Denial of Service

Game Design Document

{ Many years into the future, most civilized governments have collapsed, leaving many smaller factions and city-states. War has evolved along with technology; moving off the physical battlefield and into the virtual realm of the 'Net. Through our programming prowess, along with DirectX, and new, stylized art we have realized that future as an interactive game. Here is **Denial of Service**… **};**

Colab Co. is {

Benjamin Ward Childress;

// Graphics, Sound, and System Programming

Ryan Clause;

// Game and Stat Balancing, Play Testing

Tim Gregorio;

// Building Art, Unit Art, Character Design

Niall Mackinnon;

// Concept Art, Unit Art, Building and Unit Programming

};

Artist Statement {

The growth of corporate regimes and the ever-progressing technological "revolution" can lead to only one possible future. Wars will continue to be fought. Technology will continue to march on. The two will combine to change the future landscape of combat. Warfare will no longer command technology; technology will command warfare. It is inevitable.

};

WHY Factor {

Fans of strategy games will enjoy **Denial of Service**'s unique expansion of a time-tested genre while enjoying our vision of the future of warfare. **Denial of Service** evolves from the contributions of such predecessors as:

// Alpha Centauri
// Civilization
// Command & Conquer
// Heroes of Might and Magic
// and Warcraft

while still maintaining a unique style, presentation, and view.

};

Target Audience/Distinctive Factors {

Denial of Service will attract those fans that have enjoyed such classic gameplay elements as:

// **Turn Based Play** - One player takes a turn and issues commands followed by the next player who takes his turn and issues commands. Gameplay continues as such.
// **Strategy -** Your decision making and unit management abilities will be the major factors in achieving victory.
// **Competitive Game Play -** Multiplayer functionality allowing you to compete against a computer controlled AI or another human player.

};

Introduction & Story {

Many years into the future, most civilized governments have collapsed, leaving many smaller factions and city-states. The large corporations of the past have also emerged as powerful nations, controlling the lives of millions.

Technology has grown to encompass most aspects of daily life. The internet now connects essentially all the computers in the world into one large network.

War has evolved along with technology; moving off the physical battlefield and into the virtual realm of the 'Net. In this time of unrest, nations employ virtual weapons in order to take down vital systems of their enemies, causing real life casualties and forcing their opponents to surrender.

You are the leader of some faction wanting to take over the world. You have at your disposal the computer mainframes and the computing power of your country. Your goal is to use these resources as well as the recruitment of hackers and other mercenaries to gain control of your enemies' computer systems, forcing them to submit to you.

};

Projected Socio/Cultural Impact {

The desired impact of **Denial of Service** is to raise awareness of the potential danger posed by internet based attacks, the possible future of warfare, and the effects that this new form of warfare could have on the global population.

};

Delivery System {

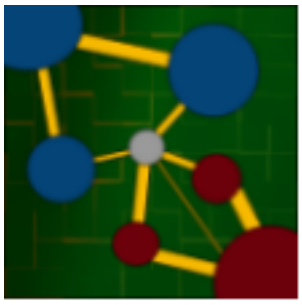
Denial of Service is designed and programmed for the Microsoft Windows OS using DirectX. The world is presented from a bird's-eye-view allowing for a large field of vision necessary for managing a virtual army. A sidebar and message pop-ups will serve as the major means for player/game interaction. The player will control his army through mouse and keyboard inputs while the game will control the player through sound and graphics output.

/* See [Figure 01] in Appendix for concept images. */

};

World Layout {

The internet will be represented as many circular networks connected by wires. Players will start in different networks and will have to navigate the grid of wires and subnets in order to conquer their enemy. The levels consist of a multitude of networks in varying layouts containing friendly, enemy, and neutral controlled mainframes, units, and systems.



};

Visualization, Characters, Flow Charts {

The graphical style of **Denial of Service** will consist of an eclectic mix of anime, post-cyberpunk, and neo-technological design to create a unique presentation that the user will enjoy. A series of screenshots of the **Denial of Service** prototype are included at the end of the document.

/* See accompanying document for Unit/Building Flow Chart */

};

Music/Sound Design {

The musical score of **Denial of Service** will consist of pieces borrowed from cutting edge electronica. The sound effects will consist of humorous and inventive real-world sounds that will uniquely identify specific game elements.

};

Gameplay Elements {

Environment:

At the beginning of **Denial of Service**, you start off with one mainframe in a small circular area, which represents your country's intranet, or internal network. This is where you build most of your other mainframes which can be used to create units, research software, and collect resources. As your strength increases, you will be able to expand the size of your intranet, allowing you to build more mainframes and various other systems.

