Projet 2019

Risk

Auteurs

Nom	Adresse
Auteur_1	auteur_1@etu.univ-nantes.fr
Auteur_2	auteur_2@etu.univ-nantes.fr
Auteur_3	auteur_3@etu.univ-nantes.fr
Auteur_4	auteur_4@etu.univ-nantes.fr

Introduction

Description du projet

Organisation du document

Analyse du domaine

Introduction

Objectif

Ce chapitre constitue le premier livrable d'une série de quatre chapitres destinés à fournir une analyse et une conception par objets complètes répondant au cahier des charges qui nous a été fourni. Ce document présente l'ensemble de la démarche suivie ainsi que les résultats obtenus lors de la phase de l'analyse du domaine de notre système. Il se décompose en plusieurs parties.

Dans la première partie, nous présentons de manière détaillée l'ensemble des cas d'utilisation que nous avons dégagés lors de l'analyse. Nous utiliserons pour cela le canevas proposé par Cockburn que nous compléterons par des instantanés ainsi que par des post-conditions exprimées en OCL (Object Constraint Language) et quelques scenarii. Cette partie constitue une étape clé de la phase de l'analyse du domaine.

Dans la deuxième partie, nous présentons le diagramme de classes métiers (i.e. diagramme de classes au niveau analyse) que nous avons construit à partir de l'analyse réalisée. Ce diagramme fournit une vue statique et synthétique du domaine de notre projet. Cette partie constitue également une étape clé de la phase de spécification des besoins.

Dans la troisième er dernière nous fournissons le dictionnaire des données que nous avons construit suite à l'analyse du domaine. Il s'agit d'un listing de l'ensemble des termes relatifs au domaine étudié ainsi que leur définition précise.

Dans une quatrième et dernière partie,

Organisation du chapitre

Ce chapitre est organisé en n sections. Dans le première section, nous décrirons….

Cas d'utilisation

Mise en place d'un jeu

Use Case Template

Version 1.20

Instructions for removing the 'Hints, Guidelines and Examples' from this document

After you have completed the Use Case document, you may want to remove the hints and guidelines provided in the document.

To remove the hints: (This procedure applies to Microsoft Word XP and higher)

1. Click on any text formatted as Hint.

2. Then, click the right mouse button.
3. A pop-up menu will appear, choose ‘Select text with similar formatting’
4. All Hint text will now be selected in the document.
5. **Ensure that none of the text that you have entered is in the selection.**
6. Press the *Delete* key to remove the *Hints , Guidelines and examples.*.

Revision History

Date	Author	Description of change

Use Case: <Enter Use Case name here>

<Enter a short name for the Use Case using an active verb phrase.

e.g. Withdraw Cash, Register Customer, Rent Video, Calculate Sales Tax, etc.>

Id: UC- <Enter value of Id here>

<Enter a unique numeric identifier for the Use Case. e.g. UC-113>

Description

<Enter description here>

<Briefly describe this use case.

e.g. Customer brings selected video(s) to the sales counter for the purpose of renting them. Store clerk processes the rental payment, records the rented video(s) against customer’s account, and hands over the video(s) to the customer.>

Level: <Enter Use Case Goal Level here>

<Enter the goal level of this Use Case. Specify whether the Use Case level is - High Level Summary, Summary, User Goal, Sub-Function, Low Level>

Primary Actor

<List the Primary actor here>

<List the Actor who's goal is being satisfied by this Use Case and has the primary interest in the outcome of this Use Case.

e.g. Store Clerk>

Supporting Actors

<List supporting actors here>

<List the Actors who have a supporting role in helping the Primary Actor achieve his or her goal.

e.g. Customer, Store Manager>

Stakeholders and Interests

<List Stakeholders and their interests here>

<List the various entities who may not directly interact with the system but they may have an interest in the outcome of the use case. Identifying stakeholders and interests often helps in discovering hidden requirements which are not readily apparent or mentioned directly by the users during discussions.
e.g. In a Use Case 'Generate Salary Stub', the entity Internal Revenue Service(IRS) has no direct interaction, however, it sure has interest in ensuring that the proper tax deduction has been made from the employee's salary. This can be written as follows:

