DGA Game Concepts Guide

Zhanat S. Skokbayev

This section is currently a draft, and is subject to change.

Version: 0.1
Date: 10.04.2025
Description: The first publication.

Contents at a Glance

This section is currently a draft, and is subject to change.

Table of Contents

4. Product Concept

5. Systems Concept

Table of Figures

Document History

This section is currently a draft, and is subject to change.

Version	Author	Date	Description
0.0.1	Zhanat S. Skokbayev	19.09.2018	The first draft.
0.0.2	Zhanat S. Skokbayev	25.04.2019	The first presentation.
0.1	Zhanat S. Skokbayev	10.04.2025	The first publication.

Preface

This section is currently a draft, and is subject to change.

The DGA Game Concepts Guide ('the Guide') specifies design concepts and guidelines of a game called 'DGA Distant Ground Attack' ('DGA', 'the DGA Game', or simply 'the Game).

This chapter introduces common topics related to the Guide.

Audience

The Guide is intended to be read by programmers, artists, testers, producers, managers, players, and everybody else involved in the Game's design, development, testing, and use.

Legal

Prerequisites

The DGA Game is based on the Java Platform and written in the Java programming language. DGA's software architecture is closely related to the Java Platform's technologies. If you are new to Java, spend some time getting up to speed on the language and platform; a good place to start is dev.java/learn.

Each topic in this tutorial provides some background information, but in general, we assume you have a basic knowledge of the technologies each Java Platform's feature works with. Another field important for understanding the Guide is game development. We assume that you have some basic understanding in these fields.

Related Documentation

For more related information, see the following documents of the Game:

- The DGA Game Architecture Guide provides all necessary information about DGA's systems, software, and network architectures.
- The DGA Game Design Guide details the current Guide and serves as a blueprint from which the Game is being built.

Conventions

Throughout this document, we use the following typographic conventions:

Convention	Meaning	Example
Boldface	Boldface type indicates a new term defined in text below or in the glossary. Also it marks graphical user interface elements associated with an action.	**Heaven** signifies night and day, cold and heat, times and seasons. From the **File** menu, choose **Open Project**.
Italic	Italic type indicates book titles, emphasis, terms in the text or placeholder variables for which you supply particular values.	Read Chapter 6 in the *User's Guide*. Do *not* save the file. The command to remove a file is `rm` *`filename`*.
`Monospace`	Monospace type indicates the names of files and directories, commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.	Edit your `.login` file. Use `ls -a` to list all files. *`machine_name`*`% you have mail.`
very important	Underlined type indicates texts of especial importance.	**The Commander** stands for the virtues of wisdom, sincerity, benevolence, courage and strictness.
(Book, page)	Letter code and number in round brackets indicates index of a text resource through the *Bibliography* and the page number(s) within the text.	(ECGG, p. 319) (TAOWG, pp. 27-29)

1. Introduction

This section is currently a draft, and is subject to change.

DGA is a free/libre and open-source, cross-platform, massively multiplayer online role-playing video game featuring weapons.

DGA implements the following concepts and models:

1. Free/Libre and Open Source Software
2. Free-To-Play Business Model
3. Fair-To-Play Conceptual Model
4. Free/Libre-To-Play Business and Development Model
5. Free/Libre-To-Game Business and Development Model

This chapter explains the listed concepts and models in detail.

1.1. Free/Libre and Open Source Software

Free/Libre and Open Source Software (F/LOSS) is the indisputable technological mainstream of modernity.

The concept of Free/Libre Software was pioneered by Richard Matthew Stallman who founded the Free Software Foundation (FSF) to promote computer user freedom and support the free/libre software movement.

Richard Stallman wrote and published the Free Software Definition, which provides a concise and distinguished explanation of what free/libre software is and how it must be implemented.

The open-source software movement was promoted by Bruce Perens and Eric Steven Raymond who founded an organization called Open Source Initiative (OSI) to educate about and advocate for the benefits of open-source software.

Bruce Perens and Eric Raymond drafted a set of open-source guidelines, which were adopted by the OSI as the Open Source Definition. The definition distinguishes criteria and requirements an open-source program must correspond to and serves as the most common standard for open-source software.

The figure below illustrates the notions of closed-source, free/libre, and open-source software, and shows how they are related.

A programmer writes a computer program in a programming language comfortable for humans. At this stage the program is called 'a source code program' or 'a source program' or simply 'source code'. When the source program is finished, the programmer compiles/translates source code into a computer program comfortable for execution by computers (computing machines). This program is called 'a compiled program' or 'object code'. The compiled program is the final result of the programmer's work and suitable for distribution among users.

Proprietary Software (Closed-Source Software) requires that only the compiled program is transferred to users. The program's source code is the private property of the programmer and/or the company the programmer is working for. An improved variation of proprietary software assumes that users can obtain the program's source code but under the terms of some proprietary, non-free licence. For that reason, proprietary software is also called *non-free software*.

Free/Libre Software requires that both the compiled program and its source code must be accessible to users who, in case of need, can recompile the source code by themselves with or without modification. Moreover, a user who received the program with its source code can't limit further distribution to other users, even when the user has modified the program; in this case, the user must distribute the modified program both in the compiled form and with its source code.

Open-Source Software implies that the computer program can be distributed as the compiled program and its source code, but both forms are not strictly required. A user can modify an open-source program and then distribute this modified program only in a compiled form without providing the modified source code. This opportunity may be used to limit the rights of other users but in some cases can be considered as noncritical or compensated by other benefits. The readiness to sacrifice the user rights to some practical benefits differs Open-Source Software from Free/Libre Software.

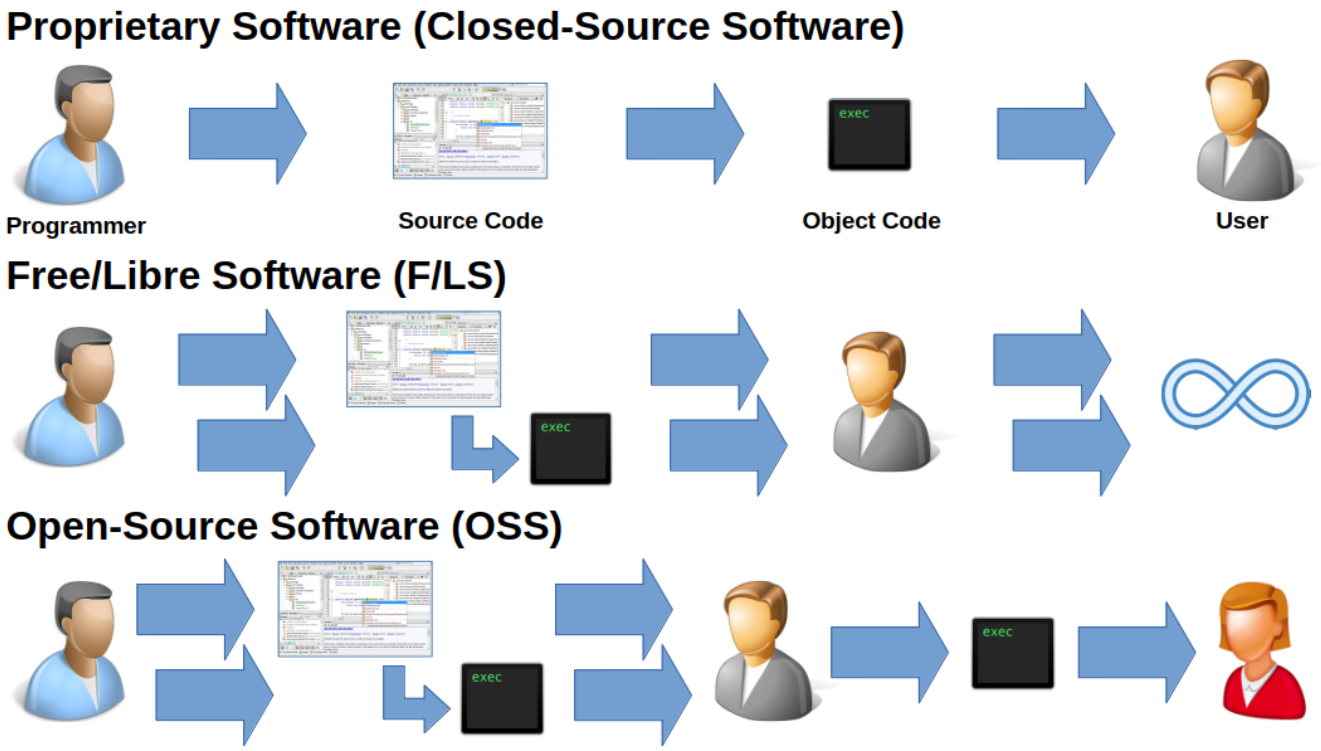

Figure 1.1. Free/Libre, Open-Source, and Closed-Source Software

Free/Libre Software and Open-Source Software are implemented with their software licences.

The Free Software Foundation maintains a set of licensing guidelines, which provides a non-exhaustive list of free-software licences to be used for development and distribution of free/libre software.

The Open Source Initiative maintains its proper licence review process, which defines a list of approved open-source licences that comply with the Open Source Definition.

DGA is fully and unconditionally committed to the ideas and ideals of Free/Libre Software. We strongly believe in computer user freedom and the rights of software users. Only for some limited cases, when it is

appropriate, we let in the possibility of developing and distributing our programs as Open-Source Software. In every case we strive to implement the most free/libre conditions possible. Accordingly, the DGA Game supports fully and unconditionally both definitions as follows:

- The Free Software Definition
- The Open Source Definition

DGA and all its components as well as their source codes are developed and distributed under the terms of the GNU General Public Licenses (GNU GPLs) as follows:

- The GNU General Public License
- The GNU Affero General Public License
- The GNU Lesser General Public License

Some components of the DGA Game, in case of necessity and appropriateness, can be developed and distributed under the terms of the Apache License as it is defined and maintained by The Apache Software Foundation:

- The Apache License 2.0

1.2. Free-To-Play Business Model

The Free-To-Play (FTP, F2P) Business Model is an economic model that become extremely popular in gaming. Free-to-play model is also commonly referred to as *free-to-start model* due to not being entirely free of charge.

The Free-To-Play Business Model is a strategy employed by businesses, particularly in the digital gaming and mobile app industries where consumers can access and use a product or service without an initial purchase cost. Instead of charging upfront fees, the business generates revenue through optional in-app purchases, microtransactions, advertising, or premium content. This model aims to attract a large user base by eliminating barriers to entry and then monetize a portion of those users through various revenue streams. (FourWeekMBA)

The business model is based on the following key concepts:

- Zero Entry Cost (Zero-Entry Barrier);
- In-App Purchases (IAPs or microtransactions);
- Ads and Sponsorships (Income from advertisers, sponsors, and partners);
- Virtual Currency (In-game digital currency);
- Premium Content (Premium features unlocked through one-time purchases or subscriptions).

The attributes below characterise free-to-play:

- Mass User Acquisition (Acquiring a large user base);
- Continuous Engagement (Ongoing user engagement through content updates, events, and challenges);
- Monetisation Strategies (Diversification of revenue streams);

Data-Driven Optimisation (Data analytics used to refine business offerings and pricing strategies).

The free-to-play model offers several benefits and considerations:

- Massive User Base (A massive user base increases the potential for monetisation);
- Flexible Monetisation (Flexibility and the ability to cater to various user preferences);
- Market Competition (Free-to-play businesses must stand out and continually innovate to succeed);
- User Trust and Fairness (Maintaining user trust and fairness is crucial).

It is worth noting that user trust and fairness are considered as essential for the Free-To-Play Business Model in gaming. Transparency in pricing, fairness in gameplay, and responsible ad practices are vital for building and retaining a loyal user base. Users should feel that optional purchases are genuinely valuable and enhance their experience. (FourWeekMBA)

The next figure shows the graphical depiction of the free-to-play model, which is based on FourWeekMBA's Model and extended with technical aspects.

Iterative Game Development

While developing a game, a game studio realises an iterative game design cycle, which comprises five consecutive stages:

1. Design (creating, testing, and refining game ideas in the service of an overall vision for the game).
2. Implementation (building the game to refine the design).
3. Testing (checking the game design, functionality, and usability).
4. Playtesting (having players play the game while it is being designed and reported back on their experience).
5. Evaluation (making estimations allowing for continuous improvement based on playtesting and feedback).

The classic Free-To-Play Business Model assumes that the game studio makes the game as closed-source software, and therefore the development cycle is inaccessible for people outside the game studio: players, streamers, creators, developers, etc. These people form the game's ecosystem and are involved into the game's distribution cycle. For that reason, free-to-play in itself is often considered as a distribution model, rather than a complete business model. In reality, the way the game gets distributed influences the whole business model for the game studio.

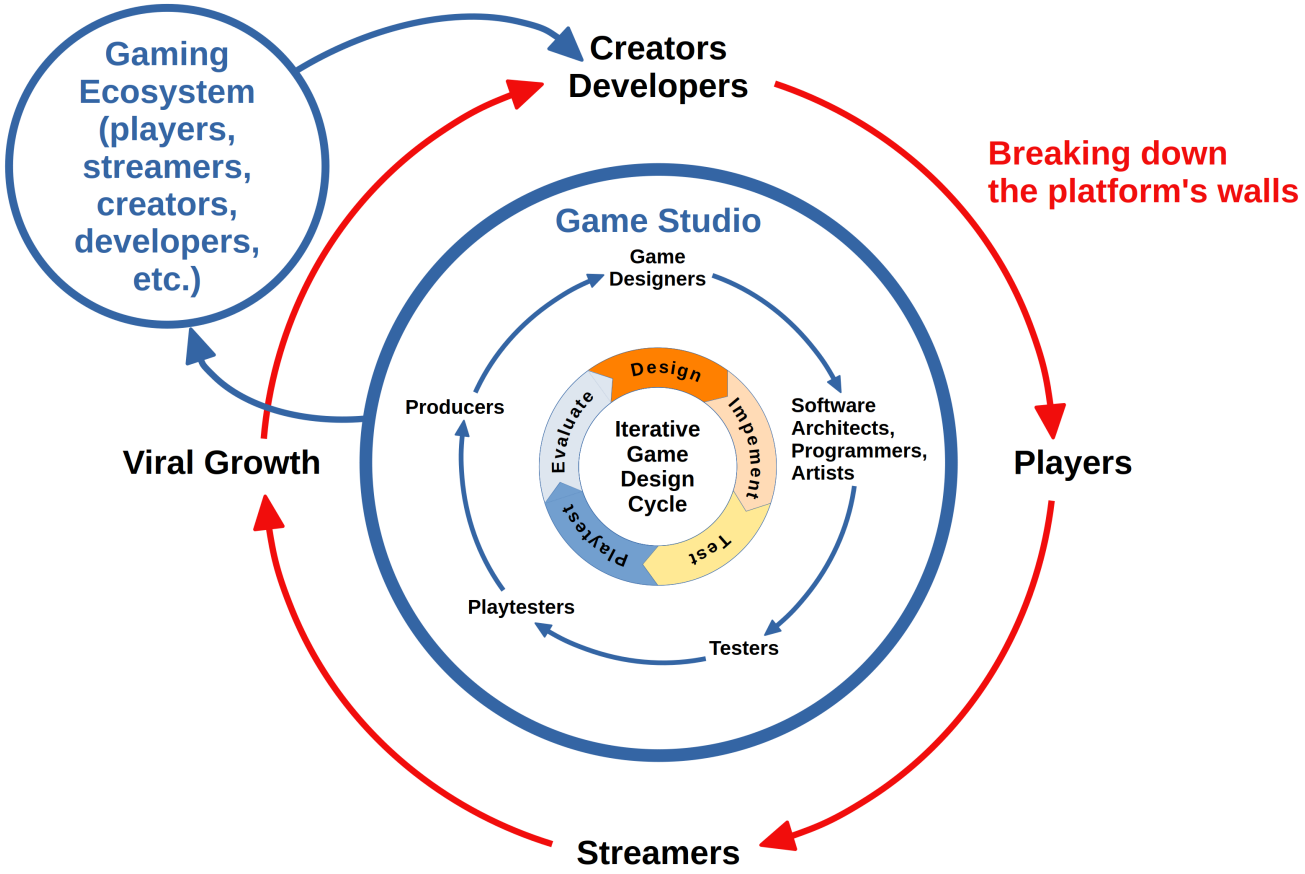

Figure 1.2. Free-To-Play Business and Development Model

The image below shows the submodels and components of the free-to-play model and their interrelations according to FourWeekMBA's Model.

The Free-To-Play Business Model comprises the following submodels:

- Ecosystem (Distribution Model);
- User Value (Value Model);
- Revenue Streams (Financial Model);
- Products and Services (Technological Model).

