Game Programming Laboratory

MBB - Marsianische Bundesbahnen

„Trains, but on Mars“

Samuele Decarli – Producer + (Programmer, Designer)
Jesús Martín Berlanga – (Artist, Musician, Designer, Programmer)
Moritz Wettstein – (Programmer, Designer, QA)
Jonathan Granskog – (Programmer, Designer, Visual Artist)
Sandro Marcon – (Programmer, designer, QA)

Chapter 1. Formal Project Proposal

1 Game Description

1.1 Overview

A 3D, single-player, strategy and management game about building a railway infrastructure on Mars. The map is a grid, with tiles of different types and heights, dotted by colonies and resources. The player has to connect cities and resources together with railways, allowing the cities to grow and develop. As the cities develop their needs change, and they can make use of different resources.

The player earns money when the railroads are used, and pays upkeep for their maintenance. Different terrain tiles (resulting in tunnels, bridges, etc.) have different upkeep costs. Therefore, deciding whether a certain connection is worthwhile is a key aspect of the game.

As time passes Mars is terraformed, leading to a changing landscape, with parts of the map becoming flooded or being hit by asteroids, changing the upkeep cost of already existing connections.

The player loses when he has debt for a prolonged time and go bankrupt or when the research level of civilization is not high enough before Martian aliens defrost. An alternative sandbox mode allows the player to set their own objectives without having to worry about aliens.

1.2 Background Story

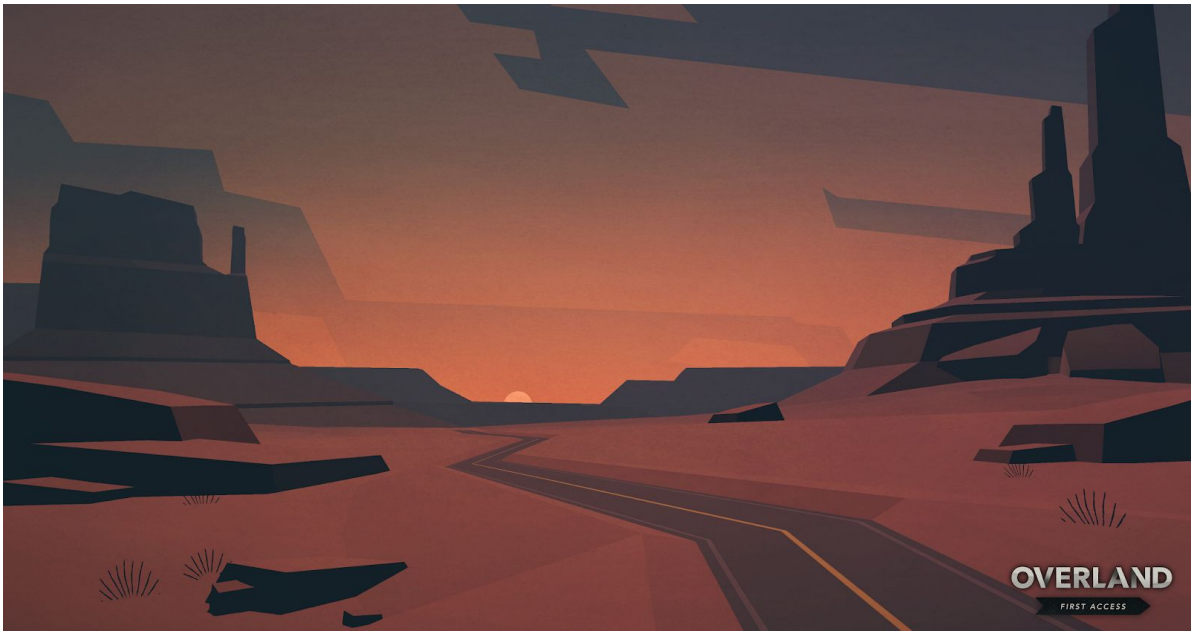
After years of effort, humanity has colonized mars. The first small cities have been founded, protected from the harsh Martian atmosphere. But as more and more colonists arrive, those cities hunger for resources. A team of researchers at ETH gets a brilliant idea: bring back Escher to life, so that he can do what he did to Switzerland many years ago, build up a strong infrastructure. They succeed, creating Cyborg Escher, who is promptly put in charge of the MBB and of the Martian Credit Group, a newly created railroad company on Mars, and a banking group with local banks at cities.

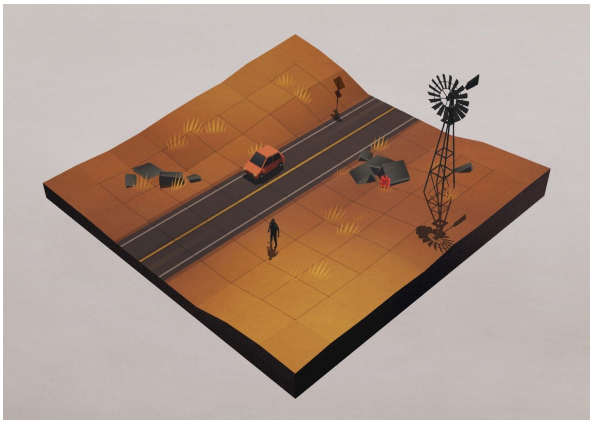
As Mars heats up thanks to terraforming, its ice caps begin to melt and strange rumours begin to circulate among the colonists. Rumours of alien creatures frozen deep beneath the ice, that might eventually be freed. A race against time begins as Cyborg Escher has to make sure that the nascent martian colony has the resources to fight back against the alien menace before they are freed.

1.3 Design Decisions

The visual style of the game will be a minimalistic low-polygon look and our color scheme will mainly be based on orange and red colors to fit our Mars theme. Contrasting colors, like blue, will be used to make it easier to separate cities from the surrounding environment. The shading will be flat and non-photorealistic. We have drawn inspiration for the game's visual style from the indie game Overland and the low-poly art of Timothy Reynolds (below). The 3D assets that we will need, in addition to the procedurally generated map and railways, are city and building models, a few train models and models for the mines. We will also need 2D assets such as icons for the menus and

textures for our models. The user interface will show important user information such as current time, city resources, money. In addition, there will also be a basic menu, when starting the game, to select game mode and set other options.





1.4 Gameplay

To win the game the player has to help evolve civilization to reach a research level that allow to fight the aliens before they defrost. To achieve this goal the player will also have to manage wisely its capital, trying to maximize profits (when railways are used) that in turn will also help to create a better infrastructure between cities and resources. Railways constructions differ in prices, one going through a mountain (a tunnel is needed) might cost more than one that goes around. However, transportation would be faster.

The so known leverage effect (investment strategy of using borrowed money: specifically, the use of various financial instruments or borrowed capital to increase the potential return of an investment) can be used with the money population deposit at the banks to play more aggressively, for example building a costly railroad across a mountain to rapidly exploit a mine before this is covered by water, at the cost of extra risk: If the player ends up bankrupt by getting into too much debt he also loses. Given overall lack of resources, or non payment of debt, cities will increase the probability of having lengthy riots which make population angry, taking control over the main station at the city and refusing to pay for tickets of trains leaving the city.

During the game the water level increases, as a consequence of the heat produced by the industries and the cities. This results in regions being flooded with possibly severe consequences for the train infrastructure. The player has then either to upgrade it or simply build some new rails somewhere else. Some random events such as asteroids can also destroy the infrastructure.

City Resources:

- Lack of **H_2O**, which is used to generate oxygen and necessary for the survival of humans, will increase the rate of death among the population. At the beginning of the game it is the most precious resource that can be extracted from ice glaciers before they start defrosting.

- The Lack of **Iron** will not cause a city's population to die but will prevent its growth limiting its maximum capacity since it is a material heavily used in city architecture and it is needed for its expansion. Silicon is present in low quantities 5%, on iron ores.

- Lack of **Gold** in a city creates a sense of distrust among the population since they feel that the bank currency is not backed up by something real, limiting the amount of capital people deposit at the bank and that is available for investments. Interest rates are also increased to account for the risk the people feel they are taking.

