Bright-Crayon, LLC
3126 West Cary Street #407 Richmond, Virginia 23221 U.S.A.

Partners: George Kelly Flanagin Dr. Pamela Kiecker
+1 804 677 3500 Susan R. Jacobs John N Pastore, Jr
www.bright-crayon.com Greg Provo

Empire Detailed Design & Implementation Spec

Compiled by George Kelly Flanagin

Under Construction

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Serial Number	Date of last Revision	Number of Pages
46	2004/12/09 1657	29

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

1 Overview

1.1 Note to the reader

Because this document contains something of interest for many different readers, we have supplied explanations of many terms that might be confusing. For example: readers looking at the business model may already be familiar with the terms of marketing, but technically inclined readers are less so. Because we want to reduce document clutter and page flipping, the explanations of specific terms in specific disciplines have been confined to footnotes.

1.2 What is this document?

This document covers the implementation details of the Empire game engine. Each section of this document covers a module, and for each module (where appropriate) there is a discussion of these things:

[1] How the module interacts with other modules in the game.

[2] The public interface of the module, and how it is used.

[3] Documentation of interactions that may not be obvious when one is looking at only one source file.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

2 Architectural Basics

2.1 Player/Nation

Each human player is associated with a “nation” in Empire. The nation is known to the other players by a name of the player’s choosing, and the game is setup to completely conceal the name of the person associated with a nation.

Players log in to play the game using a password authentication scheme, and a player may only be logged in once to the game, thereby preventing three or four people from simultaneously engaging in combat on behalf of a single nation.

2.2 World

Empire’s world is a bit unusual. In an ordinary sense, the world contains a collection of islands and the open water between them, known as the ocean. The islands are randomly sized, and randomly placed in the world at the time the game is set up.

The coordinate system may be thought of as a grid of squares lying in the first quadrant of the plane we know from Geometry 101. The world is square, with its top and bottom edges joined, as well as the left and right edges. This apparent attempt to produce a sperical world actually results in a toroidal world, but with few playing complications and several benefits.

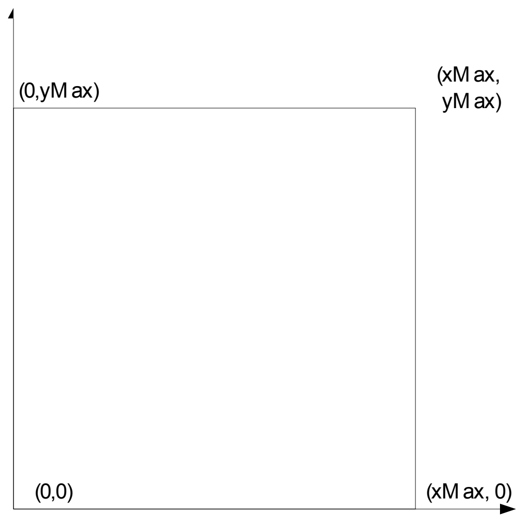

Figure: The coordinate system of Empire

One of the benefits of this system is that no player’s starting point in the game is near or far from the “edge of the world,” and no one is cramped into “polar regions” that would be found in a truly spherical world.

The Empire game engine controls each player’s view of the world. A player is informed about the overall size of the world, but he can only see the neighbourhood in which he has a presence. The situation is extreme at the beginning of the game where a player can only see the island on which he starts.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

2.3 Island

The game can be configured to contain a number of islands. The islands consist of a contiguous group of sectors that lie within some rectangle. This number is generally chosen to be about twice the number of expected players. One of the benefits of this arrangement is that each player gets to start out alone on an island, and has the chance to develop the nation's strength for a while before other players are encountered and combat may begin.

