Game Programming Lab 2012

Dawn of Time

Mattis Paulin / Romain Prevost / Marco Zimmermann

GAME PROPOSAL

GAME DESCRIPTION

STORYLINE

In the very dawn of time took place an explosion whose magnitude would never be encountered again. It happened 13.7 billion years ago and it shaped our universe the way we know it, the way physicists have spent centuries trying to figure out. Yet, the beginning of ages still remains a mystery, and the truth is harder to handle than most would think.

These were indeed troubled times. Using the powers of darkness, a pitiless entity going by the name of the Dark Architect, spawned antimatter at a sickening pace, in an attempt to annihilate the newly born matter. If the universe was able to survive in this boiling chaos and havoc, it was not without the help of a few brave entities, whose dedication and leading skills ultimately gave them the upper hand.

The Architects, as they were called, each had their designs and views about the way matter should form, and how it should be shaped to build the most robust structures. In the end, the strongest and slyest was able to survive, and using the strength of his dead former companions, pushed the Dark Architect back to the edges of outer space and laid the bricks of the universe we now live in.

GAME CONCEPT

In this Tower-Defense-like game, each player is one of the Architects. Two types of resources (nucleons and electrons) can be collected, the first from a shared source, the second from an individual one. Both sources should be protected at all cost from the antimatter waves thrown by the Dark Architect. To do so, players can craft atoms out of these resources. With these atoms they can build basic structures (atom by atom or by small predefined structures) and special constructions like turrets and reflectors.

Waves strike randomly and aim indifferently at all sources. Sources, as well as turrets and other special buildings, have health that they lose when hit by anti-matter. Between-player interaction is also strongly suggested, as we allow stealing unguarded piles of atoms and specific particle-induced actions.

The last man standing wins. Yet, if the common source falls, the game is a draw between the remaining players.

STRATEGY

Our game framework is designed to render an optimal strategy very hard to find. Players should find a fine balance between cooperation to protect the main source, and defending himself.

To encourage cooperation, players will be awarded more resources as they defend the main source. Stealing from other players should be rewarding but also risky.

Our main goal will be to tune the gameplay to challenge the players in their choices of strategies.

'BIG IDEA' BULLSEYE

RELATION TO THE THEME 'ATTRACTION'

Nucleons and electrons are combined together due to attraction (electromagnetism, and strong interaction). These are the building blocks for all the structures in our game. Our technical achievement will be to determine if the inter-atoms attractions are powerful enough to withstand the gravitational attraction. Finally the antimatter particles launched by the Dark Architect are attracted by the other particles, destroying the work of the Architects.