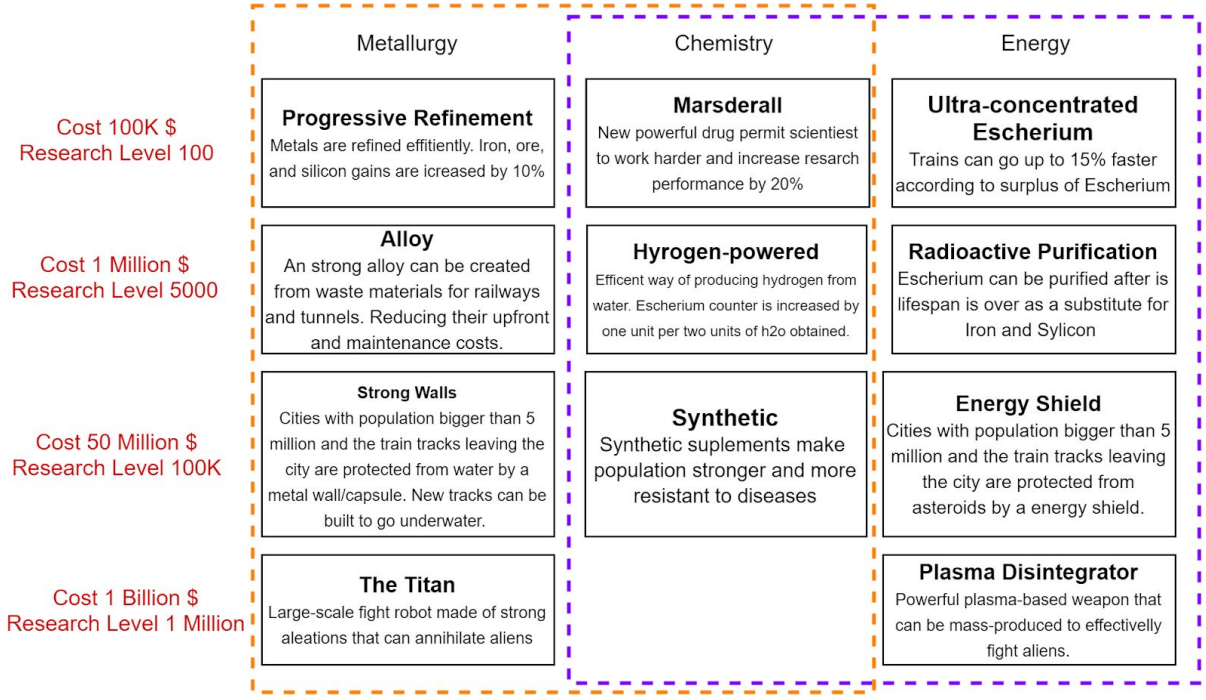
- Lack of **Silicon** in a city has negative effects on the research performance. On Mars it is found in Citrine silica quartz, which has a transparent orangish color due to small amounts of suspended hydrous iron oxide.

- **Escherium** is a new radioactive material discovered on Mars and it is heavily used for energy generation. Its lack can affect the city's station performance, decreasing the frequency of trains travels.

Over time civilization will want to expand to other cities on other regions indicated with a marker. We can choose to make a connection to this area and a small city will be created beginning with enough resources (+100 of every resource for example).

Research

To unlock new features in the research upgrades tree we need both capital and a certain research level. Research level is increased per unit of time according to each city population size and research tier (which can be upgraded with money after that a certain population size has been reached).
Only one of the three upgrades can be chosen from those at the same row and it is impossible to have upgrades in both metallurgy and energy: the player has to decide if he is more energy or metallurgy based depending of availability and positioning of resources in the generated procedural random map. “Strong Walls” upgrade, for example, would be useful to exploit large metal deposits that are endangered by the possibility of being overflowed by water late in the game. On the other hand, areas in the map with a lot of craters might indicate that area is usually impacted by asteroids and the energy upgrades is theoretically superior.

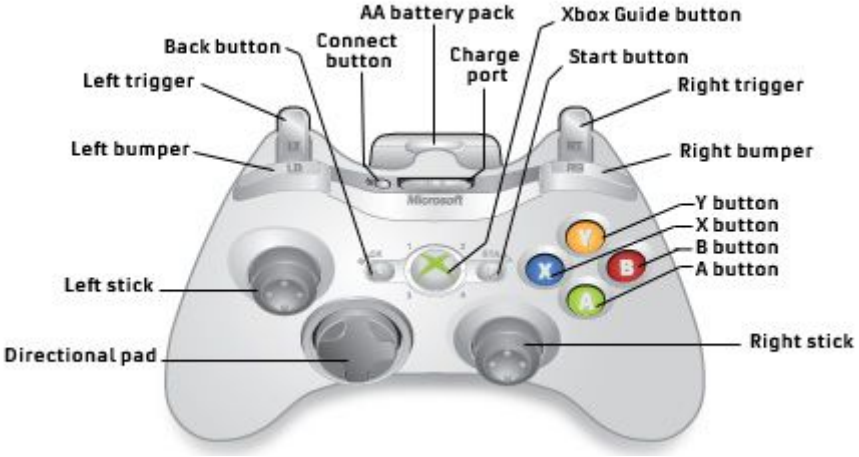


Alfred Escher Cyborg

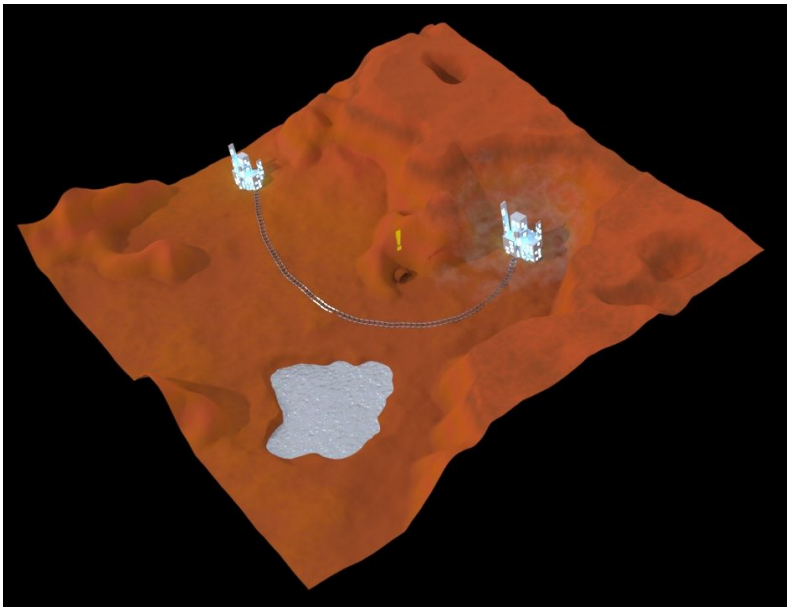
Alfred Escher can move from city to city using the railroad, which takes some time depending of the length of the travel. To upgrade the research tier of a city, Alfred must be at its inauguration. If the railroad network is disconnected, he will not be able to upgrade the research tiers of some cities until the railroad is repaired. Additionally, the presence of Alfred in a city boosts the research performance by 50%. In each city there is a local bank from the Credit Mars Group where Alfred can make use of money that citizens have deposited as long has he returns the money with some interest to the citizens so that they are happy.

Controls and possible actions

The player moves his camera with one analog stick and moves his cursor with the other analog stick. He can move the cursor somewhere and press A to start a railway, then move the cursor somewhere else and confirm with a second button press to build a railway between those points, with X (or B) to cancel. While he is doing this, the railway to be built will be shown so he knows where it will go and its cost displayed. He can change between different kinds of railways (allowing tunnels, and/or bridges) to be built by using RB and LB. He can rotate the camera by 90 degrees with LT/RT. Bringing his cursor over a city or mine and pressing a button will show some information about the city. Pressing Start opens the menu to save, load, etc.



As their needs are satisfied the colonies develop: their population grows, they create new industries, requiring different and more resources, which leads to them paying more for the trains. This is visualized by having a different (larger) model on the map. The player can also inspect the cities by moving the cursor over them and pressing a button. This will bring up a display showing more detailed information about the city.



