

DADIU May Production 2010

# Game designer's report - Imachination

Written by Jonas Wæver

Teacher: Miguel Angel Sicart Vila

3908 words (but at least it has a lot of pictures too)



## Contents

Introduction.....	2
Conceptualisation.....	2
Pre-Production .....	3
Production .....	4
Post-production.....	5
In Hindsight.....	7

## Introduction

*Imachination* is a pseudo-isometric arcade shooter in the tradition of *Robotron: 2084*, where you can build your own tank-like construction vehicle by dragging weapons onto sockets on a base unit and then use it in arena-like levels to fight swarms of melee enemies, destroy their buildings, and defeat their bosses who drive vehicles similar to yours. The twist of the game mechanics that separates *Imachination* from *Robotron: 2084* and its myriad imitators is that your weapons fire automatically at regular intervals, allowing you to focus on the task of moving and positioning your vehicle in time with your attacks.



I was the game designer on Team 5, dubbed Team We Do It for reasons that should be apparent to anybody who's played *Imachination*. Our office was at Copenhagen University's institute of computer science, and our team counted 13 developers: a game director, a project manager, four programmers, four artists, an animator, an audio designer, and myself. As the game designer, it was my responsibility to design the core game loop, including controls, camera, win conditions, sub-goals, obstacles, enemies, and fail states. It was also my job to define and balance the factors of the game system, mainly the combat system's variables such as health, damage, rate of fire, areas of effect, movement speed, and turning arcs. Finally, it was my responsibility to define what sort of information the player needed and how to make that information available, for example with GUI elements or visual or aural feedback.

I was the game designer on Team 5, dubbed Team We Do It for reasons that should be apparent to anybody who's played *Imachination*. Our office was at Copenhagen University's institute of computer science, and our team counted 13 developers: a game director, a project manager, four programmers, four artists, an animator, an audio designer, and myself. As the game designer, it was my responsibility to design the core game loop, including controls, camera, win conditions, sub-goals, obstacles, enemies, and fail states. It was also my job to define and balance the factors of the game system, mainly the combat system's variables such as health, damage, rate of fire, areas of effect, movement speed, and turning arcs. Finally, it was my responsibility to define what sort of information the player needed and how to make that information available, for example with GUI elements or visual or aural feedback.

On this production, due to a slight recruitment failure on the part of the architect school, we didn't have a dedicated level designer. Our project manager Ali was convinced that we would have to use one of the programmers to implement the levels, possibly assigning an environment artist to improve the aesthetics afterwards, but I was adamant that I would have time to handle the level design, which I realised could easily be constructed modularly for this project. By using Unity's terrain editor and a very limited set of assets produced by our art director, I had no trouble setting up our levels in no time at all.

## Conceptualisation

Shortly after our constraints were publicised on May 15, Ali and I met our director Kristian at the IT University to brainstorm game ideas. Three ideas emerged, one of which was centred around trucks and construction equipment inspired by the construction yard just outside the window at the ITU. Kristian went home and toyed around with these three ideas, emerging with mood boards full of pictures relevant to each of them. We then met with the rest of the team, which in practice meant *half* the team: most of the leads and a couple of others who had time to attend. It was immediately clear that there was much greater interest in the construction machine idea than the other two ideas, so we picked up on that and let the brain storm roll on. We all agreed that narrative is important, but should never get in the way of the gameplay, so the discussion focused on what we wanted to let the player do.