Internal Revenue Service – Has interest in ensuring that the tax deduction is made from each employee's salary as per the tax table.>

Pre-Conditions

<List Pre-Conditions here>

< List the system state/conditions which must be true before this Use Case can be executed.

e.g. Store Clerk must be logged in to system.>

Post Conditions

Success end condition

<List success end condition here>

<Enter the successful end condition of the Use Case where the Primary Actor's goal is satisfied.

e.g. Video is rented to the customer and customer is charged for the rental. Rental store's inventory is updated to reflect the rented video.>

Failure end condition:

<List failure end condition here>

< Enter the failure end condition of the Use Case if the Primary Actor's goal has not been achieved.

e.g. Customer is unable to rent the video. Rental Store's video inventory remains unchanged.>

Minimal Guarantee

<List minimal guarantee here>

< The guarantee or assurance that this Use Case provides to all Actors and Stakeholders to protect their interest regardless of whether the Use Case ends with success or failure.

e.g. For Withdraw Cash (ATM Use Case), minimal guarantee could be, Customer is logged out of the ATM system.
This minimum guarantee ensures that the system will ensure that no unauthorized withdrawals can be made from the ATM thus protecting the interest of the Bank Customer as well as the Bank's stakeholders. >

Trigger

<List Use Case trigger here>

<The event that starts this Use Case.

Example

For *Rent Video* Use Case - Customer brings the Video to the sales counter.

For *Withdraw Cash* Use Case - Customer inserts the bank card into the ATM machine.>

Main Success Scenario

1. <Enter steps here>
2. <Enter steps here>
3. <Enter steps here>

<Enter the Main flow of events. i.e. The steps narrating/illustrating the interaction between Actors and the System. Describe Actor's actions/stimuli and how the system responds to those stimuli. Describe the 'happy path/day' scenario, meaning the straight and simple path where everything goes 'right' and enables the primary actor to accomplish his or her goal. Main flow/path should always end with a success end condition.>

Extensions

<Enter Extensions and their steps here>

<Enter any extensions here. Extensions are branches from the main flow to handle special conditions. They also known as Alternate flows or Exception flows. For each extension reference the branching step number of the Main flow and the condition which must be true in order for this extension to be executed.

Example of an Extension in Rent Video Use Case:

4a. In step 4, if the customer has accumulated late returns fee greater than ten dollars

1. System will prompt for payment of the dues

2. Customer pays the dues

3. Store clerk adds the amount to the total

4. Use Case resumes on step 4.

>

Variations

<Enter variations here>

<Enter any data entry or technology variations such as – different methods of data input, screen/module invocation, etc.

e.g.
3'. In step 3, instead of reading Video Id using a bar code scanner, the store clerk may enter it directly using the keyboard.>

Frequency: <Enter Frequency of execution here>

< How often will this Use Case be executed. This information is primarily useful for designers.

e.g. enter values such as 50 per hour, 200 per day, once a week, once a year, etc.>

Assumptions

<Enter any assumptions, if any, that have been made while writing this Use Case.

e.g. For *Withdraw Cash* Use Case(ATM system) an assumption could be:
The Bank Customer understands either English or Spanish language.>

Special Requirements

<Enter any special requirements such as Performance requirements, Security requirements, User interface requirements, etc. Examples:

Performance

1. The ATM shall dispense cash within 15 seconds of user request.

User Interface

1. The ATM shall display all options and messages in English and Spanish languages.

2. The height of letters displayed on the display console shall not be smaller than 0.5 inches. (Reference - Americans with Disabilities Act, Document xxx, para xxx).

Security

1. The system shall display the letters of PIN numbers in a masked format when they are entered by the customer.
i.e. Mask the PIN with characters such as **. Rationale – This is to ensure that a bystander will not be able to read the PIN being entered by the customer.

2. The ATM system will allow user to Cancel the transaction at any point and eject the ATM card within 3 seconds. Rationale – In case the customer in duress/in fear of own security he/she needs to quickly get away.

3. The ATM system shall not print the customer's account number on the receipt of the transaction.

>

Issues

1.

<List any issues related to the definition of the use case.

Example

1.What is the maximum size of the PIN that a use can have? >

To do

1.

<List any work or follow-ups that remain to be done on this use case.