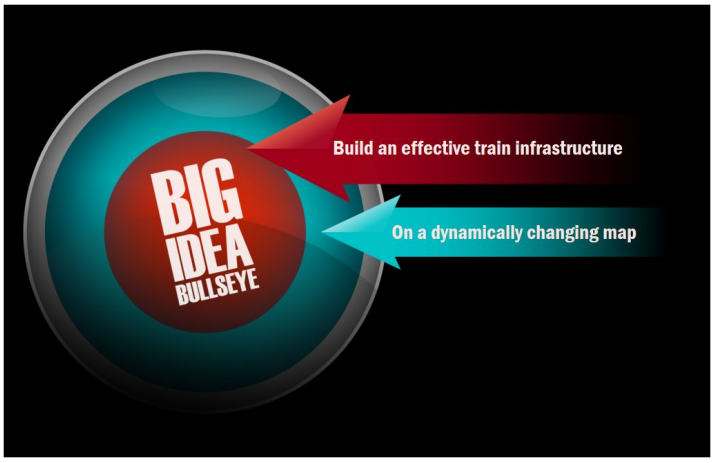
Game Modes

- Survival: The water on Mars is defrosting and there was a kind of dangerous hidden frozen lifeform for millions of years. The colonies on Mars have to be advanced enough to be able to face the aliens by the time they break free. This means to advance into the research tree until one of the leaf is reached.
- Sandbox: There is no timing or impending danger, the player is free to build to their heart's content. The player can also choose some objectives (achieve a certain total population, earn X amount of money, etc.) to compete in a scored leaderboard.

The game parameters (overall amount of resources in the map, frequency of asteroids hitting, etc.) can be configured for different difficulty levels and the first time the game is played some helpful information appears on the GUI with the purpose of teaching him the game basics.

2 'Big Idea' Bullseye

The big idea is to build an effective railway infrastructure while making sure that cities get as much resources as possible before the player runs out of money, or the map is flooded or invaded by aliens. The interesting technical component of our game is the map changing over time and the player having to account for these changes in their strategy.



3 Technical Achievement

The map will be generated procedurally and it will change dynamically meaning that over time cities will evolve and grow and the map will change through random events. The initial map will be generated procedurally through a combination of noise functions and scattering of interesting regions, such as small villages, mines and glaciers. During the game, the map can be affected by events such as rising water levels due to glaciers melting, asteroids hitting the planet, diseases spreading through cities such that isolating the railway network might be necessary (cities with lack of resources might be more severely affected), or the atmosphere thickening. Cities will evolve when they receive more resources. The player can help cities to grow faster, which would increase the player's earnings, by connecting railways to a mine containing resources for example. The growth of the city will depend on how much resources are provided to the city.

4 Development Schedule

4.1 Layered Task Breakdown

4.1.1 Functional Minimum

- Points on the map that produce resources, static objectives **!**
- Cities only need one type of resource
- Train tracks can be created in a flat grid terrain (non textures/art yet)
- Basic GUI

4.1.2 Low Target

- Multiple different Resources. Cities need resources that are displayed in its details, resources that each particular city is looking for is marked with **!** sign
- Terrain: Mountains, Tunnels, Height map, Textures
- Changing map: asteroids, defrosting ice and water leveling up, …
- Game won when population reach a certain research level to be able to fight aliens (only leafs nodes of research tree implemented)
- Basic menu - new game, load game.

4.1.3 Desired Target

- Growing/changing cities, connections between cities influence grow behavior
- Dynamic objectives **!**
- Procedural map
- Art for menus and GUI
- Leaderboard (Rank in function of time to finish and money)
- Notice board: news can appear to make the game history more compelling e.g. meteorologist say an asteroid storm could be coming in year XXX.

4.1.4 High Target

- Research tree upgrades
- Diseases appearing and spreading over cities.
- Banking System and possibility of using money of citizens as long as it is returned with some interest.
- Expanded notice board
- Own composed original music soundtrack. Style Inspired in DOOM soundtrack and industrial metal (e.g. Fear Factory). Recorded bass and guitar with sampled drums.
- Animation 2D slides for with voiceover for game intro and when game lost by aliens.
- GUI tutorial
- Configurable game settings

4.1.5 Extras

- Multiplayer
- Water flows - rivers

4.2 Task List and Timeline

Tasks	Time	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14	W15
Deadlines		19.3	26.3	2.4	9.4	16.4	23.4	30.4	7.5	14.5	21.5	28.5	1.6
Prototype		All											
Demo					All								
Interim report							All						
Alpha release									All				
Playtest										All			
Final presentation												All	
Conclusion													All
Functional minimum													
Rendering environment	10h		JG										
Controller input	10h		MW										
Railway creation and costs, profits and maintenance	10h		SD										
Basic resources	10h		SM										
User interface	10h		JMB										
Low target													
Terrain Rendering	6h			JG									
Evolving map	12h			SD									
Assets creation (incl. gameplay sounds effects)	20h/ each				JG	and	JMB						
Adv. handling resources and winning condition	15h			SM									
Basic Menu and Terrain Programming	20h			MW									
Desired target													
Evolving cities	8h							JG					
Leaderboard	6h					SM							
Procedural map	12h					JG							
Dynamic objectives	12h					SD							
GUI Art and Sounds	10h					JMB							
Notice Board	10h					JMB							
High target													

Research Tree	18h	SM
Banking System and Cyborg Escher Control	14h	SD
Diseases and configurable game settings	14h	SD
Adv. notice board and GUI tutorials	10h	MW
Music soundtrack *	20h	JMB
2D Slides Animations for Intro and Ending *	20h	JMB
Miscellaneous		
Finalising		All

*** JMB will work on this also during Easter break on his free time, specially the soundtrack**

5 Assessment

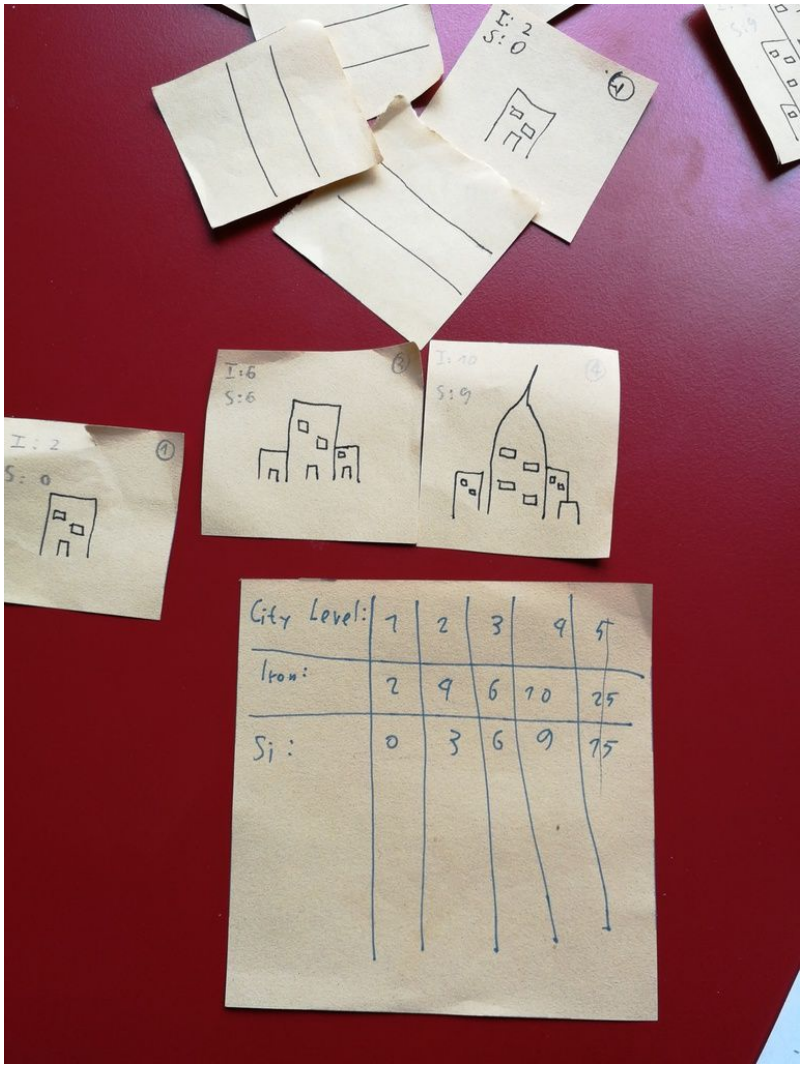
What differentiates our game from most of the strategy games, e.g. Age of Empires, is the absence of a direct enemy to fight. Nevertheless the challenge is to wisely plan where and when to build a railroad, considering aspects such as the current and future demands of the cities, the supply, the capacity and the dynamically changing map. The power of the player is however not limited in building the train infrastructure only, as he can invest money into cities, industries and research, influencing therefore the supply and demand equilibrium and bringing to light innovative technologies. Beside the strategic part, the appeal of the game also consist in the ability of a player to shape the nascent colony on Mars using his own fantasy and imagination.