It was decided that we would need two components for the game: a workshop where you build your own machine, and a level where you put it to the test. Kristian wanted a game where you could construct a

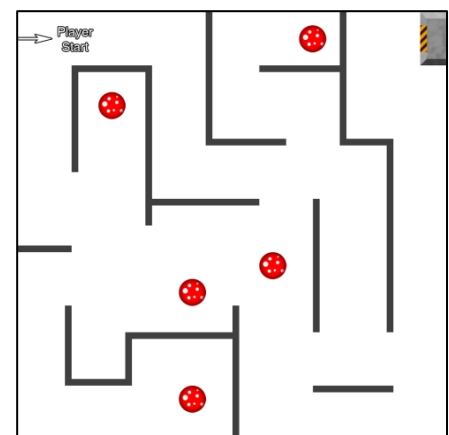
building with your machine and then it would come alive to fight you, much like building a nice LEGO house and then tearing it apart again. Sensing that this game would be extremely demanding to produce content for, and that giving the player a feeling of progression through the level would be difficult, I proposed that each level should be structured like an arena with several buildings that the player has to destroy in order to summon a boss for the sort of climactic fight that Kristian wanted and defeat the boss to complete the level. In order to support the aesthetic that your machine is huge and powerful, an endless stream of small and easily dispatched enemies would be spawned from the buildings for the player to plough through. This would allow us to produce more levels fairly easily simply by rearranging the elements of the levels to change their layouts.

On this and the final conceptualisation meeting with just the team leads, I further proposed a pseudo-isometric game with RTS-style controls where the player's weapons would fire automatically, in order for the game to work with the one-button constraint. The team supported this, and this was the idea we brought with us into pre-production.

## Pre-Production

When we finally met with the rest of the team, we were relieved to find that everybody was immediately on board with the game concept we had formulated, meaning we could start fleshing out the game concept with no further ado. Our director sat down with the artists and came up with a solid aesthetic while I sat with the programmers to plan out the specifics of the game system and the basic interfaces for each layer of the game. Two days later we had a working prototype for each layer and a fairly original setting and narrative for the game. Prior to the pre-production week, I had written a quick synopsis for a satirical super-hero-themed narrative I could propose if nothing better was produced by our director, but the idea of a child digging a hole through the Earth to go to China was much more charming and original than what I had come up with, so I simply never brought up my own setting and story ideas.

Apart from writing the design treatment for our design document, I spent the pre-production week setting up a spreadsheet for the system balance, drawing some level layouts as very simple diagrams in Photoshop, and familiarising myself with the Unity game editor. I also wrote down a long list of possible attachments for the player's vehicle, including armour, engines, and different types of mobility (wheels, robot legs, etc.) which I then went over with the other leads, prioritised according to how different and interesting they were in gameplay terms, and narrowed down to five weapons. In planning the weapons, I made sure to stick to the functionality and leave the rest up to the artists to ensure thematic consistency - the weapons as I designed them were simply a slow impact-based melee weapon, a constantly active drill-style melee weapon, a flame thrower, a laser, and a slow projectile weapon with a large blast radius. The artists decided that a shovel, a bucket-wheel excavator, a flood-light with a magnifying lens, and a TNT launcher would serve these purposes. And, uh, a flame thrower, which is apparently a construction tool.



## Production

During our three-week production phase, my job consisted of equal parts level design and system design, with the additional important function of observing our play tests and deciding how to correct the problems these tests illuminated. For our first play test with the target audience on the Tuesday of May 4, I'd set up all our levels with simple boxes, containing nothing but hives with no functionality other than to take damage and be destroyed. The purpose of this early play test was to establish whether or not the target audience would understand our control scheme in both the workshop layer and the level layer, and whether they could navigate the relatively complex layout of the third level to find all the hives, which was something that concerned our director greatly. As there were no problems understanding the control scheme or navigating even the most complex level, we chose to simply proceed as planned.

As our CG artists were quite occupied generating vehicle, weapon, and character assets (especially characters were prioritised to ensure that our animator would have something to do as soon as possible), our art director Nikolaj took on the task of creating assets for the levels. I was quite concerned with conserving resources, so we came up with a very limited list of art assets that could be assembled modularly to create the levels. A set of hand-drawn level textures were created first, which I could use to sculpt the terrain - the first half was useful for all three planned levels, followed by a small set of level-specific textures.