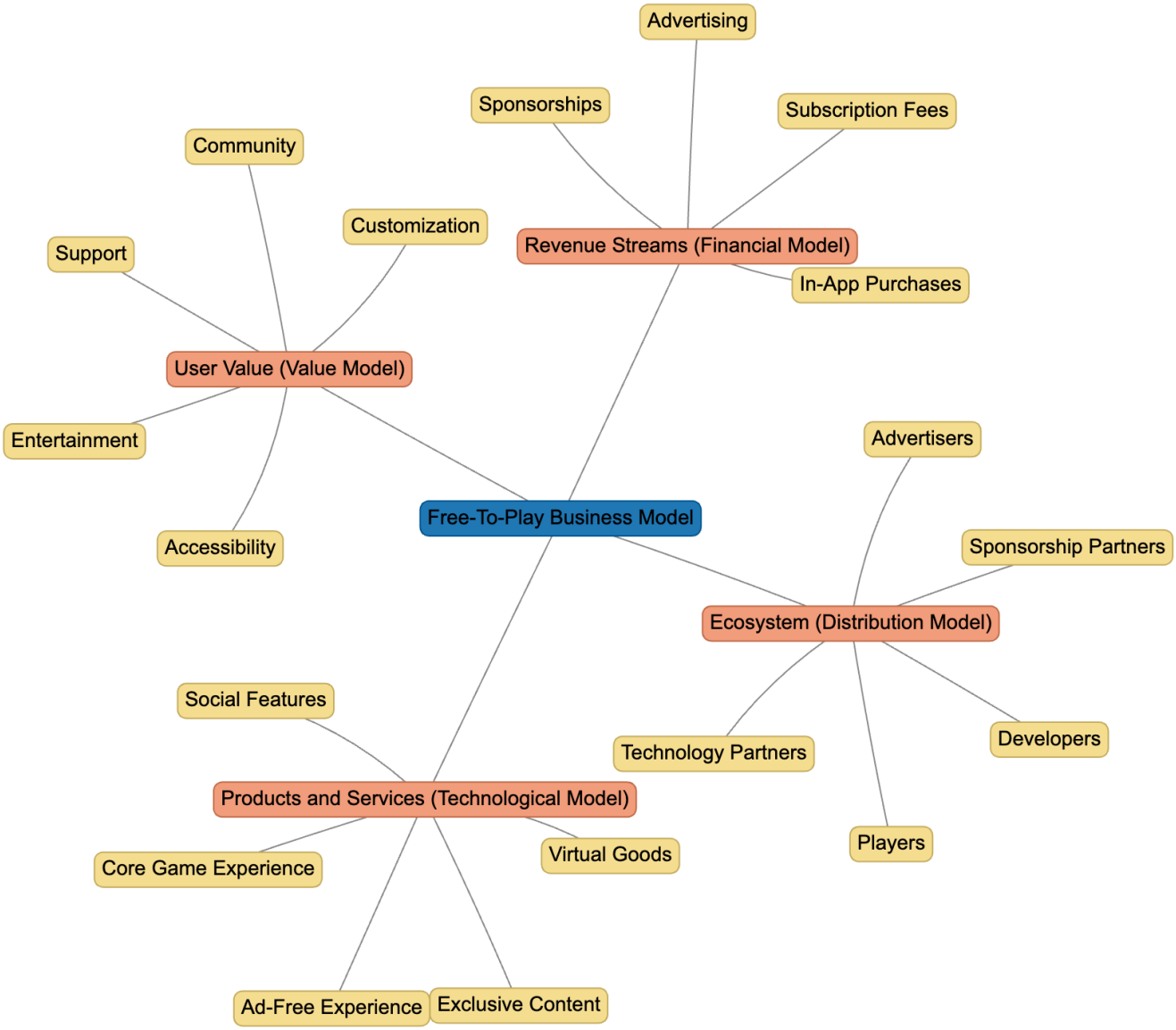

Figure 1.3. Free-To-Play Business Model's Submodels and Components (FourWeekMBA)

The Free-To-Win (FTW, F2W) Business Model is a variation of the free-to-play model in gaming. Free-to-win affirms that a player has equal opportunities with all other players and can gain an advantage over them only by personal skills.

In fact, the free-to-win model can't fulfill its promises because of the lack of transparency: players have no means to verify neither other players don't have unfair advantages over them nor the game server gives preferences to some categories of players. Nowadays, both the free-to-play and free-to-win models are considered as varieties of the freemium model.

The Freemium Business Model is an economic model that offers basic services or a product for free while charging for premium features or additional content. It aims to attract a large user base with free offerings and then convert a portion of those users into paying customers by providing enhanced features or value-added services. This model is commonly used in software, gaming, and digital services industries to acquire users and generate revenue through upselling or subscription models. (FourWeekMBA)

By their closed-source nature, freemium games very quickly start to follow the pay-to-win/pay-to-earn model, which definitely assumes inequality of players when paying players have advantage over their non-paying peers. In case of subscription, the pay-to-win model works as pay-to-play games.

1.3. Fair-To-Play Conceptual Model

The Fair-To-Play (FRTP, FR2P) Model is a strategy employed by institutional and individual developers in the digital gaming and mobile app industries where gamers can verify whether the game is still a fair play or not.

The fair-to-play model is introduced by The FLEISS Software Foundation and aims to compensate deficiencies of the business and development models existing in the digital gaming industry. The key deficiency is the unfair interpretation of game cheating affirming that only players can cheat but not game developers and owners. Moreover, game developers are increasingly embedding into games anti-cheat systems, which are becoming more and more intrusive. These systems already operate at the kernel level of user operating systems, which means they have deep access to players' computers. In other words, players are always considered as potential cheaters and must prove their trustworthiness. At the same time, game developers and owners are always considered as trusted and their credibility is not in question.

The common argument explaining why game developers and owners are not interested in cheating assumes that an equal starting position for all players allows the game developers to create a more balanced game. Such a game provides a more positive experience to its players and thereby attracts more users who bring more money to the game developers and owners. Stated another way, there are compelling business reasons to prevent game developers and owners from cheating.

Cheating is the Result of Game Balance's Issues

Cheating is usually treated as issues of game balance. The following quote perfectly illustrates this approach:

> *Putting the participants in an equal starting position is a hard problem to solve, which is tackled by game balancing in game design (Adams 2014, pp. 403–405). A balanced game is fair, meaning that all players have an equal chance of winning at the start, and it should be appropriately challenging (i.e., not too hard nor too easy) for the players. The skill in the actual task of the player, rather than in the game created on top of the tasks solved through gamification, should be the most important factor in determining the player's success. (ECGG, p. 815)*
>
> — *Encyclopedia of Computer Graphics and Games*
> *Ed. by Newton Lee (Springer 2024)*

In reality, the game studios and their owners quite often demonstrate that their business interests are much more complicated and contradict the interests of their proper users. Gamers shall not completely depend on the game studios' goodwill towards them. Gamers shall have effective technical means to accomplish

independent verification whether a game still abides by the principles of fairness, equality, justice, honesty, and integrity to its players or already not.

The next figure shows the graphical depiction of the fair-to-play model, which is based on Michael Sellers' Systemic Model of Games.

A game developed by a game studio has its game+player system containing the game's interactive core gameplay loop. The loop is activated when a player provides input to the game via his/her behaviour, which changes the game's state. The game processes this input and provides feedback responses that are input for the player, changing his/her internal state. These interactions create reciprocating loops that is the essence of interactivity.

While a gamer plays the game, i.e. becoming a player, the game studio uses its anti-cheating system to check the player for fair play, i.e. check that the player doesn't cheat. In an online game the game studio can use the anti-cheating system not only on the player's side (within the game's client program) but also on the server side (within the game's servers). That is how the games work today.

The fair-to-play model requires that the gamer must have technical means to check the game for fair play, i.e. the player of the game must have the same rights as the game studio does. The game studio must provide these means to the gamers. In this case the game can be recognised as fair to play.

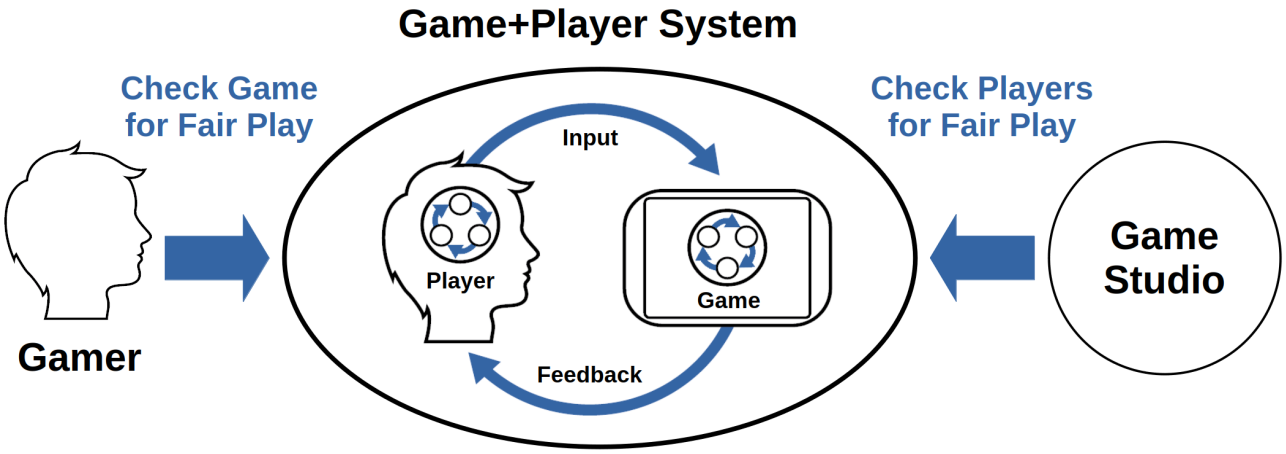

Figure 1.4. Fair-To-Play Conceptual Model

Fair-to-play in itself is a conceptual model, which can be implemented in different business and development models. So, 'game studio' can be a company, non-commercial organisation, team, or individual developer.

1.4. Free/Libre-To-Play Business and Development Model

The Free/Libre-To-Play (FLTP, FL2P) and Free/Libre-To-Game (FLTG, FL2G) Models are business and development models introduced by The FLEISS Software Foundation as implementations of the Fair-To-Play Conceptual Model.

The free/libre-to-play model implements the principles of fair-to-play by following the way of Free/Libre and Open Source Software.

The F/LOSS Way

Think over the quote below and apply its statements to the contemporary computer games, especially massively multiplayer online ones.

> *Proprietary software, also called nonfree software, means software that doesn't respect users' freedom and community. A proprietary program puts its developer or owner in a position of power over its users. This power is in itself an injustice.*
>
> *The point of this directory is to show by examples that the initial injustice of proprietary software often leads to further injustices: malicious functionalities.*
>
> *Power corrupts; the proprietary program's developer is tempted to design the program to mistreat its users. (Software designed to function in a way that mistreats the user is called malware.) Of course, the developer usually does not do this out of malice, but rather to profit more at the users' expense. That does not make it any less nasty or more legitimate.*
>
> *Yielding to that temptation has become ever more frequent; nowadays it is standard practice. Modern proprietary software is typically an opportunity to be tricked, harmed, bullied or swindled. (GNU.org)*
>
> — *The Free Software Foundation*
> *Philosophy of the GNU Project: Proprietary Software Is Often Malware*

F/LOSS is the most consistent approach to ensure fair play in digital games.

The free/libre-to-play model is based on the following approaches and models:

1. Free/Libre and Open Source Software as the strategic approach in software development.
2. Free-To-Play Business Model as the basic business model for further improvement.
3. Fair-To-Play Conceptual Model as the strategic approach in gaming.
4. Iterative Systemic Approach in game design and development.

The figure below shows the graphical depiction of free/libre-to-play.

A game studio implements a game within an iterative development cycle. Since the game is a free/libre/open-source software program, the game's community is an integral part of the game's ecosystem.

The game studio and game community form the common ecosystem of the game and are inseparable. Employees working at the game studio, the game's users, and the game community's participators,

contributors, and leaders can be the same persons, which activities are interrelated and interconnected. This interconnection may create some conflicts of interest, when a person is an employee of the game studio and at the same time an active member of the game community, but these conflicts can be effectively arranged.

The working process conducted at the game studio and various activities going on within the game community constitute the game's business and development workflow.

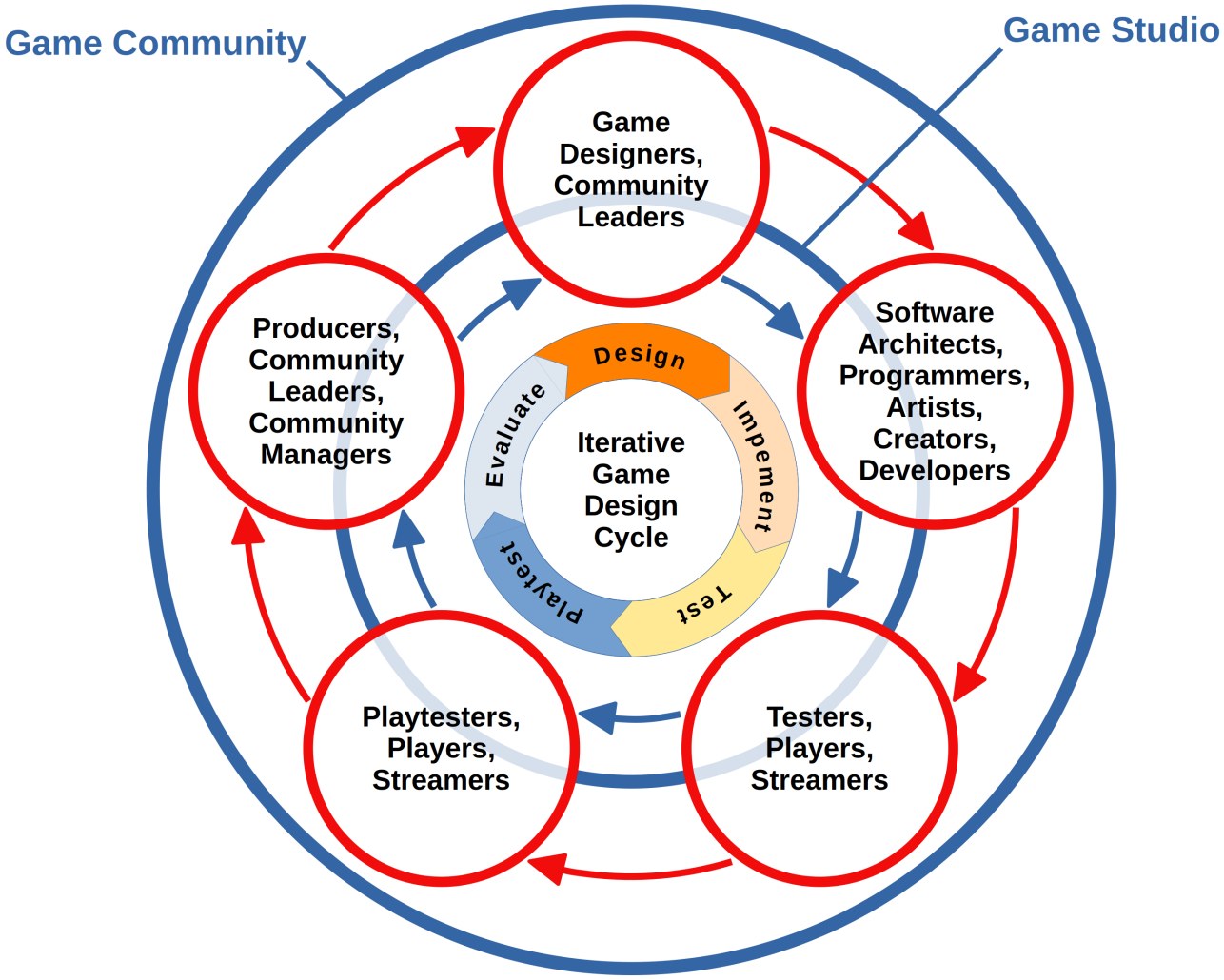

Figure 1.5. Free/Libre-To-Play Business and Development Model

1.5. Free/Libre-To-Game Business and Development Model

The Free/Libre-To-Game Business and Development Model is an advanced version of free/libre-to-play and serves as a superior implementation of the fair-to-play concept.

The free/libre-to-game model takes into account the architecture of multitier game systems and requires that not only a game's client but all the game's server components must be free/libre/open-source software.

When a game can be played in a single-player mode only, the free/libre-to-play model is sufficient to ensure the game is fair-to-play. Such a game has only a client program installed on the gamer's computer and does not require interaction with remote servers to be played.

Whereas the game is a multiplayer one and requires continuous interactions with remote servers, the game client's source code accessible to the gamers is not sufficient to ensure fair-to-play, since the game servers

can still remain closed-source software. This situation is not adequately covered by the free/libre-to-play model.

Here free/libre-to-game comes into action. The model stipulates that not only the game's client installed on the player's side and sufficient to play the game, but also the game's servers and other components and services necessary for the game to be played — all of them must be free/libre/open-source software. The gamers must have opportunity to rebuild the game's client and servers from their source codes and reproduce the game's infrastructure to verify its fairness, equality, justice, honesty, and integrity.

The figure shows the concepts and models free/libre-to-play and free/libre-to-game are based on and interrelated.

