Architectural Design Document

The TOTAL COVER Insurance Company Case Study

W.F. Schellekens
O.M. Schinagl
T.A. van Roermund

Version 1.0

25th October 2005

Contents

Abstract

This document contains the Architectural Design for the TOTAL COVER Customer Relationship Management System (TCCRMS).

The architectural design of TCCMRS was constructed from the Software Requirements as stated in the [URD].

The document structure is based on the IEEE 1471 standard [2] and the Kruchten 4+1 standard [1].

Document Status Sheet

Document Status Sheet		
Document Title	Architectural Design Document	
Document Identification	ADD/1.0/	
Authors	W.F. Schellekens / O.M. Schinagl / T.A. van Roermund	
Document Status	draft / **internally accepted** / conditionally approved / approved	
Document History		
Version	**Date**	**Reason for change**
1.0	27-01-2003	Document creation

Document Change Record

Document Change Record		
DCR Number		
Date		
Originator		
Approved by		
Document Title	Architectural Design Document	
Document Identification	ADD/1.0/	
Page	**Paragraph**	**Reason for change**
-	-	-

Chapter 1

Introduction

1.1 Purpose

The purpose of the Architectural Design Document (ADD) is to describe the basic system design for the software to be made in this project. Furthermore the ADD defines a decomposition of the software system into components and the distribution of these components over the hardware. It also defines the external interfaces of the system.

1.2 Scope

The system being developed will be called TCCRMS. TCCRMS enables organizations to beter serve their customers through the introduction of reliable process and procedures of interacting with those customers. This CRMS in specific will improve the company's strategic position.

1.3 List of definitions

ADD	Architectural Design Document
CRMS	Customer Relationship Management System
UR	User Requirements
URD	User Requirements Document
SR	Software Requirements
TCCRMS	TOTAL COVER CRMS

1.4 References

[1] Philippe Kruchten, Brian Selic, and Wojtek Kozaczynski. Describing software architecture with uml. In *ICSE '01: Proceedings of the 23rd International Conference on Software Engineering*, pages 715–716, Washington, DC, USA, 2001. IEEE Computer Society.

[2] Rikard Land. *Applying the IEEE 1471-2000 Recommended Practice to a Software Integration Project*. CSREA Press, Las Vegas, Nevada, June 2003.

1.5 Overview

This chapter contains the general information about the document and the project.

Chapter 2 describes the overall design. It contains a system overview and describes the different stakeholders. A view template is described, which is used in chapter 3, 4 and 5.

Chapter 3 describes the logical view. It contains a description of the class diagram.

Chapter 4 describes the development view. It contains a description of the layer diagram.

Chapter 5 describes the deployment view. It contains a description of the deployment diagram.

Chapter 6 contains a use case diagram and models each use case as a separate sequence diagram.

Chapter 7 shows a traceability matrix. The matrix is used to connect each requirement to the operations and attributes, which will implement the requirement.

Appendix A shows some larger versions of diagrams used in the chapters.

Chapter 2

Overall design

2.1 System overview

TCCRMS enables the organization to better serve their customers through the introduction of reliable process and procedures of interacting with those customers.
The system will support or automate the tasks in the insurance proposal and/or an insurance policy making process. It provides detailed information about the client, claims and payments, to employees. Employees will also use this system to track and manage there own careers and those of other employees.
The system will be build from scratch, because this isn't an update of an earlier system.

2.2 Stakeholders

The system has a couple of stakeholder groups.

The first group consists of the users of the system. This group contains the sales representatives, clients (via website), administrative personnel and the business department. The main purpose of this document for this group is to show that every requirement is adequately provided by the system. The sequence diagrams are the views for this particular group.

The second group are the people who will maintain the system. The main purpose of this document for this group is to show how the system is build and deployed, so it can be understood and maintained. The logical view shows how the system is divided into classes and describes the functionality of the the system. The deployment view shows where each component or layer will be installed.

The third group are the developers who will implement the system. The main purpose of this document for this group is to show what has to be implemented. The logical view shows the functionality that should be implemented.

The last group consist of the managers who will divide the work, assign development teams and must calculate project metrics. The main purpose of this document for this group is to show how the system can be divided into smaller subsystems. The development view shows how the system can be divided.

2.3 View template

This is a description of the template of used to describe all of the views.

2.3.1 Primary presentation

The primary view shows the elements and the relationships among them that populate the view. It contains information which we want to convey about the system, in the vocabulary of the view. It includes the primary elements and relations of the view.

2.3.2 Element catalog

The element catalog details the elements and relations depicted in the primary presentation. Any elements or relations which where relevant to the view that were omitted from the primary presentation, are introduced and explained. The behavior and interfaces of an element are also described here.

2.3.3 Context diagram

The context diagram shows how the system depicted in the view relates to its environment in the vocabulary of the view.

2.3.4 Variability guide

The variability guide shows how to exercise any variation points that are a part of the architectural shown in this view.

2.3.5 Architectural background

The architectural background explains why the design reflected in the view came to be.

Rationale

The rationale part will explain why the decisions reflected in the view were made and why alternatives were rejected.

Assumptions

The assumptions part will describe any assumptions reflected in the design.

2.4 Chosen views

See *stakeholders* section for explanation of chosen views.

2.5 Mapping between views

There is a clear mapping between the logical view and the development view. The classes described in the logical view are divided into layers in the development view.

The sequence diagrams are derived from the operations of the classes in the logical view.

Chapter 3

Logical view

3.1 Primary presentation

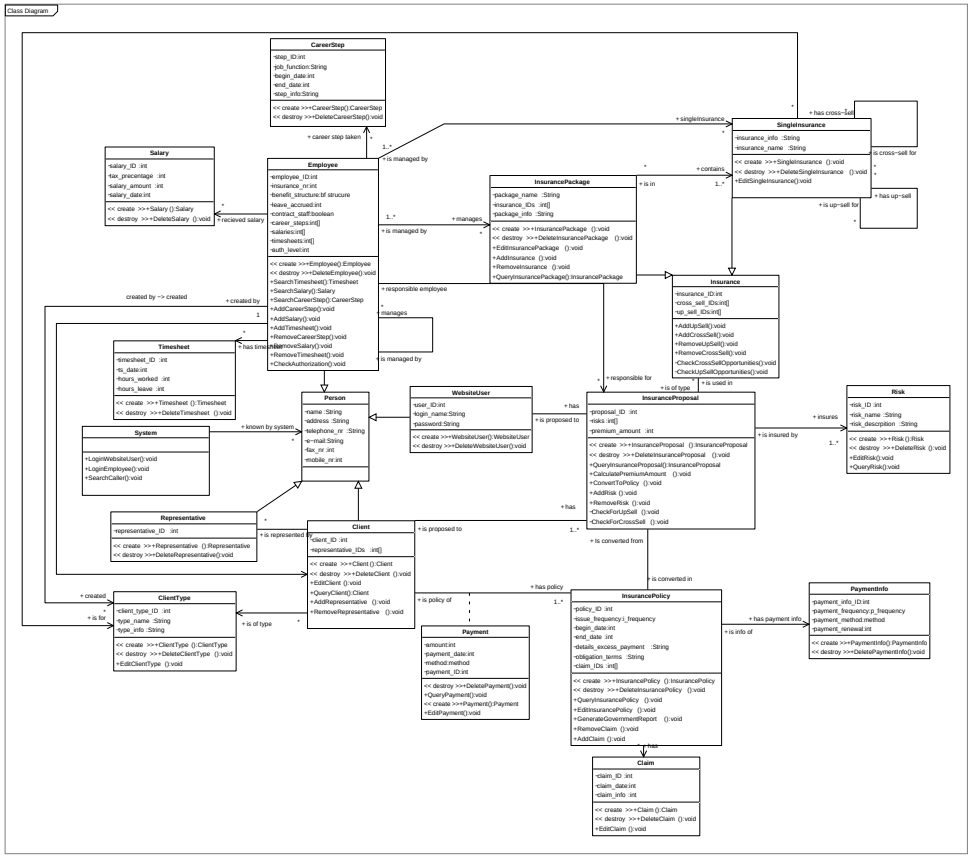


Figure 3.1: Class Diagram

A larger image can be found in appendix A, Class Diagram Large.

3.2 Element catalog

This section describes all of the classes in the class diagram as shown above. Each class description contains a general purpose of the class, the attributes of the class, the operations of the class and the relations with other classes. At the end of the class descriptions, three data types are described.

Person

This class is a general class which specifies the common attributes for a person.

The class attributes are:

ID	Attribute name	Description
AT0101	name	Represents the name of the person
AT0102	address	Represents the address of the person
AT0103	telephone_nr	Represents the telephone number of the person
AT0104	e-mail	Represents the e-mail address of the person
AT0105	fax_nr	Represents the fax number of the person
AT0106	mobile_nr	Represents the mobile number of the person

This class has the following relations:

Related with	Multiplicity	Description
Employee	-	Generalization
Client	-	Generalization
WebsiteUser	-	Generalization
Representative	-	Generalization
System	*	Known by system

Employee

This class represents an employee of TOTAL COVER.

