the system to retrieve the information on this advertising group.	2.1 The system retrieves the information on the advertising group and displays this information, if any.
Alternative course 2	1.2 The administrator fills in the data required to identify the advertising group and asks the system to delete this group.	2.2 The system retrieves the information on the advertising group and deletes the group, if possible.
Alternative course 3	1.3 The administrator fills in the data required to retrieve the advertising group and asks the system to retrieve the information on this group.	2.3 The system retrieves the information on the advertising group and displays this information to the user.

	2.3 The administrator modifies the information on the advertising group and asks the system to save the information.	2.3 The system updates the information on the advertising group.

E.4.1.2.5 Use Case: Manage advertiser

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify or delete advertiser.	
Overview:	The administrator enters the data on the advertiser to be added, modified or deleted and asks the system to execute the action.	
References:	*Re:* P5, P10	
	Actor action	**System response**
Typical course of events	1 The administrator fills in the data on the new advertiser and asks the system to create this new advertiser.	2 The system creates the advertiser.
Alternative course 1	1.1 The administrator fills in the data on a new advertiser and asks the system to retrieve the information on this advertiser.	2.1 The system retrieves the information on the advertiser and displays this information, if any.
Alternative course 2	1.2 The administrator fills in the data required to identify the advertiser and asks the system to delete this advertiser.	2.2 The system retrieves the information on the advertiser and deletes the advertiser, if possible.
Alternative course 3	1.3 The administrator fills in the data required to retrieve the advertiser and asks the system to retrieve the information on this advertiser.	2.3 The system retrieves the information on the advertiser and displays this information to the user.
	3.3 The administrator modifies the information on the advertiser and asks the system to save the information.	4.3 The system updates the information on the advertiser.

E.4.1.2.6 Use Case: Manage product category

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify or delete a product category.	
Overview:	The administrator enters the data on the product category to be added, modified or deleted and asks the system to execute the action.	
References:	*Req:* P5, P10	
Typical course of events	**Actor action**	**System response**
	1 The administrator fills in the data on the new product category and asks the system to create this new product category.	2 The system creates the new product category.
Alternative course 1	1.1 The administrator fills in the data on a new product category and asks the system to retrieve the information on this product category.	2.1 The system retrieves the information on the product category and displays this information, if any.
Alternative course 2	1.2 The administrator fills in the data required to identify the product category and asks the system to delete this category.	2.2 The system retrieves the information on the product category and deletes it, if possible.
Alternative course 3	1.3 The administrator fills in the data required to retrieve the product category and asks the system to retrieve the information on this category.	2.3 The system retrieves the information on the product category and displays this information to the user.
	3.3 The administrator modifies the information on the product category and asks the system to save the information.	4.3 The system updates the information on the product category.

E.4.1.2.7 Use Case: Manage product

Actor	Marketing_administrator

Type	Primary and essential	
Purpose	Create, modify, delete product.	
Overview:	The administrator enters the data on the product to be added, modified or deleted and asks the system to execute the action	
References:	*Req:* P5, P10	
	Actor action	**System response**
Typical course of events	1 The administrator fills in the data on the new product and asks the system to create this new product.	2 The system creates the new product.
Alternative course 1	1.1 The administrator fills in the data on a new product and asks the system to retrieve the information on this product.	2.1 The system retrieves the information on the product and displays this information, if any.
Alternative course 2	1.2 The administrator fills in the data required to identify the advertiser and asks the system to delete this product.	2.2 The system retrieves the information on the product and deletes it, if possible.
Alternative course 3	1.3 The administrator fills in the data required to retrieve the product and asks the system to retrieve the information on this product.	2.3 The system retrieves the information on the product and displays this information to the user.
	3.3 The administrator modifies the information on the product and asks the system to save the information.	4.3 The system updates the information on the product.

E.4.1.2.8 Use Case: Manage campaign

Actor	Marketing_administrator
Type	Primary and essential
Purpose	Create, modify or delete a campaign.
Overview:	The administrator enters the data on the campaign to be added, modified or deleted and asks the system to execute the action
References:	*Req*: P5, P10

	Actor action	System response
Typical course of events	1. The administrator fills in the data on the new campaign and asks the system to create this new campaign.	2. The system creates the new campaign.
Alternative course 1	1.1 The administrator fills in the data on a new campaign and asks the system to retrieve the information on this campaign.	2.1 The system retrieves the information on the campaign and displays this information, if any.
Alternative course 2	1.2 The administrator fills in the data required to identify the advertiser and asks the system to delete this campaign.	2.2 The system retrieves the information on the campaign and deletes it if possible.
Alternative course 3	1.3 The administrator fills in the data required to retrieve the campaign and asks the system to retrieve the information on this campaign.	2.3 The system retrieves the information on the campaign and displays this information to the user.
	3.3 The administrator modifies the information on the campaign and asks the system to save the information.	4.3 The system updates the information on the campaign.

E.4.1.2.9 Use Case: Query campaign

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a campaign with the information requested by marketing.	
Overview:	The user selects the code of the campaign to be queried and asks the system for the information.	
References:	*Req*: P5, P10	
Typical course	**Actor action**	**System response**

	1 The user fills in the data required to identify the campaign and asks the system to display the information.	2 The system retrieves the information on the campaign and displays it to the user.

E.4.1.2.10 Use Case: Manage contract

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify or delete contract.	
Overview:	The administrator enters the data on the contract to be added, modified or deleted and asks the system to execute the action.	
References:	*Req* P5, P10	
Typical course of events	**Actor action**	**System response**
	1 The administrator fills in the data required to retrieve the contract and asks the system to retrieve the information on this contract.	2 The system retrieves the information on the contract and displays this information to the user.
	3 The administrator modifies the information on the contract and asks the system to save the information.	4 The system updates the information on the contract.

E.4.1.2.11 Use Case: Query contract

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a contract with the information requested by marketing.	
Overview:	The user selects the code of the contract to be queried and asks the system for the information.	
References:	*Req* P5, P10	
Typical course of events	**Actor action**	**System response**

	1 The user fills in the data required to identify the contract and asks the system to display the information.	2 The system retrieves the information on the contract and displays it to the user.

E.4.1.2.12 Use Case: Query booking

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a booking with the information requested by marketing.	
Overview:	The user selects the code of the booking to be queried and asks the system for the information.	
References:	*Req* P5, P10	
Typical course of events	**Actor action**	**System response**
	1 The user fills in the data required to identify the booking and asks the system to display the information.	2 The system retrieves the information on the booking and displays it to the user.

E.4.1.2.13 Use Case: Query customer

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a customer with the information requested by marketing.	
Overview:	The user selects the code of the customer to be queried and asks the system for the information.	
References:	*Req* P5, P10	
Typical course of events	**Actor action**	**System response**
	1 The user fills in the data required to identify the customer and asks the system to display the information.	2 The system retrieves the information on the customer and displays it to the user.

E.4.1.2.14 Use Case: Query agency

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on an agency with the information requested by marketing.	
Overview:	The user selects the code of the agency to be queried and asks the system for the information.	
References:	*Req* P5, P10	
Typical course of events	**Actor action**	**System response**
	1 The user fills in the data required to identify the agency and asks the system to display the information.	2 The system retrieves the information on the agency and displays it to the user.

E.4.1.2.15 Use Case: Get contract report

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Get a report on contracts that match given criteria by means of a query.	
Overview:	The user enters the search parameters and asks the system for the information.	
References:	*Req:* P1, P2, P10	
	Actor action	**System response**
Typical course of events	1 The actor fills in the parameters required for the query to be made (accounting period, advertiser, products, product groups) and asks the system for the report.	2 The system displays the contracts and their related information.

E.4.1.2.16 Use Case: Export report

Actor	Marketing_administrator, Marketing_user
Type	Primary and essential
Purpose	Export a generated to a file with Microsoft Excel format

Overview:	Having requested a report, the user asks the system to export the file data to an Excel file.	
References:	*Req:* P12	
Typical course of events	**Actor action**	**System response**
	1 The use case starts when the actor has requested a report and asks the system to export the report to Excel along a given route.	2 The system generates the report.

E.4.1.2.17 Use Case: Query Monitor planning

Actor	Sales	
Type	Primary and essential	
Purpose	View the information on Monitor program bookings, booking details and distribution by weeks, towns and circuits as similarly as possible to the wall planning now used in the department.	
Overview:	The actor selects the distribution search criteria and the system displays the information. The user can view the information on a booking in more detail.	
References:	Req: N1 to N7	
	Actor action	**System response**
Typical course of events	1 The use case starts when the user selects the campaign bookings search parameters.	2 The system displays the bookings that match the search criteria.
	3 The user selects a specific week for a circuit to view the details of the bookings in the week in question.	4 The system displays the information on the bookings for the week in question.

E.4.1.2.18 Use Case: Query affixation job tickets

Actor	Technical
Type	Primary and essential
Purpose	Retrieve the information on the tickets belonging to an affixation job.

Overview:	The actor enters the data that identify the affixation job and asks the system to retrieve the related tickets.	
References:	*Req* R3, R5	
	Actor action	**System response**
Typical course of events	1 The actor enters the query parameters for the affixation job tickets and asks the system to retrieve the information.	2 The system retrieves the information on the tickets and displays it to the user.

E.4.1.2.19 Use Case: Change ticket status

Actor	Technical	
Type	Primary and essential	
Purpose	Modify the status of a ticket or group of tickets.	
Overview:	The actor selects the tickets whose status is to be changed and asks the system to change their status.	
References:	*Req*: R1	
	Actor action	**System response**
Typical course of events	1. The actor selects a ticket or series of tickets and asks the system to change their status.	2. The system checks that the selected tickets belong to the actor's depot and changes their status.

E.4.1.2.20 Use Case: Query affixation job

Actor	Technical	
Type	Primary and essential	
Purpose	View the information on an affixation job.	
Overview:	The actor enters the search parameters for the affixation job to be queried and asks the system to display this information.	
References:	*Req:* R5	
Typical course	**Actor action**	**System response**

	1. The actor enters the query parameters for the affixation job and asks the system to retrieve the information.	2. The system retrieves the information on the affixation job, which it displays to the user.

E.4.1.2.21 Use Case: Change affixation job data

Actor	Technical	
Type	Primary and essential	
Purpose	Modify the information on an affixation job.	
Overview:	The administrator enters the data on the affixation job to be modified and asks the system to save the information.	
References:	*Req:* R2	
Typical course of events	**Actor action**	**System response**
	1. The actor modifies the information on the job and asks the system to save the information.	2. The system updates the information on the job.

E.4.1.2.22 Use Case: Request affixation job report

Actor	Technical	
Type	Primary and essential	
Purpose	Get a report on the affixation status.	
Overview:	The administrator enters the search parameters for the affixation job on which the report is required.	
References:	*Req:* R6, R4	
	Actor action	**System response**
Typical course of events	1 The actor fills in the parameters required for the query to be made (booking, period or job code) and asks the system for the report.	2 The system displays the affixation job report.

E.4.1.2.23 Use Case: Query prohibitions

Actor	Technical

Type	Primary and essential	
Purpose	Ask the system for information on the advertising signs that have prohibitions.	
Overview:	The actor selects the criteria to be queried such as prohibition type, municipality, region, site, and the system displays the information to the user, which the user can then ask to be exported to Excel.	
References:	Req: I1 a I5	
	Actor action	**System response**
Typical course of events	1. The use case starts when the user selects the search criteria and asks the system for the information.	2. The system searches for the information that matches the selected criteria and displays the information.
	3. The actor asks to export the information to Microsoft Excel.	4. The system prepares the file with the requested information and returns it to the actor.

E.4.1.2.24 Use Case: Load interface file

Actor	Invoicing, Marketing	
Type	Primary and essential	
Purpose	Load a previously generated interface file to its respective tables.	
Overview:	The user selects the interface file and file type, and asks the system to load the file.	
References:	Req: M1, J3	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects a file and loading type and asks the system to load the file.	2. The system validates the file structure based on the selected type. If the structure is correct (AC1), the system loads the file.

Alternative course 1		2.1 If the file structure is NOT correct, the user is informed of this error.
	3.1 The use case continues as in step 1.	

E.4.1.2.25 Use Case: Generate audience profitability report

Actor	Marketing	
Type	Primary and essential	
Purpose	Generate a report that consolidates the rental payment information generated by SAIB, the sales information generated by PHOENIX and the audience information received from an external consultant to evaluate asset quality and profitability.	
Overview:	The user has previously loaded the information from the interface files on which he is working (see use case "Load interface file"). The user selects the files to be used, the PHOENIX period against which the data are to be compared and asks to generate the report. The system returns the information.	
References:	Req: J1 to J4	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects the files and period that he intends to use and asks the system to generate the report.	2. The system searches for the information that matches the selected criteria and displays the information.
	3. The actor asks the system to transfer the report in Excel.	4. The system returns the Excel file with the requested data.

E.4.1.2.26 Use Case: Generate campaign hoardings report

Actor	Sales
Type	Primary and essential
Purpose	Generate a report with the campaign hoardings.
Overview:	The user selects a booking or contract and a period, the system displays the hoardings.

References:	Req: J1 to J4	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects the booking or contract to be queried and the period.	2. The system searches for the information that matches the selected criteria and displays the information.
	3. The actor asks the system to transfer the report in Excel.	4. The system returns the Excel file with the requested data

E.4.1.2.27 Use Case: Generate Phoenix-Siges interface

Actor	Invoicing	
Type	Primary and essential	
Purpose	Generate an interface file that contains the PHOENIX invoicing data for a specific accounting period.	
Overview:	The actor selects the accounting period and asks the system to generate the report. The system checks that there are no inconsistencies between the Phoenix application data and the SIGES application. If everything is in order, the system generates and the user receives the report. If there are any problems, the system displays the problems and asks for them to be corrected.	
References:	Req: L1 to L5	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects the accounting period for which the interface is to be generated and asks the system to generate the interface file.	2. The system validates the data. If everything is correct, it generates the report. (AC1).
		3. The system displays the report for downloading.
Alternative		

		2.1 If there are any inconsistencies between the two applications, the system reports the inconsistencies for correction in the applications.

E.4.1.2.28 Use Case: Load SAIB – Siges interface

Actor	Invoicing	
Type	Primary and essential	
Purpose	Insert the SAIB supplier payment interface into the SIGES application, after validation.	
Overview:	Once the file has been transferred from the SAIB application, the file is selected, loaded and validated. If it is correct, it is added to the SIGES DB.	
References:	Req: M1 to M3	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects the file to be loaded and asks the system to add the file.	2. The system validates the file structure and that it has not been previously loaded.
		3. If everything is in order it adds the information to the SAIB DB and informs the user (AC1)
Alternative course 1		
		3.1 If there are errors in the file structure, the user is informed.

E.4.1.2.29 Use Case: Change Phoenix bookings status

Actor	Marketing, Sales, SOS, Assets management, Technical
Type	Primary and essential
Purpose	Directly change, without going through the PHOENIX application, the status of one or more invoices from B (Blocked) to I(Invoiced).

Overview:	The user selects one or more bookings and asks the system to change the status from B to I or from I to B.	
References:	Req: O1, O2	
	Actor action	**System response**
Typical course of events		1. The system displays the bookings grouped by status.
	2. The actor selects the bookings of one type and asks the system to change the type.	4. The system updates the information and displays the new status.

E.4.1.2.30 Use Case: Generate Phoenix manual invoicing report

Actor	Invoicing	
Type	Primary and essential	
Purpose	Generate a report of the manual invoices generated for an accounting period.	
Overview:	The actor selects the accounting period and asks the system to generate the report.	
References:	Req: C1 to C4	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects an accounting period and asks the system to generate the report.	2. The system retrieves the information based on the selection criteria and displays the information.

E.4.1.2.31 Use Case: Update data

Actor	Sales, Invoicing
Type	Primary and essential
Purpose	Get the information that will be needed in the DB to generate reports from external applications.
Overview:	The actor asks for the information to be loaded to the database and the system updates the working DB
References:	This is done so as not to affect application DB performance.

	Actor action	System response
Typical course of events	1. The use case starts when the actor asks for the information to be updated.	2. The system loads the information into the working DB.

E.4.1.2.32 Use Case: Generate outdoor assets report

Actor	SOS	
Type	Primary and essential	
Purpose	Generate a report on the outdoor assets distribution by type for a given period.	
Overview:	The user can query the advertising signs distribution by type for a given period.	
References:	Req: D1 to D5.	
	Actor action	**System response**
Typical course of events	1. The use case starts when the user selects a PHOENIX period and asks the system for a visualisation.	2. The system checks whether this report has already been generated, in which case it returns it immediately. (AC1)
Alternative course 1		2.1 If the report has not been generated, the system informs the user that it is processing the report.
		3.1 Once it has been processed, the system returns the generated report.

E.4.1.2.33 Use Case: Generate exclusives assets report

Actor	SOS
Type	Primary and essential
Purpose	Generate a report on the distribution of exclusives assets by type for a given period.

Overview:	The user can query the distribution of advertising signs by type for a given period.	
References:	Req: D1 a D5.	
	Actor action	**System response**
Typical course of events	1. The use case starts when the user selects a PHOENIX period and asks the system for a visualisation.	2. The system checks whether this report has already been generated, in which case it returns it immediately. (AC1)
Alternative course 1		2.1 If the report has not been generated, the system informs the user that it is processing the report.
		3.1 Once it has been processed, the system returns the generated report.

E.4.1.2.34 Use Case: Generate average painting price report

Actor	SOS	
Type	Primary and essential	
Purpose	Generate a report on the average price of painted hoardings for a given period.	
Overview:	The user selects a PHOENIX period and asks for the report. The system calculates the information and displays the report.	
References:	Req: F1 to F3	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects a PHOENIX period and asks for the report.	2. The system makes the calculation on the basis of the input parameters and returns the information.

E.4.1.2.35 Use Case: Generate objectives report

Actor	Assets management	
Type	Primary and essential	
Purpose	Generate a report that shows the deviations between the established objectives and the current status of the assets.	
Overview:	The actor should parameterise the objectives by circuit. The actor requests the objectives report and the system calculates and displays the data.	
References:	Req: H1 to H6	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor decides to modify the defined objectives (AC1)	2. The system retrieves and displays the information for the actor.
	3. The actor makes the modifications and asks to save the data	4. The system saves the information in the DB.
Alternative course 1	1.1 The actor decides to query the deviations between the objectives and the assets.	2.1 The system retrieves and displays the information.
	3.1 The actor asks to export the information to Excel.	4.1 The system returns the Excel file containing the requested data.

E.4.1.2.36 Use Case: Administer manuals

Actor	Content manager	
Type	Primary and essential	
Purpose	Enter manuals in the database with the information required for later search.	
Overview:	The actor requests the information on the inserted manuals, as this information can be modified or deleted or a new manual can be entered.	
References:	Req: Q1 a Q4	
	Actor action	**System response**

Typical course of events	1. The use case starts when the actor decides to insert the information on a manual.(AC 1, AC2)	2. The system displays the fields to be filled in for this manual.
	3. The actor fills in the fields and asks to save the information.	4. The system saves the information.
Alternative course 1	1.1 The actor asks to query all the manuals.	2.1 The system displays all the manuals.
	3.1 The actor selects a manual.	4.1 The system displays the information on the selected manual.
	5.1 The actor modifies the information and asks the system to save the information.	6.1 The system saves the information.
Alternative course 2	1.2 The actor asks to query all the manuals.	2.2 The system displays all the manuals.
	3.2 The actor asks the system to delete a manual.	4.2 The system deletes the manual.

E.4.1.2.37 Use Case: Query manuals

Actor	Marketing, Sales, Invoicing, SOS, Assets management, Technical	
Type	Primary and essential	
Purpose	Query the information on the inserted manuals according to a series of criteria.	
Overview:	The actor enters the selection criteria, the system displays the manuals that match these criteria, and the actor selects the desired manual and views its information.	
References:	Req: Q1 to Q4	
	Actor action	**System response**

Typical course of events	1. The use case starts when the actor enters the selection criteria and asks the system to retrieve the information.	2. The system lists the manuals that match the criteria.
	3. The actor selects the manual in which he is interested from the list.	4. The system displays the detailed information on the manual.
	5. The actor asks to download the manual.(AC1)	
Alternative course 1		
	5.1 The actor enters a remark on the manual and asks to save the remark.	6.1 The system saves the information.

E.4.1.2.38 Use Case: Generate Geomex assets status report

Actor	Marketing	
Type	Primary and essential	
Purpose	Generate a report containing information on the status of GEOMEX assets with respect to PHOENIX.	
Overview:	The user asks the system to output the status of the assets between two periods. The system returns the information on any additions, disposals and modifications	
References:	Req: B1 to B6	
	Actor action	**System response**
Typical course of events	1. The use case starts when the actor selects the starting and ending period and asks for the information.	2. The system retrieves and displays the information.
	3 The actor asks to export the information to Excel.	4 The system returns the Excel file containing the requested data.

E.4.1.3 Sequence diagrams

These diagrams describe the interaction between the actors and the system for each use case, both for the typical course of events and for the major alternative courses.