Chapter 2. Physical Prototype

1. Prototype Setup

The game is played in turns. The player starts with 10 coins that he has to spend to build railways. Each railway tile costs two coins. The goal of the game is to provide resources to cities by building railways. We have modeled cities in four different levels, which represent the sizes of the cities. For each different level a fixed amount of resource supplies are required per turn, for simplicity in the prototype we only have iron and silicon.

If the resources for the level of the city are provided the city upgrades to next level, which needs more resources. If only one of the iron or silicon needs are met then the city remains at the same level, but when none of the resource needs are met the city downgrades by one level. The player loses when some city downgrades to level 0. This means that the player has failed to meet the minimum survival requirements of the city.

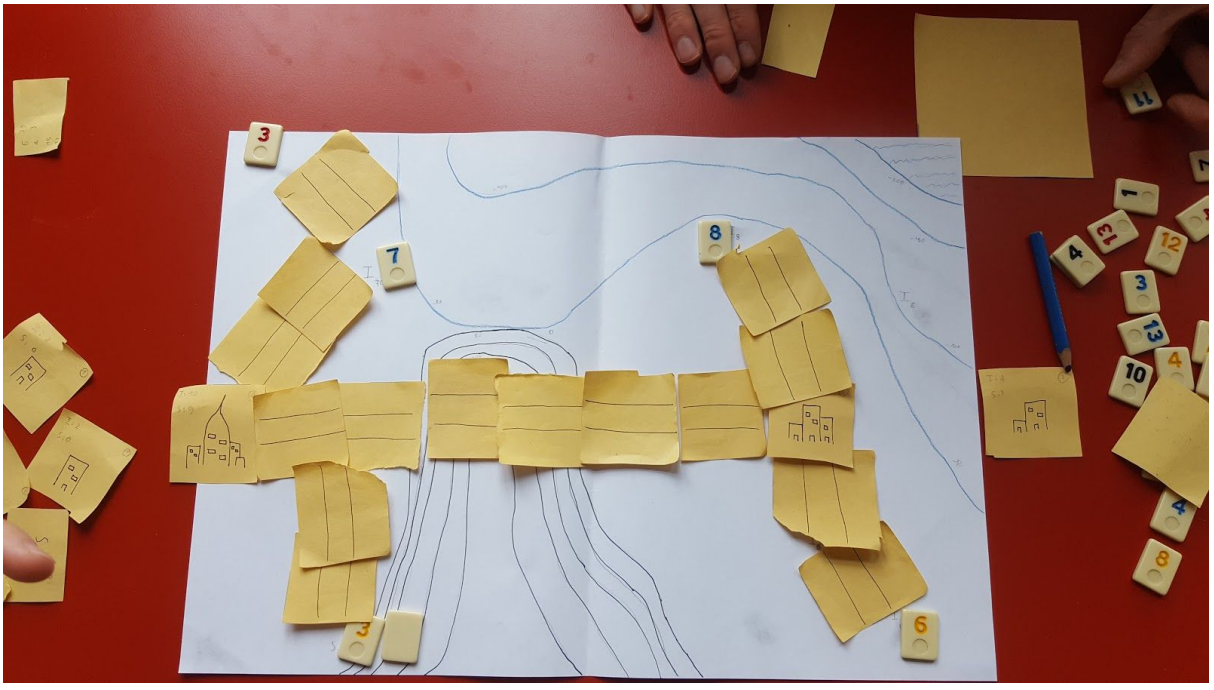


The needs of the cities in resources are:

City LVL	**1**	**2**	**3**	**4**
Iron needed	2	4	6	10
Silicon needed	0	3	6	9

At the end of every turn, the player is paid for the resources transported by the railway tracks. Each unit of resource transported is paid for with one coin. For example, if a city requires 4 units of iron, and a mine on its network can provide it, the player gets paid 4. At the end of the turn the resources used by the cities are taken out of the connected mines, reducing their total capacity. This only happens when the player meets the whole need of some resource for a city. The player cannot fulfill partial requirements. It is either all or nothing for each mineral.

The map is drawn on top of a A3 paper. It contains the position of the cities and resource capacities (we use tokens with numbers to display and update capacity counters), water, and height isolines. Water rises by one level at the end of each turn (the water level is visualized with cut pieces of paper painted in blue). Every city level has its own post-it sketch, where the number of buildings in the drawing symbolize what level the city is at. These notes are placed on top of the city locations and then exchanged whenever a city levels up. The city notes also contain information on what resource needs that level has. Similarly, each train tile is a note with two straight lines drawn on it and these are placed by the player on the map to create tracks. Money is counted manually by another person who is not playing the game.

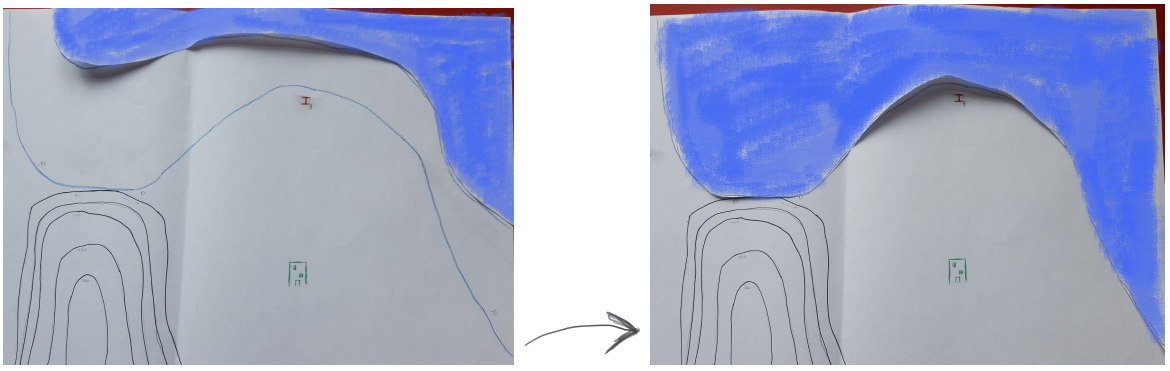


The player has to wisely decide in which connections to invest, as some mines will not be worth the investment at a certain moment. Furthermore, the player needs to plan ahead, and

decide which mines to use first, as some will be flooded later in the game or if it is worth to invest in a tunnel (cost 10 per tile) instead of building the railway in an area that will be flooded by water later.

The player wins if he manages to get both cities to upgrade to level 4 before the map runs out of resources.

Below there are some captures of how we display the water rising in our prototype.



2. Playing Experience

Thinking through your future turns to figure out what the cities are going to need is both fun and challenging. Sometimes it may feel like there is an obvious move that the player should play, but that move could lead to a losing strategy so foresight is required in order to win. However, we could not include all of the complexities that we had planned for the actual game in this physical prototype because it is already difficult to track resources, levels of cities and money manually. And for the physical prototype, we do not want the player to track these themselves so we need to have an outsider, a member of our team, track the progress of the game. This also makes the game more slow-paced than what the finished virtual game will be like. We do not think the pace of the game makes it less enjoyable, but it simply caused the game too feel more like a puzzle game.

One of our team members also let his family try the game and all of them found it enjoyable to play, but some changes had to be made in order to make it more difficult as the balance of the initial iterations was not perfect. For example, some of the resource sources had too many resources available or were too easily accessible. This caused the game to be less fun.

3. Findings and Conclusion

Balancing the game is very challenging and requires to think through the rules, the map layout and the placement of resources thoroughly, otherwise the game would be too easy or unsolvable. This is partially because the game in its current form is fully deterministic, which will change in the digital game with the addition of random events. We also had to evaluate the resource capacities in detail and try to figure out how to prevent players from being able to simply see what is the best choice. On top of that, the digital game should last longer which makes it a lot harder to think through the entire game plan from the map at the start.

However, the balancing issues will also be complicated for our actual game and like we have mentioned before we will need to spend a lot of time playtesting.

We realized that the generation of maps will be more challenging than expected, as we will need to finetune the parameters such that the game is hard enough but not impossible. We may have to implement smarter ‘random’ events that make the game harder if the player is winning and easier if he is losing. For example by spawning more resources if it is impossible to win in the current state or making an asteroid destroy some train tracks if it seems like the player will win no matter what they do. Although, detecting whether the player is currently on his way to win or not will be a challenge as well. Once again, all of this will require us to playtest a lot and figure out how to make the game feel enjoyable, but still challenging for the player. In addition to the random events, we will need to decide the initial placement of cities and resources procedurally. Our algorithm should place these such that they are spread apart from each other, but still not impossible or too expensive to access. For the resource sources, we will also need a method to decide the capacities of each source. We cannot have sources that, once you get access to them, will make the rest of the game too easy for the player.

