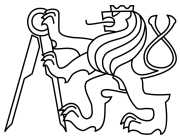
CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF INFORMATION TECHNOLOGY



ASSIGNMENT OF BACHELOR'S THESIS

Title:	Analysis, Design, and Simulation of Standalone Gaming Machine
Student:	Bc. Martin Novák
Supervisor:	Ing. Jiří Hunka
Study Programme:	Informatics
Study Branch:	Information Systems and Management
Department:	Department of Software Engineering
Validity:	Until the end of summer semester 2017/18

Instructions

Perform business analysis, technical design, and analysis of costs and benefits of a standalone gaming machine for the German market. Respect the law requirement defined by [1] and [2]. The business analysis will contain a list of requirements for implementation. The technical design must contain appropriate UML diagrams and a project plan using WBS (Work Breakdown Structure). Design, implement, and test a simulator of the gaming machine in Java SE that demonstrates the main functionalities including user interface.

References

[1] Technical Guideline Version 5.0 dated 27th January 2015 by Physikalisch Technische Bundesanstalt.
[2] Gaming Ordinance in the version published on 27 January 2006 (Federal Law Gazette I p. 280).

L.S.

Ing. Michal Valenta, Ph.D.
Head of Department

prof. Ing. Pavel Tvrdík, CSc.
Dean

Prague October 11, 2016

Czech Technical University in Prague
Faculty of Information Technology
Department of Software Engineering

Bachelor's thesis

Analysis, Design, and Simulation of Standalone Gaming Machine

Martin Novák

Supervisor: Ing. Jiří Hunka

8th January 2017

Acknowledgements

I would like to thank especially to my thesis supervisor Mr. Jiří Hunka, who helps me a lot during the process of laying down the topic of this thesis. Next big thanks belong to Mr. Miroslav Balik, vice dean of Faculty of Information Technology, who proved a lot of understanding of situations, thru which Saturday school students are generally dealing with.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended. In accordance with Article 46(6) of the Act, I hereby grant a nonexclusive authorization (license) to utilize this thesis, including any and all computer programs incorporated therein or attached thereto and all corresponding documentation (hereinafter collectively referred to as the "Work"), to any and all persons that wish to utilize the Work. Such persons are entitled to use the Work in any way (including for-profit purposes) that does not detract from its value. This authorization is not limited in terms of time, location and quantity. However, all persons that makes use of the above license shall be obliged to grant a license at least in the same scope as defined above with respect to each and every work that is created (wholly or in part) based on the Work, by modifying the Work, by combining the Work with another work, by including the Work in a collection of works or by adapting the Work (including translation), and at the same time make available the source code of such work at least in a way and scope that are comparable to the way and scope in which the source code of the Work is made available.

In In Prague on 8th January 2017 .....

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Novák, Martin. *Analysis, Design, and Simulation of Standalone Gaming Machine*. Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2017.

Abstrakt

Smyslem této práce je rozbor problematiky návrhu výherního hracího přístroje určeného pro trh Spolkové republiky Německo. Návrh je zpracován v souladu s technickými požadavky definovanými směrnicí TR5 pro výherní hrací přístroje tohoto typu. Součástí návrhu je soupis požadavků, vysvětlení problematiky a klíčových charakteristik daného trhu, analýza řešení v notaci UML, strukturovaný soupis implementačních prací v notaci WBS a zdrojové kódy prototypu demonstrujícího základní funkcionalitu v programovacím jazyku Java.

Klíčová slova Závěrečná práce, LaTeX, výherní hrací přístroj, TR5, WBS, UML, analýza nákladů a přínosů, Java.

Abstract

The purpose of this paper is to analyze the requirements for gaming machine destined for the market of Germany. The technical design is prepared in compliance with the technical requirements laid down by Directive TR5 for gaming machines of this type. The paper includes a list of requirements and explanation of the key characteristics of the market, analysis of solutions in UML

notation, structured inventory of the implementation work in algebraic notation WBS and source codes prototype demonstrating the basic functionality of the Java programming language.

Keywords Thesis, LaTeX, Gaming Machine, TR5, WBS, UML, cost-benefit analysis, Java.

Contents

List of Figures

Chapter 1

Thesis Task

Perform business analysis, technical design, and analysis of costs and benefits of a standalone gaming machine for the German market. Respect the law requirement defined by [1] and [2]. The business analysis will contain a list of requirements for implementation. The technical design must contain appropriate UML diagrams and a project plan using WBS (Work Breakdown Structure). Design, implement, and test a simulator of the gaming machine in Java SE that demonstrates the main functionalities including user interface.

References

1 Technical Guideline Version 5.0 dated 27th January 2015 by Physikalisch Technische Bundesanstalt.

2 Gaming Ordinance in the version published on 27 January 2006 (Federal Law Gazette I p. 280).

Introduction

The topic of this final thesis is focused on problematics of new case law in German gaming industry. The new law came to a force in January 2016 and let to the big disruption affecting the market. German legal authorities made significant step forward in their fight against gambling and created series of precisely specified requirements for vendors and operators in order to allow them to produce slot machines according to new regulation.

The purpose of the thesis is to summarize legal requirements mentioned above and design a solution which would either satisfy these requirements but also demonstrate the proper functionality of the concept. The designed solution is meant to be as an approach toward future changes on currently functioning system. This means that its based-on assumption that there already exists a functioning general system representing the graphical view of the games and general functionality. This document is not focused on winning mathematics of the games, their visual appearance or managing and handling peripherical devices such as coin acceptors and dispensers. It is focused strongly on solution specific to new regulation of German gaming market and what are necessary changes which needs to be implemented on every new deployed gaming machine to the market.

The reason for choosing this topic was gained knowledge during employment of the author in one of major Czech gaming machine manufacturer. The final design, which was the outcome of analysis described in this thesis, is used as foundation for further development. Major contribution of this thesis is therefore enlightenment into this problematic. Because by the time of publishing this thesis there is no other academic material or any other public information. The only resource is the legal document and other technical legal specifications which only mention scenarios and requirements about functionality which is forbidden or must never occur. This thesis takes that information under accountment and is focused on actual solution. That means the solution how the final product should look like and behave in order to satisfy legal requirements and be competitive on the market.

The thesis contains multiple sections, each focused on specific part of the topic. In introduction part, there are several information about history of gaming machines worldwide and brief description of specifics regarding German market with gaming machines. I other sections thesis briefly summaries major legal requirements defined by new upcoming law and new regulations of the market. The practical part is focused on invention of solution matching these requirements. That section contains technical description of final design, UML diagrams outlining logic of the system, WBS describing foundation for development planning and functional prototype simulating the designed logic in practice.

Chapter 2

General overview

2.1 Description of gaming machine

Gaming machine (sometimes referred as slot machine or fruit machine) is electromechanical device. It contains multiple reels which spin when an appropriate button or lever is pushed or pulled. In the past, the reels were mechanical as well as other analog user interface elements for the player. Nowadays, the reels and many buttons are placed on electronic visual display, which allows more possibilities and better user experience.

As the term indicates, the gaming machine is designed for gaming. It provides the player with amusement during gameplay, but also provides a chance for winning the money. What is probably the most curious question, what is the probability of win? The probability of win differs from manufacturer to manufacturer. It depends on the type of mathematics inside the games, but there are several general characteristics for all the machines. It must provide the same chance of win to every player in long run. Sometimes, machines offer to switch between multiple mathematics profiles. So, the operator might choose the profile, based on what types of players attend his casino or gaming hall. But generally, the mathematics of the game is the most valuable logic in the machine so it is under watch of certification authorities. Again, the certification process differs from country to country, but usually, while the machine is manufactured, it must be submitted for certification, after it receives certification stamp, it might be operated. But most of the time, the certification authority doesn't want anybody to tweak the game mathematics, so it grants the certification only if there is certainty, that nobody will be allowed to adjust game mathematics in the field. Because that would lead into fraudulent behavior against the player.

Machines also provide some possibility for the player to insert banknotes or coins. So, the machines are equipped with coins or banknotes acceptors or card readers. Paying out of the money is realized via coin or banknote dispensers, sometimes via printing out of a winning ticket.

Figure 2.1: Sittman & Pitt's Poker Machine from a year 1891.[2]

The concept of this type of games is to win in form of receiving the winning combination of symbols displaying on reels. Player start the game by appropriate means, the reels start spinning and then they randomly stop. If the reel position contains two or more symbols of same type, which are in horizontal or diagonal position toward each other, then player receives the price. The value of price also depends on the winning combination, especially on the type of symbols. Some symbols appear more rarely and therefore there is lower probability of getting the winning combination with rare symbols. So logically, if rare winning combination appear, player receives higher price.

2.2 Brief History

The origins of slot machines can be traced back to the late 19th Century. The first slot machine was developed by the New York based company, Sittman and Pitt in 1891. The game had 5 drums with a total of 50 playing cards. The machine could be found in many bars, and cost a nickel to play. Players would insert their money and pull the lever to play. Payouts were made for lining up poker hands on the reels. In order to increase the house edge, 2 cards were removed from the machine – the ten of spades and the jack of hearts. This reduced the odds of getting a royal flush by half. The machine had no direct payout mechanism, so wins were paid at the bar. These were non-monetary prizes, such as free drinks and cigars.[1]

The problem with poker machine was pay out mechanism and mechanism which would determine the level of win. Poker cards allowed many combina-

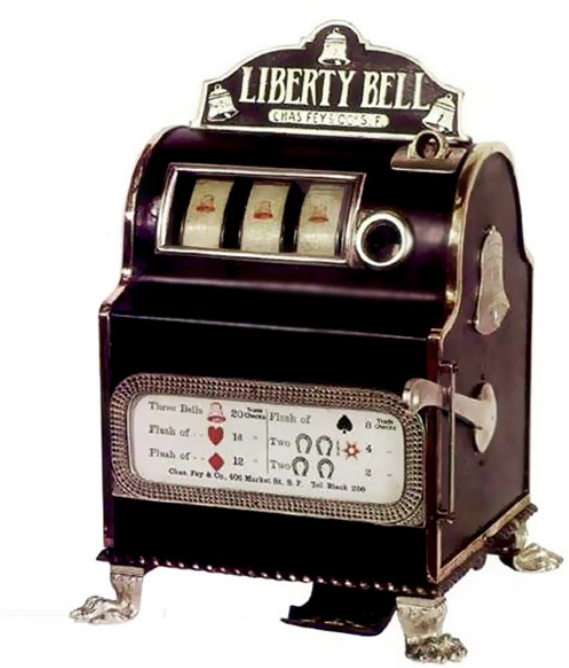


Figure 2.2: Charles Fay's Liberty Bell slot machine.[4]

tions, where was nearly impossible for the mechanical machine to detect the winning combination. So there was need for a reduction of these possibilities. That was a biggest contribution of Charles Fay and his Liberty Bell machine.

The first mechanical slot machine, as we know them today, was the Liberty Bell, invented in 1895 by car mechanic, Charles Fey (1862–1944) of San Francisco. The Liberty Bell slot machine had three spinning reels. Diamond, spade, and heart symbols were painted around each reel, plus the image of a cracked Liberty Bell. A spin resulting in three Liberty Bells in a row gave the biggest payoff, a grand total of fifty cents or ten nickels.[3]

The biggest boom of slot machines arrived in 30's in 20th century and continued till late 70's when it was possible to find this type of the machine anywhere around the world. Last half of 20th century was also crucial about regulation of gaming industry where American market held leading role. During this era, gaming industry was extremely profitable and it did not take long when legal authorities noticed. 60's and 70's was the golden era in United states and despite all the externalities gaming industry produced, it also attracted organized crime because of its profitability. Very soon the legal authorities started to regulate the industry in order to prevent people from gambling addiction. Secondary reason was that even the government wanted cut a share on gambling profits in form of special taxation. In these days, American market is the most regulated one in the world and gaming industry

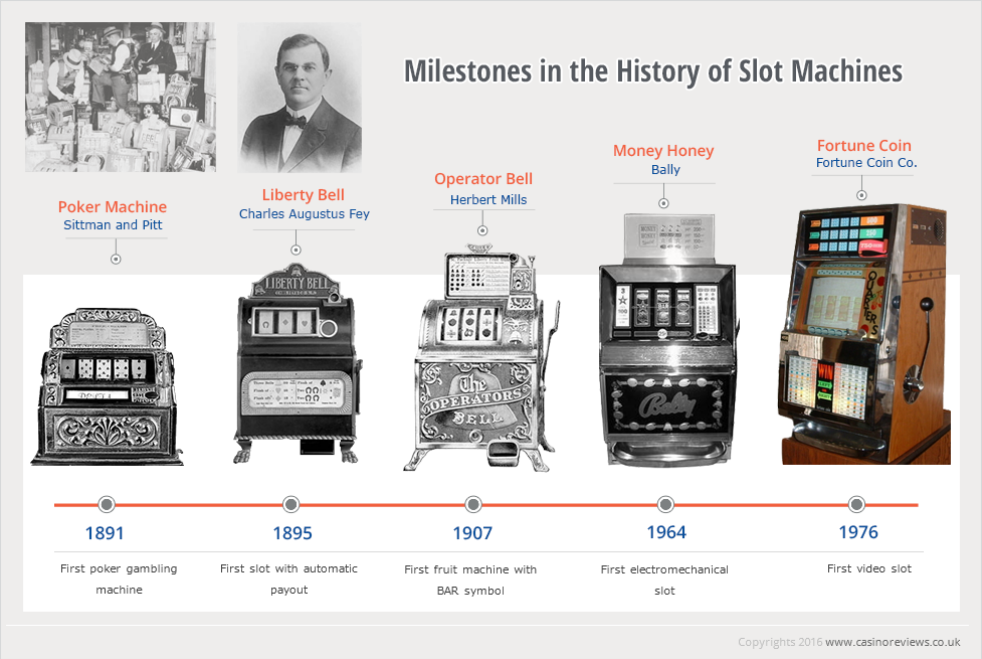


Figure 2.3: Slot machine evolution milestones.[5]

is not associated with organized crime anymore.

2.3 Gaming machines at present time

Current situation in gaming market is very diverse. With the growing technological possibilities during 21st century, the gaming machines became very diverse and games became more and more sophisticated. Market is now used to more advanced gaming experience which is brought by higher game interactivity via multiple special bonuses, free games and other features. What remained the same is the concept of spinning the reels. This idea still outlived all the other dynamical changes in the industry and it doesn't seem that it's going to change anytime soon. This concept is the most characteristic for this type of game so it has remained the same for more than hundred years.

With the growing possibilities for gaming machines has also grown the regulation. Like in the United States, in many other countries was a need for regulating this industry. Current world trend is that each country has its own regulatory politics. In some countries, the power over the ban of gaming is granted to smaller jurisdictional units like counties or districts. Then it's up to the local governments if they want to allow gaming in their region. And they must always consider both sides of this decision.

Chapter 3

General functionality of gaming machine

This chapter describes the general mechanism and behavior of game in gaming machine. Following chapters will be based on extending this model in order to satisfy requirements for German market.

3.1 Hardware

Hardware structure of gaming machine mostly depends on individual solution of each manufacturer. Although, it is possible to name many similarities, which are common for this type of device.

All hardware which is visible to the player is usually metal case, LCD screens, panel with hardware buttons and possibly coin or banknote acceptor (sometimes even dispenser). On the inside there is usually PC based infrastructure with other hardware elements. Usually, manufacturers don't fabricate hardware by themselves so they use third party hardware which is not supported by normal PC platform by default. So sometimes it is necessary to use another hardware interfaces in order to make all components to work together. On the inside of a machine, there is usually visible cashbox and hardware/mechanical counters which independently measure bets and wins during the gameplay and serve as additional level of protecting the owner of the machine to fake his earnings due to tax purposes.

3.2 Software

3.2.1 Architecture

Nowadays, many gaming machines are built on PC based platform. OS runs an application which represents the logic of a game. From architectural point of view, solutions differs from manufacturer to manufacturer.

The most important architectural differences are in connectivity of the machine. Nowadays, gaming machine are of two basic types - online and offline. These two versions have their positives and negatives. When it comes to online version, it is considered to be a solution with higher protection against unwanted intervention to the machine (for example hacking the machine). Because of all the winning combinations and random number generator runs on distant server, which is also monitoring all the events on the machine, it is much harder for the attacker to influence the outcome of a game and earn some benefit from it. Online solution is usually more preferable for the operators of the machines. Because they don't have to buy the machine in order to run it. They are only renting it from the manufacturer, which is saving the initial cost and they are also provided with support, upgrades of the game and the rent cost is also tax-deductible. Its similar like operation leasing. On the other hand, the offline solution is fully independent from the manufacturer. The operator is buying the machine, he is paying full price for it and therefore it become his property. Then operator can even resell the machine after he doesn't want to use it anymore.

The design made for German martket has to be the offline version. It is not choice of the oprator or manufacturer, but it is one of the legal requirements.

3.2.2 Game Menu

All slot machines are usually made with several built-in games. From the marketing point of view, it is always better to offer the player some kind of diversity. So in order for the player to be able to navigate from game to game, there needs to be an game menu, where player can choose which game to play. Machine can also contain more games, which don't fit into a menu on one screen, so some machine offers multiple screen menu. The transition to some other screen is usually triggered by appropriate button. After the player selects the game, he simply presses the button on the screen containing logo of particular game and a game application starts.

3.2.3 Game GUI

Game GUI contains several interactive elements. The largest part is usually the area with reels. Then, at the bottom of the screen, there is panel with buttons and forms/labels. Labels displays the current level of credit available for gameplay (usually in monetary units or chips), value of bet for spin and value of win from previous round. If game allows player to choose number of winning lines, then there is also a label displaying this information. When it comes to buttons, then there is an interactive button for starting the game (starts spinning reels), then there are buttons for increasing/decreasing bet and button for AutoStart, which has a functionality of automatically initiating new game at the end of previous one.

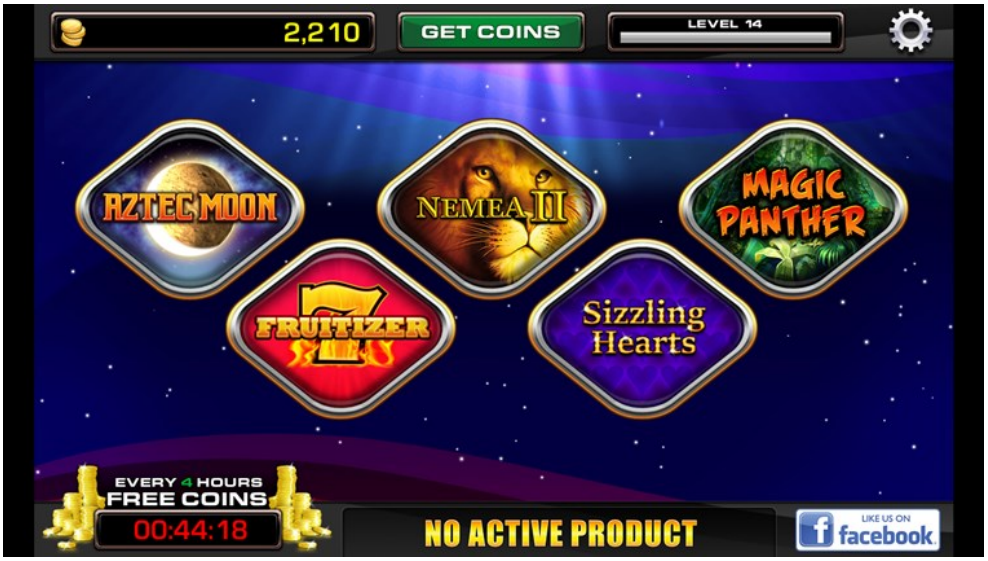


Figure 3.1: Slot machine game menu.[6]

3.2.4 Administration

In order to perform some administrative action on the machine, such as reading the game statistics, reset the credit value, set connection to gaming server (if available) and much more, there is a possibility to enter administrative menu. For entering the administrative menu, most manufacturers are using concept of so called administrative key.

Administrative key can have multiple forms. Some manufacturers are using RFID card, some are using regular mechanic key, some are using 6,5 millimeters JACK connector etc. By inserting the key into the keyhole of the machine, the administrative menu appear and operator can perform administrative action. By removing the key, the machine returns into previous state, displaying game menu or game screen.

Figure 3.2: Slot machine game example.[7]

Chapter 4

Specifics of the German market

4.1 Current situation on the market

The situation on German market with gaming machines has radically changed with new regulation in form of TR5. The older technical guideline (TR4) became obsolete and new TR5 came with many newly introduced requirements. So, former machines operated under TR4, will not be usable after the expiration of their certificates. Thats another important thing. When an application of a new law comes to a force, it doesn't mean that all operators and all manufacturers have to change the design of the machines overnight. There is always a transition period, when market is adapting to newer legislative. And that is the same with transitioning from TR4 to TR5. Each gaming machine certified under TR4 got its own certificate of approval, which is valid for 2 years. Then the owner of certified machine can operate it for 2 years (with assumption, he owns a license for operating gaming machines of course). So, with the upcoming TR5, many casino and gaming halls operators tried to get as many new certificates under old TR4 for as possible, so they could operate the machines they know, and which players know, as long as possible.

The TR5 passed thru legislative process in 2015, with coming to force with 1st January 2016. From that point, no new machine could pass the certification process under TR4. From this point, it is certain that by the 1st January 2018, there will be no machine with certification under TR4 operated in the whole market. Old certification approval will keep expiring and operators will be having just one possibility and that is to purchase the machines certified under the new TR5. So, the race of manufacturers has begun. There are less than two years for designing a machine, which would have specifics according to TR5, assemble the machine, or build it, and pass the certification process. The certification process can take up to a year itself or even longer. This depends on how many applications for type approval will be passed to certification authority (PTB). So many manufacturers know that who comes first will probably take a big portion of the market.

4.2 Specifics of gaming machines on the German Market

Before the introduction of TR5, the gaming machines deployed on the German market had to comply with TR4. Former technical ordinance was really similar to the newly introduced TR5, but was quite outdated in multiple ways. The main areas of regulation regarding the gameplay was length of playing, maximal bets and maximal wins. The technical requirements in TR5 involve the same areas, but in more specific way. The biggest obstacle for the manufacturers were the requirements regarding maximal wins in specific time period. This requirements have major impact on all the mathematics of the game. And they also affect the operators in a way of loosing the customers. Because players were used to the idea that they are able to win quite big amount of money in every spin. And now suddenly, there was a legal requirement ordering the manufacturers to implement logic, that when player receive winning combination and win some amount of money in one round, it must not be possible he would win another winning combination anytime soon. Then machine won't be allowed to grant player of another win in certain time period. And also, the maximal allowed value of next win would be dependent on time passed between those two wins. Same idea was applied on betting limits. Technical guidelines state that player is allowed to bet certain amount of euro in one round, but then he must be forbid to perform another bet in some time interval. And after the time interval, the value of allowed bet is really low (in amount of eurocents). The allowed value of bet can get higher again, but player must choose not to perform low level bet and wait. And after some time, the value of allowed bet gets higher and higher, after it hits the maximal value of bet.

The limitations of bets and wins were really crucial for the manufacturers of gaming machines, because it was obvious that these type of changes would completely ruin the classical idea of gaming machine that the players were used to. That was probably an intention of the new regulation. But the market found its way. It is hard to tell who came first with the idea, but soon, all the manufacturers found the way how to go around of the idea of the regulation of bets and wins. The trick was in adding new game feature of "converting" the real money into "chips". And all the prices from spinning the reals were added or subtracted from the amount of chips, player had available. From the legal point of view, the conversion process could be considered as the gameplay as the law was describing it. All the conversion from real money to chips were bets, and all the conversions from chips to real money were winnings. So, all the legal requirements about bets and wins were applied into conversion process, which was considering the legal limits in specific time. The only problem with this idea could be the fixed conversion ratio of money equivalent to chips. Because certification authority could see that conversion

process doesn't involve any random chance of getting some wins or losses. So, German manufacturers solved this issue by implementing random conversion algorithms, which add some source of randomness into the conversion, so every time player performs some conversion from real money to chips, he gets different amount of chips each time. And also, when player performs conversion of some amount of chips to real money, he gets different amount of euros. This logic influenced the German market the most. Players started to get used to when they wanted to start playing the reels, first they had to input the banknote into the machine, then perform conversion of some allowed value of money to chips and then start betting the chips inside the game. When they wanted to withdraw the winnings from the game, they had to convert won chips back into real money and cash out the machine. Because of the limits, sometime, when it is needed to convert big amount of chips into money, it took quite a while. Sometimes even hours. But the players and operators got used to this way at the end.