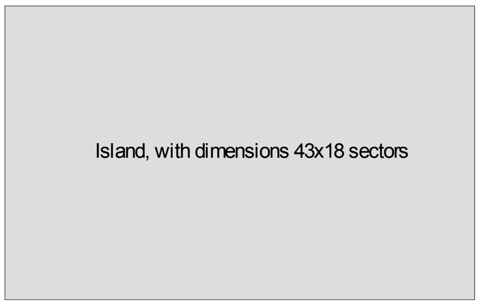

Figure: Example island region

The island shown in the example is made up of a contiguous block of sectors that lie within a rectangle 43x18, measured in terms of sectors. Islands never overlap, and the game has a parameter that controls the minimum distance between islands. The islands are randomly placed (within the constraints of size and separation) within the world.

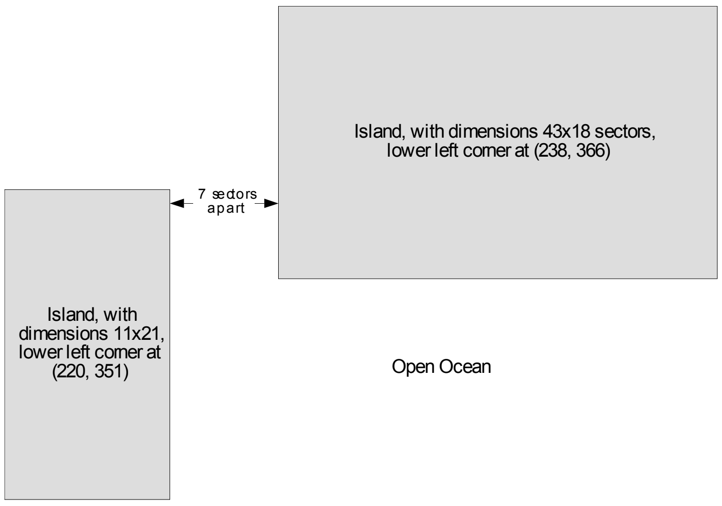
Island, with dimensions 43x18 sectors, lower left corner at (238, 366)
7 sectors apart
Island, with dimensions 11x21, lower left corner at (220, 351)
Open Ocean

Figure: Two islands, somewhere in the great ocean

When the game is configured by the game engine, part of the process also creates each island's topography. This is a several step process whose explanation is important to the reader's understanding of certain other parts of play. The following figure shows an example.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

