MicroWar

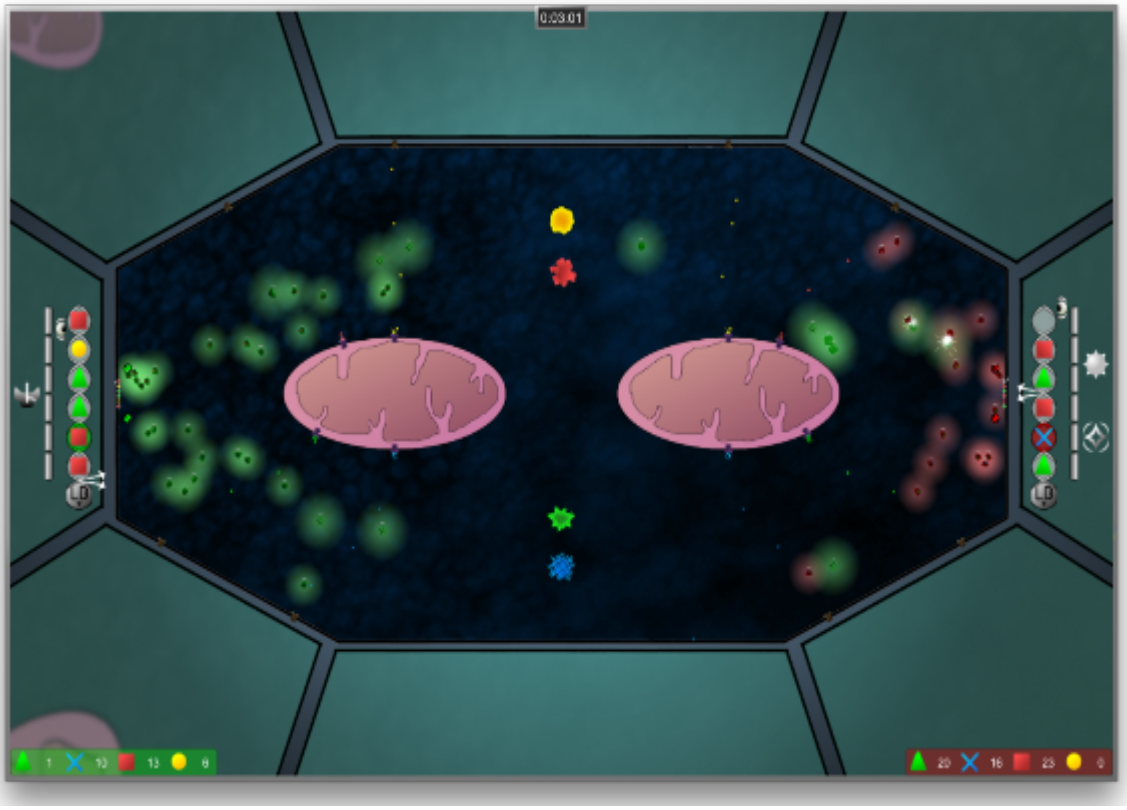
Conclusion Chapter

Team 4

David Niggli, Marcel Lüdi, Gaetano Paganini, Jonas Krucher

Introduction

It was a lot of fun to produce this game. We enjoyed working on it as a team and losing our patience many times. The problems were wicked and the solution sometimes a compromise. Nevertheless the end result should be an interesting game for all people who like strategy games.



“Big idea” Bullseye

Our big idea was:
“Mutate your DNA in order to survive as the fittest.”
In the last few weeks, we were able to create a strategy game, which builds on this basic mechanic. Further led the development and playtesting of the game to a huge refinement of this mechanic.

We defined the technical achievement as follows:
“Smart self-controlling unit behaviour on procedurally generated maps.”
This technical property of self-controlling units was one of the hardest parts to achieve, because the behaviour of an autonomous unit has many degrees of freedom. Our task was, to reduce this degrees of freedom, so we get a balanced and interesting strategy game.
The procedurally generated maps were removed, because it was noticed in the middle of the development process that different maps do not make much sense for MicroWar.
To compensate for that, implementing an AI opponent was a technical achievement that was included although it was not planned at the beginning.

Achievements since alpha release

The final development schedule below is without functional minimum and low target, because we finished this part already in the interims chapter.

