UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Markus Sulg

Designing a Construction System for Blastronaut Players

Bachelor's Thesis (9 ECTS)

Supervisor: Jaanus Jaggo, MSc

Tartu 2023

Designing a Construction System for Blastronaut Players

Abstract: Base-building is an essential part of the Blastronaut game. The goal of this thesis is to figure out problems of the existing construction system and improve it based on player feedback. The implementation is deployed in two major Blastronaut updates, which were used for user testing. The resting results concluded that the developed systems are impactful as in most cases the usability effectiveness score is rated over 80%. The thesis ends with suggestions for future development.

Keywords: construction system, Steam community, game development, usability testing, Godot engine

CERCS: P170 Computer science, numerical analysis, systems, control

Ehitussüsteemi loomine Blastronaut mängijatele

Lühikokkuvõte: Baasi ehitamine on oluline osa Blastronaudi mängust. Käesoleva töö eesmärgiks on selgitada välja olemasoleva ehitussüsteemi probleemid ja täiendada seda mängijate tagasiside põhjal. Loodud lahendus on kasutuses kahes Blastronaudi versiooniuuenduses ja nende põhjal viidi läbi kasutaja testimine. Testimise tulemusena leiti, et loodud lahendused on kasulikud, kuna enamikel juhtudest oli nende kasutatavuse efektiivsuse skoor üle 80%. Töö lõpus pakuti välja ettepanekud ehitussüsteemi edasi arendamiseks.

Võtmesõnad: ehitussüsteem, Steam kommuun, mänguarendus, kasutatavuse testimine, Godot mängumootor

CERCS: P170 Arvutiteadus, arvanalüüs, süsteemid, kontroll

Table of contents

1 Introduction

An essential part of a game is having a base. Base is a home, which holds the player's items and important machines. Bases are commonly built or upgraded by the player. Games with an in-depth construction system allow the player to shape the base according to their imagination [1].

Blastronaut[1] is a 2D side-scrolling exploration game that takes place in a procedurally generated world. The player's goals are to explore the alien planet, mine resources and build bases. Exploring is guided by a jetpack for faster traveling and a weapon for mining blocks to gain resources (Figure 1). Resources are important for building advanced constructions in abandoned bases. These abandoned bases typically have platforms, which hold some broken machines, that can be repaired. One of these machines, the Main machine, has a construction interface, which is used for constructing machines and blocks. The construction is guided by a blue grid and a transparent texture for the selected object preview.

Figure 1. Blastronaut screenshot from Steam.

Since the 27th of July 2022 Blastronaut is in Steam Early Access[2]. Early Access means that the game is still being developed and it should be considered unfinished. Blastronaut is developed by Jaanus Jaggo, who is hereafter regarded as the main developer. Blastronaut is developed in

[1] https://store.steampowered.com/app/1392650/BLASTRONAUT/
[2] https://store.steampowered.com/earlyaccessfaq/

Godot[3] game engine version 3.4.4. The game is in active development, having a roadmap with major updates planned for every 3 months. This thesis takes part in 2 major updates. Notably there have been 5 prior theses written based on Blastronaut.

In this thesis new systems are implemented on top of the pre-existing construction system. The developed solution focuses on solving Blastronaut problems by analyzing user feedback. The development is deployed in two Blastronaut major updates. Chapter 2 describes the construction system problems that were highlighted by Blastronaut users. Chapter 3 explores solutions to these problems in game design papers and in an existing game Factorio[4]. The previously mentioned background information is used in Chapter 4 to create a general implementation plan. Chapter 5 describes the implementation process with 3 development iterations, where each iteration ends with the analysis of player feedback. Chapter 6 details the results and possible future improvements. Lastly, final conclusions are made in Chapter 7. The appendices include the following items:

- Appendix I - Testing materials and feedback of the conducted usability testing,
- Appendix II - Link to the private Gitlab repository containing game code,
- Appendix III - The license.

1.1 Previous construction system

This chapter describes important components of the previous construction system. The previous construction system was accessed from the Main machine, which had multiple construction options and created a blue grid for placing the selected objects (Figure 2). The Main had two types of construction groups, namely machines and blocks. The latter group, blocks features 6 craftable blocks (Figure 2), which all correspond to a 1x1 block area. The machine group features 13 options that can be grouped into 3 categories: crafting, storing and unique purposes.

[3] https://godotengine.org/
[4] https://www.factorio.com/

Figure 2. Main construction groups, machines on left and blocks on right.

1.1.1 Trade and crafting

Crafting machines have a trade, which is used to exchange players' items for something in return. In other words, trade has an input, which represents the resources taken from the player's inventory and an output, which are the resources the player receives after crafting is completed (Figure 3, left). Notably trade outputs are obtained in two ways. Firstly, a trade output can be received instantly. Secondly, trades can take a certain amount of time to process, after which the player can collect the items. This crafting process shows a green progress bar and some numbers on top of the machine (Figure 3, right). The green bar represents the current craft progression, whereas the left and right numbers display queued and completed crafts accordingly.

Figure 3. Left shows the furnace trade before crafting is queued and right shows the furnace trade in progress.

1.1.2 Trade resources

Trade inputs and outputs are known as trade resources. These resources are categorized into multiple tiers (Figure 4). The lowest tier contains raw resources, which are obtained by either breaking the environment blocks or slaying enemies. Some of these raw resources are metals, which are important for crafting next tier resources. The second tier includes polished resources,

which are then used to craft the advanced tier resources. Finally, there is also a special category for money, which is use

d as an input or output in a lot of trades.

Figure 4. Resources by tiers, generally lower are more advanced.

1.1.3 Crafting machines

Crafting machines are used for crafting most resources. These include furnaces, assemblers, fuel machines and research stations (Figure 5). More advanced machines have more crafting trades. Figure 5 shows less advanced and more advanced machine variations in groups, which get more advanced from left to right. Notably these crafting machines are used to craft different types of resources. Firstly, furnaces are used for crafting polished resources. Whereas assemblers are used to craft advanced tier resources. Notably research stations do not craft a resource, but instead unlock new trades, which can be used in other machine trades. Finally, the fuel machine is used only to craft fuel.

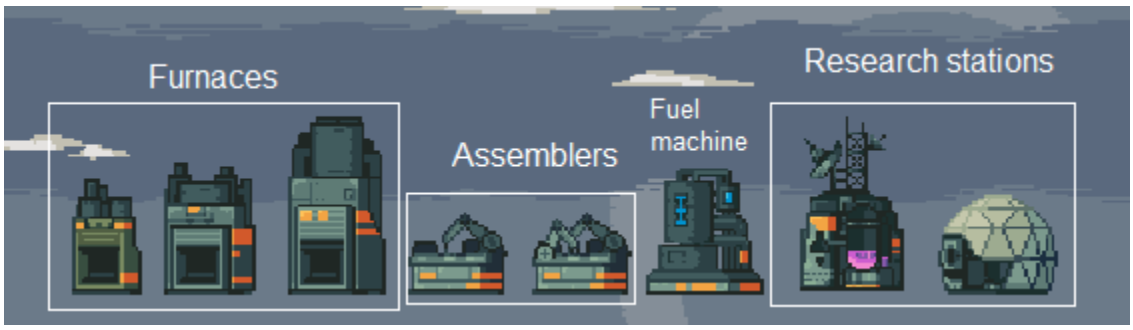

Figure 5. Crafting machines.

1.1.4 Storage machines

Storage machines are used for storing resources or equipment. There are a total of 3 constructable storage machines. Namely the container, tool station, and jetpack station, which allow to store resources, weapons and jetpacks respectively (Figure 6). Notably the container works differently to standard chests. The container can hold only one resource at a time, but makes up for it by not having a maximum storage limit. The tool and armor station are used to switch or equip previously unlocked equipment.

Figure 6. Storage machines.

1.1.5 Unique machines

Unique machines category includes 2 machines and 6 blocks (Figure 7). Firstly, the platform has the main purpose of being an unbreakable floor for machines. Secondly the antenna machine's purpose is to set a new player spawn point. A spawn point in this context means a location, where the player is teleported upon dying.

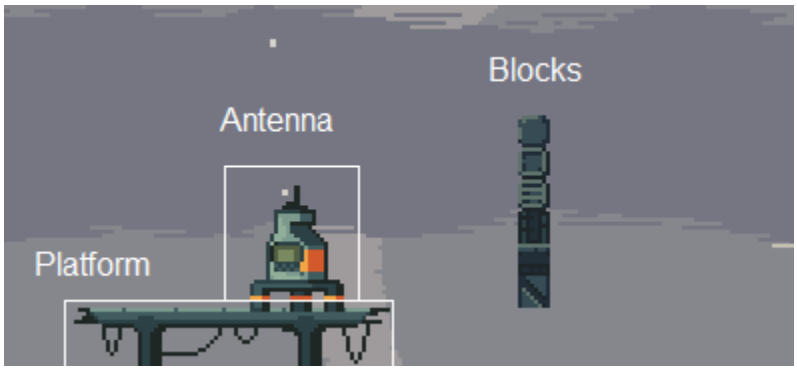

Figure 7. From left to right platform, antenna machine and blocks.

1.1.6 Template editor

In a previous Blastronaut thesis Silver Spitsõn [2] developed a template system. This combines the procedurally generated world with handmade templates in order to create unique locations to the game world. Notably, the author also developed an in-game editor, which significantly simplified the template creation. Templates are most commonly used for creating abandoned bases (Figure 8). An abandoned base is a template, which has some interactable machines. The second use for templates is to diversify the game environment by defining custom block patterns.

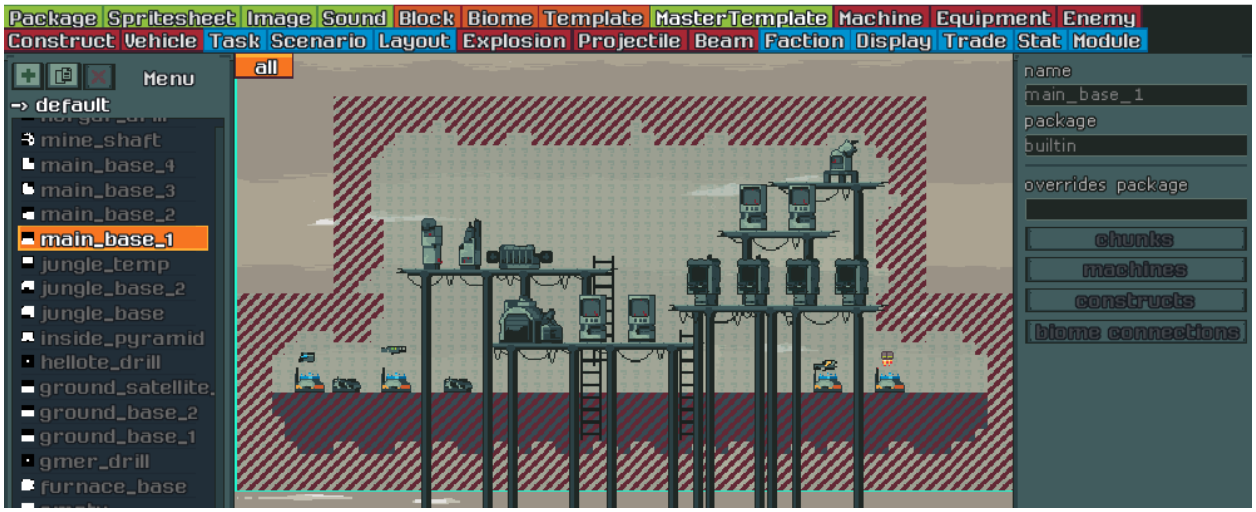

Figure 8. Abandoned base template in the editor.