Other quite specific characteristics for the German based and certified gaming machines are the idea of long term pay outing. Because of the legal requirement of limiting the winnings in time, there was a need for solving the situation when player received some big win in chips and wanted to pay it out and go home. As was said above, the legal limits of winning were now applied to conversion process so player couldn't convert all his chips to euros in small time. So, the manufacturers designed and approach how to solve this situation. The idea was that player selects or activates the feature of long term pay outing and machine automatically transfers chips to money in order not to breach any limits and keeps paying out the euros in these waves. So, the players were satisfied that he gets all his euro values of chips back, all he needed to do is wait. In the meantime, they could always play different gaming machine.

Chapter 5

Requirement Summary of TR5

5.1 States of the Machine

The most important limitation when it comes to responsible gaming is institute of mandatory breaks. TR5 defines these breaks which have to be taken after specific time period of playing a gaming machine. There are several limits which have to be obliged and control by gaming machine all the time.

5.1.1 Active State

This state is the only one, where TR5 allows gaming activity. In the case of this design, this means, this is the only state, where the conversion between Player's Wallet and Chips Pile is allowed. Also, it is the only state of the machine, where any kinds of animations on the display are allowed.

When the machine is in Active State, it must count the time spent in this state in form of so called gaming time elapsed. When machine is in Active State for specific time period, it must automatically initiate a break and enter into appropriate break state.

Last requirement about Active State is presence of player's key. The key may have a form of a chip card, RFID card, actual key or any other kind of key system. This is not specifically defined. Only thing which is defined is that player's key must be unique to every machine, must be only one and machine may be in Active State only if player's key is inserted.

5.1.2 Forced Interruption Break State

TR5 defines Interruption Break State as mandatory break, which has to be taken after at least one hour of spent gaming time. This means that after one hour of playing there has to be Interruption Break taken, which will last at least 5 minutes. During this break, no gameplay is allowed. From the TR5

point of view. In our case, this means no conversion from Player's Wallet to Chips Pile will be allowed.

Last mandatory requirement about Interruption Break is, while this break is initiated, all player's funds must be emptied out. In our case, this means, that Player's Wallet must be hopped out.

5.1.3 Forced Idle Break State

Idle Break is special form of break which so speak resets the machine into its default state. TR5 states that Idle Break must occur at least after 3 hours of gaming time elapsed and must last at least 5 minutes. Also, no conversion transactions are allowed during this break time and when this break is initiated, all player's funds must be hopped out. In this case, Player's Wallet will be hopped out. And because of mandatory reset of the machine, the Chips Pile has to be erased. This is very unfortunate, because that will cause loss of money to the player. But there is no other way to go around this. The impact of this requirement is, that there will be no longer gaming session with longer time duration than 3 hours.

5.1.4 Voluntary Breaks

This special break state solves the need of having a possibility to voluntarily interrupt gaming activity. All the breaks mentioned above are defined as mandatory breaks which have to be taken after certain time. But what if player would want to interrupt Active State willingly and take a voluntary pause? For that reason, TR5 introduces Voluntary Break State.

Any time player removes his player's key or player's card out of the machine, machine must not remain in Active State. While this is done during mandatory breaks, it has no effect on the machine. Machine must stay in mandatory break for certain time anyway, so the presence of player's key has no effect on any state transitions. But, something different is, while machine is in Active state. The TR5 requirement clearly says, that machine must not stay in Active state, when player's key is removed, so in this case, machine performs transition from Active State to Voluntary Break State.

One of the major requirements about Voluntary Break State is that while Voluntary Break is initiated, it must not be interrupted for 15 seconds. It means that during the gameplay, player cannot remove his player key and put it right back in and continue playing. If the player does so, machine must remain in Voluntary Break state for minimal mandatory time of 15 seconds.

To be more precise, Voluntary Break is not defined as special break state next to Idle Break and Interruption Break. Voluntary Break must always be of type either voluntary Idle Break or voluntary Interruption Break. So, when player removes his key during the gaming, machine can actually decide, which type of voluntary break to enter. But when entering specific type of

voluntary break, the conditions for this type of break must hold. So, if machine wants to enter Voluntary Idle Break, the chips level must be zero as well as Player's Wallet must be zero. If chips are not zero, machine could go to voluntary Interruption Break state. In the case of this design, the machine can enter Voluntary Interruption Break any time, because there is no obstacle for emptying Player's Wallet anytime. Player's Wallet contains funds in euros, and from TR5 perspective belongs to player all the time. The tricky thing is Voluntary Idle Break. Because Voluntary Idle Break is defined as break state when Player's Wallet and chips pile must be set to default value. In any words, they must be cleared out. There is no technical obstacle by clearing out Chips Pile, but that is not something which would please the player very much, that machine would suddenly cleared out all his chips.

The last thing worth mentioning is relationship of Mandatory Idle/Interruption Break and Voluntary Idle/Interruption Break. This consequence can already be derived from what was said above, but just to make this really clear. Mandatory breaks define break states, which must occur no matter what after predefined elapsed time spent in Active State. Then machine must stay in mandatory break for predefined time period or longer. Voluntary breaks can occur any time, as long the conditions for entering specific voluntary breaks are met. The machine must stay in voluntary break at least 15 seconds and can last as long as it is needed. But technical outcome, which is also mentioned in TR5 is that after some time spent in some type of voluntary break, which defines some kind of mandatory break, this voluntary break may be accounted as mandatory break taken in advance. This sentence might sound a bit confusing, so the practical example will follow.

Example of transitions and accountment of breaks:

- Machine has been running in Active State for 30 minutes, player plays games, transferred his money into chips, so he has some funds in Player's Wallet and Chips Pile at the same time. He has played for 30 minutes, so from the definition of Interruption Break and Idle Break, he has 30 minutes of gaming time left to Mandatory Interruption Break and 2 hours and 30 minutes to Mandatory Idle Break. Now player removes his player's key and machine the machine has to go to some break state, because by the definition in TR5, machine must not be in Active State, when player's key is not present. So, machine will automatically dispense player's money which are in Player's Wallet and because there are still some chips present in Chips Pile, machine will enter Voluntary Interruption Break. After let's say 10 minutes, player inserts his player's key again and machine enters Active state again. It may, because it has spent at least 15 seconds in break, which was obligatory. But because it has stayed in Voluntary Interruption Break for more than 5 minutes, this Voluntary Break got accounted as Mandatory Interruption Break

taken. Because more than 5 minutes in break has elapsed, which is a requirement for Mandatory Interruption Break. So, from now on, another Mandatory Interruption Break must occur in 1 hour of gaming time, the Mandatory Idle Break must occur still after 2 hours and 30 minutes.

- Machine has been running in Active State for 30 minutes, player plays games, transferred his money into chips, so he has some funds in Player's Wallet and Chips Pile at the same time. He has played for 30 minutes, so from the definition of Interruption Break and Idle Break, he has 30 minutes of gaming time left to Mandatory Interruption Break and 2 hours and 30 minutes to Mandatory Idle Break. Now player removes his player's key and machine the machine has to go to some break state, because by the definition in TR5, machine must not be in Active State, when player's key is not present. So, machine will automatically dispense player's money which are in Player's Wallet and because there are still some chips present in Chips Pile, machine will enter Voluntary Interruption Break. Player will now enter player's key after 3 minutes spent in Voluntary Interruption Break state. Machine enters Active State, because it can. Because the voluntary break last more than 15 seconds. But his Voluntary Interruption Break will not get accounted as mandatory break taken. Because the voluntary break last just 3 minutes and the requirement for taking Mandatory Interruption Break is 5 minutes. So, the 3 minutes spent in voluntary break will not get accounted as gaming time, but also won't get accounted as mandatory break. It means that player will still have 30 minutes gaming time left to Mandatory Interruption Break and 2 hours and 30 minutes to Mandatory Idle Break.

- Machine has been running in Active State for 30 minutes, player plays games, transferred his money into chips, so he has some funds in Player's Wallet and Chips Pile at the same time. He has played for 30 minutes, so from the definition of Interruption Break and Idle Break, he has 30 minutes of gaming time left to Mandatory Interruption Break and 2 hours and 30 minutes to Mandatory Idle Break. Now player removes his player's key and machine the machine has to go to some break state, because by the definition in TR5, machine must not be in Active State, when player's key is not present. So, machine will automatically dispense player's money which are in Player's Wallet and because there are still some chips present in Chips Pile, machine will enter Voluntary Interruption Break. And now player inserts player's key after 10 seconds. Machine will now still stay in Voluntary Interruption Break, because according to requirement of voluntary break, it must not be interrupted at least for 15 seconds while it has been initiated. So, player will have to wait at least 5 more seconds, when machine enters Active State again.

Then the 15 seconds long Voluntary Interruption Break will not get accounted as Mandatory Interruption Break, because machine has stayed in break time just for 15 seconds and not for required 5 minutes. Player will then have another 30 minutes of gaming time left to Mandatory Interruption Break and 2 hours and 30 minutes left to Mandatory Idle Break.

5.2 Money and cash insertion

By the directive of TR5, gaming machine receives money only in cash. The main issue with this is that maximal amount of money allowed to be present in machine at specific time moment is 10 euros.

When player inserts banknote, the money will be added up to so called Player's Wallet. That's an entity which will be holding the player's resources in units of real money. On the screen, there will be also visible an entity (label/form) called Player's Wallet, where the amount will be visible. If the Player's Wallet is empty, the player can input maximally 10 euros. If the Player's Wallet is not empty and player inserts money, which when added up with the amount of money present, would reach the limit of 10 euros, gaming machine will automatically pay him back the amount of money, which would exceed the 10 euros' limit. For example, if player has 5 euros in the PW, than amount which can be inserted is at most 5 euros. In this case if he inputs 10 euros' banknote, the machine will add up 5 euros into Player's Wallet and additional 5 will be hopped out.

This 10-euro limit applies only for money insertion. Player can have more money in PW because of conversion of chips to money. If the conversion process continuously proceeds, the limit of 10 euros can be breached quite easily. But in that case, banknote acceptor must be disabled, and player must be unable to input another money into the machine.

5.3 Stakes

Player pays spin prices (bets) only in chips. The actual money cannot be played for. First he needs to convert some money into chips and then he can perform some gaming (spins).

The idea is, that player cannot choose how much money he wants to convert. The actual amount of money which can be possibly converted gets incremented in time, till it reaches its maximal STAKE value (2,20 euro). That's the way how TR5 sees the actual game happening. But in this case, the TR5 requirements will be applied for converting the money into chips. So, the process of conversion from money to chips is initiated by pressing the stake button. Then the accumulated money amount gets converted into chips

using some randomness. Then the value in stake meter gets reset to zero and starts incrementing in time again.

Other thing worth mentioning is, that machine doesn't need to check the conversion limits when stake is being placed. So, when the conversion from money to chips is about to happen. Because the check of limits is performed during incrementation of stake level. In other words, if stake level has certain value, it has already been checked that transfer of this value would not breach the hourly limits.

5.4 Winnings

The concept of winning is similar to stake logic. From the point of view of TR5, winning is a positive outcome of each round of game. From the design point of view it is going to be conversion from chips to real money (from chips to Player's Wallet). TR5 defines the winning limits in time, so the final solution of the gamaing machine has to have these limits implemented by design.

5.5 Game Round

As was mentioned above, from the point of view of TR5, the game will be the conversion process from chips to real money. From the point of view of the player, it is the spin of reels. Because TR5 defines winning limits, which applies on the conversion process, the actual game (one spin of reels) is without any special legal requirements and therefore it's fully up to the design decision of manufacturers. This includes animations, game features, game mathematics and others.

5.6 Cash-Out Process

TR5 clearly states, that player must have an option to cash out his funds any time during his gameplay. There is also a requirement about hardware button, which must be enabled all the time, which will on-press immediately cash out, player's money balance.

As it comes to the design of the machines, hardware payout button will on-press pay out funds available in Player's Wallet only. Chips will not be paid out, because they have not been converted to real money equivalent yet. From the TR5 point of view, they are not in possession of a player.

5.7 Power Outage

In case of power outage, the TR5 states, that on power-up machine has to boot up into same state in which there was power loss. That is a basic approach, which must be followed. Although, there are some limit scenarios or specific situations which could cause money loss to the player.

There are 2 opinions that could happen:

- If at time of power outage the machine is in Active State
  - If interval between the power outage and most recent active status is less than 75 seconds, than the during time of power outage is added to active time
  - If interval between the power outage and most recent active status is more than 75 seconds, than exactly 75 seconds is added to active time elapsed
- If at time of power outage the machine is not in Active State, the time spent in power outage is assessed as break state

5.8 GUI

When it comes to GUI, TR5 defines many requirements which specify what must be shown to the player at what time or what is forbidden to show at what time. For example, during break state, there are no animations allowed whatsoever. Only static text. Other thing is that all the time during gameplay, there has to be visible a text about gaming addiction and information about an option of getting professional help. Also, machine must be equipped with hardware button for cashing out player's balance at any time.

The most of the UI elements are more of a matter of design.[10] There are also associated with special approach toward stakes and winnings and different perspective of understanding the meaning of the single game from the point of view of legislative and actual design of the machine. This problematic has already been discussed above.

5.8.1 In Game

As shown in Appendix A in picture A.1, the game GUI contains multiple elements. They are all necessary for navigation and control of the game flow.

There is stake meter, displaying the value of a stake, which is automatically incremented by the machine, till it reaches its maximal value. While it gets pressed on the screen, or stake hardware button gets pressed, the accumulated

[1]The further described GUI does not describe the GUI of programmed prototype, but the GUI of final machine.

value of stake gets transferred from Player's Wallet to Chips Pile. When the stake meter cannot get incremented because of the limit, the timer label will display over the stake meter, which will indicate the time left when other automatic stake incrementation can occur.

The inverse process of conversion chips into money is handled by the arrow button between Chips Pile and Player's Wallet. On-press, the conversion gets activated or disabled. If the conversion may not be performed due to some limit, the timer gets displayed over the arrow button, which will indicate the time left till the conversion might be performed again.

Other relevant GUI elements are time info bars, which indicates the time left till the mandatory breaks.

Then, there are also labels for Chips Pile and Players Wallet, which display the amount stored in these two entities and other quite standard soft buttons.

5.8.2 In Game Menu

Game menu is not as complicated as In Game GUI. It only shows the icons of games able to be selected for play and content of Chips Pile, Player's Wallet, time bars indicating the time left till the mandatory breaks. All shown in Appendix A in picture A.3.

5.8.3 In Break

Other screens are describing the placement of GUI elements during mandatory breaks. They are described in Appendix A in pictures A.5, A.4, A.6, A.7.

5.9 WBS

A work breakdown structure is a key project deliverable that organizes the team's work into manageable sections.[8] The first question that someone should ask is why use a Work Breakdown Structure (WBS) at all. The work breakdown structure approach allows us to visually see the work that is needed to complete a project. The bottom line is by using a work breakdown structure it reduces the number of surprises and improves the ability to better estimate future projects.[9] WBS is quite a handy tool, which allow every project task to break into separate subtasks and therefore easily locate either weak points. Later, in project planning and project management, it serves as a foundation for distributing resources or responsibilities.

When it comes to application of WBS to the thesis' topic, it is possible to define implementation works based on each requirement and define which implementation work depends on the other. It is also possible to define which prerequisites depend on which each implemented feature and therefore it is possible to estimate overall length of development. It is important to mention, that some of the requirements are based on platform characteristics of

exemplary manufacturer. As it was said above, the current technical design and requirement list does not define gaming machine from the scratch, but assume the conditions of pre-developed gaming machine with general functionality implemented and all the requirements for German market are defined as set of modifications needed to be realized. Therefore, WBS and list of requirements may differ in some points from a manufacturer to manufacturer, because when it comes to software engineering, each manufacturer may use different technologies and platforms.

The goal of creation of WBS is get the information about overall cost of project development in matter of needed resources. In our case the development time. In Appendix C is attached an exported preview of created WBS of the project. Complete WBS is available at the attached portable storage media.

The important outcome is that according to WBS, the overall development time spent on the implementation of changes at our exemplary company will take 1269 man-days. This information will be useful for the following cost-benefit analysis.

Chapter 6

Cost–benefit analysis

Cost–benefit analysis (CBA), sometimes called benefit–cost analysis (BCA), is a systematic approach to estimating the strengths and weaknesses of alternatives (for example in transactions, activities, functional business requirements); it is used to determine options that provide the best approach to achieve benefits while preserving savings.[10]

For the application of cost-benefit analysis in this thesis, it needs to be defined a perspective or point of view of the approach. In gaming business, there are mostly subjects of two types – manufacturers and operators. Manufacturers usually don't own a license for operating casinos or gaming halls etc. So, they focus and specialize in development of gaming devices. On the other side, there are operators who specialize in customers' service toward the players, have better knowledge about certification processes and other relevant advantages. To get operator's license, it is usually required by law to possess quite large amount of capital. But the legislative, which controls the gaming business sector, differs from country to country. In some countries, for example Netherlands, operating license was granted to only one subject, which is state-owned company.

For the usage of this thesis, the cost-benefit analysis is going to be made from the perspective of manufacturer of gaming machines. The manufacturer is not going to operate the machines. He is not in that tight contact with the final customers – players. Manufacturer's work is to build the machine, program it and pass the certification process. A legal authority usually tests the machine for electrical safety, checks the process of generation randomness and generally if all declared behavior really matches the real behaver of the machine. Then the machine can be sold to an operator, who operates it. So from the perspective of manufacturer, the cost-benefit analysis is mainly about estimation of market potential, meaning how many machines the manufacturer will be able to sell. So the revenues will be income from sold machines and costs would be the cost of development and certification process.

Both costs and revenues are very hard to measure. When it comes to costs,

it would certainly differ from manufacturer to manufacturer. Also, it differs if some company tries to design and manufacture the machine from the scratch or not. In this case, the development works would take so much time it's even unthinkable. Not mentioning the lack of information about the business about behavior of game mathematics, psychology of the players etc. So, the costs must be measured from the perspective of the time needed to modify existing model of a machine in order to pass the legal requirements for Germany.

6.1 Costs

To estimate the costs of necessary modifications, it is possible to use WBS which also contained estimation of development works length.

Average salary of developer	900 000 CZK/year[2]
Length of a year	360 days
Length of a year - (weekends + holidays)	232 days[11]
Real cost of a man-year	1 396 551 CZK[3]
Total development length	1269 man-days[4]
Total unit variable cost of development	4 922 842 CZK[5]
Other costs	6 525 628 CZK [6]
Total cost	11 448 470 CZK

6.2 Benefits

To approximate the benefits, there would be needed the amount of sold machines. Because that is really complicated to estimate and also it would be needed to take under consideration infinite time of potential possibility of selling another machine, it is more wise to choose a different approach. And that is payback period.

[2]Means 75 000 CZK per month (Gross cost with tax) - Data took from market research report made by Hays.[12]

[3]Accounted that developer's salary contains also paid weekends. This line counts with cost of developer salary, how much does it cost the company to get 360 days (one man-year) of developer's work.

[4]Data obtained from WBS

[5]Equals total development length divided by length of a year multiplied by real cost of a man-year (1269/360*1396551)

[6]Contain fixed costs, overhead costs, material and service costs (data from IBM Česká republika, spol. s r.o. annual fiscal report of a year 2014 show that software company of similar size has labor costs approx. 43% of total costs.[13]

6.2.1 Payback period

Payback period in capital budgeting refers to the period required to recoup the funds expended in an investment, or to reach the break-even point.[14]

In this context payback period is not exactly what is needed here. Because it calculates some revenue stream over the time. For this purpose, it is more convenient to calculate the desirable revenue in one round. So, what is more suitable is to calculate how many machines it would be needed to sell to get the invested money back. And from the reached point, all the revenue stream is counted as net profit.

The price of used slot machine	40 000 - 80 000 CZK[6]
Estimated price of a new slot machine	110 000
Ammount of machines needed to sell	104

6.2.2 Cost–Benefit Analysis Conclusion

The amount of 104 machines is needed to sell to return the investment of the development.

[6]Price contained from multiple offers at cited online shop.[15]

Chapter 7

Technical Design

TR5 kept the way of regulating the market from TR4, but added some new limitations. The most important ones were those which introduced a new logic of mandatory breaks during the playing the game. The idea of limiting values of bets and wins remained the same from TR4. TR4 just adjusts those limits to different height, so the manufacturers could implement the logic of euros and chips, as was already mentioned. Different thing is the concept of mandatory breaks during the gameplay and the strict limit of playing time for one player. TR5 clearly states that after three hours of gaming time the machine has to reset itself and go back to idle state it was before the player started playing. This is quite crucial, because it means that when gaming time hit three hour, player loses all his money and all his chips. It was quite tricky to deal with this requirement, but after all, the final solution described in technical design satisfy the legal requirement, while keeping sure, player wont loose his money unexpectedly. The way of solving this problem was in implementing a logic which keeps track of gaming time and warning player when the end of game session is coming and offers him possibilities how to get out all his chips back in time. The detailed solution of this approach is mentioned bellow.

7.1 System Components

7.1.1 Game Control

This software component will be responsible for the device operations bound to regulatory breaks and for requesting stake and winnings approval (Stake means the conversion action from euros to chips). It is in control of game flow. For every performed conversion to chips (Stake) and for every conversion to euros (win) has to send a request to another system component (Monitoring Device) for authorization. Game Control should check the limits of stakes and wins by itself as well. For every request of stake or win, Monitoring

Device must answer via positive response. This limit checking functionality is therefore doubled. It is because of automated testing in certified laboratory during certification process. technicians will be altering one system component thru furtherly specified interface and will be trying to breach some limits of stake or wins and check if machine reacts appropriately.

7.1.2 Monitoring Device

This new system component performs automatic monitoring during routine operations of the gaming machine. Monitoring Device receives messages from Game Control and evaluate them if they follow the limits defined by the gaming ordinance. All requests are stored in the monitoring device database. All successful requests are transferred to fiscal module as the money transactions to be exported for the financial authorities (mainly for taxation purposes). Sometimes MD is called watching device.

7.1.3 Fiscal Module

This new system component will be responsible for the storage of all successful requests to the MD. This component will check presence of the fiscal DB during the device boot up. Additionally will this component export fiscal data to the external data storage. Fiscal Module contains interface for reading fiscal data from database and exporting them in appropriate way thru USB interface or RS232 interface.

This database will contain all relevant information used for financial statistics (mainly for the official financial authorities). All approved stakes and winnings will be stored in the database which will never be cleared. Database will be in the form of daily created XML file with a digital signature described by the TR5.

7.1.4 VDAI Interface Module

This software module is responsible for exporting gaming data in a specific form for the use of the operator. Operators are logically interested in information how much money was inserted into the machine last month, how much were withdrawn, which games are most popular etc. So, the German association for gaming industry designed a standard for this use. It is a special hardware interface on the gaming machine and its communication protocol. It is realized with RS232 port. The general usage of this feature is that generally the staff of casino or gaming hall possess a miniprinter device, which is programmed to read data via VDAI protocol. This device gets interconnected with the machine via RS232 connector, gaming machine and miniprinter executes defined handshake and then machine sends all the statistical data to the miniprinter device.

This module is not specifically required by TR5. It is just that majority of the market is used to this feature. Therefore, the gaming machine should have implemented this functionality to be competitive on the market.

7.1.5 PTB interface

Gaming machine must contain a device to allow connection of external testing device during acceptance testing made by the PTB institute. PTB device is connected with EGM via connector RS-232. When PTB's monitoring device is connected, it performs handshake exchange of data and starts one of three testing scenarios. Machine is then unable of regular gameplay and communicate with PTB's testing device which is testing machine internal system components, like Monitoring Device and Game Control. PTB's testing device simulates placement of stakes and wins, sending to the machine's system components requests for the approval and waits for the authorization or rejection. All the data obtained by the testing device are then subjected for evaluation and they are crucial for further approval of gaming machine.

7.2 Hardware components

Peripheral components of the gaming machine, which are needed for passing certification approval according to TR5:[1]

- Start button
- Increase Bet Button
- Decrease Bet Button
- Back Button
- Payout Button
- Stake Buton
- SW-ID Button
- Acceptor for money
- Dispensor of coins
- LCD screens

[1]What should be probably stated is, that TR5 does not require the machine to have Start Button, PC board and other hardware components or interfaces. But the architectural design of functioning machine which would comply with TR5, will most probably need to have these components in order to work properly and guarantee declared functionality.

- USB stick with bootable software
- Interface for fiscal module (USB)
- Interface for PTB's testing Device (RS232)
- Interface for VDAI export data (USB)
- Mass storage media (SSD disc)
- Keysystem
- PC board

From the list above, the only thing which has not been mentioned is SW-ID button, which is also one of the requirements of TR5. It is an isolated hardware button, which must be placed somewhere on the machine and must be available always. It is designed to be used by certification authority, to verify machine's software anytime, when ma chine will be deployed into use. When SW-ID button is pressed, machine must immediately calculate hash of Gaming Control and Monitoring Device module and display it somewhere on the screen. The government official then compare calculated hash with the hash calculated during the certification process and approves, that machine's controlling software has not been changed or modified in any way.