The class attributes are:

ID	Attribute name	Description
AT0201	employee_ID	Represents the unique ID number of the employee
AT0202	insurance_nr	Represents the insurance number of the employee
AT0203	benefit_structure	Represents the benefit structure the employee has
AT0204	leave_accrued	Represents the amount of leave the employee has accrued
AT0205	contract_staff	Represents if the employee is a contract staff
AT0206	career_steps	Represents the list of career steps of the employee
AT0207	salaries	Represents the list of salary records of the employee
AT0208	timesheets	Represents the list of timesheets of the employee
AT0209	auth_level	Represents the authorization level of the employee

The class operations are:

ID	Operation name	Description
OP0201	Employee	Creates a new employee
OP0202	DeleteEmployee	Deletes the employee
OP0203	SearchTimesheet	Searches a specific timesheet of the employee
OP0204	SearchSalary	Searches a specific salary record of the employee
OP0205	SearchCareerStep	Searches a specific career step of the employee
OP0206	AddCareerStep	Adds a career step of the employee
OP0207	AddSalary	Adds a salary record of the employee
OP0208	AddTimesheet	Adds a timesheet of the employee¡/p¿
OP0209	RemoveCareerStep	Removes a career step of the employee
OP0210	RemoveSalary	Removes a salary record of the employee
OP0211	RemoveTimesheet	Removes a timesheet of the employee
OP0212	CheckAuthorization	Checks if employee has right authorization level

This class has the following relations:

Related with	Multiplicity	Description
Employee	*	Is managed by
Employee	*	Manages
Timesheet	*	Has time sheets
Salary	*	Salaries recieved
CareerStep	*	Career steps taken
Client	*	Manages
ClientType	*	Manages
InsurancePackage	*	Manages
SingleInsurance	*	Manages
InsuranceProposal	*	Responsible for
Person	-	Inherits from

Client

This class represents a client of TOTAL COVER.

The class attributes are:

ID	Attribute name	Description
AT0301	client_ID	Represents the unique client number
AT0302	representative_IDs	Represents the list of representatives of the client and can contain at most 2.

The class operations are:

ID	Operation name	Description
OP0301	Client	Creates a new client
OP0302	DeleteClient	Deletes the client

OP0303	EditClient	Edits a client
OP0304	QueryClient	Queries a client
OP0305	AddRepresentative	Adds a representative to the representative list
OP0306	RemoveRepresenative	Removes a representative from the list

This class has the following relations:

Related with	**Multiplicity**	**Description**
Representative	*	Is represented by
Employee	1	Managed by
ClientType	1	Is of type
InsurancePolicy	1..*	Has
InsuranceProposal	1..*	Has
Payment	*	Made
Person	-	Inherits from

Representative

This class represents a representative of a client.

The class attributes are:

ID	**Attribute name**	**Description**
AT0401	representative_ID	Represents the unique representative number

The class operations are:

ID	**Operation name**	**Description**
OP0401	Representative	Creates a new representative
OP0402	DeleteRepresentative	Deletes the representative

This class has the following relations:

Related with	**Multiplicity**	**Description**
Person	-	Inherits from
Client	1	Represents

WebsiteUser

This class represent a registered user of the website.

The class attributes are:

ID	**Attribute name**	**Description**
AT0501	user_ID	Represents the unique website user number
AT0502	login_name	Represents the login name of the website user

AT0503	password	Represents the password of the website user

The class operations are:

ID	Operation name	Description
OP0501	WebsiteUser	Creates a new website user
OP0502	DeleteWebsiteUser	Deletes the website user

This class has the following relations:

Related with	Multiplicity	Description
InsuranceProposal	0..1	Has
Person	-	Inherits from

System

This class represents the system which has common and high level operations.

The class operations are:

ID	Operation name	Description
OP0601	LoginWebsiteUser	Does login a website user
OP0602	LoginEmployee	Does login an employee
OP0603	SearchCaller	Searches the callers details if available

This class has the following relations:

Related with	Multiplicity	Description
Person	*	Knows about

Timesheet

This class represents a time sheet that an employee filled in for a particular work day.

The class attributes are:

ID	Attribute name	Description
AT0701	timesheet_ID	Represents the unique timesheet number
AT0702	ts_date	Represents the date for which the timesheet is about
AT0703	hours_worked	Represents the number of hours the employee worked on that day
AT0704	hours_leave	Represents the number of hours the employee took leave that day

The class operations are:

ID	Operation name	Description
OP0701	Timesheet	Creates a new timesheet
OP0702	DeleteTimesheet	Deletes the timesheet

This class has the following relations:

Related with	Multiplicity	Description
Employee	1	Is of

Salary

This class represents a salary record of an employee.

The class attributes are:

ID	Attribute name	Description
AT0801	salary_ID	Represents the unique salary record number
AT0802	tax_percentage	Represents the tax percentage calculated over the salary
AT0803	salary_amount	Represents the salary amount of the salary record
AT0804	salary_date	Represents the payment date of the salary

The class operations are:

ID	Operation name	Description
OP0801	Salary	Creates a new salary record
OP0802	DeleteSalary	Deletes the salary record

This class has the following relations:

Related with	Multiplicity	Description
Employee	1	Is of

CareerStep

This class represents a career step which an employee has taken.

The class attributes are:

ID	Attribute name	Description
AT0901	step_ID	Represents the unique career step number
AT0902	job_function	Represents the job function of the employee in the career step
AT0903	step_begin_date	Represents the begin date of the career step
AT0904	step_end_date	Represents the end date of the career step
AT0905	step_info	Represents additional info about the career step

The class operations are:

ID	Operation name	Description
OP0901	CareerStep	Creates a new career step
OP0902	DeleteCareerStep	Deletes the career step

This class has the following relations:

Related with	Multiplicity	Description
Employee	1	Is of

ClientType

This class represents a type of the client.

The class attributes are:

ID	Attribute name	Description
AT1001	client_type_ID	Represents the unique client type number
AT1002	type_name	Represents the name of the client type
AT1003	type_info	Represents info about the client type

The class operations are:

ID	Operation name	Description
OP1001	ClientType	Creates a new client type
OP1002	DeleteClientType	Deletes the client type
OP1003	EditClientType	Edits a client type

This class has the following relations:

Related with	Multiplicity	Description
Client	*	Describes type for
SingleInsurance	*	Describes type for

Insurance

This class represents the common operations and attributes of an insurance.

The class attributes are:

ID	Attribute name	Description
AT1101	insurance_ID	Represents the unique insurance number
AT1102	cross_sell_IDs	Represents the list of insurance IDs which are cross-sell opportunities for this insurance
AT1103	up_sell_IDs	Represents the list of insurance IDs which are up-sell opportunities for this insurance

The class operations are:

ID	Operation name	Description
OP1101	AddUpSell	Adds a up sell opportunities to the list
OP1102	AddCrossSell	Adds a cross sell opportunities to the list
OP1103	RemoveUpSell	Removes a up sell opportunities to the list
OP1104	RemoveCrossSell	Removes a cross sell opportunities to the list
OP1105	CheckCrossSellOpportunities	Checks for proposals which apply to a new cross opportunities
OP1106	CheckUpSellOpportunities	Checks for proposals which apply to a new up sell opportunities

This class has the following relations:

Related with	Multiplicity	Description
SingleInsurance	-	Inherits from
InsurancePackage	-	Inherits from
InsuranceProposal	*	Used in

SingleInsurance

This class represents a single insurance which can be used in an insurance proposal.

The class attributes are:

ID	Attribute name	Description
AT1201	insurance_name	Represents the name of the single insurance
AT1202	insurance_info	Represents the info about the single insurance

The class operations are:

ID	Operation name	Description
OP1201	SingleInsurance	Creates a new single insurance
OP1202	DeleteSingleInsurance	Delete the single insurance
OP1203	EditSingleInsurance	Edit the single insurance

This class has the following relations:

Related with	Multiplicity	Description
SingleInsurance	*	Is up-sell for
SingleInsurance	*	Has up-sell
SingleInsurance	*	Is cross-sell for
SingleInsurance	*	Has cross-sell
InsurancePackage	*	Is in

ClientType	1	Is of type
Employee	1..*	Managed by
Insurance	-	Inherits from

InsurancePackage

This class represents an insurance package.

The class attributes are:

ID	Attribute name	Description
AT1301	package_name	Represents the name of the insurance package
AT1302	package_info	Represents the info about the insurance package
AT1303	insurance_IDs	Represents the list of insurances contained in the package

The class operations are:

ID	Operation name	Description
OP1301	InsurancePackage	Creates a new insurance package
OP1302	DeleteInsurancePackage	Deletes the insurance package
OP1303	EditInsurancePackage	Does edit the insurance package
OP1304	QueryInsurancePackage	Does query the insurance package
OP1305	AddInsurance	Adds an insurance to the list
OP1306	RemoveInsurance	Removes an insurance from the list

This class has the following relations:

Related with	Multiplicity	Description
SingleInsurance	1..*	Contains
Employee	*	Is managed by
Insurance	-	Inherits from

InsuranceProposal

This class represents an insurance proposal created for a client or website user.