1.1.7 Rare machines

Abandoned bases can include rare machines, which can not be built by the player. Rare machines can be split up to 3 different types according to their purpose (Figure 9). Firstly, selling machines have a trade, which are used to sell certain resources for money. Notably, these machines are the most reliable way of obtaining money. Secondly, there are buy machines, which sell weapons or jetpacks. Lastly, upgrade machines are used to exchange money for increasing player survivability attributes.

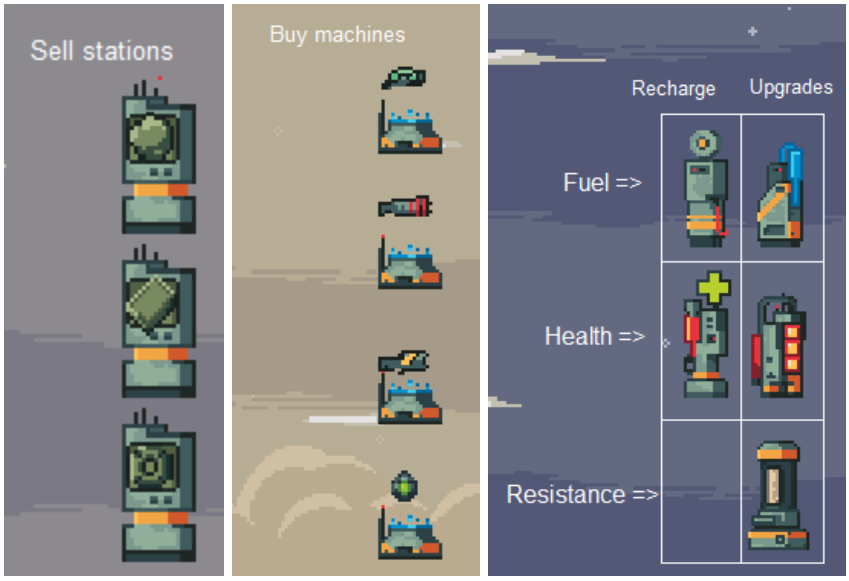

Figure 9. From left to right sell, buy and upgrade machine groups.

2. Problems

This chapter focuses on gathering perceived problems from the Steam community. Studies from Christian Moro et al. [3] and Dayi Lin et al. [4] present the gaming community as a valuable source for improving games. Furthermore, Dayi Lin [4] finds that most Early Access games have more activity, especially on the discussion forums.

Blastronaut game community feedback is gathered from 3 sources, namely the Trello feedback board[5], Steam customer reviews[6], and Steam discussion threads[7]. In total, these feedback sources have 307 posts, 112 reviews, and 149 threads accordingly. The feedback included a large variety of problems, which could be categorized into the following 7 problem categories:

- deletion system,
- fast travel,
- system depth,
- construction boundaries,
- upgrade machine,
- rare machine,
- container.

Specifically, the previous problem categories are derived from a total of 8 Trello posts, 13 reviews, and 13 discussion threads. This feedback included suggestions from a total of 41 different users. The importance of these problem categories were evaluated by total problem mentions. Importantly, the discussion thread allows multiple comments from one user. This means that there is a possibility that the same user can give the same feedback multiple times, for example in a Steam discussion thread and in a Trello post. Table 1 summarizes these problems and uses the column “Total mentions” for counting unfiltered mentions and the column “Total mentions by users” to limit problems to one comment per user. Table 1 is sorted in descending order first by total mentions by user and secondly by total mentions.

[5] https://trello.com/b/ZLXQOhPg/blastronaut-feedback
[6] https://store.steampowered.com/app/1392650/BLASTRONAUT/
[7] https://steamcommunity.com/app/1392650/discussions/

Table 1. Player feedback by problem and feedback source (updated 21th of February).

	Trello	Reviews	Discussions	Total mentions	Total mentions by users
Deletion system	5	1	14	20	20
Fast travel	3	6	8	17	16
System depth	2	8	6	16	14
Construction boundaries	1	5	3	9	8
Upgrade machine	1	2	4	7	6
Rare machine	0	3	3	6	6
Container	2	1	1	4	4

2.1 Feedback analysis

This chapter describes the specific comments and suggested solutions that are associated with each problem category. The problems are summarized in descending order according to total mentions by users.

2.1.1 Deletion system

The most mentioned problem was regarding the deletion system, which included feedback before and after the first implementation iteration. Although, this analysis focuses on the feedback prior to the first iteration. The latter feedback was added only to highlight the importance and frustration of players.

The deletion system feedback consensus was uniform. All the respondents highly requested this feature and were clear that the deletion system needs to be implemented before the game leaves Steam Early Access and is fully released. The deletion system was requested for two use cases, first to move misplaced machines and secondly to delete unwanted machines. The lack of not being able to delete objects caused harsh feedback due to frustration. The frustration was

accelerated even more by high construction costs, which made the misplacement of machines very punishing.

However, there was logic for deleting blocks. Deleting blocks was implemented by using a secondary action when building blocks. For keyboard players this was set to the right mouse button. Some users found this and another separate key binding in the options menu, which was labeled as “remove”. This caused some players to believe that the deletion system worked for machines, which was not the case. These previous partial deletion solutions confused the users and added to the frustration. Users were very hopeful to get a proper implementation of the deletion system.

2.1.2 Fast travel

The fast travel problem is about dissatisfaction with backtracking. Users like to explore as far and as deep as possible. This makes the long journey back difficult to survive and backtracking unpleasant. Some users even pointed out that they do not have a home base, because getting to the base is frustrating. Users suggested multiple ideas to fix this issue. The most popular suggestion included a teleportation building, which instantly teleports the player to a base or any specific location. This idea has been experimented in Blastronaut, but was not implemented, because it disrupted the natural flow of exploration. Some people mentioned classical solutions that are used in many mining games. These solutions include the possible implementation of elevators, minecarts, tubes or trains. Additionally, some users mentioned that horizontal traveling options are exceptionally lacking. Vertical traveling is effectively accelerated by the jetpack, but horizontal traveling does not have any faster option than player movement. One idea to fix this specific issue took inspiration from the jetpack in the form of a rocket board that speeds up the player horizontally. The feedback provided many great ideas, but these ideas require a lot of logistics planning, which could potentially be complex enough to be a separate study.

2.1.3 System depth

The system depth problem is about the general lack of construction systems. This means that users would like new complex systems that would bring more benefits for base-building focused gameplay. Most people pointed out that the construction system had too few options. Notably,

most users complained about high construction costs, which may be the main reason behind this criticism. Users requested that the construction costs would be lowered to be able to build more easily with less resources.

One user pointed out that the limited building options benefit a scavenger playstyle, where moving base to base is more beneficial than building a huge base. The base-centric approach could be improved with templates as described in one feedback, which requested generating larger bases instead of many smaller bases. Similarly, some users would incentivize the base-centric playstyle by adding options to build staircases, bridges, and background wall panels.

Multiple users requested a base-centric playstyle with the addition of an automation system. For example, users suggested the implementation of dynamic tools to connect bases or machines to one another. This way automation pipelines could be formed, which are used to autonomously transport resources. Suggested automation pipeline solutions included use of drones, pipes, elevators. A specific user pointed out that the automation could work by having pipes from containers lead to a furnace. Additionally, the player could set up the automation process by first selecting what the furnace crafts and then filling the containers with the necessary resources. The containers would transport the resources to the furnace and crafting would continue as long as there are enough resources. Notably, one user had a great contribution to this idea by mentioning new digging machines that could create nearly infinite amounts of resources. This digging machine would complement the automation system by providing the necessary starting resources. This automation implementation would benefit the construction system by adding system depth and would motivate a base-centric playstyle.

2.1.4 Construction boundaries

The construction boundaries problem is related to the Main machine construction system design. Multiple users mentioned that it is unconventional for a construction system to be bound to a machine, especially without any zooming and moving capabilities. The players were unhappy that they could not construct very far. Some criticized that building far away from a base can only be accessed by building multiple Main machines in that direction. One user even mentioned that having a building range at all was bad for the construction system. Some suggested solutions

include zooming out, moving around in construction view, building on a map, using a consumable object for building a Main machine, building with a weapon that constructs machines.

This problem was accelerated by the Main machine accessibility. The Main machine is initially locked behind research and expensive. This leads to the same subproblem that was encountered in Chapter 2.1.3. In conclusion, scavenging new bases was more beneficial for the player, than upgrading a previous base.

2.1.5 Upgrade machine

The upgrade machine problem is about users requesting more variety in terms of upgrading existing equipment and machines. Some users mentioned that there were few variations of equipment. The same users suggested that upgrading could be done by attaching some parts to equipment, which provide unique bonuses. Another feedback about modules presented a similar idea. The feedback mentioned a tool station, which would be a machine that allows tools to be customized with different parts. These parts could upgrade the tool stats by providing unique bonuses. Another user suggested that these parts could be found or crafted. The idea of a tool station and crafting parts are worth analyzing and experimenting. Particularly, crafting could be implemented in construction options, which would increase the general construction system depth. The current equipment stations could be expanded with an interface to upgrade equipment. Another good idea came from one user who requested upgrading existing furnaces. Multiple users' ideas complement each other and when combined create a solid foundation for an upgrade system, which should be analyzed and experimented.

2.1.6 Rare machine

The rare machine problem includes a controversial design decision where some machines are made not buildable and immovable for the player. Users mentioned that finding important stations was difficult for them, because the player had to be lucky to find them. Users defined sell stations and upgrade stations as important stations. Firstly, sell stations are important, because multiple users mentioned scenarios where they wanted to sell a specific resource, but they were not able to find the correct sell station. Secondly, the upgrade stations are important,

because the early game is really difficult without accessing health and fuel upgrades. These attributes are generally of importance for survivability. Users suggested that the machine design could be changed, so they could be buildable or movable in some way.

2.1.7 Container

The container problem is a problem category, which was mentioned in a few user's feedback. The feedback included two possible solutions to the container machine. The first requested solution was an instant option for storing items. The users thought that the previous system was too slow. The second problem discussed the stacking of container machines on top of one another. Stacking containers would greatly reduce the room needed, however it would require a more difficult control scheme.

2.2 Filtering problems

This thesis focuses on improving the general construction system. The most popular problems are chosen, except the faster travel problem, which needs further analysis. Hereafter the discussed problems will be filtered to the following:

- deletion system,
- system depth,
- construction boundaries,
- upgrade machine.

3 Similar works

This chapter discusses the possible solutions to Chapter 2.2 problems in 2 parts. The first part covers the solutions in research and the second part analyzes the Factorio construction system.

3.1 Similar research

Construction system design is analyzed according to the system depth, deletion system, construction boundaries and upgrade machine problems. Notably, this research focuses mostly on Ernest Adams' book, which is about construction game fundamentals [1].

3.1.1 System depth

Ernest Adams in his book [1] describes that in construction games the player must understand how to use and manipulate the game's internal economy to produce economic growth. Adams refers to construction as an investment, which eventually pays off. In the case of Blastronaut, the economy largely revolves around money, which is a central resource that can be gained and used for multiple purposes. Alternatively, Maithili Dhule [5] sees base-building game progression in chronological steps, starting with collecting resources, then creating tools, then developing buildings, and finally advancing in world technology. Relatively to Adams' and Dhule's statements the Blastronaut construction system should have ways to construct rapid economic growth with the investment in advanced technology. The most straightforward beneficial implementation is the subsolution of the automation system discussed in Chapter 2.1.3. The automation system would be an investment that eventually greatly amplifies resource production.