GAME CONCEPT ARTWORKS

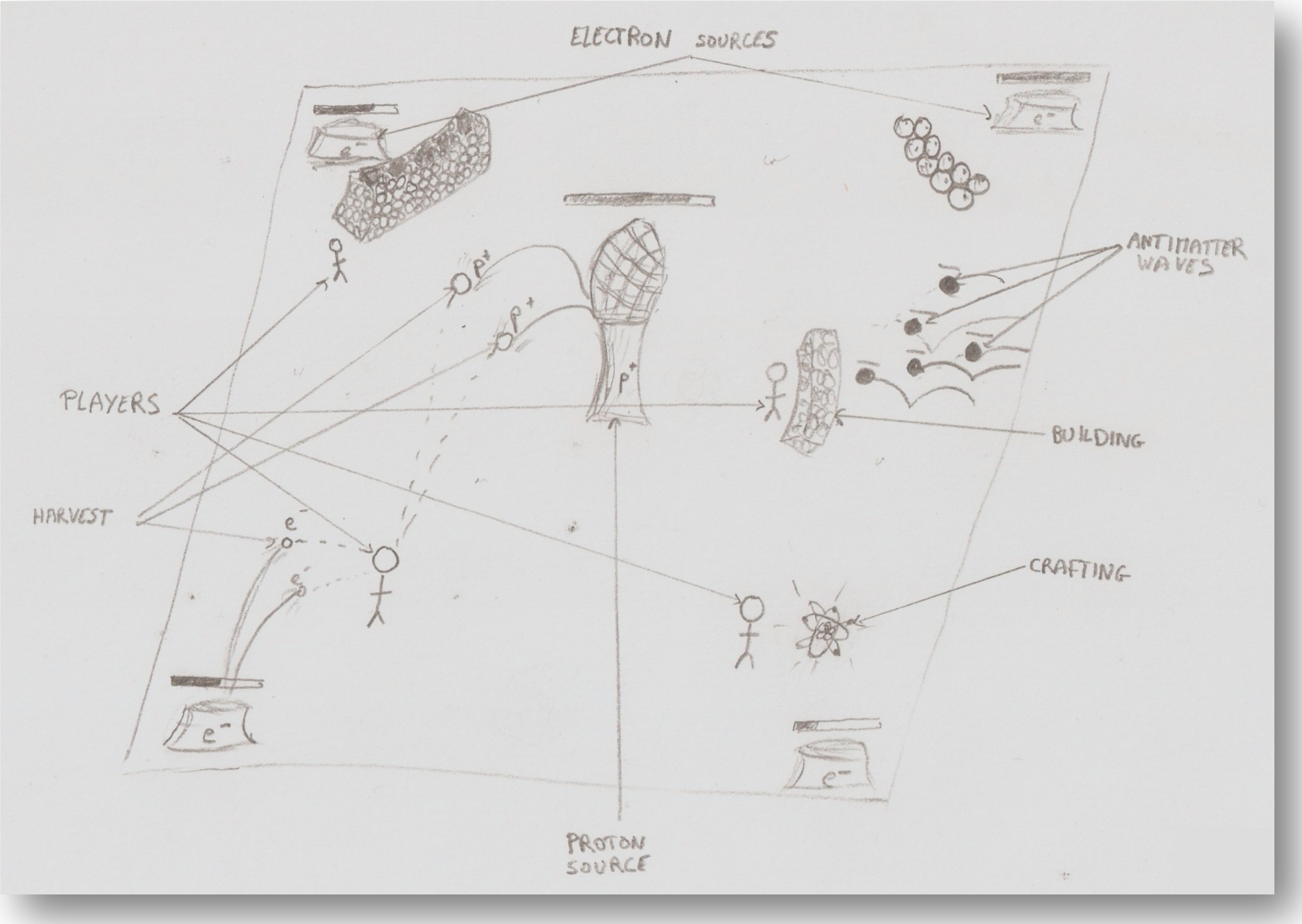


Fig 1: Map design with particles sources and other gameplay elements

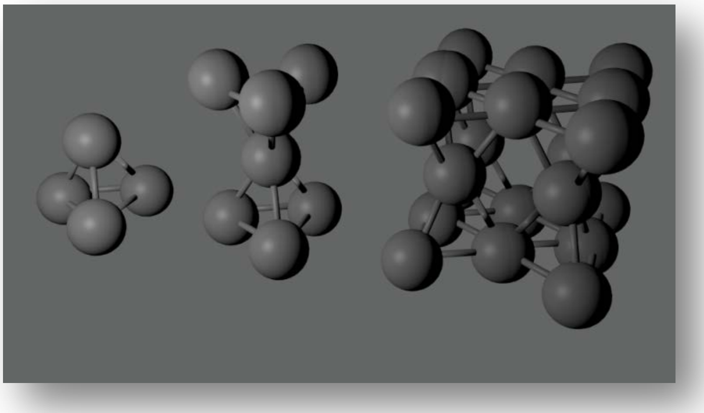


Fig 2: Concept of tetrahedral structures with atoms

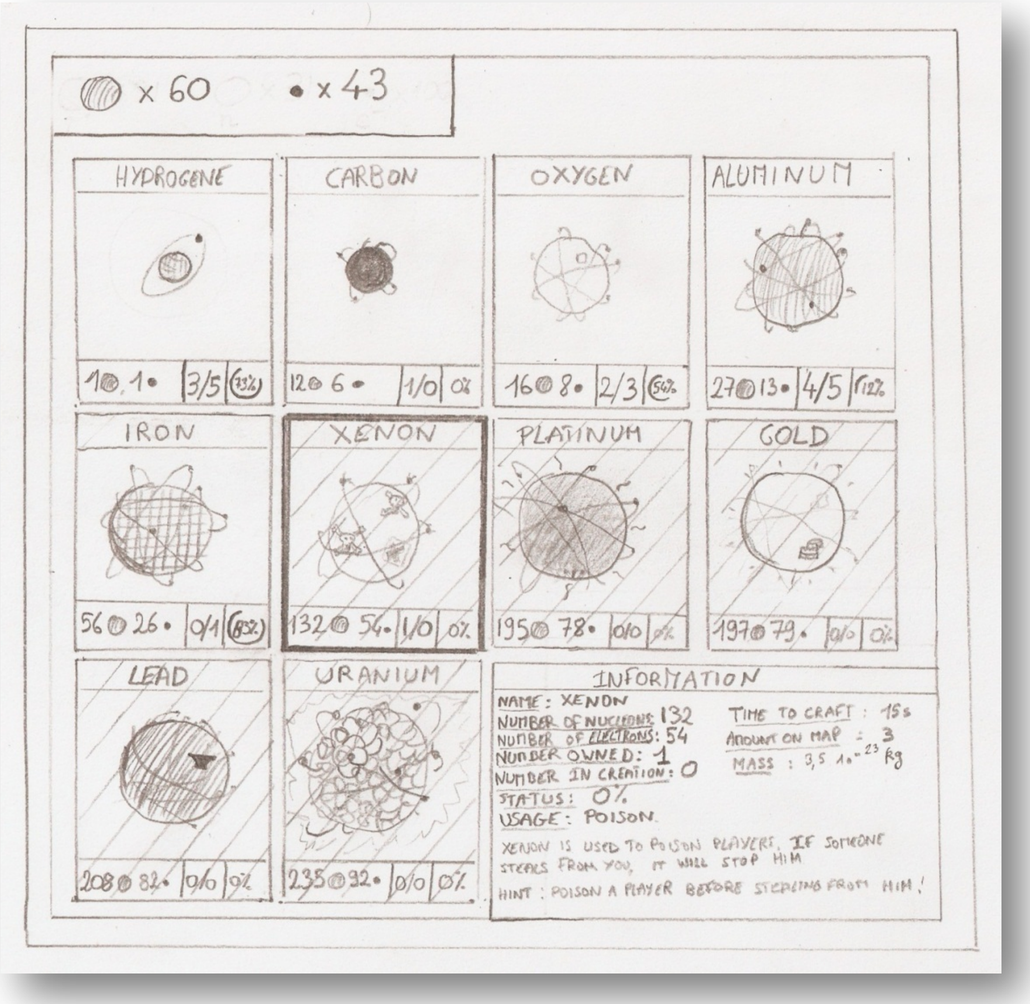
FORMAL ELEMENTS

- **Players:** the game is designed for 2 to 4 players.
- **Interaction pattern:** one of the strengths of this game is the complex interaction pattern, interleaving cooperation and multilateral competition.
- **Objectives:** the main goal of the player is to protect the shared nucleon source and his personal electron source, which provide him with particles.
- **Procedures:** in order to protect himself, the player needs to create atoms with the elementary particles he collected and combine them to build more robust defenses.
- **Rules:** each atom has a particular purpose which is described later in this document. Special buildings can also be built.
- **Resources:** the primary asset is the particles and the atoms that you possess. You can use them to protect yourself and also to attack other players.
- **Boundaries:** the spatial constraints correspond to the map design.
- **Outcome:** Antimatter waves target both the shared source and the individual ones. If the shared source goes down, the game is a draw. The winner is the player whose source is the last one standing.

ATOMS DESCRIPTION

There are 10 different atoms that you can create by putting nucleons and electrons together. You can do it in the player interface really quickly. There are two categories of atoms the ones you can use for your buildings (in green) and the ones with special effects (in red).

- **Hydrogen** (1 nucleon + 1 electron): this is like a snowball that you can throw to other players to prevent them to steal your atoms, or just to annoy them. This will also be the munitions for the turrets.
- **Carbon** (12 nucleons + 6 electrons): this is the basic atom for your constructions.
- **Oxygen** (16 nucleons + 8 electrons): it makes you move faster.
- **Aluminum** (27 nucleons + 13 electrons): this is an intermediate atom for your constructions.
- **Iron** (56 nucleons + 26 electrons): this is a strong atom for your constructions.
- **Xenon** (132 nucleons + 54 electrons): this can be used to poison another player for example if he tries to steal from you.
- **Platinum** (195 nucleons + 78 electrons): this acts as a catalyst and can be used to accelerate the process of building structures.
- **Gold** (197 nucleons + 79 electrons): this is the strongest atom for your constructions.
- **Lead** (208 nucleons + 82 electrons): this atom is less strong than gold, but it cannot be stolen by other players.
- **Uranium** (235 nucleons + 92 electrons): this is an explosive atom and will make huge damage if you throw it on constructions.