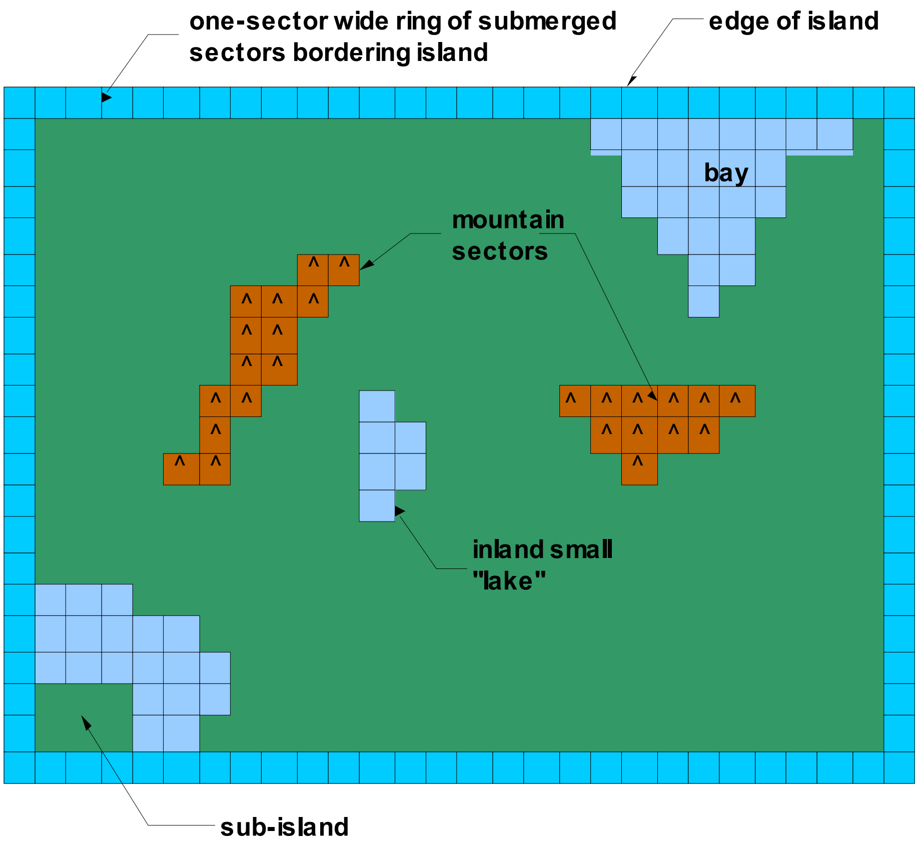

Figure: Drawing of island topographic features.

The outer ring of sectors is always submerged to create a shallow coastal (non-ocean) area around the island. Then, a fractal algorithm creates a mixture of flat land, mountain ranges, and lakes in the remainder of the island. The smoothing nature of algorithm means that most islands wind up with a somewhat irregular coastline (more so than is shown in the drawing), and it is more likely to find mountains toward the interior, particularly if the island is large.

The island on which a player starts is called the “home island,” that player who starts there has the right to name it until the player exits the island.

2.4 Sector

As we have seen, Empire’s world is made up of “sectors,” which in the case of the sectors that make up islands may be defined as the smallest unit of land area that is separately accessible to the command language of the game. Sectors are owned by one player at a time, and they have contents such as population, weaponry, factories, etc. Sectors are also the unit of sustaining damage, so when a sector is attacked the sector is damaged when the contents are destroyed.

There are quite a few types of sectors:

Capitol	Each player has at most one sector designated as the capitol sector on each island that he occupies.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Factories	These sectors are designated to produce various wares of the game, all of which are connected with warmongering. Each factory produces only one type of item. In this version of the game, they are limited to artillery pieces ("guns"), explosives ("shells"), airplanes, and ships.
Mines	Mines produce the universal substance, ore, from which the factories produce items of use. Loaded onto ships, ore becomes the fuel that lets them move about.
Special Sectors	This version of the game supports sectors designated as radar installations, forts, urban centers, airports, and canals.

Table of Sector types in Empire

2.5 Ships

Ships are necessary to move people and materiel between islands. Although the game does have aircraft, they are simply instruments of combat, and do not really carry anything resembling cargo.

Ships come in a variety of types, which correspond to "real world" ships both in name and function. A nation might have several hundred ships in a big game, and these ships may be organised into fleets for the player's convenience.

Type of Ship	**Purpose/Function in the Game.**
PT Boat	Rapidly getting to another island to "plant the flag."
Barge	Transporting cargo around a single island.
Ferry	Transporting civilians around a single island.
Mine Sweeper	A dedicated ship for removing mines from coastal water.
Destroyer	Depth charging subs, and laying mines.
Freighter	Transporting materiel across the open ocean.
Liner	Transporting civilians to islands to settle it.
Submarine	Stealthy exploration, and attack against enemy fleets.
Tanker	Transporting fuel, usually with a fleet, to keep the fleet fuelled.
Transport	Transporting ground combat troops across the ocean.
Cruiser	Defense of large fleets, shelling islands from offshore.
Battleship	Destructive offshore assault of islands.
Carrier	Transporting aircraft at sea.

Table of Ships used in Empire

When military are placed on ships, they become its crew. When explosives are loaded, they become the ship's shells. When artillery pieces are loaded, they become the ship's guns. When ore is loaded onto ships, the ore becomes fuel.