3.1.2 Deletion system

Adams [1] briefly mentions deletion in construction games with a statement that construction games generally need a demolition system. Demolition is used to express the player's creative freedom and lacking demolition reduces that freedom. This could be the main reason that the deletion system implementation was heavily criticized. The players could not shape their bases according to their imagination, which caused frustration. Notably, the author mentions 3 important cost types of demolition systems, which are described relative to the player benefit.

Namely, demolition can cost resources, cost nothing or earn some resources back. Adams mentions the latter 2 as the most fitting options with the exception of earning money, which additionally needs to be less than the original build cost. The reasoning behind this is that the player should not be penalized for demolition and on the other hand the player needs some consequences to make resource managing decisions.

3.1.3 Construction boundaries

Importantly both Adams [1] and Dhule [2] define a good construction game as a game, where the player can exercise their imagination and create something unique. This means that there should be only simple boundaries to construction and the player should be able to make something personal. This statement relates to most problems that were present in the Chapter 2.1 feedback. Namely, most users mentioned that there were not enough options to build a personalized base. This problem is best highlighted in the construction boundaries problem, where the feedback mentioned that the Main machine construction interface was too constraining and frustrating to use. This meant that the player's creative freedom was limited, which is a fundamental problem according to Adams [1] and Dhule [2].

3.1.4 Upgrade machine

Dhule [5] mentions that power-ups greatly enhance the gameplay depth. Similarly, Rogers [6] mentions that upgrading characters keeps things interesting. According to the latter author, these power-ups are used to generally boost the character abilities. According to both authors implementing upgrades for equipment and possibly machines is generally beneficial. In particular, upgrades can create new interesting content for Blastronaut and raise the game depth.

3.1.5 User interface

Adams [1] finds it important that construction games have enough analytical tools for the player to understand game processes. On the same note, Rogers [6] regards a heads up display as the most effective way of communicating with a player. In conclusion, both of the authors value some component of a user interface highly in terms of player communication.

Adams additionally points out that a user interface should be easy and enjoyable to use. This ties into another mutual point by Adams [1] and Rogers [6], where the player's view should be clear and unobstructed. Blastronaut has followed this approach by keeping the player heads up display clear of most information. Blastronaut uses context-sensitive prompts, which are regarded by Rogers [6] as effective teaching tools. Finally, Adams [1] mentions that different players prefer different amounts of information on the screen. This means that user interfaces should be involved in testing.

3.2 Factorio

Factorio is a game that has a very deep construction system. It is a complex game that is split into beginner, advanced and expert phases according to the official wiki [7]. Each of the bases use different automated systems, which makes this a good game for reviewing system depth and possibly other solutions. Additionally, the game has been involved in multiple research papers, including Kenneth Reid et al. study [8], which mentions Factorio having a lot of potential academic research problems.

3.2.1 System depth

The beginner phase starts with using 6 resources. These resources include wood, coal, copper ore, iron ore, stone, and water. Most Factorio resources are used to create machines that automate resource production. Mineable resources like ores and the stone are gathered by using a mining drill. The mining drill has a visible output, whose direction is shown as a yellow arrow (Figure 10). The mined resources are needed to be transported to furnaces, which smelt the ore into polished resources. In the beginner phase, transportation is done with conveyor belts to transport resources and inserters for placing and taking resources on the conveyor belt. These 2 constructs are the core for automating resource production, which is effectively designed in Figure 10.

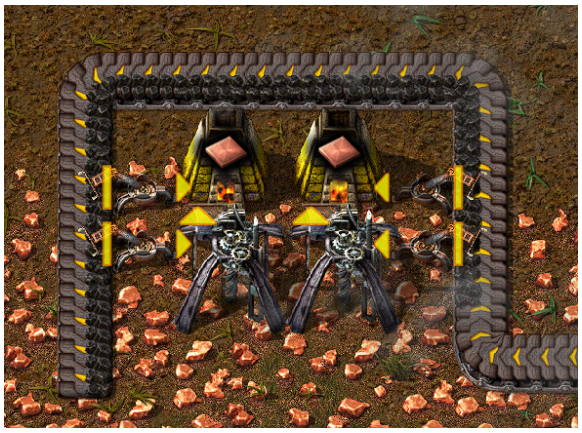

Figure 10. Automation example[8].

Notably, the Factorio game core concept revolves around automation. Importantly, the game's later phases introduce two new automation systems. Firstly, railways are used as an efficient long distance resource transportation method, which can be configured to work autonomously. Secondly, after unlocking robot logistics the player gets access to construction drones, which are autonomous flying robots that are capable of construction tasks. For example, construction drones can be given orders to repair, build or remove structures. The game has multiple automation systems for automation and significantly rewards investing in automation.

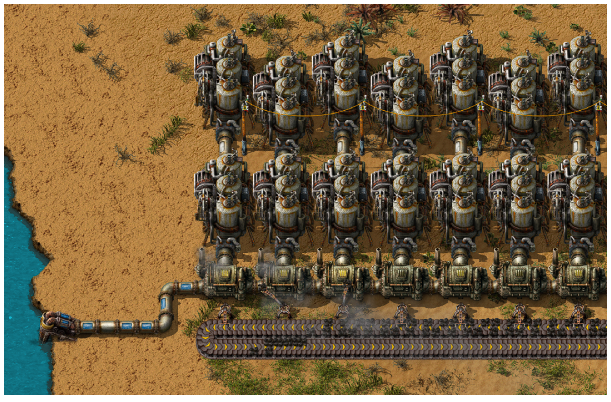

Figure 11. Electrical power setup[9].

3.2.2 Deletion system

Factorio features two variations of the deletion system. The first variant of deletion is player pickup, which allows the player to manually pick up objects to their inventory. Any structure can

[8] https://wiki.factorio.com/Tutorial:Quick_start_guide
[9] https://wiki.factorio.com/Tutorial:Quick_start_guide

be picked up by holding the right mouse button over it. The second variation is the deconstruction planner, which is mostly used in later phases with construction drones. The deconstruction planner is visualized by a red box, which can be dragged on top of multiple objects to stage them for deletion. The staged objects are marked with a red 'X', which will be eventually removed by the construction robots. Similarly to the first option the selected objects are picked up.

3.2.3 Construction boundaries

Factorio construction happens in the user interface, which shows the player inventory on the left and the crafting recipes on the right (Figure 12). The crafting recipes can be used to craft different machines or constructs. Machines and constructs can be placed by selecting them in the inventory. Selecting a construct previews it with a transparent overlay, which is green if the machine can be placed and red otherwise. The buildable area range is around the player, which can not be increased by moving or zooming. Notably the building area range is quite big, because the game is shown in a top-down view, where the default perspective is quite zoomed out.

Figure 12. Thesis author example of the Factorio inventory.

3.2.4 Upgrade machine

Factorio has modules, which are items used to greatly improve the resource production efficiency of crafting machines. The game has three types of modules, namely speed, productivity and efficiency modules. Starting with the speed module, it is used to increase

machine production efficiency per second, but in exchange this module increases the machine electricity cost. Secondly, the productivity module is used to generate some free items occasionally, but similarly increases electricity and additionally slightly lowers production speed. The third efficiency module decreases the overall machine electricity cost. Each previously mentioned module has three different tiers, which increasingly give more bonuses and in exchange also cost more.

4 Goals

This chapter sets specific goals for the Blastronaut construction system implementation. Goals are divided into a general usability evaluation and specific solutions according to problems.

4.1 Usability

As Adams mentioned in his book [1], construction interfaces should be easy and pleasant to use. This is commonly referred to as the software usability metric, which is used for improving user experience [9]. This thesis focuses on the ISO 9241-11[10] definition of usability, which focuses on software effectiveness, efficiency, and satisfaction. This focus is slightly adjusted with Finstad's study [9] on a Usability Metric for User Experience (UMUX). This metric is used as a quick and efficient tool for usability evaluation. Finstad in his research [9] describes usability with the following components:

- effectiveness,
- satisfaction,
- overall,
- efficiency.

The previous usability components need to be considered in implementation and be eventually evaluated in user testing.

4.2 Problem specific goals

This chapter sets implementation goals for each of the previous problems discussed in Chapter 2.2. Additionally the previous problems are restructured according to their respective subproblems that more accurately represent planned solutions.

4.2.1 Deletion system

The deletion system is needed to express the player's full creative freedom and it is an important feature of most construction games according to Chapter 3.1.2. This is why this feature will be implemented according to deletion system player needs, which were discussed in Chapter 2.1.1.

[10] https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en

This includes the general goal of having a system, which can remove misplaced or unwanted machines or blocks. The deletion system is designed to take little inspiration from the Factorio deconstruction planner by marking and unmarking structures into a delete phase. Although this delete phase is different in terms of deletion time. Concerning possible deletion accidents the selected constructs will not be deleted straight away and will be deleted only when construction mode is closed.

4.2.2 System depth

System depth was another popular problem amongst Blastronaut users, which as mentioned in Chapter 2.1.2 can be solved by most other solutions. When reviewing Factorio a big part of system depth was automation, which is planned as the sole implementation for system depth category. Additionally the automation system implementation fits the design goal of construction games as discussed in Chapter 3.1.1. Even more, the automation development was also supported by multiple users in Chapter 2.1.2, which mentioned a simplistic machine connection solution. This feedback and Factorio automation systems covers multiple specific automation solutions, but notably both of these mention drone solutions. The drone design is even more fitting, because the first iteration is about implementing drones, namely the “Companion” update. Another interesting idea was raised in Chapter 2.1.2, which mentioned new digging machines that could complement automation. Factorio features similar machines in its default choice of drilling machines. This drill machine idea should be experimented and tested to see if this automation complementary addition is desired by the player community..

4.2.3 Construction boundaries

Next up is the construction boundaries problem, which had two solutions discussed in Chapter 2.1.3. The first solution is to increase the construction system placement range by zooming out or moving around the Main construction system view. The second solution is to make a new construction mode, which allows the player to move while building. Both solutions are equally supported by Adams [1] and Dhule [5]. They mentioned that a good construction system allows the player to express their creative freedom. Both of these solutions would help construction by allowing more player expression.

4.2.4 Upgrade machine

Finally the upgrade machine problem had multiple users suggesting parts for upgrading equipment stations and crafting machines in Chapter 2.1.4. A similar implementation is developed in Factorio with modules, which allow upgrading crafting machines with 3 module types. The module types affect the crafting process by increasing speed, productivity, or fuel efficiency. A similar idea can be experimented on Blastronaut, using buildable parts, which attach to machines. These modules are planned to work both for upgrading equipment and crafting machines..

5 Implementation

Implementation is split into 3 iterations according to major events in the following chronological order: Blastronaut major “Companion” update, Game Developers Conference expo, Blastronaut major “Aqueous” update. Each iteration describes the implemented solutions and ends with player feedback analysis.

5.1 Blastronaut “Companion” update

This iteration covers development leading up to the Blastronaut “Companion” update, which was hosted in the University of Tartu Delta center as part of an Computer Graphics Projects event on 27th of January. This update featured the main developer integrating drones into Blastronaut. There was an idea to combine the drones with the construction system, which is implemented this iteration with a drone module solution. In general this iteration implemented 2 new systems, namely the deletion and upgrade systems. Finally, this iteration ends with gathering and analyzing in-person user feedback from the Computer Graphics Projects event.

5.1.1 Deletion system

The deletion system first needed a way for notifying players, which machine was hovered. A machine hover shader was implemented that highlights machine borders in white color upon hovering the machine (Figure 13, first from the left).

