



Dawn of Civilization

4pwny Productions

Emily, Jason, Skyler, & Beky
Harvey Mudd College
Software Design
fall 2009



Table of Contents

| | |
|------------------------------------|---|
| High Concept | 3 |
| Overview..... | 3 |
| Key Features | 3 |
| Competitive Analysis..... | 4 |
| Key Requirements..... | 4 |
| Game Design Rationale..... | 5 |
| Proposed Project Overview..... | 6 |
| Proposed Development Process..... | 7 |
| Risk and Feasibility Analysis..... | 8 |
| Our Team | 9 |



High Concept

“Dawn of Civilization” is a slow real-time strategy that transports you back to 10,000 BC, a time when mankind had to survive in a world of vicious wild animals with only primitive tools to aid them. Guide a small tribe of people as they try to gather food while avoiding becoming food themselves. Nurture them so that they might survive long enough to learn to farm and better protect themselves from a hostile environment. Find the best location to build a settlement and then expand from your small hovels to palaces in a great city capable of trading with the other great powers of the ancient world. Finally, leave your mark by building a Wonder of the Ancient World.

Will you become just another footnote in the annals of history? Or do you have what it takes to stand the test of time?

Overview

“Dawn of Civilization” is comprised of several stages, which must all be passed in order to build a World Wonder and win the game.

- *Hunter-Gatherer Stage:* The player controls a small nomadic tribe and must direct them to gather food, water, and other supplies in order to survive. In order to progress to the next stage, the tribe must domesticate animals.
- *Herding Stage:* The tribe continues to wander the land together with its group of animals. In order to graduate to the next stage, the player must discover farming, the ability to plant and cultivate her own crops.
- *Farming Stage:* The player constructs farms in particular places on the map, and may build permanent settlements. In order to progress to the next stage, the player must build up a surplus of food so that some of her citizens can stop farming and specialize in particular occupations.
- *Specializing Stage:* Once some citizens are able to train in highly specialized work, a culture begins to develop. In order to move to the final stage, the player must develop aesthetics and architecture.
- *Construction Stage:* In the final stage, the player's tribe has grown into a civilization, spanning a large area of land containing farms and cities. Her final task is to construct a Wonder of the Ancient World by dividing her population into farmers, builders, and artisans in an appropriate balance.

It is possible to move back to a prior stage if the player chooses to. She will retain all of her discoveries and techniques from later stages, and will not need to re-complete any tasks in order to progress. However, her farms, buildings, and villages will degrade if they are abandoned.

Key Features

- *Display information:* A variety of messages will be shown in order to further the storyline, provide historical background, provide help with playing the game, tell the user her current goals, and show statistics about the user's current civilization.



- *A map that updates continuously:* Updating the map frequently is important for giving the user quick visual information about what is available for resources and what she has already created.
- *Unlocking new abilities:* As the game progresses, the user gets better at performing tasks such as gathering resources and gains the ability to create new structures, such as farms, that are not available at the beginning of the game.
- *Building structures:* The user can build various structures such as houses and storage warehouses that then affect what the user can do: more houses and storage locations means higher population and food caps respectively.
- *Division of labor:* The user can assign more people to a task that she deems important and less or no people to less important ones. This changes the speed at which a task is completed.
- *Trading:* The player will be able to orchestrate bartering of resources that she has a surplus of for those that she needs.

Competitive Analysis

“Dawn of Civilization” is a turn-based strategy game similar to the Civilization series. Key similarities include:

- *Similar gameplay:* Common actions include constructing cities, farming the land, and researching new technologies. The workforce can be distributed among the different activities based on how important the player considers each task.
- *Similar goals:* The player's main goal in the game is to protect and expand her civilization. She is threatened by beasts, blight, overcrowding, and insufficient resources, and her progress is measured by population growth and research completed.

Our proposed game differs from Civilization in several major aspects:

- *Shorter:* “Dawn of Civilization” only covers the time period spanning the Agricultural Revolution, whereas Civilization begins there and extends into the future. To help make the game shorter, “Dawn of Civilization” follows a fairly linear progression and has specific mini-goals to lead the user to the completion of her main objective.
- *More educational:* Civilization is historically accurate but does not aim to educate players about a particular period of history. Since “Dawn of Civilization” is meant to be educational, it includes messages that provide information about important points of the Agricultural Revolution.
- *Non-violent:* “Dawn of Civilization,” unlike Civilization, has no war with computer-generated “rival civilizations” since conquest has little to do with the learning objectives.

Key Requirements

- *Must be free:* This is an essential requirement; if our clients can't afford our solution, then they can't use it, making our product worthless. Therefore we need to use a free game engine so as not to incur additional costs to them.