2.6 Aircraft

The aircraft model in the game is simple. There is only one type of aircraft, and it is the fighter/bomber jet. Aircraft are always located either in sectors designated as airports, or aboard aircraft carriers. Aircraft can be flown from one type of airbase to any other without use of movement time or fuel. To drop a bomb, they must have one "explosive" put

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

aboard them. To fly the plane, the airport or ship must have military personnel to fly the plane.

2.7 Passage of Time

When a game is set up, a rate of the passage of game time is chosen. The rate of time passage in the game is arbitrary, and the unit is referred to as a "clock tick." The game does not support day and night, nor seasons or weather. From the standpoint of game setup, experience has shown that 100 clock ticks per calendar week is a fairly slow game, and 100 per 24 hours is almost too fast to allow for fully competitive play. A frequently chosen length of a game is 2000 clock ticks.

Time is accrued by ships, and by the capitol sectors on each island. Some explanation of each concept is required, and they are gone over in detail later in this document.

2.7.1 Ship Time

Ships have a maximum speed that is measured in sectors per ship-day. A ship day is a real day multiplied by some number that the person who set up the game decides. If the factor is 1.0, then a ship day is 24 hours, and a ship with a maximum speed of "15" can travel 40 sectors in 24 hours.

Ships that are at rest accrue up to one ship-day worth of movement that can be used for interactive manoeuvring when a player is actively logged on. The game engine keeps ships that are on course moving even when a player is not currently connected to the game.

Ships burn a certain amount of fuel even when at rest, and this is a value that is determined by the length of the ship-day, and the type of ship.

2.7.2 Land Time

Land time is accumulated on a per-island basis. To accumulate any land movement time, a player must have a sector designated as a Capitol on the island. With each clock tick, the Capitol sector accumulates one unit of "movement time." The player can use these units of movement time to reposition materiel on that island, attack enemies, etc.

When a unit of movement time is expended, it becomes "update time." The purpose of update time is to increase population, cause factories to produce wares, and to cause mines to produce ore.

Land time is completely managed by the player during interactive play. However, the game does provide automatic defences from attack. For example, if a sector is bombed and has the ability to throw flak at the incoming planes, this response will take place whether or not the player is actively connected to the game.

2.8 Efficiency

Efficiency is a measure of how well the materiel of Empire works. On the land, efficiency is associated with the individual sectors of each island; each ship has its own operating efficiency.

Land efficiency actually runs from 1% to 100% rather than from 0% to 100% as one might expect. Unimproved land is 1% efficient. After the player designates what type of sector

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

the new land is to become, its efficiency rises one percent for each clock tick. When a sector becomes 99% efficient, the factory begins to produce its wares. If a sector is attacked, it is quite likely that the sector's efficiency will fall, and the production drops off.

The operating efficiency of ships affects everything about their operation. When first built, a ship operates at 100% efficiency. If a ship is damaged during a conflict, its efficiency drops, and with that drop in efficiency comes reduced maximum speed, and a reduced capability to withstand further attacks.

2.9 Population Growth

Update time causes civilian population to grow. The player can conscript troops on sector, island, or nationwide basis. Conscription does not alter the overall population, just the balance of civilian to military population. Once conscripted, that segment of the population is unaffected by updates.

2.10 Order of play

Empire does not support the notion of turns for players. Any player with movement time can use it at any time without regard to what other players may be doing.

2.11 News / Mail / Chat

The game engine keeps a partial log of player events. These events are stored so that they can be sorted by the players, islands, and ships involved in the incident of interest. From this data, the game engine creates a summary which amounts to a "daily newspaper" that is distributed to all the players.

As in real combat, players may want to communicate for purposes of alliance or taunting. The game provides the ability to send another player a message in an anonymous way, either through POP or instant message chat. This is in keeping with the idea that players may want to remain unidentified.

2.12 Client Server Parts

Since Empire is a multi-player game, it consists of client and server parts. The client parts support the presentation of a player's status and the choices of what to "do," and the server parts support the processing of players "moves," and keeping the game synchronised. Although the players interact via the internet, it is not a "web based" game in the usual meaning of the word.