The class attributes are:

ID	Attribute name	Description
AT1401	proposal_ID	Represents the unique insurance proposal number
AT1402	risks	Represents the list of risks which the insurance proposal insures
AT1403	premium_amount	Represents the premium amount for the insurance proposal

The class operations are:

ID	Operation name	Description
OP1401	InsuranceProposal	Creates a new insurance proposal
OP1402	DeleteInsuranceProposal	Deletes the insurance proposal
OP1403	QueryInsuranceProposal	Queries the insurance proposal
OP1404	CalculateInsuranceProposal	Calculates a premium amount
OP1405	ConvertToPolicy	Converts the proposal into a policy
OP1406	AddRisk	Adds a risk to the list
OP1407	RemoveRisk	Removes a risk from the list
OP1408	CheckForUpSell	Checks for up sell opportunities when proposal is created
OP1409	CheckForCrossSell	Checks for cross sell opportunities when proposal is created

This class has the following relations:

Related with	Multiplicity	Description
Employee	1	Responsible employee
WebsiteUser	1	Is proposed to
Client	1	Is proposed to
Risk	1..*	Insures
Insurance	1	Is of type
InsurancePolicy	1	Is converted in

InsurancePolicy

This class represents an insurance policy of a client.

The class attributes are:

ID	Attribute name	Description
AT1501	policy_ID	Represents the unique policy number
AT1502	issue_frequency	Represents the frequency with which the policy is issued
AT1503	begin_date	Represents the begin date of the policy
AT1504	end_date	Represents the end date of the policy
AT1505	details_excess_payment	Represents the details of excess payment
AT1506	obligation_terms	Represents the obligation terms of the policy
AT1507	claim_IDs	Represents the list of claims of the policy

The class operations are:

ID	Operation name	Description
OP1501	InsurancePolicy	Creates a new insurance policy
OP1502	DeleteInsurancePolicy	Deletes the insurance policy
OP1503	QueryInsurancePolicy	Queries the insurance policy

OP1504	EditInsurancePolicy	Edits the insurance policy
OP1505	GenerateGovernmentReport	Generates a government report
OP1506	RemoveClaim	Removes a claim from the list
OP1507	AddClaim	Adds a claim to the list

This class has the following relations:

Related with	**Multiplicity**	**Description**
InsuranceProposal	1	Is converted from
PaymentInfo	1	Has
Claim	*	Has
Client	1	Is policy of
Payment	*	Has

Risk

This class represents a risk which is insured by a proposal.

The class attributes are:

ID	**Attribute name**	**Description**
AT1601	risk_ID	Represents the unique risk number
AT1602	risk_name	Represents the name of the risk
AT1603	risk_info	Represents a description of the risk

The class operations are:

ID	**Operation name**	**Description**
OP1601	Risk	Creates a new risk
OP1602	DeleteRisk	Deletes the risk
OP1603	EditRisk	Does edit the risk
OP1604	QueryRisk	Queries the risk

This class has the following relations:

Related with	**Multiplicity**	**Description**
InsuranceProposal	1	Insured by

Claim

This class represents a claim for an insurance policy.

The class attributes are:

ID	**Attribute name**	**Description**
AT1701	claim_ID	Represents the unique claim number

AT1702	claim_date	Represents the date of the claim
AT1703	claim_info	Represents a description of the claim

The class operations are:

ID	Operation name	Description
OP1701	Claim	Creates a new claim
OP1702	DeleteClaim	Deletes the claim
OP1703	EditClaim	Does edit the claim

This class has the following relations:

Related with	Multiplicity	Description
InsurancePolicy	1	Issued for

PaymentInfo

This class represent payment info for an insurance policy.

The class attributes are:

ID	Attribute name	Description
AT1801	payment_info_ID	Represents the unique payment info number
AT1802	payment_frequency	Represents the frequency with which payment is done
AT1803	payment_method	Represents the method of payment of the client for the policy
AT1804	payment_renewal	Represents the method of renewal which is automatic or manually

The class operations are:

ID	Operation name	Description
OP1801	PaymentInfo	Creates a new payment info
OP1802	DeletePaymentInfo	Deletes the payment info

This class has the following relations:

Related with	Multiplicity	Description
InsurancePolicy	1	Is about

Payment

This association class indicates a payment paid by a client for a particular insurance policy.

The class attributes are:

ID	Attribute name	Description
AT1901	payment_ID	Represents the unique payment number
AT1902	payment_date	Represents the date when the payment was made
AT1903	method	Represents the method of payment
AT1904	amount	Represents amount of the payment

The class operations are:

ID	Operation name	Description
OP1901	Payment	Creates a new payment
OP1902	DeletePayment	Deletes the payment
OP1903	QueryPayment	Queries the payment
OP1904	EditPayment	Does edit a payment

This class has the following relations:

Related with	Multiplicity	Description
Client	1	Fs from
InsurancePolicy	1	Is about

The following sections will describe the data types used in the class diagram which aren't standard. All data types introduced here are enumerations.

p_frequency

Data type which enumerates the different frequencies by which a payment can be made.

ID	Enumeration
DT0001	daily, weekly, fortnightly, monthly, quarterly and yearly

i_frequency

Data type which enumerates the different frequencies by which a policy can be issued.

ID	Enumeration
DT0002	monthly, quarterly, half-yearly and yearly

method

Data type which enumerates the different methods of payments.

ID	Enumeration
DT0003	cash, check and credit card

3.3 Context diagram

N/A

3.4 Variability guide

All of the operations are points of variability because none of the parameters are decided yet. This will be done in the "detailed design phase" of the project. The return type of most of the operations are also declared in this phase of the project.

The "Edit..." operations in a class are generalizations of all the "Set..." operations of attributes.

3.5 Architectural background

3.5.1 Rationale

All of the mayor entities in the requirements are modeled as a class, this is how the classes were generated.

Because of the 4 kinds of person types in the requirements (*employee*, *client*, *website user* and

representative), we decided to make one *person* class with all general information. Via generalization we attached the 4 types to it.

Because of the 2 kinds of insurances in the requirements (*single insurances* and *insurance packages*), we decided to make one *insurance* class with all general information. Via generalization we attached both types to it.

An *employee* has time sheets, salaries and career steps attached to him, because of administration and career tracking purposes.

A *client* is attached to his insurance policies and proposals. If the client has any representatives, they will be denoted as a *representative*. A person can both be in the system as a representative and as a client. The information will not be shared.

An insurance proposal insures one or more risks, but is only of one type of insurance (either a single insurance or an insurance package). Once a proposal is converted to a policy, the proposal will still be used to describe the type of insurance and the insured risks.

An insurance proposal created via the website can't be converted to policy. The policy can only be attached to a client.

Because a payment is concerned with a client-policy pair, it has been modeled as a association class.

Client type indicates if a client is an organization or not and indicates for what type of client an insurance is meant for.

The higher level person operations, like identifying a caller, are in the *system* class.

3.5.2 Assumptions

- Every Insurance policy is converted from an Insurance proposal.
- Every Insurance proposal remains stored after conversion, because it contains the details about risks, premiumInfo and type of insurance.
- Premium amount is stored as /month.
- An insurance policy and insurance proposal is for one single insurance or for one insurance package.
- Website users must log-in (and register) before they can make insurance proposals.
- An insurance proposal remains stored in the system if the attached policy is deleted, because of reuse.
- Insurance proposals made for a website user can't be converted to a policy.

3.6 Other information

N/A

Chapter 4

Development view

4.1 Primary presentation

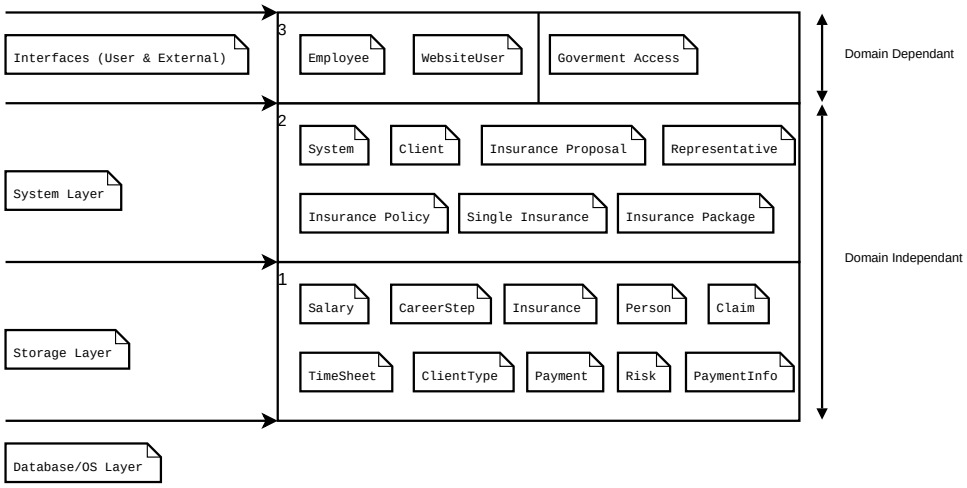


Figure 4.1: Layered Development View

4.2 Element catalog

This section describes the layers used to sub-divide the classes.