We removed the rules concerning output of mines and throughput of railway lines. Instead, the limiting factor is mines running out of resources. Now, new mines need to be connected before the cities no longer have enough resources and start to downgrade.

In conclusion, we believe that the physical prototype has shown us even more clearly that the balance of our game greatly affects how enjoyable it is and that randomness will play a big part in making the final game more interesting. The prototype has also proven to us that our idea has a lot of potential and the final product is likely to be fun to play as this simplistic version is also fun and interesting.

Chapter 3. Interim Report

1. Progress

We have implemented completely the functional minimum and low target layers and part of the desired target. However, we still have to balance more the functionalities that have been implemented and the visual style can still be improved by enhancing the lighting, the UI design and the materials of our objects. We also have implemented one of the features for high-target, our own composed dynamic soundtrack.

Level Specification

At the current stage of development, we still don't have any procedural map generation, thus maps have to be generated in advance. To address this issue the user can load different levels from the menu. A level consists of a map and some level specific parameters, e.g. the initial amount of money, the upkeep costs of a railway, the initial distribution of resources, the total research points needed to win the game, ecc. Levels are defined in an XML file, where all the information are stored. However, the terrain of the map is not coded explicitly in the XML file but it is defined as a PNG picture (see section Terrain for more details). Therefore a level can be fully specified by an XML file and an associated image.

Terrain

As anticipated in section Level Specification, the map is not stored directly in the XML file but in a picture instead. The picture has to be greyscale, where dark colors represent parts of the map on a low height level, on the contrary bright colors represent mountains. For the sake of simplicity the terrain is rendered in a discrete way. More in details there are only five different height levels and mountains are considered to be binary, i.e. all the mountains have the same height and a vertical slope. Mountains are drawn as higher tiles in the grid, while height levels that are going to be flooded are represented using different colors.

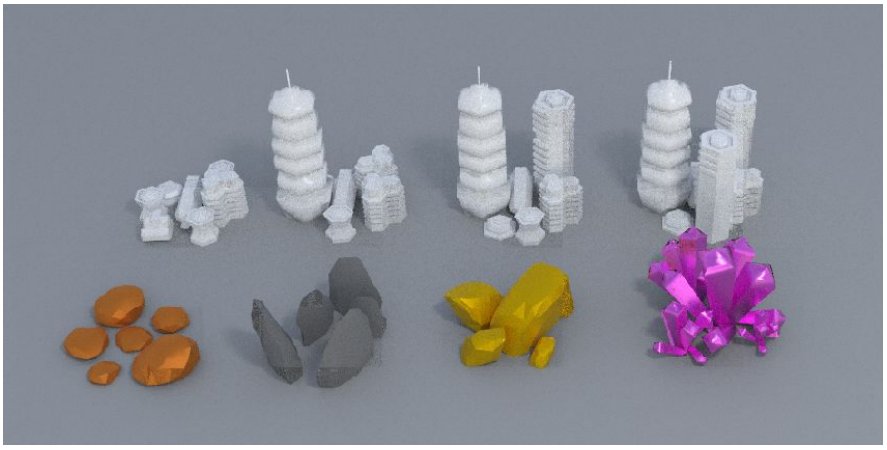
Different Resources

There are 5 different resources as we had planned: H_2O, Escherium, Iron, Gold, and Silicon. The lack of H_2O or Escherium can make the population decrease fast, iron is needed to increase maximal capacity of cities, lack of gold reduce the profits that cities generate for the usage of rails, and lack of silicon reduce the amount of research points a city generates.

Each resource has its own behavior. For example if the city is not connected to H2O the negative effects are seen very fast whereas if Escherium supplies were satisfactory in the past it can take a while until the negative effects are seen when a city is not connected to Escherium anymore.

3D models

We have added new models for the resources and the different city levels. So now it is easier to see the difference between the different resource types and understand at what stage a city is. Below is a render of the new models made with SideFX Houdini.



Random generation of resources on the map

Random events now spawn resources, destroy railways, and flood the map.

Growing/changing cities

Cities' needs for resources are computed based on the population and city level. E.g. A Village do not need any silicon at the beginning but when its level upgrades to a City it start needed silicon based on the amount of population. Cities do not generate research points until the level of City has been achieved (100K population). We create a graph where nodes are tiles on the grid and edges are railway connections and run a Max Flow algorithm to compute how much of the resources each city needs can be satisfied.

GUI

Counters

On the top-right of the screen the GUI displays the total population size, the money the player has, and the number of research points generated. The population size for individual cities and the amount of resources left for a particular resource are displayed on top of their 3D models. The value of the population size displayed to the user is smoothed using an Weighted Moving Average so that transitions of population are not sudden, this is needed because the growth of populations cannot be updated every frame for computational reasons using the graph and Flow algorithm. When a city generates research points or money some animated text is also spawned on the city.

City Names

To create a bit more of “immersion” of the player in the story of the game there is a random name generator for cities. This name generator could be used in the future for the noticeboard to alert the player one of the cities has a crisis in some resource.

Dynamic Objectives
Markers are spawned on top of the cities to indicate the need for one particular resource. Red markers indicate the need is critical and if not fixed population will die, Yellow markers are important and indicate if not fixed soon the situation will be critical. There are pale yellow markers as well to indicate less important needs, such as need for gold that although it will reduce the amount of money available to the player it will not cause the destruction of a city.

Dynamic Soundtrack
Our own original soundtrack has been created for the game and then cutted into pieces and programmed dynamic transitions for the game. The transitions are done based on the change of total population and on the number of active warnings.

- Full sequential soundtrack: https://youtu.be/HvvgFGY1Sec
- Dynamic demo: https://youtu.be/YjM4o7PD3U4
- ‘Asteroid Storm’ Guitar Playthrough: https://youtu.be/m_qXztvZSvo
- Guitar Improvisation: https://youtu.be/JzJs95-iOQE

The tools used to create the soundtrack are:

- Reaper: Digital Audio Workstation (DAW)
- Omnisphere: Synthesizer with many samples and presets
- Bias FX and Bias Amp: Virtual electric guitar effects and amplifiers
- EZDrummer: Sample-based drum software synthesizer
- EZKeys: Tool to quickly lay out musical chords

The following video showcase the Reaper project and the tools used: https://youtu.be/Lzu8O4dgcYk

Game can be won

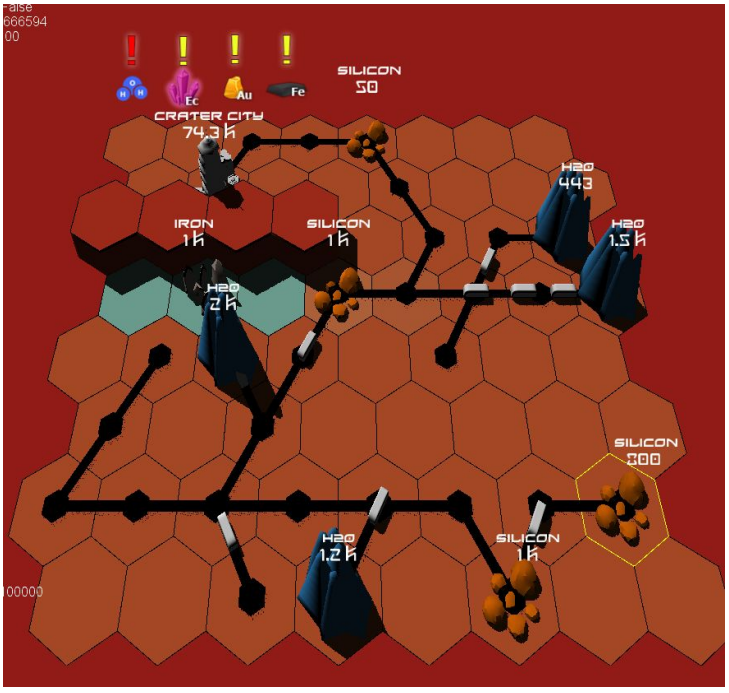
In order to win the game a given number of research point has to be reached. Cities generates research points starting from a given population and only if it gets enough resources. At the current stage a screen indicating that the player has won and a leaderboard are still missing. The user is simply notified by a text popping up and can play further. A game over screen is also missing.