NAME : XENON
NUMBER OF NUCLEONS: 132
NUMBER OF ELECTRONS: 54
NUMBER OWNED: 1
NUMBER IN CREATION: 0
STATUS: 0%
USAGE: POISON

TIME TO CRAFT : 15s
AMOUNT ON MAP : 3
MASS : 3,5 10^{-23} kg

XENON IS USED TO POISON PLAYERS. IF SOMEONE STEALS FROM YOU, IT WILL STOP HIM

HINT : POISON A PLAYER BEFORE STEALING FROM HIM!

Fig 3: Help screen for atom information

GAMEPLAY

The main view for the player will be 3D. Using the gamepad the player will be able to move around the scene, to craft new atoms, to throw atoms on his buildings to fix holes, and to use special atoms (for example to oxygen run faster as described before).

The second view for the player is the building framework, in which he can create his defenses. It will basically be a top view around the player position, in which he can use his atoms to build several structures (walls, turrets…). (cf. Fig.4)

BUILDINGS FRAMEWORK

- Wall: A simple template for fast structure creation. Can be destroyed and stolen particle by particle.
- Matter turret: Defensive structure that shoots at anti-matter. Cannot be stolen and has health like sources.
- Anti-player turret: Same as the latter but aims only at sneaky players trying to steal your valuables.

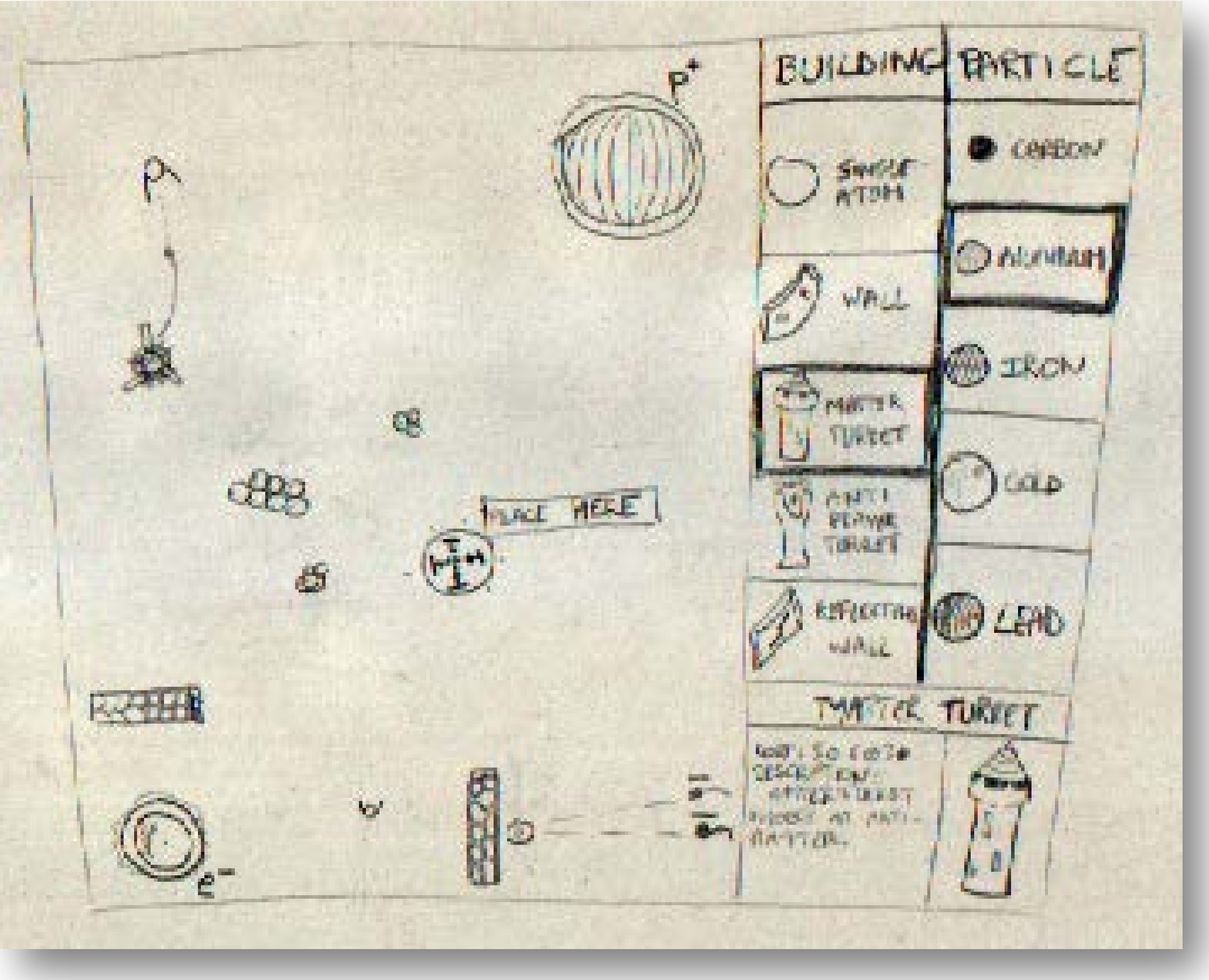


Fig 4: Building screen

DEVELOPMENT SCHEDULE

FUNCTIONAL MINIMUM

At this stage, the player will be able to move on the map, craft atoms and build structures.

- Environment setup (camera, basic rendering…)
- Building screen and game screen
- Basic HUD for atom crafting
- Basic building framework

LOW TARGET

At this stage, we can have several players on the map, but they cannot really interact with each other.

- Collision detection for players
- Basic map with sources
- Basic menus (starting screen, game options…)
- Multiplayer gameplay

DESIRED TARGET

At this stage, the gameplay follows the scenario and you can actually play the game in its whole.

- Antimatter waves and dark architect inclusion
- Better rendering (textures, lights…)
- Character modeling
- Interactions between players (atoms effects, stealing, turrets, …)
- Map design

- Advanced menus

HIGH TARGET

These are features that will make the game nicer to play.

