# Table of Contents

[Table of Contents 1](#_Toc510720968)

[Abstract 3](#_Toc510720969)

[Acknowledgements 3](#_Toc510720970)

[Important Disclaimer 3](#_Toc510720971)

[Chapter 1: Introduction to the Project 4](#_Toc510720972)

[1.1 Motivation & Context 4](#_Toc510720973)

[1.2 Considerations 4](#_Toc510720974)

[1.3 Objectives 4](#_Toc510720975)

[1.4 Planning (Gantt Chart) 4](#_Toc510720976)

[Chapter 2: Background and Research 5](#_Toc510720977)

[2.1 Chess as a game 5](#_Toc510720978)

[2.1.1 Rules and difficulties 5](#_Toc510720979)

[2.1.2 Past attempts to make a chess AI 5](#_Toc510720980)

[2.2 Technical research 5](#_Toc510720981)

[2.2.1 Algorithms 5](#_Toc510720982)

[2.2.1.1 MinMax 5](#_Toc510720983)

[2.2.1.2 NegMax 5](#_Toc510720984)

[2.2.1.3 Alpha Beta Pruning 6](#_Toc510720985)

[2.2.1.4 Monte Carlo Tree Search 6](#_Toc510720986)

[2.2.1.5 Neural Networks 6](#_Toc510720987)

[2.2.2 Design Patterns 6](#_Toc510720988)

[2.2.3 Software Engineering Practices 6](#_Toc510720989)

[2.2.3.1 Documentation 6](#_Toc510720990)

[2.2.3.2 Coding 6](#_Toc510720991)

[2.2.3.3 Debugging 6](#_Toc510720992)

[2.2.3.4 Testing 6](#_Toc510720993)

[2.2.3.4.1 Unit Tests 6](#_Toc510720994)

[2.2.3.4.2 Alpha Releases and Testers 6](#_Toc510720995)

[2.2.3.4.3 Nim 6](#_Toc510720996)

[2.2.3.4.4 Chess End Games 6](#_Toc510720997)

[2.2.4 Game API 6](#_Toc510720998)

[2.2.4.1 Description 6](#_Toc510720999)

[2.2.4.2 Original API 6](#_Toc510721000)

[2.2.4.3 Features 6](#_Toc510721001)

[2.2.4.4 Changes 6](#_Toc510721002)

[2.2.5 Other considerations 6](#_Toc510721003)

[Chapter 3: Software Requirements 7](#_Toc510721004)

[3.1 Functional Requirements 7](#_Toc510721005)

[3.2 Non-functional Requirements 7](#_Toc510721006)

[Chapter 4: Feasibility Study and Choices Taken 8](#_Toc510721007)

[4.1 Software choices 8](#_Toc510721008)

[4.1.1 Eclipse JEE 8](#_Toc510721009)

[4.1.2 GitHub 8](#_Toc510721010)

[4.1.2.1 Motivation 8](#_Toc510721011)

[4.1.2.2 Folder structure 8](#_Toc510721012)

[4.1.3 Google Drive 8](#_Toc510721013)

[4.2 Model choices 8](#_Toc510721014)

[4.3 Other choices 8](#_Toc510721015)

[4.4 Time and resources estimation 8](#_Toc510721016)

[4.5 Feasibility of the project 8](#_Toc510721017)

[4.6 Testing choices 8](#_Toc510721018)

[4.7 User interface 8](#_Toc510721019)

[Chapter 5: Design, Architecture and Diagrams 9](#_Toc510721020)

[5.1 Overview 9](#_Toc510721021)

[5.2 Class Diagrams 9](#_Toc510721022)

[5.3 Sequence Diagrams 9](#_Toc510721023)

[5.4 Other design information 9](#_Toc510721024)

[Chapter 6: Components & Implementation 10](#_Toc510721025)

[6.1 Most relevant classes 10](#_Toc510721026)

[6.2 User Interface 10](#_Toc510721027)

[Chapter 7: Testing 11](#_Toc510721028)

[7.1 Unit tests 11](#_Toc510721029)

[7.2 toString() debugging 11](#_Toc510721030)

[7.3 Nim 11](#_Toc510721031)

[7.4 Chess end-games 11](#_Toc510721032)

[7.5 Playing against human subjects 11](#_Toc510721033)