The following drawing is a high level sketch of the major architectural components.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

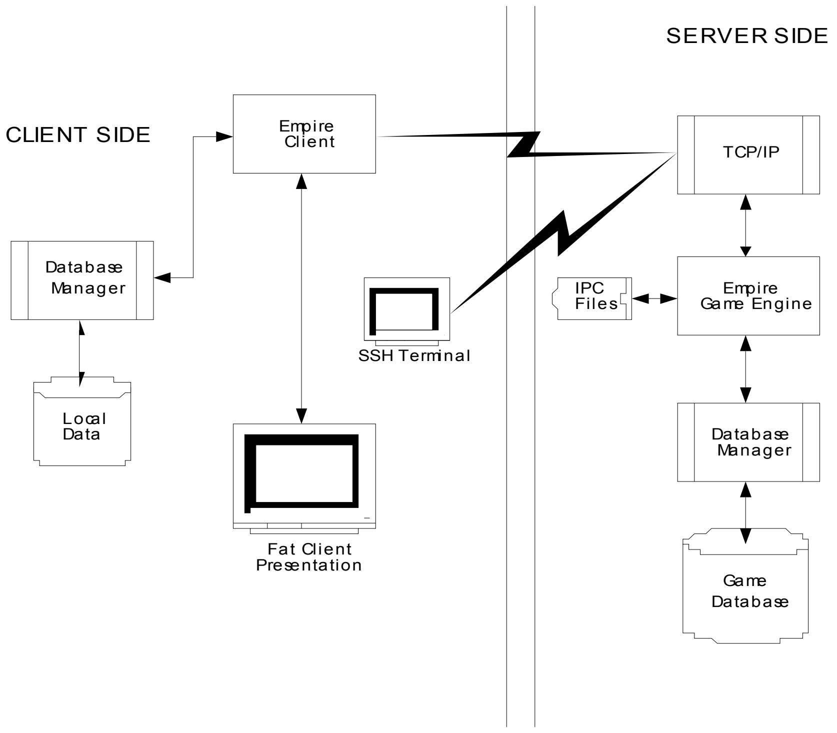

Figure: Highest level view of the architectural components

2.13 Empire Game Engine (Server Process)

Empire's game engine is given XML documents via the connection maintained between the web server and the player. It processes them in the order in which they are given so that there is never more than one player's command executing against the database at once. The outcome of combat or the change in positions of ships and materiel is reflected in changes to the database, and the results are repackaged in an XML document and set back to the originator.

There may also be background tasks executing while players are engaged. These include updating the game's clock, moving ships that are on course, and executing stored commands that are to be executed when the game's clock reaches a certain point.

The following major section dealing with "scenarios for use" deals in detail with the functions of the game engine.

It is important to understand that the Empire game engine is the only process connected to the database. Therefore, database level locking strategies are not employed. Rather, the queuing process takes place inside the engine.

2.14 Empire Game Database

Empire's database stores all the information about the islands, their sectors, and all the ships. It also contains the "static data" for the game: parameters that are somewhat arbitrary (ex: maximum speed of an aircraft carrier), but control the feel of the game.

The only connection to the database is from the game engine itself – individual players do not connect directly to the database.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

2.15 Empire Client

Empire may be played in text mode, via terminal emulation that supports SSH protocol. In this case the client is “thin,” and actually resides and executes on the server. This thin client allows for play in situations where the data exchange rate needs to be lowered, and has assisted in the development and writing of the game.

Empire’s graphic client is “fat.” There are several reasons for this:

- The richness of the information presented to the players.
- The variations in how different browsers render pages.
- The need to allow players to plot strategy and test out scenarios while not connected to the game.

2.16 Presentation Layer

Ordinarily, one might leave this item out of a discussion of high level architecture. However, in the case of Empire the topic is germane. From an architectural standpoint, the presentation layer is something that may need to undergo a bit of evolution, particularly as new features are added to the game. The methods for presenting complex information in a small space (such as a CRT) are always being researched, and with the research come new opinions about what system works best.