Rendering

Shadow Mapping

Shadows are now rendered for all the models and the grid. We saw that in order to make our game look more modern we would have to upgrade our rendering code and introduce more realistic lighting. In order to achieve this, we decided shadow mapping was necessary. This proved to be more challenging than expected as it was difficult to debug shader code. And, It

is not completely working for larger maps yet as we have not implemented methods that dynamically construct the light's projection and view matrices based on what is visible to the camera.



Creation of train tracks

Tunnels

Building a railway between two points is not always possible in our game. At the moment it is impossible to traverse flooded areas and if there is a mountain a tunnel has to be built. The user can build a railway simply by pressing "A" on the starting grid, moving to the ending grid and pressing "A" again. A* search algorithm is used to find the shortest path between these two points, avoiding all the obstacles. If such a path doesn't exist or the user wants to build a tunnel, the right shoulder button has to be pressed. The algorithm will then return the shortest path between the two points without considering the obstacles. Of course building a railway on such a path costs much more.

Trains Movement

2. Challenges

Even though we have been able to implement some of the features we had planned to implement we still need to work on balancing the game. Another aspect that was challenging for us was to keep the code clean when multiple people are working on the game and each one has their own coding style and preferences. At a few times we had to stop and reorganize parts so that it would make more sense and be easier to add new features in the future.

3. Future Work

Among the features we have not yet implemented or that need to be improved we think we should first focus on the following ones:

- Balancing the game
    - Adjusted growth/requirements behavior of cities (maybe consider cities needs to grow linearly and not exponentially with population - use progress parameter of cities instead of population)
    - Configuration of difficulty and game settings
    - Better randomization and placement of resources

- Generation of height maps
- Rendering of only the part of the grid visible.
- Advanced GUI: Message boxes, Map, calendar, noticeboard, improved menu art
- More advanced materials

Chapter 4. Alpha Release Report

1. Progress

Victory and defeat screens

As the research tree was not implemented, the game ends when enough research points are collected, i.e. the colony has the necessary knowledge to handle the alien menace. If the player doesn't collect enough points on time or the colony dies, it loses the game. At the end of the game a screen indicating whether the player won or lost appears. As there is only one possible way of winning, the victory screen is chosen at random from the set of possible ones (representing the leafs of the research tree). Note that in the case of a victory it is possible to continue playing freely, otherwise the player is redirected to the menu.

Cleaner UI

Big maps require a lot of resources in order to be solvable. Displaying the name and the quantity of every resource could therefore result in a map full of text and a worse game experience. To tackle this problem we decided to only display information (an icon and a number representing the resource type and the capacity respectively) if the resource source is selected (cursor is on the same tile) or connected to the network. This results in a much cleaner and understandable map.

More informative UI

As this is a strategic game, the player needs to know a lot of information about the current demand and supply of a city, otherwise it is not possible to plan ahead. When a city is selected a box containing all the necessary information is shown. Also a warning sound is played when a new red/yellow warning mark appears.

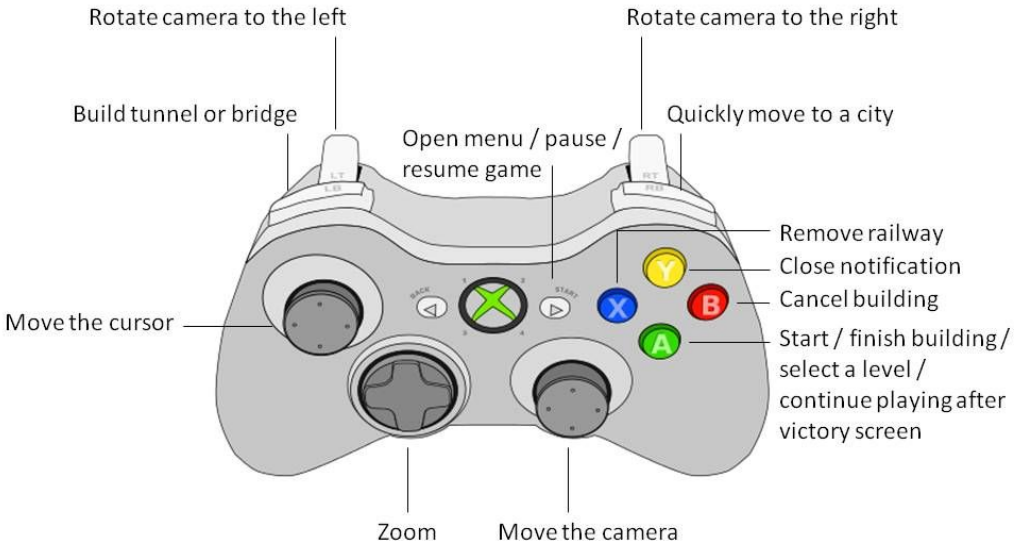
Furthermore sometimes some extra information have to be provided to the user, as in the case of random events happening, level objectives or warnings. A textbox now appears on the screen in such cases. This component is necessary for the tutorial level.

Additional models

To help with the clarity of the game, we created additional 3D models. For example, cities are now more visible on the map because of the suburbs that cover surrounding tiles. These are just there to make cities appear larger, but they do not change anything about cities' functionality. Each resource also has three different models to show how much is left of the resource. Whenever an asteroid hits the map, there is also a small animation of pieces flying off the ground playing. And finally, mountains also change their model whenever they are turned into a tunnel. You can now see the railway going into the mountain through a tunnel and then appearing on the other side.

Improved controls

On big maps we noticed that moving the cursor between different cities can take some time and thus influencing the game experience negatively. Pressing RB allows now the player to centre the cursor on a city. The controls look as follows:

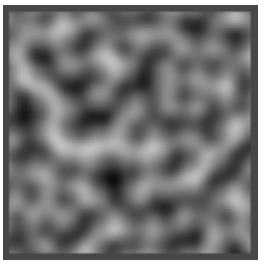


Tutorial level

Since the game is quite complex to understand at the first glance, we decided to create a tutorial level which is designed as a gentle introduction to the game mechanics. Message boxes guide the user towards winning the level and predefined events should warn the user about the effects of some random events during a real game. Note that both predefined random events and game objectives can now be defined in the XML file representing a level.

Procedural map

We have our first version of a procedural map now. It is based on a Simplex noise function which generates a heightmap for the terrain.



Game balancing

For the game balancing we first calculate the minimum amount of resources, money, and research points per second, needed to win the game in 10 minutes by running a simulation of the game with a given number of cities. A margin from these values is added in function of the difficulty (easy, medium, hard). Finally we spawn the calculated number of resources on map and set money and research point generation multipliers.

We also use the difficulty configuration (easy, medium, hard) to decide how much percentage of the resources can be on floodable areas and set the reaction time against the lack of a resource before negative effects start happening. We also do other minor tweakings such as set the price of railways based on difficulty.

More random events

In order to make the game more challenging and fun some new random events were added to the gameplay. Cities are now founded at random if there is too much population in the other cities, collapses in a mine can cause the loss of precious resource deposits, diseases can spread in a city resulting in loss of population and a bad investment might cause the loss of money.

Trains

Rather than being simple eye-candy, trains now display the flow in the network. They only run if there is a flow of resources along a path, and do not visit unused paths. This provides the player with a way of understanding whether a particular railway is in use, or can be removed.

2. Conclusion

We feel that we have come far with our project and it is looking much better now. The user interface is better and easier to understand. The game is also more enjoyable to play now that the game is more balanced and the controls of the game are better. One important addition was the ability to jump between cities. However, the balancing still needs a little bit more tweaking and our controls could be improved a little bit somehow. For example, it can sometimes take a long time to get to a resource if the map is large. And in the end game, this time could be crucial as it could determine whether a city dies or not. Another problem with the end game is that, you can sometimes end up with a lot of money and then you are able to connect the whole map and the game almost plays by itself. A fix for this could be to increase the chance of asteroid impacts even more drastically towards the end of the game or to add something as dramatic as an “meteor storm” where a lot of impacts occur within a short interval. Another alternative would be to introduce some “administrative costs” which increase exponentially with the number of rails.

We also have plenty of work to do on making the game look more visually interesting. We have a skybox feature added as well, but we are not currently using it as we have not found any fitting environment map for our game. There are also some fixes that could be done to the user interface and the models. Our cities are still missing textures and colors. And we are still using the same model for the water resource that we were using in the beginning. The model of the train could also be exchanged to something more visually interesting.