After the terrain textures were done, Nikolaj created three different wall pieces that could seamlessly overlap each other to form longer walls, each with a specific lava texture for the second level. I used these walls to quickly replace my simple boxes. Finally, a few decorative objects were produced - a rolling boulder for the player to shove around for shits and giggles, and three different crystals to act as light sources in the third level and to serve as its visual gimmick like the god rays in the first level (which I made out of three transparent cylinders and a very basic particle system) and the second level's lava pools. With so few means, and by changing the colours of the lighting in each level, we managed to produce three very different environments.

Unfortunately that meant I was largely done with the levels after our first week of production, which forced me to get around to the unpleasantly math- and spreadsheet-based task of balancing the combat system. Despite my fondness for complicated role-playing games, system design is not and has never been my strong suit, but this was as good a chance as any to get some practice. Thankfully the programmers worked out the movement speed and manoeuvrability of the player's vehicle on their own, so I only had to worry about balancing weapon damage and rate of fire against the health of all our damageable objects. Naturally I got a lot of it wrong, and I had to make countless adjustments throughout the rest of the production in order to make the dynamics of the weapons work as intended, but it was a decent place to start, and it gave the programmers a good idea of what variables we needed.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1		Interval	Damage	Minion multiplier	Minion damage	Minion DPS	Hive multiplier	Hive damage	Hive DPS	Boss multiplier	Boss damage	Boss DPS	Minion kill time	Hive kill time	Boss kill time
2	Shovel	3	10	2	20	6,666666667	3	30	10	3	30	20	3	20	25
3	Digger	0,5	5	1	5	10	1	5	10	1	5	10	2	20	50
4	Flamethrower	0,5	5	1	5	10	0,6	3	6	1	5	10	2	33,33333333	50
5	Laser	3	60	1	60	20	0,25	15	5	1	60	20	3	40	25
6	TNT	5	100	1	100	20	0,3	30	6	2	200	40	5	33,33333333	12,5
7															
8	Player HP		3000												
9	Enemy HP		20												
10	Hive HP		200												
11	Boss HP		500												

The only major change we made to the system balance was to introduce damage multipliers for each weapon on every damageable game object, which was used to introduce a rock-paper-scissors-style dynamic. In order to keep the game simple, the damage per second of our weapons were originally balanced very equally against each other, such that outfitting your vehicle would be a matter of play style and personal preference rather than min/maxing your damage output, eg. do you want to deal 30 damage every 3 seconds with your shovel, or would you rather deal 5 damage every half second with the bucket-wheel excavator? One of the first play tests with our consultants revealed that this probably wasn't deep enough to encourage the sort of replayability we were hoping for. I decided that melee weapons should be more useful against the hives and the minions because they require you to be within the attack radius of these enemies, while the ranged weapons would be more effective against the bosses who are themselves equipped with ranged weapons, and who move fast enough that they're more challenging to hit.

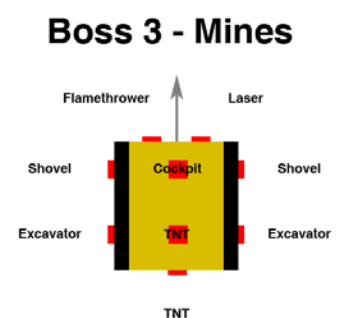
Another problem that the consultant group raised was that killing the hives was essentially downtime - at that point, the hives could not attack you, so destroying them was simply a matter of parking yourself next to them with a melee weapon and waiting for them to fall apart. To fix this issue, we equipped the hives with an area-of-effect shockwave attack that fires every 8 seconds, damaging the player slightly and knocking the vehicle back. This adds a tiny incentive for players to move away from the hives once in a while, but moreover it means something is always going on while the player is attacking the hive.

Finally, it had the hilarious side-effect of letting players "surf" the shockwave from the hives by placing their vehicle close to the hive and then moving away from it just as the shockwave hits, which gives them an enormous speed boost. The actual usefulness of surfing the shockwaves like this is questionable at best, but its entertainment value is indisputable.