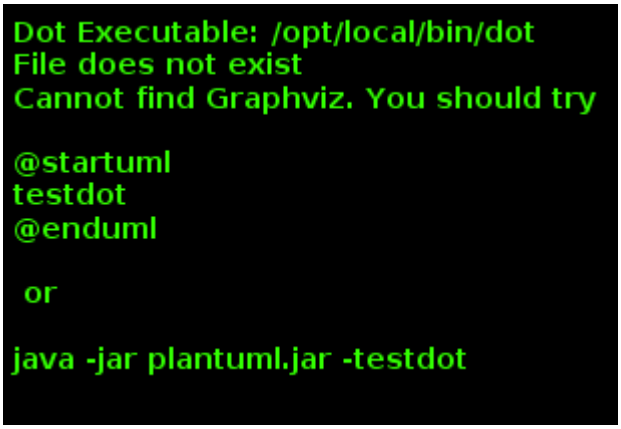
Example

1. Obtain the sales tax table for computation of tax on video rentals from user.

2. Need to ensure that we have covered all parties under the 'Stakeholders and Interests' heading. >

To learn more about "TopTeam Analyst for Use Cases" visit www.technosolutions.com

Modèle de classes du domaine



Invariants

```
context Etudiant::age() : Integer
post correct: result = (today - naissance).years()

context Typename::operationName(param1: type1, ...): Type
post: result = ...

context Typename::operationName(param1: type1, ...): Type
post resultOk: result = ...
```

Dictionnaire de données

Term	Short Description	Meaning
Port	a	ax
Publication	Activity	Registers a Service with a DNS Responder.
Registration		See Publication
Package		Minimum size = 9,000 bytes
One-Shot Multicast DNS Queries	bla	bla
Continuous Multicast DNS Querying	bla	bla
Service Instance Name	a	a
Service Type	bla	bla

Conclusion

Requirements Specification

Introduction

Purpose

The purpose of this document is to describe the requirement specifications for the project «Risk» for software engineering students.

The intended audience of this specification includes the prospective developers of the tool, as well as the technical assessment personnel.

Document Conventions

None so far.

Intended Audience and Reading Suggestions

Project Scope

The software system to be produced is a simplified version of the Hearthstone online game, which will be referred to as «Risk» thorough this document.

The Risk system will allow players from different locations to confront each-other in short and intensive games.

References

1. IEEE Standard 830-1993: IEEE Recommended Practice for Software Requirements Specifications

Overview

The rest of this document contains an overall description of the Risk software system (section Overall Description), the specific functional requirements (section System Features), and the non-functional requirements for the system (see Other Nonfunctional Requirements.

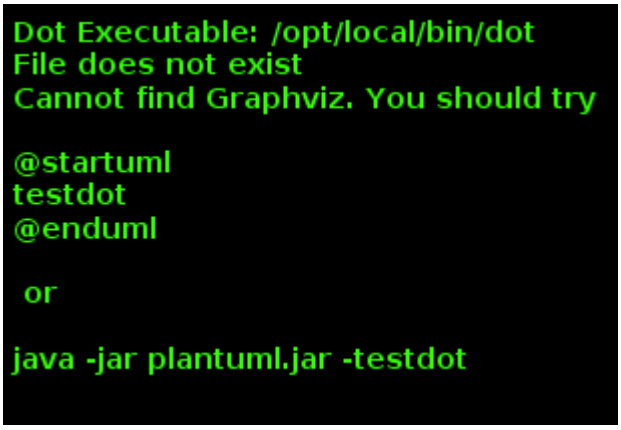
Overall Description

Product Perspective

Hearthstone is a card game where two players confront each-other. The Risk software should allow players that are connected to the Internet to use their connected devices to play. Thus, «Risk» is an online, electronic version of the card game.

While the system is distributed and organized in different components, players should perceive it as a single piece of software. Figure UML Deployment Diagram presents the overall architecture of the software. Players interact with a Web Client, which uses the HTTP protocol to communicate

with (at most) one Game Server. Servers use TCP/IP to communicate with a Database Management Server, which stores all software data.



UML Deployment Diagram

Product Functions

The Risk software must provide two main functions:

1. Game creation: allowing two players to discover each other and start a game.
2. Game play: allowing players to actually play Risk until the victory of one of them.