There is still more work that we could do but we are happy with what we have achieved so far. Almost everything in the desired target has been implemented and a few of the high target features have also been added to the game. The game itself is finally enjoyable to play but still difficult. However, we still need to finetune our parameters to find the perfect balance between fun and challenging. And we should also spend some more time polishing the look of the game. But, we are really interested in seeing what kind of responses we will get from our playtesting and these will help us decide what to focus on the most for the final version of the game.

Chapter 4. Playtesting

1. Format

We opted for several different playtesting sessions instead of a single big one for several different reasons, namely: organising small events is easier than setting up a big one, small sessions with less players might give more focus to the actual game testing, and ours is a single-player game, so we are not interested in observing how players interact with each other. The playtesters were in fact able to play for 10 / 20 minutes without waiting too much before starting. There was also more time to discuss about their feedbacks later. Soft drinks and beer were offered as refreshment and a thank-you gift.

2. Players

Fourteen different people overall participated to our game testing sessions. They come from different contexts, e.g. friends, family members and students living in the same accomodation of one of our team members. A lot of them didn't have many experiences with videogames (a bunch of them had never played any video games before), but they still managed to play and enjoy the game, although with some difficulties.

3. Tasks and results

Every player was asked to follow the tutorial first and then play one or two of the easy levels once they successfully managed to complete the tutorial. Surprisingly many of them didn't manage to win the tutorial at the first try and needed to try again once or twice times more. This happened especially to the playtesters with no or little experience with the XBOX and videogames due to the fact that they didn't feel very confident with the controllers and took too much time to perform the requested actions. Another bunch of players, even the ones with more experience, felt confused at the beginning about the goals of the game. More about these points will be discussed in the next sections.
All the playtesters managed to solve the tutorial after at most three tries and were willing to play an actual level afterwards. The two options we offered them were Level 1 Easy and Terrain Generation Easy. In the first case, the map is always the same and designed by us, while in the latter the map is procedurally generated. Even though only four players managed to win one of the easy level, all of them managed to play until the end without dying and were very close to win. They lost in fact only due to the time running out, before the amount of research points needed in order to win the level was collected.
One of the most common error was not connecting the two cities at the correct moment, resulting in the death of one of the two cities. In this way only roughly the half of the optimal amount of research points can be collected, which is not enough to win the level in time. The strategically optimal strategy is in fact to connect the cities at a certain point in time, such that they share the same rail infrastructure and resources, effectively allowing the player to treat them as a single city. The overall impression was therefore that players who were not playing strategically lost the game. As this is a strategic game this consequence was to be

expected, but maybe the easy level should be even easier to allow the new players to start feeling more confident.

3. Feedback and observations

During the game and after that the game ended we collected a lot of interesting feedback from the players, both positive and negative. We will start with the negative ones:

- Initially a lot of players couldn't understand what the game is about, and therefore didn't know exactly what they were supposed to do in order to win.
- The tutorial is too simplistic as it doesn't expose the players to all the possible game dynamics. For example rail upkeep, the fact that unused rails do not produce revenue, and different consequences to shortages of different resources are not explained in details. This resulted in an initial greedy and expensive strategy of connecting all possible resources even if it is not needed.
- Long text messages were too small and thus not easily readable. The problem was quickly fixed after the very first session.
- The controllers were initially difficult to use, especially for building tunnels / bridges, as the player is supposed to keep LB pressed to enable more special and more expensive constructions.
- A lot of players asked about the story of the game and thought that it would have been useful to understand the game dynamics quicker and in a deeper way.
- Playtesters found it difficult to distinguish among different resources on the map, especially gold and silicon which both have similar models and similar colors.
- Another element of confusion is that some players thought that the icon appearing on the tile next to a resource was another resource.
- Smoothed population is unintuitive and unclear, as there are situations when a city dies but the displayed population is still bigger than zero.
- The quadratic growth of research points confused some players as they couldn't understand until the end how well they were performing.
- Some playtesters suggested that displaying a list of the cities with their needs on the edge of the screen would be useful.
- Some competitive players asked for a leaderboard such that they could see how well they performed.
- Some players found the text warnings notifications too big and annoying, and did not realize they could dismiss them.
- There is still some cluttering in the UI, with text from different resources and cities overlapping.

Luckily we didn't only receive negative comments and possible suggestions, but also positive ones:

- After getting familiar with the controllers the players found the game to be exciting and fun to play. We could also see it by looking at the player's expressions.
- They liked the fact that sometimes the game becomes very intense and one has to react quickly, while in other situations one can carefully plan ahead.
- They found some random events funny.
- Playtesters appreciated the soundtrack a lot and they were interested by the fact that the music was changing accordingly to the current status of the game.

- Testers were not bored by the game, on the contrary they didn't notice that they had played for around 20 minutes.
- A player appreciated some of the art design, particularly how the hexagon shape of the grid is repeated and re-used in other instances (for example city models).

4. Conclusions and future plans

We found the game testing session very useful as we had the opportunity to hear the opinion of new players regarding our game. The session was also very useful to spot some bugs in the code. After analysing the results of the playtesting sessions we came to some conclusion and determined therefore our future plans.

As the game has to be stable we decided that solving the spotted bugs has to be prioritised (at least for the ones which could potentially break the game). After that we would like to offer the players some new features which improve the usability of our game. For example we decided to color the border of the tiles which contain a resource with a resource specific color. In this way players can distinguish resources more easily.

A very important part of our future work is to redesign parts of the game in order to help the inexperienced players getting familiar with the gameplay as soon as possible. It would be very helpful to make the tutorial more informative and complete and to integrate the game story more into the gameplay, e.g. with some animations and notifications. Another very important issue that needs to be solved is the controller: we will need to think carefully on how to simplify the construction of tunnels and bridges.

Lack of important information seems to be an overarching problem, with the tutorial not teaching all mechanics, and some useful information not being displayed by the game. We will address these issues by improving our tutorial and UI.

Even though this has nothing to do with the playtesting session we decided that implementing the research tree should also be prioritised. We believe that it will make the game more interesting and make the late stage of the game more fun. It also fits very well with the background story.

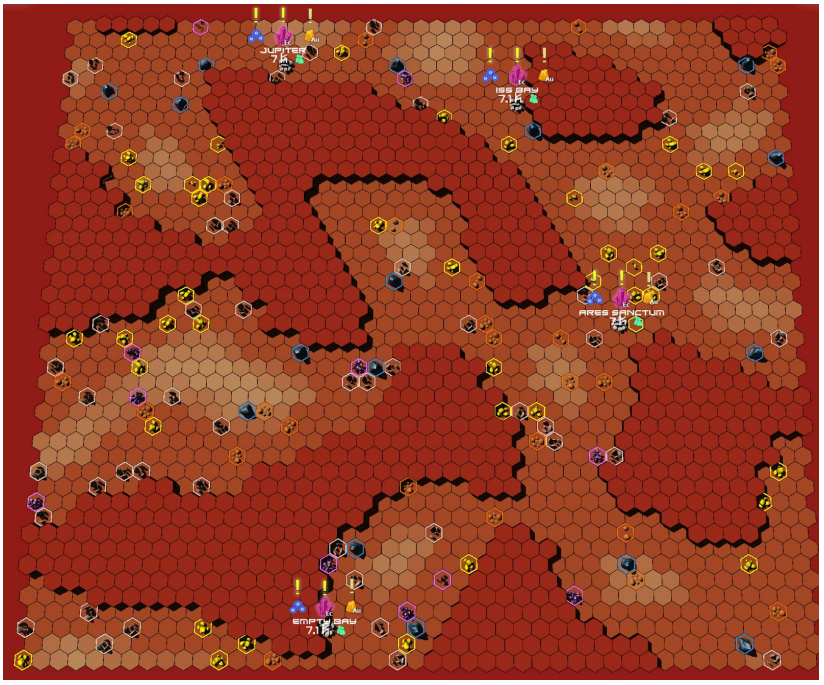
If we have enough time we will implement a leaderboard as well. On one hand having a leaderboard can engage competitive players more, but on the other hand this is not a necessary feature and it makes little sense when coupled with the procedural map generation, as every time the player is going to play on a different map and the performance is thus not always comparable.