4.3 Context diagram

N/A

4.4 Variability guide

N/A

4.5 Architectural background

4.5.1 Rationale

OS/Database Layer

The OS/Database layer makes it possible for the classes to communicate with the underlaying system as well as to store data in a database.

Storage Layer

The storage layer contains the classes that control data that needs to be stored. E.g. the class *Person* contains interfaces to retrieve information such as address and telephone numbers from a person, a person-record. Via an interface to the lower database layer it is therefore possible to store and retrieve person-records.

System

The system layer does all the work. Calculations, generation of reports etc. The data required for all these operations is retrieved from the lower storage layer.

Interfaces (User & External)

The interfaces layer provides graphical user interfaces or network interfaces for people and other systems to use the Total Cover Insurance Customer Relationship Management System.

4.5.2 Assumptions

N/A

4.6 Other information

Chapter 5

Deployment view

5.1 Primary presentation

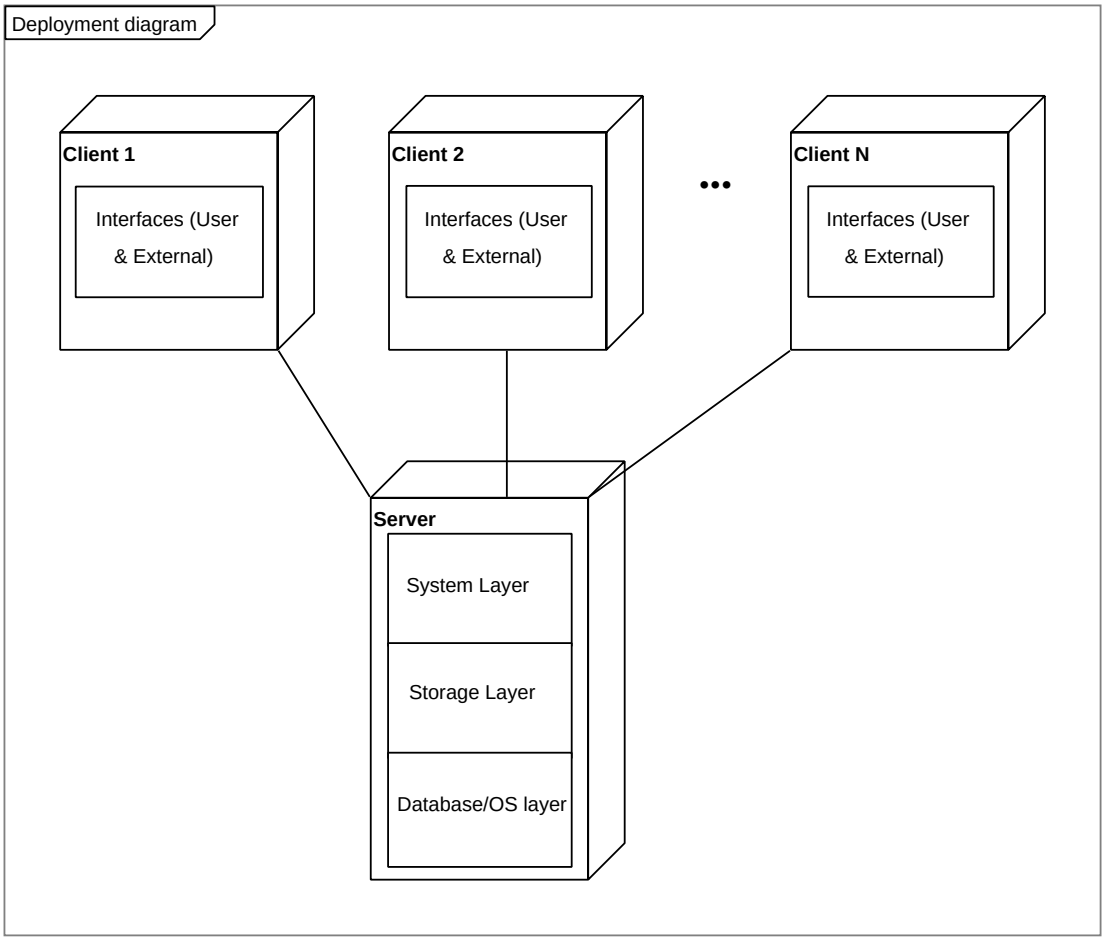


Figure 5.1: Deployment Diagram

5.2 Element catalog

This section will describe physical components of the deployment diagram.

Client

The client is the hardware (or PC) which is used by an employee for using the system, by a client to access the website, or by the government instance to access the website.

It only contains the highest layer of the development diagram, which is the systems interface to the users. This is the interface of the website which the client or government instance uses to get information from the system. The employees uses this interface to work with the system.

Server

The server is the hardware (or server) which communicates with the system, does all the calculations and stored all the data.

It contains the three lower layers of the development diagram. For a description of the three levels see the *Development view*.

5.3 Context diagram

N/A

5.4 Variability guide

N/A

5.5 Architectural background

5.5.1 Rationale

The system uses a Client-Server architecture style. The components are servers and clients, which are both used in our model. The exact style is a so called *thin client* style. The largest part of processing is at the server-side.

The reasons why we've chosen this style are:

- The style ensures data security, because all calculations are done server side and data is server side.
- It also makes the configuration management simple, because it only has to be done on the server.
- The style makes the system more robust, because the clients have no state.

5.5.2 Assumptions

- There will be only one main server for the system.

5.6 Other information

Chapter 6

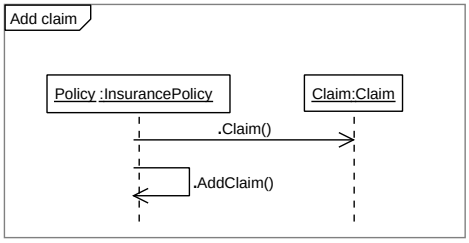
User Scenarios

This chapter will illustrate all the use cases derived from the requirements. Each requirement is modeled in the use case diagram as a single use case. The use case diagram is shown below.

In the following sections, a sequence diagram is modeled and described for each individual use case.

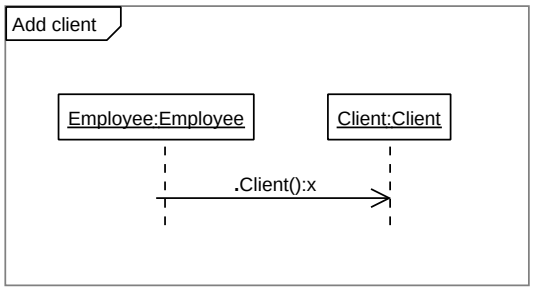
Add a new claim

An employee can add a claim to the claim list of a policy, first the claim has to be created.



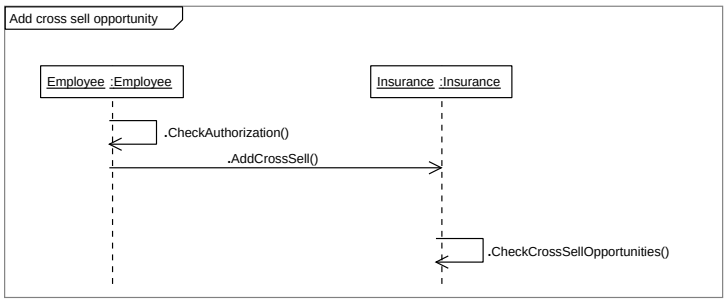
Add a new client

An employee can add a new client to the system.



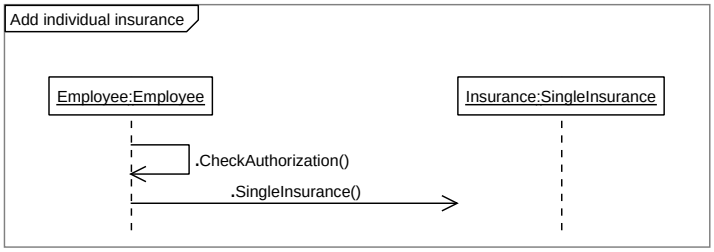
Add a new cross sell opportunity

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can add an up sell opportunity to the up sell opportunities list. After it is added, the system will check if their are any insurances which apply to the opportunity.