Other hardware components are responsible mailnly for fulfilling of the general functionality of gaming machine. Buttons, screens, acceptors, dispensers are general components which are characteristic for this type of device. USB stick is used for storing encrypted machine's software, because of easy upgreading of offline machines. And mass storage devices are needed because of database which will keep records of all booking data (stakes and winnings) as well as other gaming events.

Chapter 8

UML

This chapter describes technical concept of machine's logic via UML. UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. OMG is continuously putting effort to make a truly industry standard. UML (Unified Modeling Language) is different from the other common programming languages like C++, Java, COBOL etc. UML is not a programming language but rather a pictorial language used to make software blue prints.

So UML can be described as a general purpose visual modeling language to visualize, specify, construct and document software system. Although UML is generally used to model software systems but it is not limited within this boundary. It is also used to model non-software systems as well like process flow in a manufacturing unit etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design. After some standardization, UML is become an OMG (Object Management Group) standard.[16]

8.1 UML Design

For the purpose of the thesis, the creation of UML diagrams was done via software tool called Enterprise Architect, made by company Sparx System. In Appendix B there are several main diagrams, which show major overview of the system.

8.1.1 Use Cases Diagram

Figure B.1 describes the Primary Use Cases Diagram. This diagram is useful for highest overview of the system logic. It shows multiple actors and their

usages of the system. In the case of the machine's design, the main actors are player, attendant and system itself. Player uses the machine in perspective of gaming use cases, attendant is responsible for the maintenance of the machine and system itself is responsible for watching the gaming limits, incrementing the stake level, performing automatic conversions of chips etc. All mentioned in diagram in Figure B.1.

8.1.2 Activity Diagrams

Every use case can be described as set of activities, which follow another. For this reason, UML defines activity diagrams. Activity diagrams are intended to model both computational and organizational processes (i.e. workflows) [17]. In the Appendix B, there are set of activity diagrams, which describes detailed workflow of already mentioned use cases. Every use case has its own activity diagram, which gives a more detailed look into the logic of actions.

8.1.3 State Machine Diagram

State machine diagram is very useful tool for understanding machine's behavior. It is especially handy in the phase of this design, because behavior of gaming machine can be very easily described via separate states and states' transitions. The State Machine Diagram of designed gaming machine is shown in Appendix B in Figure B.2.

For more detailed look into the logic of each state, there is multiple activity diagram, describing the internal actions of states and evaluations of conditions in time that needs to be performed. The logic of each state can be described as infinite loop which is checking the development of conditions and values of some variables in time and when some conditions are met, the loop exits and transition to another state is going to be performed.

8.1.4 Class Model

For demonstrational purposes, portable storage media contains also picture with class model of the prototype application. It could not be included in Appendix section, because of its size. The class model is good tool for getting an idea of dependency each class and and nicely demonstrates the usage of the software in reality. From the architectural point of view, it is easy to see, that logical part of the software is separated from user interface layer and therefore the main approach of MVC is satisfied. The main class representing all the functionality of the machine has multiple "getters" and "setters", which are also defined by its interface class, so the class is easily portable to different environment and may serve as foundation for usage in practice.

Chapter 9

Prototype

9.1 Overview

The functional design of gaming machine under new legislative is quite complicated and hard to understand from the technical point of view. For this reason, it is much more user friendly to demonstrate the functionality in practice. This was one of the reasons for creating the working prototype of the machine.

The portable storage medium which is attached to the thesis contains source code written in programming language Java and precompiled and executable JAR file. In order to run the prototype, it is necessary to have installed Java SE, which is freely available on the website of Oracle company. To run the full functionality of the prototype, it is required to have set up database server and connect it properly to the application. The manual can be found in attached storage media.

However, the prototype is useful for demonstration of the functionality, its main usage is rather more extended. The main usage may be in real application of logic layer in real gaming machine. The way it is designed, allows easy portability to different platforms and systems. That was also the reason of choosing programming language Java, because it may be run under many different operating systems. All information's about the machine state and all information regarding timers, limits in time etc. are accessible thru one interface of the main object. When the object gets initialized, it restored last saved state from the database, performs all necessary calculations of time passed and expired limits and continue its logic. Like if the gaming machine would lose power or was switched off. This was also one of the required functionality. If the object logic fails to connect to database, it gets initialized like it would be initialized for the first time. So, the demonstration of the machine may be performed either without connection to the database server.

So, as it has been said, the prototype is designed as separate object, that can be called thru defined interface to receive all kinds of information which are

relevant for gaming logic. The third-party platform may any time call specific method to receive information it needs for displaying mandatory information, transiting to mandatory breaks etc. And the prototype object keeps track of time, calculates the limits and all other system logic is taken care of internally in the prototype object.

To be more specific about the interface of the prototype model, its design is based on pattern of state machine, which accepts some calls/triggers from the outside, responds properly and change its internal state if it is appropriate.

The main triggers are for example:

- pressedStakeButton() - should be called when player presses STAKE button in order to transfer chips from his Player's Wallet to Chips Pile
- acceptsBanknote() - should be called when player inputs money into the machine
- insertPlayersKey() - should be called, when player inserts player's key into the machine
- removePlayersKey() - should be called, when player removes player's key from the machine
- insertAdminsKey() - should be called when attendant inserts admin key to the machine
- removesAdminsKey() - should be called when attendant removes admin key from the machine
- ...and others

So the main system (platform) has to call these methods to inform the prototype object about the events from the outside world. Then it can take these events under consideration, perform some internal logic, evaluation of limits and other adjustments. Apart from that, prototype model's interface contains method, which provides information to main system. In IT world, so called "getters".

Those are for example:

- getPlayersWallet() - returns an amount of money present in Player's Wallet
- getChipsPile() - returns an amount of chips in Chips Pile
- getGamingTimeLeft() - returns an amount of time left till mandatory Idle Break in milliseconds
- getTimeToMandatoryBreak() - returns an amount of time left till mandatory Interruption Break in milliseconds

- isConversionFromChipsPileToPlayersWalletEnabled() - returns information if automatic conversion from Chips Pile to Player's Wallet is in progress
- isAcceptorEnabled() - returns information if money acceptor should be turned on or off
- ...and others

So, that is the basic idea of the functionality. The prototype included in this thesis also contains UI interface, which allows to control the main logic object and show the functionality for the presentational purposes. UI also contain separate module for simulation of spinning reels and generation of random winnings. The main prototype logic object does not contain the mathematics for generation game outcome, because it is expected that every gaming machine manufacturer already have its own game mathematics developed, because this is one of the most important know-hows in this business. So, the outside system must either supply logical prototype object with events from outside world, like insertion of keys or pressed buttons, but also with the wins and losses. That may be done via methods winChips(int) and looseChips(int) (winChips(int) must be called while winning is recieved, looseChips(int) when bet is placed).

The prototype model does not implement all of the functionality needed for passing the certification process and for being really competitive on the market. Some functionalities which use external peripherals have also been neglected. To be more precise, it has not been implemented the requirements about VDAI export, fiscal data export and automated unit test procedure during certification process. The reason for not implementing VDAI export is that it contains mainly data about game performance and behavior of the machine (door open, specific game combinations won etc.). The fiscal data export and automated unit test could have been implemented, but there was not enough time and resources. The prototype model has already a size which is beyond the sizing for this kind of thesis (over 6000 lines of code).

9.2 Architecture

Architecture of prototype model is based on state machine pattern. The defined states from the analysis, including requirements from TR5, are described in Appendix B in picture B.2. Based on this logic was also implemented the structure of prototype's states.

Main object is composed mainly of two logical unit. In same way as TR5 describes the main logic. Its a component performing logic of gaming control and component performing logic of monitoring device. This main object has one thread running in infinite loop which is checking if game control

component has some message needed for evaluation. If it does, it takes the message, send it to monitoring device component, waits for the response and delivers the response to game control component. Also from the requirements of TR5, Gaming Control and Monitoring device have to be two separated components, which have to be hashed independently in order to verification check.

9.2.1 Game Control

Game Control component is the main object responsible for gaming logic. It controls the limits, handles events and runs the main state machine. While initialized, it reserializes the last saved state from database and continues the main game flow. Events passed to the main upper object are just resend to game control component which takes care of particular event and response properly according to each state logic.

Each state is implemented as separate class, which has accessible methods, like "setters" mentioned in above section. When some event happens in outside world (insertion of key, button pressed etc.), this event is passed to particular state class, that machine is currently in and state performs its particular logic, check for limits and as return value it either return itself or some other state.

9.2.2 Monitoring Device

Monitoring device, as the name suggest, is responsible for controlling the transaction events. The usage of this component is directly specified in TR5 (as well as functionality of Game Control). Monitoring Device has to check request for every transaction between Player's Wallet and Chips Pile. Also, there must never be a situation that game control would refuse any kind of transaction. Because that would mean a malfunction of machine's logic. So, Game Control must evaluate on its own what message can be send for verification in which time. That means that control logic of limits in time in Game Control and Monitoring Device is actually the same. Even from implementation point-of-view. The one could ask, why the requirements are put this way, because it does not make much sense, but actually there is reason for this. When machine will be tested during certification process. Certification authority also define process of automated testing, when it is going to replace one of the components of the machine (Game Control or Monitoring Device) is going to be externally replaced by component of the certification authority. The certification authority will then run some unit tests and check the results if some limits were not breached.

Conclusion

The goal of this thesis was to perform business analysis, technical design, and cost-benefit analysis of a standalone gaming machine for the German market, with respect to the legal requirements specified in technical guideline TR5. The business analysis should contain a list of requirements for implementation, the technical design should contain appropriate UML diagrams and a project plan was meant to be described by WBS. The final part should be creation of a simulator of the gaming machine in Java SE which would demonstrate the main functionalities.

Overall, the main conclusive fact about this project, was that it got much more complex and more extended, than it was planned. The German legislative goes quite far and precisely defines requirements which goes deeply into the detail. So, first part of the research and gathering of requirements was sometimes frustrating, because with every new information, or with every newly interpreted information, it has meant a destruction of previously designed logic. But at the end, it was managed to cover the problematics and come with a possible solution.

From the above-mentioned research, analysis and work, it is obvious that conclusion will have multiple layers. In the thesis, there has been covered topics from requirements overview, analysis of the problematics, project planning, cost-and-benefits analysis and implementation of working prototype.

The part of the thesis defining the requirements has been fully covered. Each requirement is linked to section paragraph in TR5, and it has been taken care of, that every requirement from TR5 is properly taken under consideration in the design. During the process of gathering the requirements, there has been kept quite close touch with the partner side in Germany, for clarification of any misunderstandings. It is obvious, that development may raise some unrevealed issues, which has not been covered, but that's a general risk of every project. But in the current situation, with the information and resources available, it has been made the highest effort with summarizing the requirements for further analysis.

After the finish of the process of gathering the requirements, next step was to come up with technical design of a machine that would match the requirements. This was the trickiest part of all. The thing is that requirements were not specifying how the final product should look like and how it should behave, but instead they had more of forbidding character. It means that they were only specifying the scenarios which must not occur. It makes sense, because that is the way how legal documents are written. They just specify forbidden behavior, end everything which is not forbidden, must be allowed. So, the approach to technical specification had to take under consideration the requirements in their form, interpret them properly and come up with the product that either satisfy the legal requirements and would be competitive on the market. At this point, obviously, it cannot be known, if the product will be successful on the market. In the future, the success of the gaming machine depends on the marketing strategy, sales, logistics and many others. So anytime it is not easy to distinguish, wherever the success or failure is caused by the way machine is designed. Anyway, at this point in time, it has been done the most of all possible works and the technical design covers all known legal and other requirements.

Another part of the thesis was to plan a foundation for project development. In this process, it was defined how every requirement is connected to another and what is its place in project structure. As a tool for this, it was used WBS notation and development works were defined in connection with environment of existing development company (gaming machine manufacturer). The approach has expected the that the company already possess a functioning physical model of gaming machine so development works were specified as set of changes, which are needed to be done to match the requirements list. This assumption is generally applicable for every company which is involved in this business. So, from practical point of view, the WBS can serve as foundation for project planning for every company, which would be interested in this topic.

The last but one part was the cost-and-benefit analysis. In this chapter, it was summarized the pros and cons of the project. Both costs and revenues are hard to estimate, because they as well depend on many variables. But still it was outlined the main costs in form of development resources needed and what should be the income in order to make the project profitable.

Finally, the functioning prototype model. For demonstrational purposes, but also for main possible practical usage, it was created the software, which applies the logic of the requirements. The prototype is designed so its internal logical layer is portable to different environment, so it would work on specific platforms of different gaming machine manufacturers.

The thesis covers all the topics and problematics defined in thesis task.

Bibliography

[1] Casino Reviews: *The History of Slot Machines [online]*. [cit. 2016-10-27]. Dostupné z: http://www.casinoreviews.co.uk/slots/history/

[2] Best Pokies: *Pokies History [online]*. [cit. 2016-10-27]. Dostupné z: http://bestpokies.me/pokies-history/

[3] Bellis, M.: *The History of Slot Machines – Liberty Bell [online]*. About Money, [cit. 2016-10-26]. Dostupné z: http://inventors.about.com/od/sstartinventions/a/Slot_Machines.htm

[4] Info Slotmachine: *Origins of the slot machine [online]*. [cit. 2016-10-27]. Dostupné z: http://www.infoslotmachine.com/history.php

[5] Casino Reviews: *The History of Slot Machines [online]*. [cit. 2016-10-27]. Dostupné z: http://www.casinoreviews.co.uk/slots/history/

[6] Gaming, S.: *Mega Slots HD 2 [online]*. Best Windows 8 Apps, [cit. 2016-11-01]. Dostupné z: http://bestwindows8apps.net/app/mega-slots-hd-2/

[7] Play Slots: *Reel Slot Machine [online]*. [cit. 2016-10-31]. Dostupné z: http://strategy.minnim.org/reel-slot-machine.html

[8] workbreakdownstructure.com: *Work Breakdown Structure (WBS)*. [cit. 2016-19-04]. Dostupné z: http://www.workbreakdownstructure.com/

[9] workbreakdownstructure.com: *PMBOK - Work Breakdown Structure*. [cit. 2016-19-04]. Dostupné z: http://www.workbreakdownstructure.com/work-breakdown-structure-according-to-pmbok.php

[10] David, R.: *A cost-benefit analysis of document management strategies used at a financial institution in Zimbabwe: A case study*. SA Journal of Information Management, [cit. 2016-12-04].

[11] Calendar.sk: *Fond pracovní doby - Česko 2016*. Calendar.sk, [cit. 2016-18-04]. Dostupné z: http://calendar.zoznam.sk/worktime-czcz.php

[12] Corp., H.: *PLATOVÝ PRŮZKUM HAYS, PRACOVNÍ TRH V ROCE 2016*. Hays, [cit. 2016-18-04]. Dostupné z: https://www.hays.cz/cs/groups/hays_common/@cz/@content/documents/digitalasset/hays_1602080.pdf

[13] PWC: *účetní závěrka [2014], zpráva auditora, C 692/SL76/MSPH*. PWC CZ, [cit. 2016-18-04]. Dostupné z: https://or.justice.cz/ias/content/download?id=f8c2db591f5b4f88be45e3ccab5d8aac

[14] Farris, P. W.: *Marketing Metrics: The Definitive Guide to Measuring Marketing Performance*. Upper Saddle River, New Jersey: Pearson Education, Inc., první vydání, ISBN ISBN 0-13-705829-2.

[15] Slot Machines ltd.: *Product-category*. [cit. 2016-18-04]. Dostupné z: http://slotmachinesltd.com/product-category/igt/igt-video-slots/igt-044-video-slot-machines

[16] TutorialsPoint: *UML - Overview [online]*. TutorialsPoint, [cit. 2016-12-01]. Dostupné z: https://www.tutorialspoint.com/uml/uml_overview.htm

[17] Force, U. R. T.: *OMG Unified Modeling Language Specification*. Object Management Group, [cit. 2016-12-04]. Dostupné z: http://www.digilife.be/quickreferences/Books/OMG%20UML%20Specification%201.4.pdf

Appendix A

The List of Used Abbreviations

GC Game Control

GUI Graphical user interface

LCD Liquid crystal display

MD Monitoring Device

MVC Model-view-controller

OS Operating sytem

PC Personal computer

PTB Physikalisch-Technische Bundesanstalt (German certification authority)

PW Player's Wallet

RS232 Recommended Standard 232

RFID Radio Frequency Identification

TR4 Technische Richtlinie 4.0 (former technical legal ordinance)

TR5 Technische Richtlinie 5.0 (technical legal ordinance)

USB Universal Serial Bus

VDAI Verband der Deutschen Automatenindustrie (Association of German Gaming Industry)

WBS Work Breakdown Structure

XML Extensible markup language

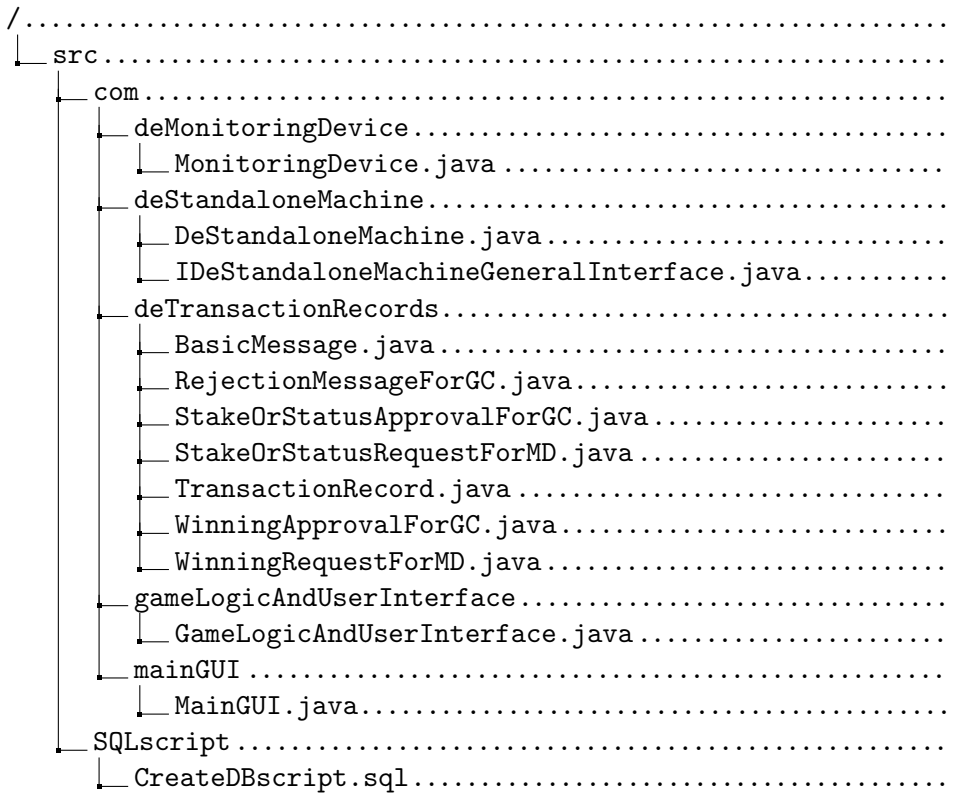
Appendix B

Content of the Attached Storage Media

The final thesis comes with attached portable storage media with source codes, precompiled prototype model, UML diagrams and WBS project including all used pictures and other artifacts.

The media contains following directory tree:

```
/
├── Final Thesis - novakm44.pdf ..... Final thesis in PDF form
├── app ..... Folder with prototype application
│   ├── DESTANDALONEMACHINE.jar ..... Executable JAR file
│   ├── README.TXT ..... Set up instructions
│   └── images
│       ├── Banana.png
│       ├── Bar.png
│       ├── Bean.gif
│       ├── Bell.png
│       ├── Cherry.png
│       ├── Clover.png
│       ├── Diamond.png
│       ├── Plum.png
│       ├── Seven.png
│       └── Watermelon.png
└── src ..... Folder with prototype source codes
    └── com
        ├── databaseConnection
        │   └── DatabaseConnection.java
        ├── deCertificationTestCaseStates
        │   └── CertificationTestCaseState.java
        ├── deConstants
        │   ├── DatabaseConstants.java
        │   └── GameConstants.java
        └── deGameControl
            ├── CounterTimer.java
            ├── GameControl.java
            ├── MoneyAndChips.java
            ├── TimersMoneyAndState.java
            └── deMachineStates
                ├── ActiveState.java
                ├── AdminMenuState.java
                ├── ForcedIdleBreakState.java
                ├── ForcedInterruptionBreakState.java
                ├── IdleStateFromForcedIdleBreak.java
                ├── IdleStateFromVoluntaryIdleBreak.java
                ├── IEventsFromButtonsAndKeys.java
                ├── LongTermPayoutActiveState.java
                ├── LongTermPayoutBreakState.java
                ├── LongTermPayoutForcedInterruptionBreakState.java
                ├── State.java
                ├── VoluntaryIdleBreakState.java
                ├── VoluntaryInterruptionBreakDueToInactivityState.java
                └── VoluntaryInterruptionBreakState.java
```


/
src
com
deMonitoringDevice
MonitoringDevice.java
deStandaloneMachine
DeStandaloneMachine.java
IDeStandaloneMachineGeneralInterface.java
deTransactionRecords
BasicMessage.java
RejectionMessageForGC.java
StakeOrStatusApprovalForGC.java
StakeOrStatusRequestForMD.java
TransactionRecord.java
WinningApprovalForGC.java
WinningRequestForMD.java
gameLogicAndUserInterface
GameLogicAndUserInterface.java
mainGUI
MainGUI.java
SQLscript
CreateDBscript.sql

- /
  - uml — Folder with UML diagrams
    - activatesconversion.png
    - cashoutwallet.png
    - ClassModel.png
    - deactivatesautostart.png
    - decreasespinprice.png
    - enteringthegame.png
    - fixingjammedcoin.png
    - increasespinprice.png
    - initialise.png
    - insertsmoney.png
    - performsautostart.png
    - possiblyconversechipstomoney.png
    - possiblydisablefurtherconversion.png
    - possiblyenablefurtherconversion.png
    - possiblyincrementstake.png
    - PrimaryUseCasses.png
    - printsvdaireport.png
    - resetmachine.png
    - spin.png
    - stakeplacement.png
    - StateMachine.png
    - stopsconversion.png
    - takesvoluntarybreak.png
    - technicalrefill.png
    - uiforcedidlebreak.png
    - uigame.png
    - uigamemenu.png
    - uiinactivitybreak.png
    - uiinterruptionbreak.png
    - uivoluntarybreak.png
    - unlockingmachine.png
  - wbs — Folder with WBS artifacts
    - thesisWbsExportPart1.png
    - thesisWbsExportPart2.png
    - thesisWbsProject.mpp — Microsoft Project file containing WBS