[Chapter 8: Project Evaluation 12](#_Toc510721034)

[8.1 Overview 12](#_Toc510721035)

[8.2 Strengths 12](#_Toc510721036)

[8.3 Problems encountered 12](#_Toc510721037)

[8.3.1 Critical problems 12](#_Toc510721038)

[8.3.2 Other problems and solutions 12](#_Toc510721039)

[8.4 Potential improvements 12](#_Toc510721040)

[Chapter 9: Conclusions 13](#_Toc510721041)

[9.1 Overview 13](#_Toc510721042)

[9.2 Lessons learnt 13](#_Toc510721043)

[9.3 Should (not) have done 13](#_Toc510721044)

[9.4 Personal opinion 13](#_Toc510721045)

[Chapter 10: Future Work 14](#_Toc510721046)

[10.1 State of the project 14](#_Toc510721047)

[10.2 Future new features 14](#_Toc510721048)

[10.3 Improvements to existing features 14](#_Toc510721049)

[10.4 Expectations 14](#_Toc510721050)

[Chapter 11: Bibliography 15](#_Toc510721051)

# Abstract

The following report aims to describe the steps, decisions and various aspects involved in the development of a *Chess Delta* as part my Undergraduate Thesis in the past few months.

*Chess Delta* is not only a game that you can play with other people in a single computer, but also features several different AI modes.

Every aspect of the project will be discussed in the different sections of this report, including future ideas, design decisions that were made as well as flaws in the project and insights from the author on how these could have been solved or approached in different ways.

The whole project will follow best practices in Software Engineering, such as clean and properly commented code, unit and system tests, version control, documents following the pertinent IEEE standards, etc.

# Acknowledgements

Firstly, I would like to thank my family and friends for their support and patience with me during these months.

Special thanks to my father, for getting me interested in chess and teaching me how to play at a very young age, and my mother, for supporting me throughout this whole degree and project.

I am extremely grateful for all the people that believed in me since the beginning and gave me the motivation for a project of this size and level of difficulty.

I would also like to thank Dr Daniel Polani, my project supervisor, for his support and valuable advice throughout the whole process, and Dr Guy Saward, for making my stay at the University of Hertfordshire a truly pleasant experience that taught me a lot both academically and as an individual.

# Important Disclaimer

It is important to note that although the project here described IS a final version of the software, including all required features to allow subjects to play, as well as a fully functional artificial intelligence engine, some more features might be added for the presentation at my home university (Complutense University of Madrid), since the presentation date at the tribunal (if applicable) will be in July. The potential future features will be discussed at the 10th chapter of this document.

Chapter 1: Introduction to the Project

# 1.1 Motivation & Context

Since a very long time, AI has been a subject of interest for me. Although this final year project is mostly on the subject Software Engineering, some Artificial Intelligence concepts were also included as part of the project. I found the Chess game proposal

Throughout the past four years of study, I have worked on multitude of projects in different languages.

# 1.2 Considerations

…

# 1.3 Objectives

…

# 1.4 Planning (Gantt Chart)

…

Chapter 2: Background and Research

# 2.1 Chess as a game

## 2.1.1 Rules and difficulties

Chess as a game has been around for almost two centuries[[1]](#footnote-1) in different versions. However, the most standard version nowadays, dates back to the final decades of the 15th century[[2]](#footnote-2). In the early years of the game, special moves such as castling or pawn opening double moves did not yet exist, but were later introduced into the game, which saw different rules (such as letting either white or black move first) but preserved the essence until the 19th century, when all of the current rules were finally fixated.

Some of these special rules increase the game complexity. Their main purpose is avoiding certain loopholes, such as the possibility of matches that would last for an infinite number of moves. While being greatly useful to enhance the possibilities and strategies in human vs human matches, they do make it a lot harder to develop artificial intelligence engines suitable for this game, as well as making it more complex to evaluate instances of the board, if all special rules are taken into account.

The main rules to take into account are Castling, En-Passant, Pawn Promotion, draw by Stalemate (formerly considered to be a win condition and still argued by some[[3]](#footnote-3)), Fifty-Move rule and Threefold repetition of position.