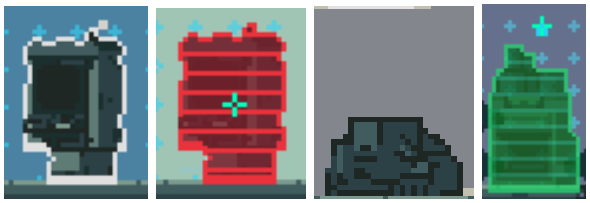

Figure 13. Left to right hover, delete phase, demolition and build effects.

The deletion system was implemented according to design goals, which included a delete phase concept. The deletion phase was set up by choosing a button for triggering the deletion delete phase change. The buttons selected were chosen similar to the already existing block deletion buttons, which were the right mouse button for keyboard and Xbox B button for controllers. Additionally, the delete button under game settings was also added, which added the DEL button

for keyboard players. If any variant of the deletion button is pressed on a machine, then the deletion phase is set, which is animated with a red texture animation (Figure 13, second from the left). Similarly, the previous building texture animation was changed to the same texture (Figure 13, fourth from the left). Additionally, deselecting the delete phase is possible with pressing the same deletion button on the machine. Finally, previously set machines are demolished whenever the Main construction interface is closed, which leaves behind some debris of the original machine (Figure 13, third from the left).

5.1.2 Upgrading system with modules

Upgrading machines is supported with new modular parts called modules. Modules are rectangular objects that upgrade the machine performance according to the module type. This iteration covers 4 different module types including speed, power, drone and fuel modules. These modules were implemented for weapon and jetpack stations to upgrade weapons and jetpacks accordingly.

To attach modules to machines first slots were implemented. Slots are placeholders for showing, where modules can be attached to (Figure 14). Slots were implemented to work with the Blastronaut editor, which was developed by Spitsõn [2]. In the editor it is possible to set the slot locations relative to the machine. The slots are programmatically initialized under a generic machine script.

Figure 14. Slots appearing in construction view and each machine has 1 speed module attached.

Next the module logic was created by creating a decorator class, which is used to attach modules to equipment programmatically. This decorator class is created, whenever equipment is picked up from a machine, applying stacking module effects to the selected equipment. The decorator class implements multiple functions to calculate the effects of particular modules. For example, the drone module effect calculation simply sums up the count of the drone modules. A more

advanced example, the speed module implementation for weapons, decreases weapon shooting delay, which is calculated by an exponentiation of a constant 0.9 value by the amount of speed modules attached. Notably, this calculation logic was added to specific code instances, which can be affected by modules.

Notably, there are 8 module equipment and module interactions, because there are two types of equipment and 4 types of modules. Firstly, the speed module on weapons decreases the fire delay between shots, whereas the jetpacks implementation increases the player's overall movement speed. Secondly, the power module increases a weapon's explosion radius, whereas the same module decreases any damage received for jetpacks. Finally, the drone and fuel modules have similar interactions on both equipment. Notably, the drone module spawns a companion robot that picks up fallen resources, which are in the player's close proximity. This robot was designed by the main developer. Lastly, the fuel module reduces any equipment fuel costs. All of these modules were integrated into the existing construction interface, by making a new interface "Modules" group (Figure 15).

Figure 15. Speed, power, drone and fuel modules shown in the construction view left to right.

Finally, logic for attaching modules to slots was implemented. The previous construction system logic was modified by adding a helper feature for modules. This helper feature is used to automatically attach modules to the closest slot location. However, this feature has a minimum activation range, which is shown in Figure 16.

Figure 16. Module snapping example.

5.1.3 Upgrading system weapon slot changes

The new modules system was complemented by new changes to the weapon slots, which were developed by the main developer Jaanus Jaggo. First the weapon slots were increased from 1 to 3 weapon slots so the module system would be easier to experiment with. Secondly, changing weapons needed input controls, which were chosen to be numerical 1, 2, and 3 buttons and mouse middle wheel up and down scrolling controls for the keyboard and mouse players. Additionally, the weapon UI slots were made interactive to change to the specific weapon. Whereas the first numerical buttons refer to the weapon slots left to right and the mouse middle wheel up and down controls change weapons to right and left weapon slots accordingly. Alternatively the controller buttons were chosen to be the R1 and L1 buttons for the next and previous weapon slots accordingly. Lastly, visual indications were added to equipment to represent attached modules. The visual indications are small circles below the weapon that show the module types with colors accordingly. The modules speed, power, drone, and fuel are shown in Figure 17, which have the yellow, red, gray, and blue colors accordingly.

Figure 17. Circles left to right indicate speed, power, drone and fuel module.

5.1.4 Feedback

Testing was conducted in the University of Tartu delta center, where 4 computers were set up with the Blastronaut previous tutorial. This tutorial did not have direct scenarios that used the new systems. Users were recommended to try out a testing scenario, which was built to be able to show construction system changes. The testing scenario featured a long horizontal base that

notably had weapon and jetpack stations pre-built. Most users were new players, playing the game for the first time and did not like to try out the advanced testing scenario. Notably, these players like the scavenger playstyle of exploring around rather than building in bases. This made it difficult for the players to find any new implementations independently. To gather some feedback, players were encouraged to try out the new systems in the specific testing scenario. This testing was conducted by giving players verbal tasks to complete. The completion of these rewards was perceived and additional questions were asked later. Questions were asked about how using the new system was and how it felt. The answer to these questions was lackluster, because players were a bit overwhelmed with the new information and did not understand the new systems. This possibly implies an usability issue, especially with the ease of use of these implementations.

Although some constructive feedback was gathered that have played similar games. In particular, one participant suggested improving the construction system by adding automation like in the game Factorio. This suggestion was mentioned, because the depth of the construction system seemed lacking. The automation suggestion was discussed further and another complementary idea was suggested. Similarly to the Steam system depth feedback, this idea requested an implementation of a drill machine. This suggested drill machine could mine a lot of resources and would serve as a starting point for resource automation. Importantly, this drill machine would provide the necessary resources to crafting machines. This feedback confirms that automation could be a valuable implementation in the next iterations.

Lastly, feedback was also received from the Steam discussion threads, where players pointed out a big problem with the deletion system. In a specific discussion thread[11] users highly criticized the deletion implementation. Multiple users agreed that the deletion system was terrible, because it was really easy to accidentally delete machines. This was caused by users not understanding how the deletion system was implemented, because there were minimal introductions to its usage. The problem solution was discussed with Steam users on the same discussion thread, where an acceptable solution was formed. The solution agreed on was a confirmation popup implementation, which would appear after the construction view is closed and before selected

[11] https://steamcommunity.com/app/1392650/discussions/0/3763355214785912662/

objects are deleted. Notably, the deletion system had another problem. The problem was an undesired feature, which deleted any stored resource alongside machine deletion. Importantly, these fixes were deployed in a hotfix within 2 days of the first thread post.

5.2 Game Developers Conference expo

The second iteration covers development between the Blastronaut "Companion" update and the Game Developers Conference expo. The latter was hosted in San Francisco Moscone center in the time span of 20th to 24th of March.

This iteration new implementations were made in the deletion system, modules, and automation solutions. Additionally, a drill machine solution was experimented and a special testing scenario was made for testing. This iteration directly takes inspiration from last iteration's feedback by implementing some requested features for the deletion system and the automation system.

5.2.1 Deletion system

As mentioned in last iteration's feedback, the deletion system needed precautionary measures to avoid accidental deletion. This problem was solved by a confirmation popup, which asks if the deletion of selected objects should be finalized (Figure 18). This confirmation popup was designed to show every time that the construction view was closed. With the new implementation deletion is only finalized when choosing the left "Yes" option in the confirmation popup (Figure 18). When choosing "No" or exiting the confirmation popup (Figure 18) then nothing is deconstructed.

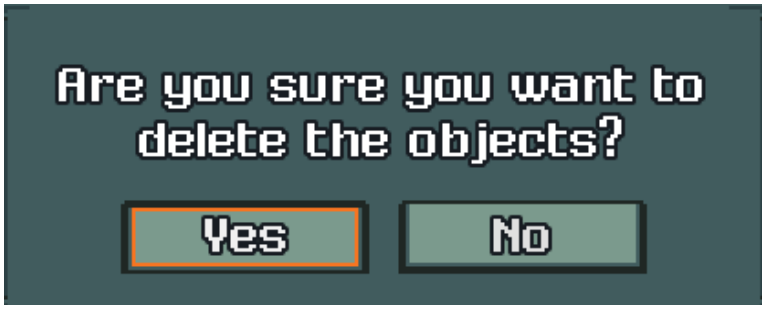

Figure 18. Deletion confirmation popup.

Additionally, another problem was addressed with the implementation of dropping machine stored resources, when the machine is deleted. This implementation also includes the integration

of separately stored automation resources, which are used in the automation system implementation.

5.2.2 New modules for machine upgrades

This iteration the previous modules except the fuel module were integrated for crafting machines. The speed and power modules were implemented similarly to the design of Factorio modules, which were introduced in Chapter 3.2.1. Firstly, the speed module is used to decrease craft time. Secondly, the power module is used to increase crafting productivity. Crafting productivity means that 1 additional free crafting output is added every time that a certain amount of crafts are completed. For this a variable was added to machines that keeps track of the number of crafts completed. For example, 1 power module generates 1 additional output every time that 18 crafts are completed. With every extra attached power module, the craft amount threshold is decreased by 3, which means that with 4 modules an additional output is generated every time that 9 crafts are completed (Figure 19). Notably, the fuel module's attachment to crafting machines was disabled, because the fuel module had not been implemented. The drone module is used for automation.

Figure 19. Furnace with 4 power modules, generating an extra resource on the 9th craft.

5.2.3 Automation with the drone module

The drone module was implemented for the container and crafting machines for the purpose of automation. The goal of the drone module is to allow players to transport resources between machines and create automated crafting networks. The drone was implemented for creating these crafting networks.

The drone logic starts with finding potential destinations for its attached machine stored resource. The drone searches in a 500 pixel radius for machines that require the given resource. If a machine requires the stored resource then the drone adds that machine to its list of destination machines. The destination machines list is shuffled and iterated through where each machine will receive 1 drone transported by the drone. When a destination machine receives a resource then the resource is stored in the machine logic and the craft trade is checked. If the craft machine finds that the machine has all the necessary resources, then the crafting trade is automatically executed.

In conclusion, automation can be set up in three steps. Firstly, a drone module should be added to a machine. Secondly, the machine should have specific resources stored. Lastly, the destination machines should have an active trade that requires the previously mentioned specific resource.

5.2.4 Drill machine

The drill machine idea was tested by experimenting with an example trade that uses fuel to generate raw resources (Figure 20, left). This needed extra logic for checking if the trade can be executed, which was done by checking the player fuel amount. This development was not integrated with this Blastronaut update, although this works as a base implementation for the next iteration.

Figure 20. Drill machine fuel trade.

5.2.5 Tutorial scenario design

The last iteration tutorial scenario did not work for showing off construction system changes. This iteration a new tutorial was made specifically for the Game Developers Conference expo. A big limitation of this scenario was the estimated user playtime, which was estimated to be about 3 minutes. The scenario was designed to quickly teach new players the basic game mechanics and dive into some construction system changes. Quick game introduction was done by adding specific quests designed to lead the player. Specific quests were made for showing the new modules. Namely, quests were made to upgrade equipment and create automated resource production with modules. Other features like the deletion system were expected to be encountered while completing the tasks. Importantly the tasks were accelerated with the implementation of defining pre-attached modules to machines. This was used to spawn in machines with the necessary modules already attached.