The key to victory involves capturing neutral networks that exist throughout the Net. Upon capturing the mainframe in the center of the network, you will need to spend time and resources in order to subjugate the real life owners of that network. (Not all networks have mainframes, some are just mail servers or message boards, so it is not possible to gain control of the people operating those systems.) After you control the network and the people who maintain it, you will be able to build new systems and expand that particular intranet. Without controlling the people of a network, you will only be limited to draining resources from that network. Be warned; a network that is not fully under your control could be lost at any time.

Connecting each network are thick lines that permit travel between networks. Each line has, associated with it, a bandwidth and latency. The bandwidth limits how many units can be on a line at one time; whereas, the latency determines how quickly a unit can move along the line.

Resources:

// **Computing Power -** At the beginning of each turn, you are given a certain amount of computing power based on the number of mainframes that are under your control. Essentially, computing power is used to issue commands. Computing power is needed to create new units, to research new software, to issue orders to units, and to execute software skills on a particular system (ie. tell one computer to DOS attack another system).

// **Money -** At the beginning of a game, you are given a certain amount of starting capital which you must use to build up your initial forces. As the game goes on, you will capture other networks which contain banking systems. While you have control over a bank's computer systems, you'll be able to drain money from them. This money is collected at the beginning of each turn. Money is used in the recruitment of hackers and other mercenaries. It is also used to add new systems to your networks, and upgrade your machines.

// **Power -** Power is much like the food of this game. It determines how many units and systems you can have on the map at a given time. Power is collected by controlling the computer systems of power plants. With a power plant's computer system under your control, you can freely direct the flow of current towards your networks.

Buildings:

There are several types of buildings, referred to as systems, that exist throughout the Net. The most important being the Mainframe. This system essentially anchors down a network, allowing it to function. Without this system networks would cease to functions, and the other systems connected to the network would become useless. There are some specialized networks without mainframes, such as mail server networks and message board networks.

The mainframe also provides much of your computing power, allowing you to issue many commands on a single turn. Additionally, mainframes allow for the training of basic units and skills.

Another system is the power plant system which is used to direct energy towards your networks. Message board systems are used as recruitment bases for hackers and other mercenaries. Subsystems, such as compilers, can be attacked to your mainframe to allow for the development of more advanced software. Finally, another example of a useful system is the Norton License, which is used to create antivirus units needed to protect your networks from intruders.

Various defense systems exist to protect your networks from outside attack. One example of this is the firewall, which can be constructed only within

a network that you control. The firewall's purpose is to keep anything unfriendly out. It does not detect an enemy's undetectable units, that is the job of the antivirus software. Although firewalls are strong, they are not indestructible, but they are among the best defense from an outside invasion of your computer networks.

Capturing a System:

In order to capture a system, you will need to use either viruses or other hacker type units. A virus is useful because they are undetectable units that can easily be snuck into an enemy's network. The down side is that viruses are slow acting. Another option is to use hackers to forcefully take over another network. The speed at which the hackers gain control of a system depends on the number of hackers allocated to the task. Be careful because enemy units can easily take down an attacking hacker, making him unavailable for future use.

Once a network's mainframe is taken over, the entire network falls under your control; it then becomes a case of gaining control of the various systems that exist within the network, but without the mainframe, these systems don't put up much of a fight.

Conditions for Victory:

Victory is achieve in one of two ways: the first being that you are able to capture all of your enemy's networks, and the second being that the enemy voluntarily surrenders to you.

You are defeated if all of your networks are captured by the enemy.

};

Program Structure {

Denial of Service is written as a Microsoft Windows executable that will use their DirectX 8.0 SDK to render our two-dimensional game board, characters, etc. The game itself is composed of a number of states. The player's turn is represented as a state. Followed by the state where his actions and commands are resolved. Then the computer (or other player) has a chance to respond. The turns are bandied about in this manner until a winner has been declared. From a technical perspective, each turn consists of a number of primary, high-level engine-based commands. For example, no matter whose turn it is the graphics are drawn first. Then the inputs are polled for data. The data is then manipulated based on the state; if it is the player's turn then clicking on a unit selects him/her for orders, otherwise the action is ignored. Next, immediate requests from the user or AI are handled. Does the screen need to be scrolled? Is it time to determine attack damage? Finally, all global game variables that require constant monitoring are updated or acted upon. E.g., the turn clock is decremented if a second has passed, the internal timer for popup dialog boxes is increased, etc. These are done partially through internal game engine code

(such as clock data storage and manipulation) or through external object-oriented processes (such as hacker information passing and graphics rendering).