Our final tweak creditable to the consultant group was the addition of a special weapon, the Repulsor, which was added as a response to the complaint that the combat felt too passive due to the automated nature of the weapons. It was suggested by the consultants that adding a special weapon that would recharge slowly and could be fired by clicking the player's own vehicle when it was ready could add a more active dimension to the combat. I was completely on board with this idea and actively pushed for its addition, but I'm not satisfied with how it turned out - everybody seems to agree that for the Repulsor to be of any real use, it should not just knock the minions around you back, but also deal a fair bit of damage, balanced against a longer recharge period.

## Post-production

Our final week was dedicated to polishing the presumably by then finished game into a shine. Of course these things rarely go quite as planned, and indeed our plans were sundered by a series of very nasty surprises during our publisher visit on the 19th and our beta test with the target audience on the 20th. On the day of the publisher visit, we were all feeling pretty good about ourselves, considering the game basically finished, having reached feature completion relatively painlessly, and being ready to present the game to the publisher and basking in his praise. Setting aside the extremely embarrassing physics glitch



that rendered our laser and flamethrower totally useless in the build the publisher played (oops), his experience was one of complete failure to comprehend the game's control scheme. I'm still not entirely sure what he was expecting - perhaps that the camera would rotate with the mouse movement - but the result was that he kept running the mouse back and forth across the screen ineffectually.

When the publisher had left, it was clear to us that we had to do something to explain make our control scheme. The entire team had a long brain storm under my direction, and I picked two solutions put forth by our art director and our lead programmer, respectively, and a solution of my own: our art director suggested that we add a ring around the vehicle to show which way it was facing, to address the problem of the publisher not understanding that the vehicle even *had* a front; our lead programmer wanted to make the vehicle turn faster to more visibly tie its movement to the player's cursor; and I proposed that we add a new arrow-shaped cursor that would always rotate to point away from the vehicle, illustrating that the player is controlling the forward vector of its movement. After a long night's crunch, all of these things were added in time for our beta test with the target audience in the afternoon of the next day.



The beta test, however, revealed a whole new set of usability problems. Now that our workshop was no longer a plain grey room with a couple of ugly square buttons, the children had a much harder time figuring out how to use it on their own. The same went for our levels, which in the last test were large rooms with nothing but untextured square walls and a number of hives - now that there were minions, terrain, decorative objects, and a boss garage, the children had no clear idea of

what to do anymore. From watching the children play the game and asking them a series of questions afterwards, I formulated a list of issues to be addressed and returned to the office for another debriefing followed by a long night of crunch.

Some of the new problems that had surfaced were relatively easy to solve; for example, some testers tried to shoot the hives with their laser weapons, only to immediately give up because the laser didn't deal enough damage to start the hives smoking after a single shot. Most of the problems were somewhat more complex to tackle, however, including the related problem of many testers assuming the hives were destroyed once only the base remained, which we solved by adding a health indicator to the red ring around each hive. In fact, our general solution to most of the problems we'd discovered was the addition of a heads-up display: a health indicator along the bottom of the screen to make it clear how much health the player has and whether it's going up or down at present, a similar health bar for the boss once he arrives, and a hive counter along the left side of the screen to show the player how many hives remain to destroy.

Probably our greatest problem was that the players didn't understand what they were supposed to be doing. This called for a more drastic solution - the implementation of a proper in-game tutorial. We sat down to identify a series of controls and features we wanted to teach the player during the tutorial: how to add attachments to the vehicle, how to rotate the vehicle in the workshop, how to start the game, and how to complete each level (by attacking and destroying the hives). Since most of our target audience can't read and we had no time to add voice-over to the tutorial, we decided to teach by simply constraining the player's choices.