5.2.6 Tutorial scenario quests

The new tutorial started with the same old 6 quests, which introduced how to use weapons, jetpacks, trades and buying fuel. Mission 7 was the first quest to show-off construction system changes. The quest consisted of upgrading a blue weapon in a weapon station, which had pre-attached modules (Figure 21). This was intended to show the player the new upgrade system. The 8th quest shown in Figure 22 and the 10th quest were used to show automation with the drone modules. The drone modules were pre-attached to a container and a furnace. Importantly automation was incentivized, but was theoretically. The other quests were about gathering resources and exploration.

Figure 21. Upgrade weapon quest.

Figure 22. The 8th quest that shows drone modules.

5.2.6 Feedback

The Game Developers Conference event testing was conducted by the main developer. The testing involved playing the previously mentioned new tutorial scenario. The overall user feedback was lackluster, because most players did not make it to the construction system related quests. This was due to the low average 3 minute playtime. Although some players still reached the new implementations. These users did not provide extensive verbal feedback, because they did not understand the new construction systems. The main developer conducting the testing needed to explain how these systems work. The upgrade system in 7th quest was especially difficult to understand. Players did not notice how equipping the specific weapon in an upgraded weapon station affected the weapon. Similarly the automation quests were difficult to understand as well. Notably, during the automation quests manual crafting was also possible and because of this most players did not manage to use the automation at all. Although some users used automation, they were not entertained by it. Notably players liked the previous gathering implementation of drones, but the machine drones did not have the same effect. Importantly, these drones differ by the gathering drone visually showing the carried resource on top of its head and the automation drone does not. The previous feedback implies usability problems with the modules and the automation system, which should be addressed in the next iteration.

5.3 Blastronaut “Aqueous” update

The third iteration covers development between the Game Developers Conference expo and the Blastronaut’s “Aqueous” update, which was set to launch on 27th of April, but was delayed until 6th of May. This update featured the main developer integrating water simulation into Blastronaut. The water simulation with the construction system are 2 separate systems, but an idea was formulated to combine these with a water pump implementation. The construction system changes include all the development solutions mentioned in Chapter 4.2. In conclusion, this iteration covers the development of the deletion system, modules, automation, drill machine, water pump machine, and construction boundaries solutions. Last but not least a new testing scenario is designed to conduct final testing.

5.3.1 Deletion system

The deletion system had no feedback mentioned in the previous feedback, so personal testing was conducted. The testing provided two ideas for improving usability. Firstly, a new restructuring idea was formulated, which would place the deletion option under the construction options in the construction panel. Secondly, it was perceived that the deletion borders do not match with some machine visuals. These ideas are implemented with the purpose of improving deletion system usability and accessibility.

The build card deletion system implementation begins with removing unnecessary logic for the previous deletion input control. Next specific deletion options were added under each of the construction view tabs, which are used to delete the specific group of objects (Figure 23). The machine deletion logic was reused, without any notable changes. However, the logic for deleting blocks had to be refactored, because it had no filtering of blocks that could be deleted. This was a problem, because players could use this feature to substitute mining, which is not intended. Similarly, the delete module functionality had some big changes. The completely new logic functionality used the previously developed snapping, which was used as an helper feature for building modules. This helper feature was slightly changed for deletion by having the deletion option move to built modules only. Additionally, previously modules did not work with deletion

and did not refund any resources. Refunding resources was integrated by dropping the module's original cost.

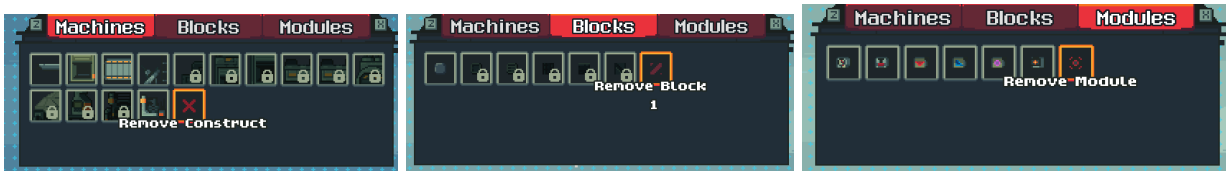

Figure 23. Deletion cards.

Secondl, the machine deletion borders were fixed by changing the colliders of most machines. The colliders of most machines had to be increased and the result is shown in Figure 24. Notably, platforms have small visuals, which makes it difficult to delete. To substitute for this the overall deletion activation radius was slightly increased.

Figure 24. Machine colliders after changes, shown with blue.

5.3.2 Fuel module

This iteration the last fuel module implementation for crafting machines was experimented. This implementation was only partly deployed to the update, namely for only the drill and water pump machines. This decision was made, because the fuel bar overloaded the machines with too much new information. The overloading of information is undesired and was further analyzed in Chapter 3.1.5.

The fuel implementation started with creating a fuel bar over the machine, which shows the machine fuel amount (Figure 25, left). Importantly, the fuel logic greatly changed the crafting

process. The crafting process required fuel to craft, consuming some fuel for each craft completion. Generally machines had a fuel limit of 200 and each craft consumed 10 fuel. Finding or creating new machines generally had the fuel amount full except for the drill and water pump machine. This was made to not alter the game balance too much, because having to insert fuel just after construction greatly increases the crafting cost. Restoring fuel was added under the crafting trade button, which activated only when the fuel bar was completely empty (Figure 25, right). The Fuel module implementation was simple, it reduced fuel consumption for each craft by 90%. For example, a crafting machine with 90 fuel would benefit from 1 fuel module by crafting 10 times instead of 9.

Figure 25. Fuel logic.

5.3.3 Drone module

This iteration the automation logic was refactored according to last iteration feedback to greatly improve the usability factor. Four new usability benefactors were added.

Firstly, the previous 1 drone module was replaced with 3 new modules, which filters the destination machines to a specific machine group. The separate machine groups are storing, crafting and selling machines. This change was accompanied with 3 new additions for both module and drone visuals. The module visuals abstractly shows the drone outline (Figure 26). Whereas the previous drone visual is still used specifically only for equipment storing machines. The 3 new drones specifically only are created on crafting and the container machine.

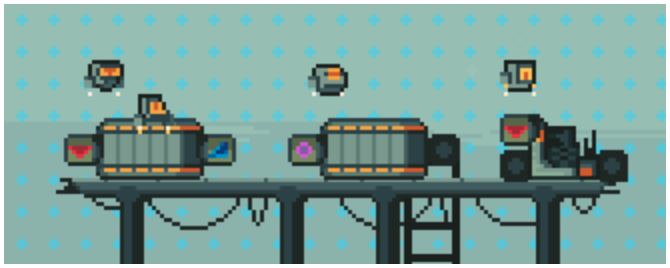

Figure 26. Drones left to right - store, craft, sell and equipment.

Secondly, according to feedback analysis drones needed to show carried resources. This implementation was completed similarly to the original drone carrying effect. The new carrying effect is shown in Figure 27.

Figure 27. A drone carrying a gmer resource.

Thirdly, automation usability was improved by visually showing the drone transported resources. This helps to manage automation by providing insight, which resource is needed. This number is shown in the machine trade and at the bottom of the specific resource with a new font (Figure 28). The new font is a smaller font that was added to differentiate this number and improve screen clarity.

Figure 28. 1 stored gmer resource in a crafting trade.

Last but not least, new logic was added to combine manual and automated crafting. This implementation includes using the available stored resources first before the player's resources. This is done by checking stored resource amounts before executing crafting trade. This logic is

shown in Figure 28, where 1 gmer resource is already stored and the trade is updated accordingly to cost 1 gmer resource instead of the original cost of 2.

5.3.4 Drill machine

This iteration finished the original drill machine idea. The main goal of the drill machine is to provide enough resources to benefit automation. The drill machine is implemented by producing raw metal resources and only requiring fuel to run. The machine output is determined based on where it spawns. The fuel can be added by executing the machine trade that was developed in the last iteration.

The drill machine spawn randomly in all biomes and is hidden behind basic blocks. When entering the close area a repair option will pop up (Figure 29, left). To repair the machine polished resources are required specifically from the next distinctive biome. Resource automation is intentionally first locked, when entering a new biome. The player first needs to manually mine resources. Only after advancing to the next biome can the player come back to invest in automation of previous resource production. Notably, when the machine is fixed a special explosion blows up the blocks that obstruct the mining process. Finally, the machine can start mining by providing fuel. On the left of Figure 29 a norgar drill is shown that requires the next polished resource, which in this case are dolicite plates.

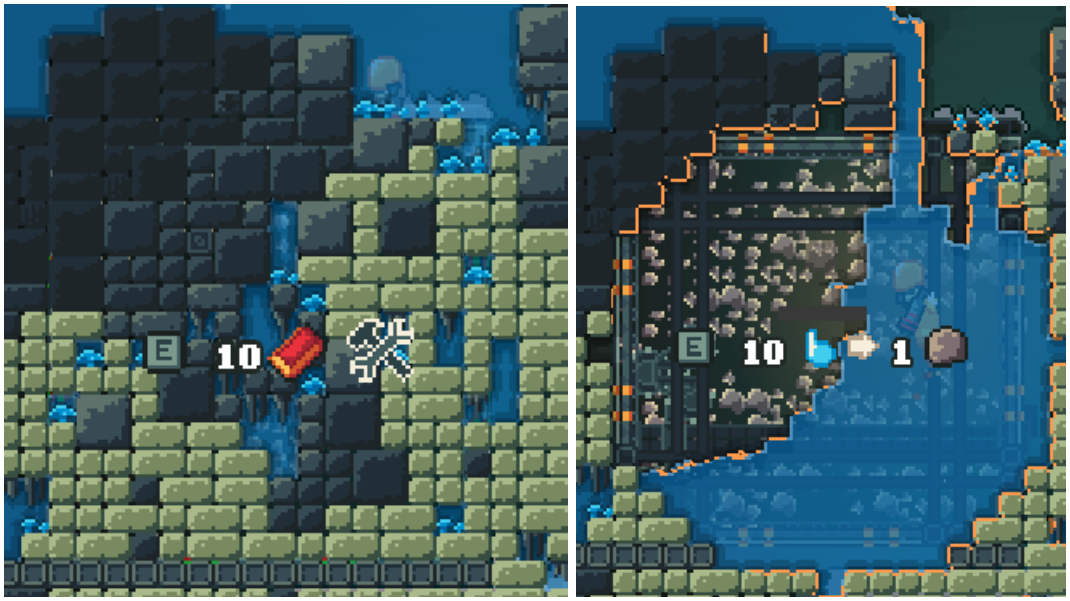

Figure 29. Hidden and broken drill machine on left and repaired drill machine on the right.

5.3.5 Water pump machine

The water pump machine idea came up in discussion with the main developer. The “Aqueous” update originally planned adding 2 new separate systems that were designed separately. Although, the water simulation and construction systems could be paired by a new feature. This feature would be a water pump machine, which is used to turn water into fuel. The machine would spawn randomly in the new water biomes. Notably, the water pump could be very impactful for complementing automation. The water pump can be connected to the drill machine by transporting generated fuel into the drill machine to increase resource automation.

The water pump logic started with creating a new trade that technically creates fuel from nothing. Next the implementation continued by checking tiles for water. To execute a trade and continue crafting water needed to be within the lower border of the water pump to start crafting, which is approximately shown with holes in Figure 30. When there was water, then the crafting was started and the water was removed from the 3 tile chunks located at the bottom of the water pump (Figure 30). Finally slots were added to the water pump to be able to connect it to the drill machine.

Figure 30. Water pump crafting process.