- *Run on 1 year old dual core processor PCs:* The game must be playable on the computers our clients have, since they will not be likely to buy new computers in order to play it. Their computers are sufficiently capable that they should be able to handle the high quality 2D graphics we intend to create.
- *Make the game fun:* The game needs to be fun in order for it to be an effective teaching tool. Our clients mentioned that ideally the game should be about “50% fun and 50% educational.”
- *Game state can be saved:* Since the students will not be playing our game in a single session, it is vital that they be able to save their progress and resume from this save file at a later time.
- *No violence:* Our clients specifically requested that, unlike the Civilization series, our game should have no military aspect. The player will deal with the threat of wild beasts and barbarians, but there will be no direct references to violence.
- *Show the benefits of an agrarian lifestyle:* The advantages of an agrarian society over a hunter-gatherer society must be made apparent via the new options that open up for the player when she discovers farming.
- *Present challenges for the player to overcome:* In order to show the benefits of agriculture, the game will present unpredictable challenges that will encourage the player to settle down and farm instead of wander around and hunt.
- *Allow the user to continue to be a hunter-gather if she so chooses:* The game follows a fairly linear progression through the five stages; however, our clients specifically requested that the player should be able to abandon her civilization and revert to a nomadic lifestyle if she wishes to.

Game Design Rationale

We chose to make a game similar to the Civilization series for two reasons: it's a fun and popular series and we feel like it will be the best way to present students with relevant information about the Agricultural Revolution. The game is mainly designed for our audience of 11-year-old students to have fun while they learn. We want the students to get excited about playing our game because, whether they know it or not, this corresponds to getting excited about learning. However, the Civilization game itself is most likely too complex for 11-year-olds who are seeing it for the first time. For this reason, our game design contains only a subset of the features from Civilization and plenty of help messages to guide the user through playing the game. We believe that this will make the game simple enough for all the students to enjoy while still keeping the underlying principle that makes Civilization so fun to play.

In regards to the pedagogical reason, there are two main concepts our clients wanted us to teach in our game. The primary goal is to show the benefits of a society based on agriculture as compared to a hunter-gatherer society. In our intended game design, the player in the hunter-gatherer stage is limited in several critical ways:

- *Forced nomadism:* The user must move from location to location on a regular basis as the



resources in that area run out.

- *No research capability:* The user must spend all of her time gathering food. Therefore, she is unable to research more efficient ways of doing tasks or new skills.
- *Greater environmental risk:* Since the user is limited by how much food her people can carry, any random effects created by the world that damage this limited food supply are a lot more dangerous.

If the player chooses to develop an agricultural society, our second goal is to illustrate how the development of agriculture led to other developments.

- *Development of settlements:* The player can found cities, where her tribe can permanently live.
- *Stable food supply:* The player's tribe will be less likely to starve from inadequate food supply.
- *Surplus:* It will not be necessary for the entire population to gather food, so the player will notice idle citizens and can direct them to other work.
- *Growth:* The player will be able to sustain a greater population and control a larger area of the world.
- *Division of labor:* Segments of the population can be trained in specific occupations or crafts, resulting in cultural developments such as art and architecture.

Proposed Project Overview

Initially we considered three options for the game engine: modify Civilization 4, modify FreeCiv, and build our own game engine. After talking to our clients, we discovered that the game had to be completely free, which eliminated Civilization 4 since it has a proprietary license, and that they wanted us to allow the students to live a hunter-gatherer lifestyle at any point during the game, which neither Civilization 4 nor FreeCiv support, eliminating both of them. These added requirements led us to decide that our best course of action is to build our own game engine.

Unfortunately, building our own engine from scratch is most likely an unrealistic goal given the resources and time available. Therefore, we have decided that the best approach is one that we did not consider immediately: we will use a much more basic game engine called PyGame. PyGame offers many of the benefits of modifying Civilization 4, and additionally carries fewer drawbacks:

- *Graphics and sound libraries:* PyGame has built-in facilities for displaying images and playing sound effects. Either using a free pre-existing tileset or creating our own graphics would be equally feasible.
- *Free:* PyGame is open source and has no monetary cost.
- *Basic:* PyGame does not provide so much structure that it limits us to a certain game play style. We can implement whatever features we want without having to trim off unnecessary features or build our own within another system.
- *Simple:* The major disadvantage of our two initial approaches was the amount of time we would have to spend to understand thousands of lines of questionably documented code in a language



some of us might not be familiar with.

- *Widely used:* There are many games already written using PyGame that publish their source code online, from which we may be able to borrow code for common functionality.
- *Well-known language:* As its name suggests, PyGame is written in Python, which most of our class used in our introductory computer science course here at Mudd. Furthermore, it's a high level language with a large set of libraries, so there is a lot of pre-existing code that we can rely on.