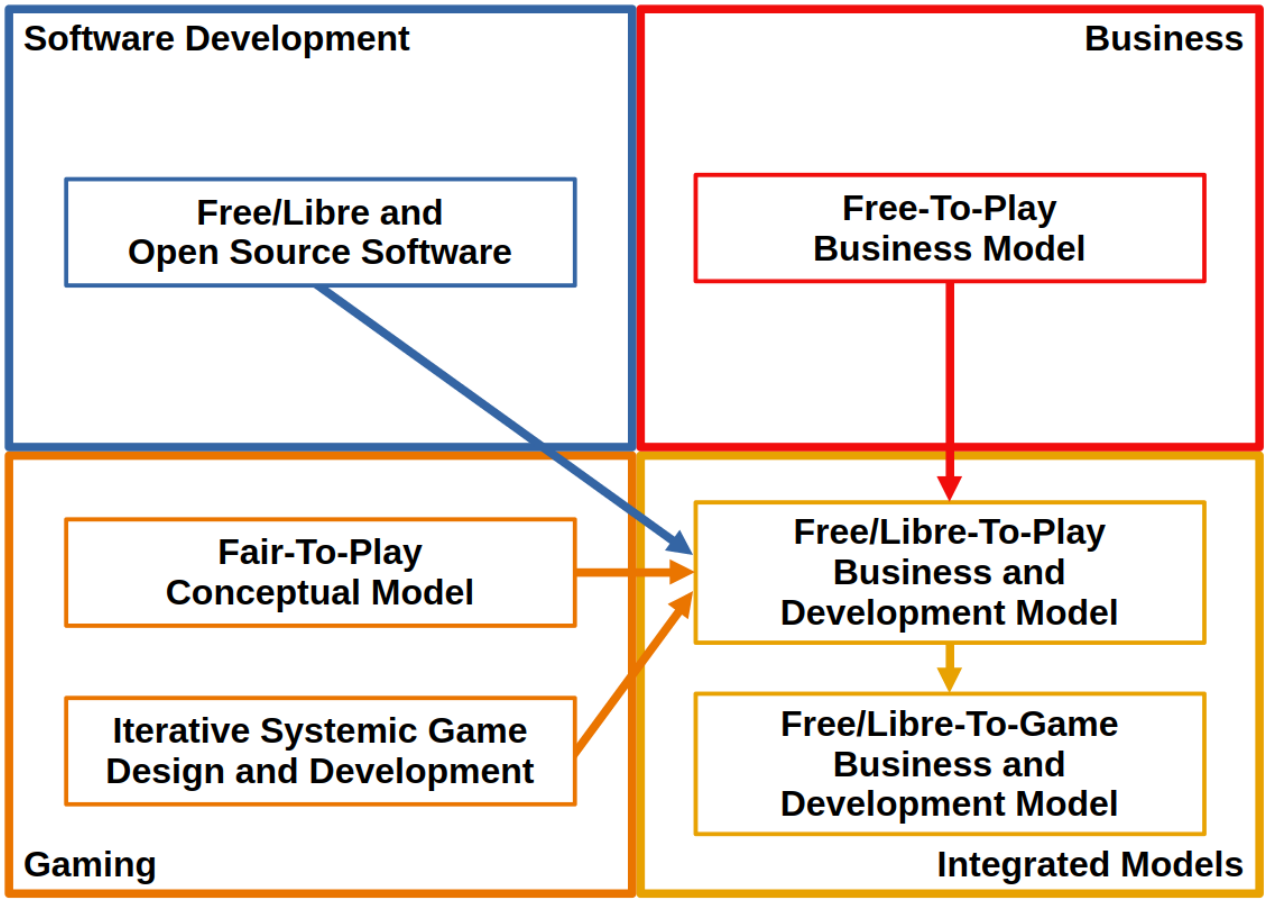

Figure 1.6. Free/Libre-To-Play and Free/Libre-To-Game Models

The free/libre-to-game model is the basis for design, development, and distribution of the DGA Game.

2. Game Concept

This section is currently a draft, and is subject to change.

DGA aims to introduce, engage, and immerse players into the art of war and warfare by featuring them with weapons. Master Sun — the longest existing and most widely studied military classic in human history — stated two and a half thousand years ago the canon of this art, which has remained a prominent force in the military arts until nowadays. (TAOWA, p. 9)

This chapter outlines the key gaming concepts of the DGA Game.

> *The art of war, then, is governed by five constant factors, to be taken into account in one's deliberations, when seeking to determine the conditions obtaining in the field.*
>
> *These are: (1) The Moral Law; (2) Heaven; (3) Earth; (4) The Commander; (5) Method and discipline.*
>
> ---
>
> ***The Moral Law** causes the people to be in complete accord with their ruler, so that they will follow him regardless of their lives, undismayed by any danger.*
>
> ***Heaven** signifies night and day, cold and heat, times and seasons.*
>
> ***Earth** comprises distances, great and small; danger and security; open ground and narrow passes; the chances of life and death.*
>
> ***The Commander** stands for the virtues of wisdom, sincerity, benevolence, courage and strictness.*
>
> *By **Method and discipline** are to be understood the marshalling of the army in its proper subdivisions, the gradations of rank among the officers, the maintenance of roads by which supplies may reach the army, and the control of military expenditure.*
>
> *These five heads should be familiar to every general: he who knows them will be victorious; he who knows them not will fail.*
>
> ---
>
> *Therefore, in your deliberations, when seeking to determine the military conditions, let them be made the basis of a comparison, in this wise:*

1. Which of the two sovereigns is imbued with the Moral Law?

2. Which of the two generals has most ability?

3. With whom lie the advantages derived from Heaven and Earth?

4. On which side is discipline most rigorously enforced?

5. Which army is the stronger?

6. On which side are officers and men more highly trained?

7. In which army is there the greater constancy both in reward and punishment?

By means of these seven considerations I can forecast victory or defeat.

The general that hearkens to my counsel and acts upon it, will conquer: — let such a one be retained in command! The general that hearkens not to my counsel nor acts upon it, will suffer defeat: — let such a one be dismissed!

While heeding the profit of my counsel, avail yourself also of any helpful circumstances over and beyond the ordinary rules.

According as circumstances are favourable, one should modify one's plans.

All warfare is based on deception. *(TAOWG, pp. 27-29)*

— Sun Tzu
The Art of War: Chapter I. Laying Plans (5th century BCE)

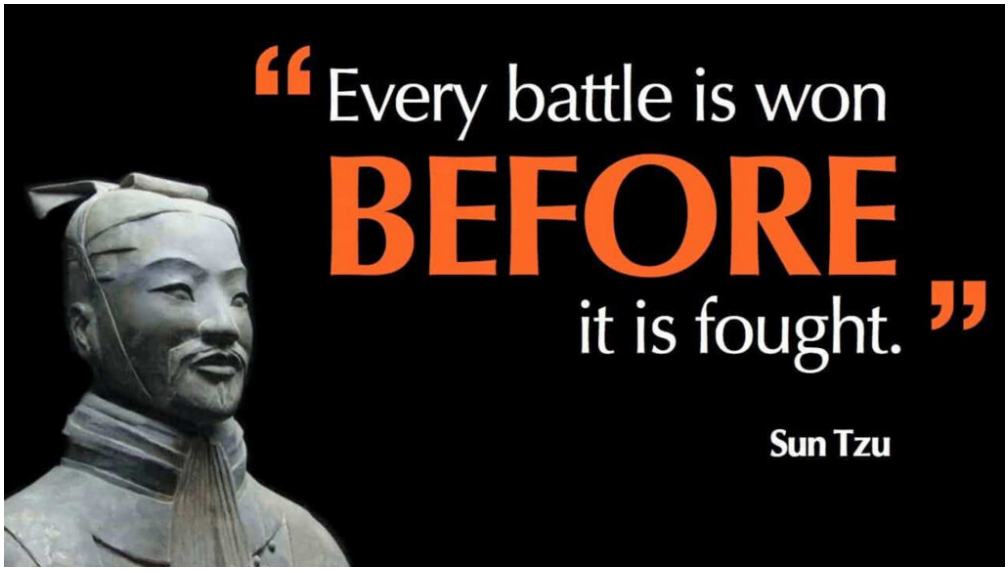

Figure 2.1. Master Sun

2.1. Game Title

DGA — Distant Ground Attack.

https://distantgroundattack.org

2.2. Game Logo

Figure 2.2. DGA Game's Logo

2.3. Game Motto

For those who game…

2.4. Game Concept Statement

DGA aims to turn a player into a commander taking control over a selected historical weapon unit, single or combined, which, as in the art of war, rigorously goes into a battle on a historic map of choice.

The player manages the unit's position, movement, firing, and communications with allied players from the third/first-person view. Different types of weaponry can simultaneously participate in a single battle session, which may have limitations on time and amount of players. The first types of weapons being realised are mechanised/motorised land combat vehicles and guns, dating from the time of World War I up to the Cold War, with giving preference to World War II.

The player chooses a weapon unit between different types of weaponry. The choice is bound to the type's limitations and player's experience only. The player goes into battle with the chosen weapon in a team that can be composed randomly as well as by agreement. The goal is to put out of action all the units of the opposite team or to accomplish some victorious condition (capture the flag, maintain a position against enemies, make the opposite team surrender, etc.). The player's progression depends only on how well he/she can utilise potential of the chosen weapon unit and coordinate his/her actions with other players in the team.

2.5. The One Question

DGA's gameplay has the following One Question: (AGDS, p. 196)

Does it intelligent, sincere, humane, fair, and just?

> *Reliance on intelligence alone results in rebelliousness. Exercise of humaneness alone results in weakness. Fixation on trust results in folly. Dependence on the strength of courage results in violence. Excessive sternness of command results in cruelty. When one has all five virtues together, each appropriate to its function, then one can be a military leader. (TAOWC, pp. 49-50)*
>
> — *Jia Lin on Sun Tzu's The Art of War: Strategic Assessments (Tang Dynasty 618–906 CE)*

Intelligent as in wisdom.

Sincere as in trustworthiness.

Humane as in humaneness.

Fair as in fair play.

Just as in authenticity and accuracy.

2.6. Game Mood

DGA's gameplay proposes to its players the following game mood:

Imperil yourself!

> *So it is said that if you know others and know yourself, you will not be imperiled in a hundred battles; if you do not know others but know yourself, you win one and lose one; if you do not know others and do not know yourself, you will be imperiled in every single battle. (TAOWC, p. 91)*
>
> — *Sun Tzu*
> *The Art of War: Chapter III. Planning a Siege (5th century BCE)*

> *When you know others, then you are able to attack them. When you know yourself, you are able to protect yourself. Attack is the time for defense, defense is a strategy of attack. If you know this, you will not be in danger even if you fight a*

hundred battles.
When you only know yourself, this means guarding your energy and waiting. This is why knowing defense but not offense means half victory and half defeat.
When you know neither the arts of defense nor the arts of attack, you will lose in battle. (TAOWC, pp. 91-92)

— Zhang Yu on Sun Tzu's The Art of War: Planning a Siege
(Sung Dynasty 960–1278 CE)

2.7. Genres

DGA is a sophisticated, visually stunning, free/libre-to-game, cross-platform by design, massively multiplayer online role-playing game, which combines:

- third/first-person action battle sessions of historical weapon units controlled by players on historic battlefields changing in times, seasons, weather, events, etc.;
- with strategic open-ended modifications of these weapon units;
- and social interactions and easy-to-learn casual aspects.

DGA is addressed to casual and core gamers. The game strives to provide players with a transparent, fair-to-play balance between historic authenticity and fast-paced shooter gameplay.

2.8. Target Audience

DGA aims to address casual and core gamers of all genders and ages who are already interested in the history of weapons, military history, and mechanics, but also those whose interest in these fields of knowledge could be piqued as well.

DGA is intended to meet all the gamers' areas of motivation and incentives as follows:

1. Action, assuming destruction and fast-paced, exciting gameplay.
2. Social, including both community and competition (which are not mutually exclusive).
3. Mastery, providing difficult challenges and long-term strategies.
4. Achievement, while completing all missions and becoming powerful.
5. Immersion, of being someone else and experiencing an elaborate story.
6. Creativity, expressing yourself via design, fantasy, crafting, and customisation, as well as tinkering and exploring. (AGDS, pp. 200-201)

DGA promises to avoid any depiction of blood, bodies, and other signs of war related to humans. However, the game contains depictions of non-realistic, non-detailed violence.

The desired ratings DGA aims to achieve are "PEGI's 7 with the descriptor Violence" and "ESRB's Everyone".

2.9. Target Platforms

DGA pursues the Cross-Platform by Design approach, which seeks bridging the gap between different devices and platforms. This approach is based on the idea to implement technologies designed to adapt and respond to different operating systems and screen sizes, ensuring that the user's experience is consistent but also engaging and efficient, no matter the device in use. (Medium)

DGA is designed and developed on and for the Java Platform — the complete and full-scaled cross-platform technology implemented on a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers.

DGA targets all platforms where Java is working on:

- Primarily the PC market with its, first of all, free/libre/open-source operating systems (GNU/Linux, BSD, HarmonyOS) and closed-source proprietary ones (Microsoft Windows, macOS);
- Then mobile platforms with their free/libre/open-source operating systems (Android, HarmonyOS) and closed-source proprietary ones (iOS, iPadOS);
- Then the game console market (Sony PlayStation, Microsoft Xbox, Nintendo Switch).

2.10. Game Flow

While playing the DGA Game, a player becomes a commander taking control over a selected historical weapon unit and goes into a battle on a historic map of choice.

DGA implements the fair-to-play principle in the field of computer games modeling historical weapons and asserts that such a weapon must be modeled in the game authentically, correctly, and without being changed for any reason other than adjustments in favor of historical accuracy.

DGA treats as unfair and inappropriate the practice of some modern warfare-themed games in which weapon units are presented as modeled to closely resemble their real-life counterparts, but several their parameters can be simplified or modified to fit game mechanics and better gameplay. This practice leads to everlasting uppings and nerfings of the weapon units, which historical accuracy afterwards players can't even estimate. Moreover, the owners of some games systematically exploit historical accuracy and authenticity to compensate game bugs, misinformation, bad balance, and declining revenues from users. Finally, it ends in disrespect and injustice to players.

The closed-source proprietary nature of computer games is the real cause of this regrettable situation in gaming. The games' owners are in the position of power over their players and once too often do not hesitate this power to misuse. For this reason, DGA is free/libre-to-game.

DGA additionally affirms that parameters of historical weapons must be hard-coded in the game. Eventually, the modeled weapons are historical and their parameters usually are well-known, described in multiple

sources, and can't be changed due to their historical nature. The only reason why the weapons in the game can be remodeled is to improve their historical accuracy.

Striving to find a fair-to-play balance between historical accuracy, authenticity, and fast-paced shooter gameplay, DGA addresses distinct categories of players and implements the following three battle modes, which establish different proportions between accuracy/authenticity and shooting gameplay:

1. Struggle Battle Mode (the most fast-paced and accurately authentic).
2. Realistic Battle Mode (reasonably fast-paced and accurately authentic).
3. Simulator Battle Mode (the most authentic and reasonably fast-paced).

DGA is conceived to feature players with an open-ended, unlimited experience in modification and customisation of the game's weapons and characters strictly according to their historical authenticity and accuracy. DGA aims to support not only production modifications and customisations but also field ones made by soldiers and service crews on the front line.

2.11. Look and Feel

DGA is a third-person shooter with a facility to switch the camera perspective to a first-person view of the weapon unit's commander and other crew members.

Consequently, DGA features the following camera perspectives:

- The weapon unit's third-person perspective.
- The weapon unit's targeted (first-person) perspective.
- The crew-member's third-person perspective.
- The crew-member's targeted (first-person) perspective.

DGA's pacing, visuals, and level of action aim to be at the same level with the best of action warfare-themed shooters.

2.12. Game Balance

DGA has a complex game balance. It is arisen out of the following factors:

- Historical authenticity and accuracy require that the weapons modeled in the game strictly match their historical counterparts, which had different parameters and properties.
- Open-ended modifications and customisations of weapons and characters produce combinatorial growth, which provides many benefits but makes difficult to test every possible combination of character, weapon, and environment in every possible moment of play.
- The fair-to-play principle stipulates that the best and most skilful player wins the battle. All other causes must be neglected in a fair battle.

Considering the factors above, DGA combines transitive and intransitive approaches to its balance.

2.13. Game Security

DGA is free/libre-to-game for its players whose security and confidentiality is assured by the game. However, there is an open discourse whether it is possible to secure a free/libre/open-source MMORPG, which source code is accessible to cheating and bullying players.

DGA relies on Kerckhoffs's principle, which states that a free/libre/open-source program is not inherently less secure than a proprietary or commercial one, but it is no more secure either.

2.14. Anti-Cheating Statement

DGA is fully committed to anti-cheating policies and measures to ensure fairness, equality, justice, honesty, and integrity to all its players. Cheating by the players not only threaten other players but also violates the fair-to-play principle.

At the same time, the DGA Game and The FLEISS Software Foundation do not consider cheating as the immanent nature of MMO games and their players. Inherently unfair players constitute an insignificant part of gaming community consisting of people who sincerely and honestly share their love for playing video games.

2.15. Unique Selling Points

While building on the tried-and-true third/first-person action warfare-themed shooter gameplay, DGA innovates in a number of key areas:

- **Free/Libre-To-Game:** DGA implements a free/libre/open-source, fair-to-play approach, which produces a fully transparent, trustworthy, respectful to all concerned parties business, development, and social ecosystem.
- **Community-Driven Development:** DGA aims to form an active, vibrant, globally distributed community of people interested in gaming, military history, mechanics, game design, game development, 3D design, etc.
- **Historical Authenticity/Accuracy:** DGA is intended to become a tool for transparent reconstruction of historical weapons — discuss and modeling them, trying them out, test and prove, compare and oppose them.
- **Cross-Platform By Design:** DGA strives to be built on the most free/libre/open-source, cross-platform, powerful technologies ensuring the possibility to avoid vendor and platform lock-ins.
- **Unlimited Game Design:** DGA gives players open-ended control over their environments, characters, weapons, as well as in the creation of their own modifications and customisations. The only limit is historical authenticity and accuracy. On the other hand, game designers and developers, as well as any volunteer can freely create derived works to be added into the game or used independently.
- **Massively Multiplayer Online:** DGA allows players to participate in battles, install new game servers and domains, create new modes and features.
- **World Domination Story:** DGA has no limits in growth, diffusion, and dissemination.

3. Game Architecture

This section is currently a draft, and is subject to change.

DGA implements a multitier client-server software architecture that is a distributed application structure, which partitions tasks/workloads transmitted over networks between resource/service providers, called **servers**, and service requesters, called **clients**.

This chapter provides an overview of DGA's software architecture and related game modes.