A workshop tutorial was devised where the player would be forced to put one or two shovels (and nothing more) on the vehicle, rotate the vehicle, and then start the game, which would be further indicated by large, animated arrows. This also marked the first and only time during the production where I got actively involved in the details of the visual design, when I insisted that the arrows for the tutorial be made to not look like a part of the game world, much to Nikolaj's discontent. In addition to the workshop tutorial, I set up a very simple level tutorial consisting of a single, straight, and narrow corridor with the garage clearly visible on the other side of a single hive, that would leave the player no choice but to move up to the hive and destroy it in order to reach the exit.

I'm not sure our tutorial is quite as effective as we hoped, insofar as we never managed to explain what the Repulsor does and how to use it or how the health regeneration works (stop taking damage for 3 seconds and your health will begin regenerating - it's exponential, meaning the longer you can dodge your enemies, the more you have to lose if you take damage again). Disorganised post-release testing indicates it at least succeeds at teaching the basics of the gameplay however, and assuming your instinct is to run away if you're getting killed - admittedly quite an assumption - the regeneration mechanic should quickly become obvious.



## In Hindsight

I feel that our game lacks the level of consistent creative vision displayed by some of the games created by the other teams, perhaps because our director and I were both so keen to not step on each other's toes or to micromanage the other team members. Everyone on the team has respected each other's roles and areas of expertise, which means the working environment on the team has been extremely positive and everybody seems to have felt some ownership of the game. On the other hand, the fact that both our director and myself have been so generous with sharing our creative power, and perhaps even the fact that neither of us has ever been so passionate about something as to outright fight about it, means the game as a whole may have suffered.

More importantly, the two most defining and unique features of the game are also the most controversial. Our sound design seems to be the sort of thing you either love or hate - many people think our sounds, made by recording children making noises into a microphone, are very charming and amusing, whereas some find them tame and unsatisfying. More relevantly to my work on the game, however, and much more divisive than our sound design, was my decision to have the weapons fire automatically and only give the player control over the vehicle's movement and facing. Somewhere around half the people who have given us feedback consider it a deal-breaker that they can't click to fire their weapons, as they find the lack of direct control unsatisfying. On one hand, this idea came about due to DADIU's constraint that the game must be playable using only the mouse and the left mouse button, but on the other hand my job as a game designer is to make the best of the constraints I'm given, and clearly I haven't succeeded in doing that.

An equal failure on my part was in designing for a target audience of 3 to 7-year-olds. Our director initially pushed to target the 3 to 5 demographic, but thankfully the rest of the team supported me in narrowing the target audience down to 6 to 7-year-olds instead. Even that has proven too low for me to accurately

design for though, as everything indicates that the final game would be better suited for 8 to 12-year-olds. Making a game for this age group would also allow us to add more text to the game, which would help us explain the goals of the game and the dynamics of the combat system much more easily. Unfortunately that is not acceptable in any way - I set out to make a game for 6 to 7-year-olds, and I couldn't manage it.

Based on the feedback of the judges and everybody else who's played Imagination, it's clear that if we were to continue developing the game outside of the DADIU context, we would be best off moving the target audience up to something like 13 years and up, recontextualising it to fit that demographic, and replacing the control scheme with something more involved - for example an MMORPG-style cool-down bar along the top of the screen showing you each of your weapons bound to a number key. Another option would be to let the players bind one or more weapons to any key in the workshop, essentially letting the players define their own control schemes on the fly like the Swedish puzzle game *Bob Came In Pieces*.

If we were to go down this route, we might also add multiplayer arena matches, new singleplayer levels and ranged or flying enemies, and many more attachments such as shields, extra engines, or different special weapons like the Repulsor. It has been suggested that Imagination would be well suited for a Freemium business model where players can pay real money to get the most powerful weapons if they don't have the time to earn them by playing the game a lot, though I'm not personally a fan of micro-payment schemes. Finally, I feel the game would genuinely benefit from the addition of challenges or achievements to reward you for cool moves such as taking out a whole line of 10 minions with your laser beam or blowing up 20 or more minions with a single blast from the TNT launcher.