## 2.1.2 Past attempts to make a chess AI

Chess is notorious for being a very computationally intensive game, having exponential complexity in n (on an board) to find an optimal strategy[[4]](#footnote-4). This means that for a given starting board, there is an estimate of possible different games of chess that could be legally played[[5]](#footnote-5). This number is known as *Shannon Number* and it was an attempt made by *Claude Shannon* to demonstrate that if someone were to create a computer that played all possible chess games until an end in order to make a decision on the move to make next, this computer would never be able to make a move, therefore suggesting that brute-force approaches to creating a chess engine should be abandoned.

# 2.2 Technical research

Given the complexity above mentioned, creating a chess engine requires a vast knowledge of computational complexity, the programming language of choice and its libraries and different computational techniques and approaches to the game (game theory). The following sections will cover the research

## 2.2.1 Algorithms

### 2.2.1.1 MinMax

### 2.2.1.2 NegMax

Mention that it has not been used in the project

### 2.2.1.3 Alpha Beta Pruning

### 2.2.1.4 Monte Carlo Tree Search

Mention that it has not been used in the project

### 2.2.1.5 Neural Networks

Mention that it has not been used in the project

## 2.2.2 Design Patterns

## 2.2.3 Software Engineering Practices

### 2.2.3.1 Documentation

### 2.2.3.2 Coding

### 2.2.3.3 Debugging

### 2.2.3.4 Testing

#### 2.2.3.4.1 Unit Tests

#### 2.2.3.4.2 Alpha Releases and Testers

#### 2.2.3.4.3 Nim

Only objective research on this game.

#### 2.2.3.4.4 Chess End Games

## 2.2.4 Game API

### 2.2.4.1 Description

### 2.2.4.2 Original API

### 2.2.4.3 Features

### 2.2.4.4 Changes

## 2.2.5 Other considerations

Mention Go or Focus (maybe move this section somewhere else?)

Chapter 3: Software Requirements

# 3.1 Functional Requirements

# 3.2 Non-functional Requirements

Chapter 4: Feasibility Study and Choices Taken

# 4.1 Software choices

## 4.1.1 Eclipse JEE

## 4.1.2 GitHub

### 4.1.2.1 Motivation

### 4.1.2.2 Folder structure

## 4.1.3 Google Drive

# 4.2 Model choices

# 4.3 Other choices

Mention rules and variants of chess.

# 4.4 Time and resources estimation

# 4.5 Feasibility of the project

# 4.6 Testing choices

# 4.7 User interface

Chapter 5: Design, Architecture and Diagrams

# 5.1 Overview

# 5.2 Class Diagrams

# 5.3 Sequence Diagrams

# 5.4 Other design information

Design patterns used

Chapter 6: Components & Implementation

# 6.1 Most relevant classes

# 6.2 User Interface

Chapter 7: Testing

# 7.1 Unit tests

# 7.2 toString() debugging

# 7.3 Nim

# 7.4 Chess end-games

# 7.5 Playing against human subjects

Chapter 8: Project Evaluation

# 8.1 Overview

# 8.2 Strengths

# 8.3 Problems encountered

## 8.3.1 Critical problems

## 8.3.2 Other problems and solutions

# 8.4 Potential improvements

Things that could have been done better

Chapter 9: Conclusions

# 9.1 Overview

# 9.2 Lessons learnt

# 9.3 Should (not) have done

# 9.4 Personal opinion

Chapter 10: Future Work

# 10.1 State of the project

Why it will continue

# 10.2 Future new features

# 10.3 Improvements to existing features

# 10.4 Expectations

Chapter 11: Bibliography

1. Niklesh Kumar Jain (2014), “Hindi and the Origins of Chess”, Chessbase.com. [↑](#footnote-ref-1)
2. Hooper & Whyld (1992), “The Oxford Companion to Chess”, pp. 173-175. [↑](#footnote-ref-2)
3. Larry Kaufman (Sept 2009): “Calling stalemate a draw is completely illogical, since it represents the ultimate zugzwang, where any move would get your king taken.” [↑](#footnote-ref-3)
4. Fraenkel & Lichtenstein (1981), “Computing a perfect strategy for nxn chess requires time exponential in n”, Journal of Combinatorial Theory, Series A (31): pp. 199-214. [↑](#footnote-ref-4)
5. Claude Shannon (1950), “Programming a Computer for Playing Chess”, Philosophical Magazine. 41 (314) [↑](#footnote-ref-5)