E.4.1.3.1 Sequence Diagram: Administer users

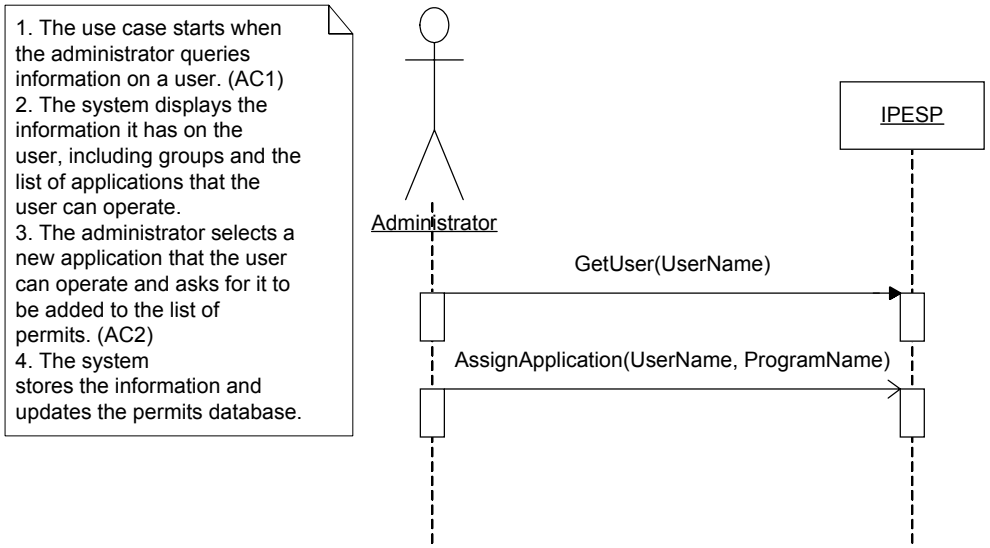

Figure 14 Sequence Diagram: Administer users

E.4.1.3.2 Sequence Diagram: Administer applications

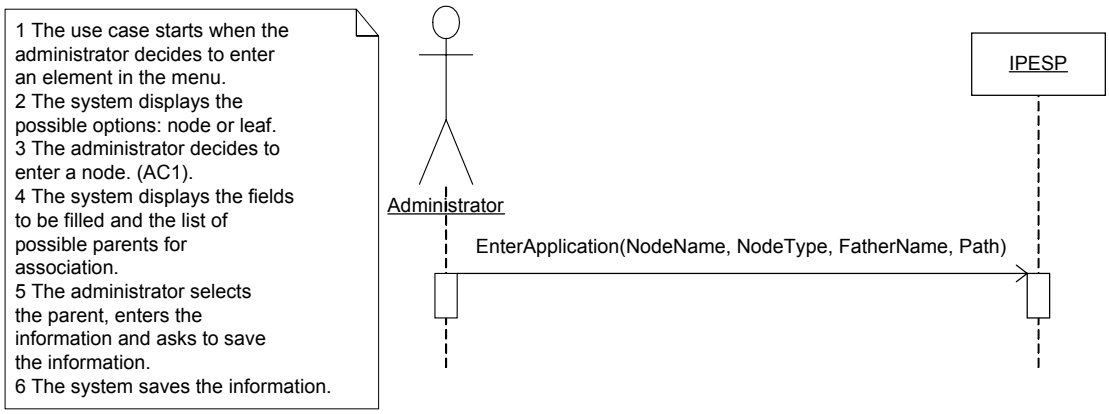

Figure 15 Sequence Diagram: Administer Applications

E.4.1.3.3 Sequence Diagram: Update contracts

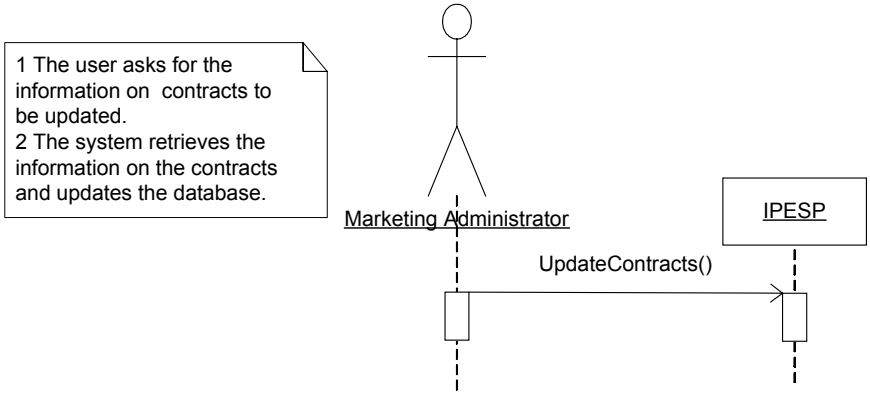

Figure 16 Sequence Diagram: Update Contracts

E.4.1.3.4 Sequence Diagram: Manage advertising group

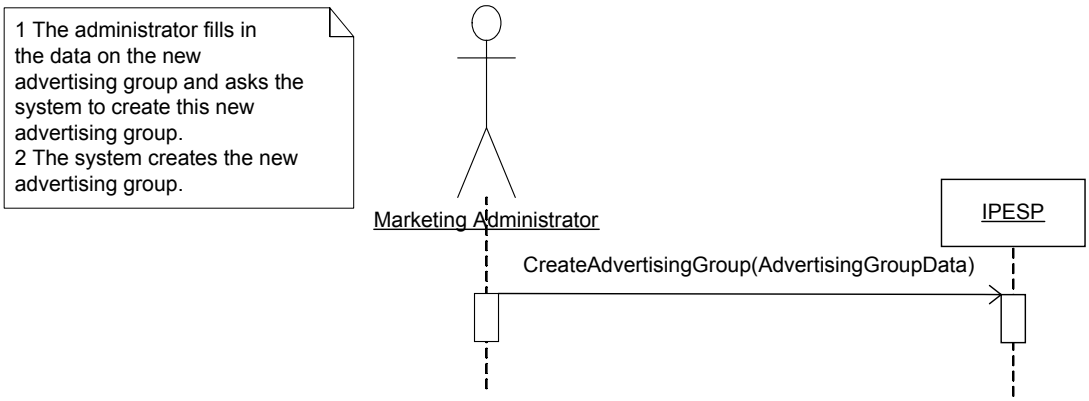

Figure 17 Sequence Diagram: Manage advertising group

E.4.1.3.5 Sequence Diagram: Manage advertiser

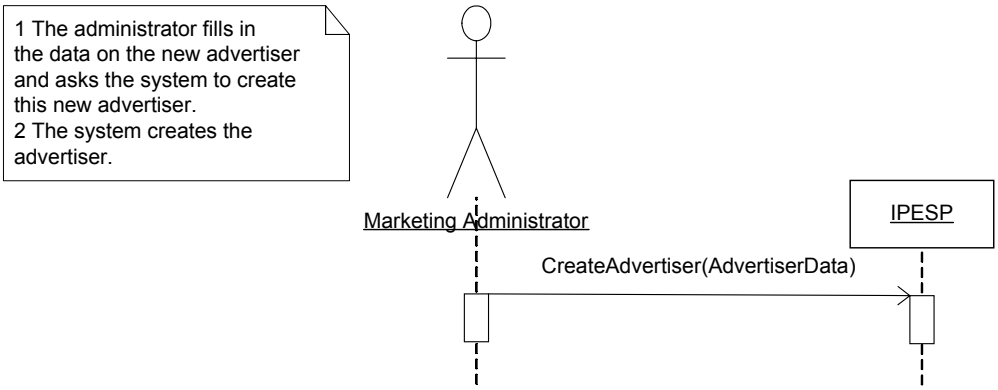

Figure 18 Sequence Diagram: Manage advertiser

E.4.1.3.6 Sequence Diagram: Manage product category

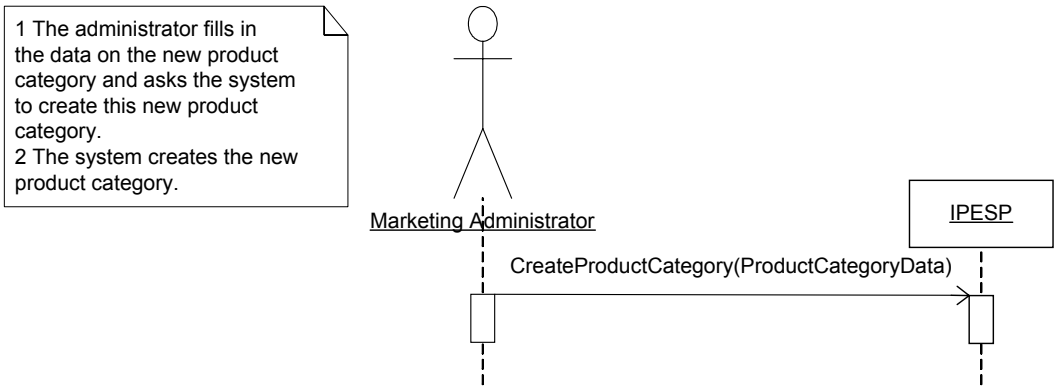

Figure 19 Sequence Diagram: Manage product category

E.4.1.3.7 Sequence Diagram: Manage product

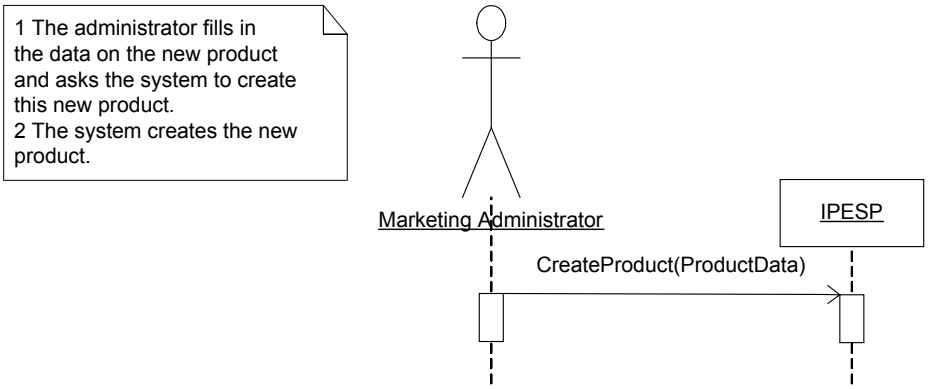

Figure 20 Sequence Diagram: Manage product

E.4.1.3.8 Sequence Diagram: Manage campaign

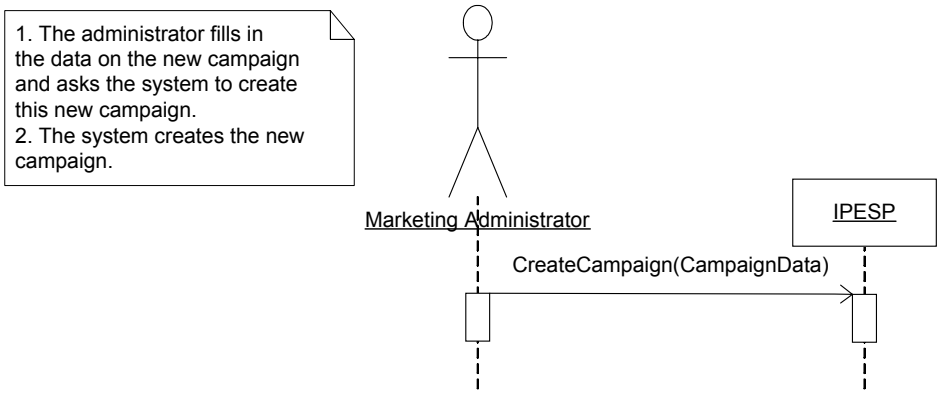

Figure 21 Sequence Diagram: Manage campaign

E.4.1.3.9 Sequence Diagram: Query campaign

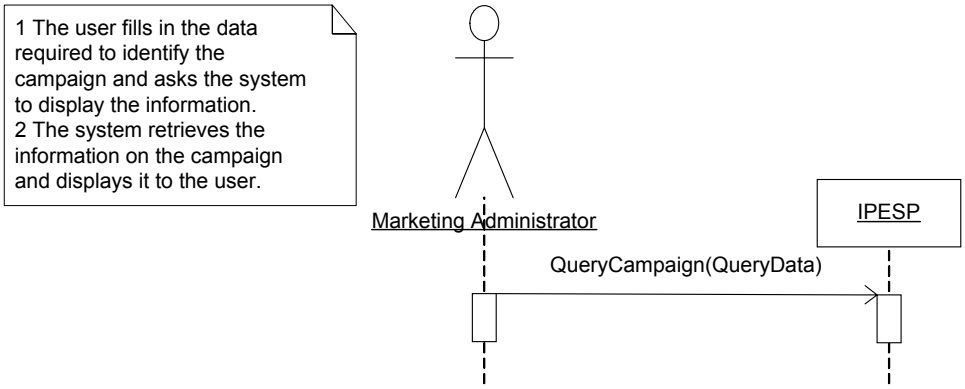

Figure 22 Sequence Diagram: Query campaign

E.4.1.3.10 Sequence Diagram: Manage contract

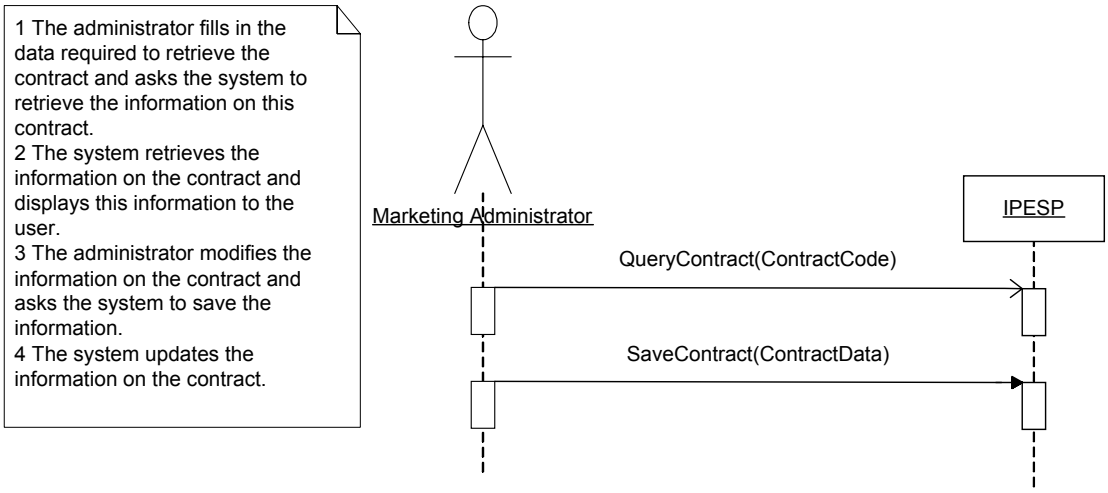

Figure 23 Sequence Diagram: Manage contract

E.4.1.3.11 Sequence Diagram: Query contract

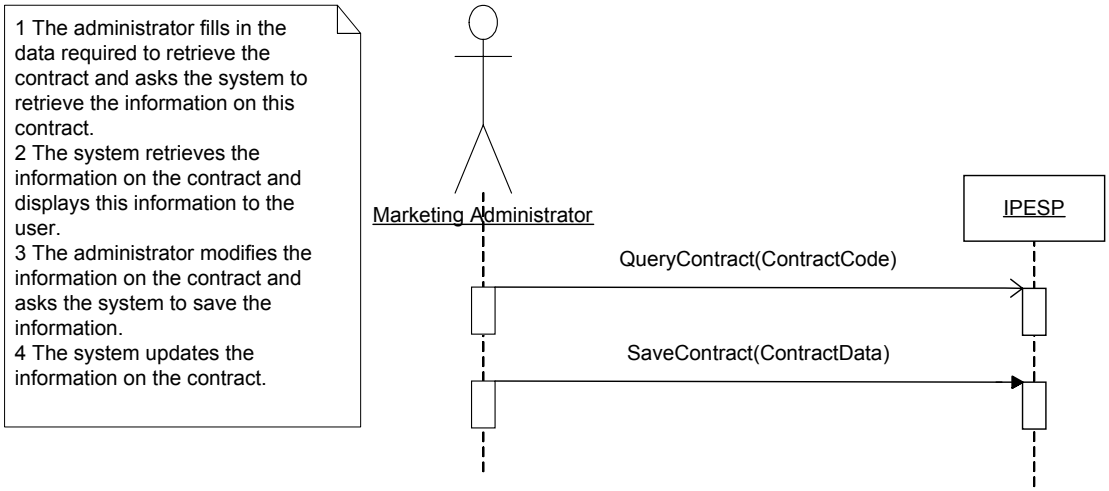

Figure 24 Sequence Diagram: Query contract

E.4.1.3.12 Sequence Diagram: Query booking

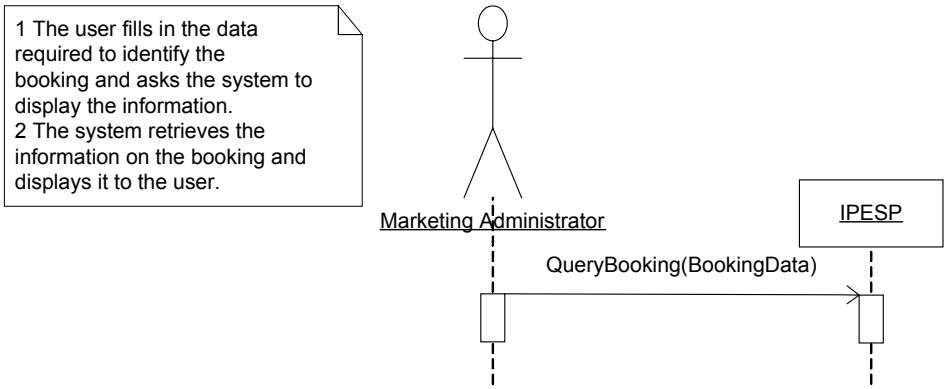

Figure 25 Sequence Diagram: Query booking

E.4.1.3.13 Sequence Diagram: Query customer

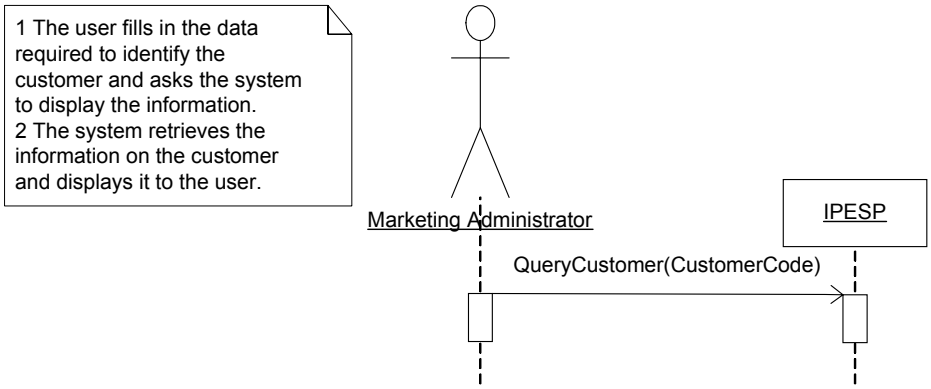

Figure 26 Sequence Diagram: Query customer

E.4.1.3.14 Sequence Diagram: Query agency

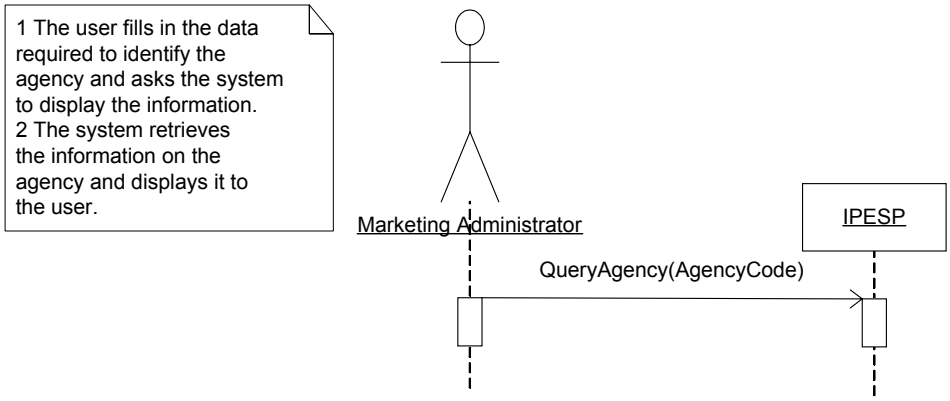

Figure 27 Sequence Diagram: Query agency

E.4.1.3.15 Sequence Diagram: Get contract report

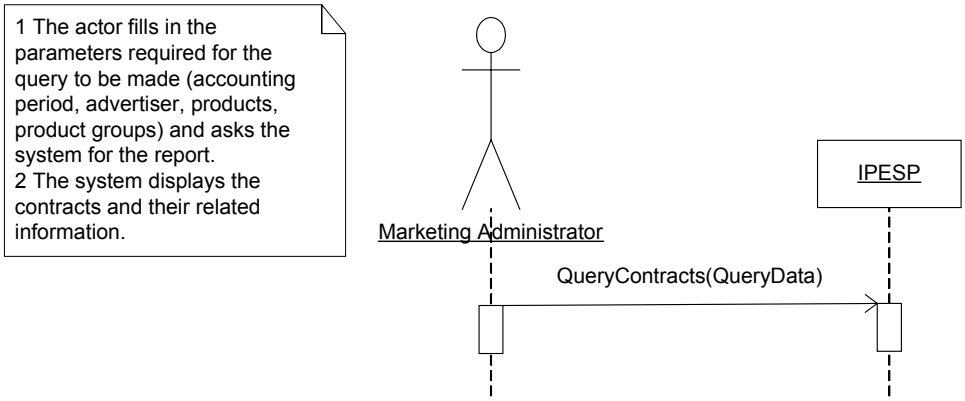

Figure 28 Sequence Diagram: Get contract report

E.4.1.3.16 Sequence Diagram: Export report

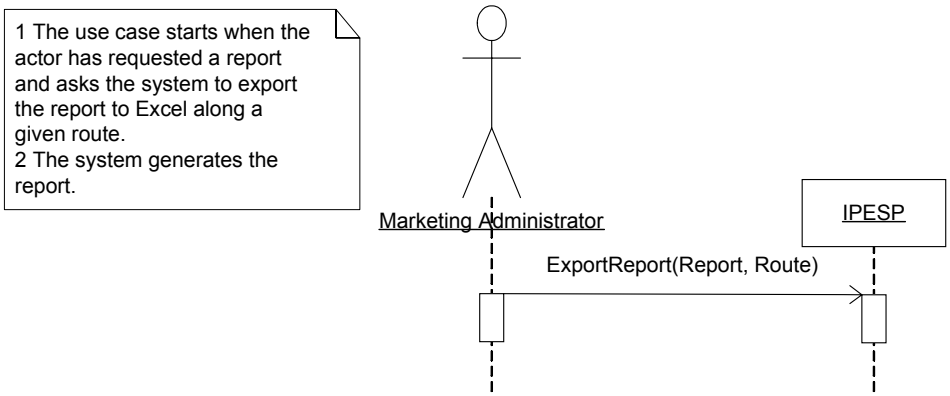

Figure 29 Sequence Diagram: Export report

E.4.1.3.17 Sequence Diagram: Query monitor planning

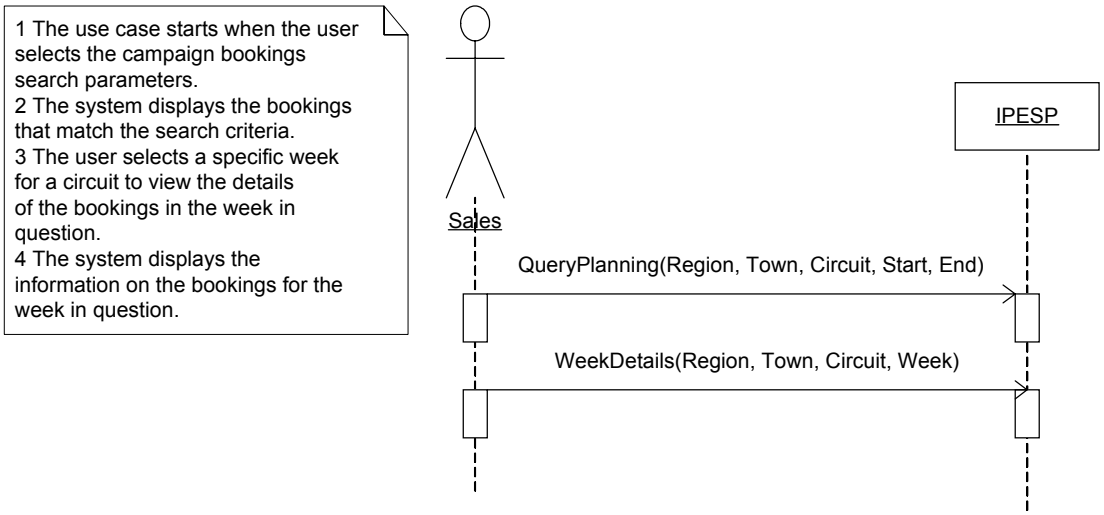

Figure 30 Sequence Diagram: Query planning monitor

E.4.1.3.18 Sequence Diagram: Query affixation job tickets

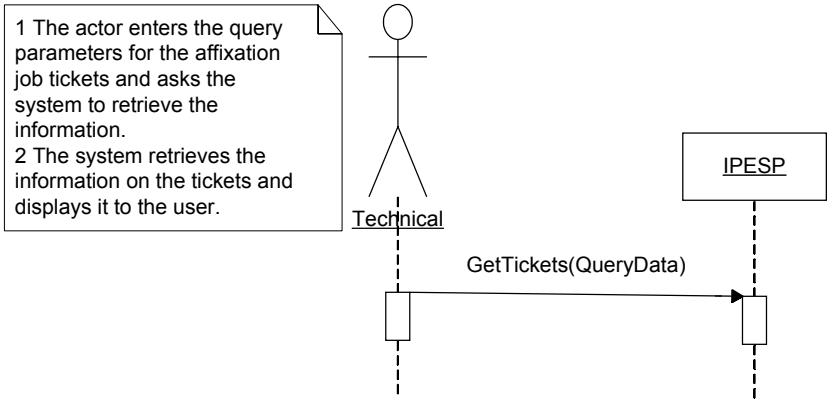

Figure 31 Sequence Diagram: Query affixation job tickets

E.4.1.3.19 Sequence Diagram: Change ticket status

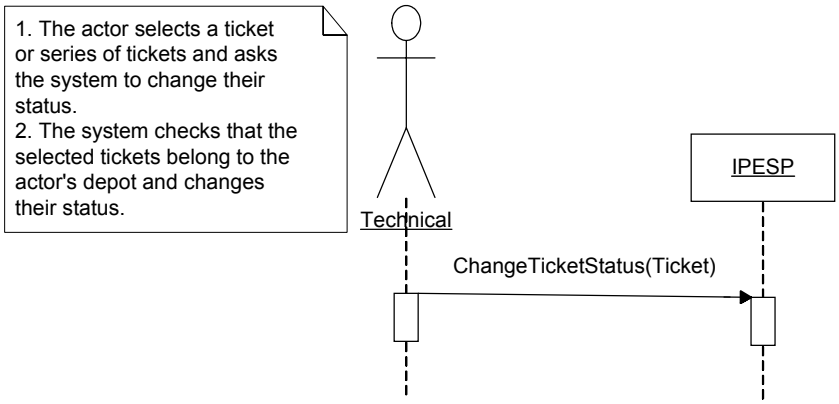

Figure 32 Sequence Diagram: Change ticket status

E.4.1.3.20 Sequence Diagram: Query affixation job

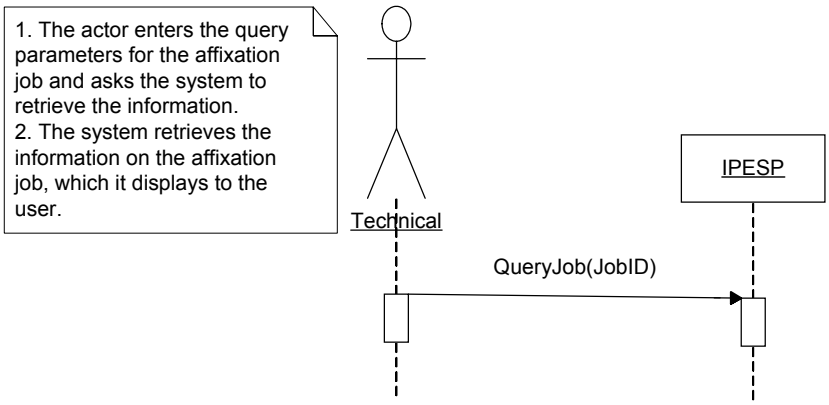

Figure 33 Sequence Diagram: Query affixation job

E.4.1.3.21 Sequence Diagram: Change affixation job data

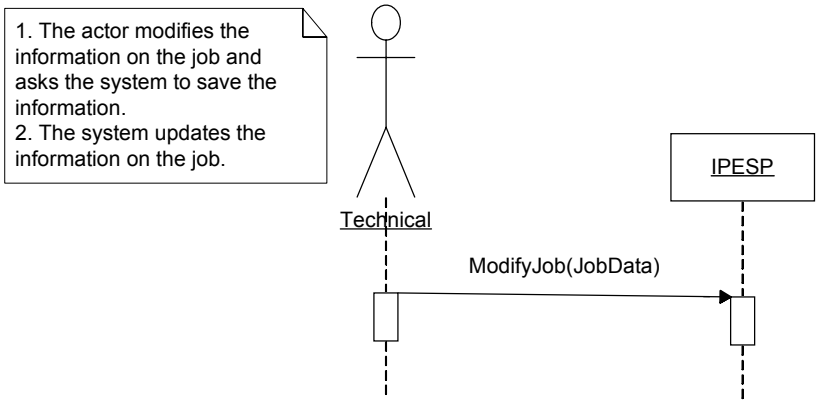

Figure 34 Sequence Diagram: Change affixation job data

E.4.1.3.22 Sequence Diagram: Get affixation job report

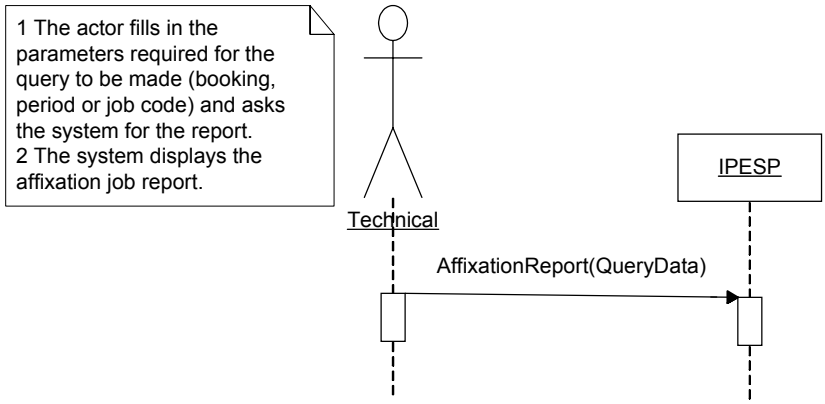

Figure 35 Sequence Diagram: Get affixation job report

E.4.1.3.23 Sequence Diagram: Query prohibitions

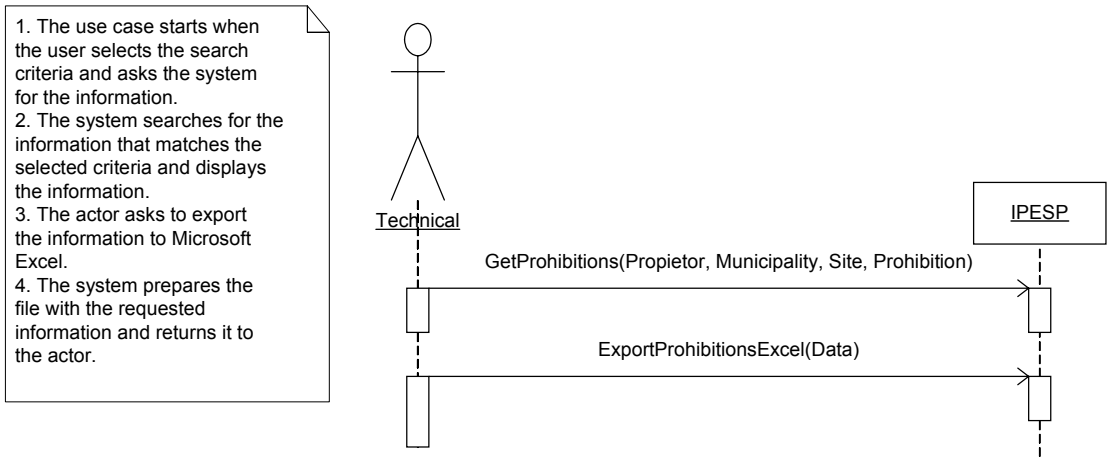

Figure 36 Sequence Diagram: Query prohibitions

E.4.1.4 Operations Contracts

The operations contracts specify each of the operations presented in the sequence diagrams and the system behaviour in more detail. The following information can be defined for each contract: name, responsibilities, type, cross-references, notes, exceptions, output, preconditions and postconditions, although not all the fields are compulsory.

E.4.1.4.1 Sequence diagram contracts: Administer users

This sequence diagram contains the following contracts:

- GetUser.
- AssignApplication

Name:	***GetUser(Name:String)***
Responsibilities:	Retrieve information on a user.
Cross-references:	Requirements A5 and A6

	Use case: Administer users.
Notes:	
Exceptions:	The user does not exist.
Output:	
Preconditions:	
Postconditions:	The information on the user has been retrieved and is listed for modification.

Name:	***AssignApplication(UserName:String, ProgramName:String)***
Responsibilities:	Assign permit for a user to access an application.
Cross-references:	Requirements A5 and A6 Use case: Administer users.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The permit for the user to access an application has been assigned.

E.4.1.4.2 Sequence diagram contracts: Administer applications and menu

This sequence diagram contains the following contracts:

- EnterApplication

Name:	***EnterApplication(NodeName:String, NodeType:String, ParentName:String, Path:String)***
Responsibilities:	Save the applications to be used in the web site and organise the applications hierarchically to build the application menu.
Cross-references:	Requirements: S1, S4, S5, S6 Use cases: Administer applications
Notes:	
Exceptions:	The defined route does not exist.

Output:	
Pre-conditions:	
Post-conditions:	The information on the application has been inserted.

E.4.1.4.3 Sequence diagram contracts: Update contracts

This sequence diagram contains the following contracts:

- UpdateContracts

Name:	***UpdateContracts()***
Responsibilities:	Retrieve information on contracts and read the Phoenix database to update this information.
Cross-references:	Requirements: P2, P3, P4, P6, P7, P8, P9 Use case: Update contracts
Notes:	
Exceptions:	No contracts exist.
Output:	
Pre-conditions:	
Post-conditions:	The information on contracts has been updated.

E.4.1.4.4 Sequence diagram contracts: Manage advertising group

This sequence diagram contains the following contracts:

- CreateAdvertisingGroup

Name:	***CreateAdvertisingGroup(AdvertisingGroupData:Object)***
Responsibilities:	Create a new advertising group
Cross-references:	Requirements: P5, P10 Use case: Manage advertising group
Notes:	
Exceptions:	The advertising group already exists
Output:	
Pre-conditions:	
Post-conditions:	The advertising group has been created

E.4.1.4.5 Sequence diagram contracts: Manage advertiser

This sequence diagram contains the following contracts:

- CreateAdvertiser

Name:	***CreateAdvertiser(AdvertiserData:Object)***
Responsibilities:	Create a new advertiser
Cross-references:	Requirements P5 Use case: Manage advertiser
Notes:	
Exceptions:	The advertiser already exists
Output:	
Pre-conditions:	
Post-conditions:	The advertiser has been created

E.4.1.4.6 Sequence diagram contracts: Manage product category

This sequence diagram contains the following contracts:

- CreateProductCategory

Name:	***CreateProductCategory(ProductCategoryData:Object)***
Responsibilities:	Create a new product category
Cross-references:	Requirements P5 Use case: Manage product category
Notes:	
Exceptions:	The product category already exists
Output:	
Pre-conditions:	
Post-conditions:	The requested product category has been created

E.4.1.4.7 Sequence diagram contracts: Manage product

This sequence diagram contains the following contracts:

- CreateProduct

Name:	***CreateProduct(ProductData:Object)***
Responsibilities:	Create a new product
Cross-references:	Requirements P5

	Use case: Manage product
Notes:	
Exceptions:	The product already exists
Output:	
Pre-conditions:	
Post-conditions:	The requested product has been created

E.4.1.4.8 Sequence diagram contracts: Manage campaign

This sequence diagram contains the following contracts:

- CreateCampaign

Name:	***CreateCampaign(CampaignData:Object)***
Responsibilities:	Create a new campaign
Cross-references:	Requirements P5 Use case: Manage campaign
Notes:	
Exceptions:	The campaign already exists
Output:	
Pre-conditions:	
Post-conditions:	The requested campaign has been created

E.4.1.4.9 Sequence diagram contracts: Query campaign

This sequence diagram contains the following contracts:

- QueryCampaign

Name:	***QueryCampaign(QueryData:Object)***
Responsibilities:	Retrieve and display data on a campaign
Cross-references:	Requirements P2, P5 Use case: Query campaign
Notes:	
Exceptions:	The campaign does not exist
Output:	

Pre-conditions:	
Post-conditions:	The campaign data have been retrieved and displayed

E.4.1.4.10 Sequence diagram contracts: Manage contract

This sequence diagram contains the following contracts:

- QueryContract
- SaveContract

Name:	***QueryContract(ContractCode:int)***
Responsibilities:	Retrieve and display data on a contract
Cross-references:	Requirements P2, P10 Use case: Query contract
Notes:	
Exceptions:	The contract does not exist
Output:	
Pre-conditions:	
Post-conditions:	The contract data have been retrieved and displayed

Name:	***SaveContract(ContractData:Object)***
Responsibilities:	Save data on a contract for persistence
Cross-references:	Requirements A5 and A6 Use case: Manage contract
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The contract data have been saved

E.4.1.4.11 Sequence diagram contracts: Query contract

This sequence diagram contains the following contracts:

- QueryContract

Defined above.

E.4.1.4.12 Sequence diagram contracts: Query booking

This sequence diagram contains the following contracts:

- QueryBooking

Name:	***QueryBooking(BookingData:int)***
Responsibilities:	Retrieve and display data on a booking
Cross-references:	Requirements A5 and A6 Use case: Query booking
Notes:	
Exceptions:	The booking does not exist
Output:	
Pre-conditions:	
Post-conditions:	The booking data have been retrieved and displayed

E.4.1.4.13 Sequence diagram contracts: Query customer

This sequence diagram contains the following contracts:

- QueryCustomer

Name:	***QueryCustomer(CustomerCode:int)***
Responsibilities:	Retrieve and display data on a customer
Cross-references:	Requirements A5 and A6 Use case: Query customer
Notes:	
Exceptions:	The customer does not exist
Output:	
Pre-conditions:	
Post-conditions:	The customer data have been retrieved and displayed

E.4.1.4.14 Sequence diagram contracts: Query agency

This sequence diagram contains the following contracts:

- QueryAgency

Name:	***QueryAgency(AgencyCode:int)***
Responsibilities:	Retrieve and display data on a agency

Cross-references:	Requirements A5 and A6 Use case: Query agency
Notes:	
Exceptions:	The agency does not exist
Output:	
Pre-conditions:	
Post-conditions:	The agency data have been retrieved and displayed

E.4.1.4.15 Sequence diagram contracts: Get contract report

This sequence diagram contains the following contracts:

- QueryContracts

Name:	***QueryContracts(QueryData:int)***
Responsibilities:	Retrieve contracts that match the query parameters and display the contracts to the actor with the required information
Cross-references:	Requirements A5 and A6 Use case: Get contract report
Notes:	
Exceptions:	There are no contracts that match the query parameters
Output:	
Pre-conditions:	
Post-conditions:	The contracts have been retrieved and displayed to the actor

E.4.1.4.16 Sequence diagram contracts: Export report

This sequence diagram contains the following contracts:

- ExportReport

Name:	***ExportReport(Report:Object, Route:String)***
Responsibilities:	Export a contract report to a Microsoft Excel format file
Cross-references:	Requirements A5 and A6 Use case: Get contract report
Notes:	

Exceptions:	The route is invalid or cannot be written to
Output:	
Pre-conditions:	The report has been generated
Post-conditions:	A new instance of the application has been created

E.4.1.4.17 Sequence diagram contracts: Query Monitor planning

This sequence diagram contains the following contracts:

- QueryPlanning
- WeekDetails

Name:	***QueryPlanning(Region:String, Town:String, Circuit:String, Start:String, End:String)***
Responsibilities:	Retrieve and display the requested planning information according to search parameters.
Cross-references:	Requirements: N1, N2, N3, N4 Use case: Classify bookings by marketing category
Notes:	
Exceptions:	There are no bookings that match the selected criteria.
Output:	
Pre-conditions:	
Post-conditions:	The requested information is returned.

Name:	***WeekDetails(Region:String, Town:String, Circuit:String, Week:String)***
Responsibilities:	Retrieve the information on the bookings according to the specified search parameters.
Cross-references:	Requirements: N5 Use case: Administer applications
Notes:	
Exceptions:	
Output:	

Pre-conditions:	
Post-conditions:	The bookings information has been retrieved for the requested period.

E.4.1.4.18 Sequence diagram contracts: Query affixation job tickets

This sequence diagram contains the following contracts:

- GetTickets

Name:	***GetTickets(QueryData: Object)***
Responsibilities:	Retrieve the tickets for an affixation job.
Cross-references:	Requirements: R3, R5 Use case: Query affixation job tickets
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The tickets for the affixation job have been retrieved.

E.4.1.4.19 Sequence diagram contracts: Change ticket status

This sequence diagram contains the following contracts:

- ChangeTicketStatus

Name:	***ChangeTicketStatus(ticket: Object)***
Responsibilities:	Change ticket status.
Cross-references:	Requirements: R1 Use case: Change ticket status
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The status of the selected ticket has been changed.

E.4.1.4.20 Sequence diagram contracts: Query affixation job

This sequence diagram contains the following contracts:

- QueryJob

Name:	***QueryJob(JobID: Long)***
Responsibilities:	Retrieve the information on an affixation job.
Cross-references:	Requirements: R5 Use case: Query affixation job.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information on an affixation job has been retrieved.

E.4.1.4.21 Sequence diagram contracts: Change affixation job data

This sequence diagram contains the following contracts:

- ModifyJob

Name:	***ModifyJob(JobData: Object)***
Responsibilities:	Update the information on an affixation job.
Cross-references:	Requirements: R2 Use case: Change affixation job data.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information on the affixation job has been modified.

E.4.1.4.22 Sequence diagram contracts: Get affixation job report

This sequence diagram contains the following contracts:

- AffixationReport

Name:	***AffixationReport(QueryData: Object)***
Responsibilities:	Display a report of the status of an affixation job.
Cross-references:	Requirements: R4, R6

	Use case: Get affixation job report.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The report containing the status of the affixation job is displayed.

E.4.1.4.23 Sequence diagram contracts: Query prohibitions

Name:	***GetProhibitions(Proprietor:String, Municipality:Int, Site: Int, Prohibition:String)***
Responsibilities:	Retrieve the information on the hoardings that have prohibitions and match the selected criteria.
Cross-references:	Requirements: I1, I2, I3 Use case: Query prohibitions
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The hoardings that match the specified criteria are displayed.

Name:	***ExportProhibitionsExcel(Data:Object)***
Responsibilities:	Convert the selected data to a MS Excel format file.
Cross-references:	Requirements: Use case:
Notes:	
Exceptions:	
Output:	
Pre-conditions:	

Post-conditions:	The MS Excel format file has been generated.

E.4.2 Development cycle 1 design

The interaction diagrams and UML class diagrams are developed in this section.

E.4.2.1 Interaction diagrams

Below, we illustrate the operations in the sequence diagrams constructed during the analysis phase by means of interaction diagrams.

E.4.2.1.1 Sequence diagram: GetUser(UserName:string)

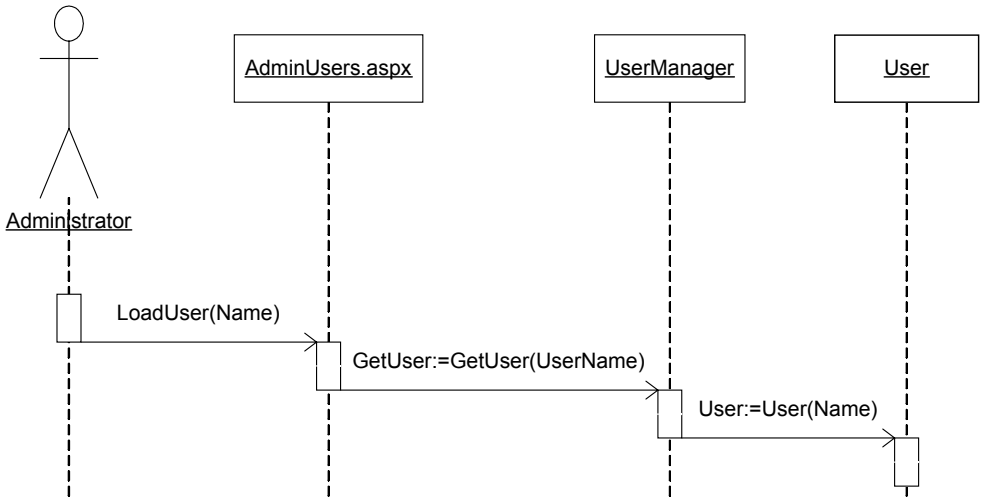

Figure 37 Sequence diagram: GetUser

E.4.2.1.2 Sequence diagram: AssignApplication(UserName:string, ApplicationName:String)

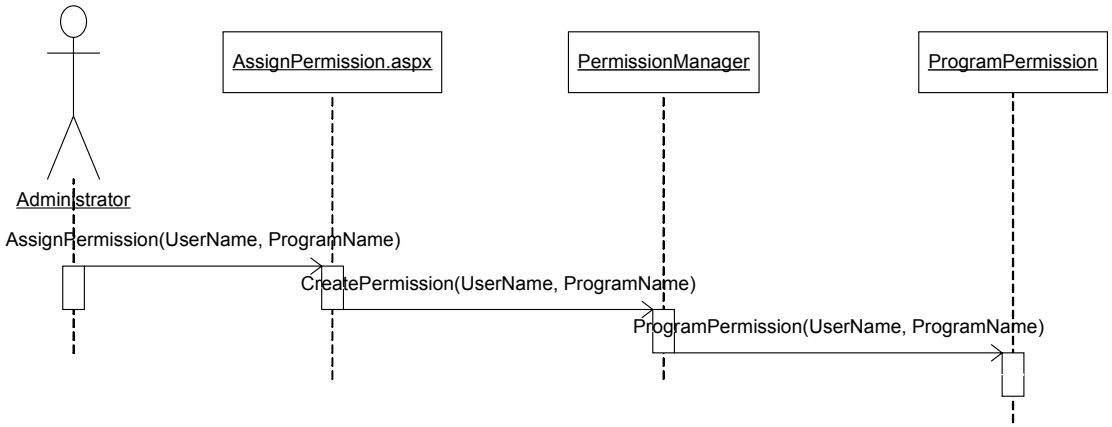

Figure 38 Sequence diagram: AssignApplication

E.4.2.1.3 Sequence diagram: EnterApplication(NodeName:String, NodeType:String, ParentName:String, Route:String)

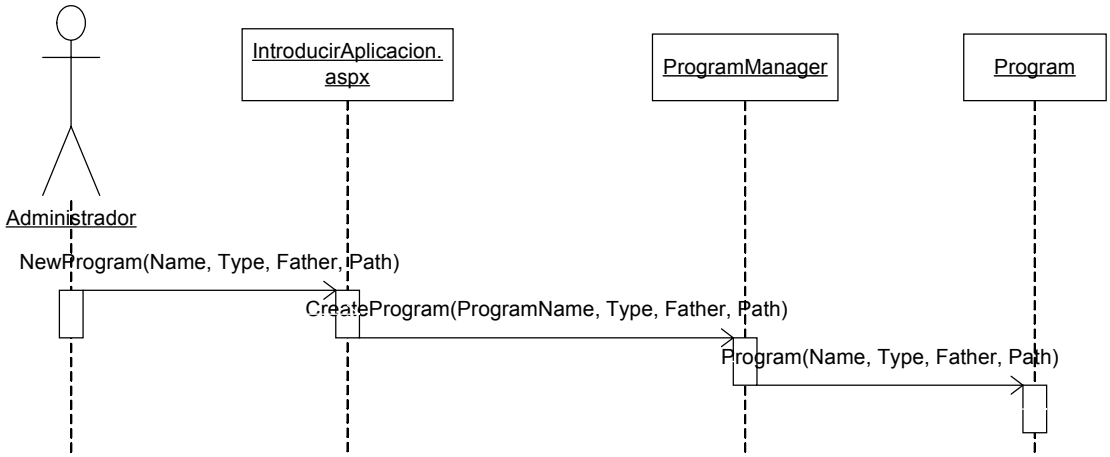

Figure 39 Sequence diagram: EnterApplication

E.4.2.1.4 Sequence diagram: UpdateContracts()

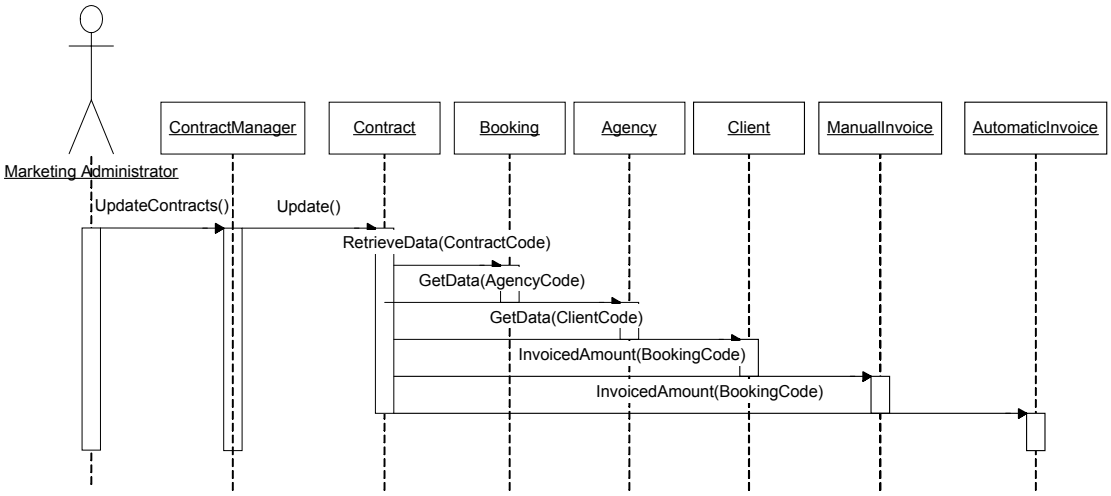

Figure 40 Sequence Diagram: UpdateContracts

E.4.2.1.5 Sequence diagram: CreateAdvertisingGroup (AdvertisingGroupData:Object)