- Visual effects for the atoms
- Physical building simulation using structural analysis
- Sound effects
- Animations for the story

EXTRAS

- Destruction using rigid-body simulation
- Additional physical effects (earthquake)
- Character animation
- Advanced maps

SCHEDULE

A: All, MP: Mattis, MZ: Marco, R: Romain

		6.3	13.3	20.3	27.3	3.4	10.4	17.4	24.4	1.5	8.5	15.5	22.5	29.5
Game Proposal	3h	A	A											
Prototype chapter	8h		A	A										
Environment setup	8h		R	R										
Building and game screens	10h		MZ+ R	MZ+ R										
Atoms crafting HUD	6h		MZ	MZ										
Basic building framework	12h		MP	MP										
Collision detection	10h				MZ	MZ								
Basic menus	10h				R	R								
Basic map	8h				MP	MP								
Multiplayer gameplay	6h					R								
Interim report	6h							A						
Antimatter	16h					A	A	A						
Better rendering	14h					MZ	MZ	MZ	MZ					
Character modeling	6h						R	R	R					
Interaction between players	10h						MP+ R	MP+ R	MP					
Map design	4h						MP	MP						
Advanced menus	6h						R	R	R					
Alpha release	20h								A	A	A			
Playtest	15h										A	A	A	
Conclusion and presentation	20h											A	A	A

ASSESSMENT

WHO SHOULD PLAY?

Our game is designed for everyone who likes to:

- Be challenged: our goal is to make the game decisions really challenging, so that in order to win you will have to find a fine balance between common and individual objectives.
- Kick asses: by using cleverly the different atoms properties you can annoy your opponents and destroy their defenses.
- Be crazy: use as many structures as you want to build your defense.

WHY SHOULD YOU PLAY?

The main strength of our game relies on the fact that each player has his unique style. To get the upper hand, you have to constantly adapt your gameplay and try to find the weaknesses of the other players.

Besides this game is fun in two different ways. First, you can build gigantic and crazy atoms structures. Your only limit is your imagination. Second, you can destroy other player's constructions. This multilateral competition will put a great strain on your ability to make the good decisions.

PROTOTYPE CHAPTER

PHYSICAL PROTOTYPE

BASIC SETUP

For the physical prototype of our game, we chose to make our game turn-based. Every round, players are provided new electrons and nucleons with which can choose to craft atoms, build structures. Alternatively, they can steal from other structures or use one of the interaction atoms (e.g. throw Uranium).

We play the game in 2 player mode and one of us plays the "bank" (providing new nucleons and keeping score) and the Dark Architect. The game is played on a hexagonal grid with small playing figures.

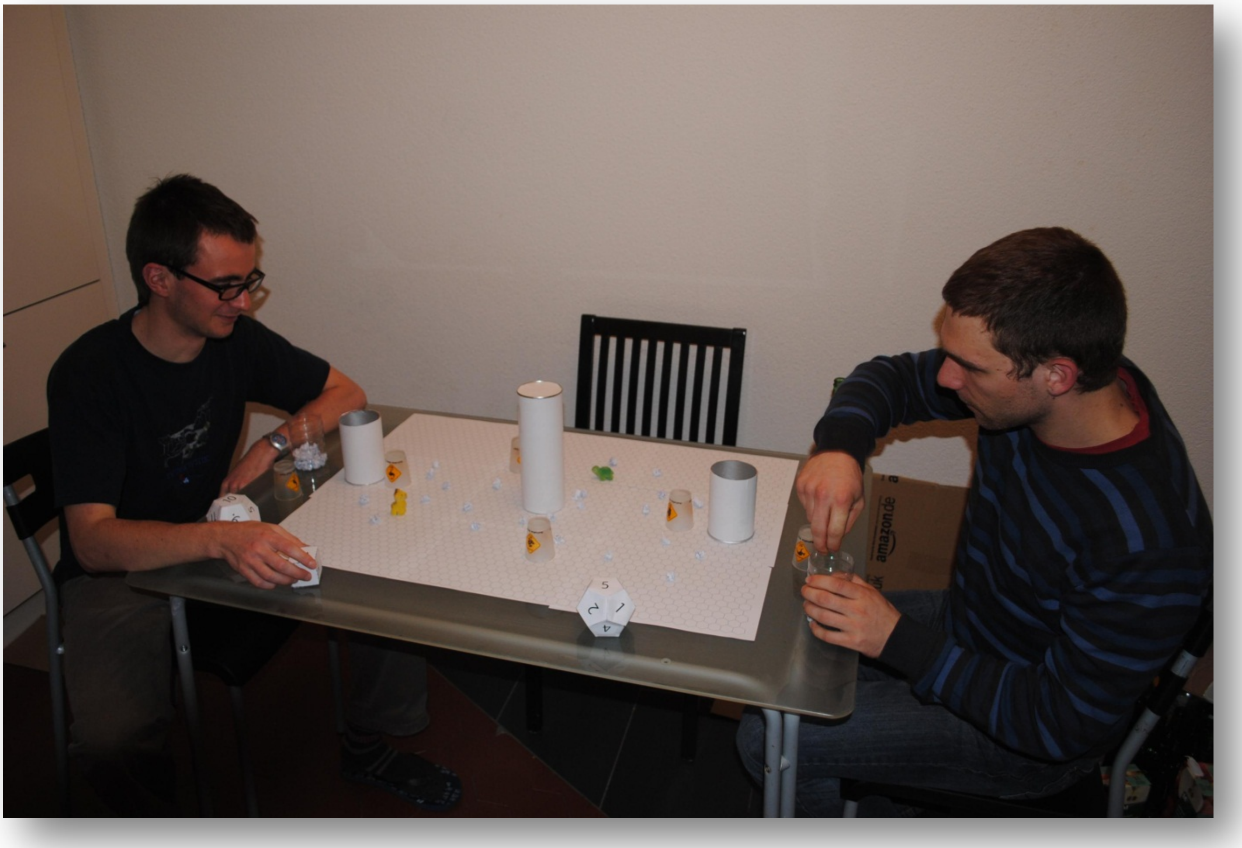


Fig 5. A fun game, for all ages.

PLAYING RULES

Each player gets some amount of nucleons every round. He can first craft atoms ("buying atoms from the bank") and then move and place them. Movement is restricted to 10 steps on the grid (and you may use less.).

A player can buy Carbon for 10, Uranium for 30 and Turrets for 60 nucleons. You can craft as much as you like, as long as you have enough resources. Instead of building, you can also steal from your opponent by moving close to an atom in his camp.

With Carbon you can build your defense: atoms protect the closest source from damage. You can throw Uranium from your position, causing every atom in a radius of 5 to be destroyed. Turrets are used to reduce by one the actual number of anti-matter particle aiming at the related source.