Legend:
achieved, not achieved

Desirable target:

- extendable DNA sequence (costs a bit more/or other resource)
- some lifepoint system (protein lifetime)
- some more units: DNA-attacker vs protein-attacker; ranged attack protein; per resource
  type gatherer; area damage, healer etc
- resources flow through the map along some paths
- sound (+ fun sounds)
- single DNA slots are destructible
- resource display
- Units move along different paths on the map

High target:

- DNA translation "scanning" animation
- even more units / building-like units, e.g. resource producing units: produce resources which have to be gathered or directly add to resource pool of player or delivery point for gatherers
- resource: fluid simulation
- more modularity for units

Extras:

- more than 2 players
- players have cells instead of just one DNA. Those can split and multiply
- the DNA creates viruses that can dock on the enemies cells but not on their own
- animations like attacking proteins devouring enemies like macrophages, e.g. particle explosions
- fog of war
- multiplayer different screens, split screen
- AI enemy, which changes DNA (with free camera)
- single player with waves
- "finite resources"
- DNA movement
- scoring system for singleplayer

Lanes

The problem of all attacking units confronting each other on one line was addressed by implementing a lane system with two paths, in which the players decide the direction in which the spawned units are to go. This is strategically interesting as players have to decide where units fight and can invest more on one side then the other in order to push the opponent back.

Finite resources

The problem of having infinite resources on the map was addressed with reducing the resource spawning frequency over time. To compensate for that, resource blocks which contain a certain amount of resources were placed in the middle of the map. Resource blocks vanish when all the resources in them were gathered. As the number of flowing resources decays in the later game, this enforces the players to control the middle of the map in order to have resources which eases the victory for the better player after a certain playing time .

Upgrades

An upgrade system for the attacking units was introduced. Additional resources can be appended to the code of a unit in order to make the unit stronger in some way. For example by appending a triangle resource to a units code the unit moves faster. This can be done as many times as the DNA size allows it, but the improvement gets weaker when more resources of the same type are added to the upgrade. For example if a unit has one triangle appended it gets +25% movement speed. If two triangles are appended the unit gets +25% from the first and +11% from the second. The increase falls off fast, such that it can be prevented that an elite unit spawns which cannot be beaten by anything. The upgrades are strategically interesting as they allow a fast attack on one lane, or a strong defence on an other.

Abilities

Fitting into the game concept, all actions a player can perform result from gene mutations and translation. As a consequence, the ability to trade three resources of one type for one of each other are also performed by inserting a specific code into the DNA. Another ability we implemented is the spike explosion which is intended as a last resort defense mechanism that can save a player who is overwhelmed but could potentially still beat his opponent in the long term. Of course many more abilities could be thought of to make the gameplay more interesting…

Tutorial

As we recognized that our game is far away from being a casual one and that the mechanisms and rules of a strategy game are complex and take a lot of practise to learn, we decided to implement a tutorial that would guide the player through the learning process. This took quite a lot of work and lots of revisions. We hope that our final result is helpful for beginners to understand how MicroWar works.

The ToDo list achievements

During the Playtesting Chapter we recognized more problems and further needs for polishing the game. The result was a long Todo list. In the end we managed to do nearly everything on that list.

Personal Impression

What was the biggest technical difficulty during the project?

The implementation of an autonomous behaviour of the units was very difficult and was also in the final game not always perfect, as for example with the 'shark-like' circling melee attackers around their targets.

What was your impression of working with the theme?

The theme "Evolution" is/was very interesting. It has a huge variety of issues and hence the choice of the basic game mechanics is also big.

Do you think the theme enhanced your game, or would you have been happier with total freedom?

The theme 'Evolution' was quite open and gave a lot of freedom for interpretations. We think the theme led us to good ideas.

What would you do differently in your next game project?

We had a few organization problems like for example when two group members worked at the same thing and the work was done twice in the end. Also the playtesting came in quite short and we were not able to get much feedback, especially for the balancing of the units. This could be improved to adapt the game to be more fun and fair.

What was your greatest success during the project?

The balance of the game was in the end acceptable. This is always a difficult part, especially in strategy games. Although there are still units which could be enhanced further, like for example the feeder which is not always very useful.

Are you happy with the final result of your project?

We are quite happy with the result, but the game could still need some refinements and it would possibly be interesting to have more different units (for example building-like units) and enhance the existing ones. Another improvement could be done with the UI, which was also a suggestion of GOBO after playing the game.

Do you consider the project a success?

Since the game is playable and interesting to play we consider the project a success, even though we could not implement everything we had hoped.

To what extend did you meet your project plan and milestones?

We managed to stay in schedule for most of the time. Especially the playtesting part in the short period of one week was stressful and of course the final presentation and game were a bit more involved and time consuming.

What improvements would you suggest for the course organization?

Maybe have more overview of where we are currently at after each lecture, as that would maybe help to keep track of what has to be done next and how much time is left overall.

Did you like the Unity engine?

Unity is easy for things like collision detection and some physics. But it also has a few flaws and problems due to which it had to be restarted every now and then. Also Mono has problems, for example when it's autocompletion suggestions just breaks down for unknown reasons.
Overall Unity seems to be a good choice for this course, as there is not much time during only one semester and Unity provides many tools which make things easier to do in a short amount of time.