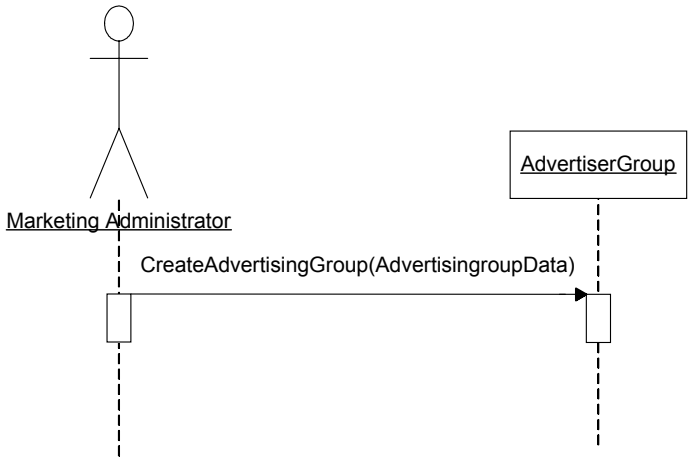

Figure 41 Sequence Diagram: CreateAdvertisingGroup

E.4.2.1.6 Sequence diagram: CreateAdvertiser(AdvertiserData:Object)

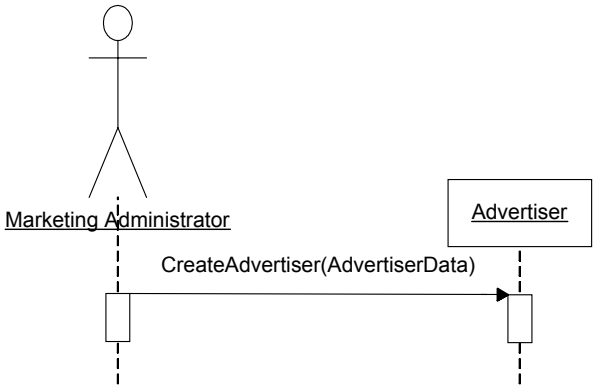

Figure 42 Sequence Diagram: CreateAdvertiser

E.4.2.1.7 Sequence diagram: CreateProductCategory(ProductCategoryData:Object)

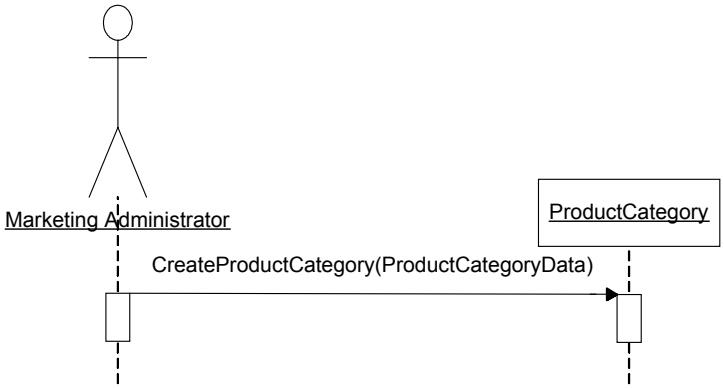

Figure 43 Sequence Diagram: CreateProductCategory

E.4.2.1.8 Sequence diagram: CreateProduct(ProductData:Object)

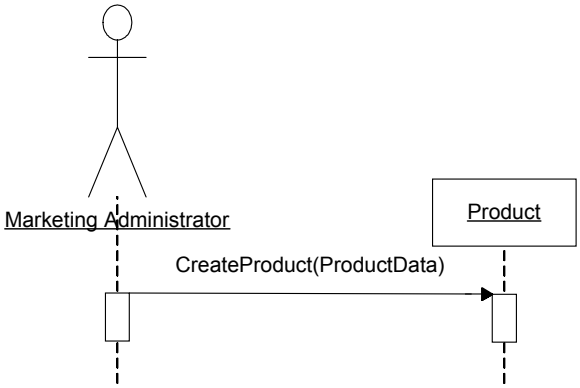

Figure 44 Sequence Diagram: CreateProduct

E.4.2.1.9 Sequence diagram: CreateCampaign(CampaignData:Object)

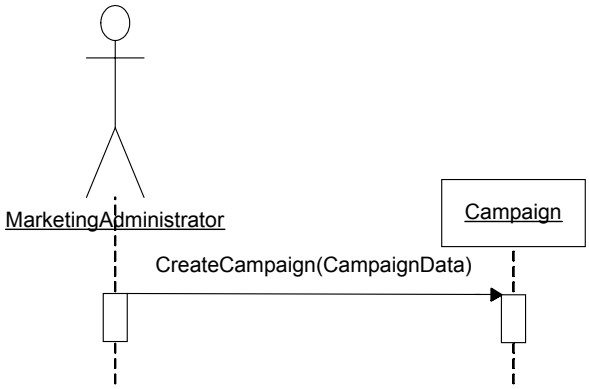

Figure 45 Sequence Diagram: CreateCampaign

E.4.2.1.10 Sequence diagram: QueryCampaign(CampaignData:Object)

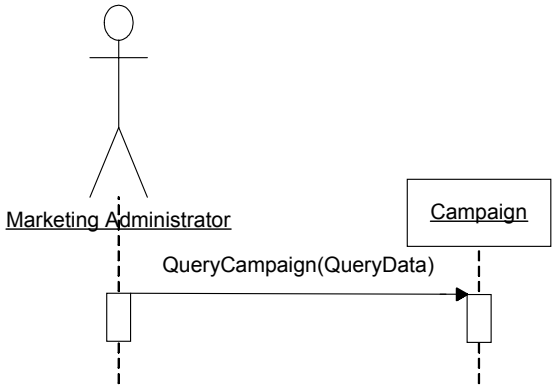

Figure 46Sequence Diagram: QueryCampaign

E.4.2.1.11 Sequence diagram: QueryContract(ContractCode:int)

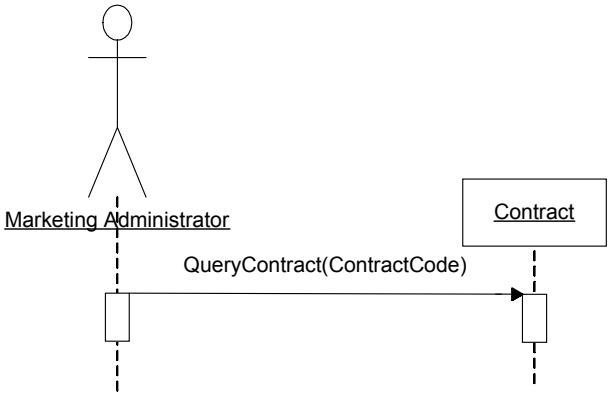

Figure 47 Sequence Diagram: QueryContract

E.4.2.1.12 Sequence diagram: SaveContract(ContractData:Object)

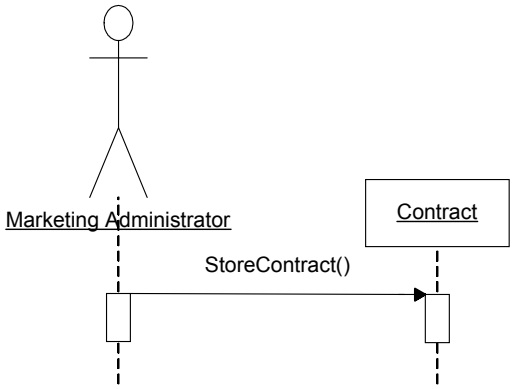

Figure 48 Sequence Diagram: SaveContract

E.4.2.1.13 Sequence diagram: QueryBooking(BookingCode:int)

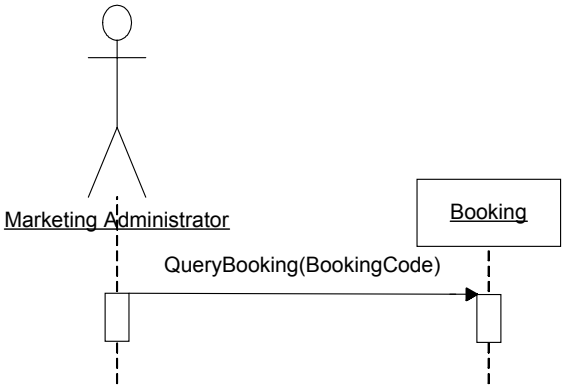

Figure 49 Sequence Diagram: QueryBooking

E.4.2.1.14 Sequence diagram: QueryCustomer(CustomerCode:int)

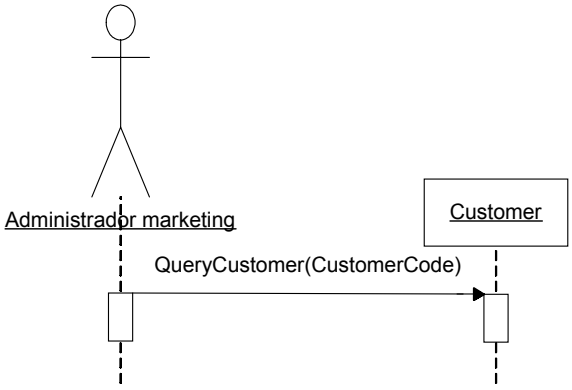

Figure 50 Sequence Diagram: QueryCustomer

E.4.2.1.15 Sequence diagram: QueryAgency(AgencyCode:int)

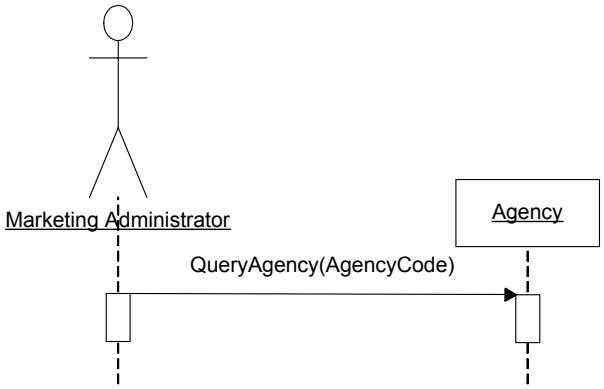

Figure 51 Sequence Diagram: QueryAgency

E.4.2.1.16 Sequence diagram: QueryContracts(QueryData:int)

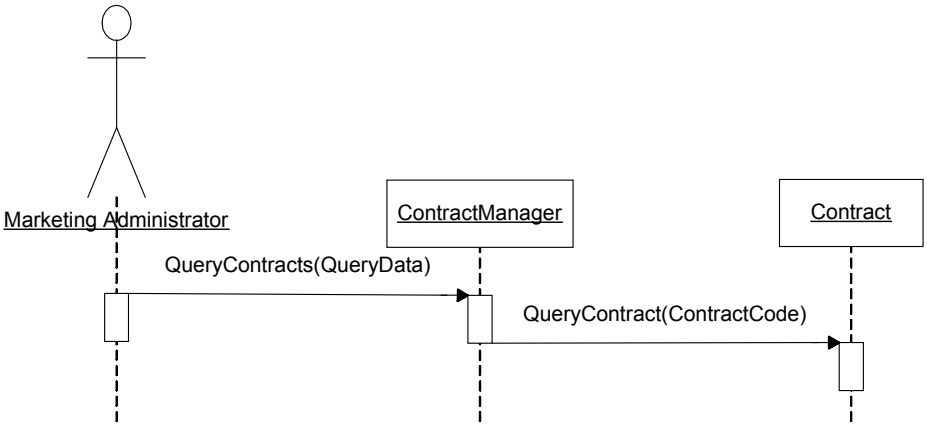

Figure 52 Sequence Diagram: QueryContracts

E.4.2.1.17 Sequence diagram: ExportReport(Report:Object, Route:String)

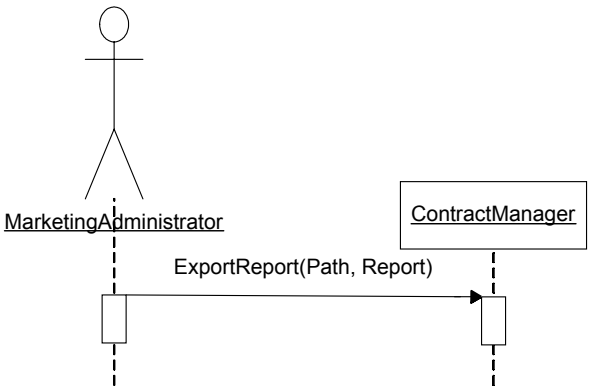

Figure 53 Sequence Diagram: ExportReport

E.4.2.1.18 Sequence diagram: ExportExcel(Contract:object)

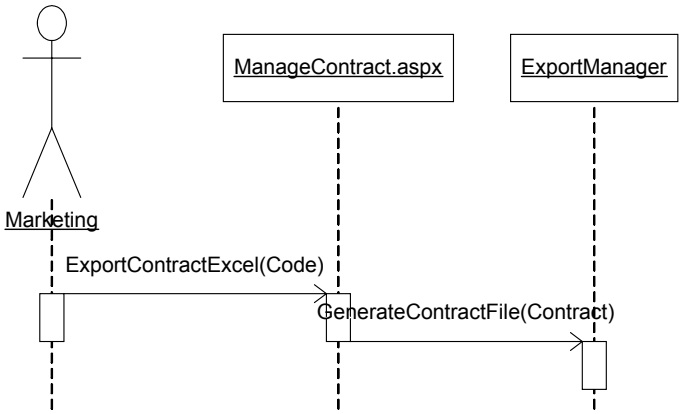

Figure 54 Sequence diagram: ExportExcel

E.4.2.1.19 Sequence diagram: QueryPlanning(RegionName:String, TownName:String, CircuitName:String, StartPeriod:String, EndPeriod:String)

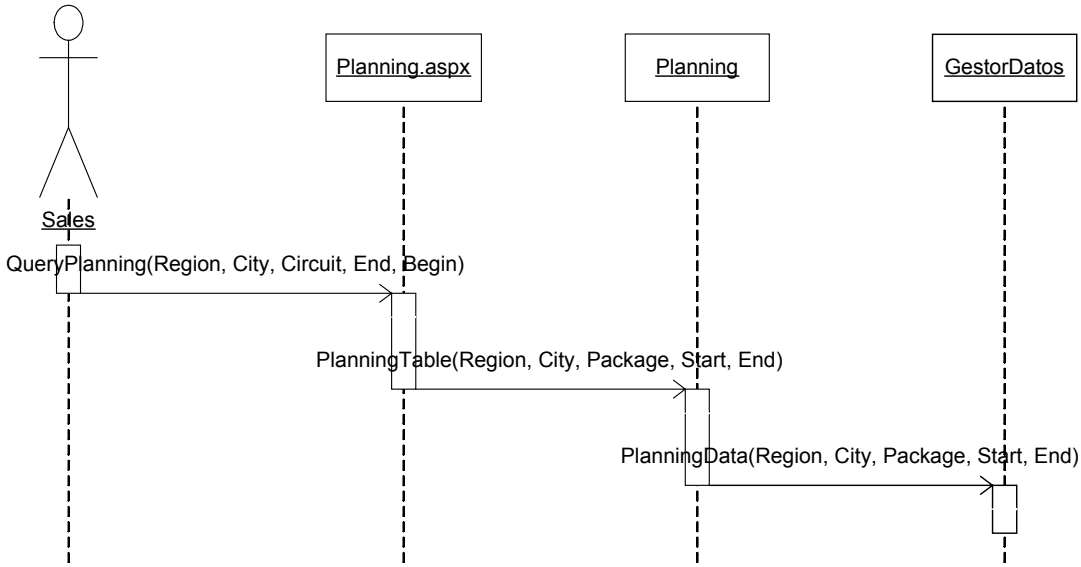

Figure 55 Sequence diagram: QueryPlanning

E.4.2.1.20 Sequence diagram: QueryWeekDetails(RegionName:String, TownName:String, CircuitName:String, Period:String)

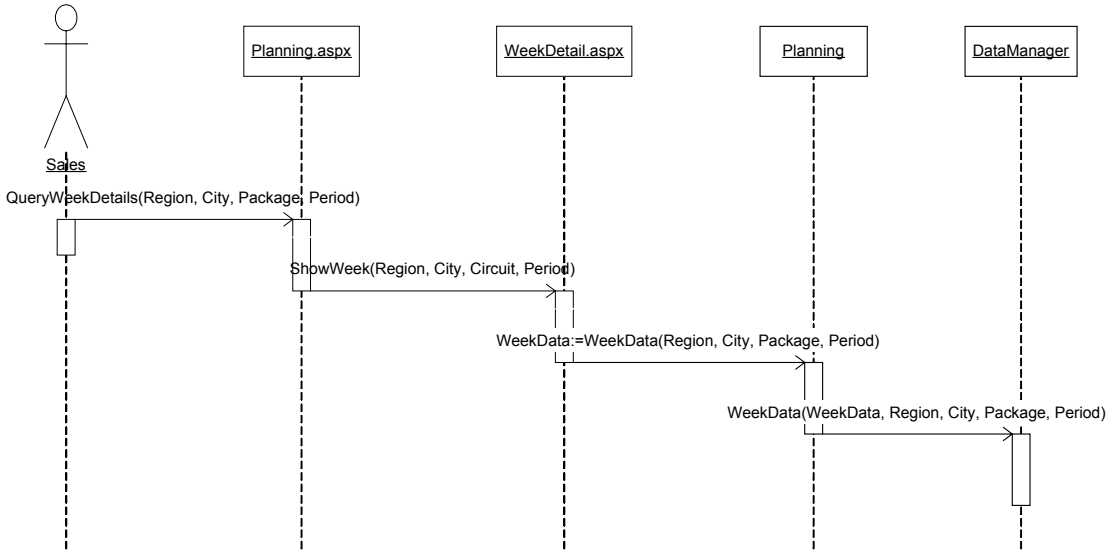

Figure 56 Sequence diagram: QueryWeekDetails

E.4.2.1.21 Sequence Diagram: Query affixation job tickets

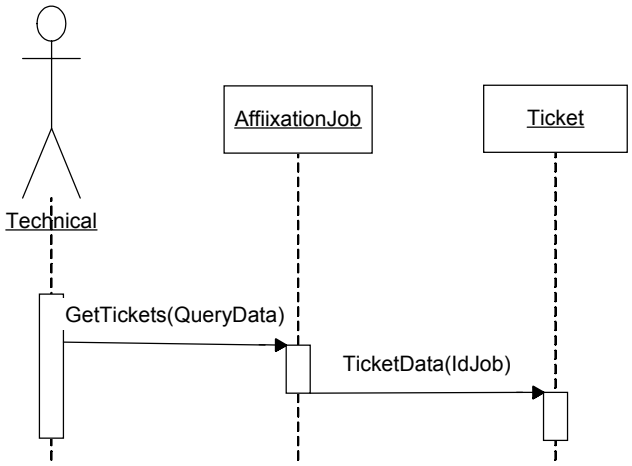

Figure 57 Sequence Diagram: QueryAffixationJobTickets

E.4.2.1.22 Sequence Diagram: Change ticket status

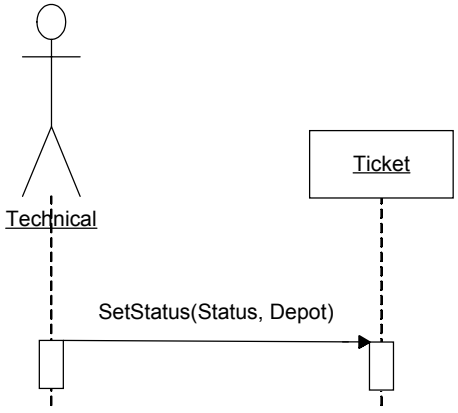

Figure 58 Sequence Diagram: ChangeTicketStatus

E.4.2.1.23 Sequence Diagram: Query affixation job

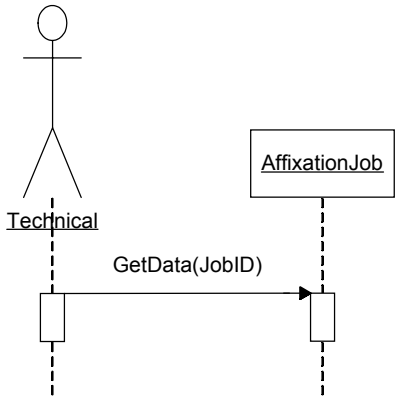

Figure 59 Sequence Diagram: QueryAffixationJob

E.4.2.1.24 Sequence Diagram: Change affixation job data

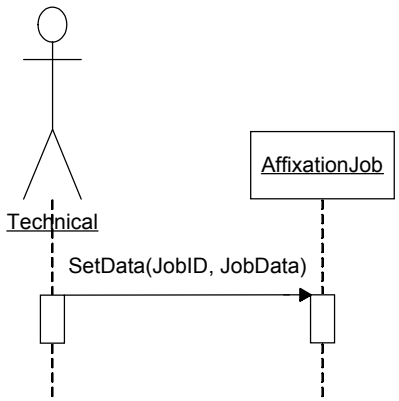

Figure 60 Sequence Diagram: ChangeAffixationJobData

E.4.2.1.25 Sequence Diagram: Get affixation job report

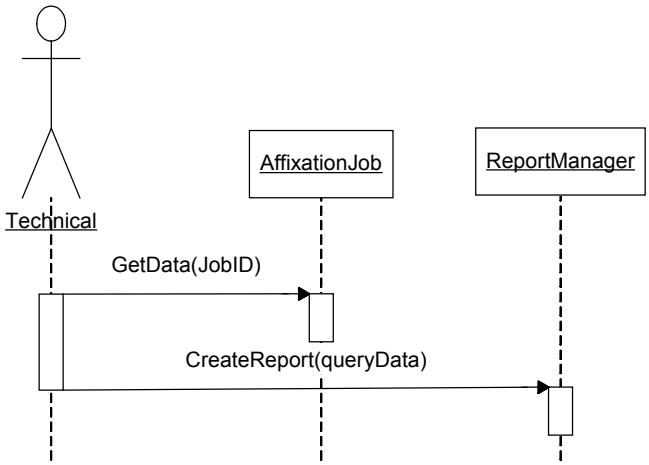

Figure 61 Sequence Diagram: GetAffixationJobReport

E.4.2.1.26 Sequence diagram: GetProhibitions(ProprietorCode:String, MunicipalityCode:String, SiteCode:String, ProhibitionCode:String, Period:String)

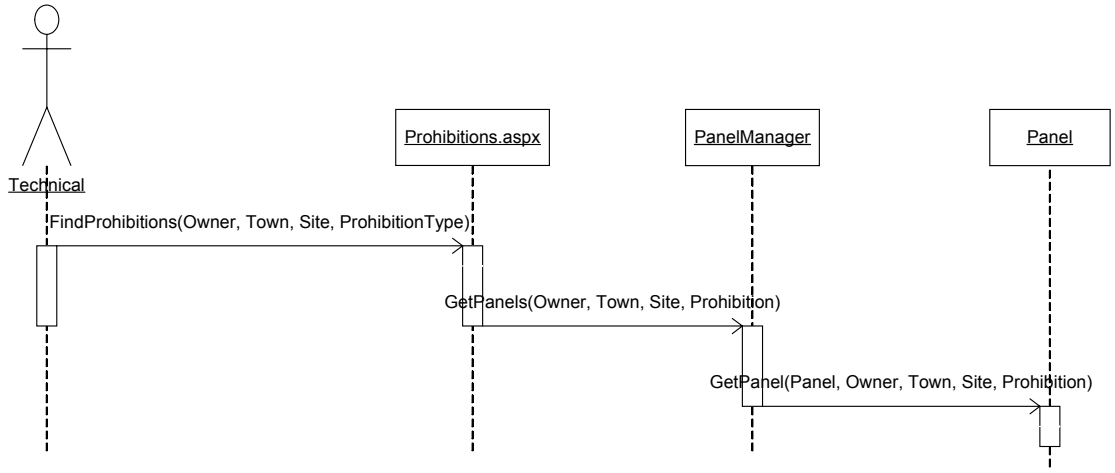

Figure 62 Sequence diagram: GetProhibitions

E.4.2.1.27 Sequence diagram: ExportProhibitionExcel(ProhibitionData:object)

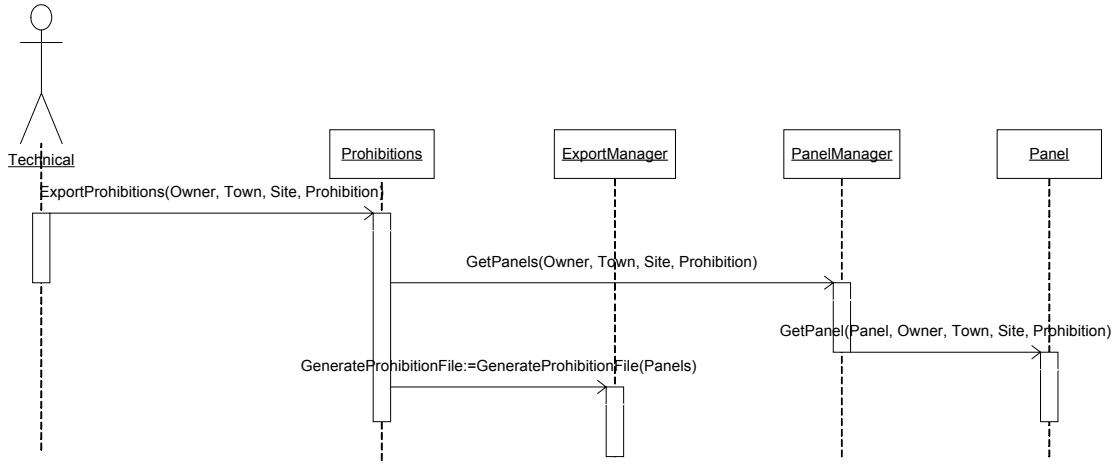

Figure 63 Sequence diagram: ExportProhibitionExcel

E.4.2.2 Class Diagrams

The following class diagram has been created from the interaction diagrams and conceptual model, which illustrates the classes that will be used in cycle 1.

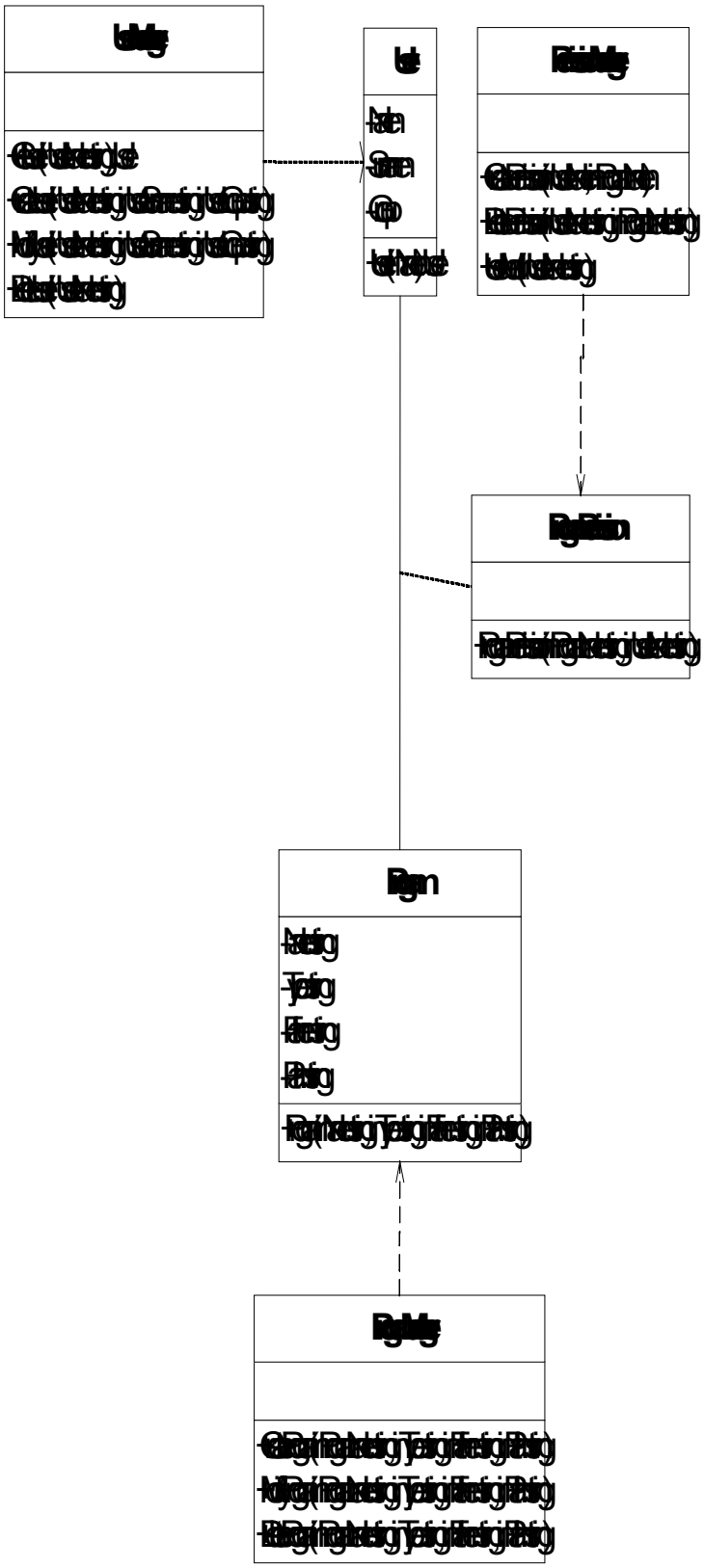

Figure 64 Class diagram (part 1)

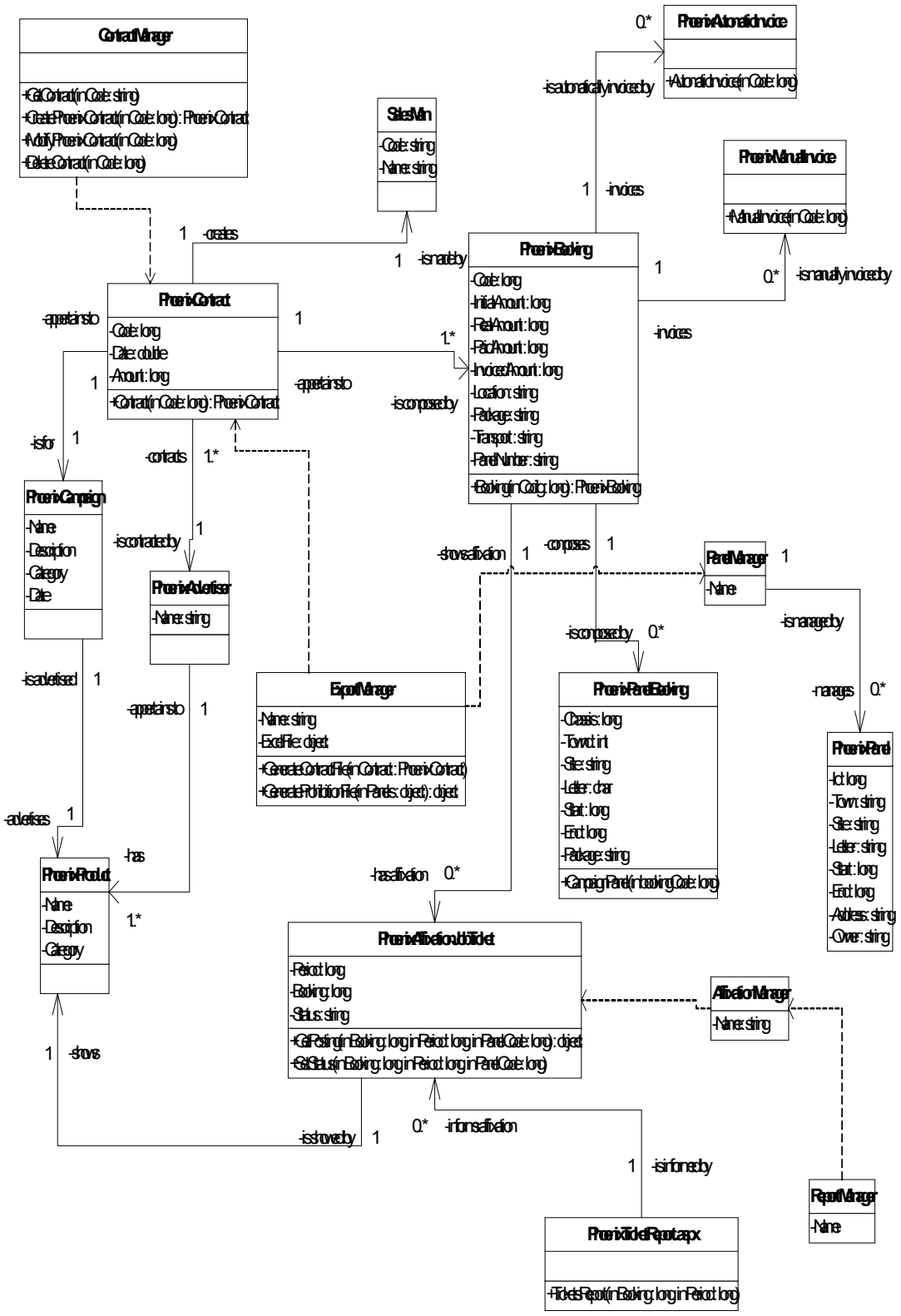

Figure 65 Class diagram (part 2)

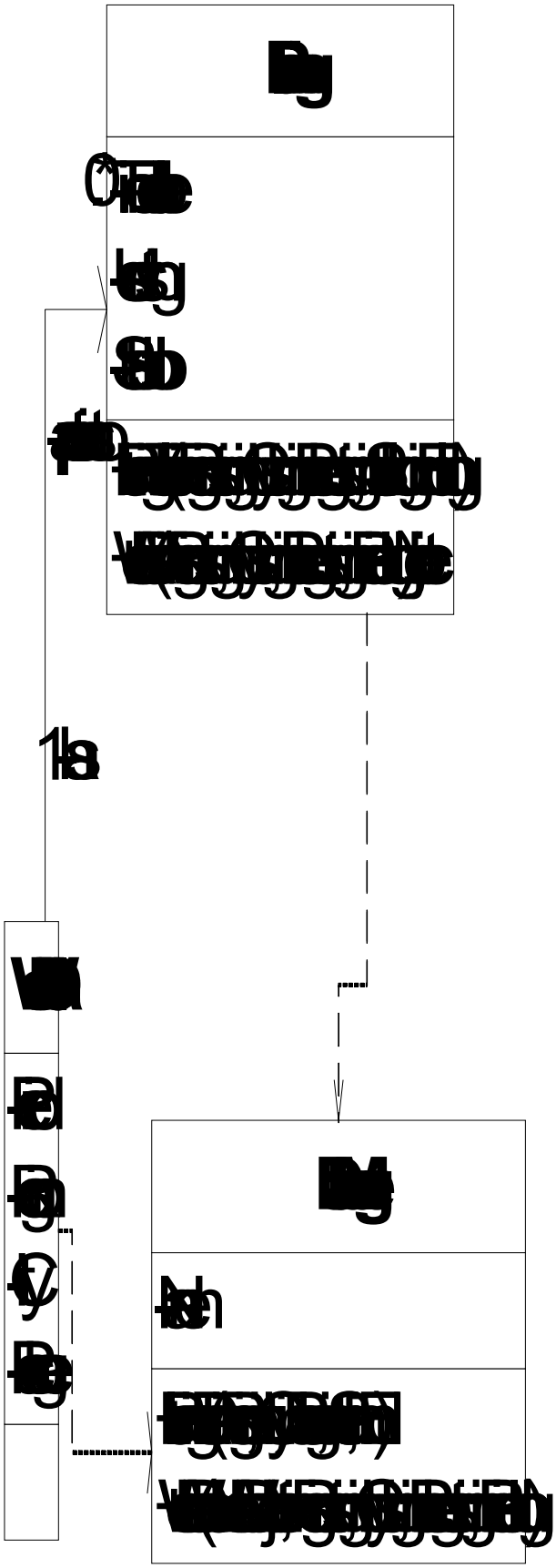

Figure 66 Class diagram (part 3)