3.1. Game Architecture

The DGA Game is designed and implemented as a multitier client-server software application based on the Java Platform.

DGA is intended to be played with an unlimited number of players accessing through their **game clients** an unlimited number of **game servers** connected into a **game network**. The Java Platform provides an integrated infrastructure that facilitates developing and running software applications written in the Java programming language. The Java Platform has been implemented for a wide variety of hardware architectures and operating systems with a view to enable Java programs to run identically on all of them. This intention made Java a truly high-performance, vendor-independent, device-agnostic software ecosystem.

In networking, DGA also seeks to apply high-performance, vendor-independent, device-agnostic solutions. Hence, the XMPP Network Protocol is implemented as the first application protocol supported by the Game. DGA uses XMPP to communicate game clients with game servers and game servers among themselves.

The figure below depicts DGA's principle software architecture.

DGA's game client is a Java application running on the jMonkeyEngine Game Engine. While the player starts the game client, he/she can choose either the single-player mode or a multiplayer one. The single-player mode doesn't require a network connection and is played locally on the player's computer. In the multiplayer mode, the player accesses a remote game server where multiple players interact and communicate with each other on a single **game world**. The massive multiplayer mode provides battle environments, which a large number of players can simultaneously participate in.

DGA's game servers are built on the Jakarta EE Platform; the servers differ on their purposes, and need to be available over a network connection. For providing a local multiplayer game, a **game world server** has to be deployed on a local network (Intranet). For making the game global, the game server needs to be available over the Internet.

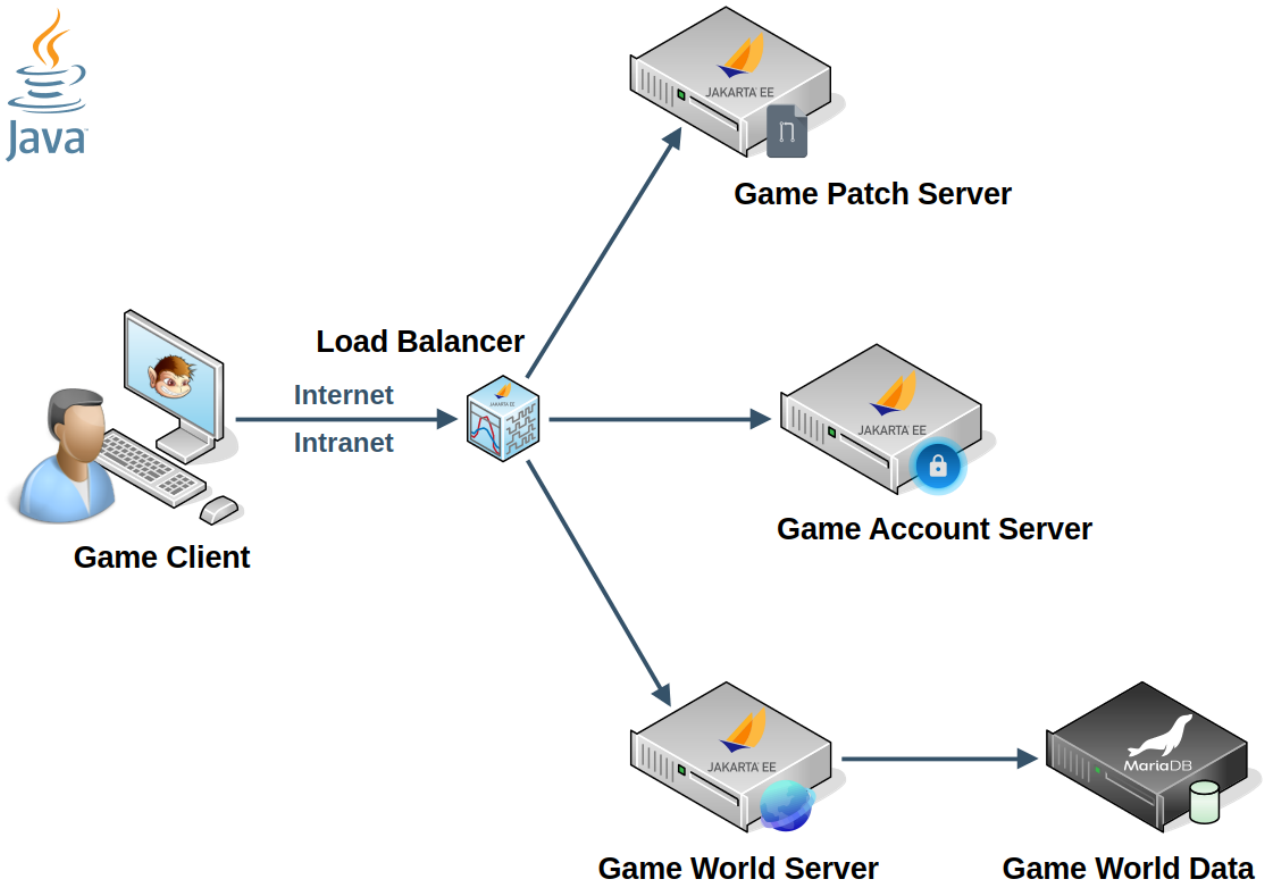

Figure 3.1. DGA Game's Principal Architecture

DGA's game modes at a player's disposal are determined by the chosen deployment architecture. The following sections describe different game modes and their variations in the context of software architecture.

3.2. Game User Modes (GUMs)

DGA's user modes define how many players can participate in a single battle session at the same time.

3.2.1. Single-Player User Mode (SPUM)

The single-player mode is intended to provide a player with battle sessions either to study and train battle maps and environments or to fight hostile mobs. In this mode the Game works locally and doesn't require network connections.

This mode also serves for testing and playtesting purposes to try out new game features and services.

3.2.2. Multi-Player User Mode (MPUM)

The multiplayer mode provides players with cooperative battle sessions to collaborate with and compete against other players. This mode needs a game world server to be deployed and accessible to players over a network. The game world server can be deployed on a player's local computer or a remote dedicated server, but network connections are used in either case.

The battles in this mode are limited by the number of participated players and duration of time.

Allowing participation of mobs in battles equally with human players is determined by the policy of the game world server, which is obliged to notify players about the such allowance before starting a battle session. Non-playing characters controlled by players are treated as bots and blocked by the server.

3.2.3. Massive Multi-Player User Mode (MMPUM)

The massive multiplayer mode provides players with battle environments allowing a large number of players to participate in a common battle session on a huge battlefield.

3.3. Game Battle Modes (GBMs)

DGA's battle modes define fighting moods differing in proportion between historical authenticity/accuracy and shooting gameplay.

DGA's battle modes and user modes are in a binary relation of the Cartesian product: every battle mode can be played in every user mode and vice versa.

3.3.1. Struggle Battle Mode (STBM)

The struggle battle mode provides players with a dynamic battle rhythm either the most fast-paced and accurately authentic. In this mode a player commands a weapon unit in a high-powered shooter gameplay, which is optimised for fast-paced actions but do not contradicts historical authenticity and accuracy.

3.3.2. Realistic Battle Mode (RCBM)

The realistic battle mode gives a player first-hand experience in commanding a weapon unit through a realistic battle environment and learning how this weapon unit behaved in reality. This mode provides a reasonably fast-paced shooter gameplay, which closely corresponds to the real weapon and do not contradicts historical authenticity and accuracy.

3.3.3. Simulator Battle Mode (SIBM)

The simulator battle mode provides a player with the most realistic control over a weapon unit, which strictly corresponds to its real counterpart. This mode is the most historically authentic and accurate with some reasonably fast-paced shooter gameplay.

3.4. Game Battle Types (GBTs)

DGA's battle types vary in how players are combined for a common battle session.

DGA's battle types are partly interrelated with the user modes and in a binary relation of the Cartesian product with the battle modes: every battle type can be played in every battle mode and vice versa.

3.4.1. Call-To Battle (CTB)

DGA's call-to battles are conceived to compose battle teams of unrelated players waiting for going into a battle, i.e. waiting for a call-to-battle. Players are picked for a team by weapon unit, battle experience, and other parameters. A battle map for the battle session is selected among the maps chosen by the players.

Call-to battles in the single-player mode are trivial. These battles may engage players in the multiplayer modes.

3.4.2. Team Battle (TMB)

DGA's team battles are designed for groups of players, who want to go into a battle on the same team. Teams and their members are composed ahead of time and registered for a battle all in one go.

Team battles work only in the multiplayer modes.

3.5. Battle Aiming Modes (BAMs)

DGA's battle aiming modes define how the player points a weapon towards targets to hit them during a battle.

3.5.1. Struggle Aiming Mode (STAM)

The struggle aiming mode provides the player with a third-person perspective when a camera is positioned above and behind, and it follows the player's game character with a weapon's aiming point towards targets on the battlefield. This mode is optimal for close combat at short and medium distances. The mode is additionally equipped with the auto-aiming feature, which can be activated against hostile units.

3.5.2. Sniper Aiming Mode (SNAM)

The sniper aiming mode provides the player with a first-person perspective when the game character's weapon is aiming directly at targets on the display. This mode is optimal for long-distance, precise shooting.

3.5.3. Grazing Aiming Mode (GRAM)

The grazing aiming mode provides the player with a third-person perspective, which shows the battlefield from above and allows long-distance **grazing fire**. This mode is equipped with two battle sights differing in the perspective: **overhead sight** and **grazing sight**.

4. Product Concept

This section is currently a draft, and is subject to change.

Following the F/LOSS Way, DGA strives to find a stable equilibrium between the public good and economic income, and as a result, to build a successful game economy.

This chapter delineates the product-oriented aspects of the DGA Game.

4.1. Player Experience

The fantasy that DGA supplies is to turn a player into a military commander going into a battle with a historical weapon unit on a historic map of choice. DGA aims at letting the player get a bunch of feelings from control over the weapon unit, estimation of its historical authenticity/accuracy, and eventually, immersion of the player into the art of warfare and the sense of war. DGA also aspires to guide the player through military routines and operations with weapons as they are practised in reality.

Since entrance into the Game, the player has to choose a weapon unit to go into battle; the first types of weapons being realised are mechanised/motorised land combat vehicles and guns, dating from the time of World War I up to the Cold War, with giving preference to World War II. The weapon units are arranged by weapon types and countries, thus the player first selects a weapon type and then a country. Every country registered in the game has a set of technology trees for every type of weapons. The weapon units manufactured, and/or exploited, and/or developed by the country are distributed across its technology tree by model, year of entering service, and technological epoch. The player selects the desired model of weapon units according to the proper interest and without any limitations. If the model is chosen for the first time then after having manufacturing a weapon unit, the player can additionally customise its standard parameters at a **weapons factory**. After having passed the first battle, the weapon unit becomes a battle unit and can be customised according to the field modifications made in real life by soldiers at the tactical level, which the player implements at a **front-line repair shop**.

After having prepared the desired weapon unit, with which the player can go into a game battle (also called a battle session). For doing that, the player has to choose a battle mode, which define the level of historical authenticity/accuracy of the game battle, and a battle type, which specifies a method of recruitment for game teams. The player always battles in a team. In the single-player mode, the player can battle with mobs and against mobs; otherwise — when mobs are disabled — the player can train and study the game alone. In a multi-player mode, other players always accompany the player into battle.

Battle sessions are limited on the time duration and amount of players participated in every team. During the battle, the player aims to put out of action all the weapon units of the opposite team or to accomplish some victorious condition (capture the flag, maintain a position against enemies, make the opposite team surrender, etc.). After having finished the battle, the players receive their rewards in experience points and cash according on their performance during the battle.

The player can spend the gained experience points to explore the battle unit's new modules and field modifications, which combinations are intended to be open-ended in variety and scope. The player can spend the accumulated cash for buying special equipment and materials to augment the battle unit's performance and skills of its crew. The battle unit's repairs, refuelling, replenishment of ammunition,

changes in weapons and camouflage, as well as other routine military activities are made automatically and without any interruption for the player.

Towards player experience, DGA's main idea as its motto suggests is to provide gamers with a game that requires them only to game by concentrating on the game's aspects without being constrained by farming, grinding, and other meaningless actions, which have nothing common neither with the art of warfare nor military technology.

At the end of the day, an overarching experience DGA seeks to discover to its players is to show through the mechanics of warfare and war the importance of human life and why war must be avoided at all costs.

> *Master Sun said:*
> *The art of warfare is this:*
> *It is best to keep one's own state intact; to crush the enemy's state is only a second best. It is best to keep one's own army, battalion, company, or five-man squad intact; to crush the enemy's army, battalion, company, or five-man squad is only a second best. So to win a hundred victories in a hundred battles is not the highest excellence; the highest excellence is to subdue the enemy's army without fighting at all. (TAOWA, p. 79)*
>
> — *Sun Tzu*
> *The Art of Warfare: Chapter III. Planning the Attack (5th century BCE)*

4.2. Visual and Audio Style

DGA's visuals promise to be art-intensive, clean, and performant at the same time. The game is intended to have a light, airy feel with bright, vivid colors, which reproduce natural scenery with picturesque landscapes and camouflaged military machines acting in various seasons and weather conditions. Since the game combines role-playing, cross-platform, and massively-multiplayer-online requirements, its visual style will be performant as much as possible to provide the most stunning gameplay available for Java games.

DGA's audio style also corresponds to natural environments, including machinery, weapons, gun sounds and the like. Music will consist of opening themes and atmospheric melodies to accompany gameplay.

4.3. Game World Fiction

DGA's game worlds are battle maps, also called battlefields, where players can apply their combat skills as military commanders. The battle maps are based on geographic locations where famous historic battles and conflicts involving significant numbers of military machines were fought. The first battle maps being implemented in the game are locations of the prominent battles of World War II.

For improving gameplay feel, the battle maps in different battle modes can vary in structure and details. It means that in the realistic and simulator battle modes a battle map will exactly correspond to the historic battle's geographic location, of course, with all reasonable assumptions made for a computer model imitating the real world. But in the struggle battle mode the same battle map can bear some fabricated modifications to improve play experience and fast-pacedness of battle sessions. Nevertheless, such modifications wouldn't be too substantial for the sake of preserving historical accuracy.

Moreover, DGA's concept assumes that additional geographic locations for the corresponding time periods can be added to the game. It means that some places where thenadays no battles took place but which are of interest to be modeled — such places could also be added to the game as battle maps for study and fun purposes.

4.4. Monetisation

As a free/libre/open-source software application following the free/libre-to-game business model, DGA implements cutting-edge business strategies for monetisation and commercialisation, which promote but not limited by the following business methods:

- Donation-Based Funding
- Branded Merchandise
- Crowdfunding
- Crowdsourcing
- Professional Services
- Partnership with Funding Organizations
- Advertising-Supported Software Features.

4.5. Technology, Tools, and Platforms

DGA has been conceived as a complex software product intended to bind a bundle of technical requirements, some of which are contradictory, conflicted, and ambivalent. This section roughly estimates how these requirements can meet technical parameters.

4.5.1. Hardware and Operating System

DGA is implemented on the Java Platform, which is a standardised, rock-solid, highly-scalable, enterprise-grade, future-proofed software platform characterised by the following essential properties:

- Write Once, Run Anywhere (WORA)
- High-Performance Execution
- Increased Memory Usage (as a result of the previous ones).

Consequently, DGA Game Client can be outlined by the next recommended system requirements to a first approximation:

- CPU: 64-bit processor and operating system
- Memory: 8 GB RAM
- Graphics: 2 GB VRAM
- Graphics Library: OpenGL
- Network: Gigabit Ethernet Network Connection (for a local multiplayer user mode) + Broadband Internet Connection (for a global multiplayer user mode)
- Storage: 2 GB available space

OS: GNU/Linux Kernel 6.12+, FreeBSD 14+, HarmonyOS 5+, Microsoft Windows 11+, macOS 15+

DGA Game Server requires this approximate system configuration:

- CPU: 64-bit processor and operating system
- Memory: 2 GB RAM
- Network: Gigabit Ethernet Network Connection (for a local multiplayer user mode) + Broadband Internet Connection (for a global multiplayer user mode)
- Storage: 1 GB available space
- OS: GNU/Linux Kernel 6.12+, FreeBSD 14+, HarmonyOS 5+, Microsoft Windows 11+, macOS 15+

4.5.2. Development Tools

Java comes with an army of development tools to conquer projects of any level of complexity, either free/libre/open-source or proprietary. Here DGA states two major concepts:

- Since DGA follows the free/libre-to-game business and development model, only free/libre/open-source development tools can be used within the project.
- Java is extensively common in multiple areas, but particularly in gaming its positions are not so brilliant. It means that some extra efforts are normally required to develop games on Java. On the other side, the Java Platform provides the richest toolset existing in the IT industry.

Thus, DGA's development tools are as follows:

- On the client side, DGA runs on the jMonkeyEngine Game Engine, which comes with its own game development environment jMonkeyEngine SDK (Software Development Kit) that is based on the Apache NetBeans IDE's RCP (Rich Client Platform).
- On the server side, DGA servers run on the Jakarta EE Platform and specifically on the Eclipse GlassFish and Payara Platform Community Edition application servers.
- Eclipse IDE as the first integrated development environment (IDE).
- Apache NetBeans IDE as the second IDE.

4.5.3. Server and Networking

DGA requires a game server and networking only for a multiplayer mode. The game server can be either a local server (running on a gamer's local network) or a global one (running online, i.e. over the Internet).

The massive multiplayer mode is a variety of the multiplayer mode enlarged in number of players and size of battlefields; this mode can also be either local or global. In other words, it's completely perfect to establish a local game server to play massively multiplayer on a local network (LAN/MAN/WAN).

DGA's game servers are intended to be federated, allied, associated, unified, and united in any forms of alliances gamers and server administrators could dream of.