At the beginning of each round, you are given 20 new nucleons from the main source, to which are added 10 per carbon and 20 per turret that protects the center. Cooperation is very well rewarded, but as we may not stress enough: don't be a fool…

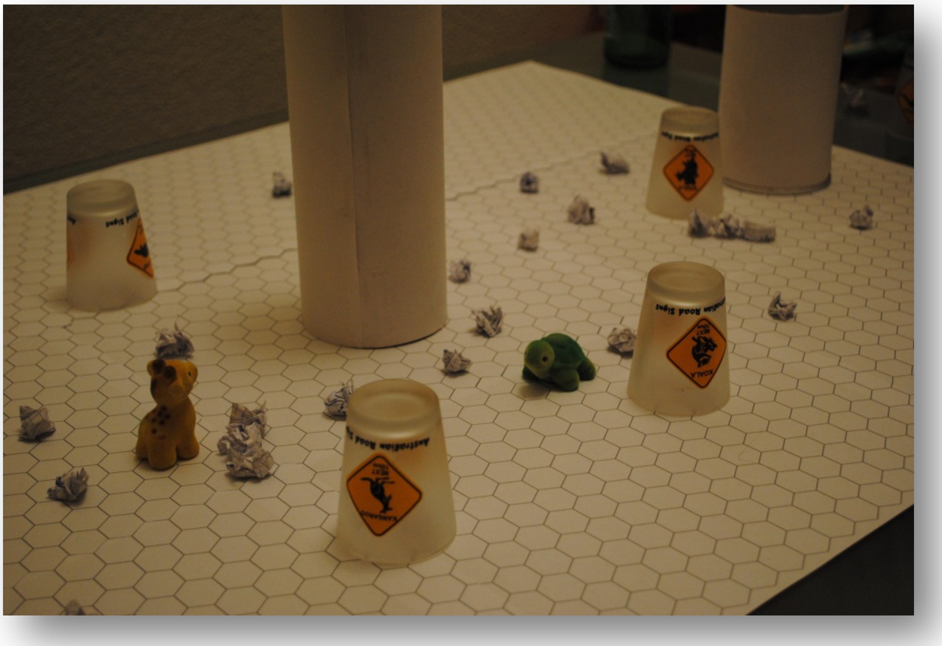


Fig. 6. The board filled with atoms and turrets.

After each round, the Dark Architect rolls two dice to determine the damage dealt to the structure. The minimum of the two numbers is the amount of atoms that each individual player loses from the protection of its own source; the maximum minus the minimum is the amount of damage that the central source will absorb. We multiply the outcome by 2, 3… over time, to make the rounds harder and harder.

PLAYING EXPERIENCE

The rules mentioned in the previous paragraph are the result of a long tuning process: we started to play with almost no rules and eventually added new ones as soon as we hit a point where they were needed. The early versions were started over quite a few times.

We tried a set of rules that induced a longer lasting game, to get a sense of the long term experience. In this version for example, another die was rolled to determine after which round the antimatter would come. It resulted in better preparation and strategic planning from the player, which gave a pleasant touch to the game.

For the presentation, we tuned our parameters to give the game a quicker. We felt that showing all the essential effects and possibilities was more important than gameplay or resemblance to the final product.

We found out to be potentially better not to use the real-world numbers given by chemistry to craft atoms, as it is easier for players to remember simple numbers (like 10 for Carbon). Also we may consider dropping electrons out of the building process by giving player and infinite number of them. This would also give us one parameter less to tune and make it easier for the players to craft, having but one number to care about.

We realized that we were going to need a lot of time to tune our parameters (costs, damages, etc.), but we believe it will work out. The board game is undoubtedly different from the actual one, but we were already able to observe some dynamics that we hope to see in our final programmed version.

INTERIM CHAPTER

DEVELOPMENT STATE

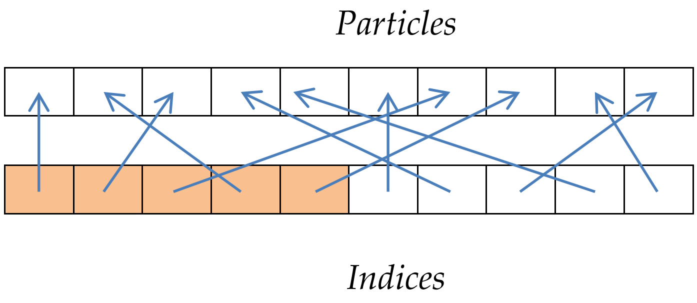
PROGRESS

Our game is progressing well. We finished the low target and are already well advanced inside the desirable target. For the moment, we are on time on the schedule, but we are careful because we think that testing the game and tuning the parameters might require a lot of time.

Some tasks have proved harder than expected. However, the time we spent thinking about software design definitely helped. Furthermore, using libraries in particular for the physics engine, allowed us to stay focus on the game itself.

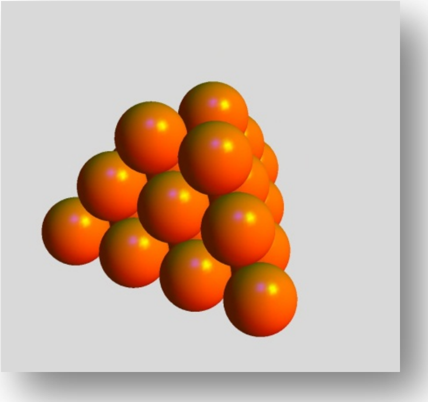
BUILDING FRAMEWORK (MATTIS)

Because we needed to be able to add and remove a lot of particles during the game, we designed a specific data structure that allowed insertion and deletion in constant time, without memory allocation. We used a preallocated array of particles, that acts as memory addresses and an array of integers, which points to particles that are to be drawn.



Keeping at all times a full mapping between indices and particles, defining as active the first k indices and using smart swapping for insertion and deletion, we get our required operations in constant time.

On top of this memory structure, we added a canvas for crystalline structures that determines the position of construction atoms. We used the cubic face centered representation, because it is fairly common and corresponds to a 3D rectangular grid of atoms whose even layers would have been translated half a unit forward and half a unit downward.



Obviously, it would have been impractical to store all possible atom locations; we therefore had to figure out a formula that transformed the 3D coordinates into crystal ones.

PHYSICS ENGINE (MARCO)

We tried out several physics engines and evaluated their performance and ease of use. Specifically, our requirements were stability and performance with a high body count. We are now using Jitter, a very lightweight open-source C# library that implements the famous rigid body paper of Gundelman et al. It's not specifically implemented for XNA, but the guy behind this engine uses XNA himself and therefore the code is well optimized with regard to our enemy number 1: the .NET garbage collector.