Appendix A

Design of GUI screens

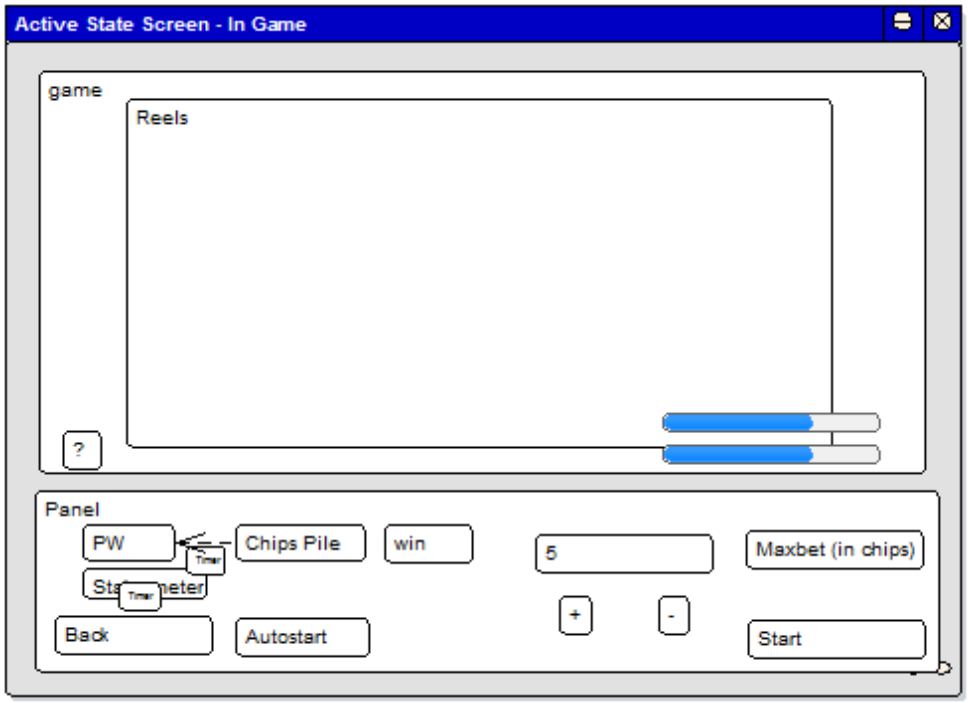


Figure A.1: GUI Screen 01 - In Game

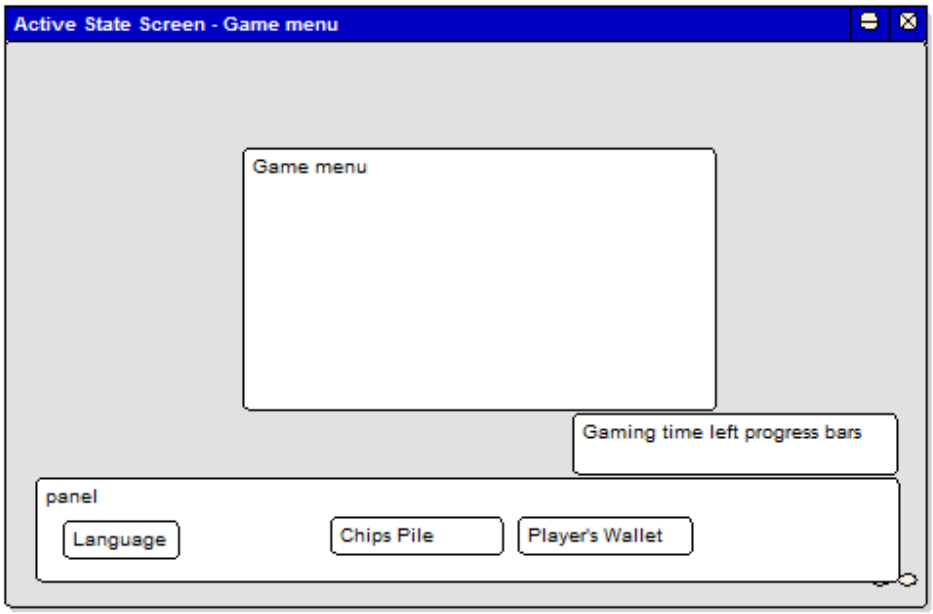


Figure A.2: GUI Screen 01 - Game Menu

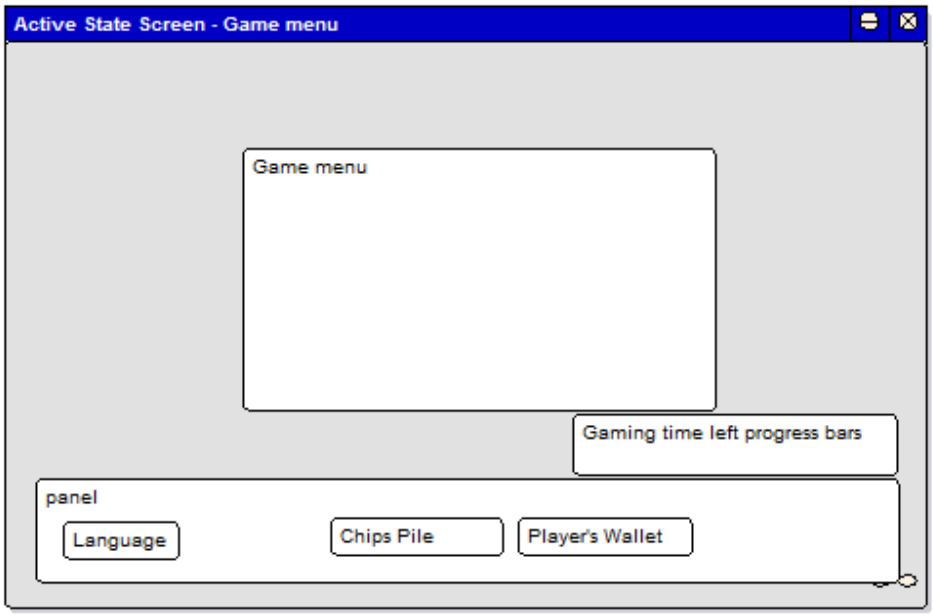


Figure A.3: GUI Screen 01 - Game Menu

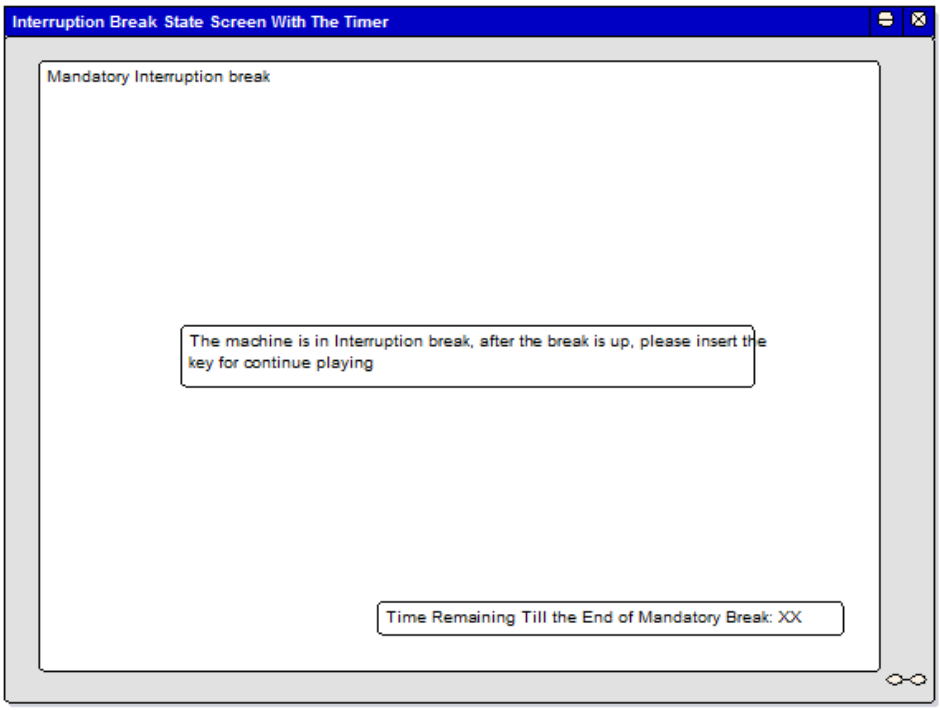


Figure A.4: GUI Screen 03 - Forced Interruption Break

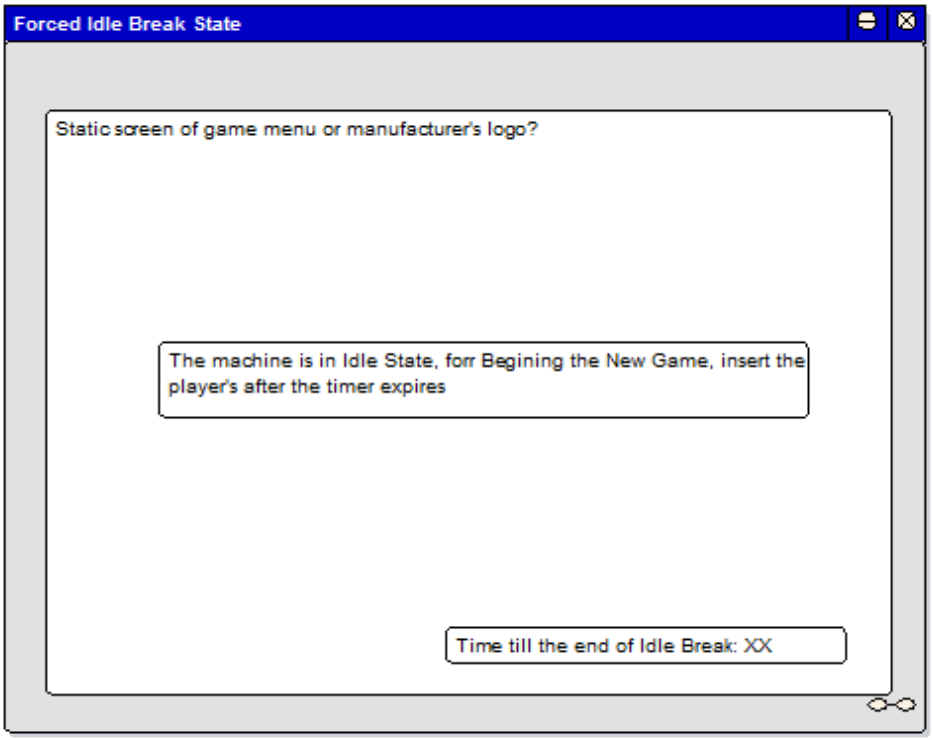


Figure A.5: GUI Screen 04 - Forced Idle Break

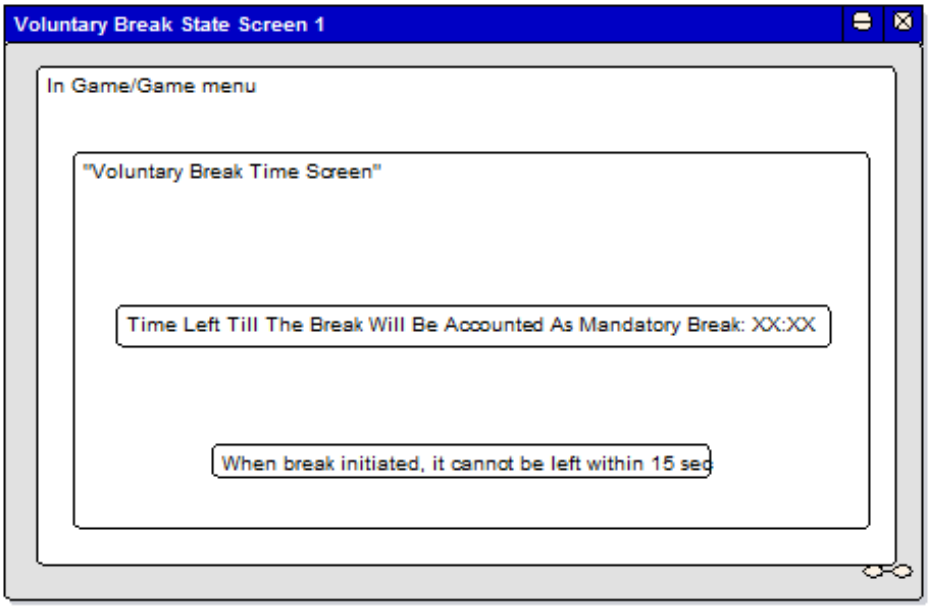


Figure A.6: GUI Screen 05 - Voluntary Break

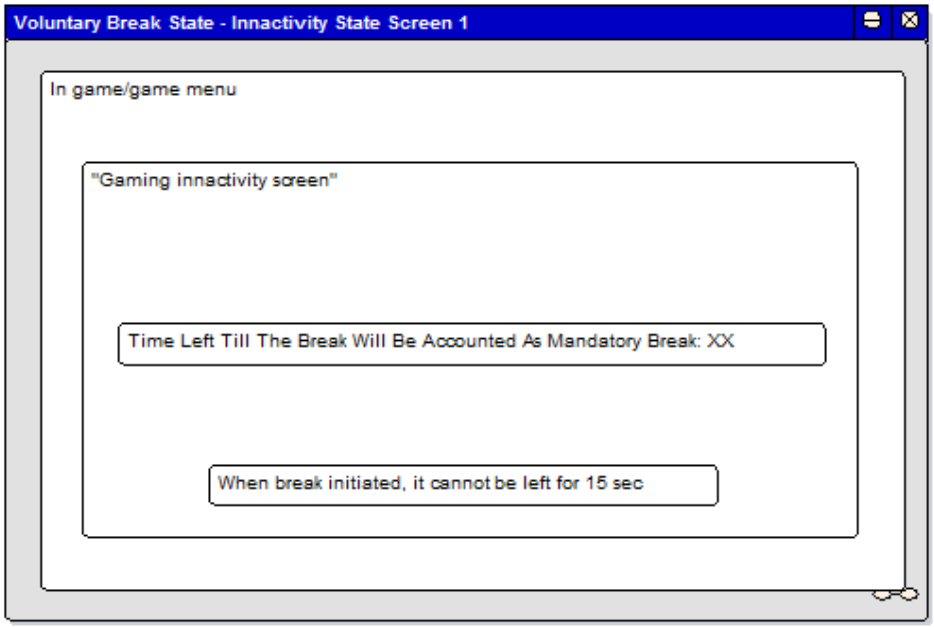


Figure A.7: GUI Screen 05 - Voluntary Break due Player's Inactivity

Appendix B

UML Diagrams

Figure B.1: Primary Use Cases Model

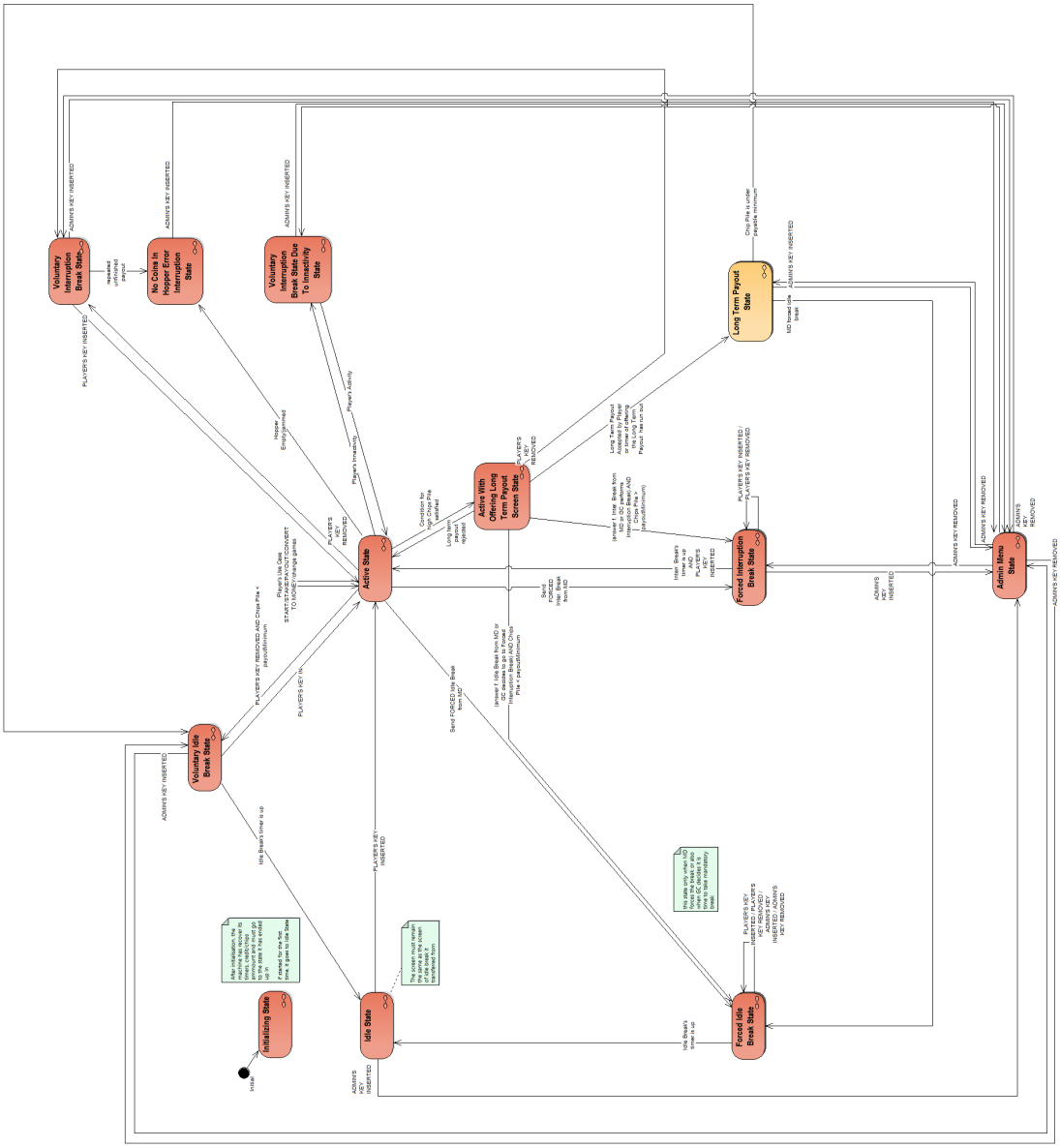