4.5.4. Monetisation and Advertising

DGA supposes to support running ads and some forms of in-game monetisation. It means that DGA's game client and servers support integration with external advertising and payment servers.

4.5.5. Internationalisation and Localisation

From the beginning, DGA is implemented as an internationalised computer software addressed to the global audience and international markets. The following locale is default for the DGA Game and its projects:

Locale	Name	Description
en-EU	English in the European Union (European English, EU English).	The English language as it is accepted in the European Union as the shared standard usage of Ireland and the United Kingdom. As a general rule, Irish/British English is preferred, and Americanisms that are liable not to be understood by speakers of Irish/British English should be avoided. However, bearing in mind that a considerable proportion of the target readership may be made up of non-native speakers, very colloquial Irish/British usage should also be avoided. Although, the metric system of weights and measures (SI System) is used by default. (EU Language Rules; EU Guidelines for Translating into English)

At the same time, DGA is intended to be localised for as many languages and countries as possible. DGA strives to communicate with every player in his/her native language or in a language of his/her preference. Nevertheless, this communication has to be well implemented.

4.6. Scope

As in case of any free/libre/open-source computer program, the scope of the DGA Game is quite difficult to designate.

On one side, DGA is simply intended to satisfy a personal want of a well-designed, well-balanced MMORPG featuring armored warfare and working on a GNU/Linux operating system, maybe for a reasonable price. In the world of free/libre/open-source software this situation is called 'starting by scratching a developer's personal itch'. (CATB, p. 23)

On the other side, MMORPGs are the most difficult and demanding kind of computer games for implementation. From the technical challenges of building and maintaining a massive online world, to the design challenges of creating and updating diverse and engaging content, to the balancing challenges of ensuring a fair and fun gameplay and economy, which constantly face a lot of competition and pressure from the market and the players — MMORPGs require a lot of resources, skills, creativity, and efforts to make them worth it and justifiable for a game studio over any other game project. In other words, it's completely meaningless to make a small, feature-limited MMORPG, because it merely doesn't worth the effort.

But how an indie developer can make a game that requires development and marketing budgets available only to large AAA-studios? Here the F/LOSS Way comes to help to leverage on people, skills, and resources in a unified effort to get an outstanding game product of the world-class quality.

5. Systems Concept

This section is currently a draft, and is subject to change.

DGA is a complex system consisting of multiple components organised into various tiers and layers, which are distributed across numerous nodes and machines. Nevertheless, the game features laying atop of all software components and computing infrastructure are of definitive value.

Whereas the previous sections describe the high-level plan for the Game, this chapter and the following ones detail the more specific gaming aspects of the DGA Game. Some of these aspects are already implemented in code, although others serve as conceptual forecasts of certain degree of accuracy.

5.1. Fictional Players

For describing DGA's gaming aspects the following fictional players are used. Personal characteristics of these players are based on Richard Bartle's taxonomy of player types (DVWS, p. 227), and they are named after the famous commentators of Sun Tzu's The Art of War. (TAOWC, pp. 37-41)

Name	Pronunciation	Bartle's Type	Bartle's Subtype	Personality	Named After
Xi	*Shi*	Achievers	Planner	They set a goal and aim to achieve it.	Wang Xi (Sung Dynasty, early 11th century CE), a scholar, author, one of the Confucian classics of ancient illustrative history.
Hao	*Haw*	Achievers	Opportunist	They look around for things to do, but they don't know what these are until they find them.	Chen Hao (Sung Dynasty, early 12th century CE), an officer of the state known for his extraordinary personal independence and his great aspirations.
Duyou	*Dooyou*	Explorers	Scientist	They are methodical in their acquisition of knowledge.	Du You (Tang Dynasty, 735–812 CE), an official military advisor, war councillor, and military inspector.

Name	Pronunciation	Bartle's Type	Bartle's Subtype	Personality	Named After
Dumu	*Doomoo*	Explorers	Hacker	They have an intuitive understanding of the virtual world, with no need to test their ideas.	Du Mu (Tang Dynasty, 803–852 CE), the grandson of Du You, known as a 'knight of unflinching honesty and extraordinary honor', he earned an advanced academic degree and served in several positions at the court, an outstanding poet.
Lin	*Lin*	Socialisers	Networker	They assess other player's capabilities.	Jia Lin (Tang Dynasty 618–906 CE), a commentator, cultural and political influencer.
Quan	*Tchuan*	Socialisers	Friend	They enjoy their company.	Li Quan (Tang Dynasty 618–906 CE), a commentator known for his military strategy, and a devotee of Taoism and martial arts.
Yaochen	*Yaochen*	Killers	Politician	Their aim is to get a big, good reputation.	Mei Yaochen (Sung Dynasty, 1002–1060), a distinguished statesman, compiler, editor, and writer.
Cao	*Tsao*	Killers	Griefer	Their vague aim is to get a big, bad reputation.	Cao Cao (Han Dynasty, 155–200 CE), a statesman, warlord, and poet.

5.2. Core Loops

DGA provides players with the following gaming core loops, which in whole form the Game's interactivity.

5.2.1. Game Client Loop (GCL)

The game client loop (GCL) defines what the player commonly does when operating in DGA's client application (the Client).

The loop is described by a scenario with details as follows:

1. Player Hao starts the Client on his personal computer (PC).

The loop is described by a scenario with details as follows:

1. In the Main Menu, Hao can choose a desired Game User Mode to play, which can be either single-player or multi-player.
2. Hao can choose the Single-Player User Mode by clicking the button 'Singleplayer'.
3. Hao can choose the Multi-Player User Mode by clicking the button 'Multiplayer'.
4. In the Main Menu, Hao can set the Client's options by clicking the button 'Options' and opening the Options Menu where the Client's parameters can be tuned up.
5. In the Main Menu, Hao can quit the Client and Game by clicking the button 'Quit Game'.

5.2.3. Single-Player Loop (SPL)

The single-player loop (SPL) defines what the player commonly does when operating in the single-player user mode (SPUM).

The loop is described by a scenario with details as follows:

1. In this mode, battles are played locally, i.e. on the Player's PC only, without using network connections, and against mobs only, thereby an authentication with a username and password for the Player isn't required.
2. The Singleplayer Menu is displayed, where the Player can set parameters for the single-player battles.
3. Hao passes to the Garage-Battle Loop.
4. Hao can leave the single-player mode by clicking the button 'Back' and returning to the Main Menu.

5.2.4. Multi-Player Loop (MPL)

The multi-player loop (MPL) defines what the player commonly does when operating in the multi-player user mode (MPUM).

The loop is described by a scenario with details as follows:

1. In this mode, battles are played on a game world server, which (1) must be deployed and accessible to the Player over a network, and (2) authenticates and authorises the Player by username and password, as well as other types of authentication.
2. The massive multi-player user mode (MMPUM) requires no special action from the Player; it differs from the ordinary multi-player user mode (MPUM) in settings made on the game server.
3. The Multiplayer Menu is displayed, where the Player can choose a game world server to join it with other players, whether online or nearby. The menu appears with a list of known servers which the Player may connect to.
4. If the required server isn't on the list, then the Player clicks the button 'Add Server', which takes the Player to a new menu where he can add a game server by entering the server name, server network address, user name and password. The Player clicks the button 'Done' to save the changes and add the server to the list.
5. Hao chooses the desired server by clicking its name on the list and automatically enters it.
6. Hao passes to the Garage-Battle Loop on the game server.

7. Hao can leave the multi-player mode by clicking the button 'Back' and returning to the Main Menu.
8. In the multi-player user mode, all the Player's actions are continuously and thoroughly examined for conformance to the Game's anti-cheating policies as in the Anti-Cheating Loop.

5.2.5. Garage-Battle Loop (GRGBTL)

The garage-battle loop (GRGBTL) is the core game loop of the DGA Game.

The loop is described by a scenario with details as follows:

1. The Garage-Battle Loop consists of two interrelated loops through which players play the Game: garage loop (GRGL) and battle loop (BTL).
2. The Garage Loop is presented through the Garage View by which players operate weapon units and their crews, their progression, customisation, budgeting, etc.
3. The Battle Loop is presented through the Battle View by which players go into battle with weapon units and crews set up in the Garage Loop.
4. The Garage Loop and Battle Loop are mutually interconnected: the Player enters the Battle View, which represents a battle session, from the Garage View, which represents a place where the selected weapon unit is repaired, prepared, and customised for the next battle. Correspondingly, after having finished the battle, the Player returns from the Battle View to the Garage View to prepare the weapon unit for the next battle.
5. From the Garage-Battle Loop the Player exits to the Main Menu.

5.2.6. Garage Loop (GRGL)

The garage loop (GRGL) defines what the player commonly does when operating in the Garage View.

The loop is described by a scenario with details as follows:

1. In the Garage View, Player Hao performs all the gaming activities except battling: operating weapon units and crews, training, upgrading, customising, budgeting, teaming, chatting, learning, etc.
2. The Player can use the Garage View to learn information about weapons, their models, units, characteristics, modifications, history, etc.
3. When Hao enters the Garage for the first time, he is invited to create his first weapon unit as in the Technology-Tree Loop; later the Player uses this loop to add weapon units to his Garage.
4. Hao chooses a weapon unit as in the Weapon Unit Selection Loop.
5. The selected weapon unit is displayed on the screen.
6. If the weapon unit is newly created and never participated in battles then only basic customisations can be applied to as in a factory — changes in camouflage, ammunition, crew, etc. The Player has to participate in battles with the weapon unit to enhance its capabilities.
7. Hao can equip the newly created weapon unit with either a new crew or an unengaged crew as in the Crew Selection Loop.
8. Hao selects the battle mode 'Struggle' as in the Battle Mode Selection Loop.
9. Hao selects the battle type 'Call-To Battle' as in the Battle Type Selection Loop.

10. Hao clicks the button 'Charge!' to go into battle as in the Battle Loop.
11. After having returned from the battle, Hao can spend the gained experience points to upgrade the weapon unit as in the Weapon Unit Upgrade Loop.
12. After having returned from the battle, Hao can spend the gained experience points to customise the weapon unit as in the Weapon Unit Customisation Loop.
13. After having returned from the battle, Hao can spend the gained cash to supply the weapon unit with additional consumable resources as in the Cash Budgeting Loop.
14. Hao can change the weapon unit's crew for another one or make changes in the crew as in the Crew Selection Loop.
15. Hao can improve qualifications of the weapon unit's crew and its crew-members with trainings as in the Crew Training Loop.
16. Hao can define custom sets of qualifications for the weapon unit's crew and its crew-members as in the Crew Customisation Loop.
17. Hao can communicate with other players within the Game as in the Team Communication Loop.
18. Hao can participate as well as organise and lead teams and other joint forces within the Game as in the Teaming Loop.
19. Hao can leave the Garage at any moment by calling the Pause Menu as in the Pause-Menu Loop, and then by clicking the button 'Back' he returns either to the Singleplayer Menu or Multiplayer Menu correspondingly.

5.2.7. Battle Loop (BTL)

The battle loop (BTL) defines what the player commonly does when operating in the Battle View.

The loop is described by a scenario with details as follows:

1. In the Battle View, Player Hao performs only one gaming activity — battling, i.e. the player is turning into a military commander going into a battle with a historical weapon unit on a historic map of choice.
2. The battle is limited on time and amount of players distributed across at least two opposing game teams.
3. The battlefield for the battle is chosen automatically by the game server, however the player can parameterise this choice by the user's preferences and game modes.
4. Hao waits while the game teams are composed and the battle is starting.
5. During the battle, the Player manages the weapon unit's position, movement, and firing from all the available weapons at the hostile weapon units. The goal is to put out of action all the units of the opposite team or to accomplish some victorious condition (capture the flag, maintain a position against enemies, make the opposite team surrender, etc.).
6. During the battle, Hao selects an active weapon as in the Weapon Selection Loop.
7. During the battle, Hao selects an active ammunition for the selected active weapon as in the Ammunition Selection Loop.
8. During the battle, Hao selects the battle aiming mode as in the Battle Aiming Mode Selection Loop.
9.

The Player gains combat experience and financial rewards for his combat performance, victorious achievements, and battle contributions showed in the battle.

10. During the battle, Hao coordinates his proper actions with teammates' actions as in the Team Coordination Loop.
11. After having finished the battle, Hao gains experience points and cash paid on his combat performance, relative contribution to the team's result, and so on, displayed in the Battle Score Menu.
12. After having examined the battle's results, Hao closes the Battle Score Menu and returns to the Garage View.
13. During the battle, Hao can interrupt the course of the Client's action at any moment by calling the Pause Menu as in the Pause-Menu Loop. However, the Player can't leave the battle until either the course of the battle's action has ended or his weapon unit has been put out of action before the battle's end. If the required conditions are met, then the Pause Menu called during the battle contains the button 'Leave Battle', otherwise the menu doesn't contain any tools to quit the battle.

5.2.8. Technology-Tree Loop (TTL)

The technology-tree loop (TTL) defines what the player commonly does when operating weapon models through technology trees displayed in the Technology Tree View.

The loop is described by a scenario with details as follows:

1. The Technology Tree View presents to players the historical timeline of the development of weapons organised in the technology trees sorted by weapon type and country of origin.
2. In the Technology Tree View, Player Hao selects a weapon model of which he wants a weapon unit to play with.
3. Hao opens the Technology Tree View.
4. Hao selects a weapon type as in the Weapon Type Selection Loop.
5. Hao selects a country as in the Country Selection Loop.
6. In the displayed technology tree, the Player chooses a weapon model by clicking it and in the appeared window presenting historical information about the weapon he clicks the button 'Create New Unit'.
7. With a newly created weapon unit the Player returns to the Garage View, which is displayed as in a factory.
8. If the Player only wants to accomplish some historical studies and observations, then he can exit the Technology Tree View at any moment without any changes in his weapon units.

5.2.9. Weapon Type Selection Loop (WTSL)

The weapon type selection loop (WTSL) defines what the player commonly does when selecting a weapon type with the Weapon Type Selection Menu.

The loop is described by a scenario with details as follows:

1. In the Weapon Type Selection Menu, Player Hao can choose a desired weapon type.
2. Hao selects a weapon type by clicking on it in the Weapon Type Selection Menu.

5.2.10. Country Selection Loop (CYSL)

The country selection loop (CYSL) defines what the player commonly does when selecting a country with the Country Selection Menu.

The loop is described by a scenario with details as follows:

1. In the Country Selection Menu, Player Hao can choose a desired country.
2. Hao selects a country by clicking on it in the Country Selection Menu.

5.2.11. Weapon Unit Selection Loop (WUSL)

The weapon unit selection loop (WUSL) defines what the player commonly does when selecting a weapon unit with the Weapon Unit Selection Menu.

The loop is described by a scenario with details as follows:

1. In the Weapon Unit Selection Menu, Player Hao can choose a desired weapon unit to operate with. Weapon units are grouped by model; the Player can create multiple weapon units of the same model and choose one of them. The last used weapon unit becomes the default unit for its weapon model. Weapon units may differ in specialisation, camouflage, crew skills, custom equipment, etc.
2. Hao selects the default weapon unit of the desired weapon model in the Weapon Unit Selection Menu.
3. If Hao wants to change the weapon unit then he opens the weapon model's menu and selects another weapon unit.
4. The selected weapon unit is displayed to the Player on the screen.

5.2.12. Weapon Unit Upgrade Loop (WUUL)

The weapon unit upgrade loop (WUUL) defines what the player commonly does when implementing standard modifications to the selected weapon unit with the Weapon Unit Upgrade Menu. Upgrading improves the weapon unit's technical and combat performance.

The loop is described by a scenario with details as follows:

1. In the Weapon Unit Upgrade Menu, Player Hao can implement standard modifications the unit's weapon model survived in production. These modifications include all historical changes in armament, modules, ammunition, equipment, camouflage, consumable resources, crew, etc.
2. For upgrading his weapon unit, Hao has to have a sufficient amount of experience points gained with this weapon unit in battles.
3. Hao opens the Weapon Unit Upgrade Menu for the selected weapon unit.
4. Hao chooses the weapon unit's part, which he wants to upgrade and for which he has enough experience points, by clicking on it.
5. Hao clicks the button 'Upgrade' for the selected part.
6. After the selected part has been upgraded, the amount of collected experience points decreases by the part's upgrade cost.

5.2.13. Weapon Unit Customisation Loop (WUCL)

The weapon unit customisation loop (WUCL) defines what the player commonly does when implementing custom modifications to the selected weapon unit with the Weapon Unit Custom Menu. Customisation improves the weapon unit's technical and combat performance, but also is a tool to personalise the weapon unit and specialise it for particular combat tasks.

The loop is described by a scenario with details as follows:

1. In the Weapon Unit Custom Menu, Player Hao can implement custom modifications the unit's weapon model survived in everyday practice on the front line. These modifications include all historical changes in armament, modules, ammunition, equipment, camouflage, consumable resources, crew, etc.
2. For customising his weapon unit, Hao has to have a sufficient amount of experience points gained with this weapon unit in battles.
3. Hao opens the Weapon Unit Custom Menu for the selected weapon unit.
4. Hao chooses the weapon unit's part, which he wants to customise and for which he has enough experience points, by clicking on it.
5. Hao clicks the button 'Customise' for the selected part.
6. After the selected part has been customised, the amount of collected experience points decreases by the part's custom cost.

5.2.14. Crew Selection Loop (CSL)

The crew selection loop (CSL) defines what the player commonly does when operating with crews with the Crew Selection Menu in the Barracks View. All weapon units in the DGA Game are crew-served weapons, which always have operating crews. An experienced crew can significantly increase the weapon unit's performance.