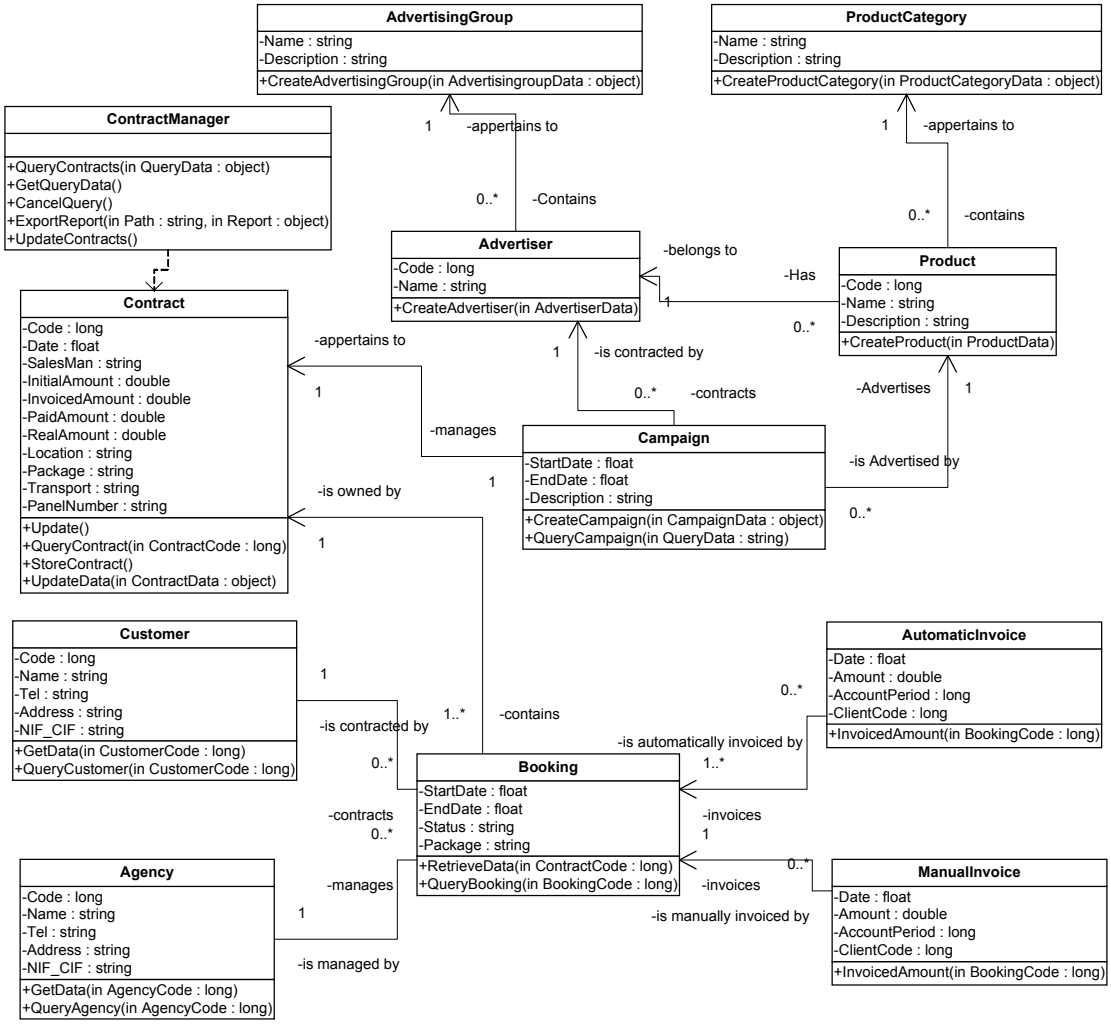

Figure 67 Class diagram (part 4)

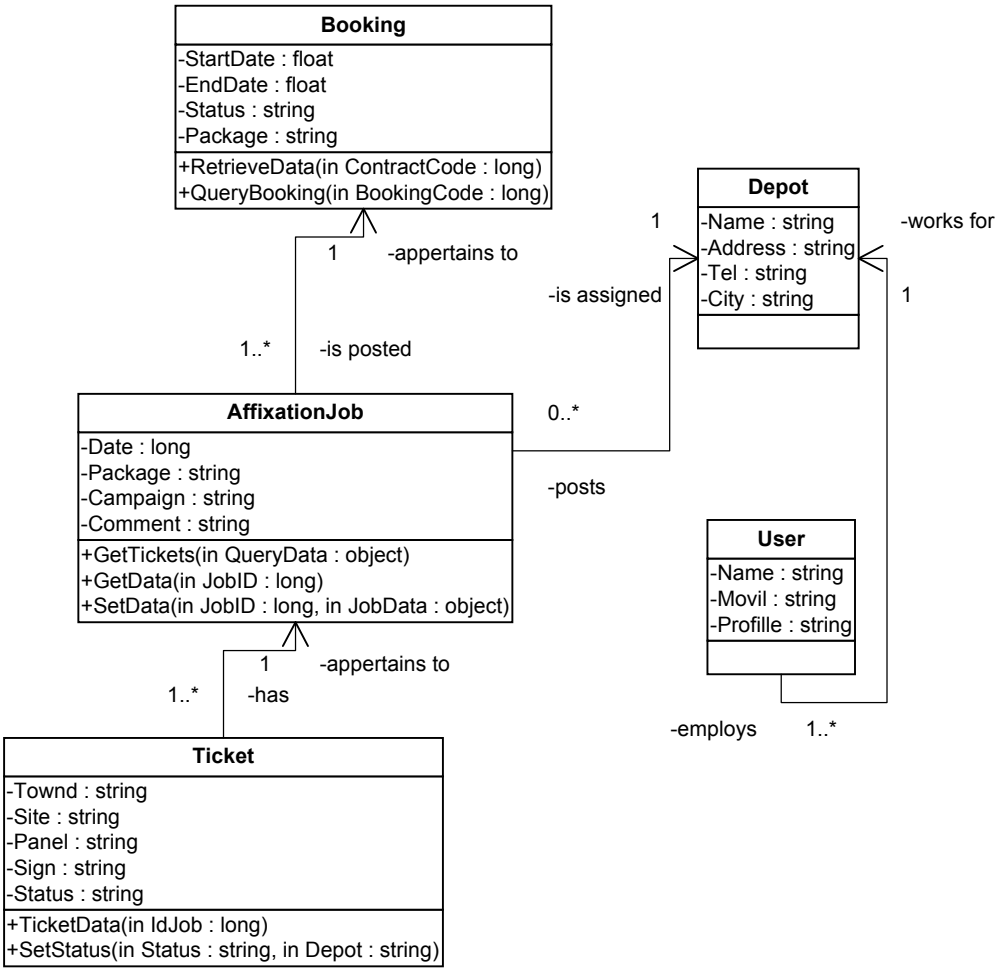

Figure 68 Class diagram (part 5)

E.4.2.3 Description of User Interfaces

In this chapter, we show the interconnection between the user interfaces and the most important functions addressed in this development cycle. Interconnection diagrams will be used for this description.

The interface interconnection diagram shown in Figure 5.2-21 shows the succession of screens used to administer the user categories.

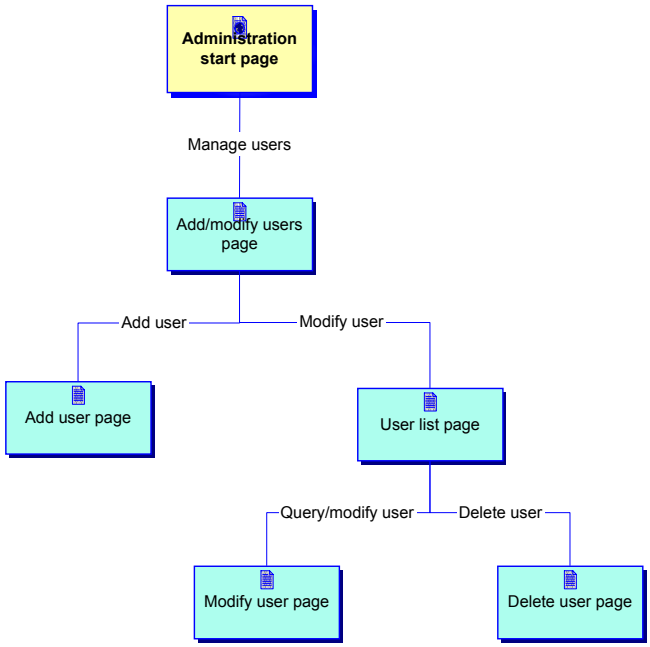

Figure 69 User management interconnection diagram

The interface interconnection diagram shown in Figure 5.1-22 shows the succession of screens used to administer applications.

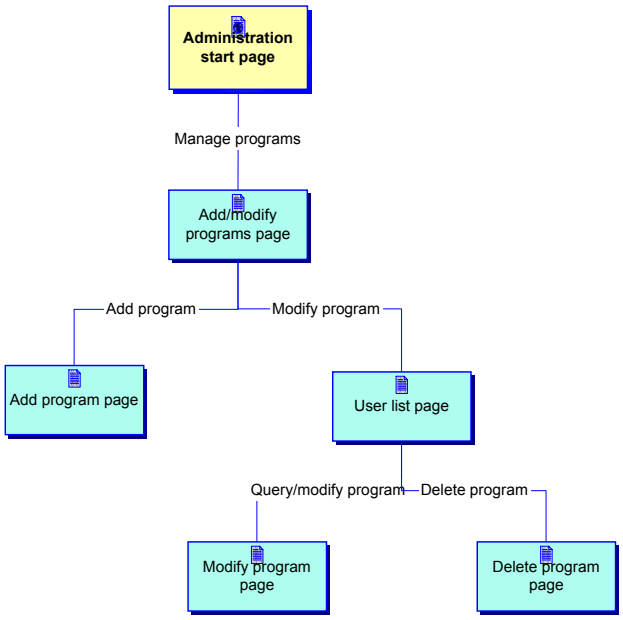

Figure 70 Application management interconnection diagram

The interface interconnection diagram shown in Figure 5.1-23 shows the succession of screens used to administer permits.

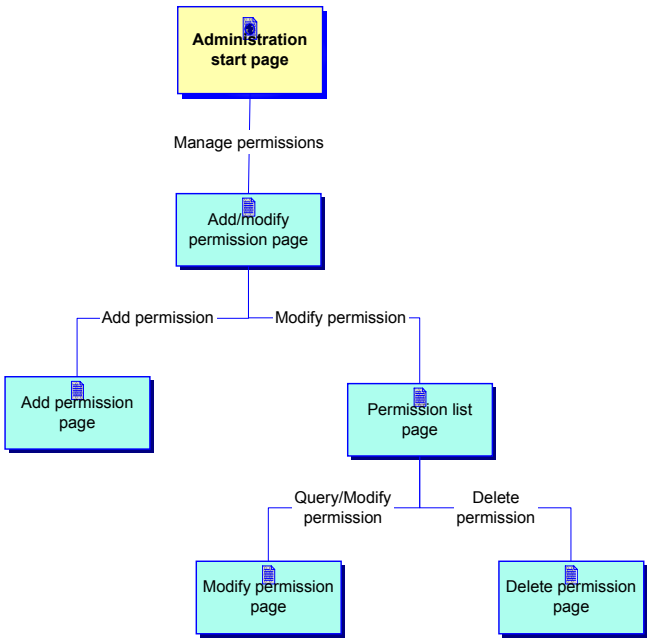

Figure 71 Permit management interconnection diagram.

The interface interconnection diagram shown in Figure 5.1-24 shows the succession of the most important pages for contract management.

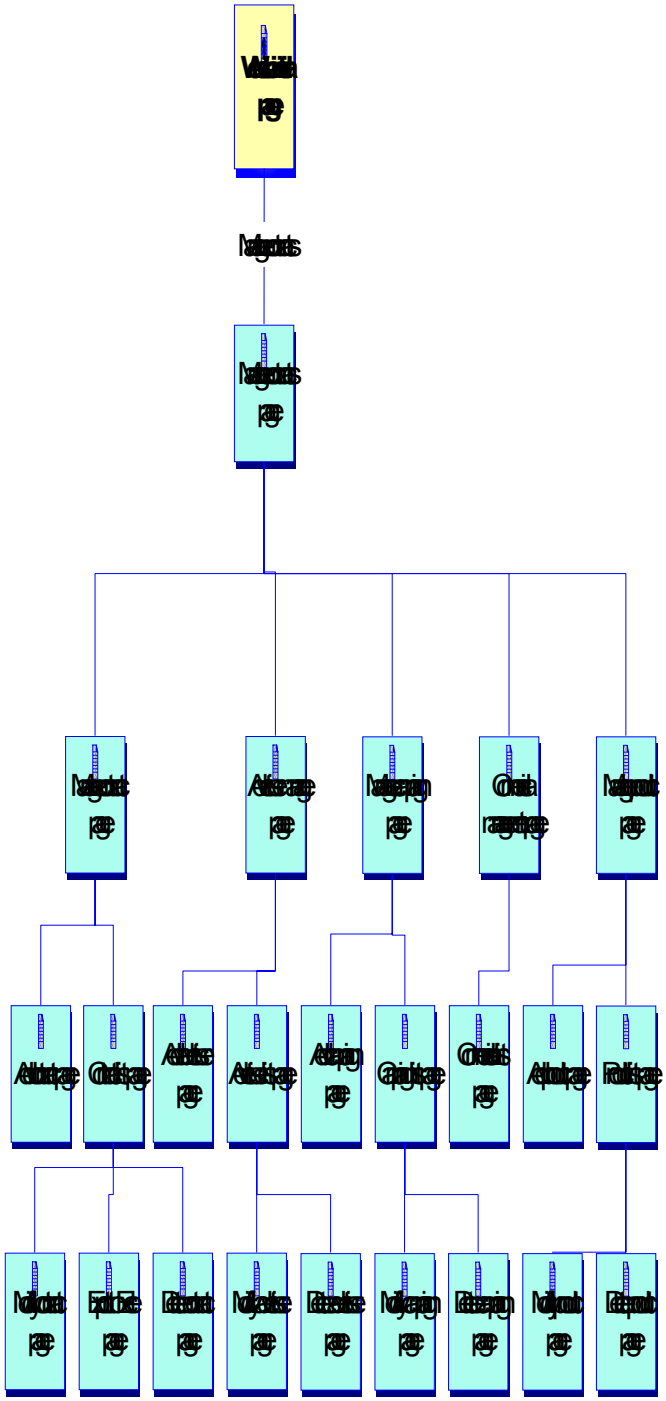

Figure 72 Contracts management interconnection diagram

The interface interconnection diagram shown in Figure 5.1-25 shows the succession of the most important pages for planning queries.

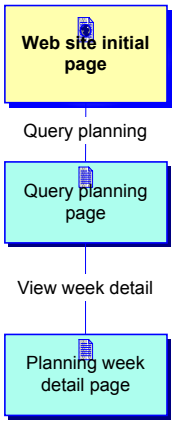

Figure 73 Planning query interconnection diagram

The interface interconnection diagram shown in Figure 5.1-26 shows the succession of the most important pages for affixation management.

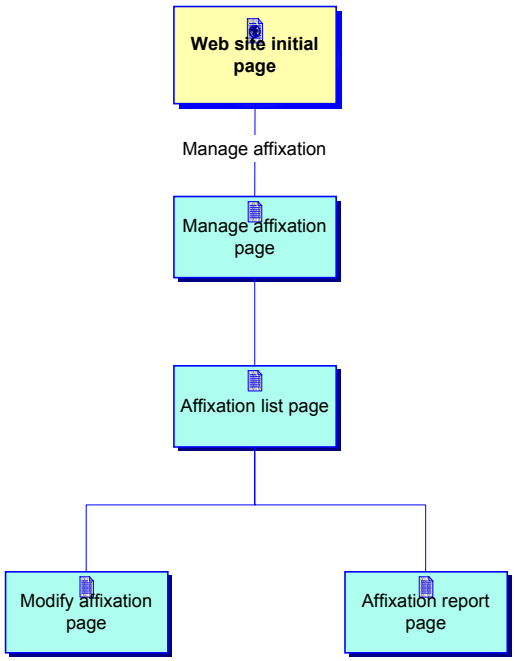

Figure 74 Affixation management interconnection diagram

The interface interconnection diagram shown in Figure 5.1-27 shows the succession of the most important pages for prohibitions querying.

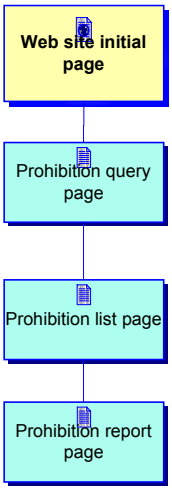

Figure 75 Prohibitions query

Test plan

This phase will be useful for discovering and detect any errors and defects generated during the implementation phase. Each test case will be run according to a series of steps:

- Unit testing: these tests check the functionality of each module separately.
- Integration testing: these tests focus on the integration of the modules tested in the above step.
- System testing: verifies that the system meets the specified requirements.

E.4.2.4 General testing environment

A testing environment that is identical to the production environment with regard to operating system, database, Web server and platform.net will be implemented to run the tests.

E.4.2.5 Testing manager

The tests will be run by the developer and author of this document.

E.4.2.6 Unit testing

The unit tests to be run on the system are presented below

E.4.2.6.1 Access control

Scope

The scope of this test is to evaluate the module responsible for controlling user access to the applications or part of the applications for which they have a permit.

Purpose

Check that the system correctly identifies and validates users and restricts the parts of the application that they are not authorised to use.

Specific resources

Tables: users, permits, applications.

Procedures

- Check that the system detects and validates users on the basis of domain validation. If it cannot validate users by this method, it should request the user name and password. If the credentials are incorrect, check that an error page is displayed, asking the user to try again.
- Having validated the user, check that the system loads the information for this user and displays the appropriate menu for the user's permit level.
- Check that the user cannot access the system without validation or, after validation, cannot access applications for which he is not authorised.

E.4.2.6.2 User management

Scope

The scope of this test is to evaluate the module responsible for managing users.

Purpose

Check that the user management process is performed correctly.

Specific resources

Tables: User

Procedures

- Check that the system validates all the fields required to enter a user. If any of the fields are incorrect, the system should display an error message.
- Check that the same user cannot be entered twice. If this happens, the system should display an error message.
- Check that the system manages errors and that the system displays a message if any error occurs.
- Check that if a user has been successfully entered, the system displays a message to this effect.
- Check that the system validates all the fields in the event of user modification. If any of the fields are incorrect, the system should display an error message.
- Check that if a user has been modified, the system displays a message to this effect.
- Check that the system correctly deletes users and displays a message to this effect.

E.4.2.6.3 Applications Management

Scope

The scope of this test is to evaluate the module responsible for managing applications.

Purpose

Check that the applications management process is performed correctly.

Specific resources

Tables: application

Procedures

- Check that the system validates all the fields required to enter an application. If any of the fields are incorrect, the system should display an error message.
- Check that the same application cannot be entered twice. If this happens, the system should display an error message.
- Check that the system manages errors and that the system displays a message if any error occurs.
- Check that if an application has been successfully entered, the system displays a message to this effect.
- Check that the system validates all the fields in the event of application modification. If any of the fields are incorrect, the system should display an error message.
- Check that if an application has been modified, the system displays a message to this effect.
- Check that the system correctly deletes applications and displays a message to this effect.

E.4.2.6.4 Permits Management

Scope

The scope of this test is to evaluate the module responsible for managing permits.

Purpose

Check that the permits management process is performed correctly.

Specific resources

Tables: application, user, permit.

Procedures

- Check that the system validates all the fields required to enter an application. This validation includes checking whether users and applications exist. If any of the fields are incorrect, the system should display an error message.
- Check that the same permit cannot be entered twice. If this happens, the system should display an error message.

- Check that the system manages errors and that the system displays a message if any error occurs.
- Check that if a permit has been successfully entered, the system displays a message to this effect.
- Check that the system validates all the fields in the event of permit modification. If any of the fields are incorrect, the system should display an error message.
- Check that if a permit has been modified, the system displays a message to this effect.
- Check that the system correctly deletes permits and displays a message to this effect.

E.4.2.6.5 Contracts Management

Scope

The scope of this test is to evaluate the module responsible for managing contracts.

Purpose

Check that the contracts management process is performed correctly.

Specific resources

Tables: contract, advertiser, campaign, sales, product.

Procedures

- Check that the system validates all the fields required to **enter** a **contract**. If any of the fields are incorrect, the system should display an error message.
- Check that the system validates all the fields required to **enter** an advertiser. If any of the fields are incorrect, the system should display an error message.
- Check that the system validates all the fields required to **enter** a **campaign**. If any of the fields are incorrect, the system should display an error message.
- Check that the system validates all the fields required to **enter** a **product**. If any of the fields are incorrect, the system should display an error message.
- Check that the same **contract** cannot be entered **twice**. If this happens, the system should display an error message.
- Check that the same **campaign** cannot be entered **twice**. If this happens, the system should display an error message.
- Check that the same **advertiser** cannot be entered **twice**. If this happens, the system should display an error message.
- Check that the same **product** cannot be entered **twice**. If this happens, the system should display an error message.
- Check that the system **manages errors** and that the system displays a message if any error occurs.

- Check that if a **contract** has been **successfully** entered, the system displays a message to this effect.
- Check that if a **campaign** has been **successfully** entered, the system displays a message to this effect.
- Check that if an **advertiser** has been **successfully** entered, the system displays a message to this effect.
- Check that if a **product** has been **successfully** entered, the system displays a message to this effect.
- Check that the system validates all the fields in the event of **contract modification**. If any of the fields are incorrect, the system should display an error message.
- Check that the system validates all the fields in the event of **campaign modification**. If any of the fields are incorrect, the system should display an error message.
- Check that the system validates all the fields in the event of **advertiser modification**. If any of the fields are incorrect, the system should display an error message.
- Check that the system validates all the fields in the event of **product modification**. If any of the fields are incorrect, the system should display an error message.
- Check that **if a contract has been modified**, the system displays a message to this effect.
- Check that **if a campaign has been modified**, the system displays a message to this effect.
- Check that **if an advertiser has been modified**, the system displays a message to this effect.
- Check that **if a product has been modified**, the system displays a message to this effect.
- Check that the system correctly **deletes** a **contract** and displays a message to this effect.
- Check that the system correctly **deletes** a **campaign** and displays a message to this effect.
- Check that the system correctly **deletes** an **advertiser** and displays a message to this effect.
- Check that the system correctly **deletes** a **product** and displays a message to this effect.
- Check that having queried the contracts, the system is capable of displaying a report and exporting the data to an Excel file.

E.4.2.6.6 Planning

Scope

The scope of this test is to evaluate the module responsible for querying and displaying the planning.

Purpose

Check that the planning querying and display process is performed correctly.

Specific resources

Tables: planning, contracts.

Procedures

- Check that the system validates all the fields required to query the planning. If any of the fields are incorrect, the system should display an error message.
- Check that the system correctly displays the planning according to the input parameters. If data are not retrieved, the system should display a message to this effect.
- Check that the system **manages errors** and that the system displays a message if any error occurs.
- Check that the information on the contracts for the period are correctly displayed in the view details.

E.4.2.6.7 Affixation Management

Scope

The scope of this test is to evaluate the module responsible for affixation management.

Purpose

Check that the affixation management process is performed correctly.

Specific resources

Tables: affixation.

Procedures

- Check that the system manages errors and if any error occurs, the system displays a message.
- Check that the system validates all the fields in the event of affixation modification. If any of the fields are incorrect, the system should display an error message.
- Check that when an affixation has been modified, the system displays a message to this effect.
- Check that having queried the affixations, the system is capable of displaying a report and exporting the data to an Excel file.

E.4.2.6.8 Query Prohibitions

Scope

The scope of this test is to evaluate the module responsible for querying prohibitions.

Purpose

Check that the prohibitions querying process is performed correctly.

Specific resources

Tables: advertising sign.

Procedures

- Check that the system manages errors and that the system displays a message if any error occurs.
- Check that the system validates all the fields required to query prohibitions. If any of the fields is incorrect, the system should show an error message.
- Check that having queried the prohibitions, the system is capable of displaying a report and exporting the data to an Excel file.

E.4.2.7 Integration Testing

After unit testing, integration tests are run on the modules. This will show up any errors due to module interaction. The strategy pursued for each application is the bottom-up integration, that is, starting from the low functionality modules upon which others are based.

1. Integration of user entry, deletion and modification.
2. Integration of application entry, deletion and modification.
3. Integration of permit entry, deletion and modification.
4. Integration of contract, advertiser, campaign and product entry, deletion and modification
5. Integration of planning querying with week details querying.
6. Integration of affixations management with access control.
7. Integration of prohibitions querying with access control.
8. Integration of modules 1 to 5 with access control.

E.4.2.8 System Testing

After running the unit and integration tests, we proceed to run the following performance tests in the production environment. These tests will be similar to the ones developed in the integration testing phase, except that they are in the production environment and being run by multiple users.

E.4.2.8.1 Contract entry, modification and deletion

Scope:

The scope of this test is contracts management.

Purpose:

Check that the contracts management operations are performed absolutely normally under conditions of routine use and during access by multiple users.

Specific requirements:

Tables: Contract

Procedure:

- Enter a series of contracts. Check that even under conditions of high activity, the system responds without problems.
- Modify the data of the entered contracts, changing the value of multiple fields. Check that everything is done normally or that, in the event of any error, errors are managed.
- Delete several contracts successively. Check that everything is done normally or that, in the event of any error, errors are managed.
- Switch the contracts entry, modification and deletion order and check data consistency from the application and by directly accessing the database.

E.4.2.8.2 Advertiser entry, modification and deletion

Scope:

The scope of this system test is advertiser management.

Purpose:

Check that the advertiser management operations are performed absolutely normally under conditions of routine use and during access by multiple users.

Specific requirements:

Tables: Advertiser

Procedure:

- Enter a series of advertisers. Check that even under conditions of high activity, the system responds without problems.
- Modify the data of the entered advertisers, changing the value of multiple fields. Check that everything is done normally or that, in the event of any error, errors are managed.
- Delete several advertisers successively. Check that everything is done normally or that, in the event of any error, errors are managed.
- Switch the advertiser entry, modification and deletion order and check data consistency from the application and by directly accessing the database.

E.4.2.8.3 Campaign entry, modification, deletion

Scope:

The scope of this system test is campaign management.

Purpose:

Check that the campaign management operations are performed absolutely normally under conditions of routine use and during access by multiple users.

Specific requirements:

Tables: Campaign

Procedure:

- Enter a series of campaigns. Check that even under conditions of high activity, the system responds without problems.
- Modify the data of the entered campaigns, changing the value of multiple fields. Check that everything is done normally or that, in the event of any error, errors are managed.
- Delete several campaigns successively. Check that everything is done normally or that, in the event of any error, errors are managed.
- Switch the campaign entry, modification and deletion order and check data consistency from the application and by directly accessing the database.

E.4.2.8.4 Product entry, modification and deletion

Scope:

The scope of this system test is product management.

Purpose:

Check that the product management operations are performed absolutely normally under conditions of routine use and during access by multiple users.

Specific requirements:

Tables: Product

Procedure:

- Enter a series of products. Check that even under conditions of high activity, the system responds without problems.
- Modify the data of the entered products, changing the value of multiple fields. Check that everything is done normally or that, in the event of any error, errors are managed.
- Delete several products successively. Check that everything is done normally or that, in the event of any error, errors are managed.
- Switch the product entry, modification and deletion order and check data consistency from the application and by directly accessing the database.

E.4.2.8.5 Planning query

Scope:

The scope of this system test is planning querying.

Purpose:

Check that the planning querying operations are run absolutely normally in conditions of routine use of the application during access by multiple users.

Specific requirements:

Tables: Planning, detail

Procedure:

- Make multiple planning queries and check that the application operates normally. If any errors occur, check that they are managed.
- Inspect multiple screens generated in the above point and check that the information matches the monitor database.
- Get details for multiple periods and check that the application works normally. If any errors occur, check that they are managed.
- Inspect multiple screens generated in the above point and check that the information matches the monitor database.

E.4.2.8.6 Affixation Management

Scope:

The scope of this system test is affixation management.

Purpose:

Check that the affixation management operations are performed normally under conditions of routine use of the application during access by multiple users.

Specific requirements:

Tables: Affixation

Procedure:

- Modify the data on multiple affixations. Check that everything is done normally. In the event of errors, check that they are managed.
- Get an affixations report and check that this matches the changes made.

E.4.2.8.7 Query prohibitions

Scope:

The scope of this system test is query prohibitions.

Purpose:

Check that the prohibitions querying operations are run normally under conditions of routine use of the application and during access by multiple users.

Specific requirements:

Tables: Advertising signs

Procedure:

- Run multiple queries of the prohibitions from several terminals.
- Export these queries to Excel.

E.5 DEVELOPMENT CYCLE 1 (WITH USABILITY PATTERNS)

E.5.1 Development cycle 1 analysis

In this cycle we address the analysis of the IPESP system taking into account usability requirements. As these patterns have been introduced in only part of the application, namely, the development of "Classification of bookings by marketing category" and "affixation control", we will only describe the parts of the analysis and design that have been modified. This will be used for comparison with the previous analysis and to identify the changes made. The steps taken will be as follows: description of the use cases in expanded format, construction of the sequence diagrams and preparation of operations contracts.

E.5.1.1 Use cases in low level format

Below, we present the low-level use cases for the development cycle 1 applications for which insufficient detail was provided by the first approximation.

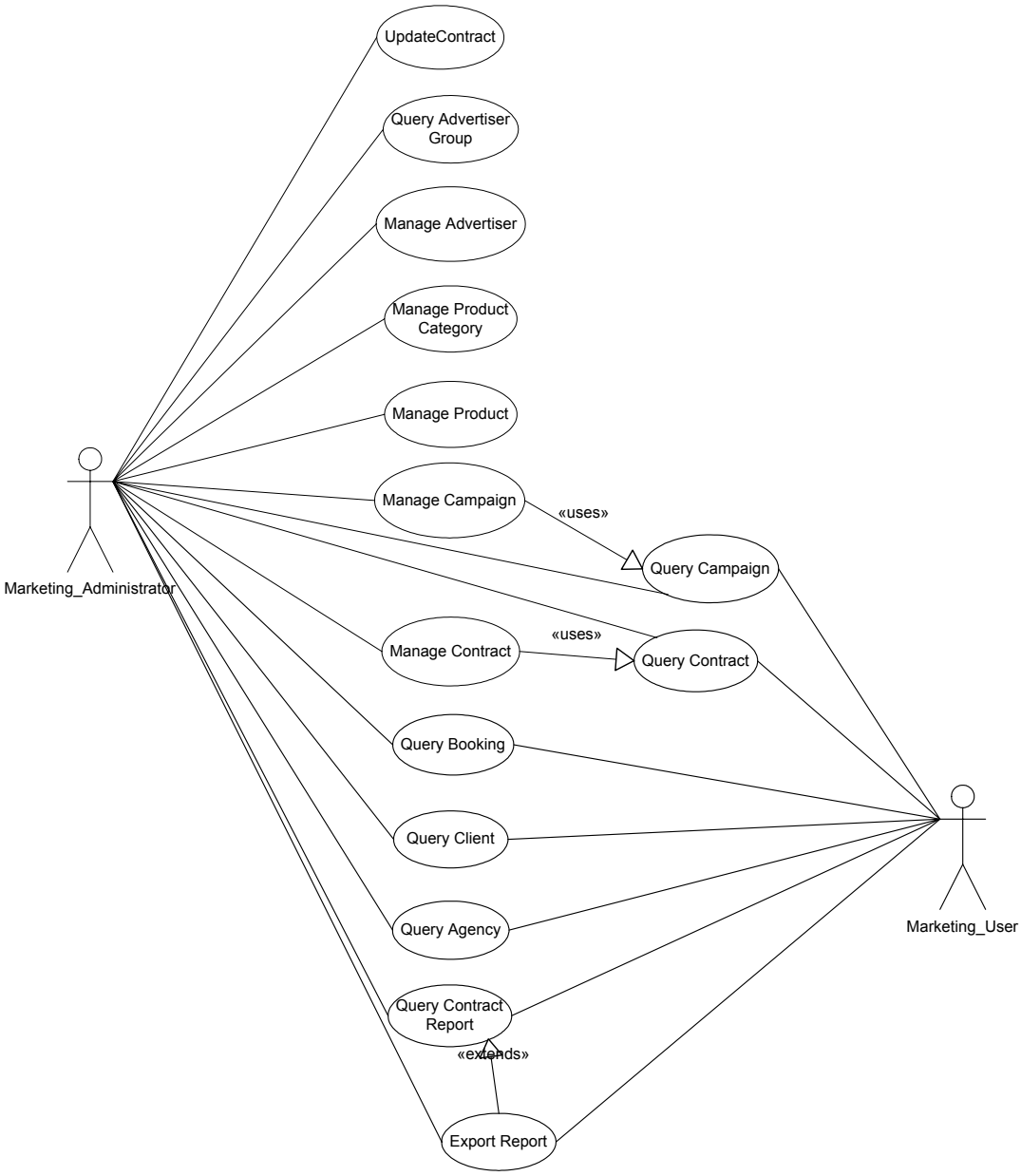

Figure 76 Marketing bookings classification use case diagram

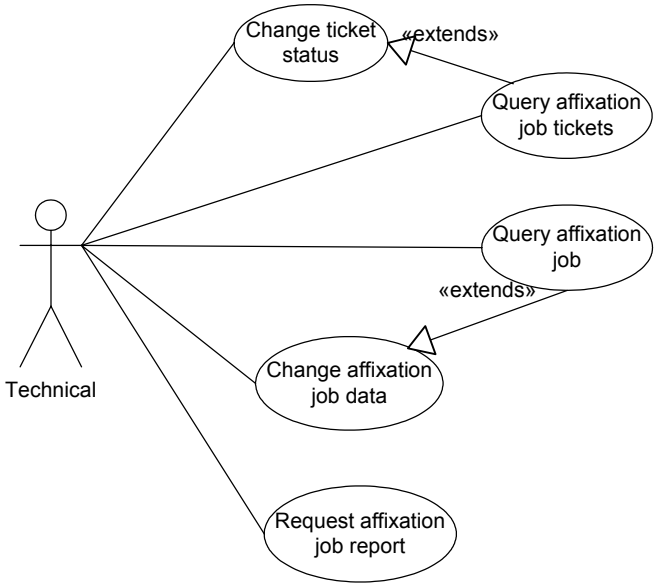

Figure 77 Control affixation use case diagram

E.5.1.2 Use cases in expanded format

The use cases are described in expanded format in more detail below. This format describes the use case at more length than the high level format, which is of considerable help for building the sequence diagrams.

As the usability patterns have been included in the classification of bookings by marketing category and affixation control, special emphasis will be placed on the parts that involve steps and system responses related to usability patterns, which will be highlighted.