As a result, we can see that the user interface may change quite a bit more often than the underlying game, and we might even want to provide alternate displays akin to the “skin” concept that is being offered in some software.

2.17 Architectural layers

The following drawing represents the “layers” in the separation between the concepts of the ideas represented in the game, and the translation of these ideas into standard computer data structures.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

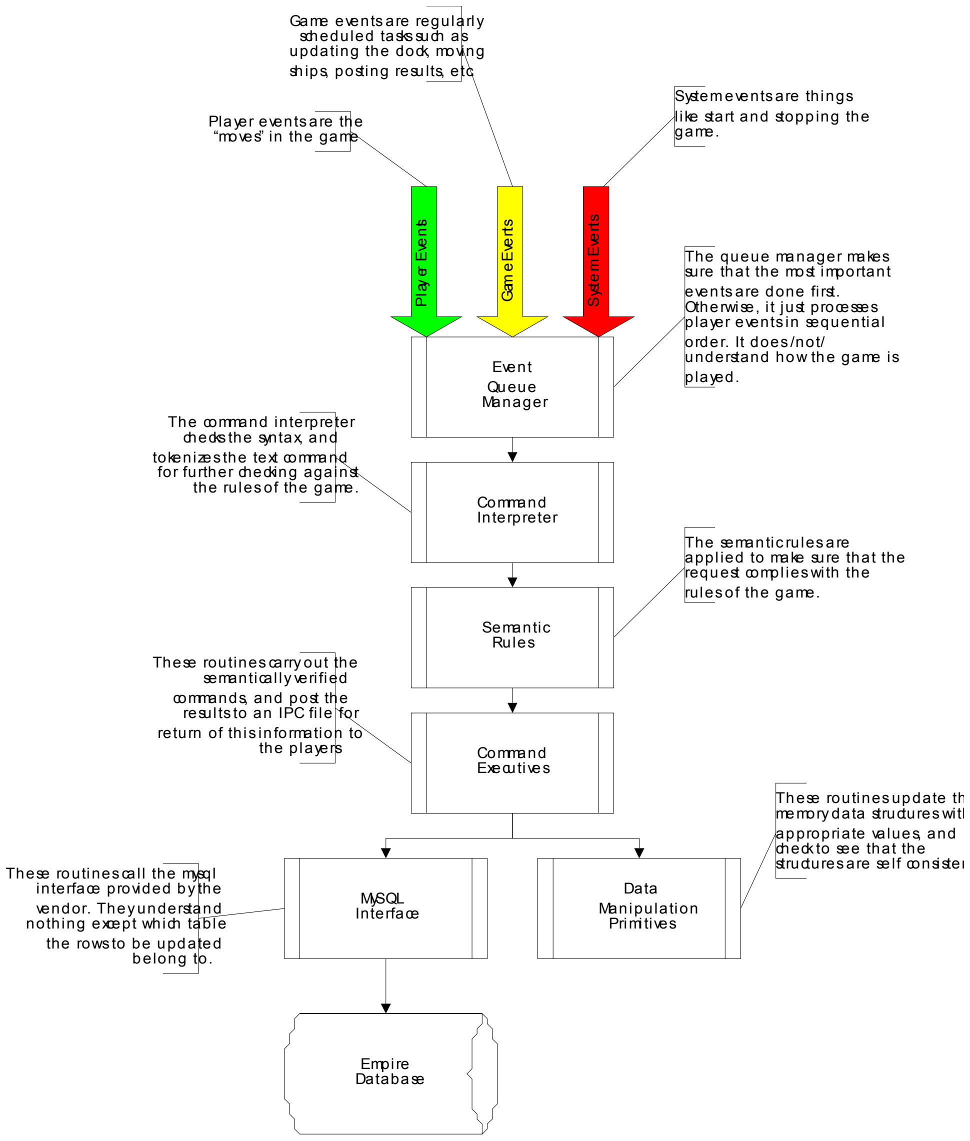
Game events are regularly scheduled tasks such as updating the clock, moving ships, posting results, etc
Player events are the "moves" in the game
System events are things like start and stopping the game.
Player Events
Game Events
System Events
The queue manager makes sure that the most important events are done first. Otherwise, it just processes player events in sequential order. It does/not/ understand how the game is played.
Event Queue Manager
The command interpreter checks the syntax, and tokenizes the text command for further checking against the rules of the game.
Command Interpreter
The semantic rules are applied to make sure that the request complies with the rules of the game.
Semantic Rules
These routines carry out the semantically verified commands, and post the results to an IPC file for return of this information to the players
Command Executives
These routines update th memory data structures witl appropriate values, and check to see that the structures are self consistei
These routines call the mysql interface provided by the vendor. They understand nothing except which table the rows to be updated belong to.
MySQL Interface
Data Manipulation Primitives
Empire Database