User Classes and Characteristics

The Risk software has only one class of user: players. Players may have different levels: beginners, intermediate, or expert. However, independently from their level, players should use the same user interface to play against each other.

Operating Environment

The Risk software should operate on any popular and recent operating system: Linux, Windows, or MacOS. The Web Client should operate on any recent web browser: Firefox, Chrome, Safari, or Edge.

Design and Implementation Constraints

1. The Game Server must be developed in Java (version 1.8), using the Spring Framework.
2. The Client must be developed in TypeScript (version 3.1), using the Angular Framework.
3. All software artifacts must use a building too: Maven or Groovy for Java, npm for TypeScript.
4. Dynamic tests must use JUnit (version >= 5.0) and Jasmine (version >= 3.5.0).
5. The code must log its main operations using SLF4J.

Verification Constraints

1. Test Doubles must be used to test each component independently.
2. Each unit test must describe its intention.

User Documentation

No user documentation is required for the first version of the software.

Assumptions and Dependencies

None until now.

External Interface Requirements

User Interfaces

Hardware Interfaces

The software does not interact directly with any hardware device.

Software Interfaces}

The client part of the software must operate on web browsers, whereas the server part must interact with a database through the Java Persistence API (JPA).

Communications Interfaces

Communications between the client and the game server must use Websockets.

System Features

Game initialization

The Risk software must allow the setup of a game with two players and automatically prepare and distribute cards.

Description and Priority

See Chapter [domain] (domain analysis) for further details.

Stimulus/Response Sequences

Functional Requirements

Game play

The Risk software must allow two players to play against each other until a winer is settled. See Chapter [domain] (domain analysis) for further details.

Other Nonfunctional Requirements

Performance Requirements}

1. The game must be *playable*, meaning that users must have fast feedback for their actions and delays due to communications/connection problems must be correctly held.
2. The Web Client must be able to execute on a personal computer with 4GB of RAM.

Safety Requirements

Security Requirements

Software Quality Attributes

Extensibility

The software must be extensible, it must be easy for developers to add new cards and heroes to the game.

Maintainability

1. The software must be readable and easy to maintain.
2. The Java source must respect Google's guidelines: https://google-styleguide.googlecode.com/svn/trunk/javaguide.html

Business Rules

Other Requirements

Appendix A: Glossary

Appendix B: Analysis Models

See Chapter [domain] (domain analysis) for further details.

Appendix C: To Be Determined List

Spécification des composants

Ce livrable correspond à la "conception préliminaire" du projet. Il comprend la division de la solution en différents composants, l'explication des fonctionnalités attendues de chaque composant, la spécification des interfaces fournies par chaque composant, ainsi que des diagramme de séquence qui valident ces interfaces.

Objectif

Préciser les objectifs de ce chapitre.

Organisation du chapitre

Cette section décrit le contenu du reste du chapitre et explique comment le document est organisé.

Description des composants

Établir les frontières du système.

Division du système en composants.

Décrire le comportement souhaité des composants.

Le composant `Game Server`

Décrire succinctement le comportement du composant.

Spécification des interfaces

Spécification de l'interface A

Présentation de l'interface en UML (ou HUTN). Description du comportement de chaque opération. Spécification éventuelle des pré-conditions en OCL.

Spécification de l'interface B

Le composant `Web Client`

Décrire succinctement le comportement du composant.

Spécification des interfaces

Spécification de l'interface A

Présentation de l'interface en UML (ou HUTN). Description du comportement de chaque opération. Spécification éventuelle des pré-conditions en OCL.

Interactions

Objectif: décrire, à haut-niveau, la collaboration entre les composants majeurs, en faisant référence aux besoins.

Utiliser des interactions, c’est à dire, des diagrammes de séquence et des diagrammes de communication.

- Ne vous limitez pas à une seule interaction par cas d’utilisation

Mise en place d’un jeu

Interaction: cas nominal

Interaction: cas A

Interaction: cas B

Tour d’un joueur

Interaction: cas nominal

Interaction: cas A

Interaction: cas B

Conception détaillée

Le composant WebServer

Le composant WebClient

Conclusion

Points forts de votre projet

Limites (points faibles) de votre projet