E.5.1.2.1 Use Case: Update contracts

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Search the information on new contracts that have been entered in Phoenix	
Overview:	The actor asks the system to update the information on contracts. The system searches the contract-related information in the Phoenix database and updates this information in the system database	
References:	*Req* P2, P3, P4, P6, P7, P8, P9, P11, P13, P14	
Typical course	**Actor action**	**System response**

	1. The use case starts when the marketing_administrator enters the update contracts application.	2. The system recognises the user language and displays the information in the respective language (UP-Different languages).
	3. The user asks the system to update the information on contracts.	4. The system retrieves the information on contracts and updates the database. During the process the system informs the user that it is updating data and when the process has finished (UP-Status Indication).

E.5.1.2.2 Use Case: Manage advertising group

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify to delete an advertising group.	
Overview:	The administrator enters the data on the advertising group to be added, modified or deleted and asks the system to execute the action.	
References:	*Req*: P5, P10, P11,P13	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the marketing_administrator enters the update contracts application.	2. The system recognises the user language and displays the information in the respective language (UP-Different languages).
	3. The administrator fills in the data on the new advertising group and asks the system to create this new advertising group using a button or an abbreviated method (UP-Shortcuts)	4. The system validates the information and, if it is correct, creates the new advertising group. The user is informed if the process ends correctly (UP-Status Indication). The user is informed if the information is incorrect or any error occurs (UP- Status Indication).

Alternative course 1	3.1 The administrator fills in the data on the new advertising group and asks the system to retrieve the information on this advertising group using a button or an abbreviated method (UP-Shortcuts).	4.1 The system retrieves the information on the advertising group and this information, if any, is displayed. The user is informed if an error occurs (UP-Status Indication).
Alternative course 2	3.2 The administrator fills in the data required to identify the advertising group and asks the system to delete the group using a button or an abbreviated method (UP-Shortcuts).	4.2 The system retrieves the information on the advertising group and, if it can be deleted, the user is informed of deletion by means of an alert. The user is informed if it cannot be deleted (UP-Status Indication).
Alternative course 3	3.3 The administrator fills in the data required to retrieve the advertising group and asks the system to retrieve the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system retrieves the information on the advertising group, which it displays to the user. The user is informed if an error occurs during retrieval (UP- Status Indication).
	3.4 The administrator modifies the information on the advertising group and asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system updates the information on the advertising group and informs the user that it has been updated by means of an alert (UP-Alert). The user is informed if a retrieval error occurs (UP- Status Indication).

E.5.1.2.3 Use Case: Manage advertiser

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify to delete an advertiser.	
Overview:	The administrator enters the data on the advertiser to be added, modified or deleted and asks the system to execute the action.	
References:	*Req*: P5, P10, P11,P13	
Typical course	**Actor action**	**System response**

	1. The use case starts when the marketing_administrator enters the update contracts application.	2. The system recognises the user language and displays the information in the respective language (UP-Different languages).
	3. The administrator fills in the data on the new advertiser and asks the system to create this new advertiser using a button or an abbreviated method (UP-Shortcuts).	4. The system validates the information and, if it is correct, creates the new advertiser. The user is informed if the process ends correctly (UP-Status Indication). The user is informed if the information is incorrect or any error occurs (UP- Status Indication).
Alternative course 1	3.1 The administrator fills in the data on the new advertiser and asks the system to retrieve the information on this advertiser using a button or an abbreviated method (UP-Shortcuts).	4.1 The system retrieves the information on the advertiser and this information, if any, is displayed. The user is informed if an error occurs (UP- Status Indication).
Alternative course 2	3.2 The administrator fills in the data required to identify the advertiser and asks the system to delete the advertiser using a button or an abbreviated method (UP-Shortcuts).	4.2 The system retrieves the information on the advertiser and, if it can be deleted, the user is informed of deletion by means of an alert. The user is informed if it cannot be deleted (UP- Status Indication).
Alternative course 3	3.3 The administrator fills in the data required to retrieve the advertiser and asks the system to retrieve the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system retrieves the information on the advertiser, which it displays to the user. The user is informed if an error occurs during retrieval (UP- Status Indication).
	3.4 The administrator modifies the information on the advertiser and asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system updates the information on the advertiser and informs the user that it has been correctly updated by means of an alert (UP-Alert). The user is informed if a retrieval error occurs (UP- Status Indication).

E.5.1.2.4 Use Case: Manage product category

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify to delete a product category.	
Overview:	The administrator enters the data of the product category to be added, modified or deleted and asks the system to execute the action.	
References:	*Req:* P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the marketing_administrator enters the update contracts application.	2. The system recognises the user language and displays the information in the respective language (UP-Different languages) .
	3. The administrator fills in the data on the new product category and asks the system to create this new product category using a button or an abbreviated method (UP-Shortcuts).	4. The system validates the information and, if it is correct, creates the new product category. The user is informed if the process ends correctly (UP-Status Indication). The user is informed if the information is incorrect or any error occurs (UP- Status Indication).
Alternative course 1	3.1 The administrator fills in the data on the new product category and asks the system to retrieve the information on this product category using a button or an abbreviated method (UP-Shortcuts).	4.1 The system retrieves the information on the product category and this information, if any, is displayed. The user is informed if an error occurs (UP-Status Indication).
Alternative course 2	3.2 The administrator fills in the data required to identify the product category and asks the system to delete the category using a button or an abbreviated method (UP-Shortcuts).	4.2 The system retrieves the information on the product category and, if it can be deleted, the user is informed of deletion by means of an alert. The user is informed if it cannot be deleted (UP-Status Indication).

Alternative course 3	3.3 The administrator fills in the data required to retrieve the product category and asks the system to retrieve the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system retrieves the information on the product category, which it displays to the user. The user is informed if an error occurs during retrieval (UP- Status Indication).
	3.4 The administrator modifies the information on the product category and asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system updates the information on the product category and informs the user that it has been updated by means of an alert (UP-Alert). The user is informed if a retrieval error occurs (UP- Status Indication).

E.5.1.2.5 Use Case: Manage product

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify to delete a product.	
Overview:	The administrator enters the data of the product to be added, modified or deleted and asks the system to execute the action.	
References:	*Req:* P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the marketing_administrator enters the update contracts application.	2. The system recognises the user language and displays the information in the respective language (UP-Different languages) .
	3. The administrator fills in the data on the new product and asks the system to create this new product using a button or an abbreviated method (UP-Shortcuts).	4. The system validates the information and, if it is correct, creates the new product. The user is informed if the process ends correctly (UP-Status Indication). The user is informed if the information is incorrect or any error occurs (UP- Status Indication).

Alternative course 1	3.1 The administrator fills in the data on the new product and asks the system to retrieve the information on this product using a button or an abbreviated method (UP-Shortcuts).	4.1 The system retrieves the information on the product and this information, if any, is displayed. The user is informed if an error occurs (UP- Status Indication).
Alternative course 2	3.2 The administrator fills in the data required to identify the product and asks the system to delete the product using a button or an abbreviated method (UP-Shortcuts).	4.2 The system retrieves the information on the product and, if it can be deleted, the user is informed of deletion by means of an alert. The user is informed if it cannot be deleted (UP- Status Indication).
Alternative course 3	3.3 The administrator fills in the data required to retrieve the product and asks the system to retrieve the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system retrieves the information on the product, which it displays to the user. The user is informed if an error occurs during retrieval (UP- Status Indication).
	3.4 The administrator modifies the information on the product and asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system updates the information on the product and informs the user that it has been updated by means of an alert (UP-Alert). The user is informed if a retrieval error occurs (UP- Status Indication).

E.5.1.2.6 Use Case: Manage campaign

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify to delete a campaign.	
Overview:	The administrator enters the data of the campaign to be added, modified or deleted and asks the system to execute the action.	
References:	*Req*: P5, P10, P11, P13, P14	
Typical course	**Actor action**	**System response**

	1. The use case starts when the marketing_administrator enters the update contracts application.	2. The system recognises the user language and displays the information in the respective language (UP-Different languages).
	3. The administrator fills in the data on the new campaign and asks the system to create this new campaign using a button or an abbreviated method (UP-Shortcuts).	4. The system validates the information and, if it is correct, creates the new campaign. The user is informed if the process ends correctly (UP-Status Indication). The user is informed if the information is incorrect or any error occurs (UP- Status Indication).
Alternative course 1	3.1 The administrator fills in the data on the new campaign and asks the system to retrieve the information on this campaign using a button or an abbreviated method (UP-Shortcuts).	4.1 The system retrieves the information on the campaign and this information, if any, is displayed. The user is informed if an error occurs (UP- Status Indication).
Alternative course 2	3.2 The administrator fills in the data required to identify the campaign and asks the system to delete the campaign using a button or an abbreviated method (UP-Shortcuts).	4.2 The system retrieves the information on the campaign and, if it can be deleted, the user is informed of deletion by means of an alert. The user is informed if it cannot be deleted (UP- Status Indication).
Alternative course 3	3.3 The administrator fills in the data required to retrieve the campaign and asks the system to retrieve the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system retrieves the information on the campaign, which it displays to the user. The user is informed if an error occurs during retrieval (UP- Status Indication).
	3.4 The administrator modifies the information on the campaign and asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	4.4 The system updates the information on the campaign and informs the user that it has been updated by means of an alert (UP-Alert). The user is informed if a retrieval error occurs (UP- Status Indication).

E.5.1.2.7 Use Case: Query campaign

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a campaign with the information requested by marketing.	
Overview:	The user selects the code of the campaign to be queried and asks the system for the information.	
References:	*Req*: P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the user enters the query campaign application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).
	3 The administrator fills in the data required to identify the campaign and asks the system to display the information using a button or abbreviated method (UP-Shortcuts).	4 The system retrieves the information on the campaign, which it displays to the user. The user is informed if the campaign cannot be displayed (UP-Status Indication).

E.5.1.2.8 Use Case: Manage contract

Actor	Marketing_administrator	
Type	Primary and essential	
Purpose	Create, modify to delete a contract.	
Overview:	The administrator enters the data of the contract to be added, modified or deleted and asks the system to execute the action.	
References:	*Req* P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the marketing_administrator enters the contract management application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).

	3 The administrator fills in the data on the contract and asks the system to retrieve the information on this contract using a button or abbreviated method (UP-Shortcuts).	4 The system retrieves the information on the contract and displays this information, if any. The user is informed if an error occurs (UP- Status Indication).
	5 The administrator modifies the information on the contract and asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	6 The system updates the information on the contract and informs that it has been correctly updated (UP- Status Indication). The user is informed if a retrieval error occurs (UP- Status Indication)
Alternative course 1	5.1 The administrator asks the system to store the information on this contract for later use as a template using a button or abbreviated method (UP-Shortcuts; UP-Reuse information).	6.1 The system saves the information.
	7 The administrator fills in the data on another contract and asks the system to retrieve the information on this contract using a button or abbreviated method (UP-Shortcuts).	8 The system retrieves the information on the contract and this information, if any, is displayed. The user is informed if an error occurs, (UP- Status Indication).
	9 The administrator asks the system to retrieve the information saved earlier and copy it to this contract. This action can be performed using a button or abbreviated method (UP-Shortcuts; UP-Reuse information).	10 The system retrieves the information, which it assigns to the selected contract. The system informs the user that the values have been correctly assigned (UP-Status Indication). The user is informed if a retrieval error occurs (UP- Status Indication).

	11 The administrator asks the system to save the information using a button or abbreviated method (UP-Shortcuts).	12 The system updates the information on the contract and informs the user that it has been correctly updated (UP- Status Indication). The user is informed if an update error occurs (UP- Status Indication).
Alternative course 2	5.2 The administrator selects a field and asks for help on this particular field (UP-Context-sensitive help) using an abbreviated method (Key F2) (UP-Shortcuts)	6.2 The system displays the information on this field (UP-Context-sensitive help) in the respective language (UP-Different languages).
Alternative course 3	3.2 The administrator asks for help using an abbreviated method (Key F1) (UP–Standard help) (UP-Shortcuts).	4.2 The system displays the general help information in the respective language (UP-Standard help) (UP-Different languages).

E.5.1.2.9 Use Case: Query contract

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a contract with the information requested by marketing.	
Overview:	The user selects the code of the contract to be queried and asks the system for the information.	
References:	*Req* P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the user enters the query contract application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).

	3 The administrator fills in the data required to identify the contract and asks the system to display the information using a button or abbreviated method (UP-Shortcuts).	4 The system retrieves the information on the contract and this information, if any, is displayed. The user is informed if the contract cannot be displayed (UP- Status Indication).

E.5.1.2.10 Use Case: Query booking

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a booking with the information requested by marketing.	
Overview:	The user selects the code of the booking to be queried and asks the system for the information.	
References:	*Req* P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the user enters the query booking application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).
	3 The administrator fills in the data required to identify the booking and asks the system to display the information using a button or abbreviated method (UP-Shortcuts).	4 The system retrieves the information on the booking, which it displays to the user. The user is informed if the booking cannot be displayed (UP- Status Indication)

E.5.1.2.11 Use Case: Query customer

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on a customer with the information requested by marketing.	
Overview:	The user selects the code of the customer to be queried and asks the system for the information.	
References:	*Req* P5, P10, P11, P13, P14	
Typical course	**Actor action**	**System response**

	1. The use case starts when the user enters the query customer application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).
	3 The administrator fills in the data required to identify the customer and asks the system to display the information using a button or abbreviated method (UP-Shortcuts).	4 The system retrieves the information on the customer, which it displays to the user. The user is informed if the customer cannot be displayed (UP-Status Indication).

E.5.1.2.12 Use Case: Query agency

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Query the information on an agency with the information requested by marketing.	
Overview:	The user selects the code of the agency to be queried and asks the system for the information.	
References:	*Req* P5, P10, P11, P13, P14	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the user enters the query agency application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).
	3 The administrator fills in the data required to identify the agency and asks the system to display the information using a button or abbreviated method (UP-Shortcuts).	4 The system retrieves the information on the agency, which it displays to the user. The user is informed if the agency cannot be displayed (UP- Status Indication).

E.5.1.2.13 Use Case: Get contracts report

Actor	Marketing_administrator, Marketing_user
Type	Primary and essential
Purpose	Get a report on contracts that match given criteria by means of a query.
Overview:	The user enters the search parameters and asks the system for the information.

References:	*Req:* P1, P2, P10, P11, P12 P13, P14, P15,P17, P18,P19,P20,P21,P22	
	Actor action	**System response**
Typical course of events	1. The use case starts when the user enters the get contracts report query application.	2. The system recognises the language of the user and displays the information in the respective language (UP-Different languages).
	3 The actor fills in the parameters required for the query to be made (accounting period, advertiser, products, product categories) and asks the system for the report using a button or abbreviated method (UP-Shortcuts).	4 The system validates the parameters received. If they are correct, it retrieves the information that matches the search parameters and indicates it status as "searching" during the search process (UP- Status Indication). If the parameters are incorrect (UP-Form/Field validation), the user is informed by means of an alert (UP-Alert).
		5 The system displays the contracts and their related information and indicates their status as "available" at the end of the query. The user is informed if there are no contracts that match the selected parameters (UP- Status Indication).
	6 The user asks for a preview of the bookings report including the data that have just been requested (UP-Preview).	7 The system displays the report preview (UP-Preview).
Alternative course 1	4.1 The actor decides to delete the parameters and ask the system to re-enter the original parameters (UP-Undo).	5.1 The system enters the original values of the parameters (UP-Undo).
Alternative course 2	5.2 The actor asks to cancel report generation (UP-Cancel).	6.2 The system cancels report generation and returns to the query screen (UP-Cancel).

Alternative course 3	1.1 The use case starts when the actor makes a remote request for bookings by entering the query parameters (accounting period, advertiser, products, product categories) (UP-Different access methods).	2.1 The system validates the parameters. If they are correct, it returns the data in the pre-established format. An empty dataset is returned if the data are incorrect or the data are not found. The system saves a remote access register, with its respective time and query parameters and the user who made the query. (UP-History logging).
Alternative course 4	1.2 The actor asks the system to open another application instance to make two queries at the same time (UP-Multi-tasking).	2.2 The system generates another application instance (UP-Multi-tasking).
	3.2 The use case continues according to the typical course of events.	

E.5.1.2.14 Use Case: Export report

Actor	Marketing_administrator, Marketing_user	
Type	Primary and essential	
Purpose	Export a generated report to a Microsoft Excel format file	
Overview:	Having requested a report, the user asks the system to export the report data to an Excel file.	
References:	*Req:* P23	
Typical course of events	**Actor action**	**System response**
	1. The use case starts when the actor has asked for a report and asks the system to export the report Excel along a given route	2. The system generates the report.

E.5.1.2.15 Use Case: Query affixation job tickets

Actor	Technical
Type	Primary and essential
Purpose	Retrieve the information on the tickets belonging to an affixation job.

Overview:	The actor enters the data to identify the affixation job and asks the system to retrieve related tickets.	
References:	*Req* R3, R5, R8	
	Actor action	**System response**
Typical course of events	1. The use case starts when the marketing_administrator enters the contracts update application.	2. The system recognises the user connecting to the application and saves the profile information of this user. (UP-User profile).
	3. The actor enters the parameters of the query affixation job tickets and asks the system to retrieve the information.	4. The system retrieves the information on the tickets, which it displays to the user.

E.5.1.2.16 Use Case: Change ticket status

Actor	Technical	
Type	Primary and essential	
Purpose	Modify the status of a ticket or group of tickets.	
Overview:	The actor selects the tickets whose status is to be changed and asks the system to change the status.	
References:	*Req*: R1, R7, R8	
	Actor action	**System response**
Typical course of events	1. The actor selects a ticket or a group of tickets and asks the system to change their status.	2. The system checks that the selected tickets belong to the actor's depot and changes the ones that belong to the user's depot (UP-User profile; UP-Action for multiple objects).

E.5.1.2.17 Use Case: Query affixation job

Actor	Technical
Type	Primary and essential
Purpose	View information on an affixation job.

Overview:	The actor inserts the search parameters for the affixation job to be queried and asks the system to display the information.	
References:	*Req:* R5	
Typical course of events	**Actor action**	**System response**
	1. The actor enters the parameters of the affixation job query and asks the system to retrieve the information.	2. The system retrieves the information on the affixation job, which it displays to the user.

E.5.1.2.18 Use Case: Modify affixation job

Actor	Technical	
Type	Primary and essential	
Purpose	Modify the information on an affixation job.	
Overview:	The administrator enters the data on the affixation job to be modified and asks the system to save the information.	
References:	*Req:* R2	
Typical course of events	**Actor action**	**System response**
	1. The actor modifies the affixation job data and asks the system to save the information.	2. The system updates the information on the job.

E.5.1.2.19 Use Case: Get affixation job report

Actor	Technical	
Type	Primary and essential	
Purpose	Get a report on the status of an affixation job.	
Overview:	The administrator enters the search parameters for the affixation job for which a report is to be generated.	
References:	*Req:* R6, R4	
	Actor action	**System response**

Typical course of events	1 The actor fills in the parameters required to make the query (booking, period or job code) and asks the system for the report.	2 The system displays the affixation job report.

E.5.1.3 Sequence diagrams

These diagrams describe the interaction between the actors and the system for each use case, both for the typical course of events and for the major alternative courses.

The system call functions that match or are related to usability cases will be highlighted in red in the diagrams.