The integration was not too hard as the library is easy to use. We used customized collision callbacks to integrate parts of our game logic. Here I lost some time, because the library is completely undocumented. We still have some problems because we add and remove bodies from the physics system a lot, which is not what the developer optimized for.

CHARACTER MODELING & ANIMATION (ROMAIN)

We decided to have nice animated character models for the players and for the Dark Architect instead of ugly colored spheres. In order to do this, we used Maya to design the animated models, and the SkinnedModelSample available on AppHub to import it into XNA.

It took me a while to get familiar with Maya's interface and to practice with several tutorials (modeling, rigging, animating…). Even if I am still a beginner, I feel like I can already do cool things.

To model the character I started from scratch and created the shape I had in mind. It was the easy (or should I say not so hard) part. Then I tried to animate it using a simple skeleton. But after several tests I realized it was too tedious even with inverse kinematics handles. I discovered that I could make this task easier by designing rigging controls. I learned this nice technique from online tutorials and I finally managed to create useful rigging controls on my mesh.

The second part, I am currently working on, is to animate this model. Now that I have my rigging controls I can create poses quite easily. The complicated task is to create an animation which looks realistic. I spend a lot of time studying other games' animations.

Finally the last part is to integrate these animations into our XNA game. We used the SkinnedModelSample as a starting point and tuned it to fit our needs. We tested a simple animation of our mesh and it works. So now, I need to create the different animations (walking, throwing atoms, stealing, sleeping…).

OTHER DETAILS

Dealing with a lot of objects the transferring of vertices to the GPU is quickly becoming the bottleneck, so we decided to implement a basic instancing technique which only transfers the geometry once to the graphics card and then only transfers the transformation matrices. We gain about 15 frames per second with this option activated.

We implemented support for sound handling, using the XACT editor and format. The XNA API in itself did not take long to figure out but we still need to work on sound choice and post-processing.