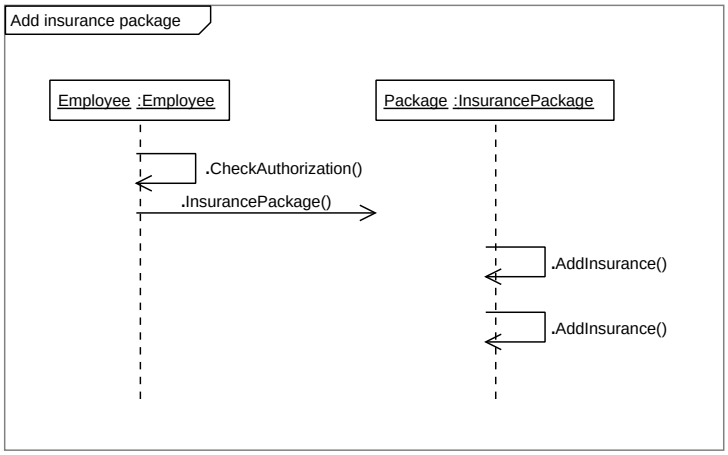
Add an new individual insurance

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can create a new insurance.



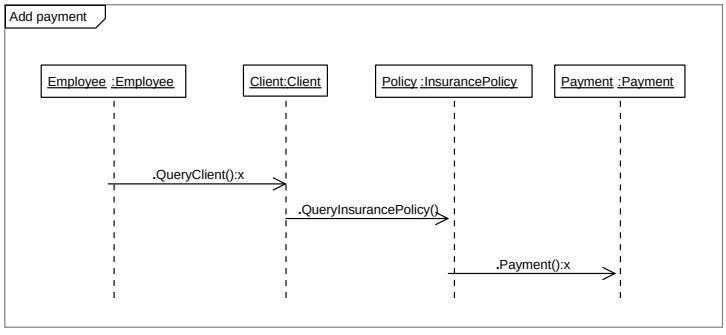
Add an new insurance package

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can create a new insurance package and add single insurances.



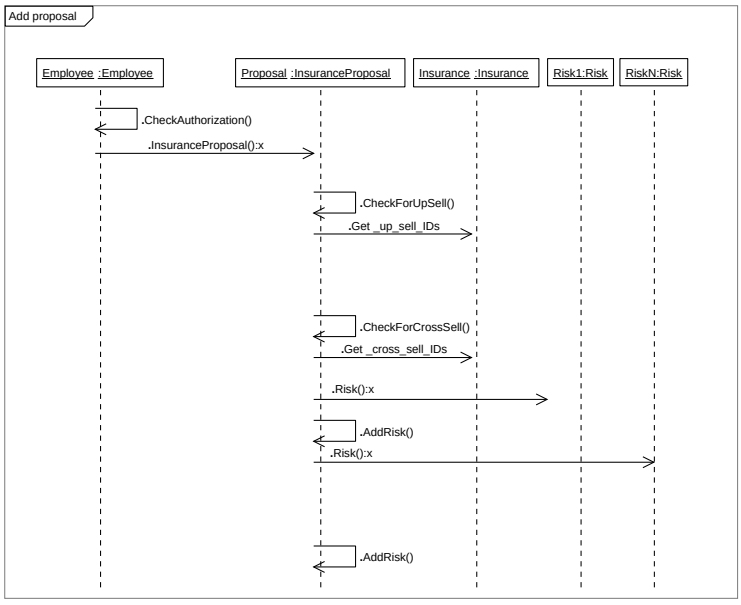
Add a new payment

An employee can add a new payment to the a client-policy pair.



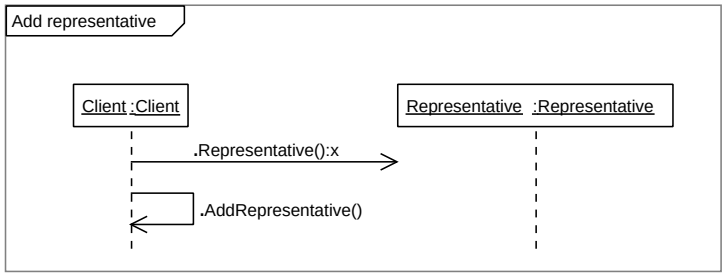
Add a new proposal

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can create a new insurance proposal. After the proposal has been created, the system will check if there are any cross or up sell opportunities that apply to the insurance used in the proposal. After the two checks, the risks which the proposal insures are created and added to the risk list of the proposal.



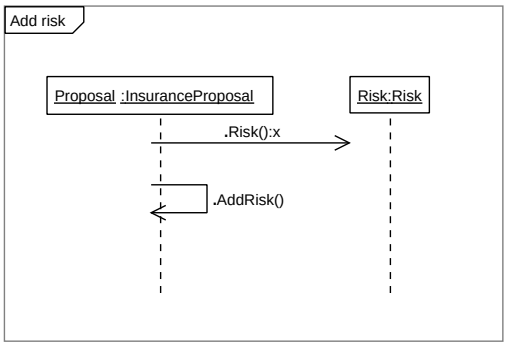
Add a new representative

An employee can add a representative to a clients list of representatives. Before the representative can be added to list, it has to be created.



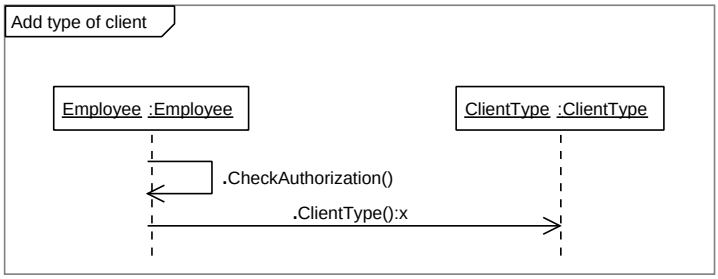
Add a new risk

Before a risk can be added to the risks list of a insurance proposal, it has to be created.



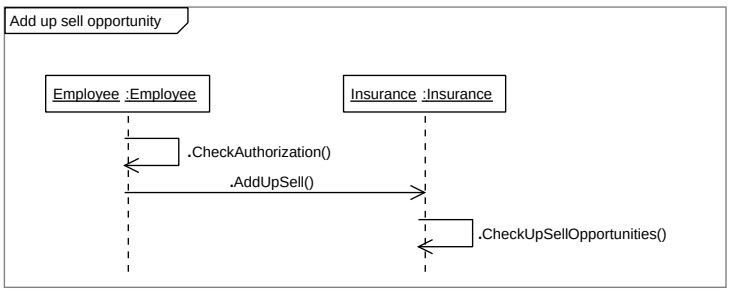
Add a new type of client

First the employee is checked to see if he/she has the right authorization level. If he has the appropriate level, he can create a new client type.



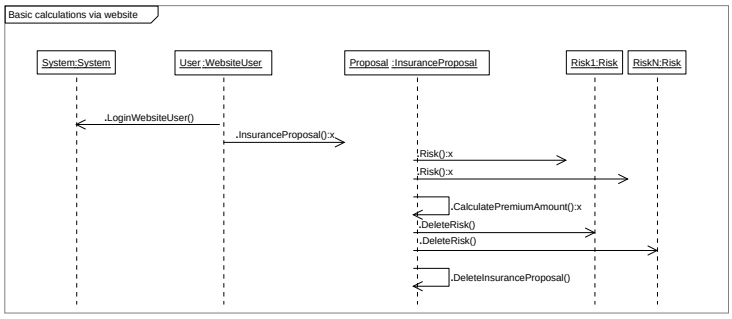
Add a new up sell opportunity

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can add an up sell opportunity to the up sell opportunities list. After it is added, the system will check if their are any insurances which apply to the opportunity.



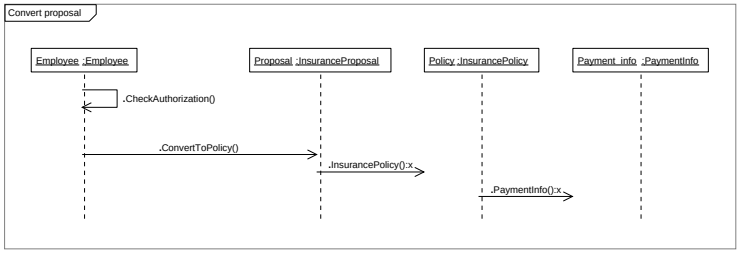
Do basic calculations via website

If a user is logged in at the website he can create an insurance proposal. If the proposal is created and the risks (1..N) are specified, it is used to calculate a premium amount. After the premium amount has been calculated, the proposal and the risks (1..N) will be deleted again.



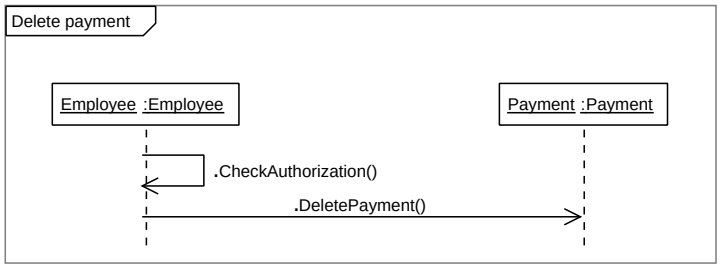
Convert a proposal into a policy

First the employee is checked to see if he/she has the right authorization level. If he has the appropriate level, he can convert the proposal into a policy. A new policy is created and a payment info will be created and will be attached to it.