E.5.1.3.1 Sequence Diagram: Update contracts

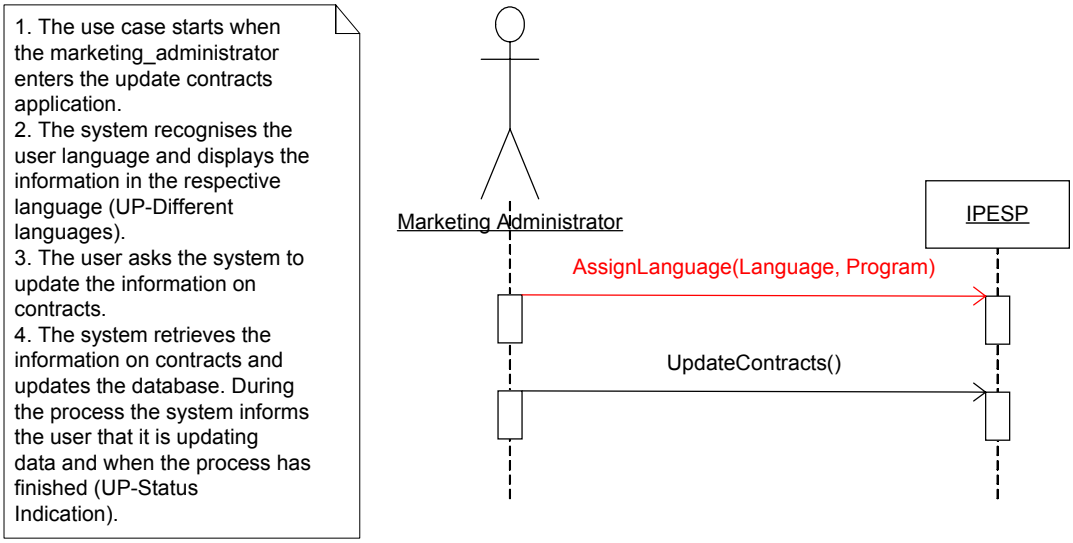

Figure 78 Sequence Diagram: Update contracts

E.5.1.3.2 Sequence Diagram: Manage advertising group

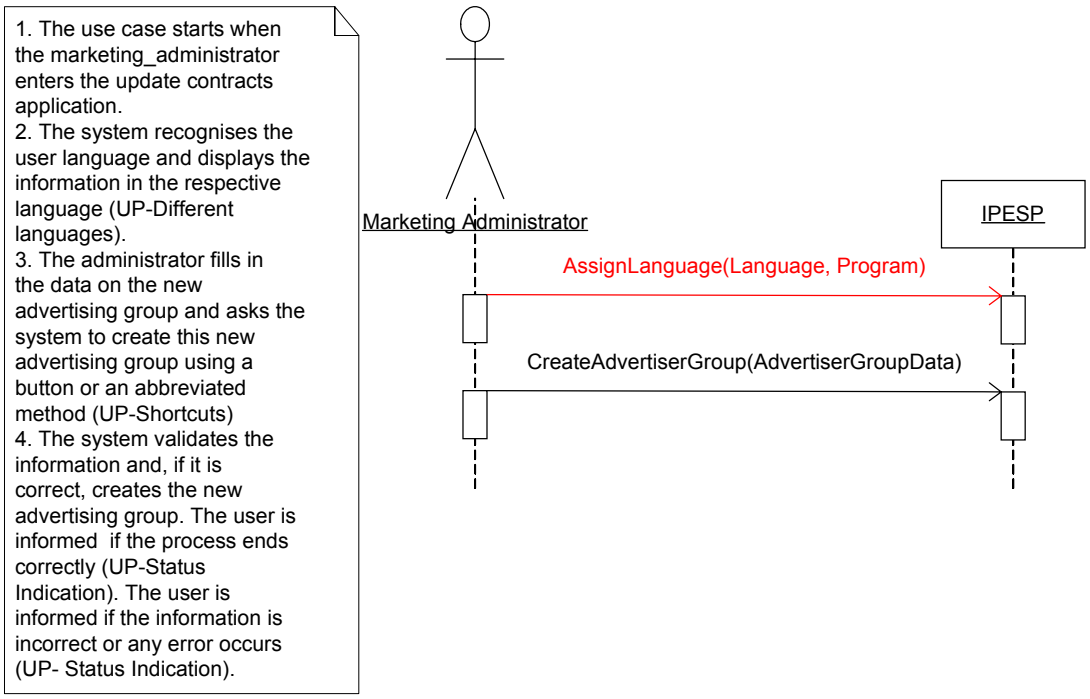

Figure 79 Sequence Diagram: Manage advertising group

E.5.1.3.3 Sequence Diagram: Manage advertiser

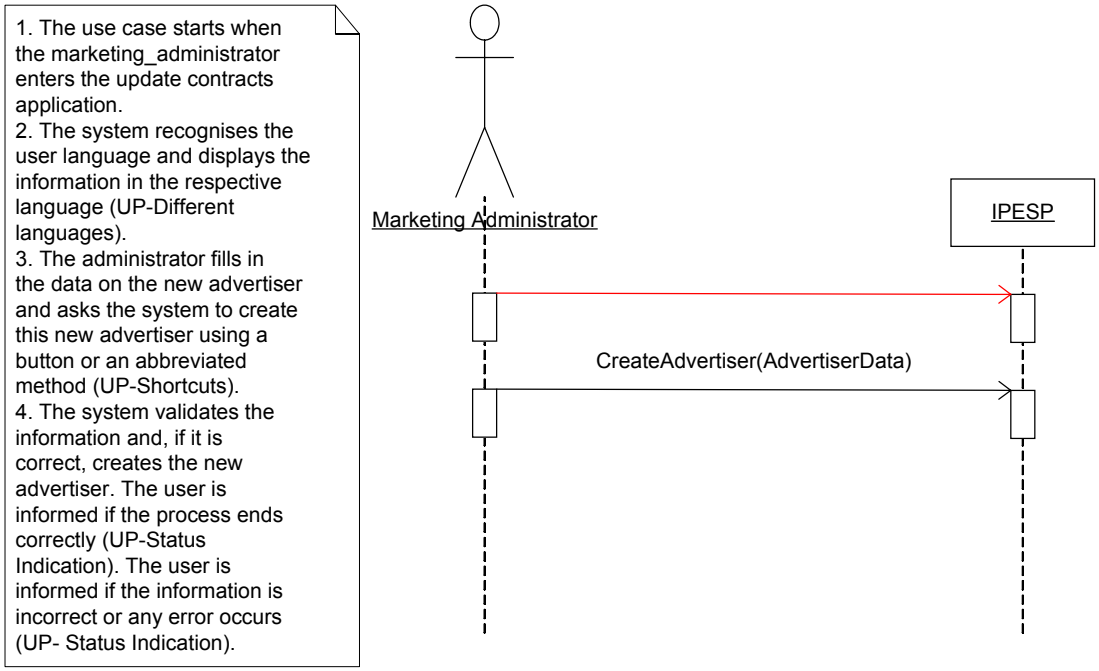

Figure 80 Sequence Diagram: Manage advertiser

E.5.1.3.4 Sequence Diagram: Manage product category

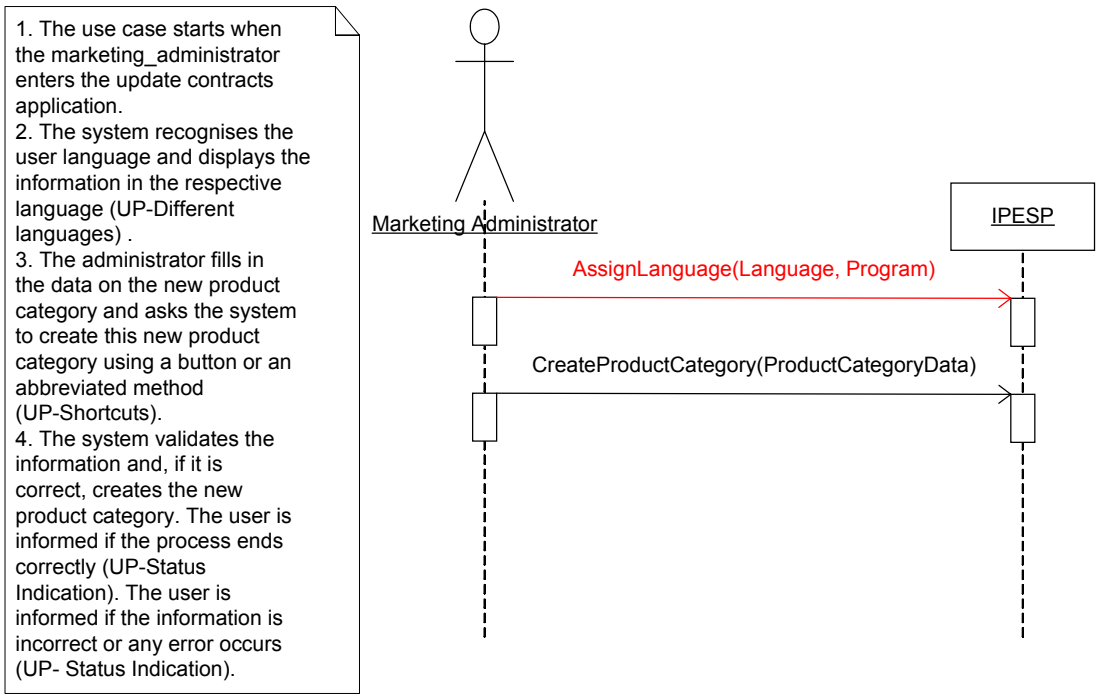

Figure 81 Sequence Diagram: Manage product category

E.5.1.3.5 Sequence Diagram: Manage product

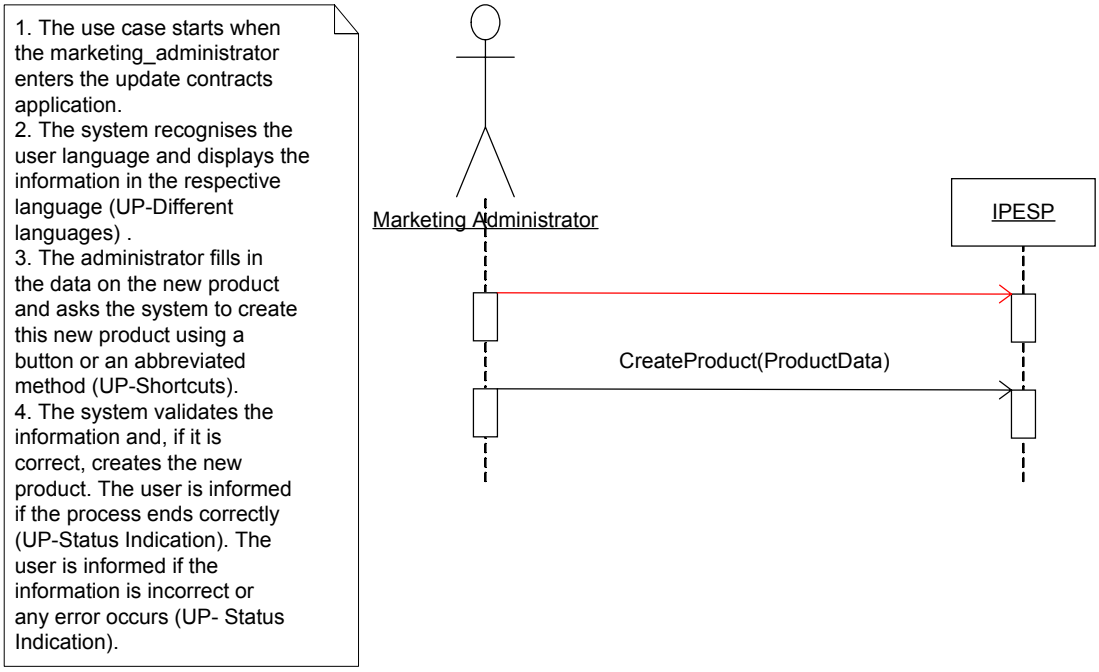

Figure 82 Sequence Diagram: Manage product

E.5.1.3.6 Sequence Diagram: Manage campaign

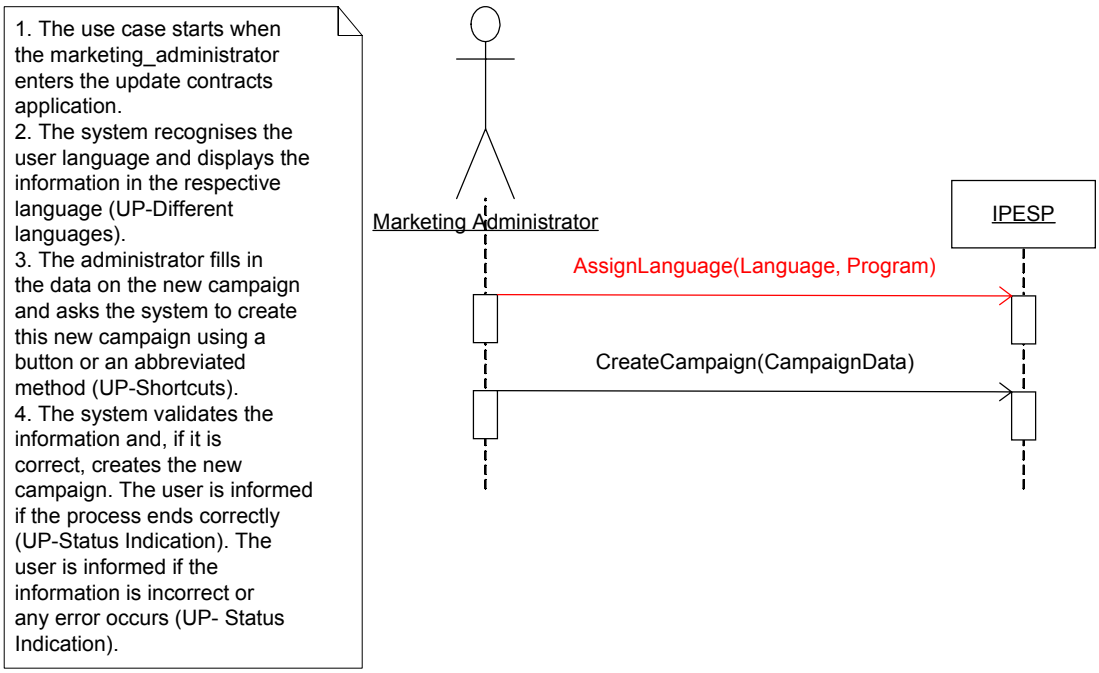

Figure 83 Sequence Diagram: Manage campaign

E.5.1.3.7 Sequence Diagram: Query campaign

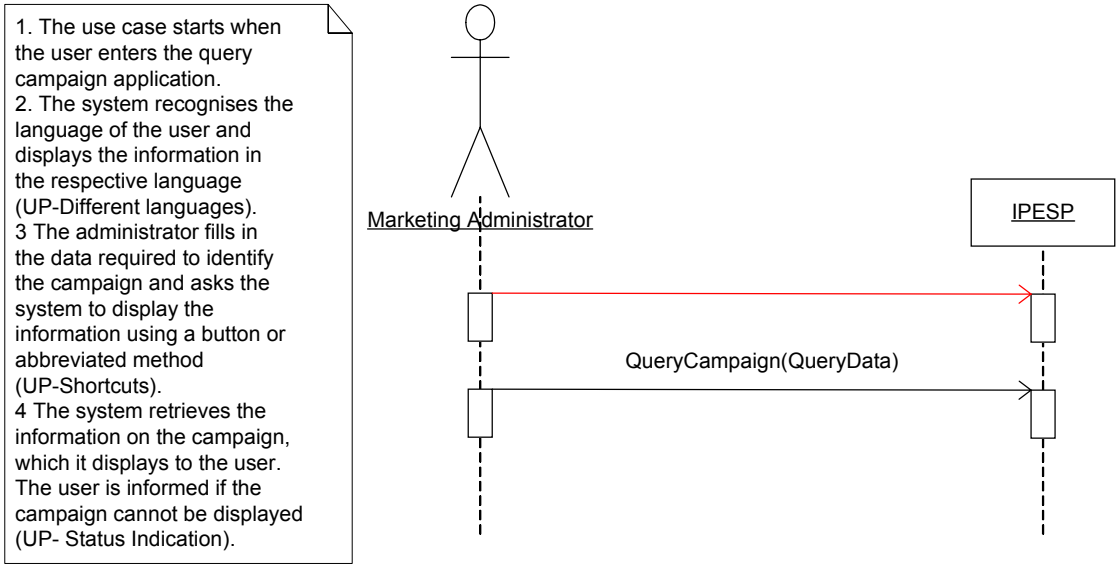

Figure 84 Sequence Diagram: Query campaign

E.5.1.3.8 Sequence Diagram: Manage contract

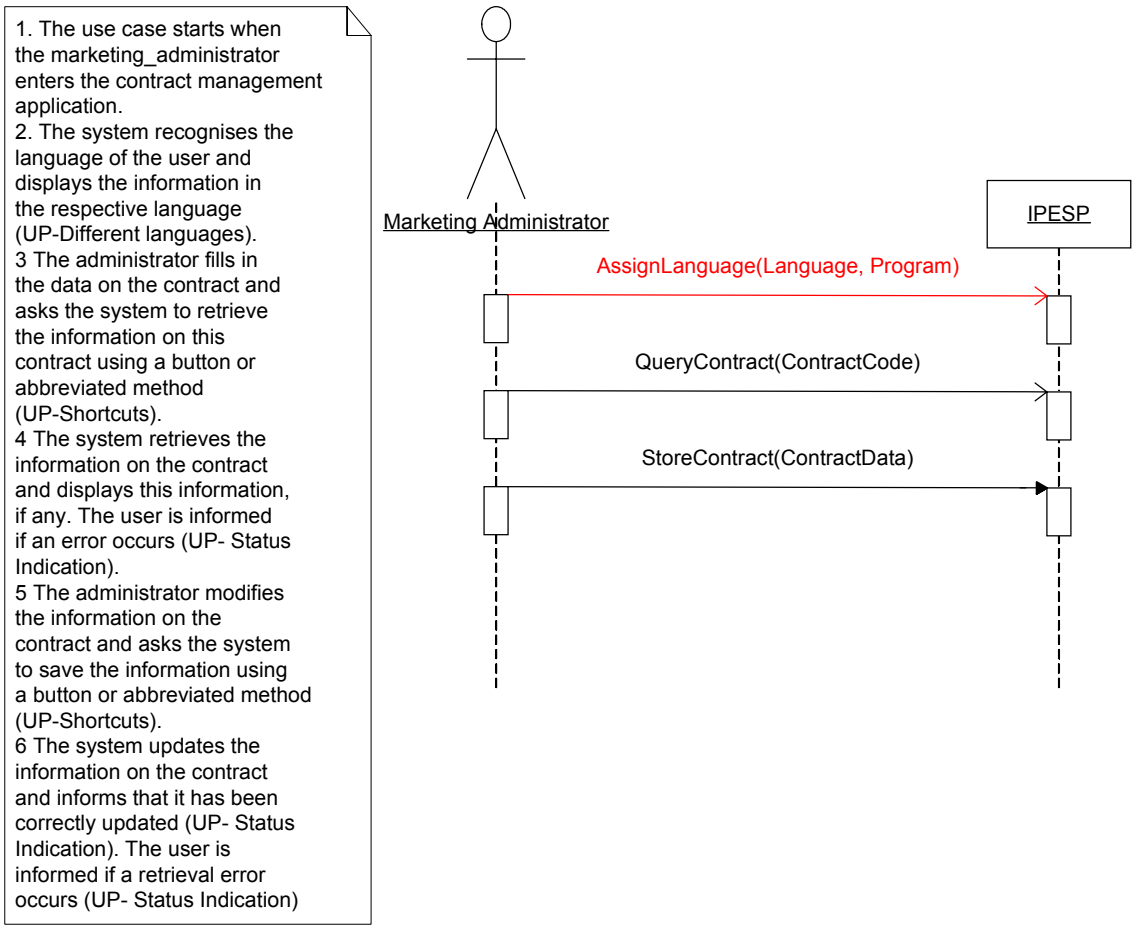

Figure 85 Sequence Diagram: Manage contract

E.5.1.3.9 Sequence Diagram: Manage contract - Alternative course 1

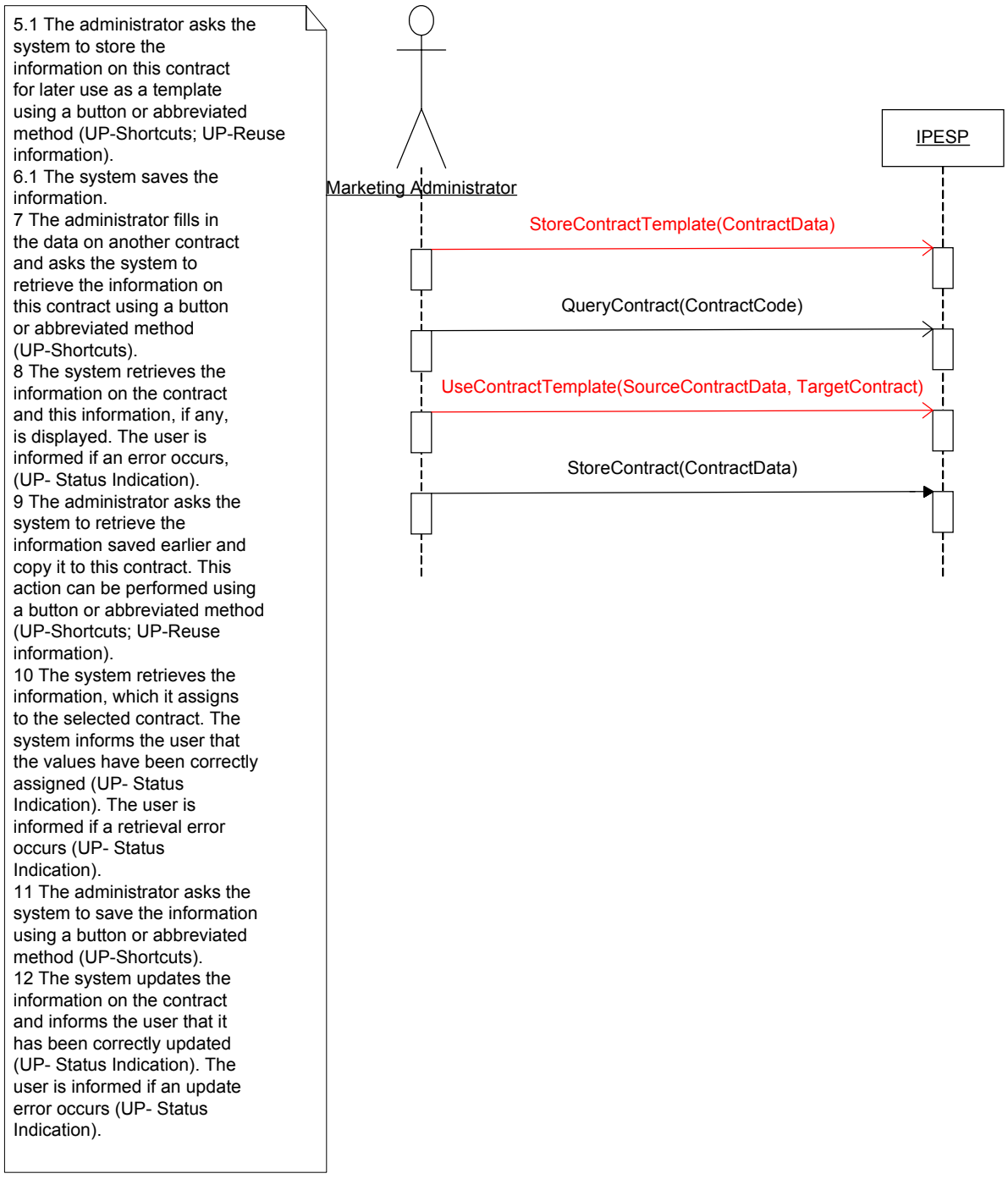

Figure 86 Sequence Diagram: Manage contract alternative course 1

E.5.1.3.10 Sequence Diagram: Manage contract - Alternative course 2

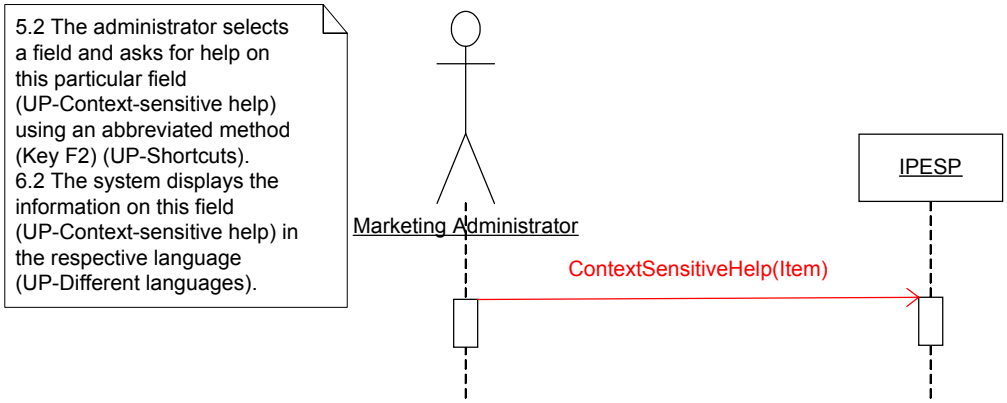

Figure 87 Sequence Diagram: Manage contract alternative course 2

E.5.1.3.11 Sequence Diagram: Manage contract - Alternative course 3

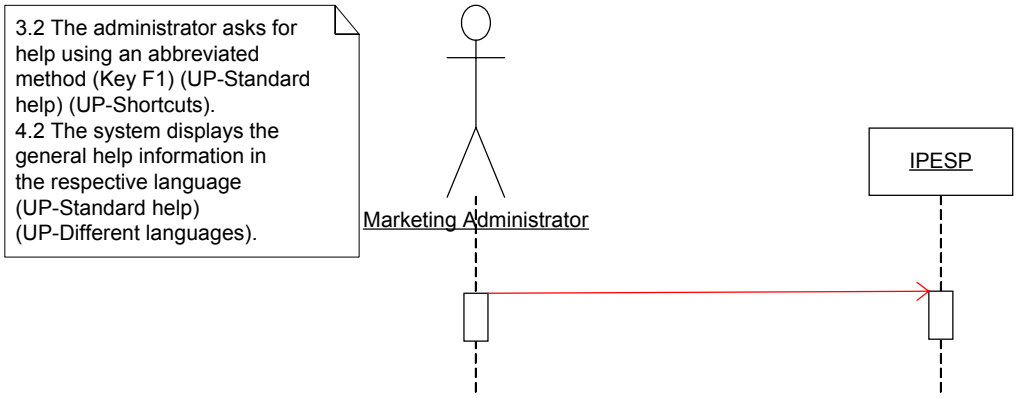

Figure 88 Sequence Diagram: Manage contract alternative course 3

E.5.1.3.12 Sequence Diagram: Query contract

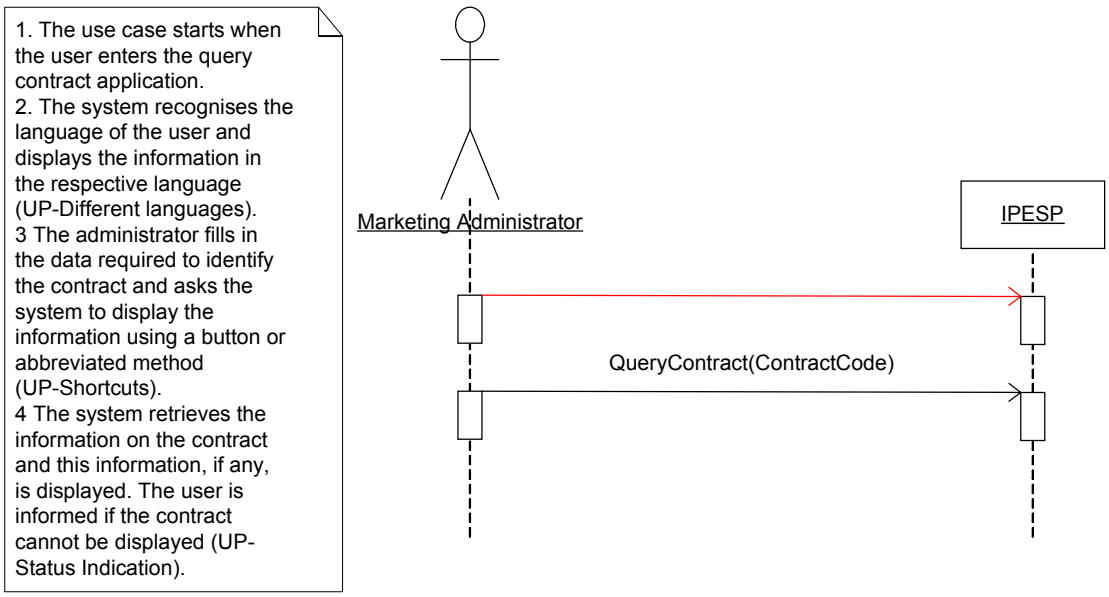

Figure 89 Sequence Diagram: Query contract

E.5.1.3.13 Sequence Diagram: Query booking

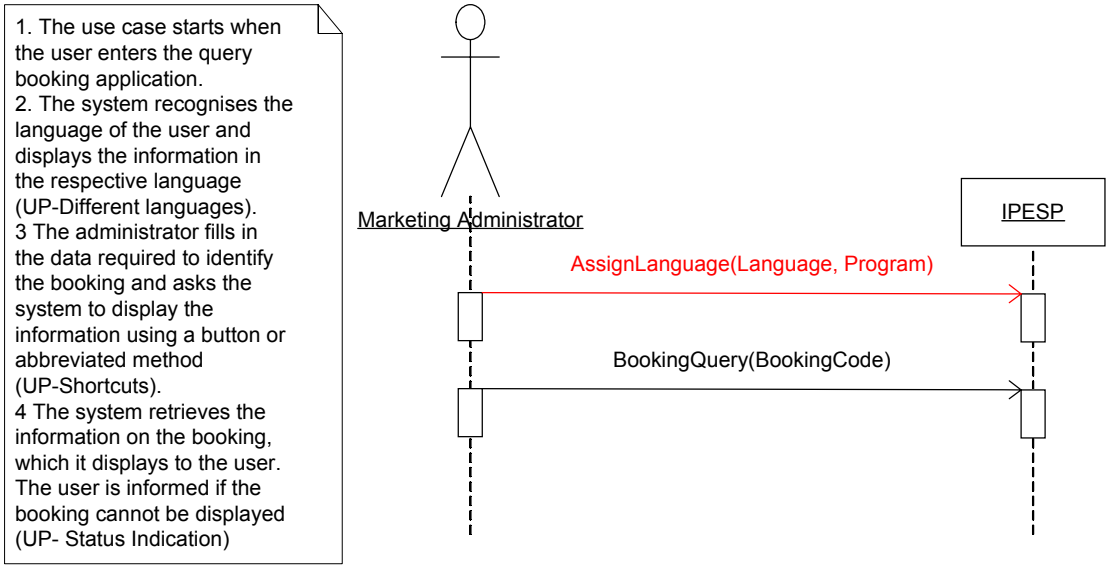

Figure 90 Sequence Diagram: Query booking

E.5.1.3.14 Sequence Diagram: Query customer

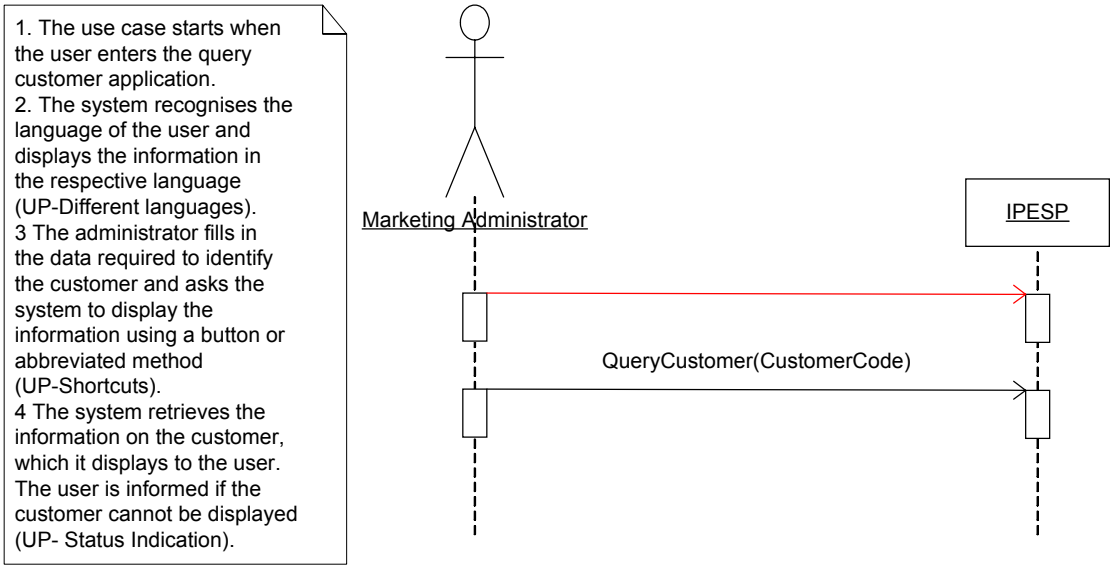

Figure 91 Sequence Diagram: Query customer

E.5.1.3.15 Sequence Diagram: Query agency

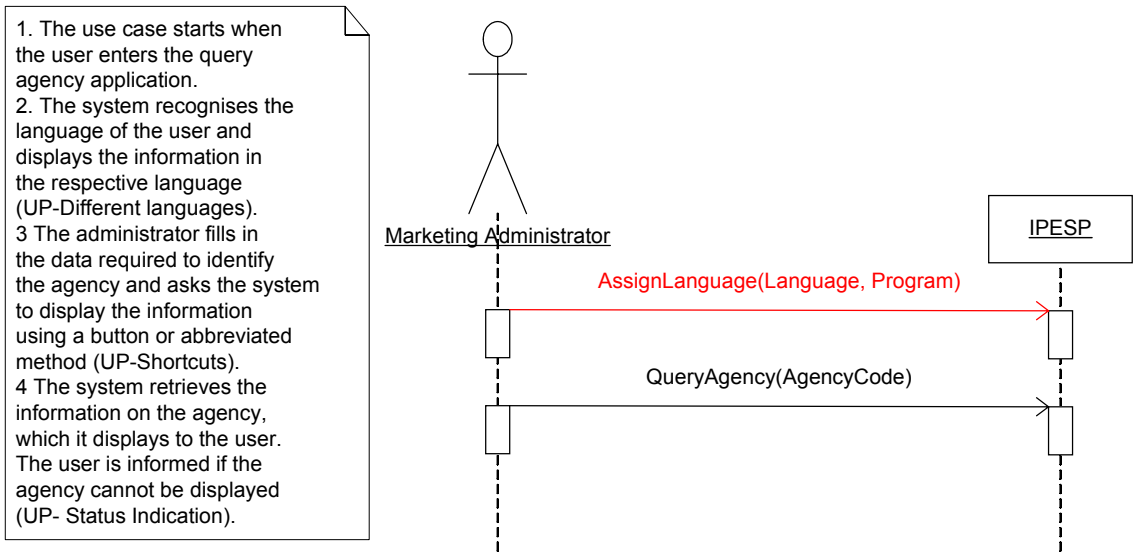

Figure 92 Sequence Diagram: query agency

E.5.1.3.16 Sequence Diagram: Get contracts report

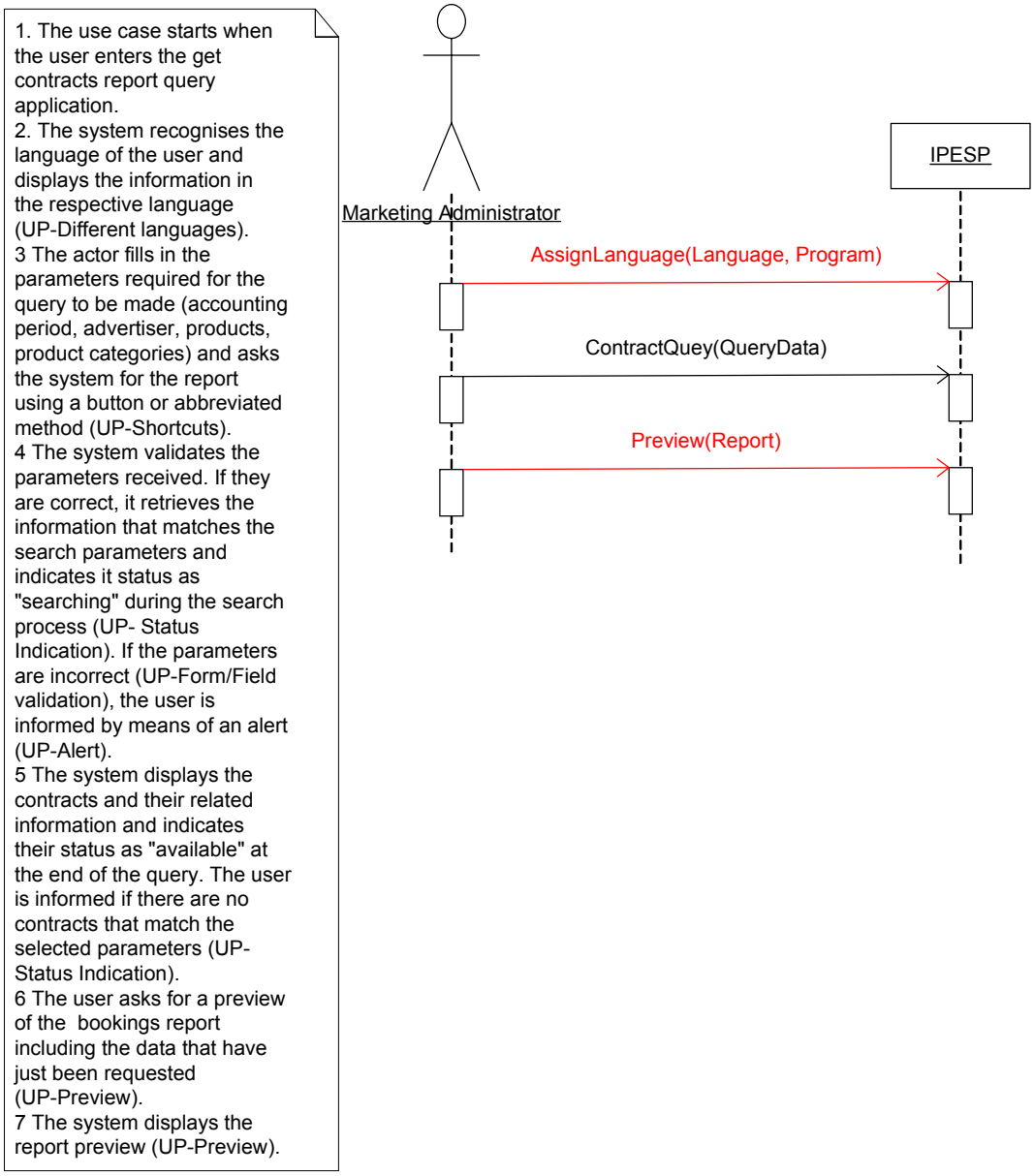

Figure 93 Sequence Diagram: Get contracts report

E.5.1.3.17 Sequence Diagram: Get contracts report – alternative course 1

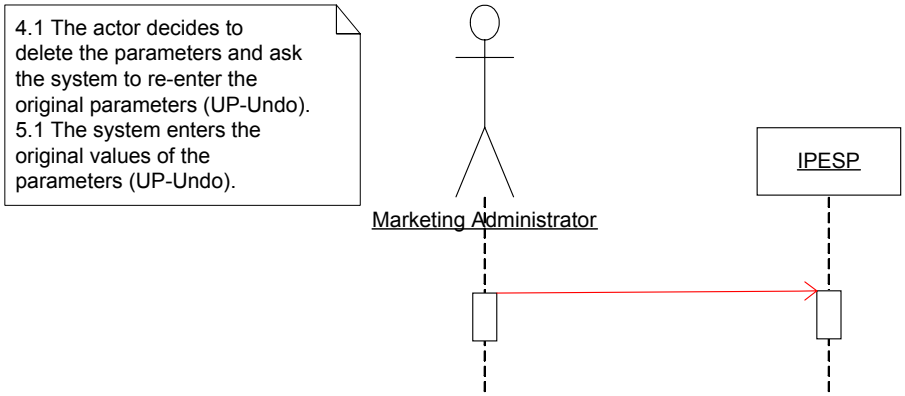

Figure 94 Sequence Diagram: Get contracts report alternative course 1

E.5.1.3.18 Sequence Diagram: Get contracts report – alternative course 2

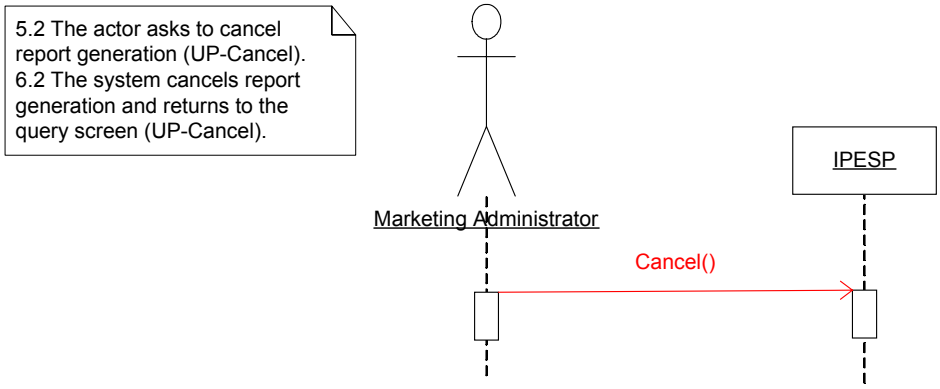

Figure 95 Sequence Diagram: Get contracts report alternative course 2

E.5.1.3.19 Sequence Diagram: Get contracts report – alternative course 3

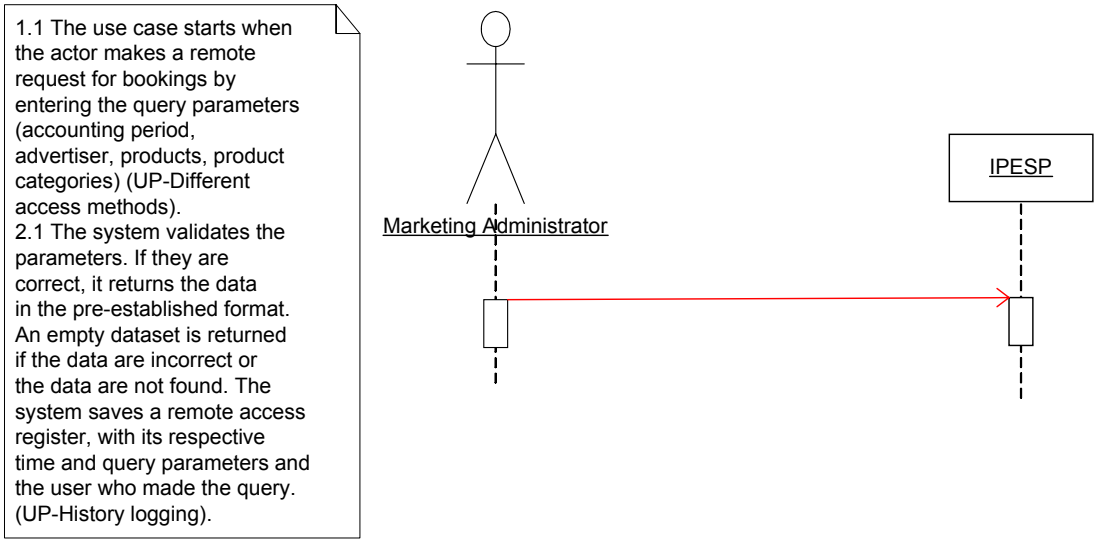

Figure 96 Sequence Diagram: Get contracts report alternative course 3

E.5.1.3.20 Sequence Diagram: Get contracts report – alternative course 4

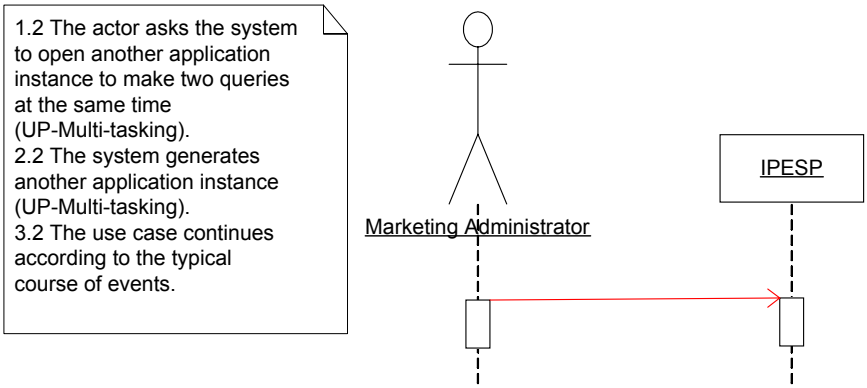

Figure 97 Sequence Diagram: Get contracts report alternative course 4

E.5.1.3.21 Sequence Diagram: Export report

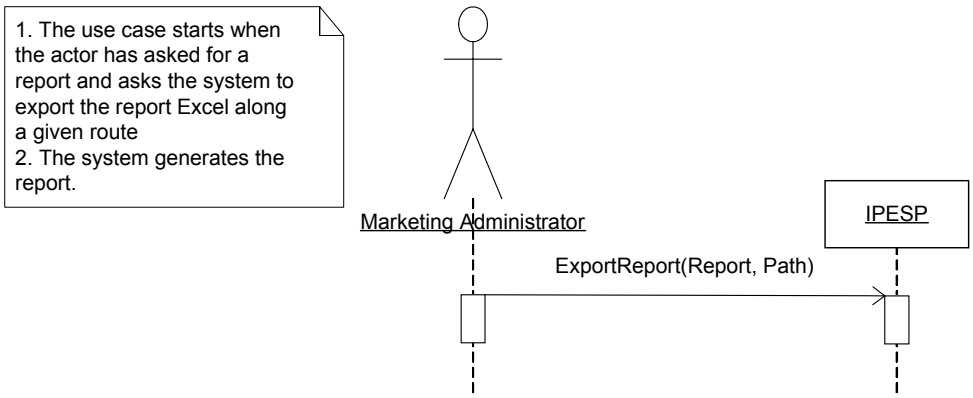

Figure 98 Sequence Diagram: Export report

E.5.1.3.22 Sequence Diagram: Query affixation job tickets

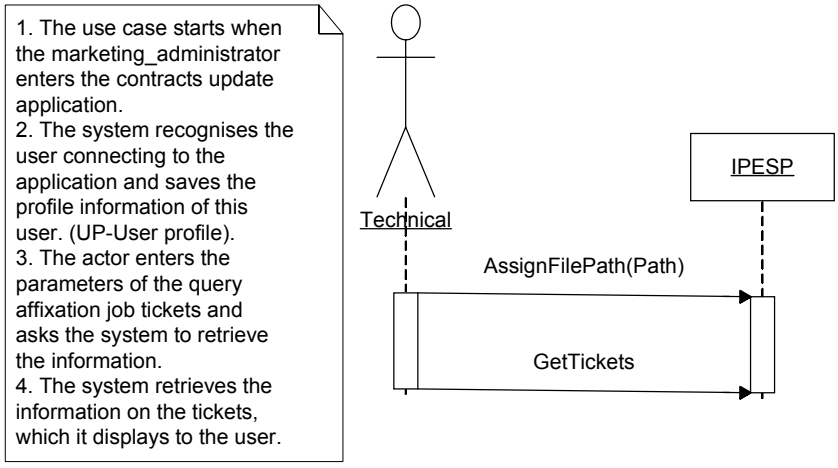

Figure 99 Sequence Diagram: Query affixation job tickets

E.5.1.3.23 Sequence Diagram: Change ticket status

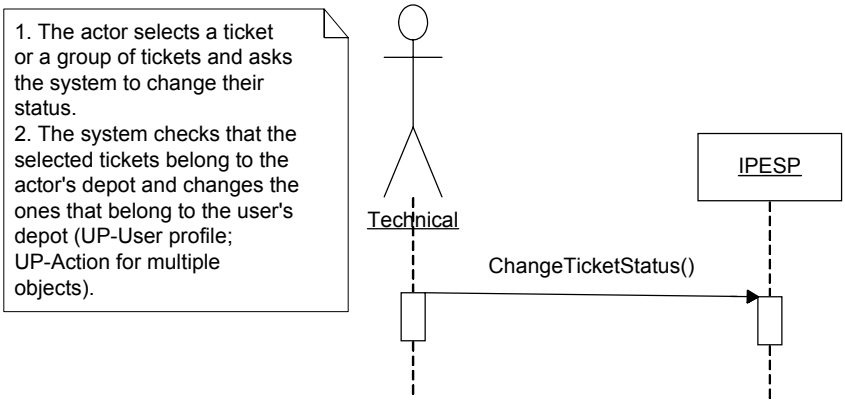

Figure 100 Sequence Diagram: Change ticket status

E.5.1.3.24 Sequence Diagram: Query affixation job

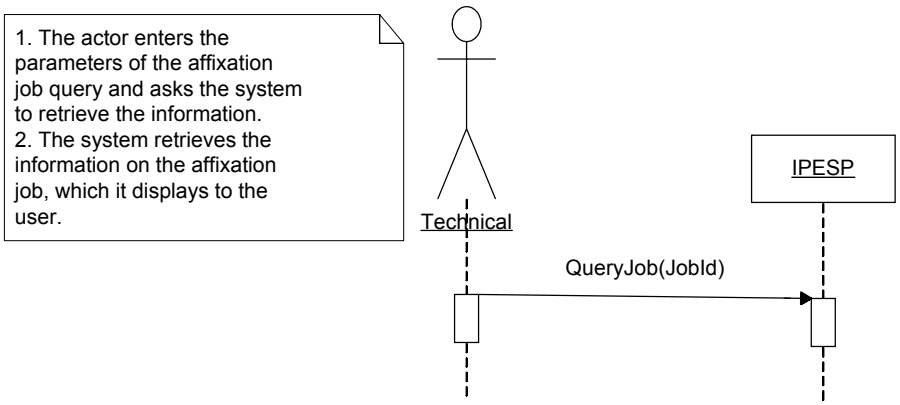

Figure 101 Sequence Diagram: Query affixation job

E.5.1.3.25 Sequence Diagram: Change affixation job data

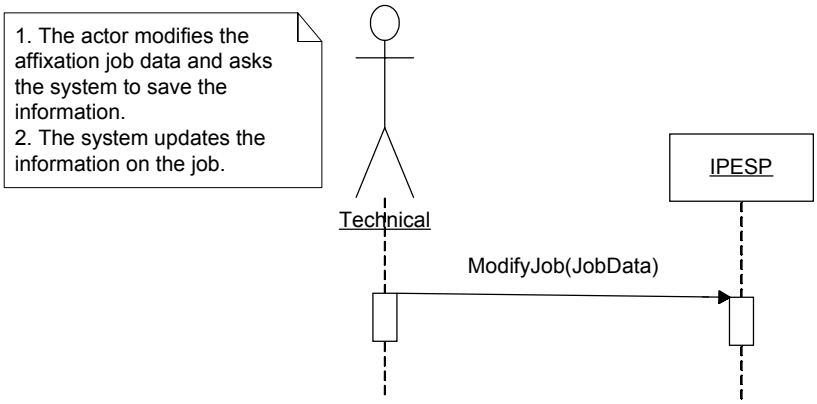

Figure 102 Sequence Diagram: Change affixation job data

E.5.1.3.26 Sequence Diagram: Get affixation job report

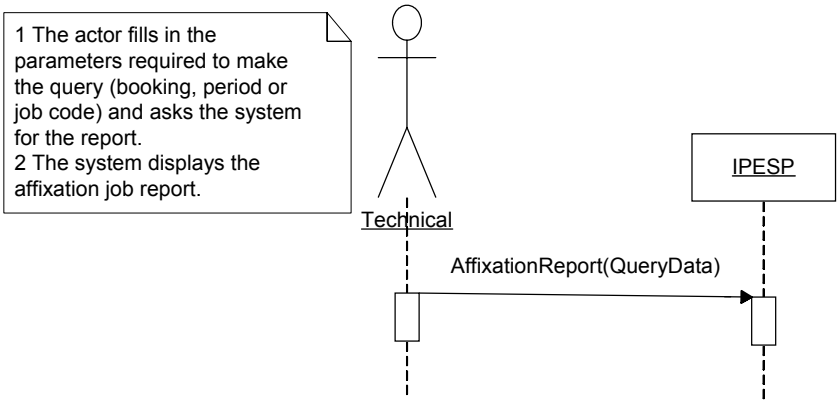

Figure 103 Sequence Diagram: Get affixation job report

E.5.1.4 Operations Contracts

The operations contracts specify each of the operations presented in the sequence diagrams and the system behaviour in more detail. The following information can be defined for each contract: name, responsibilities, type, cross-references, notes, exceptions, output, preconditions and postconditions, although not all the fields are compulsory.