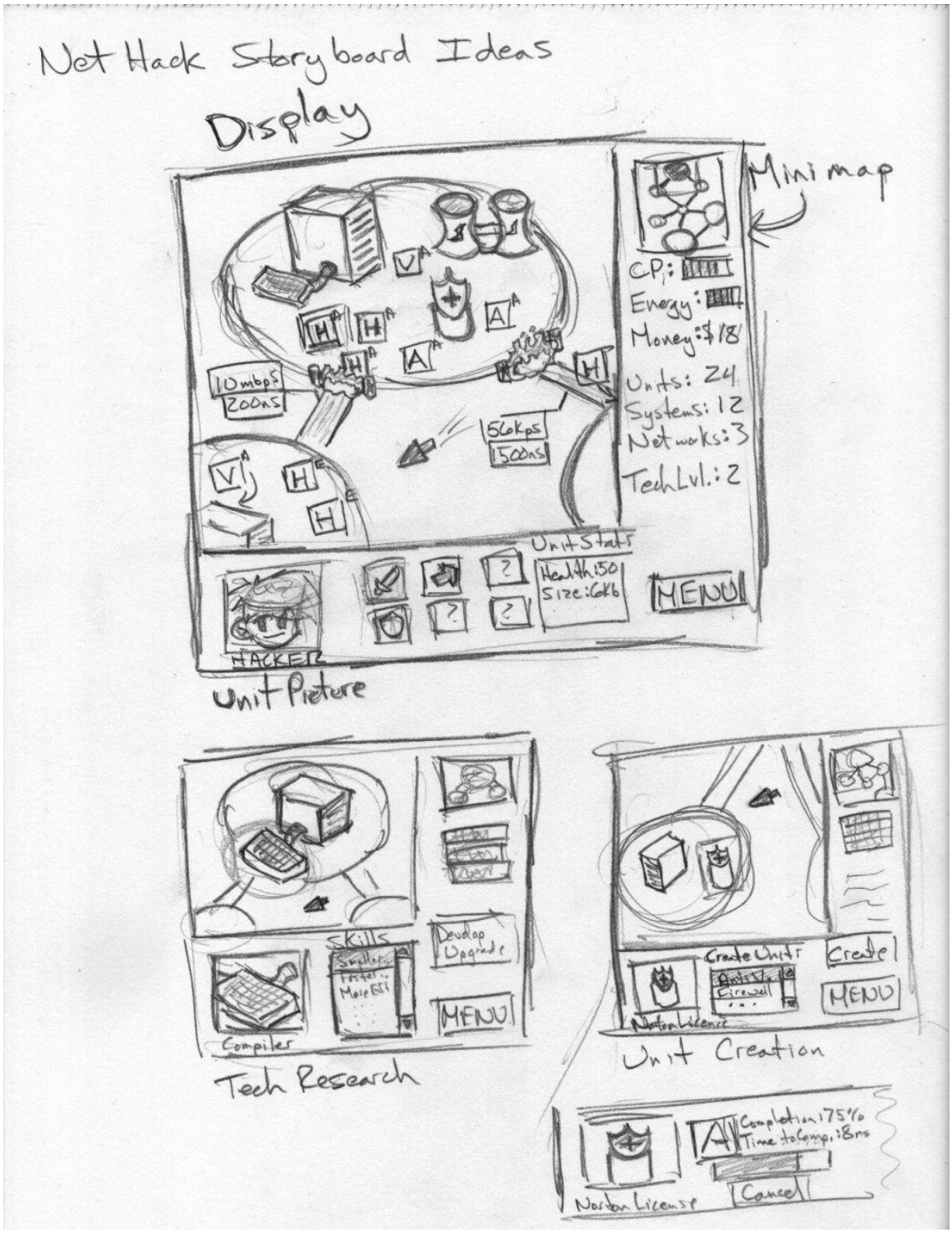
};

References {

//	Alpha Centauri	- http://www.firaxis.com/smac/
//	Civilizations Series	- http://www.civ3.com/
//	Command & Conquer Series	- http://www.eagames.com/official/cc/
//	WarCraft Series	- http://www.blizzard.com/war3/

};

Appendix {



/* [Figure 01] Rough Concepts for the User Interface */

};

Character Concept Sketches {

/* Rough Concepts for Anti Virus and Hacker */

};

Game Screen Shots {

};