5.3.6 Construction boundaries

The construction boundaries problem was handled in two separate implementation approaches that were set as goals in the 4.2 Chapter. The first approach changes the existing Main machine construction restrictions and the second allows the player to access construction while moving.

The first implementation idea was to extend the main machine borders by moving around in the construction view. This was implemented by moving the camera towards the closest screen border, whenever the construction cursor was close to a screen border. The new maximum construction borders are shown with red color in Figure 31, where another implementation is shown with the cursor being clamped inside the construction borders. The new construction borders are static at 1000x1000 pixels for all screen resolutions, whereas the previous borders were relative to the screen resolution, which was at default at the maximum 640x360 pixel resolution. In conclusion, the new construction area is about 4 times larger than the previous area, which means that there are less restrictions on the expression of player's creative freedom.

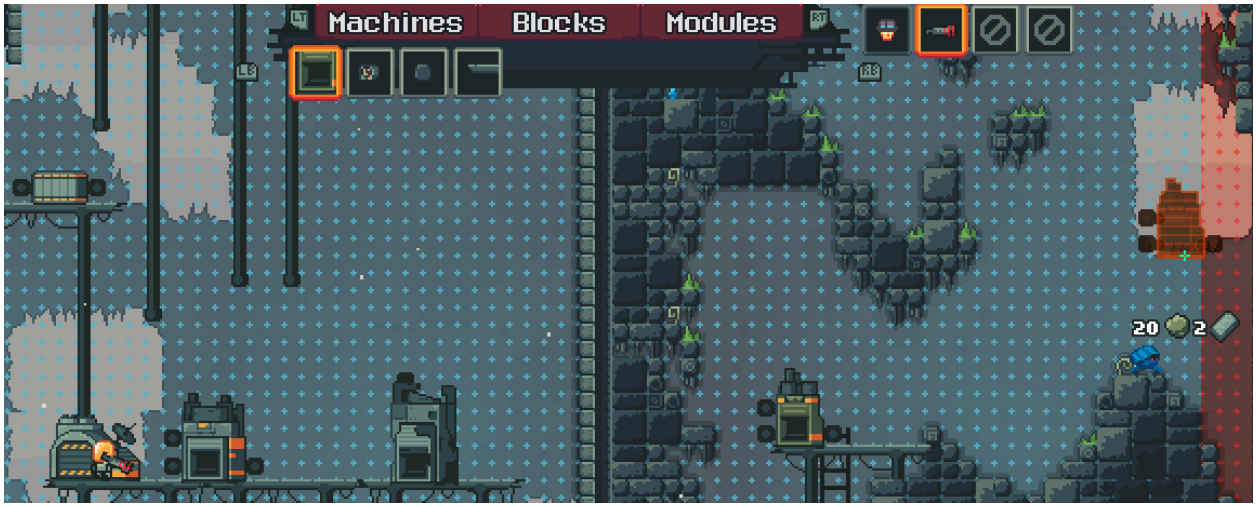

Figure 31. Right edge of main construction.

The second implementation covers a new interface for construction that allows the players to move and construct at the same time (Figure 32). The interface is accessed by equipping a special construction weapon. The construction weapon has 3 different variations, 1 for each of the construction categories: machines, blocks and modules. The construction weapon interface shows the construction options relative to the previously mentioned groups. Notably the machine

construction variant has access to only a subset of the Main machine construction options. The available options are platform, primitive furnace, container, Main, antenna, weapon, storage machines and additionally a deletion option (Figure 33).

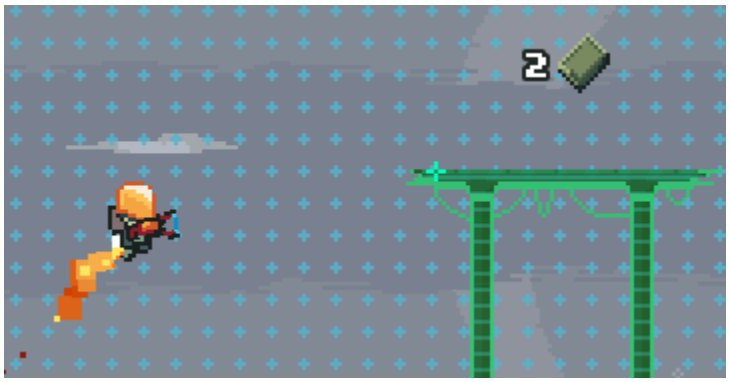

Figure 32. Moving while constructing.

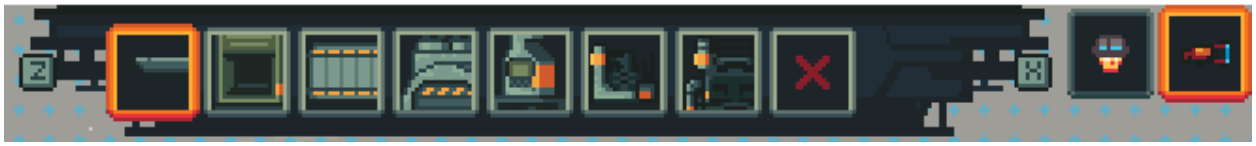

Figure 33. Mobile construction options.

5.3.7 Construction boundary changes in input controls

Big part of implementing construction boundaries was finding optimal input controls. The controls were designed to be as similar to default controls as possible for consistency, which benefits general ease of use. Importantly, the mobile construction requires more inputs than the previous construction system. Additionally, portability of Blastronaut other devices like the Steam deck and computers needs to be considered. These devices feature input controls of two groups, namely the keyboard and mouse and the controller groups. Notably, the controller has a particularly limited amount of input controls. The following input control changes are divided into two change approaches, namely general input and mobile construction specific changes.

The general changes are designed to reuse inputs in similar situations. The first change is changing construction cursor movement from controller left stick to right stick, which is caused by player movement being already assigned to left stick. Secondly, the Space button was removed for placing objects, because this action control overlaps with the jumping action.

Finally, new options were added for changing weapons on controllers, with the L3 and R3 buttons for switching weapons left and right accordingly.

Table 2. Changes in mobile construction controls

	Keyboard initial buttons	**Keyboard button changes**	**Controller initial buttons**	**Controller button changes**
Enter construction	E, Enter	Added weapon changing	Xbox X	Added weapon changing
Exit construction	E, right mouse button	Removed the right mouse button. Added weapon changing	Xbox X or B	Added exiting with weapon change
Construction movement	W,A,S,D, arrow keys, mouse cursor position	Removed WASD and arrow keys	Xbox left stick and arrow keys	Removed controller arrow keys
Next	Mouse wheel up	Replaced with X	Xbox RB	-
Previous	Mouse wheel down	Replaced with Z	Xbox LB	-
Jump	W, up arrow and space	-	Xbox A and LT	Removed A

The mobile construction specific changes (Table 2) cover control changes that only affect usage with mobile construction weapons. Firstly, accessing mobile construction was altered with both entering and exiting interactions having new added controls with weapon change input controls. The weapon change input controls are described in Chapter 5.1.2, which has a new addition of the L3 and R3 buttons for the controller. Additionally, the right mouse button is removed for exiting, because it overlaps with the input control for using the jetpack. Secondly, the Construction movement changes are caused by overlapping with movement controls. Namely, the buttons WASD, keyboard arrow keys, and controller arrow keys are removed. Thirdly, the next and previous option control inputs for the keyboard and mouse players are changed to X and Z accordingly. weapons. The previous inputs overlapped with weapon changing controls and additionally the X and Z button usage is consistent with the overall input scheme. The mentioned buttons are also used for selecting the next and previous items in the Main construction system.

Lastly, controller jumping input A was removed, because it overlapped with the construction build action input.

5.3.8 Test scenario prerequisites

Similar to last iterations the implementations are evaluated in a specific test scenario to gather relevant feedback. This test scenario can be played by downloading the build.zip in Appendix I. Contradictory to the last iterations, this test scenario only focuses on the construction system to guarantee feedback. Another contradictory change is the lack of a physical event to conduct testing. This means that testing is conducted remotely with testers playing the test scenario and providing feedback independently. This affects the testing process, because it makes it more difficult to guarantee constructive feedback. To compensate for this the focus of this testing scenario is to be as short as possible and provide enough introductions for players to be able to experiment with the new developments. The testers are guided by the use of Blastronaut quest system, which originally only features gathering quests, but this is improved with new additions. These improved quests are used to introduce the deletion system, modules in general, drone module for automation, construction weapons, drill machine, and water pump machine. However, this test scenario needed two large developments for guiding the player. The 2 implementations include a system for showing information for machine options and additional conditions to the quest system.

Firstly, the information system is designed to show more info for upgrading with modules. This includes adding descriptions to equipment, crafting machines, and modules in the construction tab. The equipment description shows the equipment stats, which are shown with a number of the specific equipment attribute and an associated icon graphic (Figure 34). The crafting machine has a similar approach, but all the crafting machines use the same type of stats. The stats are related to the selected craft trade and only shows crafting time without any modules. If a power module is added a recycling stat is shown that has a number to show the extra free crafting output cycle length. If a drone module is added then the associated drone is shown under the description. The fuel module is currently not shown, because fuel logic is disabled for most machines. Finally descriptions were added to the modules in the construction tab. This features a text of the module impact and icons, which show what stats are possibly affected (Figure 35).

Figure 34. Shown stats of a weapon.

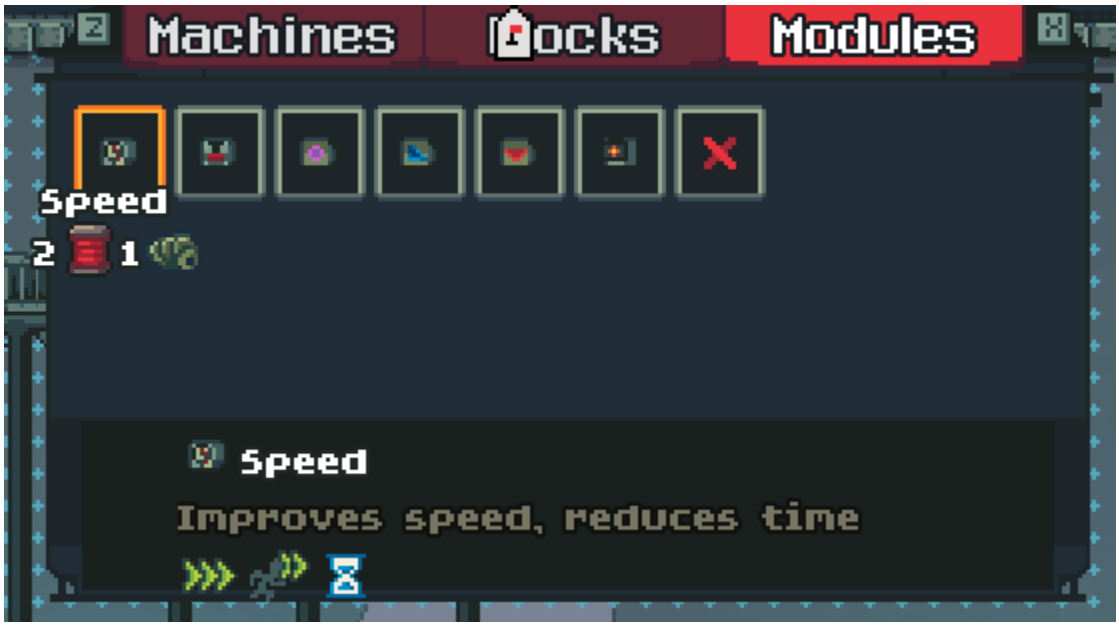

Figure 35. Speed module description.