The contracts that are related to usability patterns will be highlighted in orange.

E.5.1.4.1 Sequence diagram contracts: Update contracts

This sequence diagram contains the following contracts:

- AssignLanguage.
- UpdateContracts

Name:	***AssignLanguage (Language:String, Application)***
Responsibilities:	Assign the language in which the application is to be viewed.
Cross-references:	Requirements: P11 Use case: Update contracts
Notes:	
Exceptions:	The language cannot be assigned.
Output:	
Pre-conditions:	
Post-conditions:	The default application language has been assigned.

Name:	***UpdateContracts()***
Responsibilities:	Retrieve information on contracts and read the Phoenix database to update this information.
Cross-references:	Requirements: P2, P3, P4, P6, P7, P8, P9, P11, P13 Use case: Update contracts
Notes:	
Exceptions:	No contracts exist
Output:	
Pre-conditions:	
Post-conditions:	The information on contracts has been updated.

E.5.1.4.2 Sequence diagram contracts: Manage advertising group

This sequence diagram contains the following contracts:

- CreateAdvertisingGroup

Name:	*CreateAdvertisingGroup(AdvertisingGroupData:Object)*
Responsibilities:	Create a new advertising group
Cross-references:	Requirements: P5, P10, P11, P13, P14 Use case: Manage advertising group
Notes:	
Exceptions:	The advertising group already exists
Output:	
Pre-conditions:	
Post-conditions:	The advertising group has been created

E.5.1.4.3 Sequence diagram contracts: Manage advertiser

This sequence diagram contains the following contracts:

- CreateAdvertiser

Name:	*CreateAdvertiser(AdvertiserData:Object)*
Responsibilities:	Create a new advertiser
Cross-references:	Requirements P5 Use case: Manage advertiser
Notes:	
Exceptions:	The advertiser already exists
Output:	
Pre-conditions:	
Post-conditions:	The advertiser has been created

E.5.1.4.4 Sequence diagram contracts: Manage product category

This sequence diagram contains the following contracts:

- CreateProductCategory

Name:	*CreateProductCategory(ProductCategoryData:Object)*
Responsibilities:	Create a new product category
Cross-references:	Requirements P5 Use case: Manage product category

Notes:	
Exceptions:	The product category already exists
Output:	
Pre-conditions:	
Post-conditions:	The requested product category has been created

E.5.1.4.5 Sequence diagram contracts: Manage product

This sequence diagram contains the following contracts:

- CreateProduct

Name:	***CreateProduct(ProductData:Object)***
Responsibilities:	Create a new product
Cross-references:	Requirements P5 Use case: Manage product
Notes:	
Exceptions:	The product already exists; the related data do not exist
Output:	
Pre-conditions:	
Post-conditions:	The requested product has been created

E.5.1.4.6 Sequence diagram contracts: Manage campaign

This sequence diagram contains the following contracts:

- CreateCampaign

Name:	***CreateCampaign(CampaignData:Object)***
Responsibilities:	Create a new campaign
Cross-references:	Requirements P5, P2 Use case: Manage campaign
Notes:	
Exceptions:	The campaign already exists
Output:	
Pre-conditions:	

Post-conditions:	The requested campaign has been created

E.5.1.4.7 Sequence diagram contracts: Query campaign

This sequence diagram contains the following contracts:

- QueryCampaign

Name:	***CreateCampaign(QueryData:Object)***
Responsibilities:	Retrieve and display data on a campaign
Cross-references:	Requirements P2, P5 Use case: Query campaign
Notes:	
Exceptions:	The campaign does not exist
Output:	
Pre-conditions:	
Post-conditions:	The campaign data have been retrieved and displayed

E.5.1.4.8 Sequence diagram contracts: Manage contract

This sequence diagram contains the following contracts:

- QueryContract
- SaveContract

Name:	***QueryContract(ContractCode:int)***
Responsibilities:	Retrieve and display data on a contract
Cross-references:	Requirements P2, P10 Use case: Manage contract
Notes:	
Exceptions:	The contract does not exist
Output:	
Pre-conditions:	
Post-conditions:	The contract data have been retrieved and displayed

Name:	***SaveContract(ContractData:Object)***

Responsibilities:	Save the data on a contract for persistence
Cross-references:	Requirements A5 and A6 Use case: Manage contract
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The data on the contract have been saved

E.5.1.4.9 Sequence diagram contracts: Manage contract - Alternative course 1

This sequence diagram contains the following contracts:

- SaveTemplateContract
- UseContractTemplate

Name:	***SaveTemplateContract(ContractData:Object)***
Responsibilities:	Save the information on a contract for use as a template for assigning values to a new contract
Cross-references:	Requirements A5 and A6 Use case: Manage contract
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information has been saved for later use

Name:	***UseContractTemplate(SourceContractData:Object)***
Responsibilities:	Copy the information from a template contract to the target contract.
Cross-references:	Requirements A5 and A6 Use case: Manage contract

Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information from the source contract has been copied to the target contract

E.5.1.4.10 Sequence diagram contracts: Manage contract - Alternative course 2

This sequence diagram contains the following contracts:

- ContextSensitiveHelp

Name:	***ContextSensitiveHelp(Item:String)***
Responsibilities:	Display the information on a given field of the application to explain how to use this item and its permitted values
Cross-references:	Requirements A5 and A6 Use case: Manage contract
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information the selected item is displayed.

E.5.1.4.11 Sequence diagram contracts: Manage contract - Alternative course 3

This sequence diagram contains the following contracts:

- StandardHelp

Name:	***StandardHelp()***
Responsibilities:	Display the standard application information.
Cross-references:	Requirements A5 and A6 Use case: Manage contract

Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The standard application information is displayed

E.5.1.4.12 Sequence diagram contracts: Query contract

This sequence diagram contains the following contracts:

- QueryContract

E.5.1.4.13 Sequence diagram contracts: Query booking

This sequence diagram contains the following contracts:

- QueryBooking

Name:	***QueryBooking(BookingCode:int)***
Responsibilities:	Retrieve and display booking data
Cross-references:	Requirements A5 and A6 Use case: Query booking
Notes:	
Exceptions:	The booking does not exist
Output:	
Pre-conditions:	
Post-conditions:	They booking data have been retrieved and displayed

E.5.1.4.14 Sequence diagram contracts: Query customer

This sequence diagram contains the following contracts:

- QueryCustomer

Name:	***QueryCustomer(CustomerCode:int)***
Responsibilities:	Retrieve and display the data on a customer
Cross-references:	Requirements A5 and A6 Use case: Query customer
Notes:	

Exceptions:	The customer does not exist
Output:	
Pre-conditions:	
Post-conditions:	The data on the customer have been retrieved and displayed

E.5.1.4.15 Sequence diagram contracts: Query agency

This sequence diagram contains the following contracts:

- QueryAgency

Name:	***QueryAgency(AgencyCode:int)***
Responsibilities:	Retrieve and display agency data
Cross-references:	Requirements A5 and A6 Use case: Query agency
Notes:	
Exceptions:	The agency does not exist
Output:	
Pre-conditions:	
Post-conditions:	The agency data have been retrieved and displayed

E.5.1.4.16 Sequence diagram contracts: Get contracts report

This sequence diagram contains the following contracts:

- QueryContracts
- Preview

Name:	***QueryContracts(QueryData:int)***
Responsibilities:	Retrieve the contracts that match the query parameters and display with the required information
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	There are no contracts that match the query parameters
Output:	

Pre-conditions:	
Post-conditions:	The contracts have been retrieved and displayed

E.5.1.5

Name:	***Preview(report:Object)***
Responsibilities:	Create a print preview of the contracts report
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The print preview of the report has been displayed

E.5.1.5.1 Sequence diagram contracts: Get contracts report - alternative course 1

This sequence diagram contains the following contracts:

- Undo

Name:	***Undo()***
Responsibilities:	Undo the changes made to the query parameters and return to previous states
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	It has returned to the previous state

E.5.1.5.2 Sequence diagram contracts: Get contracts report - alternative course 2

This sequence diagram contains the following contracts:

- Cancel

Name:	***Cancel()***
Responsibilities:	Cancel the requested report generation and return to the original application state
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The contracts report generation process has been cancelled and the application is in its original state

E.5.1.5.3 Sequence diagram contracts: Get contracts report - alternative course 3

This sequence diagram contains the following contracts:

- GetRemoteReport

Name:	***GetRemoteReport(QueryData:Object)***
Responsibilities:	Retrieve the reports that match the search parameters requested remotely for processing by another application
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The requested data have been returned and the access log has been saved

E.5.1.5.4 Sequence diagram contracts: Get contracts report - alternative course 4

This sequence diagram contains the following contracts:

- NewApplicationInstance

Name:	*NewApplicationInstance()*
Responsibilities:	Create a new application instance to make multiple queries simultaneously
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	A new instance of the application has been created

E.5.1.5.5 Sequence diagram contracts: Export report

This sequence diagram contains the following contracts:

- ExportReport

Name:	*ExportReport(Report:Object, Route:String)*
Responsibilities:	Export a contracts report to a file in Microsoft Excel format
Cross-references:	Requirements A5 and A6 Use case: Get contracts report
Notes:	
Exceptions:	The route is invalid or cannot be written to
Output:	
Pre-conditions:	The report has been generated
Post-conditions:	A new instance of the application has been created

E.5.1.5.6 Sequence diagram contracts: Query affixation job tickets

This sequence diagram contains the following contracts:

- StoreProfile
- GetTickets

Name:	*StoreProfile (User)*
Responsibilities:	Detect the information on the user who has connected and save the profile for use in the functions for which it is required

	the profile for use in the functions for which it is required.
Cross-references:	Requirements: R8 Use case: Query affixation job tickets
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The profile of the connected user has been saved.

Name:	***GetTickets(QueryData: Object)***
Responsibilities:	Retrieve affixation job tickets.
Cross-references:	Requirements: R3, R5 Use case: Query affixation job tickets
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The tickets for the job in question have been retrieved.

E.5.1.5.7 Sequence diagram contracts: Change ticket status

This sequence diagram contains the following contracts:

- ChangeTicketStatus

Name:	***ChangeTicketStatus(tickets: Object)***
Responsibilities:	Change the status of a series of tickets.
Cross-references:	Requirements: R1, R7, R8 Use case: Change ticket status
Notes:	
Exceptions:	

Output:	
Pre-conditions:	
Post-conditions:	The status of the selected tickets that belong to the user's depot has been changed.

E.5.1.5.8 Sequence diagram contracts: Query affixation job

This sequence diagram contains the following contracts:

- QueryJob

Name:	***QueryJob(JobID: Long)***
Responsibilities:	Retrieve the information on an affixation job.
Cross-references:	Requirements: R5 Use case: Query affixation job.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information on an affixation job has been retrieved.

E.5.1.5.9 Sequence diagram contracts: Change affixation job data

This sequence diagram contains the following contracts:

- ModifyJob

Name:	***ModifyJob(JobData: Object)***
Responsibilities:	Update the information on an affixation job.
Cross-references:	Requirements: R2 Use case: Change affixation job data.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The information on the affixation job has been modified.

E.5.1.5.10 Sequence diagram contracts: Get affixation job report

This sequence diagram contains the following contracts:

- AffixationReport

Name:	***AffixationReport(QueryData: Object)***
Responsibilities:	Display a report on the status of an affixation job.
Cross-references:	Requirements: R4, R6 Use case: Get affixation job report.
Notes:	
Exceptions:	
Output:	
Pre-conditions:	
Post-conditions:	The report with the status of an affixation job is displayed.

Development cycle 1 design

E.5.1.6 Interaction diagrams

Below, we illustrate the operations in the sequence diagrams constructed during the analysis phase by means of interaction diagrams.

E.5.1.6.1 Sequence diagram: AssignLanguage(language:string, application:string)

Figure 5.7 shows the sequence diagram that illustrates a get user operation.

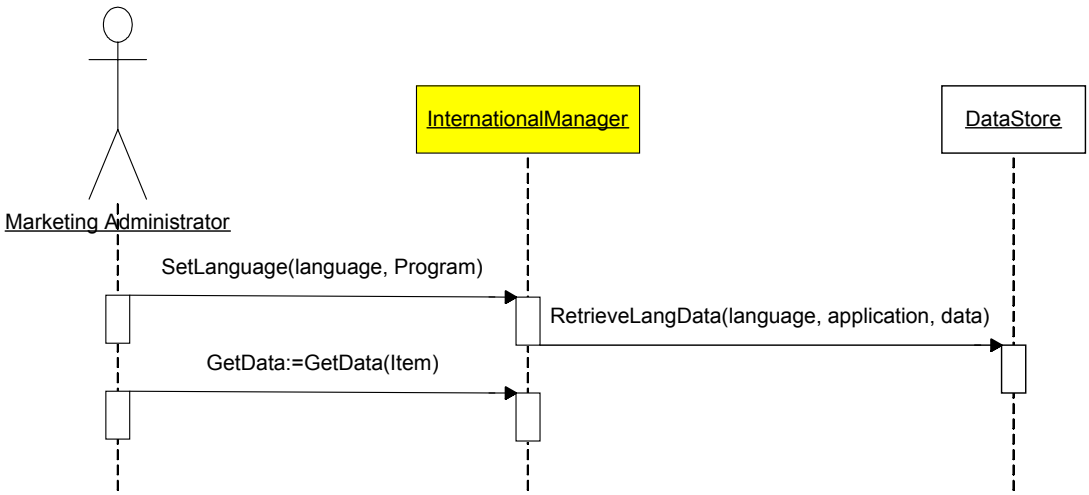

Figure 104 Sequence Diagram: AssignLanguage

E.5.1.6.2 Sequence diagram: UpdateContracts()

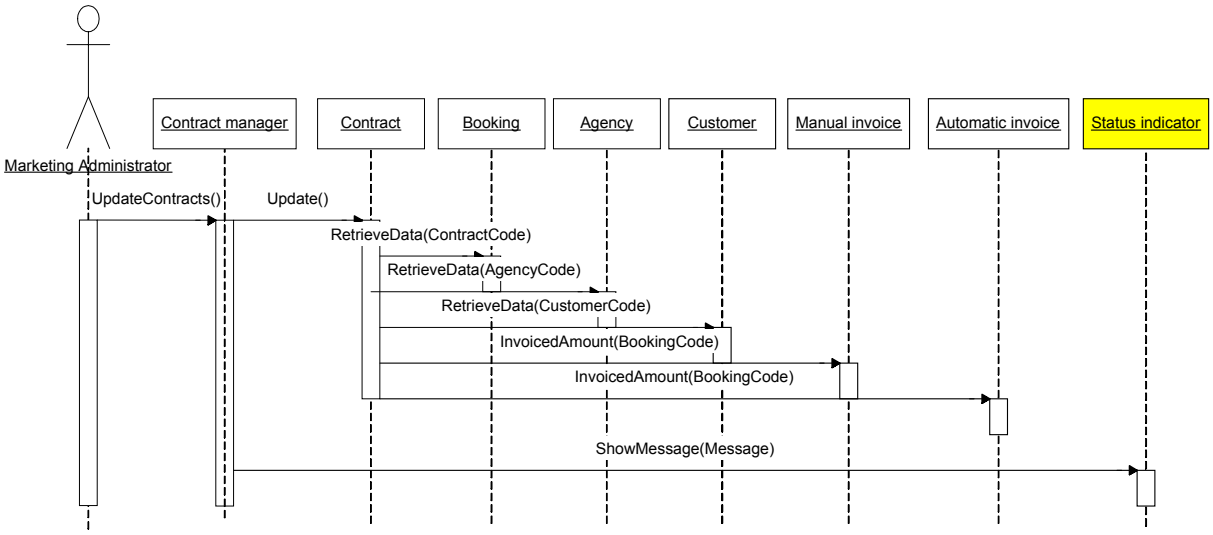

Figure 105 Sequence Diagram: UpdateContracts

E.5.1.6.3 Sequence diagram: CreateAdvertisingGroup(AdvertisingGroupData:Object)

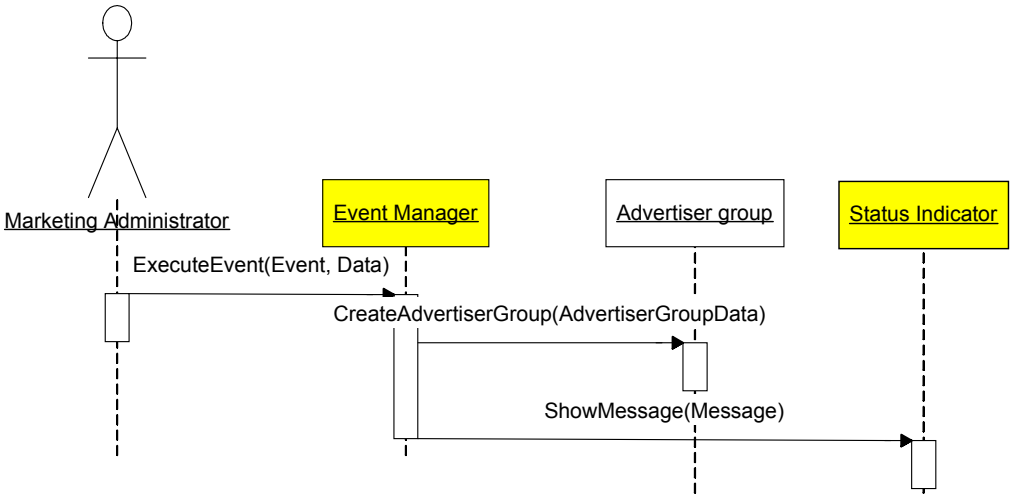

Figure 106 Sequence Diagram: CreateAdvertisingGroup

E.5.1.6.4 Sequence diagram: CreateAdvertiser(AdvertiserData:Object)

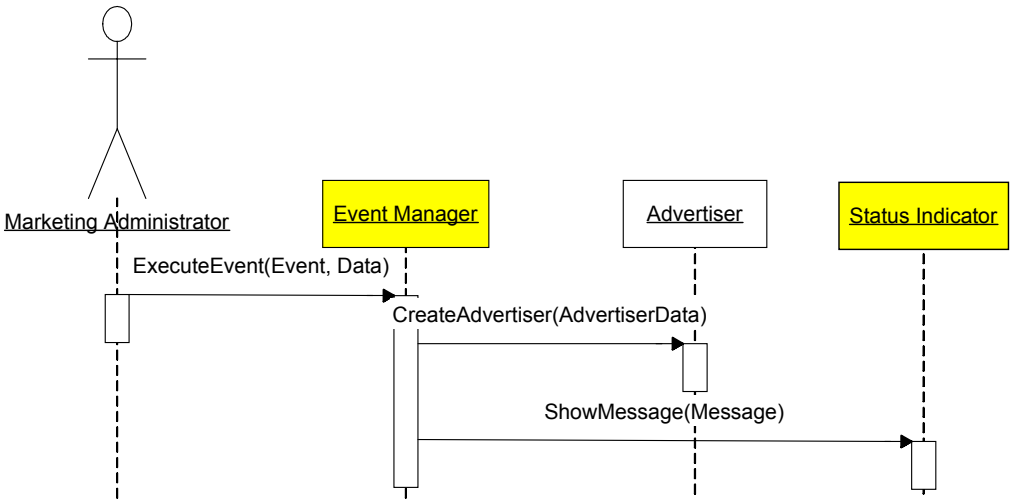

Figure 107 Sequence Diagram: CreateAdvertiser

E.5.1.6.5 Sequence diagram: CreateProductCategory(ProductCategoryData:Object)

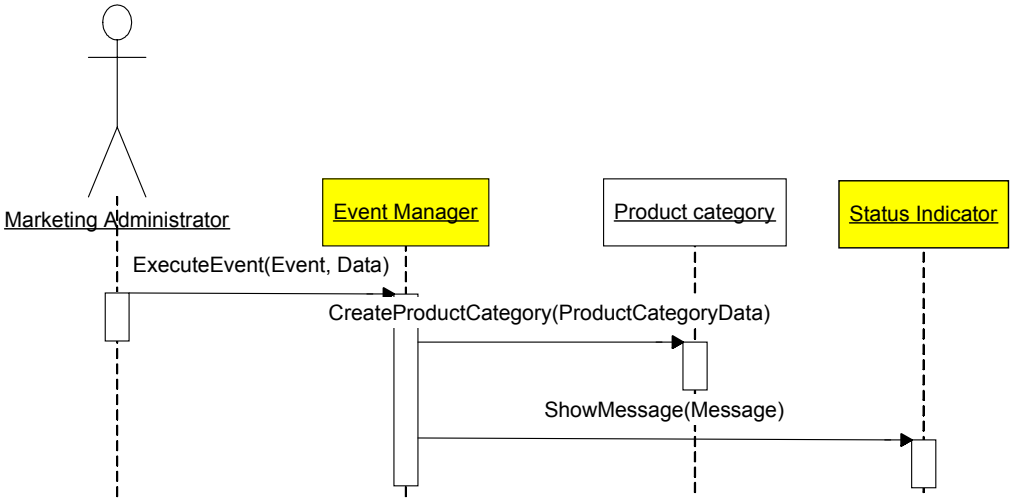

Figure 108 Sequence Diagram: CreateProductCategory

E.5.1.6.6 Sequence diagram: CreateProduct(ProductData:Object)

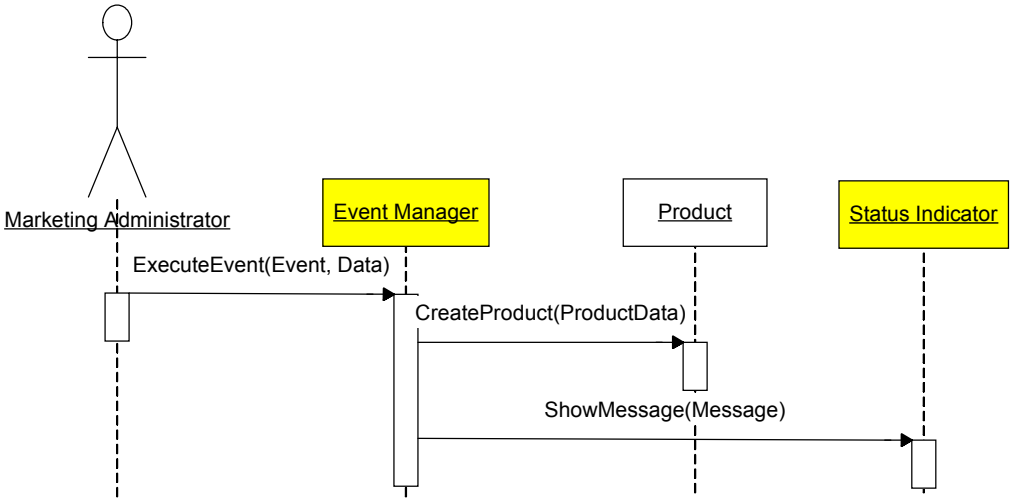

Figure 109 Sequence Diagram: CreateProduct

E.5.1.6.7 Sequence diagram: CreateCampaign(CampaignData:Object)

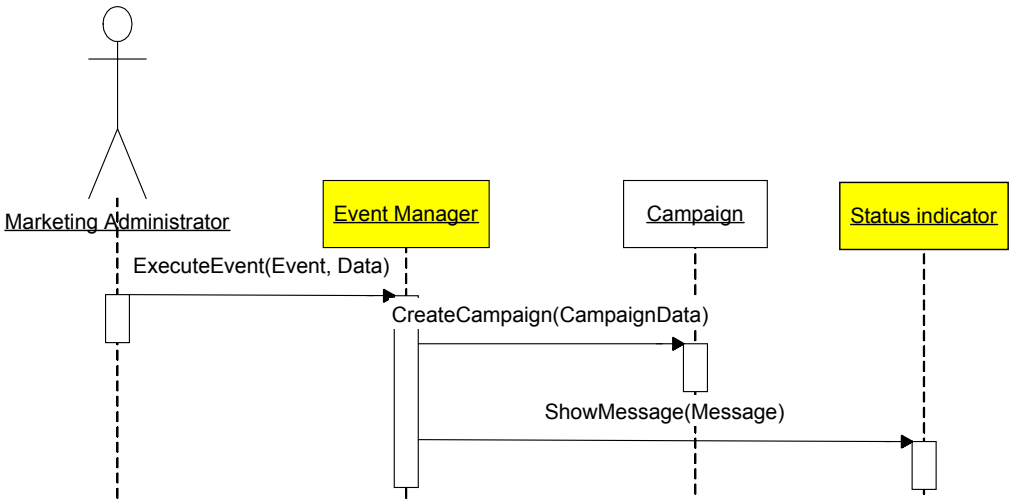

Figure 110 Sequence Diagram: CreateCampaign

E.5.1.6.8 Sequence diagram: QueryCampaign(QueryData:Object)

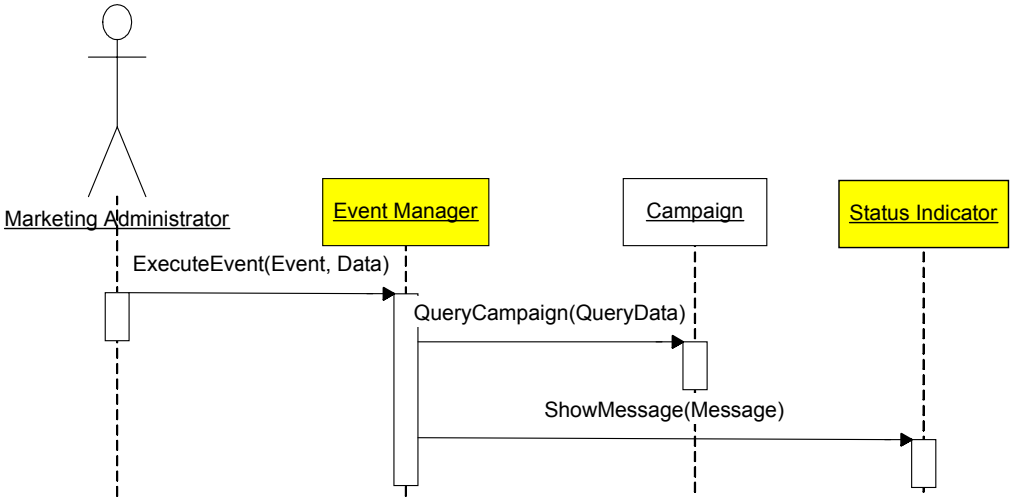

Figure 111 Sequence Diagram: QueryCampaign

E.5.1.6.9 Sequence diagram: QueryContract(ContractCode:int)

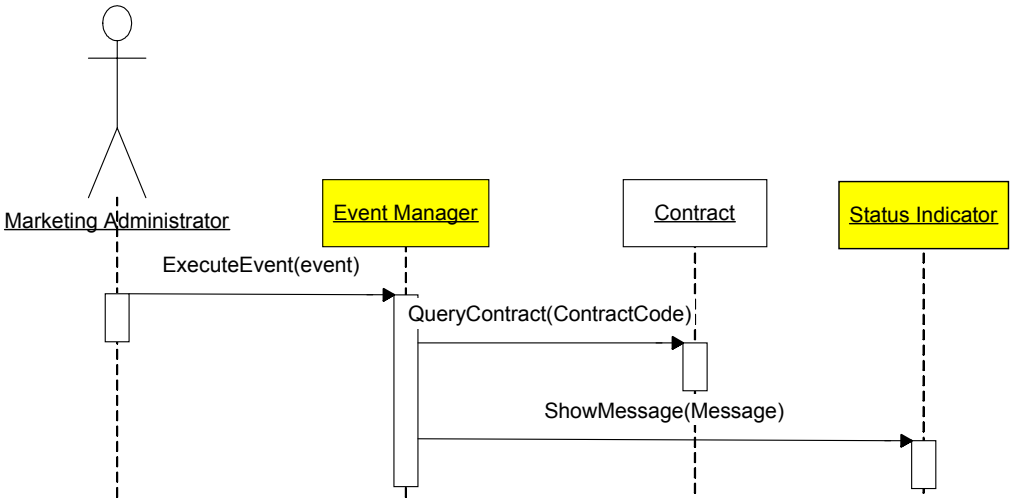

Figure 112 Sequence Diagram: QueryContract

E.5.1.6.10 Sequence diagram: SaveContract(ContractData:Object)

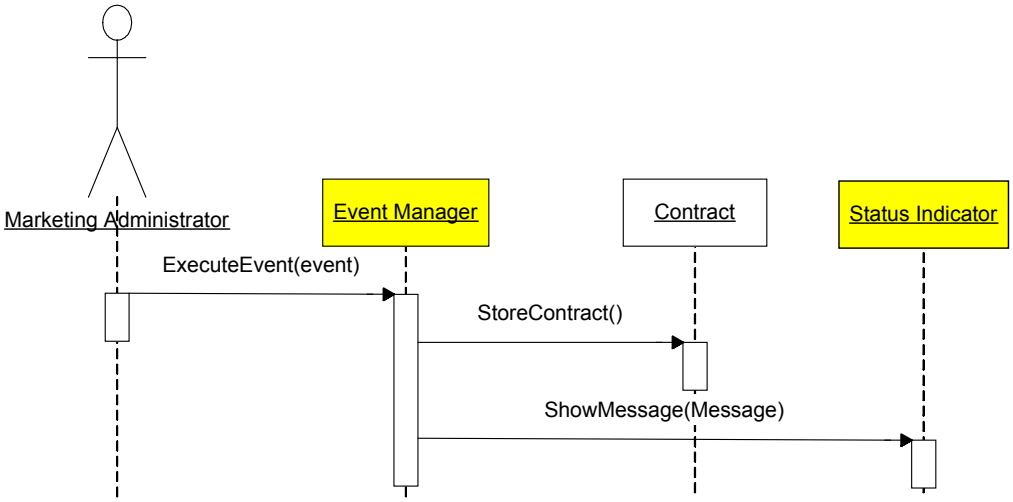

Figure 113 Sequence Diagram: SaveContract

E.5.1.6.11 Sequence diagram: SaveTemplateContract (ContractData:Object)

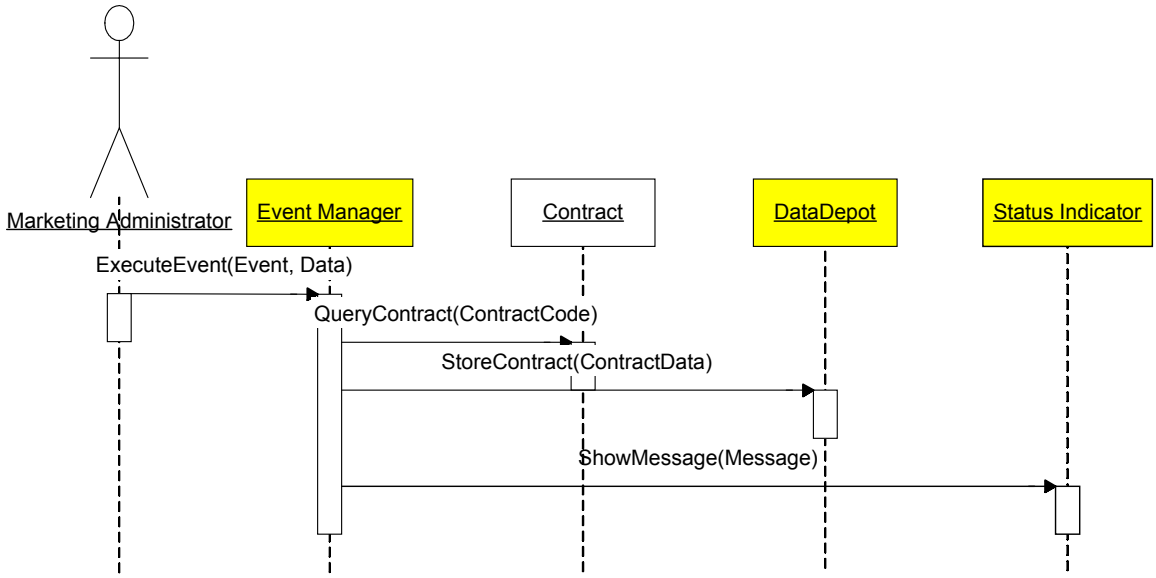

Figure 114 Sequence Diagram: SaveTemplateContract

E.5.1.6.12 Sequence diagram: UseTemplateContract(SourceContractData:Object, TargetContract:Object)

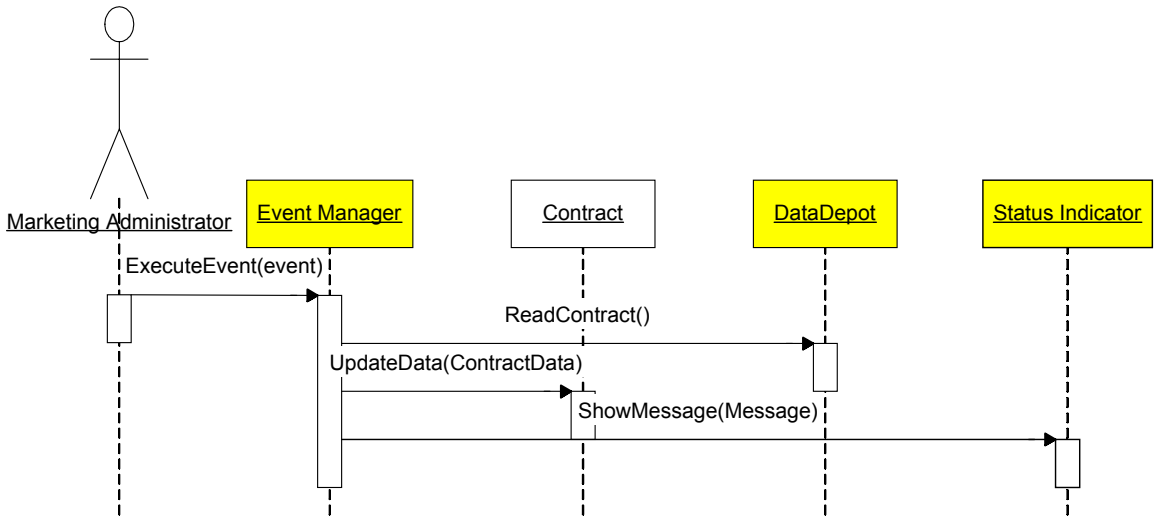

Figure 115 Sequence Diagram: UseTemplateContract

E.5.1.6.13 Sequence diagram: ContextSensitiveHelp(Item:String)

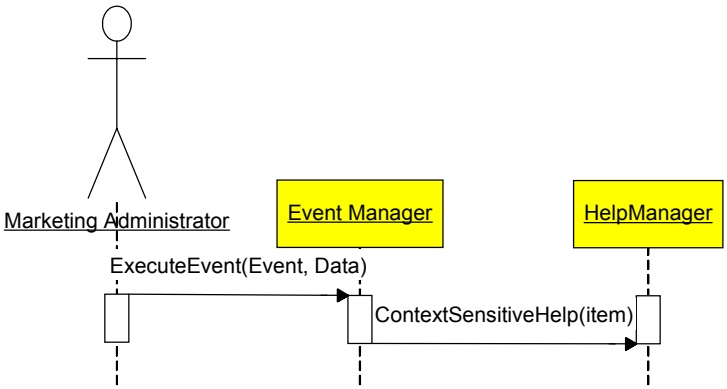

Figure 116 Sequence Diagram: ContextSensitiveHelp

E.5.1.6.14 Sequence diagram: StandardHelp()