The loop is described by a scenario with details as follows:

1. In the Crew Selection Menu, Player Hao can choose a desired crew to operate with.
2. Crews are grouped by the countries they fought for; the Player can create multiple crews for different weapon models and choose one of them. Crews may differ in specialisation, skills, supported weapon models and equipment, etc.
3. Each weapon unit has an active operating crew. If the Player wants to change the active crew, then he has to free the current crew up and select another one. The weapon unit may exist without crew or a complete crew. A crew may exist without any weapon unit.
4. The weapon unit can't participate in battle without a complete fully-trained crew.
5. A newly created weapon unit has either a newly created crew or an existing crew selected in Barracks.
6. Hao selects a crew for the selected weapon unit in the Crew Selection Menu and clicks the button 'Enlist'.
7. Hao can select to create a new crew or to make changes in an existing crew for the selected weapon unit as in the Crew Recruitment Loop.
8. If the desired crew isn't sufficiently trained, then the Player can train and upskill the crew as in the Crew Training Loop.

9. The selected crew becomes active for the selected weapon unit.

5.2.15. Crew Recruitment Loop (CRL)

The crew recruitment loop (CRL) defines what the player commonly does when operating with crew-members of weapon units' crews with the Crew Recruitment Menu in the Barracks View. Crew-members perfect their experience and skills in every battle; the crew's technical and combat performance depends on the qualification and skills of its members.

The loop is described by a scenario with details as follows:

1. In the Crew Recruitment Menu, Player Hao can either equip a crew with existing crew-members or recruit new volunteers to serve in the crew.
2. Crew-members are grouped by the country they fought for; the Player can recruit multiple crew-members for different crews and weapon models. Crew-members may differ in specialisation, skills, supported weapon models and equipment, etc.
3. The Player chooses a crew, which a crew-member is equipped for. A crew-member can't serve in multiple crews at the same time.
4. Hao selects a crew-member for the selected crew in the Crew Recruitment Menu and clicks the button 'Recruit'.
5. Hao can select to recruit a new volunteer to serve in the selected crew with the Crew Recruitment Menu and clicks the button 'Recruit'.
6. The renewed crew is displayed to the Player on the screen.

5.2.16. Crew Training Loop (CTL)

The crew training loop (CTL) defines what the player commonly does when training crews and their crew-members with the Crew Training Menu in the Barracks View. Players can considerably improve the technical and combat performance of their crews with regular trainings.

The loop is described by a scenario with details as follows:

1. With the Crew Training Menu Player Hao can conduct regular trainings for his crews and their crew-members. Regular trainings are one-time actions to improve qualification and skills of the crew in whole and all its crew-members.
2. For providing a regular training to a crew, the Player has to have a sufficient amount of experience points gained in battles.
3. Hao selects a crew he wants to train in the Crew Training Menu.
4. Hao selects a regular training for the selected crew in the Crew Training Menu.
5. Hao accomplishes the selected training for the selected crew by clicking the button 'Accomplish'.
6. The upskilled crew is displayed to the Player on the screen.

5.2.17. Crew Customisation Loop (CCL)

The crew customisation loop (CCL) defines what the player commonly does when implementing custom settings to weapon units' crews and their crew-members with the Crew Custom Menu in the Barracks View.

Crews and crew-members can be customised in specialisation, skills, supported weapon models and equipment, etc.

The loop is described by a scenario with details as follows:

1. In the Crew Custom Menu, Player Hao can compose custom sets of skills for a crew in whole and for every of its crew-members.
2. The crew skills comprise firing experience, technical skills, operational experience, familiarity of crew-members with each other, mission experience, communicating skills, controlling skills, co-deciding skills, etc.
3. The crew-member skills comprise capabilities in military specialism, technical skills, leadership, teamwork, dedication, communication, problem solving, etc.
4. Customisation prioritises skills, which will develop first in battles. The amount of skills in top priority is always less than the total amount of skills. Thereby, a custom set of skills, which differ in repertoire and value, may be created.
5. For implementing customisation, Hao has to have a sufficient amount of experience points gained in battles.
6. Hao selects a crew skill to develop first for the selected crew in the Crew Custom Menu and clicks the button 'Develop'.
7. Hao selects a crew-member skill to develop first for the selected crew-member of the selected crew in the Crew Custom Menu and clicks the button 'Develop'.
8. The renewed crew is displayed to the Player on the screen.

5.2.18. Cash Budgeting Loop (CBL)

The cash budgeting loop (CBL) defines what the player commonly does when managing gained in-game currency with the Buy Menu in the Cash Budgeting View. Players can use cash gained in battles to slightly improve the technical and combat performance of weapon units and their crews. With cash, the players can buy consumable resources (special equipment, materials, things, etc.) to add technical and combat performance to some little degree. Consumable resources, or simply consumables, represent extra gear, which additionally improve performance of weapon units and crews.

The loop is described by a scenario with details as follows:

1. In the Buy Menu, Player Hao can buy consumables for the cash.
2. Hao selects a consumable to buy for the selected weapon unit with the Buy Menu.
3. Hao selects a consumable to buy for the selected crew of the selected weapon unit with the Buy Menu.
4. Hao makes a one-time deal on the selected consumable in the Buy Menu by clicking the button 'One-Time Buy'. The selected consumable will be bought once.
5. Hao makes a regular deal on the selected consumable in the Buy Menu by clicking the button 'Regular Buy'. The selected consumable will be automatically replenished for the cash after every battle.
6. The renewed crew is displayed to the Player on the screen.
7. The renewed weapon unit is displayed to the Player on the screen.

5.2.19. Weapon Selection Loop (WSL)

The weapon selection loop (WSL) defines what the player commonly does when selecting a weapon during a battle with the Weapon Selection Menu, or keyboard, or mouse.

The loop is described by a scenario with details as follows:

1. Every weapon unit always has the default active weapon.
2. In the Weapon Selection Menu, Player Hao can change the active weapon.
3. Hao selects a new active weapon by clicking on it in the Weapon Selection Menu, or with the keyboard or mouse.

5.2.20. Ammunition Selection Loop (ASL)

The ammunition selection loop (ASL) defines what the player commonly does when selecting an ammunition during a battle with the Ammunition Selection Menu, or keyboard, or mouse.

The loop is described by a scenario with details as follows:

1. Every weapon always has the default active ammunition.
2. In the Ammunition Selection Menu, Player Hao can change the active ammunition for the active weapon selected in the Weapon Selection Loop.
3. Hao selects a new active ammunition for the selected active weapon by clicking on it in the Ammunition Selection Menu, or with the keyboard or mouse.

5.2.21. Battle Aiming Mode Selection Loop (BAMSL)

The battle aiming mode selection loop (BAMSL) defines what the player commonly does when selecting a battle aiming mode with the Battle Aiming Mode Selection Menu, or keyboard, or mouse.

The loop is described by a scenario with details as follows:

1. Every weapon always has the default aiming mode.
2. In the Battle Aiming Mode Selection Menu, Player Hao can change the aiming mode for another one.
3. Hao selects a new battle aiming mode by clicking on it in the Battle Aiming Mode Selection Menu, or with the keyboard or mouse.

5.2.22. Team Coordination Loop (TCRL)

The team coordination loop (TCRL) defines what the player commonly does when coordinating proper actions with teammates' actions in the Battle Signal View displayed through the Battle View during a battle.

The loop is described by a scenario with details as follows:

1. Battlefield coordination facilitates the integration of players that enables them to prevent conflicts, shape the combat environment and win the battle.
2. For preventing conflicts amid the players only standard signals and directives are allowed to be transmitted during the battle. Free chatting and custom signals are not allowed.

The Battle Signal View consists of several menus and controls with which Player Hao can transmit and receive signals to and from his teammates to coordinate actions during the battle.

4. Hao sends a battle signal to one or more teammates during the battle.
5. Hao receives a battle signal from his teammate during the battle.
6. After having died in the battle, i.e. when the weapon unit has been put out of action, the Player can continue his presence in the battle as a spectator sending and receiving battle signals until the end of the battle.

5.2.23. Team Communication Loop (TCML)

The team communication loop (TCML) defines what the player commonly does when communicating with other players within the Game with the Team Communication Menu. Players can freely communicate with each other outside the Battle View where only standard signals and directives are allowed through the Battle Signal View.

The loop is described by a scenario with details as follows:

1. In the Team Communication Menu, Player Hao can chat with other players without any restrictions.
2. Hao opens the Team Communication Menu and sends a message to Player Xi by the name.
3. If Xi is online, willing and ready to chat, he sends a reply message to Hao.
4. The players can convert their personal chat to a group chat by inviting other players to join them.
5. Hao can blacklist some other players if he doesn't want to chat with them.
6. Hao can whitelist only those players whose messages he consents to receive.

5.2.24. Teaming Loop (TMGL)

The teaming loop (TMGL) defines what the player commonly does when building teams and commands with other players within the Game through the Teaming View.

The loop is described by a scenario with details as follows:

1. In the Teaming View, Player Hao performs all the gaming activities related to creating, managing, and coordinating game teams and commands within the Game.
2. In the Teaming View, Hao creates a game team, which other players may be invited in at anytime.
3. Hao manages and coordinates the game team, which goes into battle together.

5.2.25. Battle Mode Selection Loop (BMSL)

The battle mode selection loop (BMSL) defines what the player commonly does when selecting a battle mode with the Battle Mode Selection Menu.

The loop is described by a scenario with details as follows:

1. In the Battle Mode Selection Menu, Player Hao can choose a desired Game Battle Mode.
2. Hao selects a battle mode by clicking on it in the Battle Mode Selection Menu.

5.2.26. Battle Type Selection Loop (BTSL)

The battle type selection loop (BTSL) defines what the player commonly does when selecting a battle type with the Battle Type Selection Menu.

The loop is described by a scenario with details as follows:

1. In the Battle Type Selection Menu, Player Hao can choose a desired Game Battle Type.
2. Hao selects a battle type by clicking on it in the Battle Type Selection Menu.

5.2.27. Pause-Menu Loop (PML)

The pause-menu loop (PML) defines what the player commonly does when operating in the Pause Menu of DGA's client application. The Pause Menu provides players with the common system actions that can be accomplished at any moment. Though, the buttons available at a specific time depend on the view the menu is called from. The Pause Menu appears on the screen when the player presses the 'Esc' (Escape) keyboard button.

The loop is described by a scenario with details as follows:

1. In the Pause Menu, Player Hao performs the common system actions, which are represented by the following buttons:
    - 'Back to Game' allows the Player to continue the Game.
    - 'Statistics' brings the Player to a list of his current results, achievements, and advancements in the Game.
    - 'Report Bugs' opens the Game's bug tracker.
    - 'Give Feedback' opens the Game's feedback site.
    - 'Quit to Title' takes the Player back to the Main Menu.
    - 'Player Reporting' brings the Player to a menu used to prevent other players from the behaviour that go against the Game's community standards.
2. When the Pause Menu is called from the Main Menu, then the Pause Menu doesn't appear on the screen because its system actions can't be fully accomplished.
3. When the Pause Menu is called from the Battle Loop, then the Pause Menu doesn't contain the button 'Quit to Title'.

5.2.28. Anti-Cheating Loop (ACHL)

The anti-cheating loop (ACHL) defines the Client's actions for implementing anti-cheating policies, rules, and measures to ensure the Game's fairness, equality, justice, honesty, and integrity to all its players. Cheating by the players not only threaten other players but also violates the fair-to-play principle.

The loop is described by a scenario with details as follows:

1. Cheating matters mostly in the Multi-Player User Mode when a cheater gains an unfair advantage over other players.
2.

Cheating is primarily concentrated on the Battle Loop where it benefits the cheater the most, however other loops and the entire Garage-Battle Loop can be cheated as well.

3. Anti-cheating measures form a complex technical system, the most part of which is implemented on the Game's server side. Nevertheless, the Client is the front line of this system.
4. The Client is intended to technically prevent players from cheating, but active countermeasures against cheaters are performed on the server side.

5.3. Objectives and Progression

DGA's core loops are closely related to the players' objectives in the Game and how they progress through it. The goals below define immediate, short-term, and long-term objectives the players could pursue in the Game.

5.3.1. Experience Growth Goal (EGG)

Growth in experience is the most visible goal in the Game. In fact, experience is the key to the Game's gameplay — it opens all the doors and shows how far a player has advanced in the Game. The more experience points the player gained in battles, the more opportunities the Game provides to the player.

5.3.2. Skill Mastery Goal (SMG)

Growth in skills is the second most visible goal in the Game. Battles bring experience that the player can spend to train weapon units' crews and augment their technical and combat performance. The more experienced and skilled is a crew, the more experience points it brings to the player and makes the gameplay immersive.

5.3.3. Weapon Mastery Goal (WMG)

Growth in experience and skills allows the player to create a highly personalised weapon unit, which combines a highly customised armoured warfare with a highly skilled crew. This combination gives the player an opportunity to feel how this warfare acted in real life and how to master it. The better the player is proficient with weapons, the higher the combat performance of the weapon unit is.

5.3.4. Combat Mastery Goal (CMG)

With a powerful, experienced, skilful weapon unit the player can achieve a high level of personal combat effectiveness and to succeed in a game team to get the victory in a battle. The higher the combat performance of the weapon unit is, the higher the combat effectiveness the player has in the game team and the more victories the player can achieve in the game team.

5.3.5. Team Communication Mastery Goal (TCMG)

The ability to effectively and efficiently communicate with other players brings to the player better combat conditions and improves the combat effectiveness of the player and as well of the game team. Team challenges require players to coordinate their efforts, communicate effectively, and support each other to overcome obstacles. The better the players communicate with each other, the higher the combat effectiveness of the game team is.

5.3.6. Tactical Teamwork Mastery Goal (TTMG)

Tactical teamwork and group training are essential for players to learn how to operate efficiently as part of a game team. Tactical teamwork emphasises collaboration, communication, and mutual support, reflecting

the realities of the battlefield where team cohesion is crucial for victory. The more the game team fosters a sense of camaraderie and trust among its players, teaching them to rely on one another to complete battlefield tasks, the more victories the game team gains.

5.3.7. Military Leadership Mastery Goal (MLMG)

Turning players from military commanders into military leaders is the most ambitious goal of the Game. Combat experience and effectiveness guide players towards various leadership qualities such as adaptability, decisiveness, and the capacity to lead by example. Team leaders contribute to efficiency, initiative, and effective combat dynamics. The more experienced, confident, and inspiring the team leader is, the more victories the game team amasses.

5.3.8. Military History Mastery Goal (MHMG)

Engaging players in military history is the covert goal of the Game. Pursuing to give a right balance between historical accuracy, historical authenticity, and dynamic gameplay, the Game aims at creating a sense of agency, immersion, and replayability, as players experience different outcomes, scenarios, and feedback when they play historical warfare on historic battlefields. The more players strive for victory, the better they know their warfare and its history.

6. Game Systems

This section is currently a draft, and is subject to change.

This chapter describes DGA's game systems, some of which players interact with directly, but others reside solely within the Game's model of the world.

6.1. Game Narrative

DGA's game narrative is outlined as follows:

DGA is set in a world of military technology and armed conflicts, where the player turns into a military commander who takes control over a weapon unit and has to find a proper way with it through battles — or die, i.e. quit the Game. No less, but no more. The Game doesn't seek to make the player learn the military history, or memorise the characteristics of different weapons, or master tactical skills, or make money from the player. All this stuff matters, but, as at war, they all serve the one purpose: to help the player to know oneself to overcome oneself to subdue the enemy. That is the art of war.

DGA's gameplay guides the player through the course of military service of the commander who rises from a young commander to experienced one and then to distinguished one. The Game arranges real-world environments and stages of the commander's career around the main area: the battlefield, where the player's character shall gain an edge over the enemy.

As the battles go on, the commander grows in experience, achievements, capabilities, and rises in ranks as well. Now communication and collaboration, tactical teamwork, group training, leadership skills, strategic vision come on the scene. Strong, effective leadership is underpinned by trust among players. Altogether, these qualities foster team camaraderie and embrace players into the Game's community.

6.2. Main Game Systems

In the DGA Game the player interacts with the following main game systems.

Visible main systems with which the player interacts directly are as detailed below:

1. User Mode System
2. Garage System
3. Technology Tree System
4. Weapon Type System
5. Country System
6. Weapon Unit System
7. Crew System
8. Battle Mode System
9. Battle Type System
10. Battle System

11. Team Coordination System
12. Combat Experience System
13. Cash System
14. Team Communication System
15. Teaming System
16. Training System
17. Barracks System
18. Pause-Menu System.

Invisible main systems, which reside within the game world hiddenly, are as detailed below:

1. Anti-Cheating System
2. Matchmaking System
3. Projectile Impact System
4. Battlefield System
5. Mob System.

6.3. Interactivity

In the DGA Game the player deals with the following forms of interaction.

Visible forms of interaction, which the player uses directly, are as detailed below:

1. Loading Screen with Progress Bar
2. Main Menu
3. Options Menu
4. Singleplayer Menu
5. Multiplayer Menu
6. Edit Server Parameters Menu
7. Garage View
8. Technology Tree View
9. Weapon Type Selection Menu
10. Country Selection Menu
11. Weapon Unit Selection Menu
12. Crew Selection Menu
13. Barracks View
14. Crew Recruitment Menu
15. Crew Training Menu

16. Battle Mode Selection Menu
17. Battle Type Selection Menu
18. Battle View
19. Weapon Selection Menu
20. Ammunition Selection Menu
21. Battle Aiming Mode Selection Menu
22. Battle Signal View
23. Battle Score Menu
24. Weapon Unit Upgrade Menu
25. Weapon Unit Custom Menu
26. Crew Custom Menu
27. Cash Budgeting View
28. Buy Menu
29. Team Communication Menu
30. Teaming View
31. Pause Menu
32. Statistics Window
33. Report Bugs Menu
34. Give Feedback Menu
35. Player Reporting Menu.