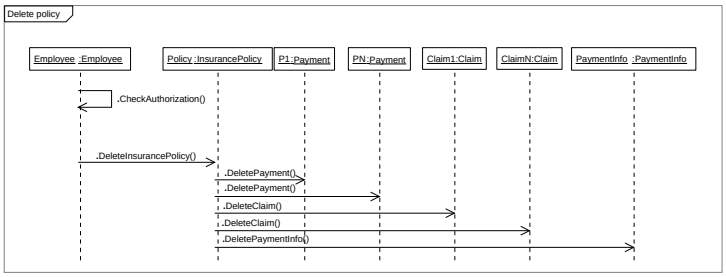
Delete a payment

First the employee is checked to see if he/she has the right authorization level. If he has the appropriate level, he can delete a payment.



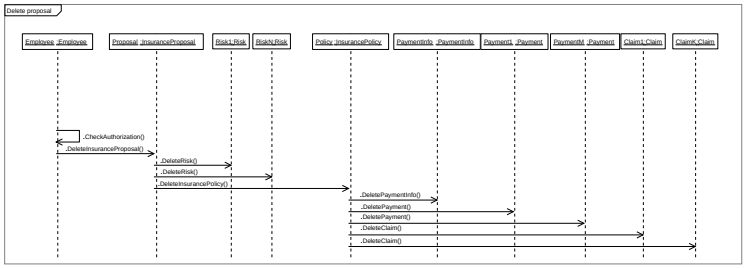
Delete a policy

First the employee is checked to see if he/she has the right authorization level. Then the policy, the payment info ,the payment (1..N) attached to it and the claims (1..M) attached to it will be deleted. The proposal attached to it will not be deleted, because it can be reused.



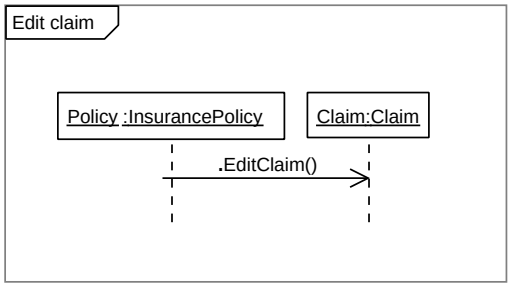
Delete a proposal

First the employee is checked to see if he/she has the right authorization level. Then the proposal and the risks (1..N) attached to it will be deleted. Because of the deleted proposal, the attached policy will also be deleted if there exists one. The Payment info, claims (1..K) and the payments (1..M) attached to the policy will be deleted last.



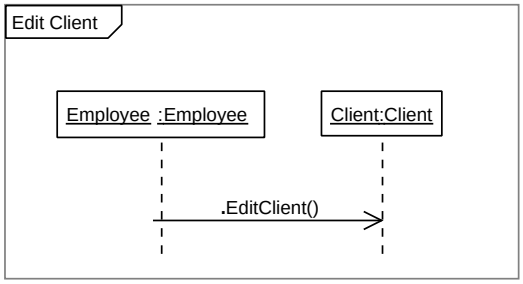
Edit a claim

An employee can edit an existing claim.



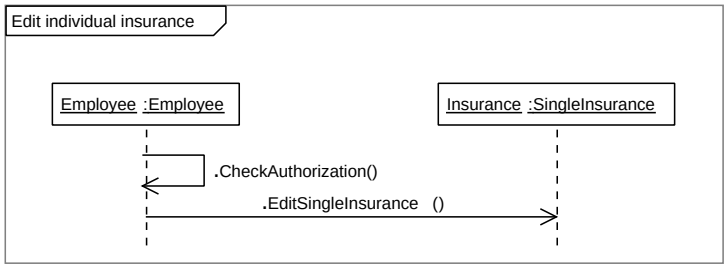
Edit a client

An employee can edit an existing client.



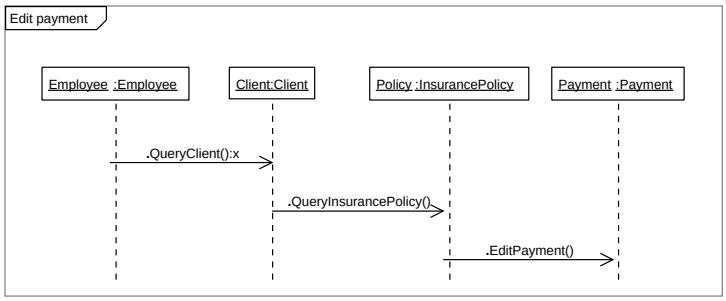
Edit an individual insurance

First the employee is checked to see if he/she has the right authorization level. If he has the appropriate level, he can edit a single insurance.



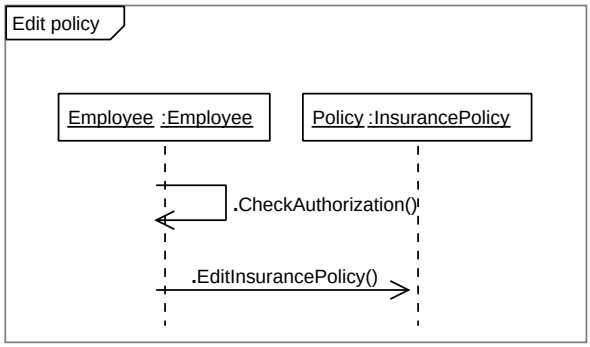
Edit a payment

An employee can edit an existing payment.



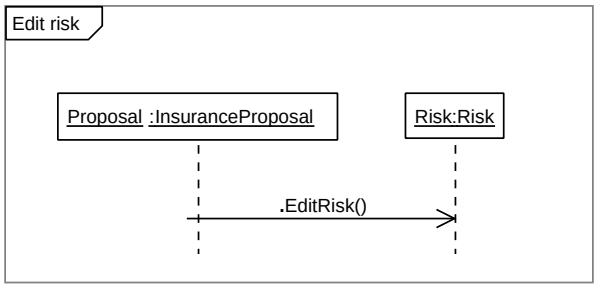
Edit a policy

An employee can edit an existing policy.



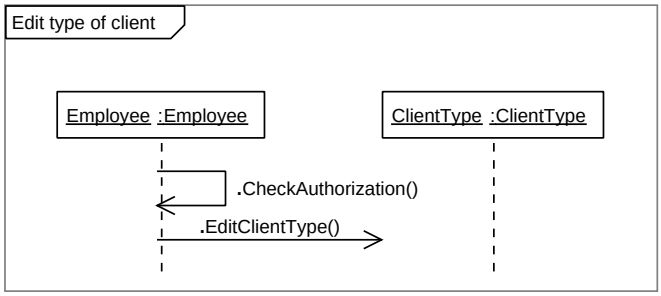
Edit risk

An employee can edit an existing risk.



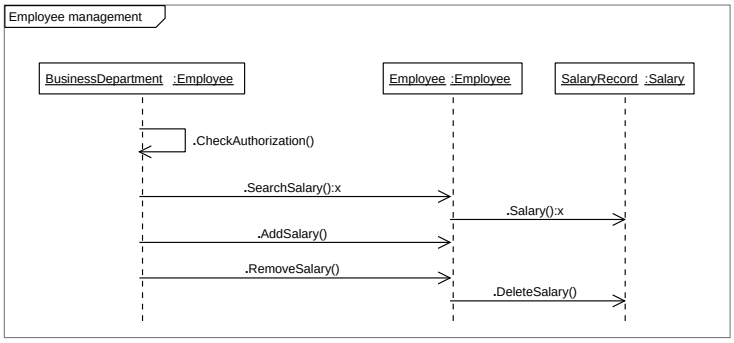
Edit a type of client

First the employee is checked to see if he/she has the right authorization level. If he has the appropriate level, he can edit a client type.



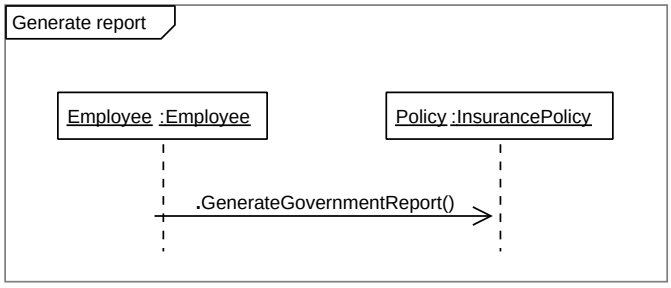
Employee management

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can search, add or remove a salary record from the list of salary records. If a salary record is added to the list, it first has to be created. When removing a salary record from the list, it will be deleted.