Chapter 5. Conclusions

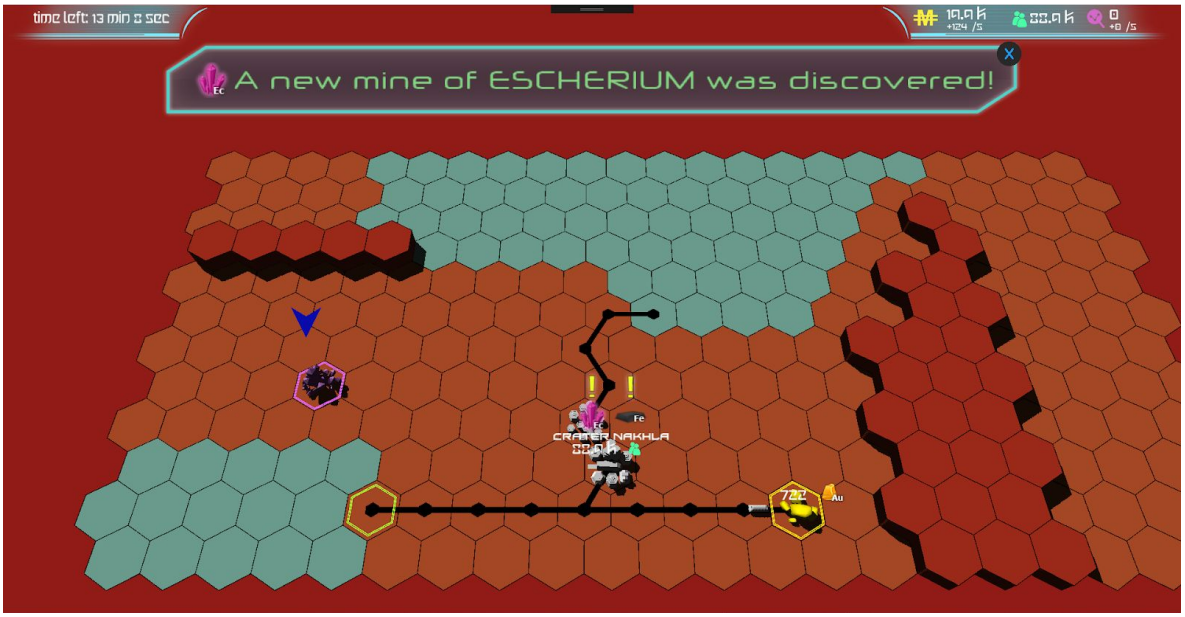
1. Evolution of the game

The game didn't change a lot from the alpha release. Most of our efforts were thrown into finding a more efficient way to draw the grid, in order to avoid a low framerate on big maps. We first tried to generate a mesh for the whole grid and draw that but ultimately we used instanced rendering for this which worked much better. Instanced drawing simply copies the objects to multiple positions with different transformations.

Now that the drawing is more efficient, it is possible to play on much bigger maps, thus we changed the size of the autogenerated terrains accordingly to the game difficulty: the easy levels are played on small maps with initially a single city, the medium levels on a medium sized map with two cities and the hard level on a big map with four cities at the beginning of the game. The level timer is always 10 minutes, however note that easy levels are winnable in just 5 minutes.



Other big changes were related to the playability. To make the game more intuitive to play for novice players, we changed the structure of the tutorial such that it is much more informative than it was before. As one of the biggest problem at the beginning is to understand which resource has to be connected, resources are now spawn only when requested and marked with an arrow (only for the tutorial level). The UI was also improved such that it looks a little bit more attractive.



The last big change from the alpha release is the research tree. Now the player can buy some upgrades using the accumulated research points. Although it was in the high target it was a necessary step for the game in order to be completely integrated with the original game story. Indeed the tree doesn't only allow the player to buy some upgrades, but it is also the mechanism which allows the player to win the game. In fact the game is now won as soon as the player buys a complete branch of the tree, i.e. he is now able to fight back the aliens.

2. Experience during the class

First of all we would like to say that having the opportunity to participate to this class, and create a game from scratch with the help of competent people was really interesting and fun. Developing a game requires a lot of resources and time, but at the end it was worth it.
We are very proud that the final version of the game resembles the original idea that we had in mind, even though for timing reasons the game has some simpler mechanics than the ones we had in mind at the beginning, resulting in a slightly easier version of what we were thinking during the project proposal. However this might be a benefit, as the game is already not very simple and intuitive to play and adding some extra complexity might have caused learning problems for potential new players. Another very important aspect is that adding new features changes the balancing of the game, which was very difficult to achieve.

We tried to follow the original development schedule as close as possible, but we had to change quite a lot due to some tasks requiring more time than planned, some others requiring less time and the different skills of the team members. As a matter of fact we exchanged our tasks very often with other team members or we decided to work together on the same task.
At the end we managed to implement all the features in the desired target, except of the leaderboard. Although some competitive playtesters would have liked a leaderboard in order to be able to compare their performance with other players, we decided that the feature had a lower priority w.r.t. other features in the high target, such as an informative tutorial. We had several reasons to implement a better tutorial level instead of the leaderboard. The first one is that the leaderboard is not necessary for the game, while without a good tutorial a player has no chances to understand how the game works especially considering the complexity of our game. The second reason is that, as we implemented the procedurally generated map, every time that a player plays the game, the map is different. Thus it doesn't really make sense to compare the performance of the players on different maps. We still have some hard coded levels, but we thought that spending too much time implementing a leaderboard only for those levels would have been pointless.
In any case we are very proud that we managed to implement some of the high targets as well, such as the tutorial, the research tree, the diseases, a more advanced noticeboard and an original music soundtrack.

Two very important phases of development which made us take important choices about the development schedule of the game were the paper prototype, as we understood that balancing the game would have been very difficult and thus we predicted to invest a lot of time in it, and the playtesting, when we decided that a more informative tutorial was necessary.

Having many different versions to submit and present to everyone forced us to keep a sane schedule where there were no weeks where we did nothing. Without the presentations, it could have been more tempting to take breaks from the project and end up with a ton of work towards the end. Therefore, we believe that having deadlines every other week was helpful to ensure that we made constant progress.

3. Personal impressions of the course

We encountered two different technical difficulties during the development. The first one was to balance the game and the other one was to draw the grid in an efficient way. The first problem was solved by creating a class which simulated how many resources are needed in order to win the game and then we introduced some different tolerances depending on the difficulty, while the latter has been solved using instanced rendering as mentioned before.

Although the game theme was unexpected and strange at the very beginning, we have to admit that it still let us a lot of freedom. Personally we find that having a theme can help, as it limits all the possible games and it forces you to focus more on the game story. This results in better and easier decisions and probably less conflicts about the overall vision of the game. Another very important advantage of having a theme and not knowing it beforehand is that all the team members have to sit down and think about a possible game. Thus everyone takes part in the design decision. Without a theme, or equivalently if the theme is known in advance, there is the risk that a single team member imposes his game idea. We are also happy with the final idea we had for our project. It feels like a creative and interesting idea that is strongly connected to the theme while still unexpected.

After developing our first game, we have gained some understanding of how much work is required to make a functioning game. And thus, we all agree that if we were to make a new game then we would plan the architecture in more detail, using things such as class diagrams, and care more about it. We would also look into other options for the framework.

MonoGame was very useful in many cases. It provided many helpful features, but we also sometimes wished that it had provided a little bit more. Whether or not MonoGame should be kept in the future, depends on what the focus of the course should be. With MonoGame you have to write a large part of the rendering code yourself which is useful if engine development is an important aspect of the course. However if the focus of the course was more on game design, then Unity would probably be a better choice.

A great success of this project is the satisfaction of having a working game programmed from scratch in a few weeks in a small group of people. This is already impressive considering the huge amount of work, the very short amount of time at disposal and the inexperience of coding in MonoGame and C#. However, the greatest achievement was for us seeing people playing and enjoying our game. Observing how some people could have fun by playing our game for more than half an hour in order to win a difficult level is probably the most satisfying experience of the course.
In summary we are very happy about the course, as it gave us the opportunity to learn something about how video game development works and to work on an actual project with some motivated people and under the supervision of experts in the sector. The cooperation with Studio Gobo and the organised guest lectures were very interesting insights about the real world of game industry. It’s a pity that this year there was no cooperation with ZHdK students, as having some visual artists in the team could have had some benefic effects on

the game. Even though we managed to create some very nice looking artworks, it might still have been interesting to cooperate with them.
Concluding we can thus say that the course was a victory!