Figure B.2: State Machine Diagram

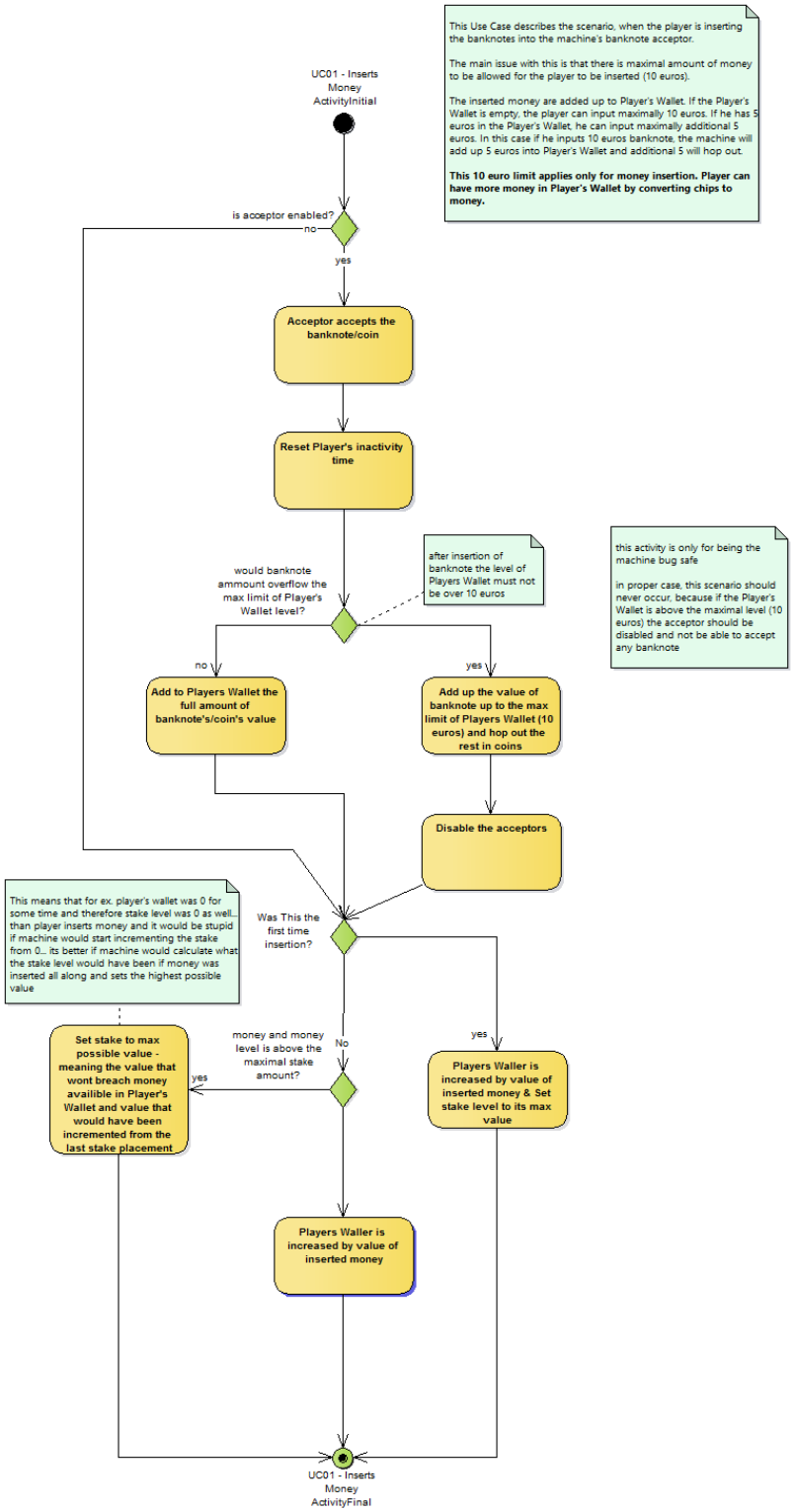


Figure B.3: Player UC01 - Inserts Money

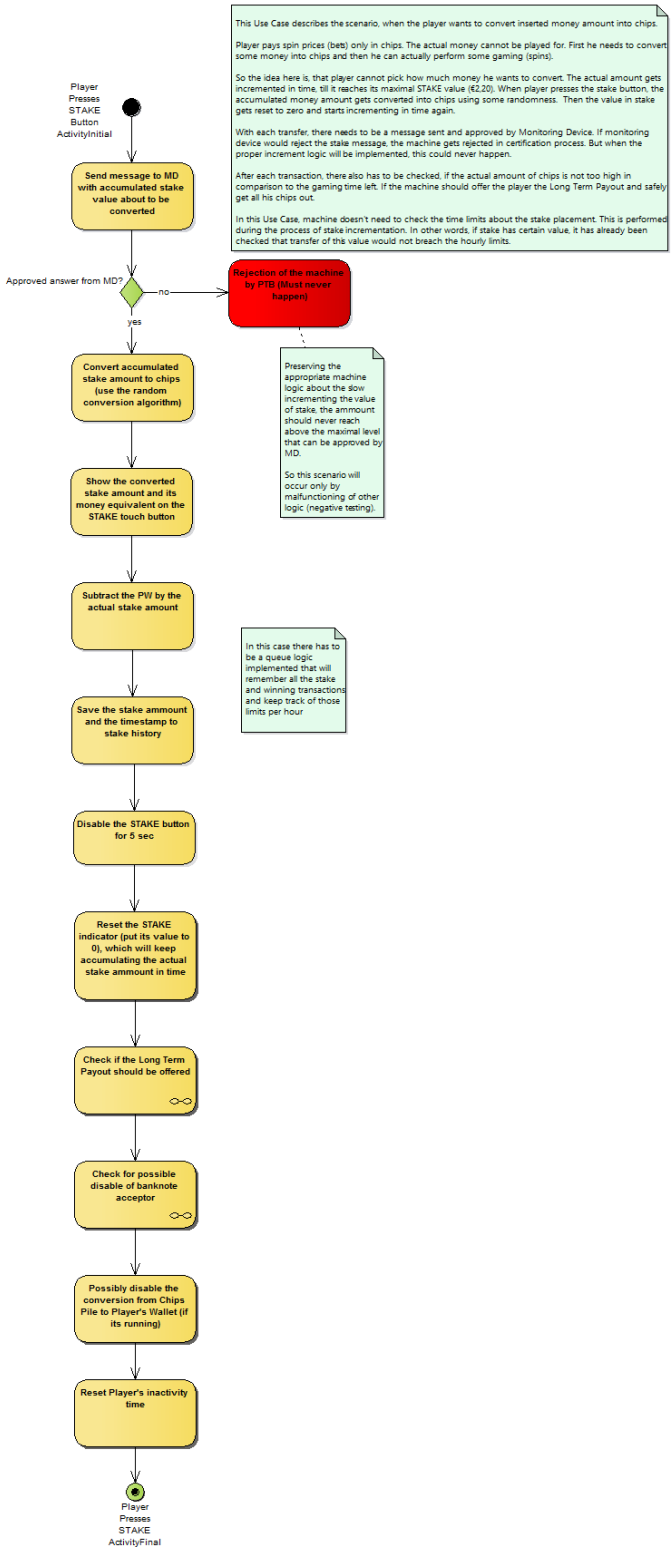


Figure B.4: Player UC02 - STAKE placement

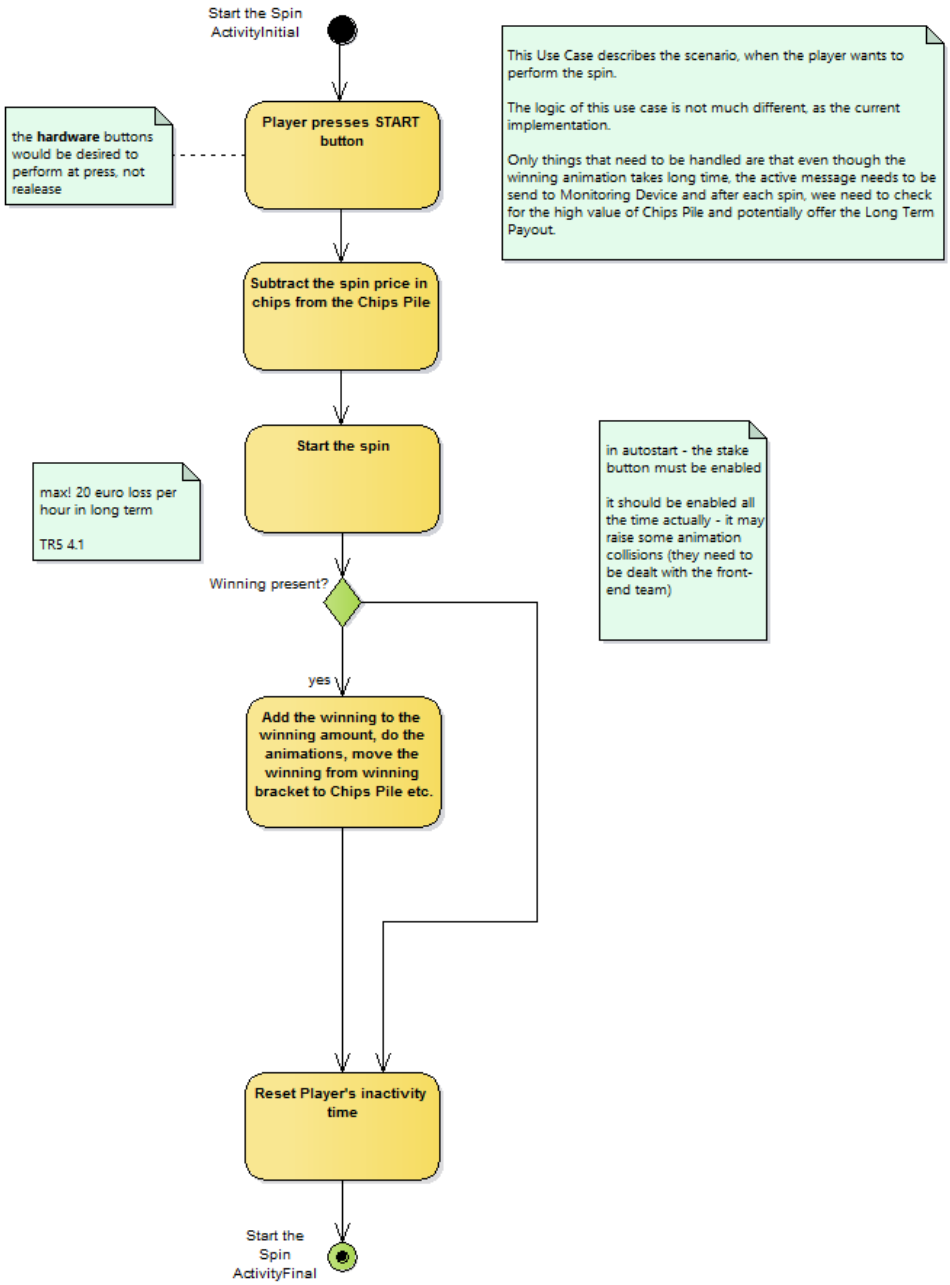


Figure B.5: Player UC03 - Spin

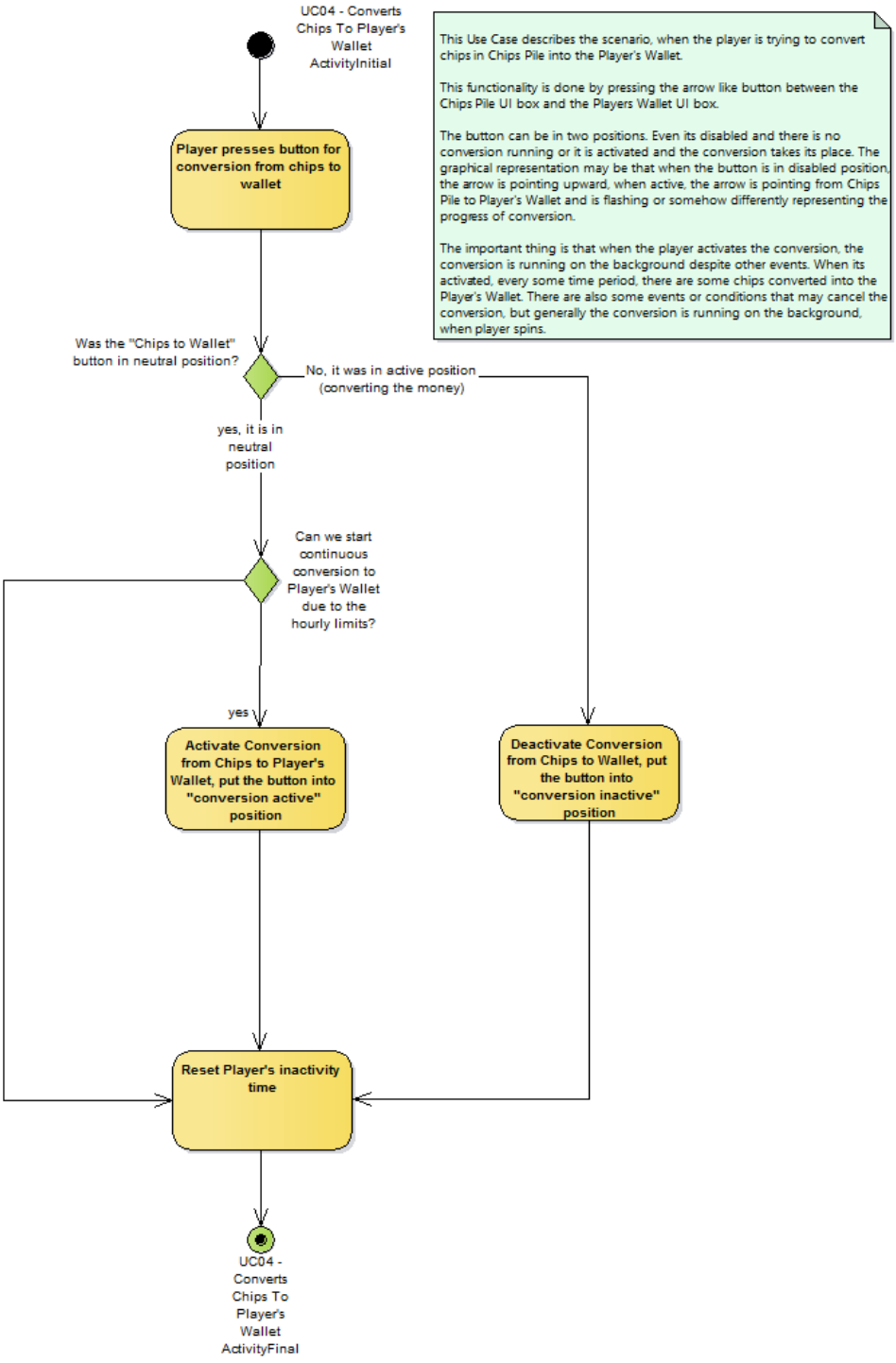


Figure B.6: Player UC04 - Activates conversion from Chips PileTo Player's Wallet

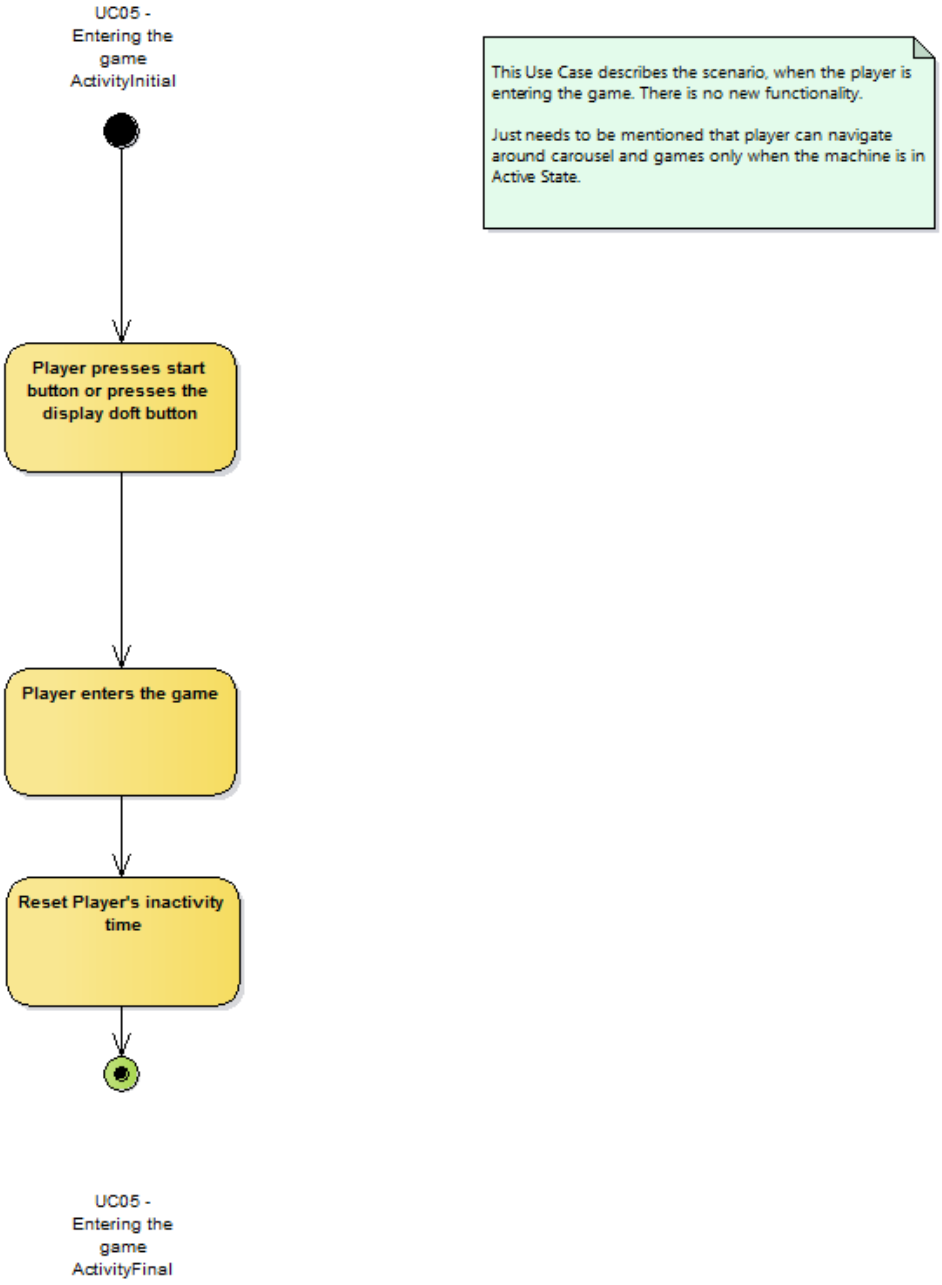


Figure B.7: Player UC05 - Entering the game

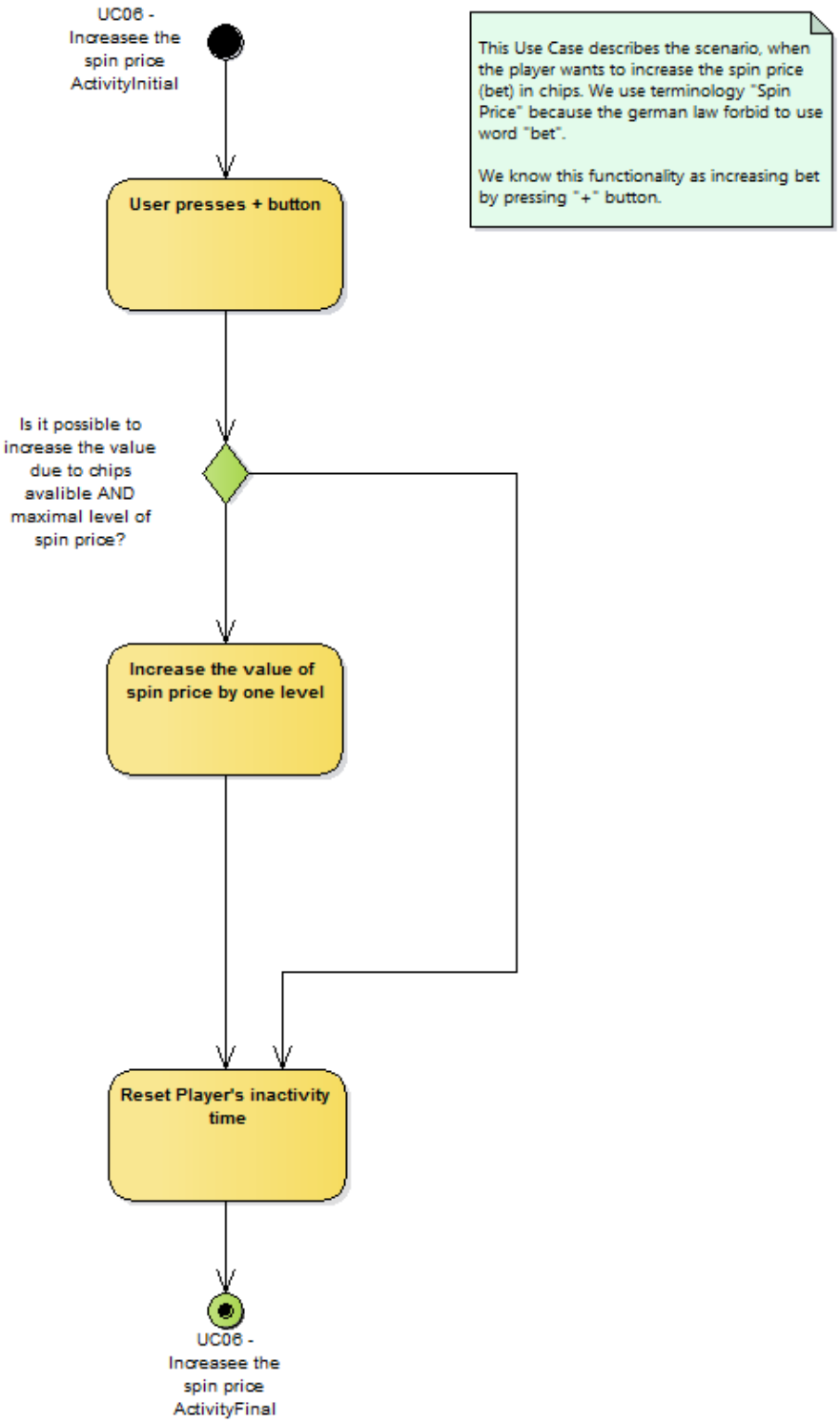


Figure B.8: Player UC06 - Increase the spin price

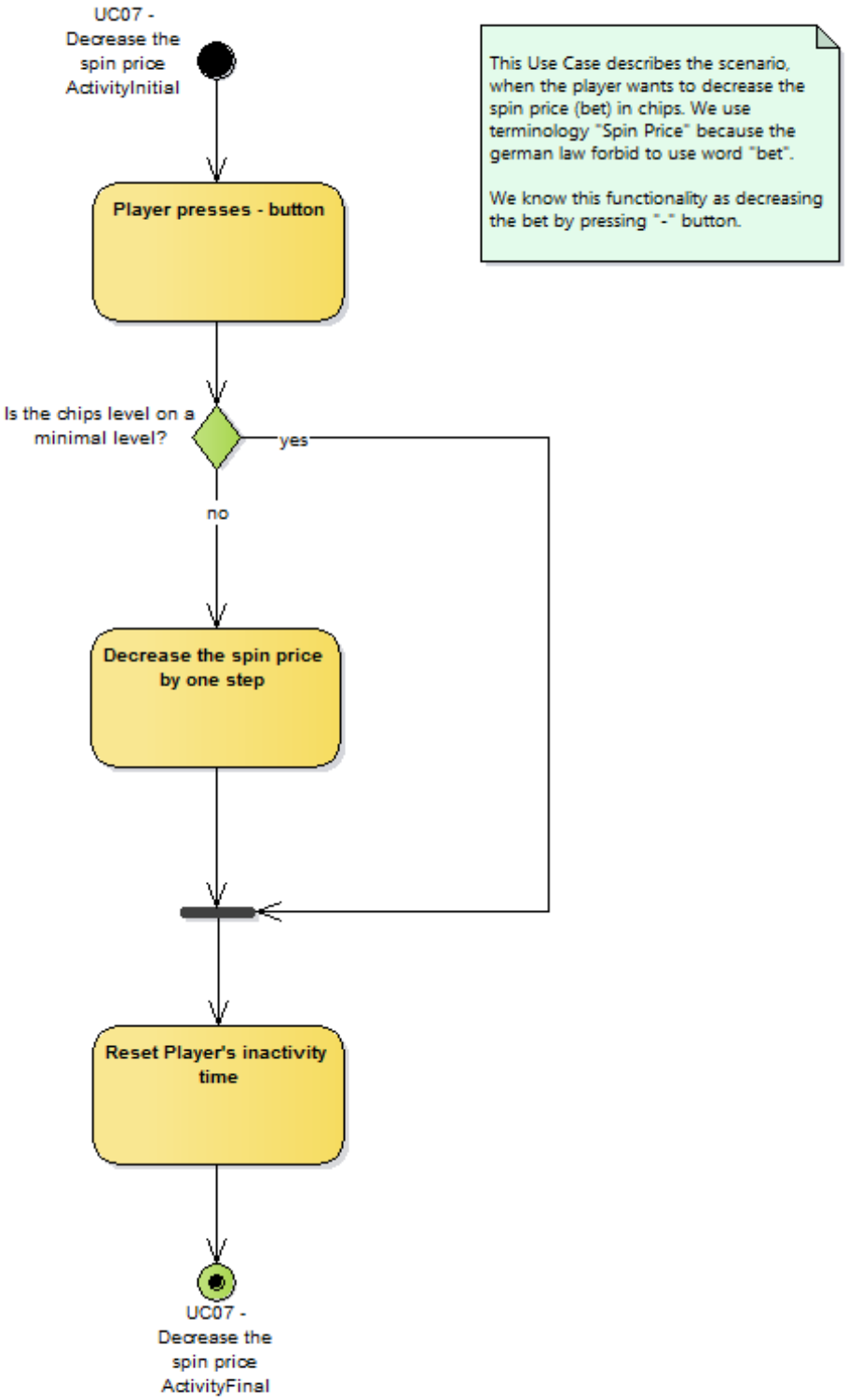


Figure B.9: Player UC07 - Decrease the spin price

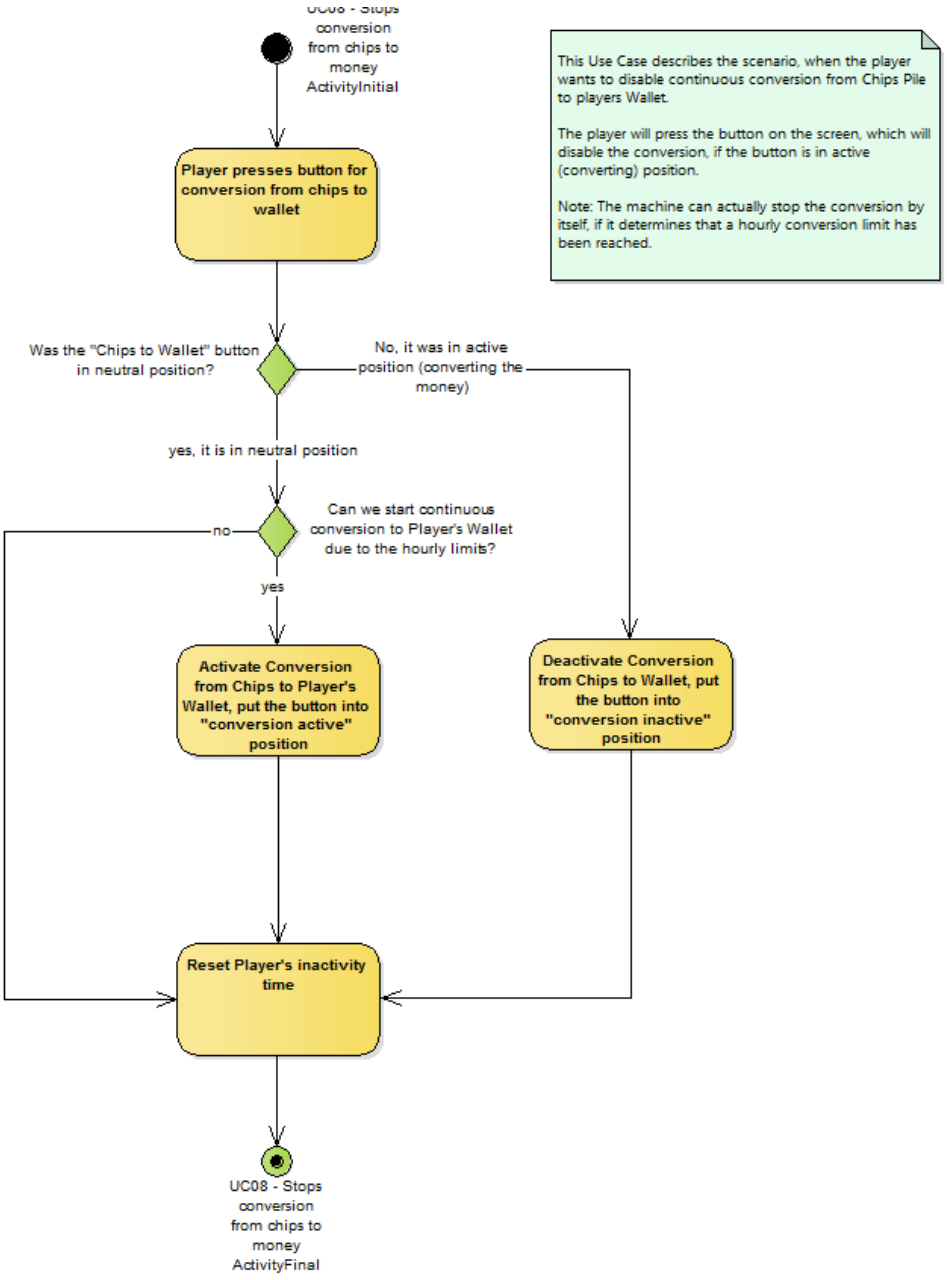


Figure B.10: Player UC08 - Stops conversion from Chips Pile to Player's Wallet