Invisible forms of interaction (typed, gestural, mouse-based, etc.), which are not visible on the screen, are as detailed below:

1. Weapon Unit Movement Control
2. Weapons Control
3. Struggle Aiming Mode
4. Sniper Aiming Mode
5. Grazing Aiming Mode
6. Overhead Sight
7. Grazing Sight
8. Auto-Aiming Feature.

7. Process Concept

This section is currently a draft, and is subject to change.

DGA is developed as free/libre/open-source software according to the free/libre-to-game model.

This chapter describes the process characteristics of the DGA Game.

7.1. Team

As in any free/libre/open-source project, the DGA Game has more members than meets the eye. We thank them all for their ongoing contributions to the Game! Hereby the list of those developers, contributors, and participators who have agreed to declare their engagement in the Game:

Id	Name	Role
zss	Zhanat S. Skokbayev	CEO & Founder, Project Leader, Principle Software Architect, Lead Software Developer

7.2. Time

DGA is developed in accordance with the following schedule:

Date	Milestone	Description
03.03.2017	Official Start Date	DGA MMORPG Project established.
15.08.2017	Milestone 1 - Concept Design	DGA Game Client Prototype implementation started.
19.08.2018	Milestone 2 - First Prototype	DGA Game Client Prototype v.0.1 released.
22.01.2019	Milestone 3 - Complete Prototype	DGA Game Client Prototype v.0.2 released.
01.02.2019	Game Client	DGA Game Client implementation started.
25.03.2020	Milestone 4 - Game Engine	DGA Game Client's Game Engine implemented.
29.12.2021	Milestone 5 - Game Client	DGA Game Client v.0.0.1 implemented.
10.01.2022	Game Server	DGA Game Server implementation started.
18.09.2023	Milestone 6 - First Play	DGA Game Client and Server v.0.1 implemented.
18.10.2023	Official Start for Publication	DGA Game started preparing for publication.
20.01.2024	Milestone 7 - Website	DGA Game's Website updated for publication.
22.01.2024	Milestone 8 - v.0.1	DGA Game Client and Server v.0.1 formed for publication.
13.08.2024	Milestone 9 - Web Domain	DGA Game's Web Domain distantgroundattack.org registered.
10.04.2025	Milestone 10 - Documentation	DGA Game Documentation's publication started.
Q2 2025	Milestone 11 - Release v.0.1	DGA Game Client and Server v.0.1 released.

Glossary of Terms

This section is currently a draft, and is subject to change.

Application Programming Interface (API)

A way for two or more computer programs or components to communicate with each other. It is a type of software interface, offering a service to other pieces of software. A document or standard that describes how to build or use such a connection or interface is called an 'API specification'. (Wikipedia)

Battle

In gaming, a battle (game battle, or battle session) is an occurrence of combat in warfare between opposing teams of players of any number or size.
In general, a battle is a military engagement that is well defined in duration, area, and force commitment. A war usually consists of multiple battles. An engagement with only limited commitment between the forces and without decisive results is sometimes called a skirmish. (Wikipedia)

Battle Map

In gaming, a battle map, or battlefield, is a background layout of the battle system themed to the location of the battle. (Wikipedia)

Battle Rhythm

Battle rhythm refers to the smooth operation of a team through a schedule of command, staff, and unit activities to maximize efficiency and synchronicity. (Military REACH)

Battle Unit

A weapon unit, which passed at least one battle.

Bot

A bot, short for robot, in a video game is a non-playing character (NPC) controlled by a software application with the intent to imitate human activity. In a client–server model, a game bot plays the client role and imitates a player accessing the game server. Game bots are able to perform simple and repetitive tasks much faster than a person could ever do, and because of that they are treated as a form of cheating, which breaks the principles of fair play.

Buff

1) An effect placed on a video game character that beneficially increases one or more of their statistics or characteristics for a temporary period. Contrast with Debuff.
2) A change intended to strengthen a particular item, tactic, ability, or character, ostensibly for balancing purposes. Similar to Upping. Related to Balance. (Glossary of video game terms)

Cash

DGA's in-game currency.

Client/Server Architecture

A client/server gaming architecture refers to a typical distributed architecture for the support of networked games. In this architecture, a single node plays the role of the server, i.e., it maintains the game state and communicates with all other nodes (the clients). The server notifies game moves generated by players and computes the game state updates. (ECGG, p. 319)

Closed-Source Proprietary Software (CSPS, PS)

Software that grants its creator, publisher, or other rightsholder or rightsholder partner a legal monopoly by modern copyright and intellectual property law to exclude the recipient from freely sharing the software or modifying it, and – in some cases, as is the case with some patent-encumbered and EULA-bound software (End-User Licence Agreement or EULA) – from making use of the software on their own, thereby restricting their freedoms. Proprietary software is a subset of non-free software, a term defined in contrast to Free/Libre and Open-Source Software; non-commercial licences such as CC BY-NC are not deemed proprietary, but are non-free. Proprietary software may either be closed-source software or source-available software. (Wikipedia)

Combat Effectiveness

The readiness of a military unit to engage in combat based on behavioral, operational, and leadership considerations. Combat effectiveness measures the ability of a military force to accomplish its objective and is one component of overall military effectiveness.

The effectiveness of a military unit in performing its mission depends on its capabilities (including equipment and personnel) and its ability to use those capabilities. Soldiers must be instructed in the use of their weapons as well as in the battlefield tactics needed to fight as a coordinated team. They also must be trained to follow orders and to make difficult decisions under intense pressure. Indoctrination and leadership play key roles, as a soldier must know his or her role and be willing to perform it. Officers must be able to bring out the best in their troops and know how to motivate them to become an effective fighting force. Thus, simply having a large or well-equipped force does not guarantee success on the battlefield. Military planning — identifying the adversary and developing a strategy that combines the most appropriate weapons, unit types, and combat plan to implement against that particular adversary — also plays an important role in combat effectiveness, as well as in military effectiveness more broadly. (Encyclopaedia Britannica)

Crew

A body or a group of people who work at a common activity, generally in a structured or hierarchical organization. A location in which a crew works is called a crewyard or workyard. Members of a crew are often referred to by the title crewman or crew-member. (Wikipedia)

Crew-Served Weapon

Any weapon system that is issued to a crew of two or more individuals performing the same or separate tasks to run at maximum operational efficiency, as opposed to an individual-service weapon, which only requires one person to run at maximum operational efficiency. (Wikipedia)

Cross-Platform By Design (CPBD) Approach

An approach when software products and capabilities have been designed and developed to provide users with a consistent and intuitive interaction across various digital mediums and platforms. The approach encompasses visual consistency, functionality, performance, and accessibility through cross-platform applications on smartphones, tablets, and desktops with the goal to offer an uninterrupted and harmonious user journey. (Medium)

Debuff

1) An effect placed on a character that negatively impacts their statistics and characteristics. Similar to Nerfing. Related to Balance.
2) Effects that nullify or cancel the effects of buffs. Contrast with Buff. (Glossary of video game terms)

DGA Game

The game this document is written for; DGA stands for 'Distant Ground Attack'.

Experience Point (XP)

A unit of measurement used in role-playing games to quantify a player character's life experience and progression through the game. Experience points are generally awarded for the completion of objectives, overcoming obstacles and opponents, and successful role-playing. (Wikipedia)

Farming

In gaming, performing repetitive actions to gain experience, points, or some other form of in-game currency. (ECGG, p. 744)

Fair Play

A behaviour that is fair, honest, and does not take advantage of people. (The Cambridge Dictionary)

Fair-To-Play (FRTP, FR2P) Model

A conceptual model, which is introduced by The FLEISS Software Foundation into the field of video games and their implementation. The model supposes that a game must not only propose to players fair rules, equal opportunities, and just rewards for their participation in the game, but provide them with technical means to accomplish independent verification whether the game still abides by the principles of fairness, equality, justice, honesty, and integrity to its players. In other words, the fair-to-play model stipulates for a simple principle to be accomplished: a fair-to-play game must ensure that the best and most skilful player wins, all other causes are neglected.

First Person Perspective (FPP)

First-person, also spelled first person, is any graphical perspective rendered from the viewpoint of the player character, or from the inside of a device or vehicle controlled by the player character. (Wikipedia)

Free-To-Play (FTP, F2P) Model

A business model in which players have access to a significant portion of a game's content without being obliged to pay. In this case, the presence of in-game stores is a key part of the business model. (ECGG, p. 320)

Free-To-Win (FTW, F2W) Model

A business model, which is a variation of Free-To-Play Model, but it supposes that all players of a game – as those who willing to pay, and those who playing for free – they all have equal opportunities within the game and a player can gain an advantage over other players only by his/her proper efficiency and skills. The player can also spend money on in-game items and microtransactions, but they only allow him/her to play more comfortably: complete the game faster, gain more experience, get more game currency for the same actions, change appearance, etc. The business model was first popularised by the company Wargaming in its game World of Tanks in 2013. (Wikipedia)
In fact, this model can't fulfill its promises because of the lack of transparency: players have no means to

verify neither other players don't have unfair advantages over them nor the game server gives preferences to some categories of players. Nowadays, World of Tanks is considered as a freemium game. (ECGG, p. 2097)

Free/Libre and Open Source Software (F/LOSS, FLOSS)

Software that is available under a licence that grants the right to use, modify, and distribute the software, modified or not, to everyone free of charge. The public availability of the source code is, therefore, a necessary but not sufficient condition. F/LOSS is an inclusive umbrella term for free/libre software and open-source software. F/LOSS is the opposite of Closed-Source Proprietary Software, which is licensed under restrictive copyright and has the source code hidden from its users. (Wikipedia)

Free/Libre and Open Source Software Licences

A *software licence* is a legal instrument governing the use or redistribution of software. (Wikipedia)
A *free-software licence* is a notice that grants the recipient of a piece of software extensive rights to modify and redistribute that software. These actions are usually prohibited by copyright law, but the rights-holder (usually the author) of a piece of software can remove these restrictions by accompanying the software with a software licence which grants the recipient these rights. Software using such a licence is free software (or free and open-source software) as conferred by the copyright holder. Free-software licences are applied to software in source code and also binary object-code form, as the copyright law recognises both forms. (Wikipedia)
Open-source licences are software licences that allow content to be used, modified, and shared. They facilitate free/libre and open-source software development. Intellectual property laws restrict the modification and sharing of creative works. Free/libre and open-source licences use these existing legal structures for an inverse purpose. They grant the recipient the rights to use the software, examine the source code, modify it, and distribute the modifications. (Wikipedia)

Free/Libre-To-Play (FLTP, FL2P) Model

A business and development model, which is introduced by The FLEISS Software Foundation as an implementation of Fair-To-Play Model. The model supposes that a game must propose a fair contract to all sides engaged into the game's development and interaction around and within the game. The contract must not only define fair rules, equal opportunities, and just rewards, but also provide the ability to control how well the contract is fulfilled. The model affirms that this ability can only be provided by usage of free/libre/open-source software for the game's implementation. Moreover, only free/libre/open-source software tools shall be used during the game's development and probably distribution. Finally, the game must be published as a free/libre/open-source software under appropriate free/libre/open-source licences. Therefore, the source codes of all the game's components must be published and accessible to all engaged sides, especially to players who can verify whether the game is still a fair play. The players must have opportunity to rebuild the game from its source code and at least to access the game's server infrastructure to verify the game's fairness, equality, justice, honesty, and integrity.

Free/Libre-To-Game (FLTG, FL2G) Model

The advanced version of Free/Libre-To-Play Model introduced by The FLEISS Software Foundation as a further implementation of the Fair-To-Play Model. The free/libre-to-game model is especially adapted for multitier game systems and it requires that not only a game's client but all the game's server components must be free/libre/open-source software. The players must have opportunity to rebuild the game's client and servers from their source codes and reproduce the game's infrastructure to verify its fairness,

equality, justice, honesty, and integrity. The free/libre-to-game model serves as the basis for development and distribution of the DGA Game.

Freemium Game

Freemium is a portmanteau of the words 'free' and 'premium', thereby, a freemium game is a free-to-play game for which fees are charged for additional features and services. (ECGG, p. 2097)

Game, Video Game, Computer Game

A systemic experience that takes place in its own context, separated from the rest of life (the game's *magic circle*). Every game is characterised by the following attributes:

1. Having its own *rules*, whether formal or tacit and dynamic
2. Requiring voluntary, non-obligatory *interaction and participation* (not simply observation)
3. Providing players with interesting, meaningful *goals, choices, and conflict*
4. Ending with some form of recognizable *outcome*, which is considered better than other outcomes, typically codified in the game's formal rules (*valorisation of the outcome*)
5. As a product of a design process, the game has specific parts that are implemented in some form of *technology* (whether digital or physical); *loops* formed by the behavioral interactions of those parts; and experiential (dynamic, dramatic) *wholes* in the game as played when interacting with the player. (AGDS, p. 95)

Game Balance

A property of the overall game+player system, so it incorporates the player's mental model and the game's model, involving psychology, the game's systems, and mathematical and other tools needed to evaluate them. It includes aspects of the game's progression for each player and the game's progression as a whole: whether the player advances too quickly and easily or too slowly with too many impediments and whether all players progress at about the same rate without feeling as if they are in lockstep with each other. A game that is balanced avoids having one narrow dominant winning path or strategy or creating situations where one player has an inherent or insurmountable advantage. (AGDS, p. 296)

Game Cheating

Any behaviour that a player uses to gain an advantage over his/her peer players or achieve a target in an on-line game is cheating if, according to the game rules or at the discretion of the game operator, the advantage or the target is one that he/she is not supposed to have achieved. (ECGG, p. 742) However, this definition of cheating doesn't imply situations when the game cheats its players, i.e. the situations when the game owners pursuing their financial interests intentionally break the declared rules of their own game, or they manipulate the game's undeclared rules and conditions with the purpose to give hidden preferences to some categories of players. Moreover, ensuring equality and fairness in games is usually treated as issues of game balance and not, a priori, as a result of the game owners' intentions. The FLEISS Software Foundation considers this interpretation of cheating as one-sided and unfair.

Game Community

A group of people that is centered on interaction through and united in activities around a game. A *community* is a group of people united by a common identity and collective purpose who engage in activities together over time. Communities develop unique sets of shared values, principles, and norms

that distinguish them from others and provide members with a sense of belonging.
An *open source software community* is a group of people united by the shared purpose of developing, maintaining, extending, and promoting a specific body of open source software. These communities are often globally distributed — their members occupy different geographic regions and work across numerous industries. What unites them is their common vision for the open source software project — as well as the spirit of camaraderie and collective identity that participating in the community affords them. (TOSW)

Game Core Loop(s)

Interactive loops between the player and game. The player forms an intent and carries out an action, providing input to the game. This causes a change to the game's internal state, and the game provides feedback to the player as to the success or other effects of his/her actions. Typically this feedback also provides the player with information about his/her progression in the game or another reward or another form of call to action to keep the player engaged with the game. The new abilities afforded by his/her progression or reward encourage the player to form a new goal or intent, and the cycle begins again.
This cycle can take place at many different levels of attention and over different lengths of time. A game's core loops are determined by the game's design and, in particular, by what form of interaction is the most significant for player engagement. This almost always includes the low-level action/feedback loop, as this is where the player and the game truly interface: the player performs an action like pressing a key, moving a mouse, or tapping a screen, and the game responds with feedback in acknowledgment.
This action/feedback loop may not be the most significant loop in the game+player system, however. The game design determines which types of interaction demand the player's primary focus, and those then form the core loops. Moving around in the game may be the primary point (often along with jumping, shooting, and so on), or it may be only a means to an end. If the player is mainly focused on constructing buildings, discovering technologies, administering an empire, or building relationships, those forms of interaction will create the core loops for the game. (AGDS, p. 156-157)

Game Design, Video Game Design

An iterative systemic process between designing the parts, the loops, and the whole of a game. Nowadays, the term 'game design' is used synonymously to 'video game design' unless explicitly stated otherwise, since the current game market is dominated by digital media. (AGDS, p. 177)

Gameplay

A playful experience that is communicated to a player as an emergent effect of structural, functional, architectural, and thematic elements of a game. (AGDS, p. 97)

Game+Player Loop, Core Gameplay Loop

The whole of the game+player system arises as an emergent effect of the interactive loops between the player and game. (AGDS, p. 129)
In systemic terms, the player's and the game's behaviours are the result of their internal state. Based on their state, each selects actions to take, which then affect and perturb the other's state. This drives new behavioural responses in return. The player provides input to the game via his/her behaviour, which change the game's state. The game processes this and provides feedback responses that are input for the player, changing his/her internal state. This creates a reciprocating loop that is the essence of interactivity. This give-and-take between the player and the game is referred to as the game's *core gameplay loop*. (AGDS, p. 127)

The core gameplay loop consists of multiple interactive loops between the player and game, which are called the game's core loops and loosely defined as 'what the player does most of the time' or 'what the player is doing at any given time'. (AGDS, p. 156)

Game Studio, Video Game Studio

A video game developer specialising in design and development of computer games, which is the process and related activities of software development and creation of video games for computers. A game studio can be a company, non-commercial organisation, team, or individual developer.