SCREENSHOTS

MENU

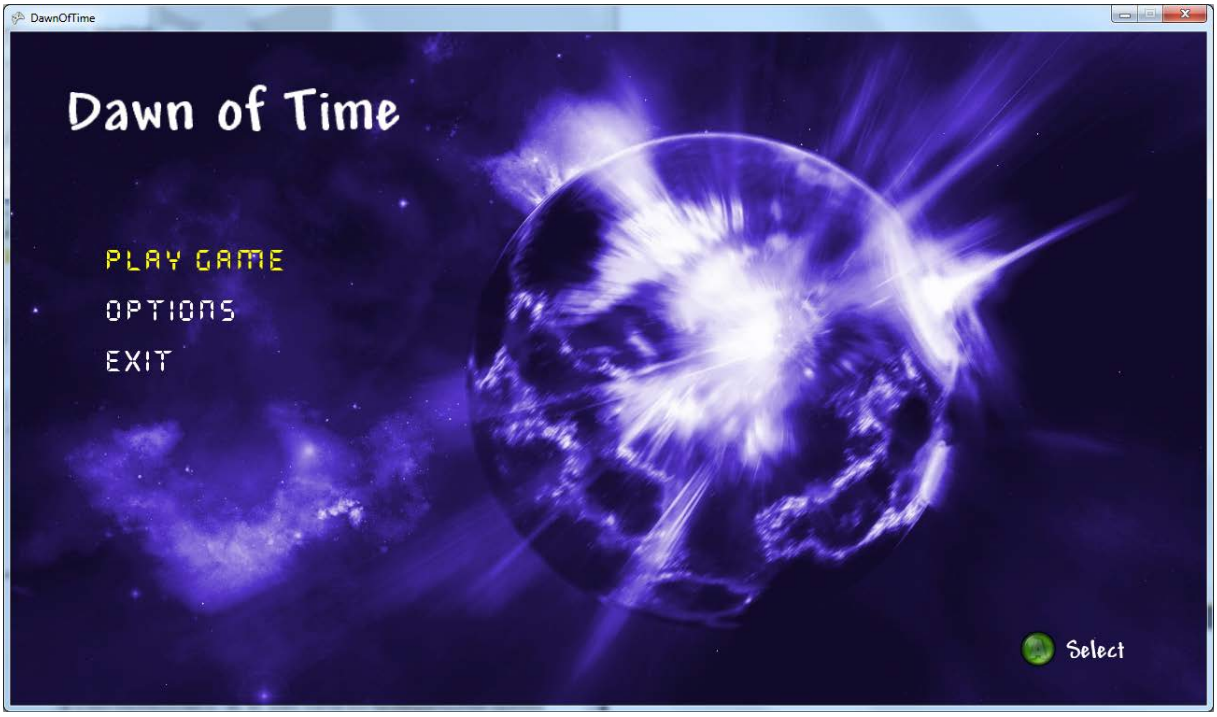


Fig : Start menu

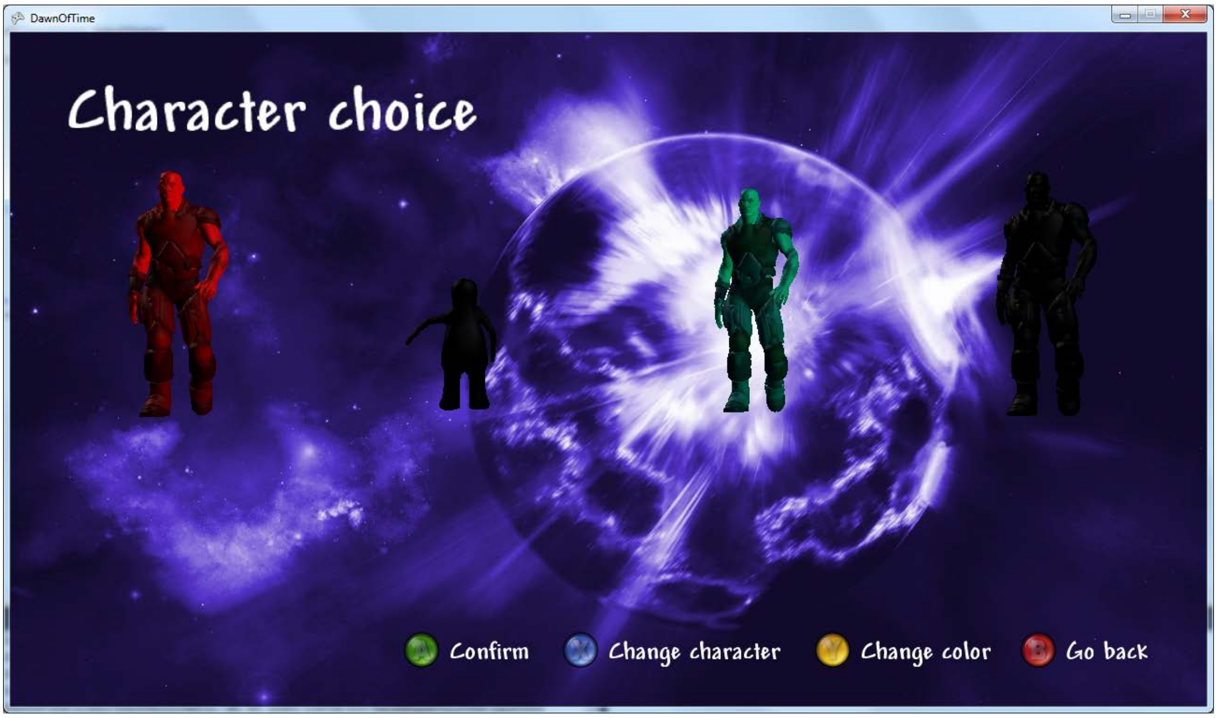


Fig : Character model choice

GAMEPLAY

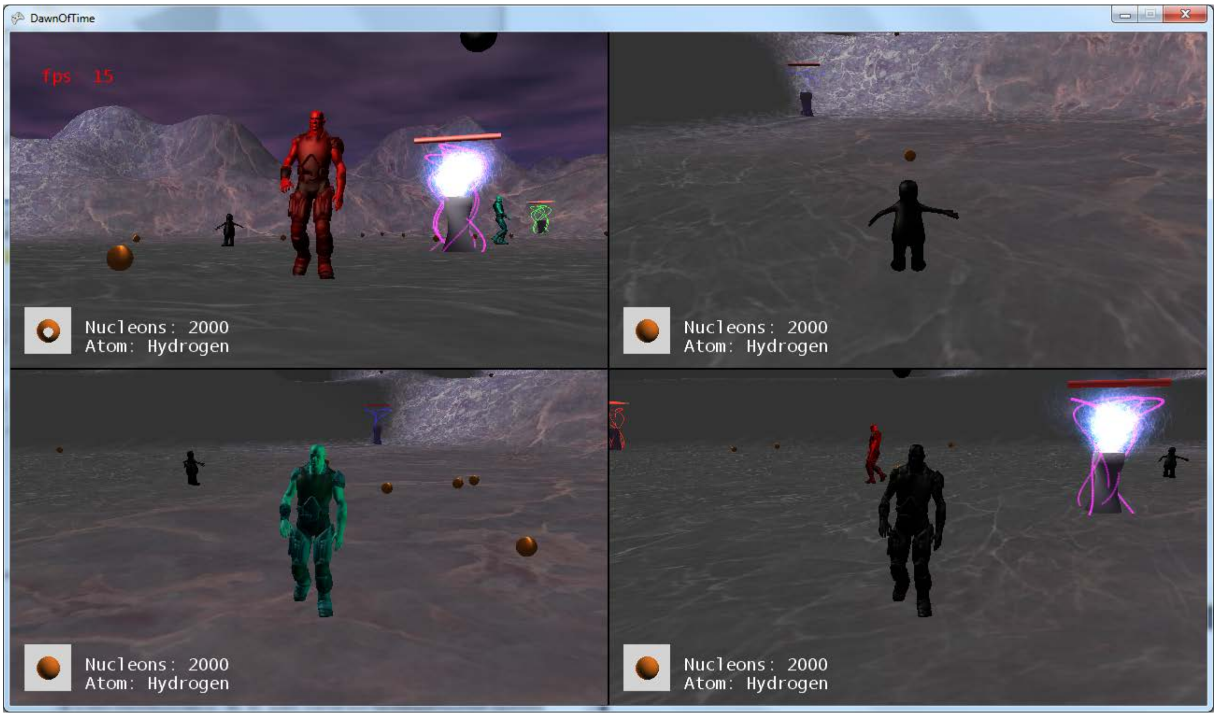
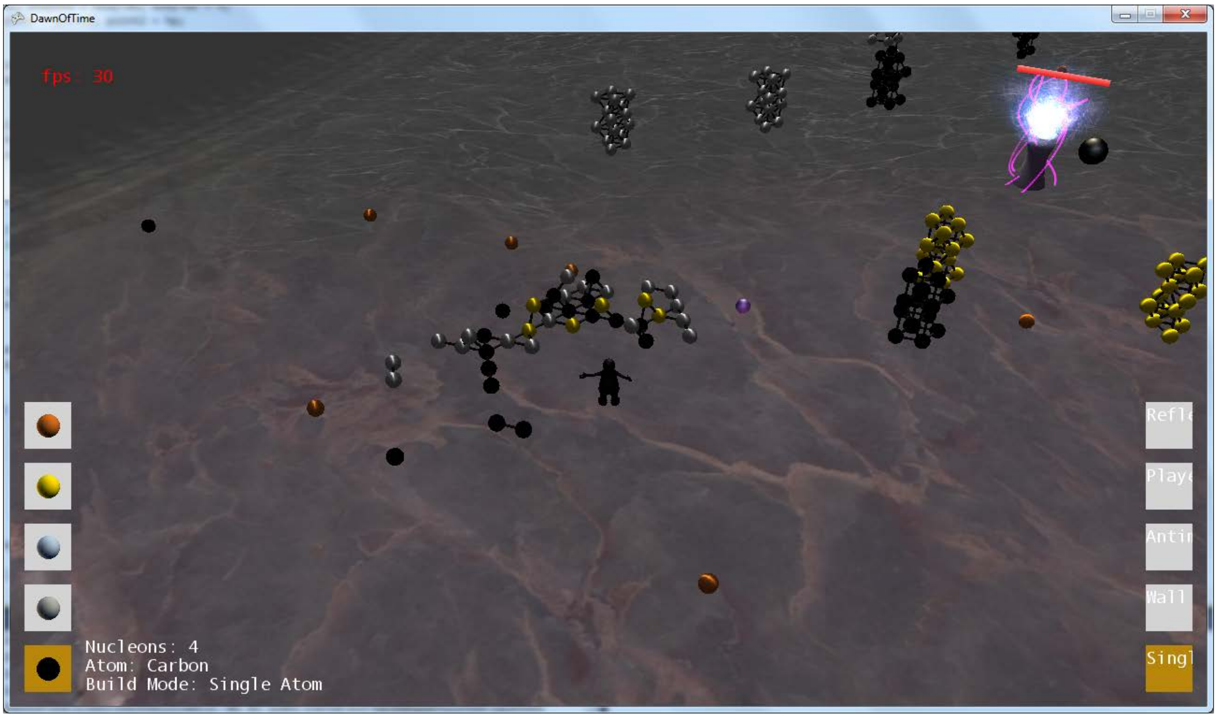