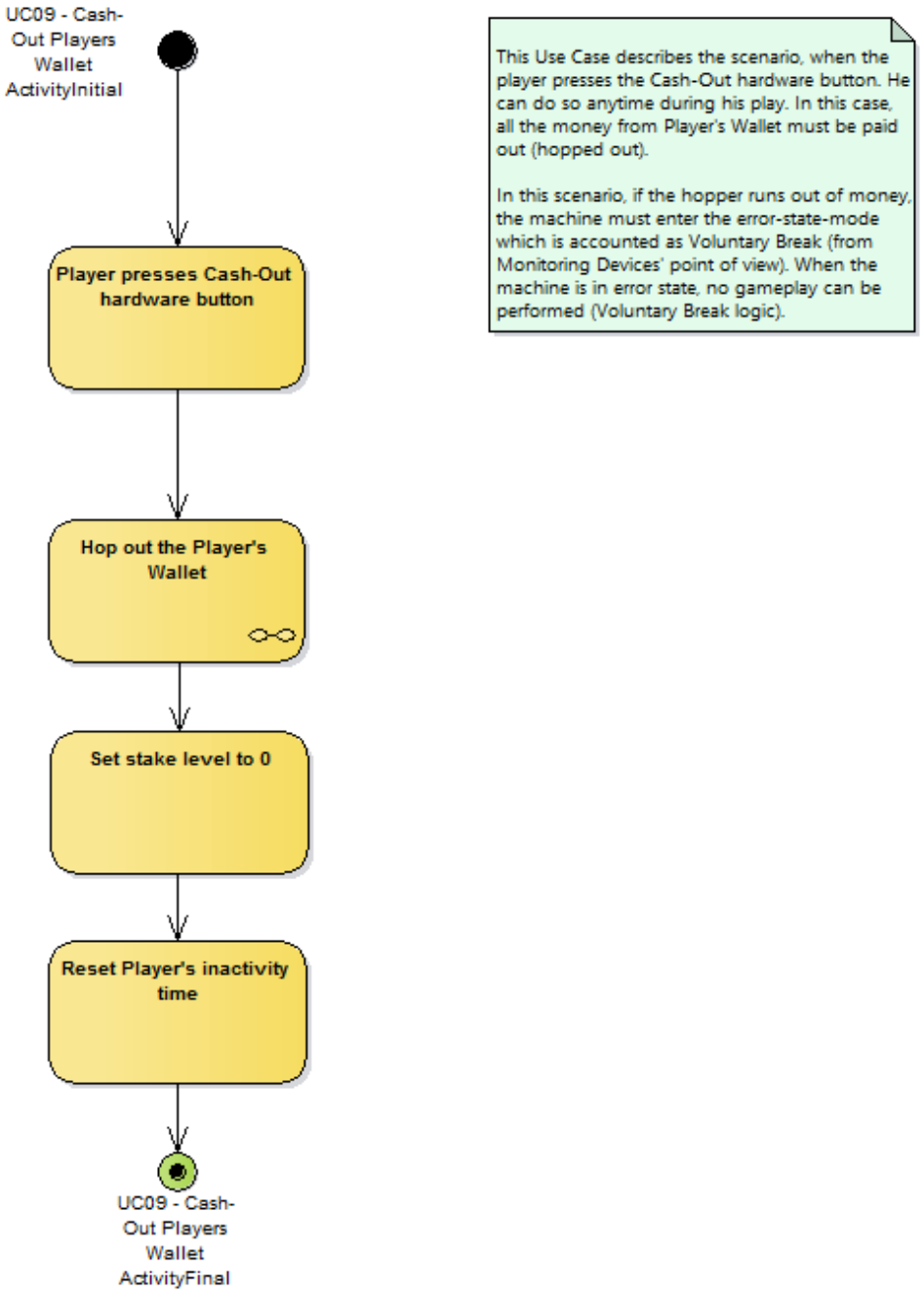


Figure B.11: Player UC09 - Cash-Out Players Wallet

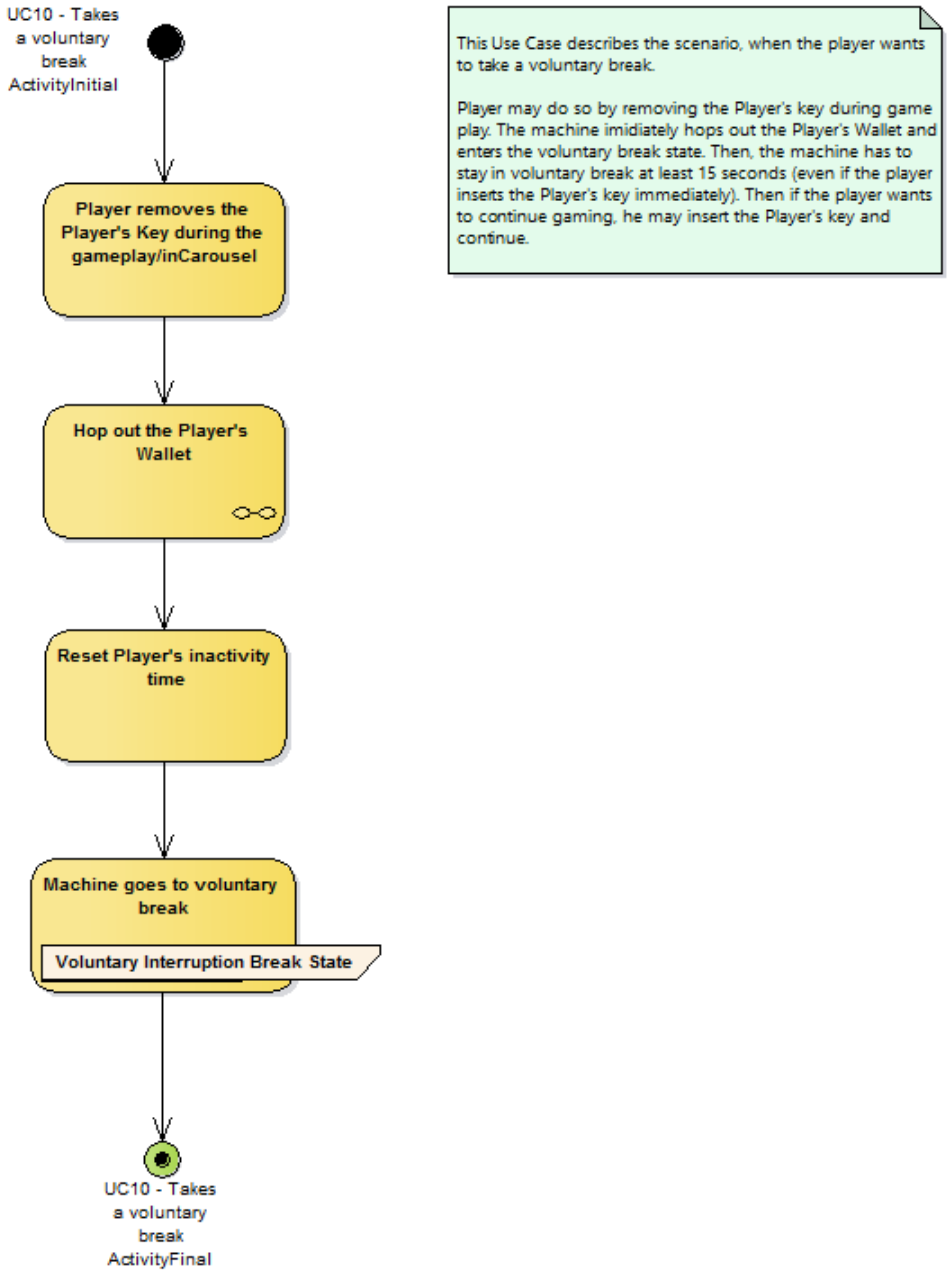


Figure B.12: Player UC10 - Takes a voluntary break

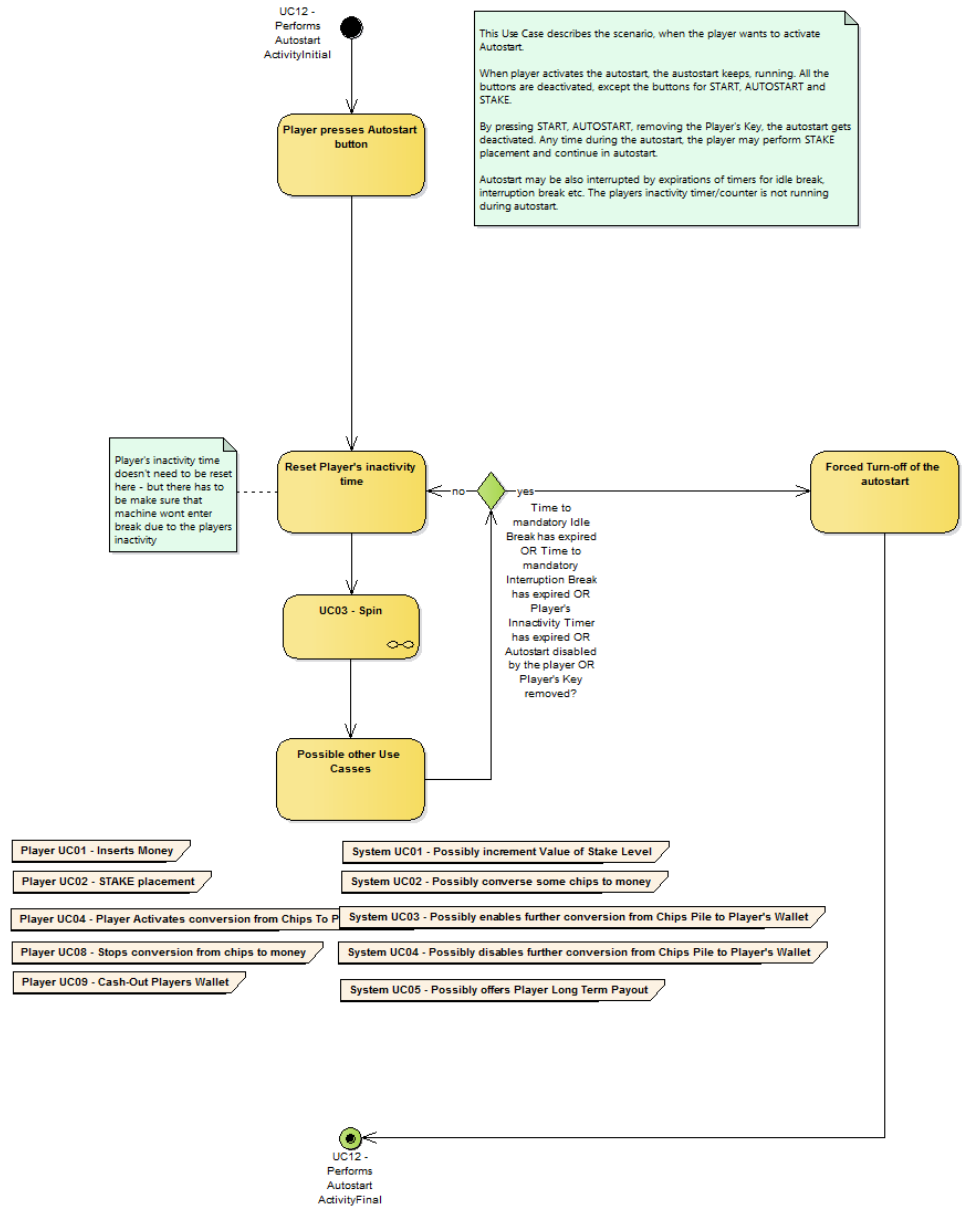


Figure B.13: Player UC12 - Performs Autostart

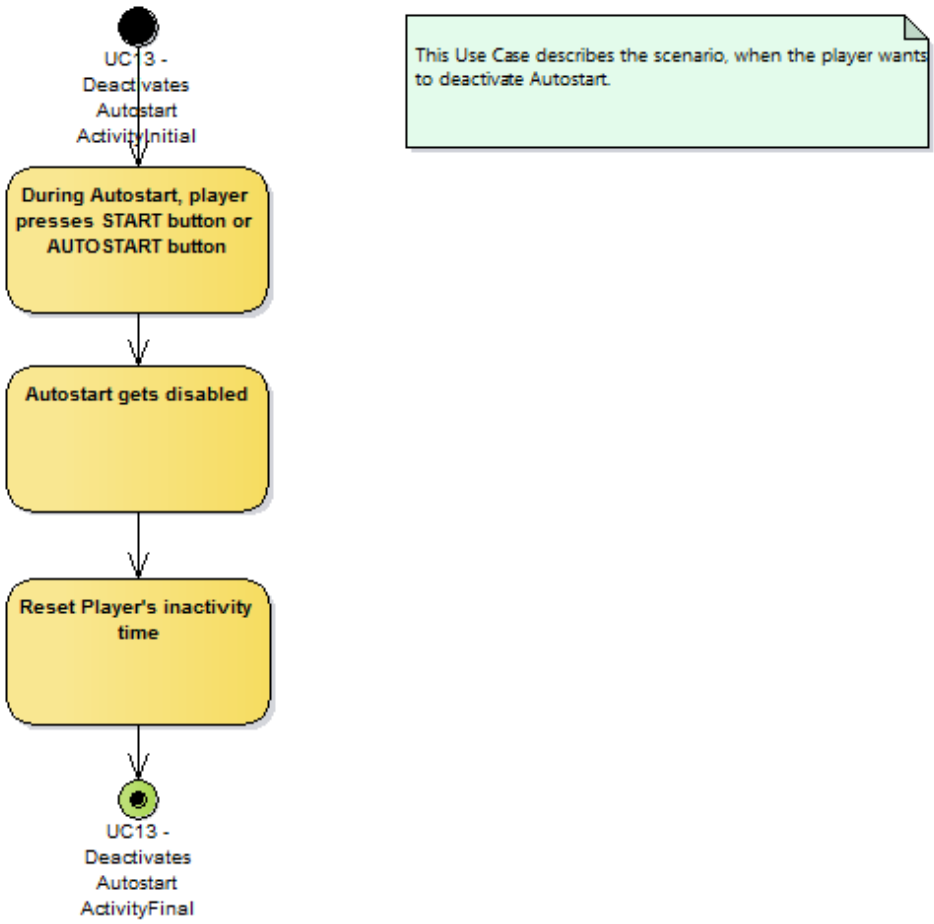


Figure B.14: Player UC13 - Deactivates Autostart

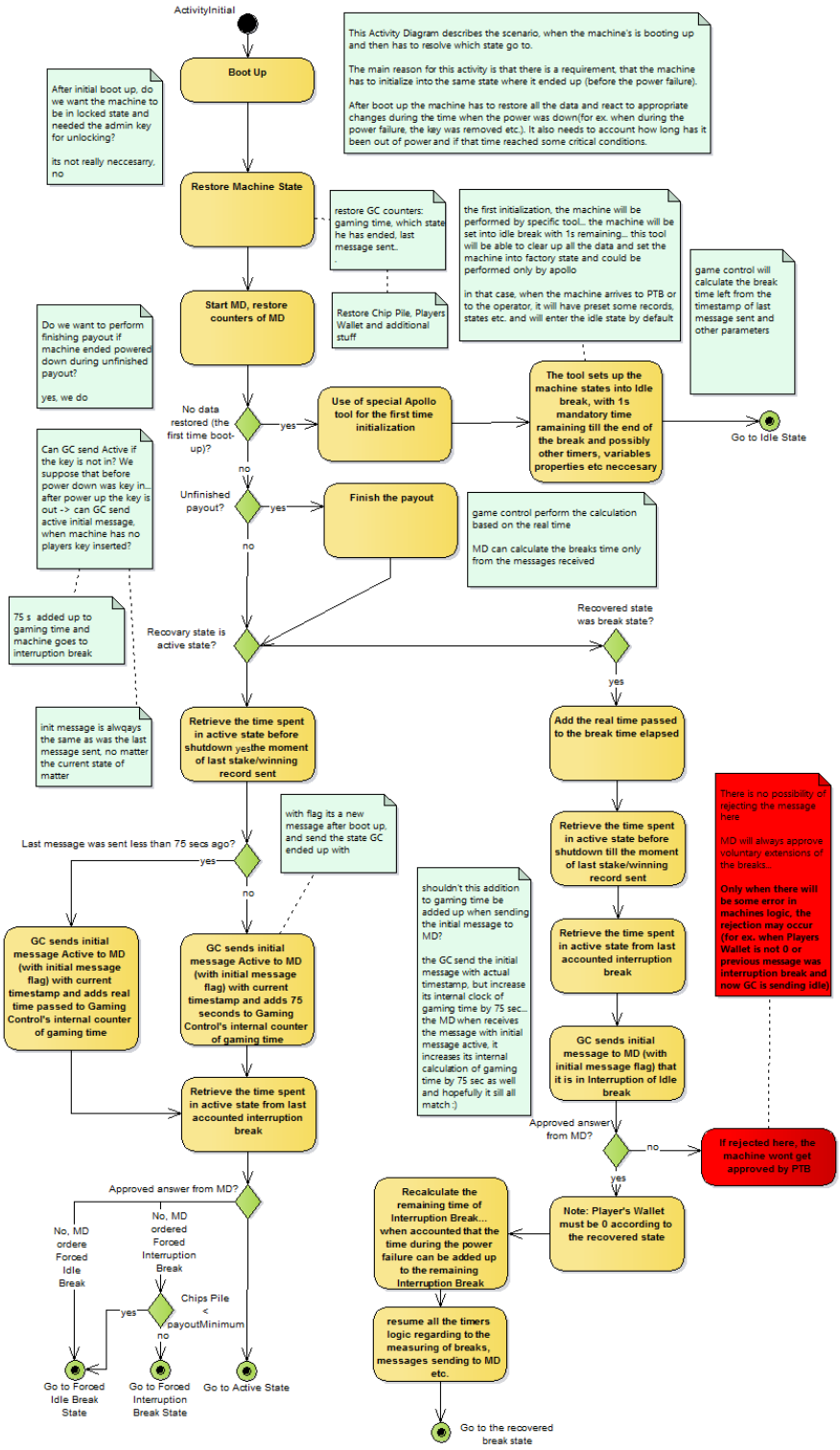


Figure B.15: System UC00 - Initialise

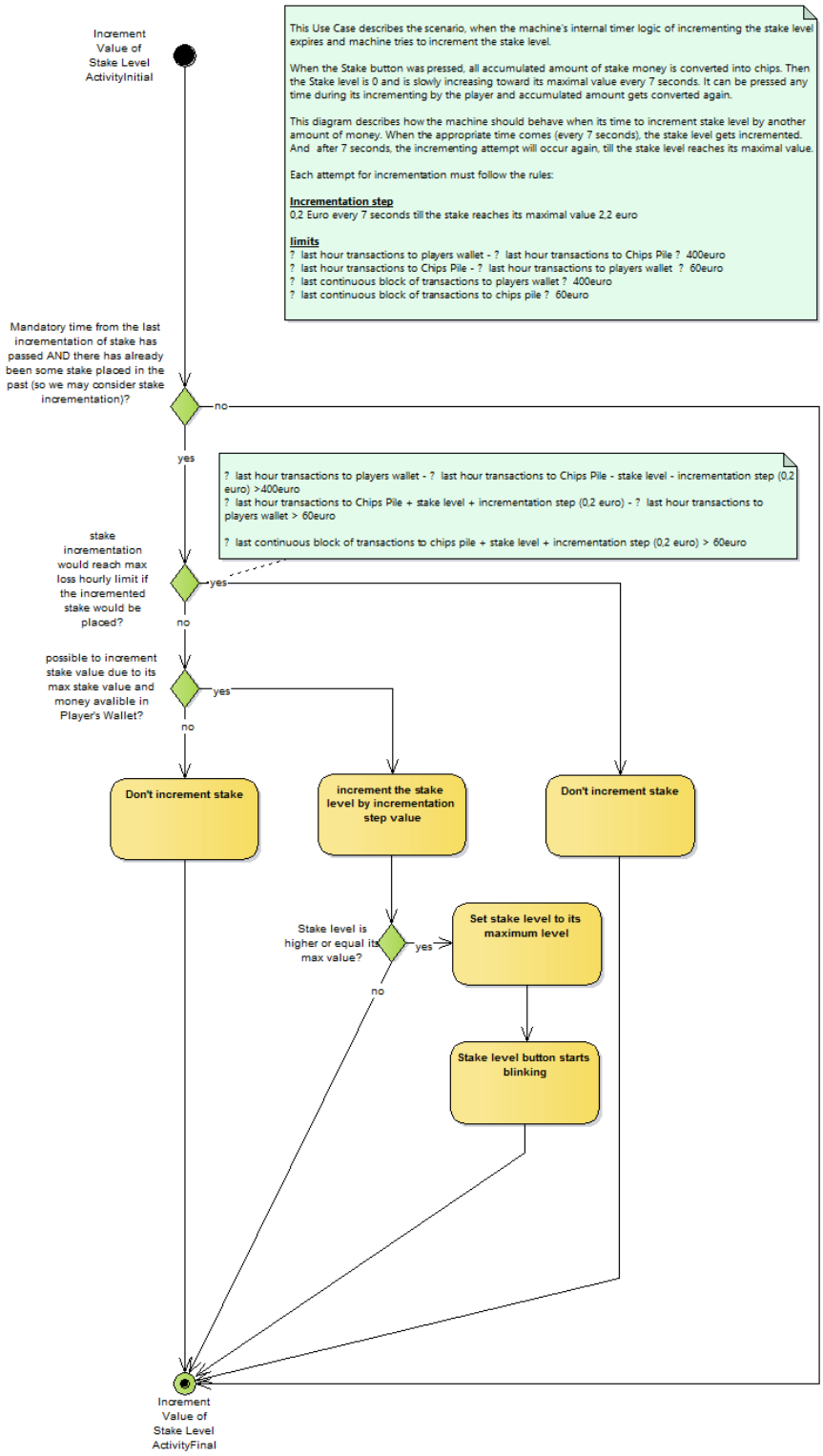


Figure B.16: System UC01 - Increment Value of Stake Level

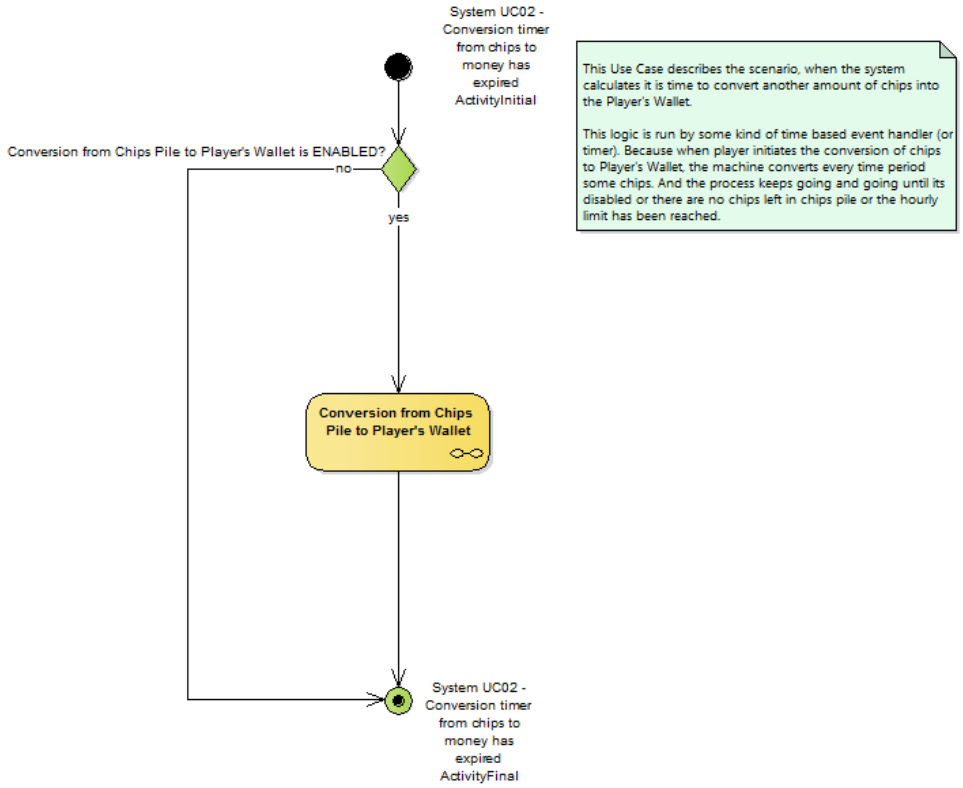


Figure B.17: System UC02 - Converse some chips to money

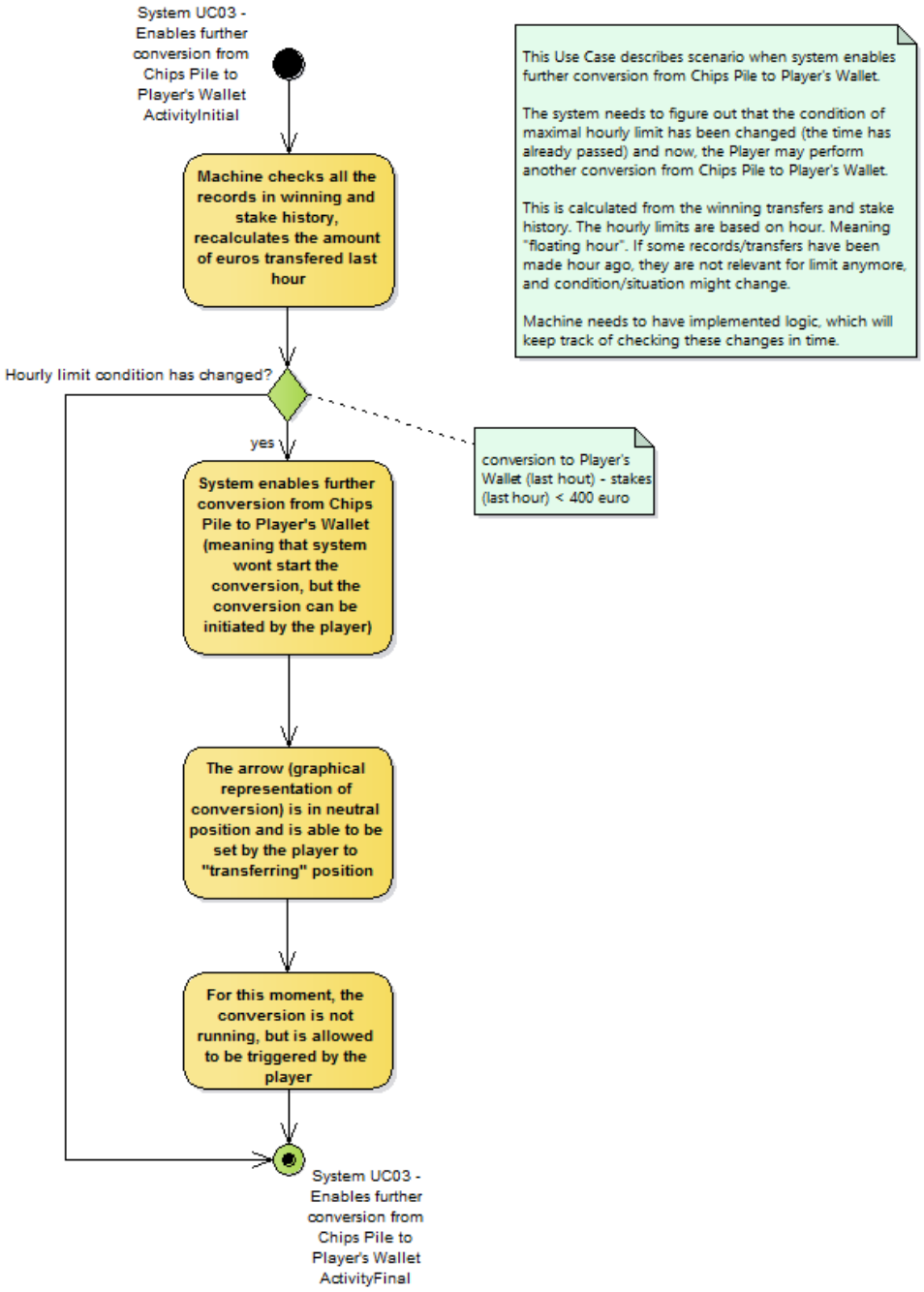


Figure B.18: System UC03 - Enables further conversion from Chips Pile to Player's Wallet