Proposed Development Process

The twenty-student staff who will develop the project will be divided into several teams who work in parallel:

- *Management team:* 1-2 students will be in charge of overseeing the progress of all other teams and facilitating communication between them.
- *Graphics team:* A team of 2-3 students will design tilesets, sprites, and any other images that are necessary for the final game. The graphics team will work closely with the interface team since they will both influence the eventual appearance of the product.
- *Logic team:* A team of size 4-5 will formalize the rules of the game, codifying what is and is not allowed in every situation. They will also chart what effects each action has in the short and long term.
- *Storyline team:* This team of 3-4 students will work on the historical and plot aspects of the game, including setting mini-goals, outlining all pitfalls and failures, and writing the historical background text. They will work with the logic team to decide which technologies can be researched and what sort of buildings can be constructed.
- *Interface team:* A 4-5 person interface team will decide the specifics of the screen layout, what actions will be associated with clicks in on various components, and how basic actions such as construction and research will be initiated.
- *Programming team:* The remaining students will not be assigned to any team, but will be in charge of programming portions of the project that are not covered by any team and integrating the other teams' modules into the code base.

Initially, all teams will work together to specify the product fully, including the basic guidelines for the interface and storyline, and produce a detailed management plan for the project. Once the specifications are complete, most of the work will be done in teams as described above. Each team should have a leader who keeps track of progress within the team, delegates tasks to the other members, and organizes communication with other teams. The entire staff should meet weekly for an update on everyone's progress and to discuss plans for the coming week. If any team entirely finishes its task, the members will be reassigned to assist other teams that may be falling behind. After all teams' work is complete, testing will be carried out by the entire group so that all portions of the product are exposed to all perspectives.



Risk and Feasibility Analysis

- *Balance of education and entertainment:* We have two main design objectives that can come into conflict with each other: to teach students, and to entertain children. If we do not strike an adequate balance between providing historical information and making the gameplay enjoyable, either the students or their teachers may find our product unfit for their needs.
- *Balance of challenge and intuitive play:* “Dawn of Civilization” is intended to be easy to learn, but hard to master. This is a tall order that is easy to miss on either side. In order to make the game better suited to each individual player, the player's success can be measured based upon her achievements in each stage and the size and well-being of her civilization. We intend to make the game easy to learn by providing a tutorial for new players and offering hints throughout the game if the player appears to be struggling. For a player who is especially successful, environmental challenges such as an attack by wild animals, a climate change, or an epidemic will be made more likely. Extensive testing will be required to determine the correct degree of challenge to keep players interested, but not frustrated.
- *Time restrictions:* Our final product must be created by twenty students in just under two months. This is a smaller staff and a significantly shorter time scale than is ideal in game development. It is possible that not all features may be implemented in the time available. Since we are limited by the length of a single semester, there is very little we can do to avoid this restriction, but our team is prepared to work hard to deliver our product on time and complete. Our prototype, which is a very limited segment of the game, was programmed by one person in approximately ten hours, so we expect that the time available will be sufficient to create a final product. Moreover, Python has been shown to have one of the lowest development time requirements of all modern programming languages (“An Empirical Comparison of Seven Programming Languages,” Lutz Prechelt, IEEE, October 2000).
- *Team interactions:* Few if any of us have worked on a team the size of our entire class, and none of us have led a project of this style and magnitude. We will have to put in extra effort to work well with each other and manage our progress adequately. We consider ourselves prepared to take on this inevitable risk, and in the process we will become better members for the next team we are each a part of.
- *Learning the PyGame engine:* One of our foremost arguments for building our own engine was that we would not have to spend time learning to work with pre-existing code. We will have to dedicate some time to learning PyGame; however, it is not a complete architecture that we must build off of, but rather a tool to build our game logic upon. PyGame and its associated utilities are well documented, so they will not require extensive study. If the PyGame library does not offer a particular functionality that we need, then we will be able to extend PyGame easily with our own code.



Our Team

Beky “Pretty Pretty Princess” Kotcon:

Team Roles:

- Peacekeeper
- Taskmaster
- Lead Ticket-Closer

Experience:

- Once shot a man in Reno, then poked him in the eye

Skyler “Whonderful Whizzard of w00t” Kaufman:

Team Roles:

- Glorious Leader
- Overmind
- Court Jester

Experience:

- Once shot a man in Reno, and looked extremely fly

Emily “NinjaNomicon” Fujimoto:

Team Roles:

- Scrivener
- Authoritator
- Head of the Writing Stuff Down Team

Experience:

- Once shot a man in Reno, then went to play Far Cry

Jason “Y-Chromosome” Garret-Glaser:

Team Roles:

- Chief Software Engineer
- Main Source of Annoyance
- Token Male

Experience:

- Once shot a man in Reno, then made a sandwich on rye