Game Team

In team-based games, teams of players compete with one another to obtain victory. There are a variety of possible team structures, including symmetrical teams like 2 vs 2 players, 3 vs 3 players, etc.; multiple sides teams like 2 vs 2 vs 2 players; the player vs all other players; all players against each other as individual players, pairs of players, or a number of small squads (battle royale, last man standing, or last team standing games), and so on. (Wikipedia)

Grinding

In gaming, the process of engaging in repetitive and time-consuming tasks before being able to advance. (ECGG, p. 744)

Hard-Coding

Hard coding (also hard-coding or hardcoding) is the software development practice of embedding data directly into the source code of a program or other executable object, as opposed to obtaining the data from external sources or generating it at runtime. Hard-coded data typically can be modified only by editing the source code and recompiling the executable, although it can be changed in memory or on disk using a debugger or hex editor. Data that is hard-coded is best suited for unchanging pieces of information, such as physical constants, version numbers, and static text elements.

Soft-coded data, on the other hand, encodes arbitrary information through user input, text files, INI files, HTTP server responses, configuration files, preprocessor macros, external constants, databases, command-line arguments, and is determined at runtime. (Wikipedia)

Historical Accuracy

Historical accuracy refers to the extent to which historical accounts, narratives, and interpretations faithfully represent the facts and events as they occurred in the past. This concept is crucial for understanding the developments and transformations that have shaped societies as it underscores the importance of reliable sources and evidence in constructing a true representation of history. (FiveAble.me)

Historical Authenticity

Historical authenticity in fiction and dramaturgy refers to the accurate representation of historical events, figures, and contexts in performance and production, ensuring that the portrayal is faithful to the time period and cultural nuances being depicted. This concept is crucial as it impacts how audiences perceive and engage with the material, affecting both the educational and emotional aspects of a theatrical experience. (FiveAble.me)

Internationalisation

Often abbreviated as i18n, internationalisation is the process of designing a software application so that it can be adapted to various languages and regions without engineering changes. (Wikipedia)

Intransitive Game Balance

Intransitive game systems may seem to be inherently unbalanced, but it is just a different way of achieving overall balance in the system and in the game. Rather than balancing all the parts against each other, as with transitive balance (making all parts effectively equal at the system level), with intransitive balance, parts are balanced on the basis of their costs and benefits. Some parts necessarily have greater benefits, but they also have proportionately higher costs. (AGDS, p. 316)

Jakarta EE Platform

Jakarta EE (formerly known as Java Platform, Enterprise Edition (Java EE) and Java 2 Platform, Enterprise Edition (J2EE)) is a set of specifications, extending Java Standard Edition with specifications for enterprise features such as distributed computing and web services. Jakarta EE applications are run on reference runtime implementations, represented by microservices or application servers, which handle transactions, security, scalability, concurrency and management of the components they are deploying.

Java EE Platform was initially developed by Sun Microsystems, later maintained by Oracle Corporation under the Java Community Process (JCP), and finally on 12 September 2017 submitted to the Eclipse Foundation. In 2018, Java EE Platform was renamed to Jakarta EE Platform.

Jakarta EE is defined by its specifications, which comprises APIs and their interactions software providers must meet in order to declare their products as 'Jakarta EE compliant'. (Jakarta EE at Eclipse Foundation)

Java Development Kit (JDK)

A distribution of the Java Platform, which implements the Java Language Specification (JLS) and the Java Virtual Machine Specification (JVMS), and provides the Standard Edition (SE) of the Java Application Programming Interface (API).

Java Platform

Java Platform is a set of computer software and specifications that provides a software platform for developing application software and deploying it in a cross-platform computing environment. Java Platform is used in a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers. (Java Platform's Home)

Java Platform also comprises a suite of programs that facilitate developing and running programs written in the Java Programming Language. A Java Platform's implementation includes an execution engine (called a Java Virtual Machine), a compiler and a set of libraries (called a Java Development Kit); there may also be additional servers and alternative libraries that depend on the requirements. The Java Platform's implementations have been developed for a wide variety of hardware and operating systems with a view to enable Java programs to run identically on all of them.

Java Platform consists of several parts intended for different classes of device and application domains:

- Java Standard Edition (Java SE) for general-purpose use on desktop computers, servers, and similar devices.
- Jakarta EE (Java Enterprise Edition) that is Java SE plus various application programming interfaces (APIs) which are useful for multi-tier client-server enterprise applications.
-

Java Micro Edition (Java ME) specifies several different sets of libraries (known as profiles) for devices with limited storage, display, and power capacities. It is often used to develop applications for mobile devices, handheld devices, television set-top boxes, printers, etc.

- Java Card is a technology that allows small Java-based applications (applets) to be run securely on smart cards and similar small-memory devices.

Java Platform, Standard Edition (Java SE)

A computing platform for development and deployment of portable code for desktop and server environments. The platform uses the Java Programming Language and is part of the Java Platform. Java SE defines a range of general-purpose APIs – such as Java APIs for the Java Class Library – and also includes the Java Language Specification and the Java Virtual Machine Specification. OpenJDK is the official reference implementation at the present time. (Java Platform's Home)

Java Programming Language

Java is a high-level, class-based, object-oriented, general-purpose programming language originally developed by James Gosling at Sun Microsystems. Compiled Java code can run on all platforms that support Java without the need to be recompiled. It means that software applications written on Java are typically compiled to bytecode that can run on any Java Virtual Machine (JVM) regardless of the underlying computer architecture. Java has a complete language specification, which was firstly published in 1996 and remains updated until nowadays. (Java Programming Language Documentation)

Java Virtual Machine (JVM)

A virtual machine that enables a computer to run Java programs as well as programs written in other languages that are also compiled to Java bytecode. The JVM is detailed by a specification that formally describes what is required in a JVM implementation. (Java Platform's Home)

jMonkeyEngine Game Engine (jME, JME)

A modern, open-source, cross-platform, developer-friendly game engine written primarily on Java. jME is intended for developing 3D games on Java, which can be run on Windows, Linux, macOS, Raspberry Pi, Android, and iOS. jME's minimalistic and code-first approach makes it perfect for developers who want the game engine's support while retaining full control over their code with the ability to extend and adapt the engine to their workflow. jME uses Lightweight Java Game Library (LWJGL) as its default renderer, although supports other renderers based on Java OpenGL. (jMonkeyEngine Project)

Kerckhoffs's Principle

Kerckhoffs's principle of cryptography was stated by the European cryptographer Auguste Kerckhoffs in 1883. The principle holds that a cryptosystem should be secure, even if everything about the system, except the key, is public knowledge. This concept is widely embraced by cryptographers, in contrast to the 'security through obscurity' principle, which is not. (Wikipedia)

The principle states that it is good design to assume the enemy knows the details of your algorithm, because eventually they will. Auguste Kerckhoffs first stated this thesis in: There is no secrecy in the algorithm, it's all in the key. (SANDL, p. 112)

Localisation

Often abbreviated as l10n, localisation is the process of adapting internationalised software for a specific region or language by translating text and adding locale-specific components. Localisation (which is

potentially performed multiple times, for different locales) uses the infrastructure or flexibility provided by internationalisation (which is ideally performed only once before localisation, or as an integral part of ongoing development). (Wikipedia)

Massive Multiplayer Online Game (MMOG)

A multiplayer game, which is especially designed to be played online simultaneously by a large number of players. (ECGG, p. 596)

Massive Multiplayer Online Role-Playing Game (MMORPG)

A multiplayer game, which is especially designed to be played online simultaneously by a large number of players who are allowed to fill and act out certain roles the gameplay provides. (ECGG, p. 596)

Matchmaking

In multiplayer video games, matchmaking is the process of connecting players together for online play sessions.

Minie Game Engine

A game engine, which is based on the jMonkeyEngine Game Engine and aims to improve the integration of Bullet Real-Time Physics Simulation and Khaled Mamou's V-HACD Library into jMonkeyEngine. The project is named after Claude-Étienne Minié, who in 1846 developed an improved bullet for rifles. The preferred English pronunciation is roughly "min-YAY", but "MIN-ee" is also acceptable. (Minie Project)

Minimum Lovable Product (MLP)

A customer-centered product that users love from the start, defined by its features offering the minimum a product needs to be loved.

Minimum Marketable Product (MMP)

The version of the software product ready to be sold to end users. Usually, the MMP is viewed as the simplest product that the market will accept before new features are added.

Minimum Viable Product (MVP)

The simplest version of a new software product that you need to build to sell it to a market; this version allows the development team to collect the maximum amount of validated learning about customers with the least effort.

Mob

A mob, short for mobile or mobile object, is any computer-controlled non-playing character (NPC) existing in a video game. Mobs can be hostile, friendly, or neutral NPCs. (Wikipedia)

Multiplayer Game

A game that is designed for multiplayer mode where two or more players are expected throughout the entire gameplay. (ECGG, p. 596)

Nerfing

A change, usually a patch, intended to weaken a particular item, tactic, ability, or character, ostensibly for balancing purposes. Similar to Debuff. Contrast with Upping. Related to Balance. (Glossary of video game terms)

Non-Playing Character (NPC)

Non-playing characters (also called non-player characters, or non-playable characters) are broadly defined as visible components of a game that are under the control of the computer, and that either work with or against the human player. (ECGG, p. 1099)

Pay-To-Play (PTP, P2P) Model

A business model, when a dedicated server permits to control the access of users to the gaming servers and to reduce (or avoid) the game piracy. Moreover, this server allows to profile users and offer additional services related to the game. An example is in-game stores in which gamers can purchase items and services. (ECGG, p. 319)

Pay-To-Win (PTW, P2W) Model

A business model, when players who are willing to pay for special items, downloadable content, or to skip cool-down timers may be able to gain an advantage over those playing for free who might otherwise hardly be able to access said items. In general, a game is considered pay-to-win when a player can gain any advantage over their non-paying peers. (Wikipedia)

Player

A companion to the game: without the player, the game itself still exists, but gameplay exists only when game and player come together (the game+player system). (AGDS, p. 97)

Player Character (PC)

A player character (also known as a playable character or PC) is a fictional character in a video game whose actions are controlled by a player rather than the rules of the game. The characters that are not controlled by a player are called non-playing characters (NPCs). The actions of non-player characters are typically handled by the game itself in video games. The player character functions as a fictional, alternate body for the player controlling the character. (Wikipedia)

Play-To-Earn (PTE, P2E) Model

A business model, also known as pay-to-earn, which is very similar to Pay-To-Win Model except it requires using of cryptocurrency and other blockchain technologies for monetisation. (Wikipedia)

Point of View (POV)

Someone's perspective or angle on a specific situation or topic.

Proof of Concept (PoC)

A software system, which implements some concept within some technology stack in a solution and demonstrates feasibility of this solution for the target business purposes and users; this software system can be as small and complete as sufficient to confirm its viability and the likelihood of the implementation's success. In computer game development, tech and game demos serve as proof of concept, so they can demonstrate graphical and gameplay capabilities of the future game.

Proof of Value (PoV)

A software system, which seeks to show how the solution can solve concrete business problems and bring significant benefits to the company and end users. This software system often involves testing in real life scenarios to quantify efficiency gains, cost savings, customer satisfaction, and other commercial benefits.

The findings made during the proof of value's elaboration are later implemented in prototypes and the minimum viable product.

Prototype

An early implementation of a software product or information system, built for demonstration purposes and/or as part of the development process. Although the terms proof of concept and prototype are used interchangeably, but in fact they imply different results and serve different purposes. The purpose of a proof of concept is to help decide whether the idea is feasible or not, and to ensure it will work as intended. On the contrary, the purpose of a prototype is to test the usability, functionality and design of a working model.

Role-Playing Game (RPG)

A genre of video games that allow the player to fill and act out certain roles the gameplay provides. (ECGG, p. 99)

Secure By Design (SBD) Approach

An approach when software products and capabilities have been designed and developed to be foundationally secure. (Wikipedia)

Single-Player Game

A game that is designed for single-player mode where only one player is expected throughout the entire gameplay. (ECGG, p. 847)

System

From the bottom-up view, a system is a set of parts that together form loops of interaction between them to create a persistent 'whole'. The whole has its own properties and behaviors belonging to the group but not to any single part within it. (AGDS, p. 50)

From the top-down view, a system is the integrated whole that arises out of independent, interacting parts. Those parts have their own internal state, boundaries, and behaviors by which they mutually affect each other. This whole persists over time, adapts to external conditions, and has its own coordinated behaviors that emerge from the interactions of its parts. The system can both contain lower-level subsystems within it and be itself part of a higher-level supersystem. (AGDS, p. 86)

Technology Tree

In gaming, a technology tree, tech tree, or research tree is a hierarchical visual representation of the possible sequences of upgrades a player can unlock (most typically representing the research progress of a given faction). (Wikipedia)

The FLEISS Software Foundation

The project, which stands behind the DGA Game. The FLEISS Software Foundation is a non-profit established to advance development of Free/Libre and Open Source Software of enterprise quality and to extend its usage in everyday life. The word 'FLEISS' stands for 'Free/Libre Enterprise Information Software System'. The preferred English pronunciation is ['fleis], but ['flais] is also acceptable. In the German language the word 'Fleiß m.' (Fleiss m.) means 'diligence, industriousness, studiousness'. (The Cambridge Dictionary)

Third Person Perspective (TPP)

Third-person, also spelled third person, is a graphical perspective rendered from a fixed distance behind and slightly above the player character. (Wikipedia)

Transitive Game Balance

Game systems made of parts with *transitive relationships* are those where every part within the system is better than one but inferior to another. The ancient game of *Rock-Paper-Scissors (RPS)* is a pervasive example of *transitive balance*: rock crushes scissors, scissors cut paper, paper covers rock. Each part in the system both overcomes another and is beaten by a different one. No one part predominates or beats all the others. (AGDS, p. 311-312)

Upping

A change intended to strengthen a particular item, tactic, ability, or character, ostensibly for balancing purposes. Similar to Buff. Contrast with Nerfing. Related to Balance. (Glossary of video game terms)

Weapon Unit

A weapons system, which is a combination of one or more weapons with all related equipment, materials, services, personnel, and means of delivery and deployment (if applicable) required for self-sufficiency. (NIST)

XMPP Network Protocol

XMPP is the Extensible Messaging and Presence Protocol, a set of open technologies for instant messaging, presence, multi-party chat, voice and video calls, collaboration, lightweight middleware, content syndication, and generalised routing of XML data. XMPP was originally developed in the Jabber open-source community to provide an open, decentralised alternative to the closed instant messaging services at that time. XMPP has the following advantages such as openness, standardness, provenness, decentralisation, security, extensibility, flexibility, diversity. (XMPP Standards Foundation (XSF))

Bibliography

This section is currently a draft, and is subject to change.

- Bartle R. A. Designing Virtual Worlds // Indianapolis, IN, USA: New Riders Publishing, 2003. – x + 937 p. // https://www.mud.co.uk/richard/DesigningVirtualWorlds.pdf (accessed 10 April 2025)
- Cuofano G. The Free-To-Play Business Model In A Nutshell // FourWeekMBA Blog. – 2024. – 19 June // https://fourweekmba.com/free-to-play (accessed 10 April 2025)
- Encyclopedia of Computer Graphics and Games / Ed. by Newton Lee – Cham, Switzerland: Springer Nature Switzerland AG, 2024. – liv + 2115 p. // https://link.springer.com/referencework/10.1007/978-3-031-23161-2
- Raymond E. S. The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary, Revised Edition // Sebastopol, CA, USA: O'Reilly & Associates, Inc., 2001. – xiv + 242 p. // http://www.catb.org/~esr/writings/cathedral-bazaar (accessed 10 April 2025)
- Schell J. N. The Art of Game Design: A Book of Lenses. – Third Edition. – New York, NY, USA: A K Peters/CRC Press, 2019. – xlii + 610 p. // https://www.taylorfrancis.com/books/mono/10.1201/b22101/art-game-design-jesse-schell
- Schneier B. M. Secrets and Lies: Digital Security in a Networked World. – Hoboken, NJ, USA: John Wiley & Sons Inc., 2016. – 448 p. // https://onlinelibrary.wiley.com/doi/book/10.1002/9781119183631
- Sellers M. J. Advanced Game Design: A Systems Approach. – Hoboken, NJ, USA: Pearson Education Inc., 2018. – xiv + 441 p. // https://www.informit.com/store/advanced-game-design-a-systems-approach-9780134667607
- Stallman R. M. Free Software, Free Society: Selected Essays of Richard M. Stallman, Third Edition // Boston, MA, USA: Free Software Foundation, Inc., 2015. – xi + 293 p. // https://www.gnu.org/doc/fsfs3-hardcover.pdf (accessed 10 April 2025)
- Sun Tzu. The Art of War / Edited with an Introduction by Dallas Galvin / Translated from the Chinese by Lionel Giles, with his Notes and Commentaries from the Chinese Masters. – New York, NY, USA: Barnes & Noble Books, 2003. – 256 p. // https://www.barnesandnoble.com/w/the-art-of-war-sun-tzu/1116627606?ean=9781593080174
- Sun Tzu. The Art of War / Complete Texts and Commentaries / Translated by Thomas Cleary. – Boston, MA, USA: Shambhala Publications, 2011. – 509 p. // https://www.shambhala.com/the-art-of-war-1835.html
- Sun-Tzu. The Art of Warfare / The First English Translation Incorporating the Recently Discovered Yin-ch'ueh-shan Texts / Translated, with an Introduction and Commentary, by Roger T. Ames. – New York, NY, USA: Ballantine Books, 1993. – 336 p. // https://www.penguinrandomhouse.com/books/3172/sun-tzu-the-art-of-warfare-by-roger-t-ames

The Open Source Way 2.0 // The Open Source Way. – Version 2.0. – 2020. – 16 December // https://www.theopensourceway.org/the_open_source_way-guidebook-2.0.html (accessed 10 April 2025)