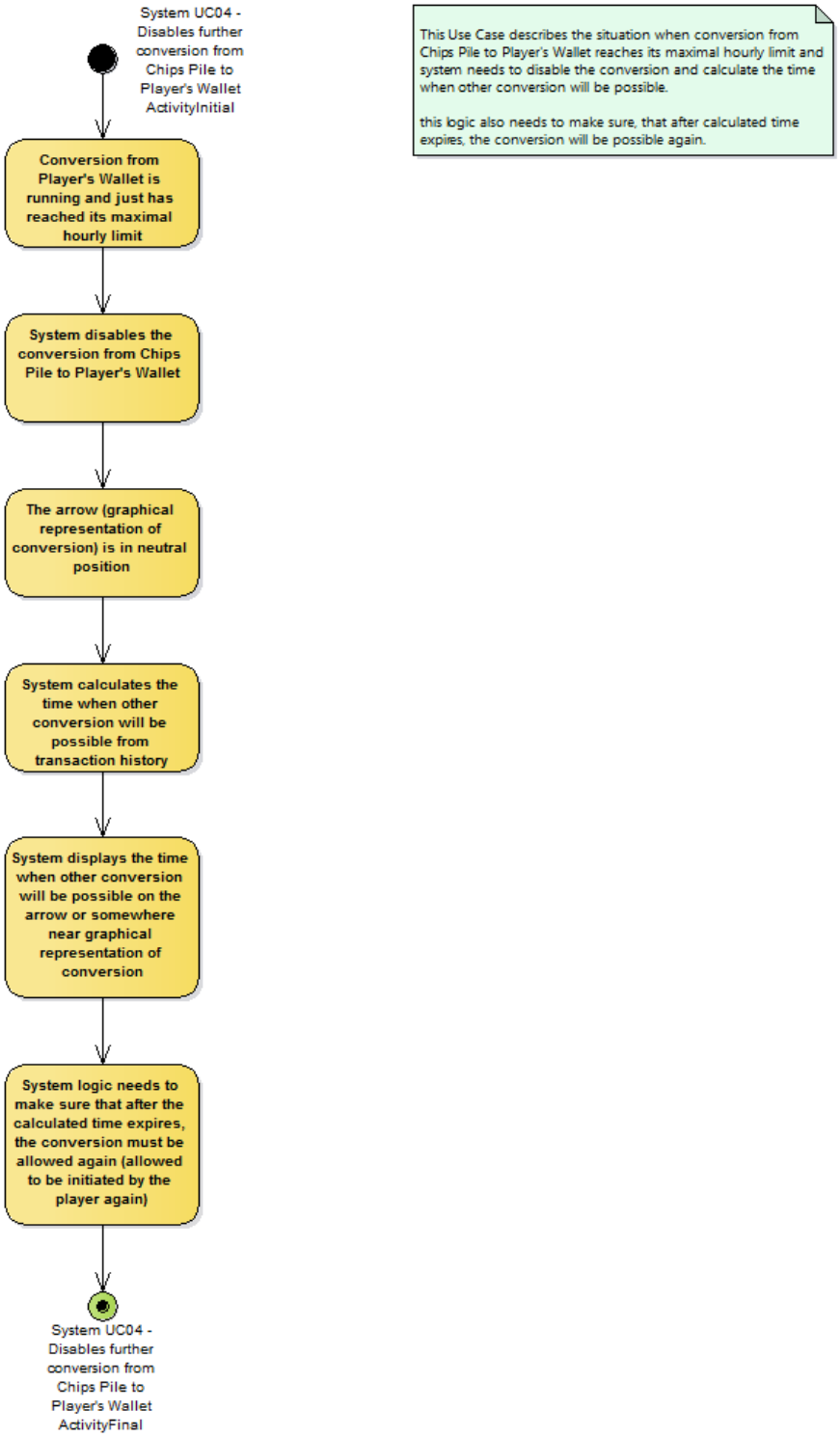


Figure B.19: System UC04 - Disables further conversion from Chips Pile to Player's Wallet

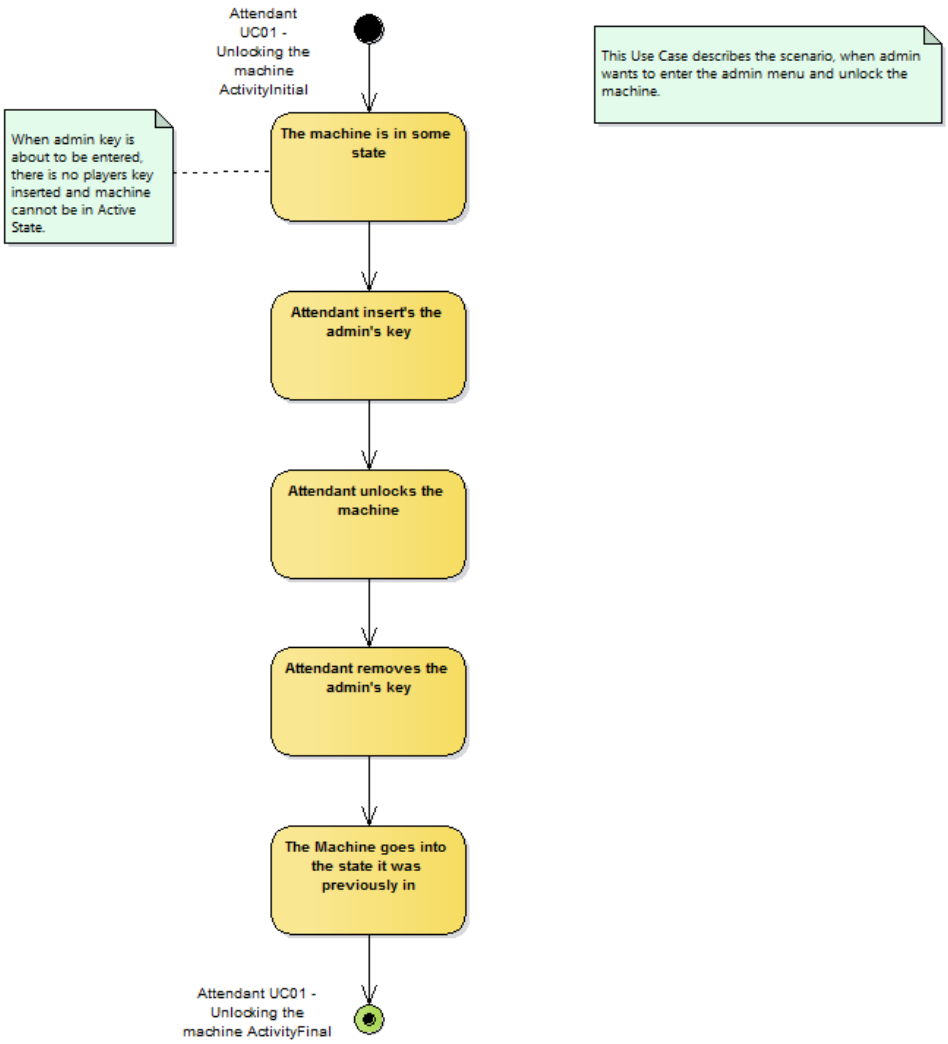


Figure B.20: Attendant UC01 - Unlocking the machine

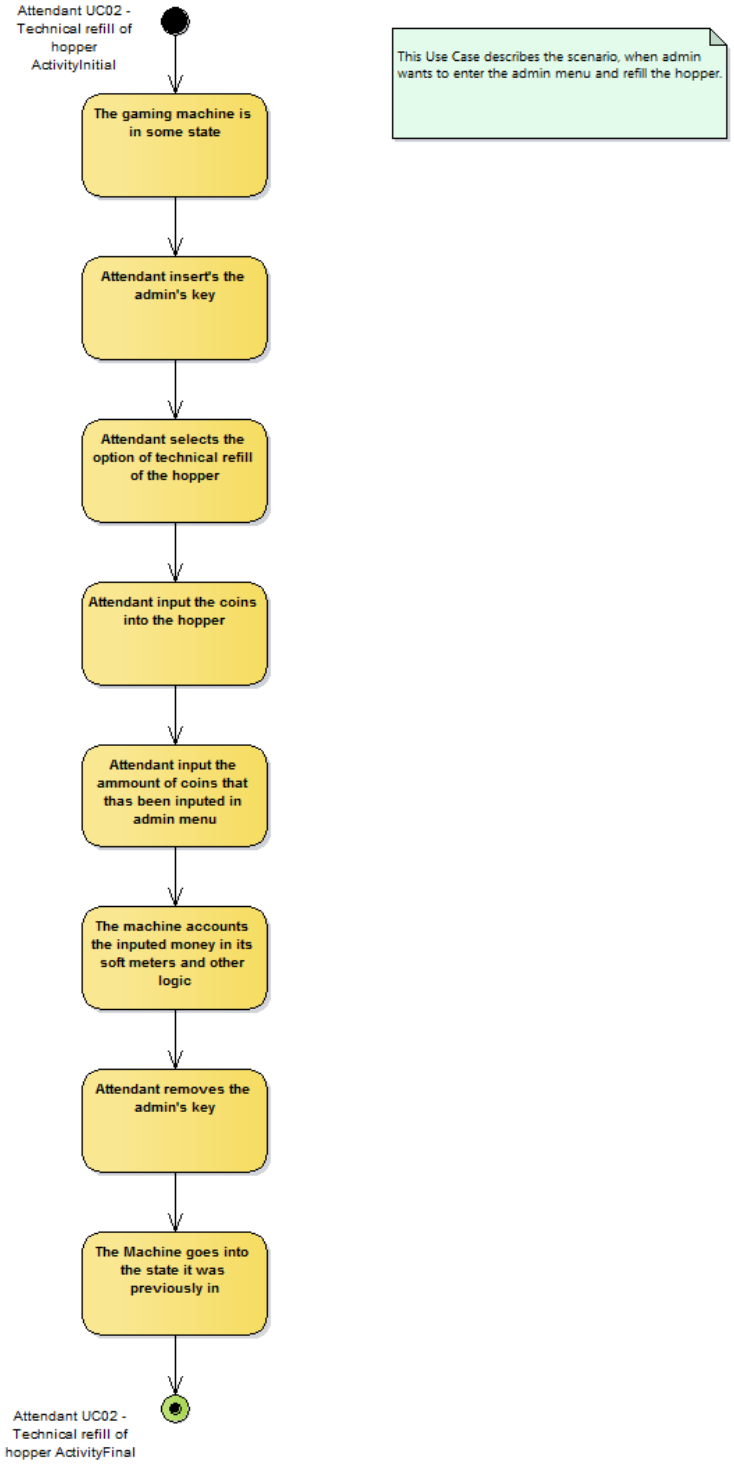


Figure B.21: Attendant UC02 - Technical refill of hopper

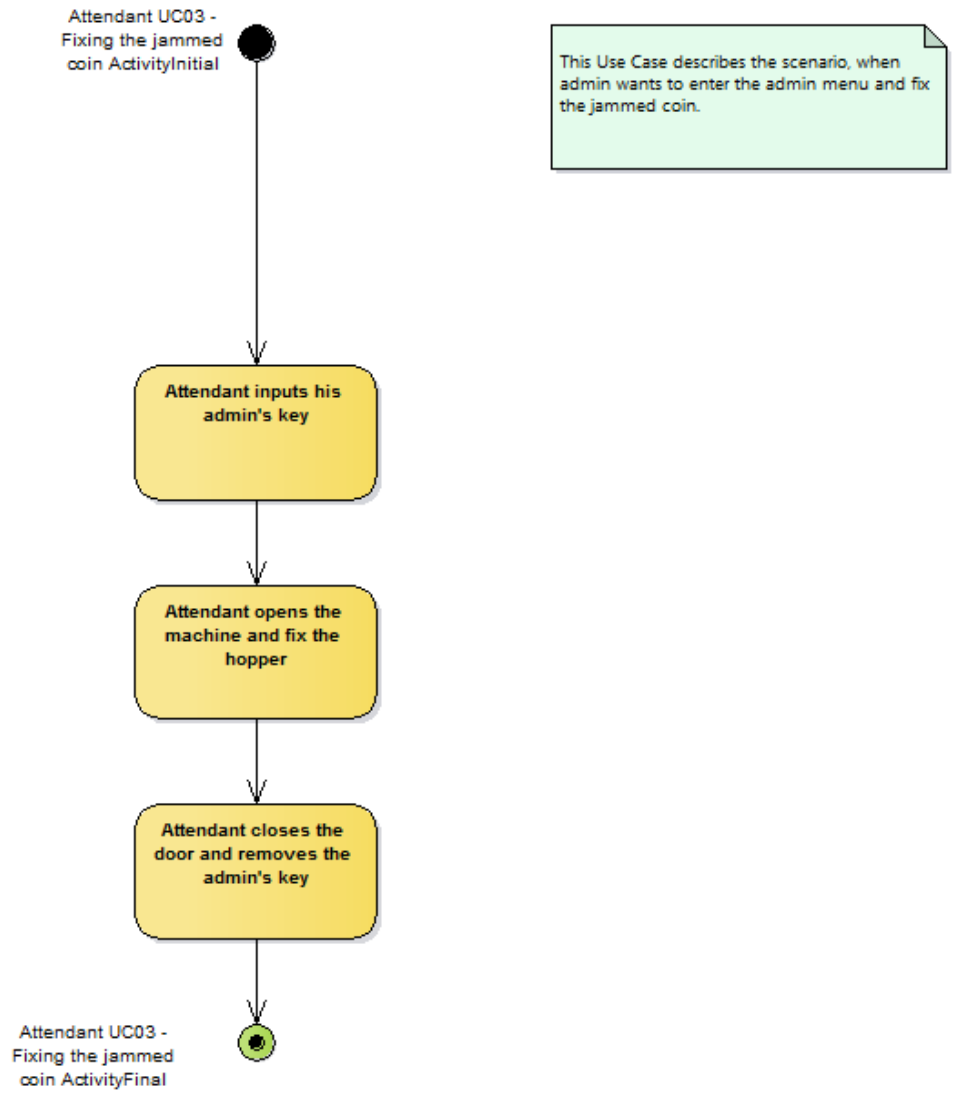


Figure B.22: Attendant UC03 - Fixing the jammed coin

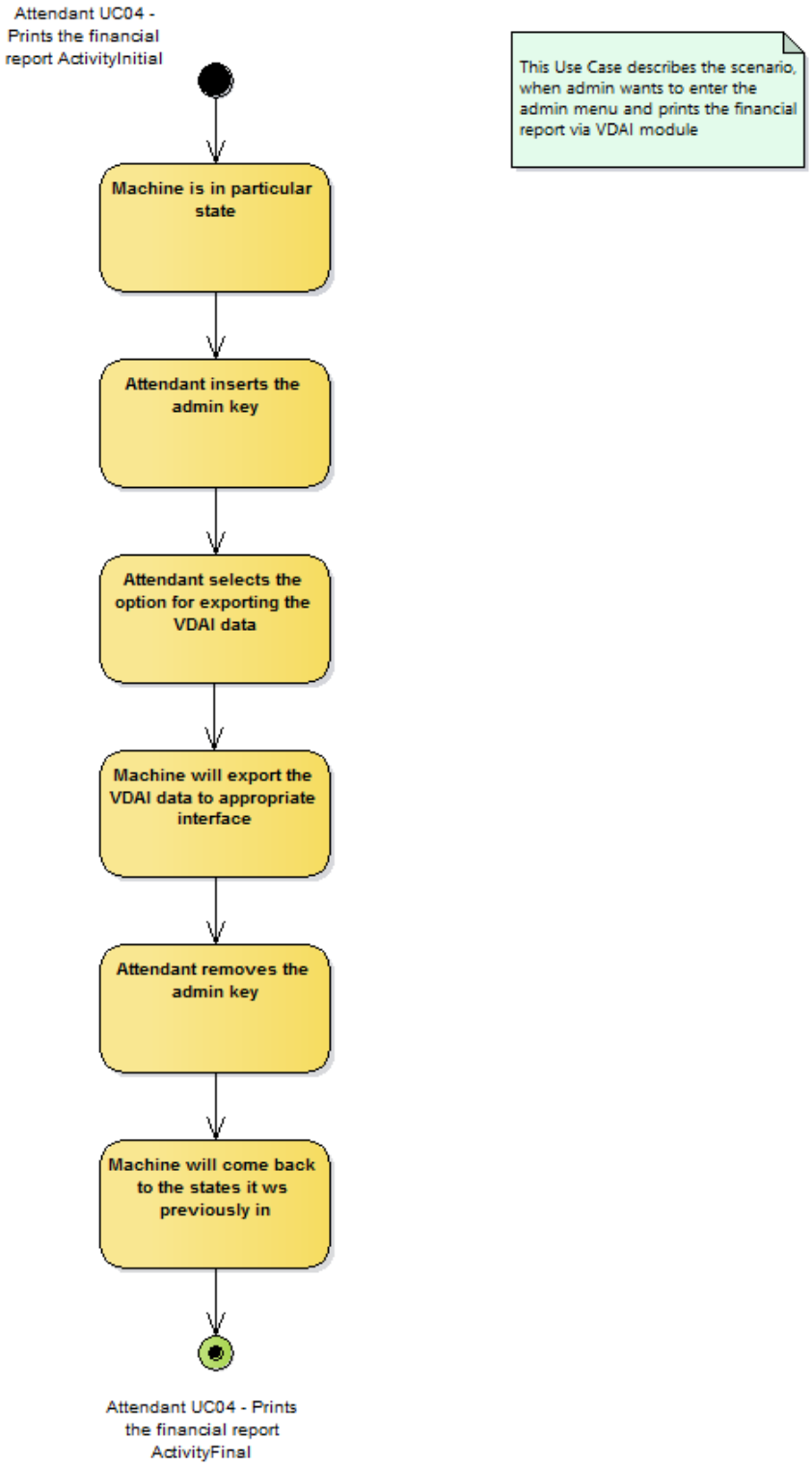


Figure B.23: Attendant UC04 - Prints the VDAI report

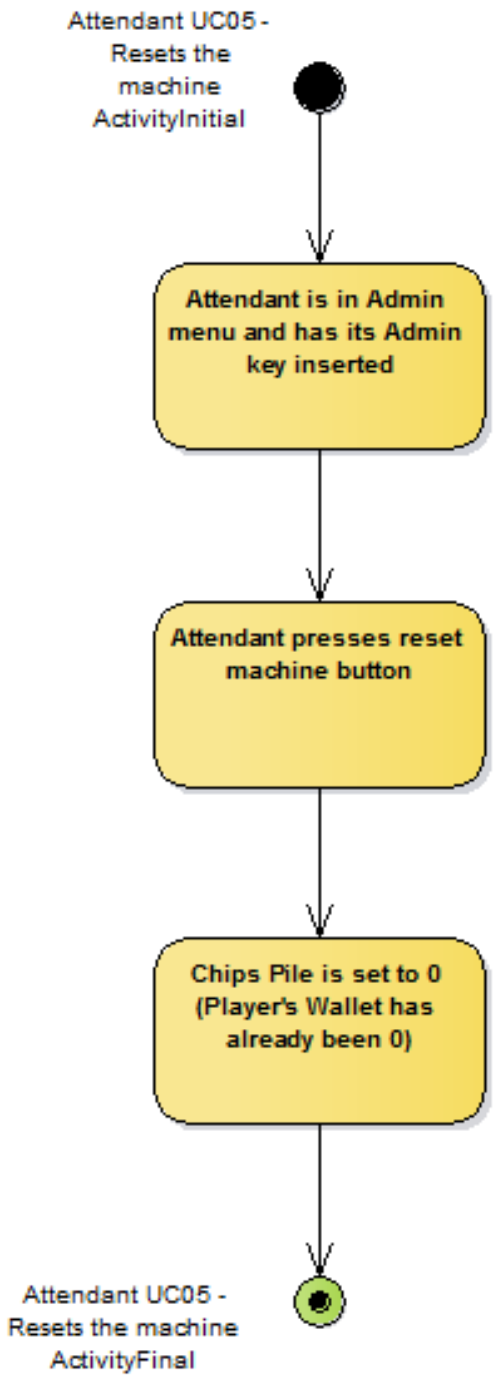


Figure B.24: Attendant UC05 - Resets the machine

Appendix C

WBS

ID	Task Mode	Task Name	Total Duration	Duration
0		**DEVHP_WBS_draft_0.2**	**1269 days**	**380 days**
1		**DE VHP**	**1269 days**	**380 days**
2		**Preparation works**	**76 days**	**40 days**
3		**New cabinet with new key system, buttons, ports**	**40 days**	**40 days**
24		**Graphics prototype**	**36 days**	**18 days**
37		**Introduction into the project**	**10 days**	**10 days**
39		**Development**	**1183 days**	**335 days**
40		**New game panel - 1 game for certification**	**45 days**	**45 days**
41		New panel for games (with buttons)	25 days	5 wks
42		New panel for games (with buttons)	20 days	4 wks
43		**New panel for carousel**	**30 days**	**35 days**
44		New adjusted panel for carousel	15 days	3 wks
45		New adjusted panel for carousel	15 days	3 wks
46		**New HW button logic (SW-ID)**	**10 days**	**25 days**
49		**PTB test module**	**140 days**	**140 days**
50		Development of testing tool of PTB test casses	20 days	4 wks
51		Implementation of forwarding the messages due to the PTB	10 days	2 wks
52		Implementation of logic initialising the PTB testing configuration	10 days	2 wks
53		Implementation of module digital signature/checksum for PTB	10 days	2 wks
54		Implementation of proxy object between the Monitoring Device	10 days	2 wks
55		Implementation of PTB port drivers	10 days	2 wks
56		PTB port initialisation sequence	10 days	2 wks
57		PTB test cases messages' data structure	10 days	2 wks
58		Simulate PTB's test cases for Apollo GC, MD	30 days	6 wks
59		Enable automatic stakes in testing mode with PTB	10 days	2 wks
60		Erasure of time-stamps (records) during test	10 days	2 wks
61		**Conversion and payouting**	**105 days**	**105 days**
62		Implementation of Chips Pile	25 days	5 wks
63		Implementation of Player's Wallet	15 days	3 wks
64		Conversion algorythm from Player's Wallet to Chips Pile and vice	20 days	4 wks
65		Conversion algorythm from Player's Wallet to Chips Pile and vice	20 days	4 wks
66		Hopping out the Player's Wallet when enforced break begin	10 days	2 wks
67		Implementation of two hoppers - possibly even coin selector	15 days	3 wks
68		**Fiscal module**	**75 days**	**65 days**
69		Fiscal data structure	5 days	1 wk
70		Fiscal database	5 days	1 wk
71		Fiscal interface handling	15 days	3 wks
72		Fiscal data export logic	15 days	3 wks
73		Fiscal data downloading in progress screen	15 days	3 wks
74		Fiscal data medium error screen	10 days	2 wks
75		Fiscal data medium error screen	10 days	2 wks
76		**State machine**	**200 days**	**180 days**
77		Development of initialisation tool for Apollo	15 days	3 wks
78		Implementation of active state	10 days	2 wks
79		Implementation of error states (hopper out of money/jammed	10 days	2 wks
80		Implementation of forced idle break state logic	10 days	2 wks
81		Implementation of idle state logic	10 days	2 wks
82		Implementation of Long Term Payout	20 days	4 wks
83		Implementation of new state-transition logic	5 days	1 wk
84		Implementation of voluntary idle break state logic	10 days	2 wks
85		Implementation of voluntary interruption break state	10 days	2 wks
86		Implementation of voluntary interruption break state due to the	10 days	2 wks
87		Interruption break following one hours' game play	10 days	2 wks
88		Interruption state of gaming machines	10 days	2 wks
89		Long term payout screen during the active state	10 days	2 wks
90		Long term payout screen during the active state	20 days	4 wks
91		Long term payout screens during the long time payout states	5 days	1 wk
92		Long term payout screens during the long time payout states	5 days	1 wk
93		Recovery of the state logic after the system reboot	20 days	4 wks
94		Rest period following three hours' game play	10 days	2 wks
95		**Game control module**	**150 days**	**105 days**
96		Game Control module implementation	20 days	4 wks
97		Extension of Admin menu logic	15 days	3 wks
98		Development of PTB unit tests	15 days	3 wks
99		Game Control - Continuous comunication with Monitoring Device	5 days	1 wk
100		Game Control - Implementation of algorithm when to offer Long	5 days	1 wk
101		Game Control - Implementation of forced payout based on high	5 days	1 wk
102		Game Control - Implementation of logic calculating the future	10 days	2 wks
103		Game Control - Implementation of logic of automatic	10 days	2 wks
104		Game Control - Implementation of logic of automatic transfer	10 days	2 wks
105		Game Control - Implementation of logic to measure the breaks	10 days	2 wks
106		Game Control - Implementation of logic to trigger transitions	5 days	1 wk
107		Game Control - Implementation of logic to trigger transitions	5 days	1 wk
108		Game Control - Implementation of logic which is listening to	5 days	1 wk
109		Game Control - Implementation of messaging system and	5 days	1 wk
110		Game Control - Implementation of tool to measure elapsed and	5 days	1 wk
111		Game Control - Implementation of transaction limits in time	5 days	1 wk
112		Game Control - Implementation of transaction record history	5 days	1 wk
113		Game Control - Long Term Payout Logic of limits and transitions	10 days	2 wks
114		**Monitoring device**	**40 days**	**40 days**
115		Implementation of initialisation procedure of the machine	15 days	3 wks
116		Monitoring Device - Implementation of logic that evaluates the	10 days	2 wks
117		Monitoring Device - Implementation of logic that would throw	5 days	1 wk
118		Monitoring Device - Implementation of message queue	5 days	1 wk
119		Monitoring Device - Implementation of messaging system with	5 days	1 wk
120		**Admin menu**	**20 days**	**20 days**
121		Implementation of new buttons in Admin Menu (Reset the	10 days	2 wks
122		Implementation of new buttons in Admin Menu (Reset the	10 days	2 wks
123		**Other**	**25 days**	**30 days**
124		Means of checking prototype copies in the field	5 days	1 wk
125		New key system handling	5 days	1 wk
126		SSD disc - formats, handling, crypto	5 days	1 wk
127		Gaming machine security	5 days	1 wk
128		Average loss	5 days	1 wk
129		**Time and money limits**	**33 days**	**110 days**
130		Device time display	3 days	3 days
131		Graphical representation of timers (bars) during gameplay	5 days	1 wk
132		Implementation of screen showing gaming statistics of last	5 days	1 wk
133		Implementation of screen showing gaming statistics of last	5 days	1 wk
134		Minimum game duration	5 days	1 wk
135		Stake and winning amounts	5 days	1 wk
136		Time measurement precision	5 days	1 wk
137		**VDAI**	**70 days**	**65 days**
138		VDAI port	5 days	1 wk
139		VDAI port	5 days	1 wk
140		Development of the tool for export VDAI reports	20 days	4 wks
141		Implementation of saving VDAI data	10 days	2 wks
142		Implementation of saving VDAI data DB	10 days	2 wks
143		Implementation of VDAI port drivers	5 days	1 wk
144		Persistent medium/memory for VDAI data	5 days	1 wk
145		VDAI data export fail screen	5 days	1 wk
146		VDAI data export fail screen	5 days	1 wk
147		**Integration and fine tuning**	**135 days**	**335 days**
148		Integration and fine tuning	40 days	8 wks
149		Graphical fine tuning	40 days	8 wks
150		Identifying game systems and variants	10 days	2 wks
151		Tuning for other games	45 days	9 wks
152		**Documentation**	**50 days**	**50 days**
153		Any special requirements with respect to servicing?	10 days	2 wks
154		Appropriate software documentation	20 days	4 wks
155		General documentation	20 days	4 wks
156		**Translation and others**	**55 days**	**55 days**
157		Translation into German language	20 days	4 wks
158		Updating of the hint inside the games by text describing the logic	15 days	3 wks
159		Updating of the hint inside the games by text describing the logic	10 days	2 wks
160		Use of Gaming Ordinance terms	10 days	2 wks