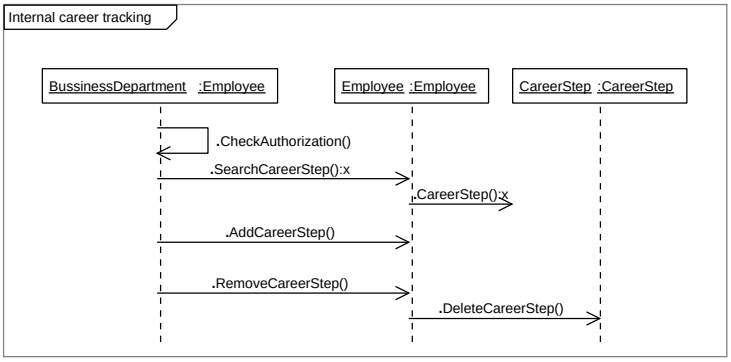
Generate government report

A report can be generated which contains all details of an insurance policy.



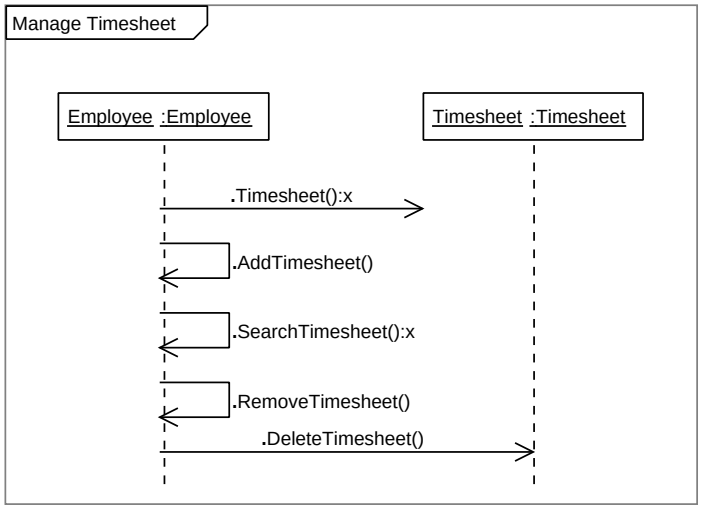
Do internal career tracking

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can search, add or remove a career step from the list of career steps. If a career step is added to the list, it first has to be created. When removing a career step from the list, it will be deleted.



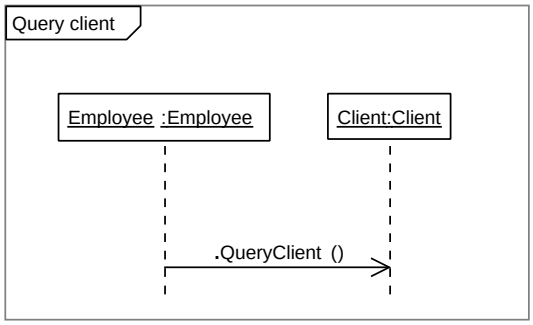
Manage a time sheet

When a employee wants to manage his timesheets, he can add, search and remove timesheets. Before he adds a timesheet to his timesheet list, he must first create it. If he removes a timesheet from his timesheet list, the timesheet will be deleted.



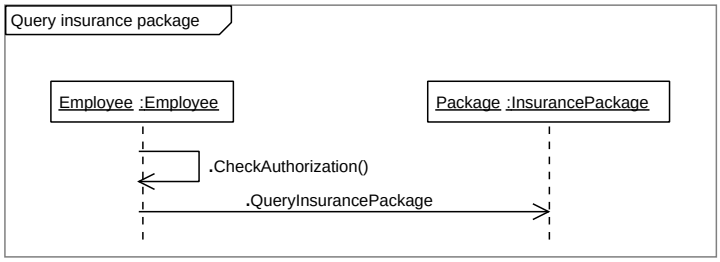
Query a client

An employee can query a client.



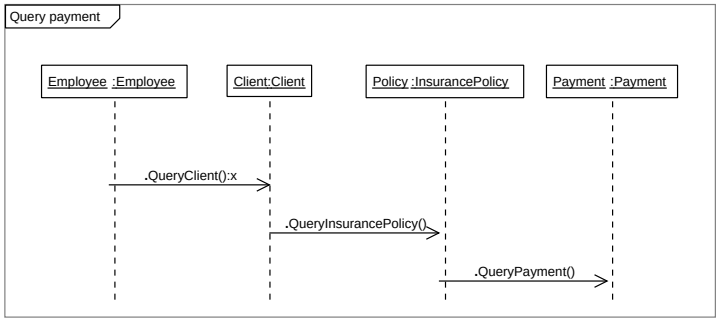
Query an insurance package

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he has the appropriate level, he can query the insurance package.



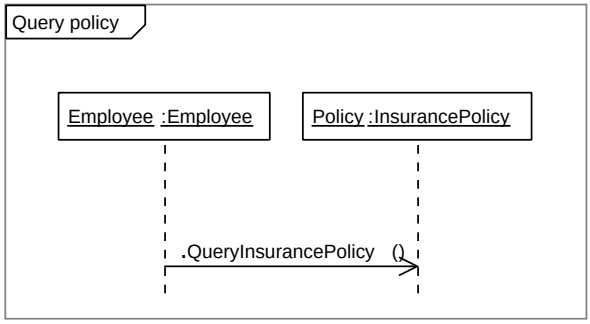
Query a payment

An employee can query a payment.



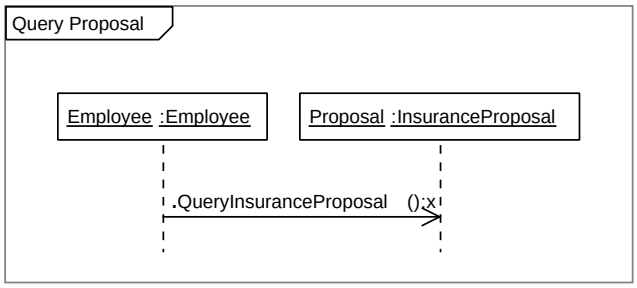
Query a policy

An employee can query a policy.



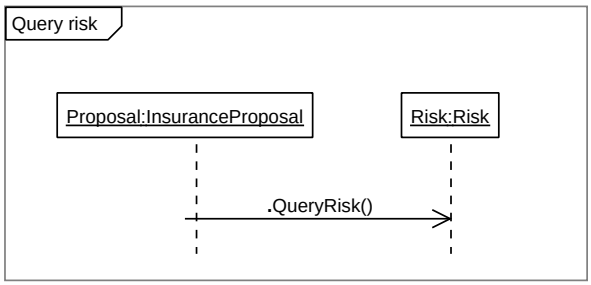
Query a proposal

An employee can query a proposal.



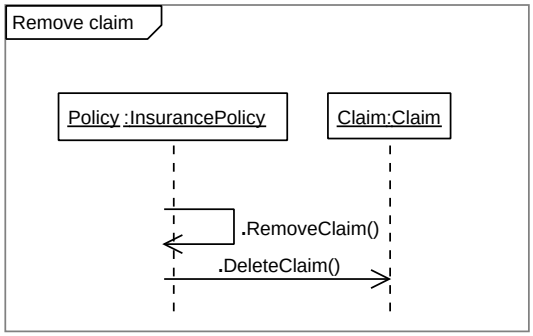
Query a risk

An employee can query a risk.



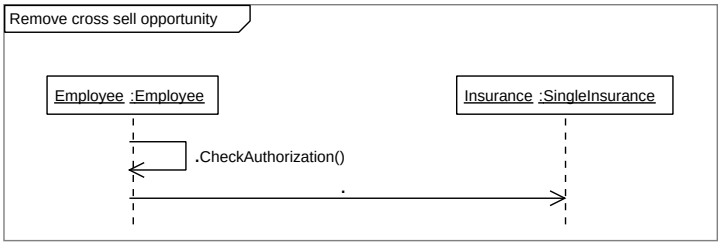
Remove a claim

An employee can remove a claim from a insurance policy list of claims. If the claim is removed from the list, it will be deleted.



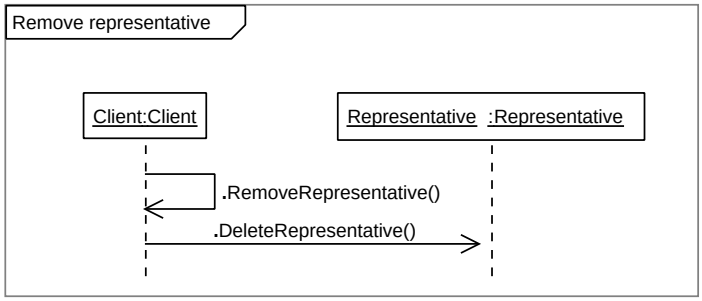
Remove a cross sell opportunity

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he/she has the appropriate level, he can remove an cross sell opportunity from the list.