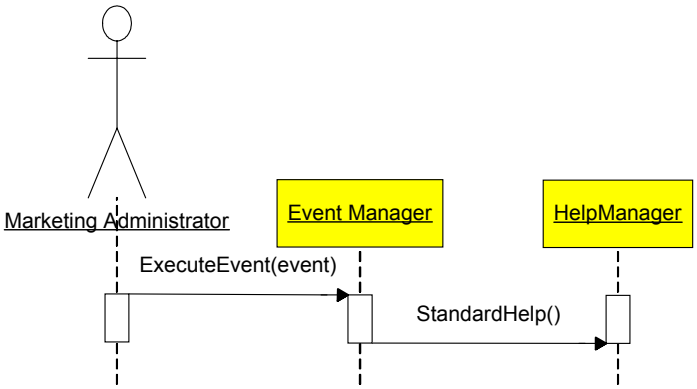

Figure 117 Sequence Diagram: StandardHelp

E.5.1.6.15 Sequence diagram: QueryBooking(BookingCode:int)

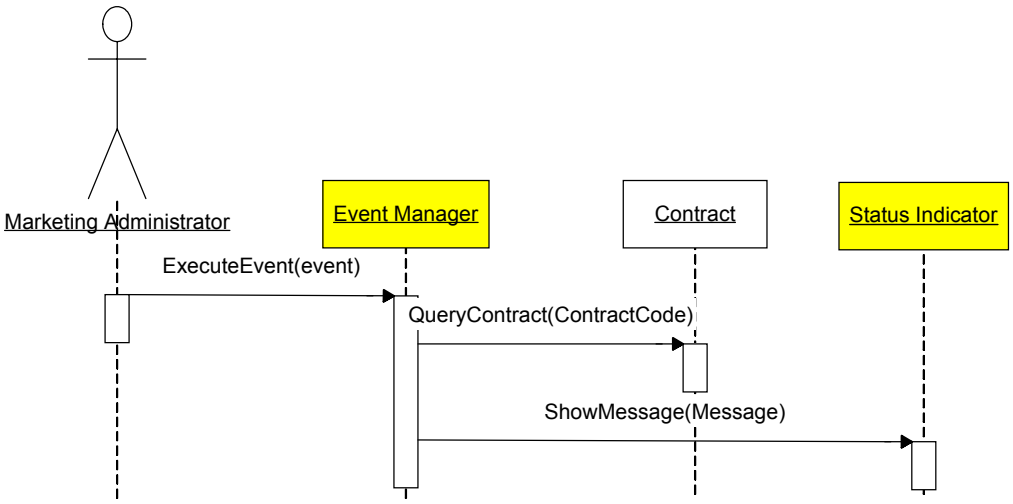

Figure 118 Sequence Diagram: QueryBooking

E.5.1.6.16 Sequence diagram: QueryCustomer(CustomerCode:int)

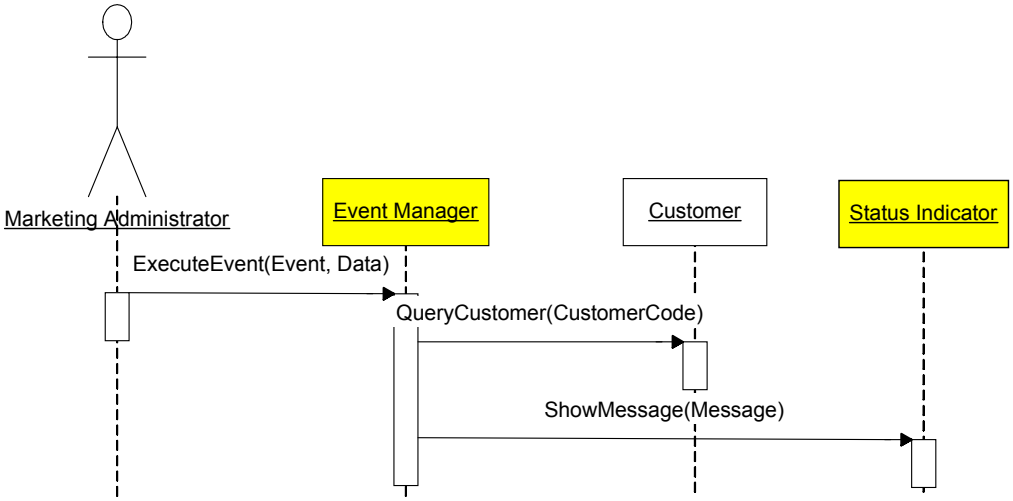

Figure 119 Sequence Diagram: QueryCustomer

E.5.1.6.17 Sequence diagram: QueryAgency(AgencyCode:int)

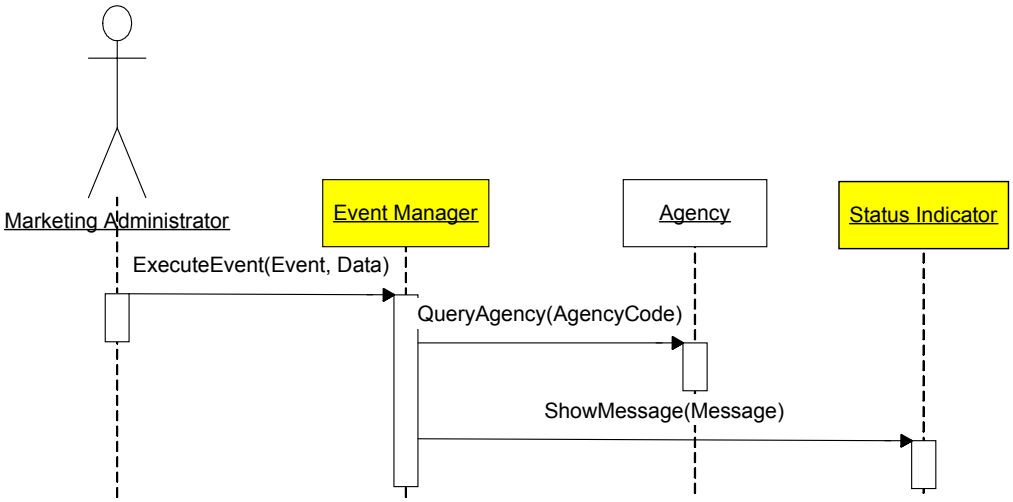

Figure 120 Sequence Diagram: QueryAgency

E.5.1.6.18 Sequence diagram: QueryContracts(QueryData:int)

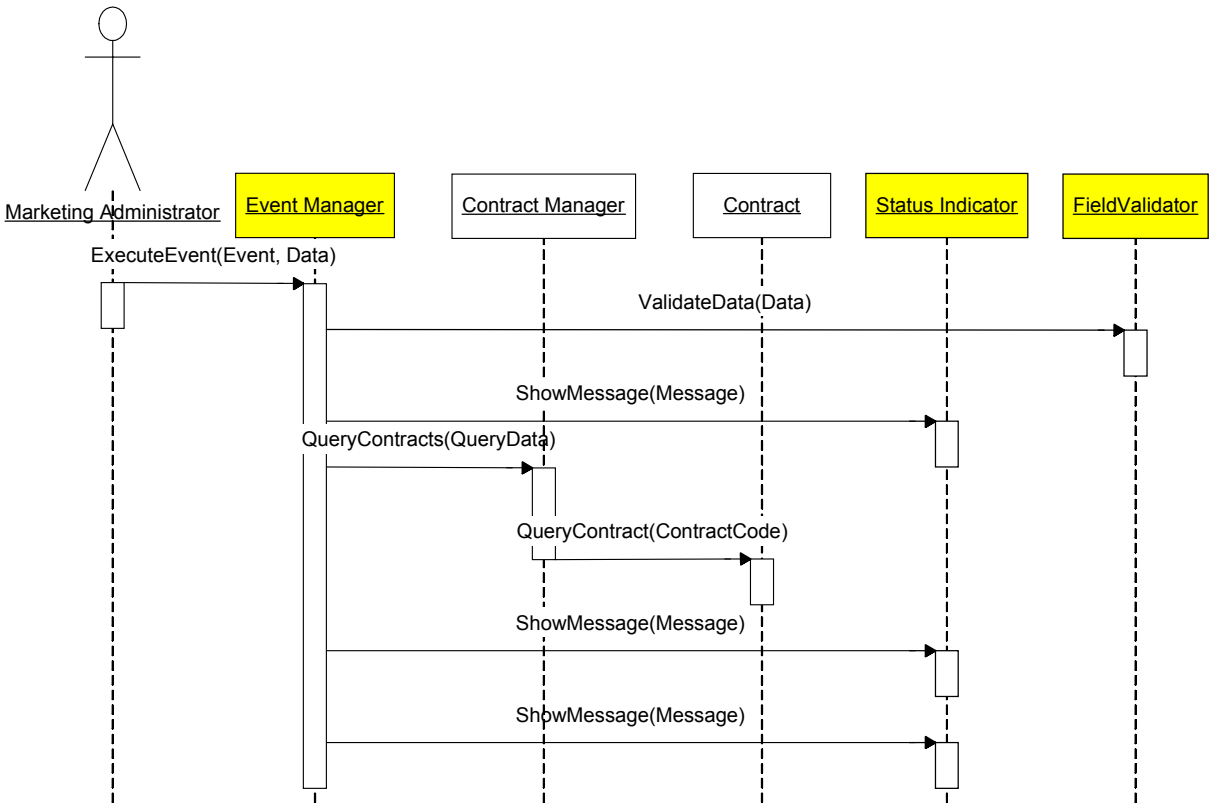

Figure 121 Sequence Diagram: QueryContracts

E.5.1.6.19 Sequence diagram: Preview(report:Object)

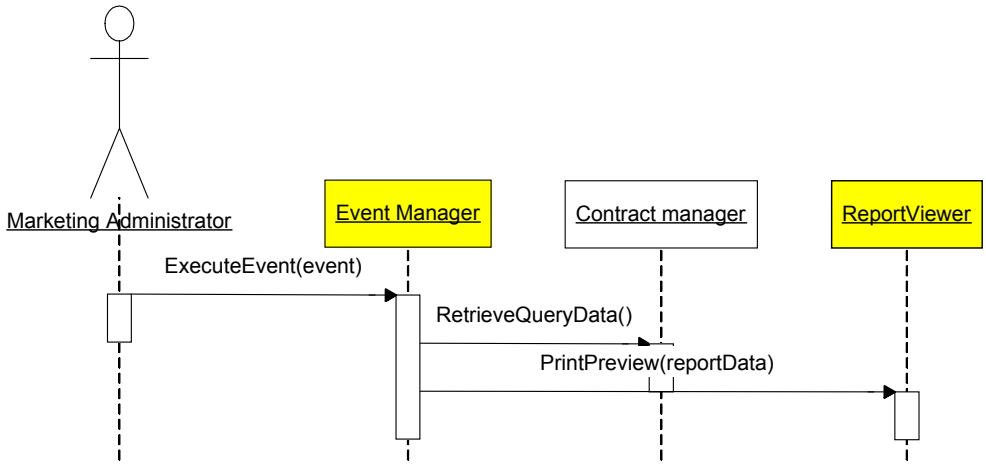

Figure 122 Sequence Diagram: Preview

E.5.1.6.20 Sequence diagram: Undo()

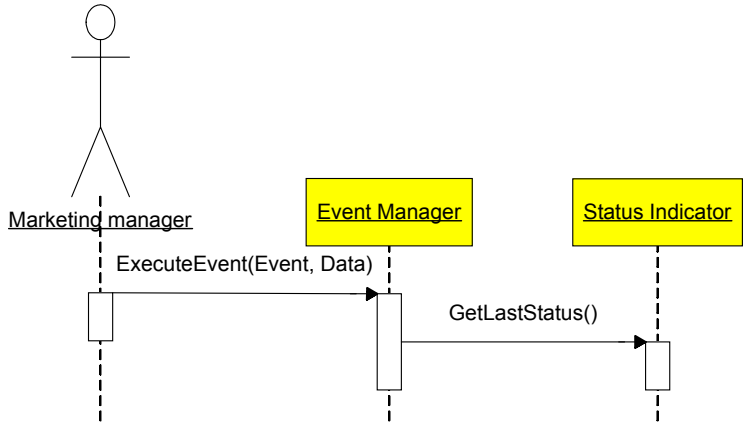

Figure 123 Sequence Diagram: Undo

E.5.1.6.21 Sequence diagram: Cancel()

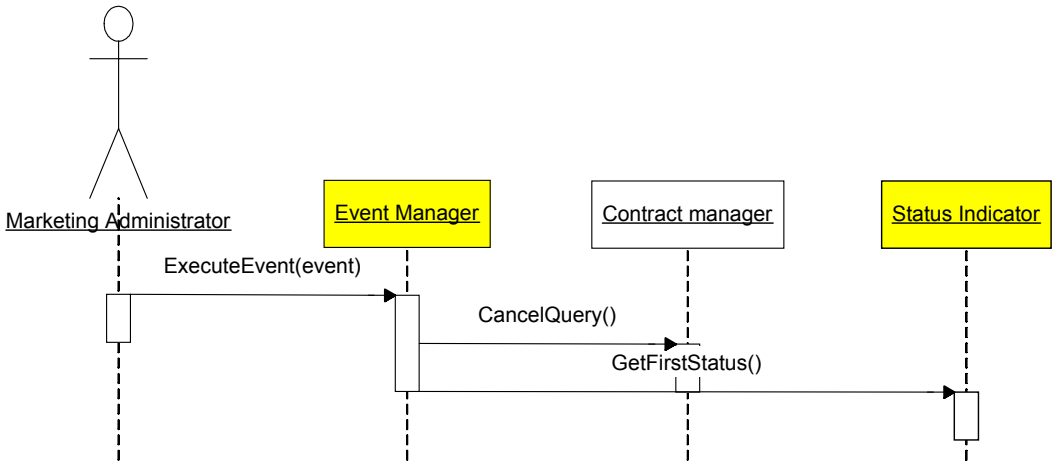

Figure 124 Sequence Diagram: Cancel

E.5.1.6.22 Sequence diagram: GetRemoteReport(QueryData:Object)

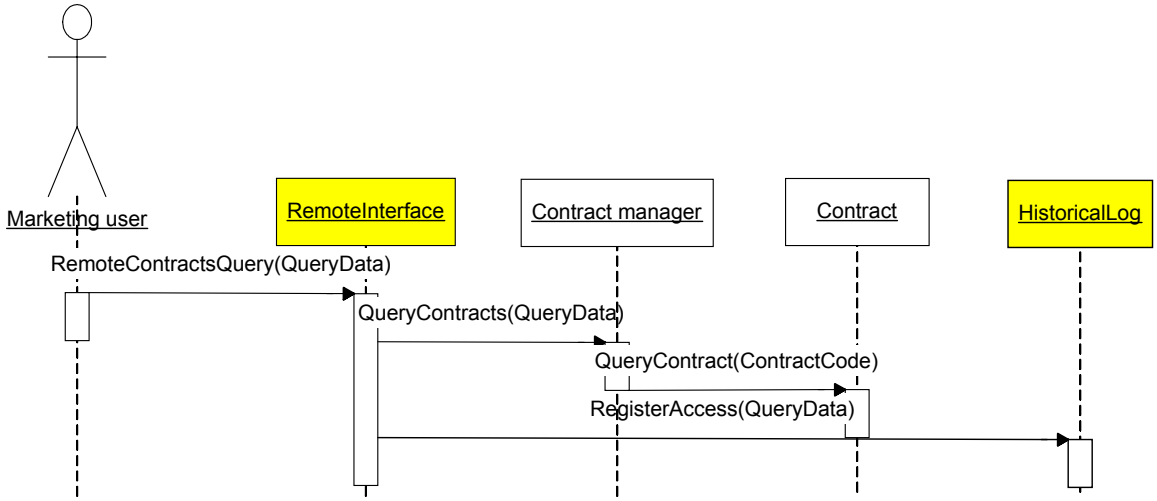

Figure 125 Sequence Diagram: GetRemoteReport

E.5.1.6.23 Sequence diagram: NewApplicationInstance()

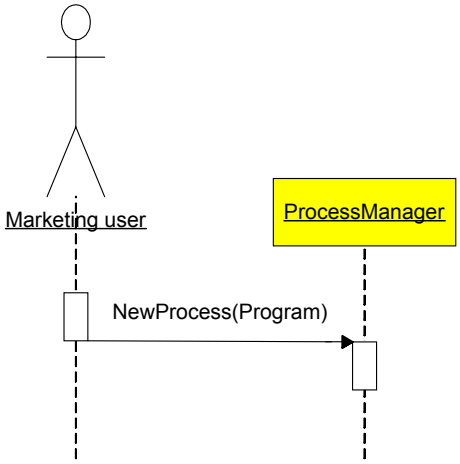

Figure 126 Sequence Diagram: NewApplicationInstance

E.5.1.6.24 Sequence diagram: ExportReport(Report:Object, Route:String)

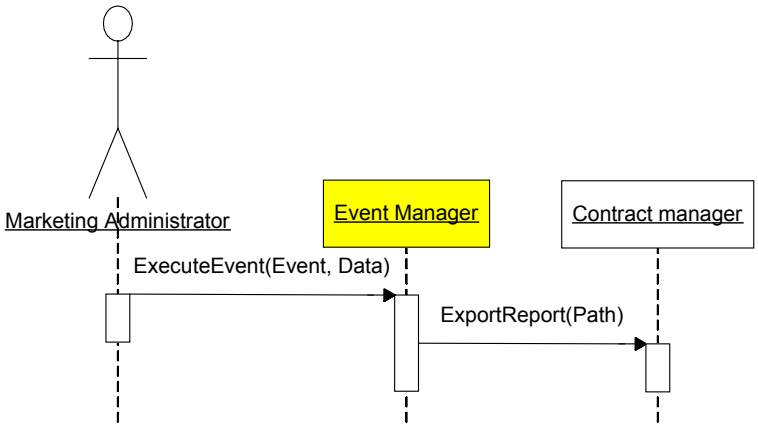

Figure 127 Sequence Diagram: ExportReport

E.5.1.6.25 Sequence Diagram: Query affixation job tickets

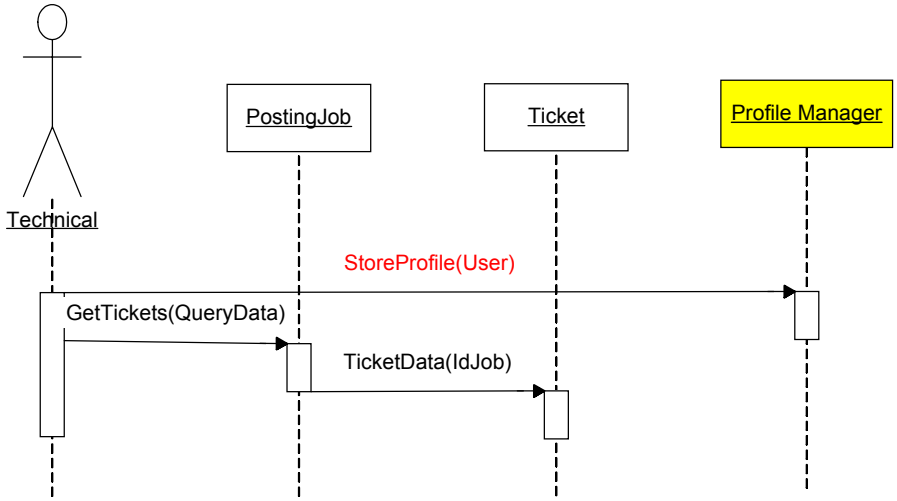

Figure 128 Sequence Diagram: Query affixation job tickets

E.5.1.6.26 Sequence Diagram: Change ticket status

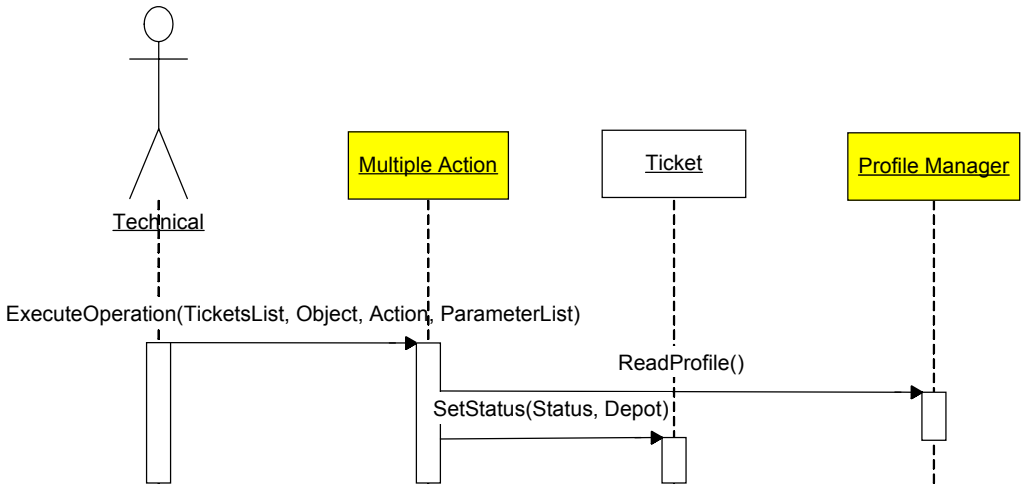

Figure 129 Sequence Diagram: Change ticket status

E.5.1.6.27 Sequence Diagram: Query affixation job

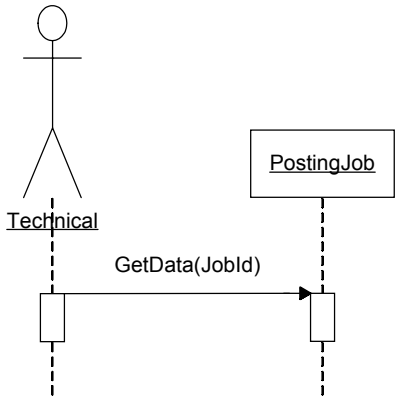

Figure 130 Sequence Diagram: Query affixation job

E.5.1.6.28 Sequence Diagram: Get affixation job report

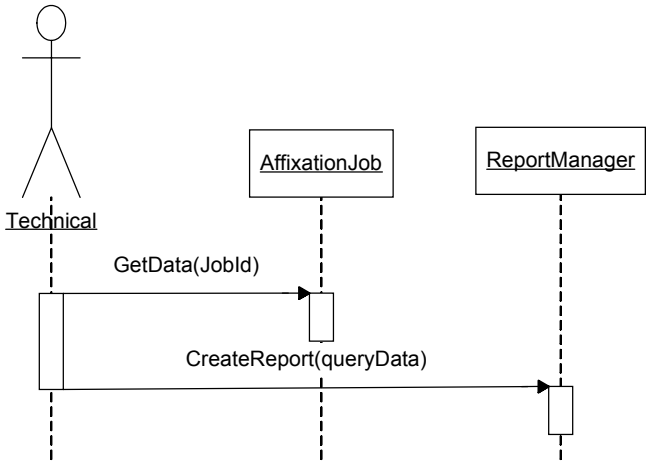

Figure 131 Sequence Diagram: Get affixation job report

E.5.1.7 Class Diagrams

The following class diagram has been created on the basis of the interaction diagrams and conceptual model.

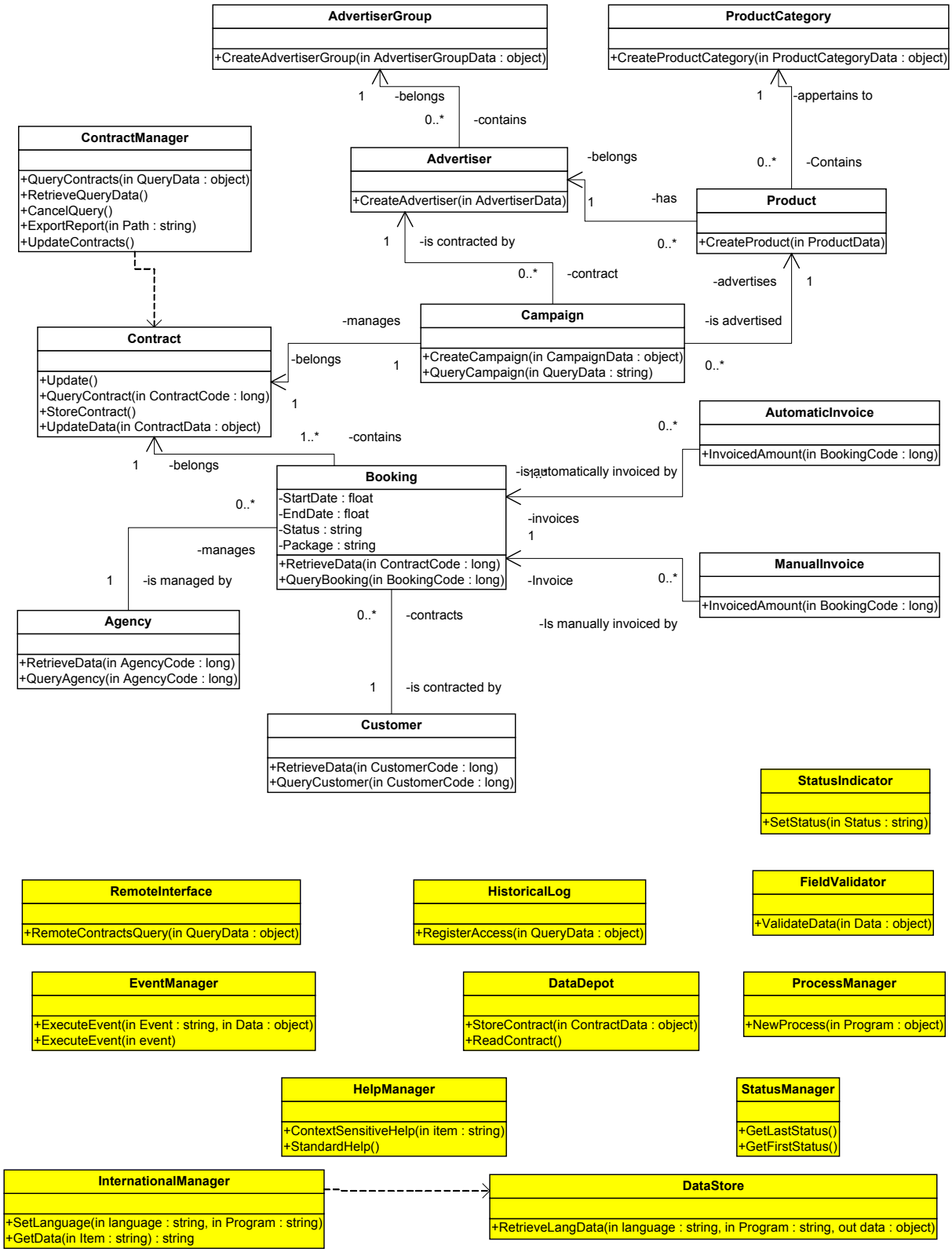

Figure 132 Class Diagram Part 1

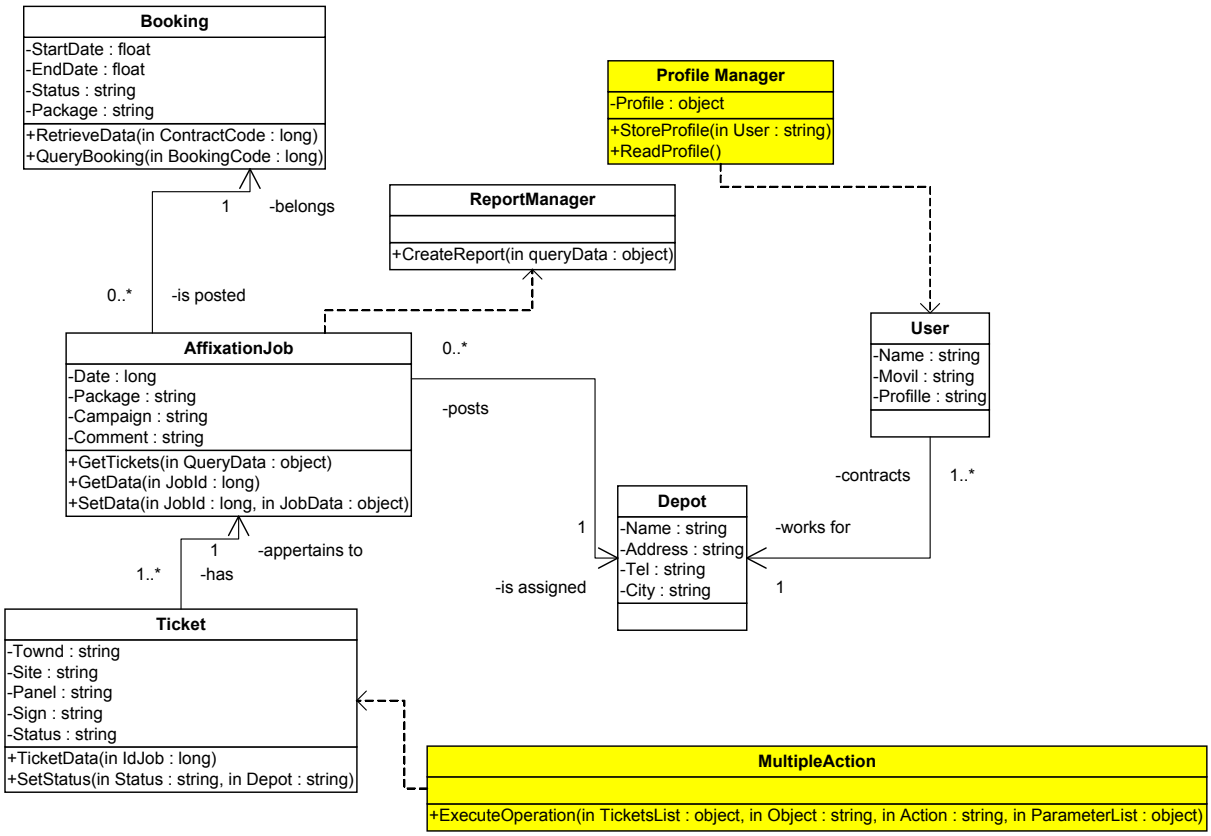

Figure 133 Class Diagram Part 2

E.6 Usability patterns

E.6.1.1 Purpose

The purpose of this annex is to introduce the concept of usability patterns, which have been used in the analysis and design phase of the document.

E.6.1.2 Usability patterns

In this document, usability pattern refers to the technique or mechanism that can be applied to design the architecture of a software system to satisfy the usability requirements identified in the requirements elicitation phase.

Name	Description
Different languages	Capability of the software to interact with users in different languages.
Different access methods	Capability of the software to be accessed using different types of physical devices.
Alert	An alert is a message from the system to the user that a change of state has occurred that the user ought to know about.
Status indication	Users are informed of the current status of the system.
Shortcut	Allows an experienced user to activate a functionality that may be hidden under the interface with one quick manoeuvre.

Form Field Validation	When a user fills in multiple fields, the system can validate these fields altogether when the data are sent (form validation) or individually each time a field is filled in (field validation).
Undo	The system's ability to undo an action and return to the previous state.
Contextual help	Context-sensitive help monitors what the user is currently doing and supplies information relevant to the completion of the task in question.
Wizard	Presents users with a structured sequence of steps to carry out a task, providing the user with guidance for each step. The user can go back and change any of the earlier steps in the process.
Standard help	The system should provide users with information and help on any of the tasks to be performed using the system.
Tour	The system should provide users with specific information showing them how to actually perform a given task.
Model Workflow	The system provides users with a series of tools or actions for executing a specific task on a dataset, before sending these data to the next person in the workflow chain who will execute a different task.
History logging	The system records the actions executed by users, allowing them to view the record of what they have done.
Preview	The system allows users to view the results of an action before executing this action.
User Profile	The system creates and saves user profiles to be able to retrieve specific system attributes associated with a user every time this user uses the system.
Cancel	Users can cancel what they are doing if they realise that they have done something wrong before an error state is reached.
Multi-Tasking	The situation where the system (and the user) can manage several tasks at the same time, switching from one task to another.
Commands aggregation	The system allows users to create and save a sequence of commands or actions that they can execute on different objects.
Action for multiple objects	The system allows users to select multiple objects on which to execute one and the same action at the same time.
Reuse information	This pattern allows users to move or copy data automatically or manually from one part of the system, thus allowing reuse

Table ¡Error! No hay texto con el estilo especificado en el documento.**-1 Usability patterns description**

E.6.1.3 Pattern application

As mentioned in the introduction, the objective of this document is to introduce usability patterns during the analysis and design phase of the document, to evaluate the modifications to be made during this phase.

The following table summarises the applications in which usability requirements have been introduced in development cycle 1.

	Different languages	Different access methods	Alert	Status indication	Shortcuts	Form Field validation	Undo	Context sensitive help	Wizard	Standard help	Tour	Workflow model	History logging	Preview	User Profile	Cancel	Multi-Tasking	Commands aggregation	Action for multiple objects	Reuse information
Affixation control															x				x	
Bookings classification by marketing category	x	x		x	x	x	x	x		x			x	x		x	x			x

Table ¡Error! No hay texto con el estilo especificado en el documento.**-2 Applications and associated usability patterns**

Table 2-3 summarises the use cases in which usability requirements have been introduced for the bookings classification by marketing category application.

	Different languages	Different Access Methods	Alert	Status indication	Shortcuts	Form Field validation	Undo	Context sensitive help	Wizard	Standard help	Tour	Workflow model	History logging	Preview	User Profile	Cancel	Multi-Tasking	Commands aggregation	Action for multiple objects	Reuse information
Bookings classification by mktg cat.	**x**	**x**		**x**	**x**	**x**	**x**	**x**		**x**			**x**	**x**		**x**	**x**			**x**
Update contracts	x			x																
Manage advertising group	x			x	x															
- Alternative 1				x	x															
- Alternative 2				x	x															
- Alternative 3				x	x															
Manage advertiser	x			x	x															
- Alternative 1				x	x															
- Alternative 2				x	x															
- Alternative 3				x	x															
Manage product category	x			x	x															
- Alternative 1				x	x															
- Alternative 2				x	x															
- Alternative 3				x	x															
Manage product	x			x	x															
- Alternative 1				x	x															
- Alternative 2				x	x															
- Alternative 3				x	x															
Manage campaign	x			x	x															
- Alternative 1				x	x															
- Alternative 2				x	x															
- Alternative 3				x	x															
Query campaign	x			x	x															
Manage contract	x			x	x															
- Alternative 1				x	x															x
- Alternative 2	x				x			x												
- Alternative 3	x				x					x										
Query contract	x			x	x															
Query booking	x			x	x															
Query customer	x			x	x															
Query agency	x			x	x															
Get contracts report	x			x	x	x								x						
- Alternative 1							x													
- Alternative 2																x				
- Alternative 3				x									x							
- Alternative 4																	x			
Export report		x																		
Query affixation job tickets															x					
Change ticket status															x				x	

Table ¡Error! No hay texto con el estilo especificado en el documento.**-3 Bookings classification by marketing category and associated usability patterns**

E.6.1.4 Usability patterns representation

The usability patterns are represented in the application as classes or class methods. Table 8-15 shows the respective IJESP system class or class method for each of the patterns used.

	Different languages	Different Access Methods	Status indication	Shortcuts	Form Field validation	Undo	Context sensitive help	Standard help		Preview	User Profile		Multi-Tasking	Action for multiple objects	Reuse information
Class – Internationalmanager Class – Datastore	x														
Class - RemoteInterface		x													
Class - StatusIndicator			x												
Class – Event Manager				x											
Class – FieldValidator					x										
Class – StatusManager						x									
Class - HelpManager							x								
Class – HelpManager								x							
Class - HistoryLog									x						
Class - ReportViewer										x					
Class - ProfileManager											x				
Method – Cancel.Object												x			
Class – ProcessManager													x		
Class - MultipleAction														x	
Class - DataDepot															x

Table ¡Error! No hay texto con el estilo especificado en el documento.**-4 Representation of usability patterns**