Secondly, the quest system additions include changing the task logic to create new types of quests. Firstly, an optional radius variable was added to tasks, which is used with the quest location to determine if the quest was completed in the correct area. This implementation is used for quests to detect construction and deletion of specific machines, blocks and modules. Secondly, an optional trade name variable was added to tasks, which is used to detect more advanced scenarios. This variable was used to detect entering construction, exiting construction, equipping a specific weapon, detecting relative resource amount change. The last use case was particularly useful for a quest, which is used to check if the player stored the correct amount of resources in a container. Thirdly, an optional stat name variable was added to tasks that is used for specifically checking if the equipped gear has a high enough number in a specific stat. Lastly, the quest marker ordering was adjusted, so quest markers would show up on top of the

construction view panel. This is an usability improvement for the construction and deletion quests.

5.3.9 Test scenario layout

The testing layout is structured vertically, starting from the top base and ending straight downwards from the starting position. This means that the first quests are on top and the quests progressively take the player underground. The quests are intentionally spaced together to engage the player focus and avoid wandering away from testing. The player is incentivized to not die during testing with the addition of more player health and fuel. Notably, the testing speed is accelerated by unlocking construction options. Furthermore, the testing speed is accelerated by giving the player optimal starting resources. In conclusion, these changes are supposed to incentivize experimentation with the new systems and not lose focus on other aspects.

Figure 36. First handmade template.

The testing scenario features a total of 4 customly designed templates, which are used for creating an optimal testing environment. The first template is associated with the first 3 quests and starts off with the player flying into the base on a space rocket (Figure 36). The base features 2 construction weapons and a give feedback button, which is used to signal to the players that feedback can be given at any point. The second template is used for the 4th quest, which has a lone furnace on a platform (Figure 37). The third template is first used in the 4th quest and is used until the 7th quest. The template features a base, where there is a Main machine and

multiple platforms, which are used to separate container, furnace, assembler and sell machines (Figure 38). Lastly, the 4th template is used for the 8th quest. The template features a base with some platforms, a Main machine, 2 construction weapons and most importantly has 2 hidden machines, which are the water pump and drill machine (Figure 39).

Figure 37. Second handmade template.

Figure 38. Third handmade template.

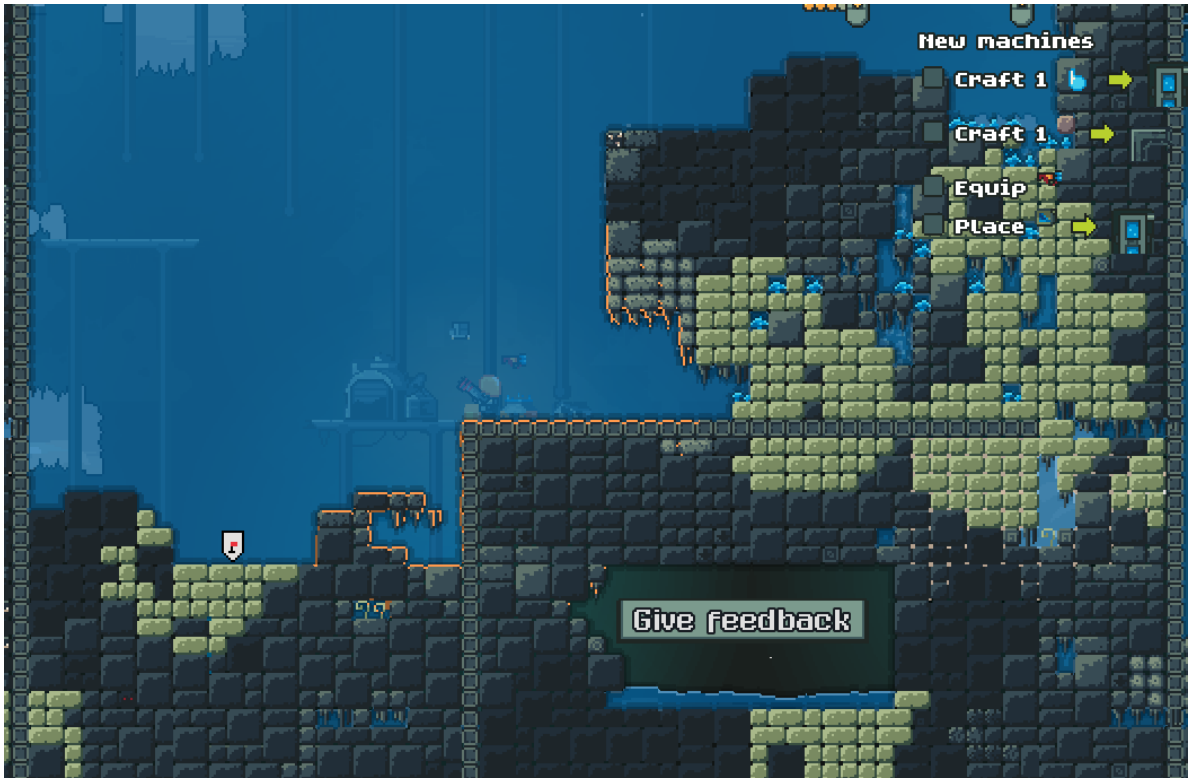

Figure 39. Fourth handmade template.

5.3.10 Test scenario quests

The test scenario includes a total of 8 interactive quests and ends with asking for feedback. The test scenario starts with showing the machine building variation of the construction weapon, which needs to be used to build 1 tool and 1 jetpack station. The second quest features another construction weapon, which needs to be used to build and delete a speed module. The third quest requests attaching specific modules to the jetpack station that would give it the desired stats. This quest is solved by attaching 3 speed modules and 1 variant of any drone module to the jetpack station and equipping the new jetpack after. The fourth quest requires the player to go into a Main machine and use the Main border movement to delete a furnace. The fifth quest is one of two quests that are used to set up automation. The quest requires the player to build a container, a furnace, and an assembler to the correct platforms. The sixth quest requires the player to set up automation by changing machine trades. This is solved by storing 50 gmer resources in a container, crafting 1 gmer plate, and crafting 1 gmer gear. The seventh quest finalizes automation by adding correct modules to the machines used in the previous quest. This includes attaching a

craft drone module to the container and the furnace and a sell drone module to the assembler. Finally, when everything is placed correctly then in a short time period money should be generated to finish the quest. The last eight quest includes finding the water pump and drill machines, crafting with them once and finally attaching a craft drone module to the water pump for resource automation. This concludes the test and the player is instructed to click on the give feedback button, which opens up a link to the associated Google Forms.

5.3.11 Feedback form

The testing scenario ends with a feedback questionnaire that is used to evaluate the usability and impact of the final implementations. Importantly, this feedback form (Appendix I, Form.pdf) was sent out on the 6th of May, which was the delayed update release date. This and the fact that the test was conducted remotely means that it was difficult to gather enough feedback to make conclusions. However, this problem was addressed by focusing on decreasing feedback completion speed, which was also present in the testing scenario design. This idea is used to get more feedback by decreasing tester effort, making this test accessible for more testers. Notably the short UMUX usability questionnaire [9] was chosen to reduce tester effort. The UMUX questionnaire features asking system feedback with 4 questions that all relate to different usability components. These questions were simplified for Blastronaut testers. The modified questions, where [system] is replaced by the specific implementation name, are in the following chronological order:

1. [system] capabilities meet my needs,
2. using [system] is a frustrating experience,
3. [system] is easy to use,
4. it took me a lot of time to understand [system].

The previous questions are answered with the common 7 option UMUX scale, where 1 stands for strongly disagreeing, and 7 stands for strongly agreeing [9]. The form was designed uniformly, with the previous 4 questions structured in a table and an additional optional field to explain the previous choices. Questions were asked relative to the general quest chronological order about the following list of implemented systems:

1. construction weapons,
2. general modules,

3. deletion system,
4. main machine border movement,
5. automation with the drone module,
6. water pump machine,
7. drill machine,
8. general construction system.

6 Usability results

This section first analyzes the testing results (Appendix I, Results.pdf). Secondly, the implementation results are summarized and finally future improvements are described.

6.1 Usability results

This chapter analyzes the overall test feedback with 2 general graphs that are calculated based on the UMUX score system [9]. The score was calculated by subtracting 1 from odd questions and subtracting even questions from 7. This way all scores were positively inclined with usability factors effectiveness, satisfaction, overall. and efficiency accordingly. Next scores were averaged between all 6 respondents. Finally each particular score was divided by the maximum possible score 6 and multiplied by 100 for percentage scale. The calculated score data is compiled into a single table (Appendix I, Scores.pdf). For figure 40 the factors were summed together, getting the commonly used UMUX score [9] for each implemented system.

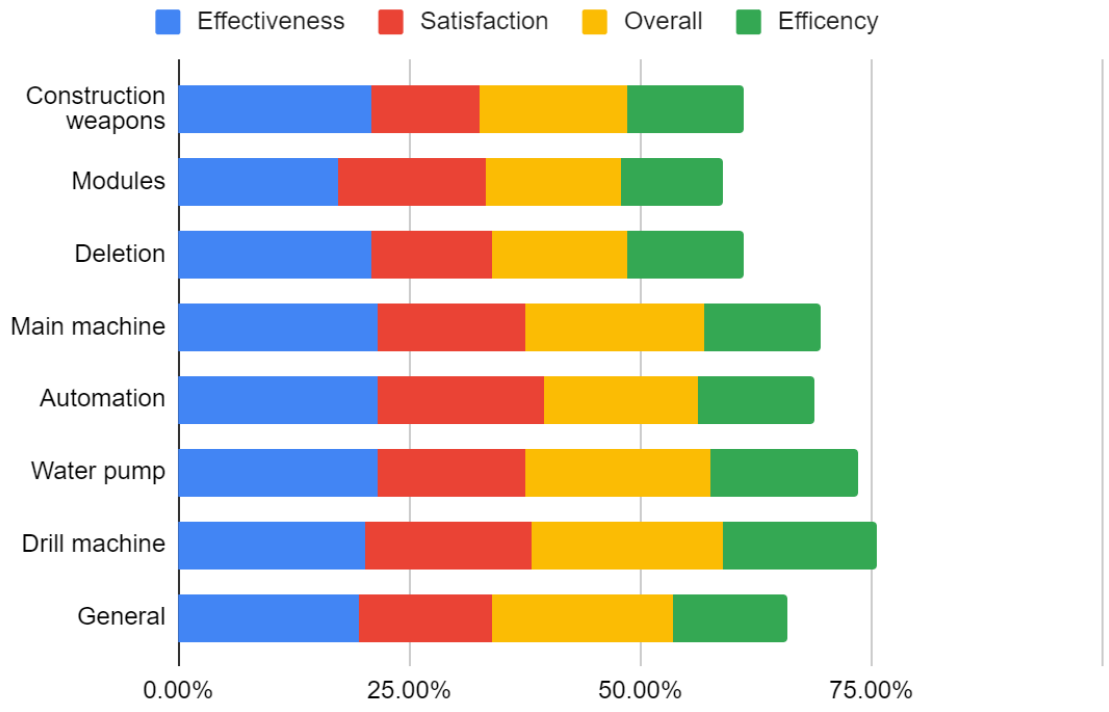

Figure 40. UMUX scores, which are grouped by system and colored by usability components.