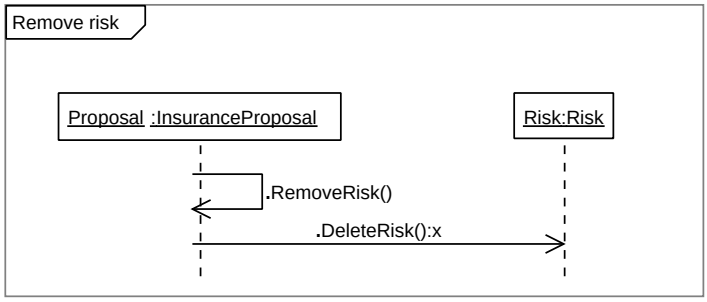
Remove a representative

An employee can remove a representative from a clients list of representatives. If the representative is removed from the list, it will be deleted.



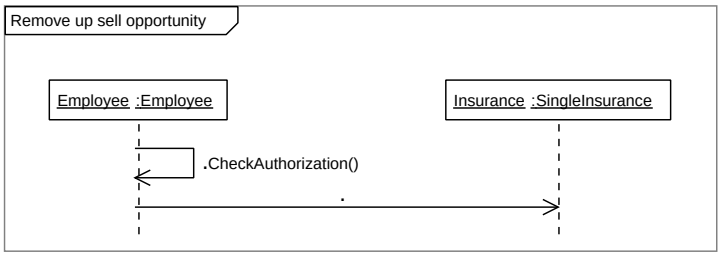
Remove a risk

An employee can remove a risk from a insurance proposal list of risk. If the risk is removed from the list, it will be deleted.



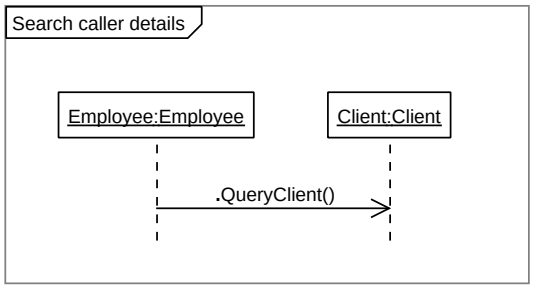
Remove an up sell opportunity

First the employee is checked to see if he/she has the right authorization level (of the Business department). If he/she has the appropriate level, he can remove an up sell opportunity from the list.



Search caller details

the system will show the caller details, if the caller is known to the system. A caller is known if he/she is stored as a client.



Chapter 7

Traceability matrix

This traceability matrix links the requirements to the attributes and properties that will implement them.

URD ID	Class diagram ID
RGE01	AT1001, AT1002, AT1003
RGE02	AT0302, OP0305, OP0306, AT0401, OP0401, OP0402
RGE03	OP1405
RGE04	OP1408, OP1409
RIN01	OP1505
RIN02	OP1505
RIN03	OP0603, AT0101-AT0106
RIN04	OP0603
RIN05	OP0601, AT0501, AT0502, AT0503, OP0501, OP0502
RIN06	OP0601, AT0501, AT0502, AT0503, OP0501, OP0502, OP1404
RIN07	-
RIN08	-
RIN09	OP1901-OP1904, AT1901-AT1904, AT1701-AT1703, OP1701-OP1703
RAD01	OP1201, OP1202, OP1203
RAD02	OP1001, OP1002, OP1003
RAD03	OP0301, OP0302, OP0303, OP0304, AT0301
RAD04	AT1301, AT1302, AT1303, OP1301-OP1306
RAD05	AT1501-AT1507, OP1501, OP1502, OP1503, OP1504, OP1506, OP1507
RAD06	AT1601, AT1602, AT1603, OP1601-OP1604, OP1406, OP1407
RAD07	AT1401, AT1402, AT1403, OP1401-OP1404, OP1406-OP1409

RAD08	AT1801-AT1804, OP1801, OP1802
RAD09	OP1405
RAD10	AT1101, AT1102, AT1103, OP1101-OP1106
RAD11	AT0201-AT0209, OP0201-OP0212, OP0602, AT0701-AT0704, OP0701, OP0702, AT0801-AT0804, OP0801, OP0802, AT0901-AT0905, OP0901, OP0902
RAD12	AT0201-AT0209, OP0201-OP0212
RAD13	AT0208, OP0203, OP0208, OP0211, AT0701-AT0704, OP0701, OP0702
RAD14	AT0206, OP0205, OP0206, OP0209, AT0901-AT0905, OP0901, OP0902
RAD15	AT0201-AT0209, OP0201-OP0212, OP0602, AT0701-AT0704, OP0701, OP0702, AT0801-AT0804, OP0801, OP0802, AT0901-AT0905, OP0901, OP0902
RIP01	AT1201, AT1202, AT1001-AT1003, OP1001-OP1003
RIP02	AT1201, AT1202, AT1001-AT1003, OP1001-OP1003
RIP03	AT1402, OP1406, OP1407
RIP04	AT1502, DT0002
RIP05	AT1401-AT1403, AT1501-AT1507
RPA01	OP1404, AT1403
RPA02	AT1803, AT1903, DT0003
RPA03	AT1802, DT0001
RPA04	AT1804
C001	AT0302

Appendix A

Diagrams

Class Diagram Large

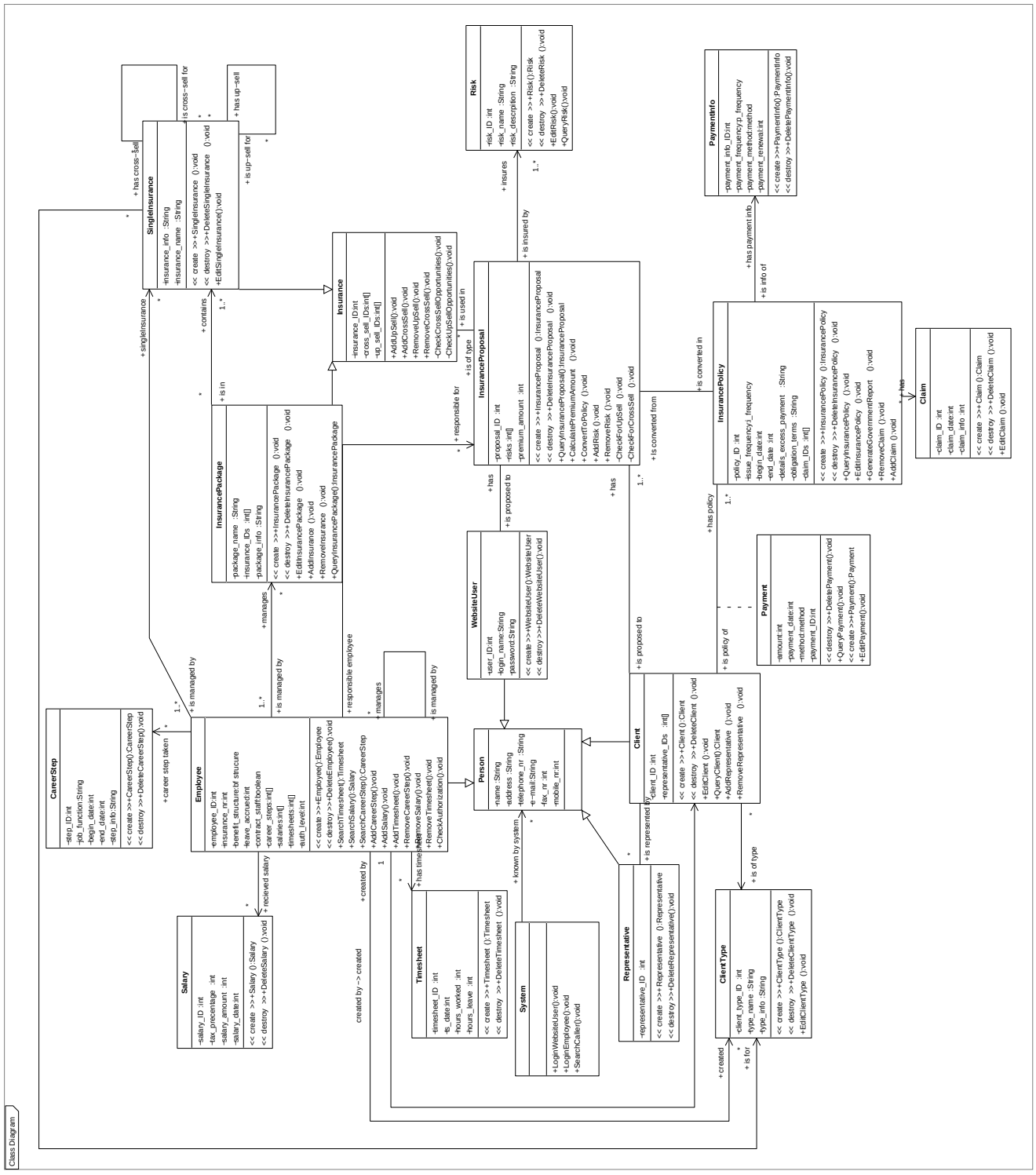


Figure 1: Class Diagram