Figure C.1: WBS

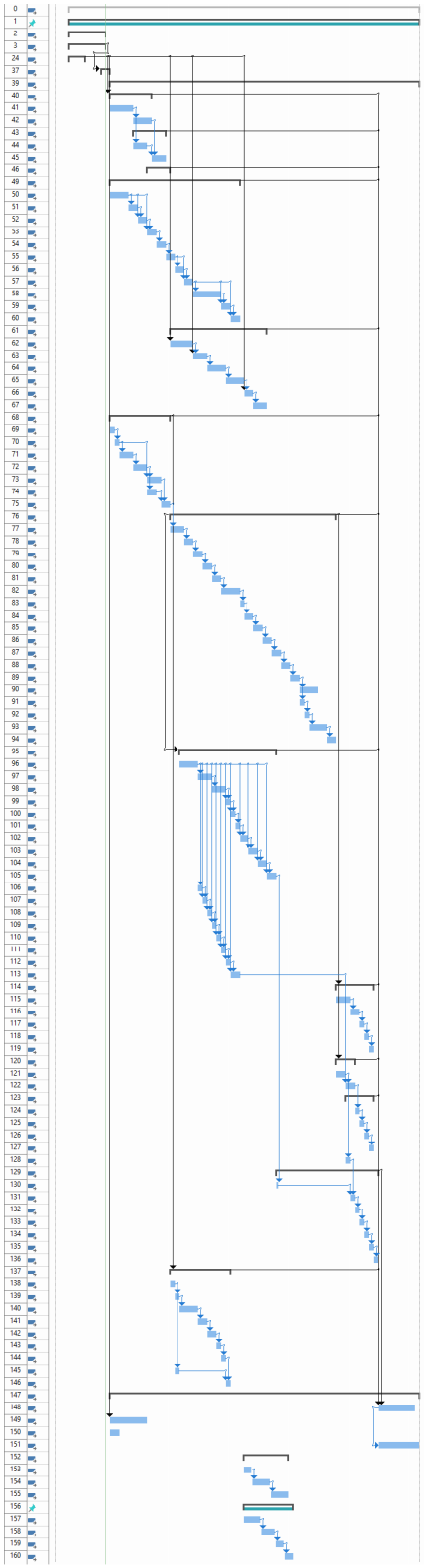


Figure C.2: WBS - Extension with work dependencies

Appendix D

Requirements List

Game Control - Implementation of transaction limits in time

TR5 Refference: **TR5 5.4**
Type: **Functional**

Description: "Implementation of logic that would check with each transaction, if it wont breach the hourly limits like:

Fiscal database

TR5 Refference: **TR5 5.16**
Type: **HW**

Description: "This database will contain all relevant information used for financial statistics (mainly for the official financial authorities).

Game Control - Implementation of logic to measure the breaks

Type: **Functional**

Description: "Implementation of module/object which will keep track and measure elapsed time from certain moment.

SW-ID Button

Type: **HW**

Description: New hardware toggle button with light inside, which indicates if the button is being toggled in or out.

Fiscal data medium error screen

Type: **GUI**

Description: When fiscal data medium is not initialized, the error screen must appear.

Enable automatic stakes in testing mode with PTB

TR5 Refference: **TR5 6.1.1**
Type: **Functional**

Description: The ban on automatic stakes can be annulled in relation to the type test - and only the type test.

Monitoring Device - Implementation of messaging system with Gaming Control

TR5 Refference: **TR5 5.13**
Type: **Functional**

Description: Logic of receiving and sending the messages between the Gaming Control and Monitoring Device

Implementation of screen showing gaming statistics of last floating hour

Type: **GUI**

Description: After initiating new player session, it must be shown to the player info screen with statistics of stakes and winning from last floating hour.

Implementation of proxy object between the Monitoring Device and Game Control

Type: **Nonfunctional**

Description: "Implementation of proxy object, which will forward the messages to appropriate modules based on current configuration.

Hopping out the Player's Wallet when enforced break begin

TR5 Refference: **TR5 4.3**
Type: **Functional**

PTB test cases messages' data structure

TR5 Refference: **TR5 7.3**
Type: **Functional**

Description: Data format of messages during PTB's test cases according to TR 5. (with content and coding)

Fiscal data downloading in progrress screen

Type: **GUI**

Description: when fiscal data are being downloaded, an appropriate downloading screen must be shown

Screen of forced idle break

TR5 Refference: **TR5 5.8**
Type: **GUI**

Description: Design of the screen, layout of the mandatory elements viz. Enterprise Architect project

Minimum game duration

TR5 Refference: **TR 5.2**
Type: **Functional**

Description: "Definition of game duration.

Stickers for marking the places where to insert coins/banknotes

Type: **Nonfunctional**

Description: Every place where the player should put in the coins or banknotes should be marked properly so its is clear where to put coins and where to put banknotes

New adjusted panel for carousel

Type: **GUI**

Description: Design of the screen, layout of the mandatory elements (gaming meters, chips pile, players wallet) viz. Enterprise Architect project

Screens of voluntary interruption breaks

TR5 Refference: **TR5 5.6**
Type: **GUI**

Description: Design of the screen, layout of the mandatory elements viz. Enterprise Architect project

Gaming machine security

TR5 Refference: **TR5 5.18**
Type: **Functional**

Description: Physical, electromagnetic security, safety against attack scenarios laid down in Appendix 4

New key system

TR5 Refference: **TR5 5.17**
Type: **HW**

Description: "Design and implementation of new key system.

Implementation of initialisation procedure of the machine

Type: **Functional**

Description: When machine is initializing, it must recover to the state it has ended up in. While ended up in active state, the additional 75 seconds are added up to gaming time and message with initialization flag is sent to MD. While ended up in break state, the real time difference is added up to break time.

Game Control - Implementation of transaction record history between the Player's Wallet and Chips Pile

Type: **Functional**

Description: "Implementation of record history, which will store all the transaction in last hour. Meaning that every second, the history needs to be recalculated and outdated records deleted.

Game Control - Implementation of tool to measure elapsed and passed time (timer/counter)

Type: **Functional**

Description: "Implementation of module/object which will keep track and measure elapsed time from certain moment and possibly trigger time-based conditions.

Reports

TR5 Refference: **TR5 3**
Type: **Nonfunctional**

Description: Subject of the safety report TR5 3.1, Safety reports TR5 3.2, Recognised bodies for safety reports TR5 3.4, Safety report validity TR5 3.5, Other reports TR5 3.6,

Implementation of new state-transition logic

Type: **Functional**

Description: Implementation of new declarative machine state logic into standalone code baseline

Other designations and descriptions

TR5 Refference: **TR5 2.13**
Type: **Nonfunctional**

Description: "With regard to other regulated functions, the following designations and labels

Erasure of time-stamps (records) during test

TR5 Refference: **TR5 6.1.3 Fig. 7.7**
Type: **Functional**

Description: Deletion of the saved time stamps in the MD in PTB's test configuration

Graphical representation of timers (bars) during gameplay

Type: **GUI**

Description: In game/In carousel design and graphical functionality of meters measuring the gaming time and gaming time left till the mandatory break

Simulate PTB's test cases for GC, MD

TR5 Refference: **TR5 6.3, Fig 6.2**
Type: **Functional**

Description: Testing cases by PTB

Fiscal data export logic

TR5 Refference: **TR5 4.4**
Type: **Functional**

Description: "Logic that would export the fiscal data. In admin menu, there will e extra fiscal data export button. The button will on press generate fiscal xml files and save them to usb stick.

Implementation of forced idle break state logic

TR5 Refference: **TR5 5.8**
Type: **Functional**

Description: "State, which cannot be left for 5 min, and after 5 min, all machine is reset and set for idle.

New panel for games (with buttons)

Type: **GUI**

Description: Design of the screen, layout of the mandatory elements like buttons(stake button, conversion arrow, stake meter, winning area, player's wallet, gaming meters..) viz. Enterprise Architect project User Interface Model

Game Control - Continuous comunication with Monitoring Device about transaction and elapsed time

Type: **Functional**

Description: "Every possible transaction has to be checked with Monitoring Device if it may be performed. All the answers must be that the transaction may be performed due to the limits otherwise the machine wont get certified. So gaming control must keep track of limits in time by its own.

Game Control - Implementation of logic to trigger transitions between the states on time-based events

Type: **Functional**

Description: During the break time or Active time, machine needs to force transitions to appropriate different states based on value of time elapsed.

Sticker for prevent minors from gaming, posibility of medical help

TR5 Refference: **TR5 2.9**
Type: **Nonfunctional**

Description: "Clearly visible warning notices11 relating to excessive playing and the protection

Means of checking prototype copies in the field

TR5 Refference: **TR5 1.4**
Type: **Functional**

Description: "Identifiers for hardware and software components, Display of Checksums for GC and MD, Device-time,

Implementation of forwarding the messages due to the PTB testing procedures

Type: **Functional**

Description: "Implementation of proxy object, which will forward the messages to appropriate modules based on current configuration.

Implementation of voluntary interruption break state due to the player's innactivity

TR5 Refference: **TR5 5.6**
Type: **Functional**

Description: Implementation of Voluntary Interruption Break due to the players Inactivity and its proper logic regarding transition, payout, key insertion, handle of the time elapsed in breaks etc.

VDAI port

Type: **HW**

Description: RS-232 connector on the machine.

Implementation of voluntary idle break state logic

TR5 Refference: **TR5 5.8**
Type: **Functional**

Description: State, when entered the state cannot be left for 15 sec and after 5 min gets accounted as forced idle break and all machine is reset.

Fiscal interface

TR5 Refference: **TR5 5.16**
Type: **HW**

Description: "Interface for transfer of fiscal data to external storage media. The interface is used to copy digitally signed XML files to USB flash disk. USB3 standard has to be used for the interface implementation.

Interruption state of gaming machines

TR5 Refference: **TR5 5.5**
Type: **Functional**

Description: "A gaming device state in which no stakes are accepted and no winnings awarded

Monitoring Device - Implementation of message queue

TR5 Refference: **TR 5.14**
Type: **Functional**

Description: Implementation of own record history of Monitoring Device, on which the evaluation of the limits will be based on.

Game Control - Implementation of logic calculating the future time of fulfilling the limits in the future

TR5 Refference: **TR5 5.4**
Type: **Functional**

Description: "During the breaks, the timer is shown when the break is over

Any special requirements with respect to servicing must be documented

TR5 Refference: **TR5 1.6**
Type: **Nonfunctional**

Description: "If servicing is necessary to ensure the operation...

Money in payout balances

TR5 Refference: **TR5 6.1.2**
Type: **Functional**

Description: Simulate the insertion of money to Player's Wallet when the machine runs out of money during test mode. When balance is too low, machine will automatically recharge itself.

Conversion algorythm from Player's Wallet to Chips Pile and vice versa

Type: **Functional**

Description: "Design of an algorithm for conversion 1 euro to 100 chips with random deviation 0,75. •

Use of Gaming Ordinance terms

TR5 Refference: **TR5 2.10**
Type: **Language**

Description: The terms like Stake/Winning must be used appropriately

No transfers from the gaming device

TR5 Refference: **TR5 5.20**
Type: **Nonfunctional**

Description: No unauthorized reading out of the data. No readout of information about game conditions (e.g. stakes, winnings, fill levels, times when the devices are operational or idle)in the last 24 hours

General documentation

TR5 Refference: **TR5 Appendix 2**
Type: **Nonfunctional**

Description: Lots of documentation, should start in time to write this.

Implementation of new buttons in Admin Menu (Reset the machine..)

Type: **Functional**

Description: "Implementation of possibility to reset the machine in matter of forcing Forced Idle Break State

VDAI data downloading in progress screen

Type: **GUI**

Description: When VDAI data are being downloaded, an appropriate downloading screen must be shown

Implementation of VDAI port drivers

Type: **HW**

Description: New connector and appropriate logic regarding its handling. VDAI port is not active all the time. Activated at admin menu (after admin key insert).

SW-ID Button logic

TR5 Refference: **TR5 2.7**
Type: **Functional**

Description: "When SW-ID button pressed, show the SW version, checksum etc. If some animation, break will terminate in 75 sec, then wait for the event to finish, otherwise interrupt the event and show SW-ID screen right away

"Place for identification sticker of the machine ""Machine designation panel"""

TR5 Refference: **TR5 2.4**
Type: **Nonfunctional**

Description: "2 options of dimensions: 120 mm x 25 mm or 60 mm x 50 mm

Stake Button

Type: **HW**

Description: New hardware button replacing our max-bet button

Screens of voluntary idle break

TR5 Refference: **TR5 5.8**
Type: **GUI**

Description: Design of the screen, layout of the mandatory elements viz. Enterprise Architect project

Updating of the hint inside the games by text describing the logic of German standalone machine

Type: **Functional**

Description: Updating of the HINT screen in games/carousel describing the machines functionality

Implementation of error states (hopper out of money/jammed coin..)

Type: **Functional**

Description: "Implementation of Error state, that if machine is out of coins, hopper jammed or any other problem, the machine goes into voluntary interruption break and keeps proper logic regarding the break logic etc.

Placement of the inspection sticker

TR5 Refference: **TR5 2.3**
Type: **Nonfunctional**

Description: "A space measuring 45 mm x 45 mm directly adjacent to the approval mark (see

Game Control - Implementation of logic of automatic transfer from Chips Pile to Player's Wallet regarding the limits

TR5 Refference: **TR5 5.3**
Type: **Functional**

Description: "Implementation of logic that would continually transfer of chips to euros. The event is triggered by the player and first transferred amount is up to 23 euros (depends on the chips available and hourly limits) and then 2 euros every 5 seconds. The check for possible breach of hourly limits has to be performed here as well and check for funds available in Chips Pile.

Implementation of two hoppers - possibly even coin selector

TR5 Refference: **TR5 5.11**
Type: **HW**

Description: One hopper for 2 euro coins, one for 20 euro cents coins

Game Control - Implementation of logic which is listening to Monitoring Device for forced breaks transitions

Type: **Functional**

Description: Game control must at some point obey the forced break commands from Monitoring Device. This logic must force transition to break states if its triggered by message from Monitoring Device.

PTB port

TR5 Refference: **TR5 7.1**
Type: **HW**

Description: "RS232 connector on the machine.

Time measurement precision

TR5 Refference: **TR5 5.15**
Type: **Nonfunctional**

Description: In 4 years, machine must deviate max 10 minutes from real time.

Placement of the approval certificate

TR5 Refference: **TR5 2.2**
Type: **Nonfunctional**

Description: "An appropriate compartment with an unrestricted observation window

No possible registrations of player

TR5 Refference: **TR5 5.21**
Type: **Nonfunctional**

Description: no player-related information may be stored

Recovery of the state logic after the system reboot

TR5 Refference: **TR5 5.9**
Type: **Nonfunctional**

Description: While machine shuts down or reboot, it must recover to the state it was before. Further logic follows depending on which state it gets recovered to. Some addition 75 seconds might be add to gaming time, if in break, the time spent in power-down is accounted to break time etc.

Implementation of saving VDAI data DB

Type: **Nonfunctional**

Description: Implementation of module and its database that will keep track on VDAI data. Meaning the transaction records between the chips pile and player's wallet.

Sticker/Sign for marking the Payout Button

TR5 Refference: **TR5 2.11**
Type: **Nonfunctional**

Description: "Payout button has to be marked properly so the functionality is clear

Implementation of Chips Pile

Type: **Functional**

Description: Implementation of new item/variable where player can transfer his actual money - need to implement proper logic of conversion, keeping the limits etc.

Identifying game systems and variants

TR5 Refference: **TR5 2.12**
Type: **Security**

Description: "Different game systems or variants within a particular design must be identified.

Extension of Admin menu logic

TR5 Refference: **TR5 5.6**
Type: **Functional**

Description: While in admin menu, the machine must keep running break logic on the background and while leaving the admin menu, machine must decide properly which state to return to.

Game Control - Implementation of messaging system and messages structure with Monitoring Device

Type: **Functional**

Description: Implementation of communication protocol and its data structure between main two system components (GC and MD)

Implementation of active state

Type: **Functional**

Description: Implementation of modifications regarding the gameplay - machine needs to measure gaming time, perform proper conversion actions of chips and money, spin, payout, time bars of measuring gaming time, perform proper transitions to break states, send message to MD at least every 75 sec and not frequently as 0,1 sec etc.

Interruption break following one hours' game play

TR5 Refference: **TR5 5.6**
Type: **Functional**

Development of the tool for export VDAI reports

Type: **Nonfunctional**

Description: Test tool

Game Control

Type: **Functional**

Description: "Implementation of module, which could manage the transition between the states based on event-based conditions, but also on time-based conditions

Rest period following three hours' game play

TR5 Refference: **TR5 5.8**
Type: **Functional**

SSD disc for databases records

TR5 Refference: **TR5 5.16**
Type: **HW**

Description: Disc for database records for VDAI module, fiscal data, other records and possibly other components

Implementation of saving VDAI data

Type: **Nonfunctional**

Description: Implementation of module that will keep track on VDAI data. Meaning the transaction records between the chips pile and player's wallet.

Monitoring Device - Implementation of logic that evaluates the new messages regarding the limits and sending back appropriate answers

TR5 Refference: **TR5 5.4**
Type: **Functional**

Description: Every received message by Monitoring Device needs to be evaluated in regards to current records in transaction history, needs to be calculated If current possible transaction would breach hourly limit and proper respond has to be send back to Game Control

Implementation of Player's Wallet

TR5 Refference: **TR5 5.10**
Type: **Functional**

Description: Implementation of new item/variable which will store player's money - need to implement proper logic, that player must not input more than 10 euros etc.

Implementation of idle state logic

TR5 Refference: **TR5 5.7 and Appendix 2 (general information)**
Type: **Functional**

Description: "Implementing the default state of machine, while machine is waiting for key insertion to begin new player session

Game Control - Implementation of logic to trigger transitions between the states on variable-based events

Type: **Functional**

Description: During the break time or Active time, machine needs to force transitions to appropriate different states based on value of the condition.

Persistent medium/memory for VDAI data

Type: **HW**

Description: Von volatile storing of fiscal data.

Screen of forced interruption break

TR5 Refference: **TR5 5.6**
Type: **GUI**

Description: Design of the screen, layout of the mandatory elements viz. Enterprise Architect project

Implementation of module digital signature/checksum for PTB

TR5 Refference: **TR5 2.7**
Type: **Security**

Description: "Requirement for PTB testing of integrity of the software for further random check by their employees in venues.

VDAI data export fail screen

Type: **GUI**

Description: When VDAI data downloading process fails, the error screen must appear.

No external influences on the gaming device

TR5 Refference: **TR5 5.19**
Type: **Nonfunctional**

Description: No 3rd party breach to the system

PTB port initialisation sequence

TR5 Refference: **TR5 7.2**
Type: **Functional**

Description: Initializing the communication for PTB's test casses

Development of unit tests

Type: **Nonfunctional**

Description: Development of internal tool simulating PTB device - MR (testing computer connected via PTB device).

Translation into German language

Type: **Language**

Description: All the buttons, messages, hints and screens need to be translated into German.

Device time display

TR5 Refference: **TR5 2.8**
Type: **GUI**

Description: "The current device time shall be displayed continuously or following activation

Implementation of PTB port drivers

TR5 Refference: **TR5 6.1**
Type: **HW**

Description: New RS-232 connector and appropriate logic regarding its handling. Implement Transmission characters, control characters and sequences

Payout Button

Type: **HW**

Description: New hardware button on the side of the machine

"Label close to the STAKE button with text ""?Dein Einsatz w?hlen und starten?""""

Type: **Nonfunctional**

Implementation of voluntary interruption break state

TR5 Refference: **TR5 5.6**
Type: **Functional**

Description: Implementation of Voluntary Interruption break state and its proper logic regarding transition, payout, key insertion, handle of the time elapsed in breaks

Game Control - Implementation of logic of automatic incrementation of stake level due to the limits

TR5 Refference: **TR5 5.3**
Type: **Functional**

Description: "Implementation of logic that would continually increment the stake level and watch the hourly limits and money available in players wallet

Monitoring Device - Implementation of logic that would throw away obsolete messages (messages older than an hour)

Type: **Functional**

Description: Queue of history should store just messages from last hour. All older messages should be thrown away.

Random aspect of the chances of winning

TR5 Refference: **TR5 4.2**
Type: **Functional**

Description: Each player must have same chances of winning, no long term trends etc.

Game Control - Implementation of forced payout based on high balance of Player's Wallet

Type: **Functional**

Description: If player inputs a banknote that the money present in Player's Wallet plus the value of a banknote would exceed the 10 euros, the amount above 10 euros gets hopped out immediately.

Appropriate software documentation

TR5 Refference: **TR5 1.1**
Type: **Nonfunctional**

Description: "Software must have been developed with due regard to the recognized rules of software engineering

Implementation of logic initialising the PTB testing configuration

TR5 Refference: **TR5 6.1**
Type: **Nonfunctional**

Description: While PTB test module is inserted into PTB port, the initialization procedure must be correctly read and further appropriate logic must follow.

Development of initialisation tool

Type: **Functional**

Description: Development of initialization tool, which sets up machine state to idle break state with 1 sec ob mandatory break remaining

Identification of the components

Type: **Nonfunctional**

Description: "Device components (hoppers, acceptor...) must be provided with type labels, inscribed plates or other

Development of testing tool of PTB test casses

Type: **Nonfunctional**

Description: Test tool