6.1.1 Overall usability

The overall feedback results are shown in Figure 40. The x axis shows the UMUX score total out of a possible 100% rating. The y axis shows the implemented system categories. The 7 system categories have a maximum variance of about 17%, with the new machines leading the Figure 40 in terms of UMUX score. Specifically the drill and the water pump have 76% and 74% ratings accordingly. These are high scores that praise the implementation of these systems. These ratings are followed by the Main machine and automation implementations, which both have around 69% in terms of UMUX score. The 69% score is also relatively high in terms of UMUX score. The last 3 specific systems are rated around 60% UMUX score. This signals that there could be a big problem in some usability factor, which should be analyzed further. Lastly the overall construction system was rated at 66%, which is really close to the actual mean of all the results.

6.1.2 Specific usability components

Figure 41 clearly highlights all the usability factor UMUX scores separately. All the implemented systems are scored out of a possible maximum 100% rating.

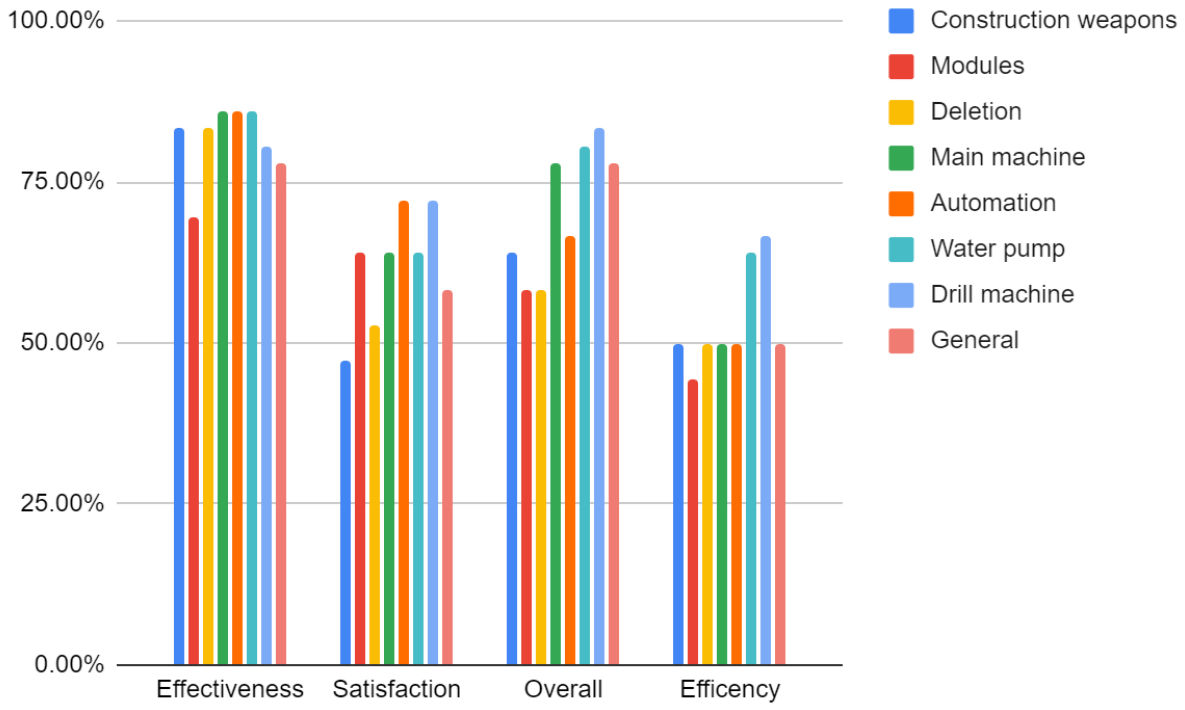

Figure 41. UMUX scores, which are grouped by usability component and separated by systems.

Starting with effectiveness, this factor is especially high with over a 80% score for all systems, except the modules. The highest result is for the automation and water pump systems, which have an effectiveness score of 86%, which is a very good result. However, the modules score slightly decreases the overall systems effectiveness score to 78%. The modules system is in total about 14% lower than all other systems with a score of 69%. This score is the probable main cause of the modules system overall relatively low rating. Notably, the modules section was the shortest used system using modules only for upgrading a single jetpack. The modules could be also used to upgrade weapons and machines, which was not required in the tutorial scenario. The module effectiveness may not have been perceived by the players, because of the tutorial design.

Continuing with satisfaction, this factor has arguably the most varying results, with construction weapons lowest at 47% score and automation and drill machine sharing the top spot with around 72% score. Notably the deletion system is also low at 53%. Both of the construction weapons and deletion systems seem to be heavily affected by the satisfaction factor, which means that players did not enjoy using those systems. This is also highlighted in the Figure 40 overall scores, which features low results for both systems.

The next overall category features results generally divided into two groups. The higher group includes drill machine, water pump, Main machine and General changes with around 79% score. The lower group includes automation, construction weapons, modules, and deletion with around 60% score. Three out of the four systems in the lower group have a low general score, the only exception is the automation system, which in this case has an uncharacteristically low score.

Finally the efficiency can be divided into 2 groups, with the higher group having a water pump and drill machine around 65% score. The other scores are uniformly around 50%, which is generally a pretty low factor score. This means that generally testers needed a lot of time to understand these systems, which is acceptable for these complex systems.

6.2 Development results

The testing results reflect that overall players were happy with the new implementations, as almost all were higher than the 50% margin. Notably most of these ratings were in terms of

efficiency, which reflects learnability. Although these systems are very complex and a high learnability curve is expected. Some other low results came from construction weapons and deletion systems satisfaction score, which represents that these systems were unpleasant to use. Notably both apply new control schemes, which are difficult to design. All the other feedback was around the 70% margin, which marks the usability requirements as fulfilled. Moreover, the problem specific requirements can also be considered achieved as this solution was officially released, tested, and had an especially high effectiveness rating.

6.3 Future improvements

The improvements are divided into 3 categories, which are improving implemented systems, user testing, and other perceived construction problems. Firstly, this thesis focused on solving significant parts of the problems, but it should be acknowledged that these solutions can have some usability improvements. Secondly, the implementation iterations provided valuable learning moments. Lastly, additional developments were requested in Chapter 2.1 that were left for future studies.

6.3.1 Improving usability of implemented systems

The testing results proved useful insight on the new implemented systems, although some systems performed moderately worse than others. Namely, the construction weapons, modules and deletion systems were relatively lowly rated at 60%. All of these systems were weak in the overall usability and efficiency factors. Additionally, construction weapons and deletion systems were low in user satisfaction and modules score was low in terms of effectiveness. Efficiency is at an acceptable level, as these are complex systems that need time to get used to. Satisfaction can be improved by more polishing in terms of construction weapon usage and deletion design. The prop system can be improved with more animations that make for a better immersive experience. The latter deletion system could use a separate button, which quickly selects the delete option and optionally a mass delete solution. Arguably the module's low effectiveness should be retested with new user testing.

6.3.2 User testing

Testing was completed in all 3 implementation iterations, with the testing process evolving during the process. The first two testing processes had little feedback, serving as good learning opportunities for the final testing. Although the final user testing was conducted late, it still provided useful feedback. It would be useful to conduct testing in a longer time window with more participants to have a better overview of perceived usability. Notably, the biggest failure came from short testing of the modules system. The system possibilities were not fully shown and because of that the modules implementation results can not be fully perceived. This is a learning opportunity for future works to not cut out too much content in exchange for testing completion speed.

6.3.3 Other construction system problems

The Blastronaut construction system can benefit from multiple developments, which were discovered in Chapter 2.1 users feedback and additionally during thesis implementation. Most feedback problems were addressed, but still some problems were not. Namely the fast travel, rare machine and container problems should be solved in future development. Particularly the fast travel implementation was highly requested and notably other systems could benefit from this implementation as well. The feedback mentioned multiple solutions, but this problem should be deeply analyzed in similar games and the whole logistic system should be thoroughly planned before developing particular travel options. The rare machine and container problems are smaller systems that could be solved relatively quickly and the implementation could have significant game experience benefits for some players. On the other hand implementation provided more insight into the solutions that would benefit the development process. Some small adjustments could be made, but the most benefactory solution would be to refactor the quest system, which would help to get more responses from the Blastronaut users. These are the new implementations that could benefit Blastronaut’s construction system.

7 Conclusion

The main goal of this thesis was fulfilled by improving the construction system in most areas. The thesis starts with setting implementation requirements according to Blastronaut construction system feedback and similar construction game design. These requirements set a basis for implementation, which was done in multiple development iterations that ended with a major event, where user testing was conducted. Developed systems include the deletion, automation, Main machine construction, mobile construction, and upgrade systems. These implemented systems have been deployed in the Blastronaut game since 6th of May. Notably implemented systems were evaluated by final usability testing, which was conducted remotely independently with the use of the academically popular UMUX questionnaire [9], which provided valuable insight for evaluation of the implementation results. Overall, the results reflect that the implemented systems have a high usability correspondence, with exceptional results in system effectiveness. Notably the effectiveness component had over 80% scores for almost all systems. Additionally, this work provides learning opportunities, which can be used in future studies to create better user testing scenarios, analyze the iterative implementation approach, and most importantly develop holistic construction systems.

8 References

[1] Ernest Adams. Fundamentals of Construction and Simulation Game Design. New Riders. 2013.

[2] Silver Spitsõn. Mallide lisamine Blastronaut mängu protseduurilisele generaatorile. University of Tartu Institute of Computer Science Bachelor's thesis. 2021.

[3] Christian Moro, Charlotte Phelps, and Jamers R. Birt. Improving serious games by crowdsourcing feedback from the STEAM online gaming community. The Internet and Higher Education. vol. 55 article 100874. 2022.

[4] Dayi Lin, Cor-Paul Bezemer, Ying Zou, and Ahmed E. Hassan. An empirical study of game reviews on the Steam platform. Empirical Software Engineering. vol 24. pp. 170-207. 2019.

[5] Maithili Dhule. Exploring Game Mechanics: Principles and Techniques to Make Fun, Engaging Games. CA: Apress Berkeley. 2022.

[6] Scott Rogers. Level Up! The Guide to Great Video Game Design. Wiley. vol. 2. 2014.

[7] Factorio. Official Factorio Wiki. [Online]. https://wiki.factorio.com (09.05.2022)

[8] Kenneth N. Reid, Iliya Miralavy, Stephen Kelly, Wolfgang Banzhar, and Cedric Gondro. The Factory Must Grow: Automation in Factorio. GECCO '21: Proceedings of the Genetic and Evolutionary Computation Conference Companion. pp. 243–244. July 2021.

[9] Kraig Finstad. The Usability Metric for User Experience. Interacting with Computers. vol. 22 issue 5. pp. 323-327. 2010.

Appendix I - Final testing Materials

Materials created for the testing and the results can be found in the attached blastronaut_testing.zip file.

Contents of the file:

- steam_api64.dll - Configuration file, required for running testing executable.
- Blastronaut-construction.pck - Configuration file, required for running testing executable.
- Blastronaut-construction.exe - Executable, which can be run to play the testing scenario. Requires the steam_api64.dll and Blastronaut-construction.pck files to run.
- Form.pdf - Testing form, which was used to gather feedback after testing scenario
- Results.pdf - Table with all the initial testing responses.
- Scores.pdf - Table with usability scores, which were calculated from the initial testing responses.

Appendix II - Gitlab repository

The Gitlab repository containing the can be found on the following link:

https://gitlab.com/perfoon/blastronaut.

To request access, please e-mail markus7sulg@gmail.com.

Appendix III - License

Non-exclusive license to reproduce thesis and make thesis public

I, **Markus Sulg,**

1. herewith grant the University of Tartu a free permit (non-exclusive license) to:

    1.1. reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

    1.2. make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

of my thesis

Designing a Construction System for Blastronaut Players,

supervised by Jaanus Jaggo,

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive license does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **09.05.2023**