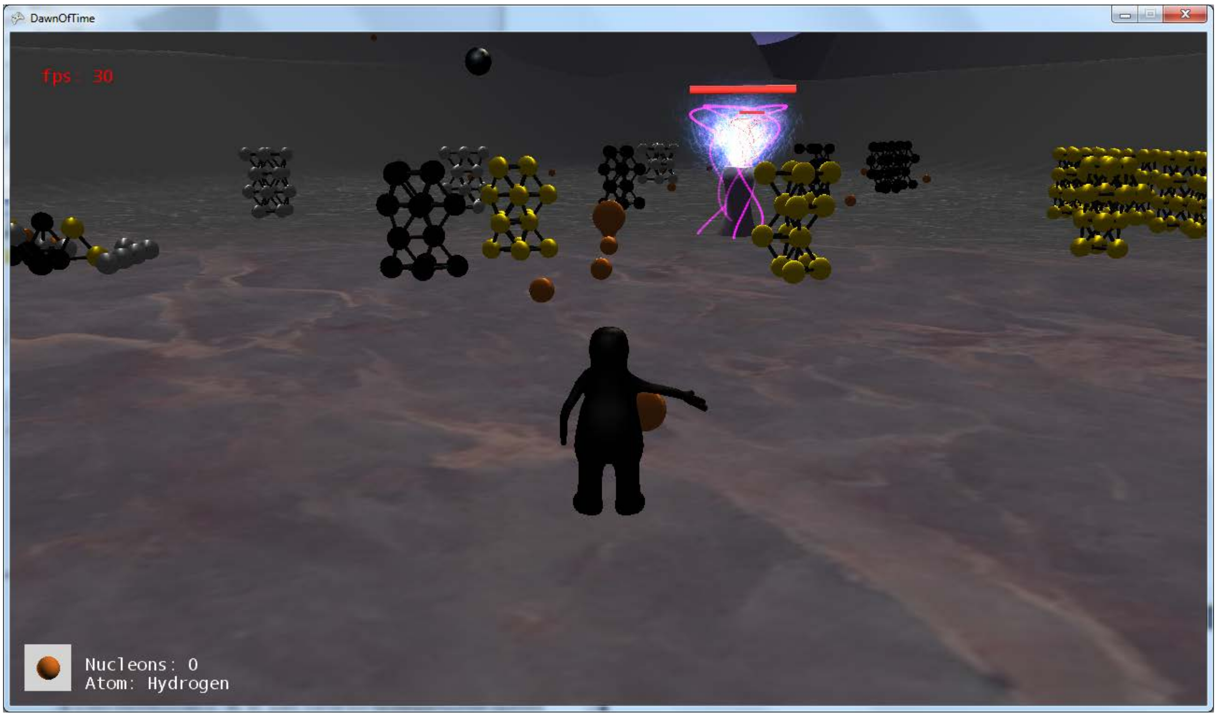
Fig : We can play up to 4 players



fps : 30

Nucleons: 4
Atom: Carbon
Build Mode: Single Atom

Fig : Building mode, you can easily create structures around your position



fps : 30

Nucleons: 0
Atom: Hydrogen

Fig : Game mode, you can move on the map, throw atoms, steal atoms…

GAME DESIGN

GAMEPLAY ADJUSTMENTS

We slightly modified our gameplay since the formal proposal, for reasons spanning from ease of programming to fun improvement.

We decided to remove atom crafting to make the game more fast-paced. Instead, the player directly selects the building or atom he wants to create and the corresponding number of nucleons is subtracted from the player's purse.

To allow for a bit of strategic planning and improve the ability of players to annoy each other, we intend to have a pseudo-realistic AI for the Dark Architect, that would randomly walk around the map throwing waves whenever it sees fit, allowing players to build mazes to redirect it to others. We also plan to have it respond to Hydrogen and Xenon thrown at it. We are aware it will add some tuning work for us but are also confident it will make things more interesting and challenging.

SCHEDULE UPDATE

Target	Task	Scheduled week	Progress	Comments
FM	Environment setup	20.3	DONE	
FM	Building and game screens	20.3	DONE	
FM	Atoms crafting HUD	20.3	DONE	
FM	Basic building framework	20.3	DONE	
LT	Collision detection	3.4	DONE	
LT	Basic menus	3.4	DONE	
LT	Basic map	3.4	DONE	
LT	Multiplayer gameplay	3.4	DONE	

Target	Task	Scheduled week	Progress	Comments
DT	**Antimatter**	17.4	DONE	
DT	**Better rendering**	24.4	IN PROGRESS	Particle system : MOSTLY DONE Lighting : TO IMPROVE
DT	**Character modeling**	24.4	IN PROGRESS	Modeling in Maya : DONE Rigging in Maya : DONE Several model variants : TO DO
DT	**Atoms behavior**	24.4	IN PROGRESS	Integration into physics engine : DONE Action callbacks : TO DO
DT	**Map design**	17.4	DONE	
DT	**Advanced menus**	24.4	IN PROGRESS	Design : DONE Character choice menu : TO DO Winning screen : IN PROGRESS
	Alpha release	8.5		
	Playtest	22.5		
	Conclusion and presentation	29.5		
NEW TASKS				
DT	**Better HUD**	24.4	IN PROGRESS	We are not fully satisfied of our HUD and we would like to make it look nicer.
HT	**Background music & Sound effects**	1.5	IN PROGRESS	Sound integration : DONE Music choice : IN PROGRESS Sound effects matching : TO DO
HT	**Character animation**	1.5	IN PROGRESS	Animation framework in XNA : DONE Animation test : DONE Animations in Maya : IN PROGESS
HT	**Dark architect AI**	1.5	TO DO	Basic walk avoiding buildings Basic behavior choices
HT	**Even better rendering**	1.5	TO DO	Shadows Character texturing