Figure: Architectural layers

The important thing to understand about the drawing is the strict partitioning of knowledge about how to play the game, as well as the mere separation of the code into modules. For example, the command interpreter is the only routine that understands the syntax of the game's language (ECL), and if that syntax is changed only this module need be modified. Similarly, the semantic rules engine contains the knowledge about what constitutes legal play, and does not need to know how the players express their moves.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

3 Use Scenarios and Explanatory Text

3.1 Game Creation and Configuration

3.1.1 Create a new game.

The game administrator, historically referred to as “The Prophet” in previous versions of the game, establishes the parameters for a new game.

The SQL daemon processes the script (see appendix) that builds the database for all of Empire’s data. If the physical server is to have more than one game running on it, then the administrator may have to edit the “CREATE” statement in the SQL script.

Next, the genesis module (not shown in the architectural diagram) allows the administrator to change the default parameters of the game. The genesis module updates each changed record in the database.

When all the parameters have been chosen, the genesis module populates the game database with a record for each island, and each sector on each island. When this process is finished, the database is ready to support play.

3.2 New Player

3.2.1 New Player Signup.

The Empire web page will contain a signup button for new players. Pressing this button will take the new player to a page where he will fill in the necessary information: Nation name, email address for the game to contact the player, etc.

If this information does not conflict with the information for existing players, then a new player record is created in the player table. The game then mails a generated password to the player at the email address supplied. The game updates the player record with the message digest (hash) corresponding to the player’s password.

3.2.2 Exceptional conditions

If the maximum number of players has already been reached, new players may not be added to the game.

If all islands are occupied, but the maximum number of players has not been reached, the player is given the option of whether he wants to join this game.

3.3 Globally Available Information

Some information is globally available. These facts, available to all players at any time during the game and without qualification as to status include:

- The circumference of the world.
- The maximum number of players in the game.
- The association between nation names and player numbers.
- The current time of the game clock.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

- The rate at which time is passing in the game.
- The ship-day factor.
- A topographic map of any island the player occupies.
- The island position and dimensions of any island that falls within a player's ship or land based radar range.
- The costs and maximum concentrations of materiel.
- A complete inventory of one's own materiel.

3.4 Connecting to Play

3.4.1 Login.

Player supplies his user name and password on.

The game verifies that the password matches the message digest stored in the database. It also checks to see if that player is already logged on.

The Empire client runs as an application/service on the player's machine, and this logon activity establishes a connection between the player's machine and the game.

3.4.2 Interruption of the Network

In cases of a client or server crash, or loss of network connectivity between client and server without a crash, play is interrupted, and a new logon sequence must be begun.

3.4.3 Logout.

When a player wishes to logout, the player can do one of two things. The client can issue a logout command from inside the client. Alternatively, the player can log out from the web page that is presented after login.

3.5 Inception of Play for New Players.

3.5.1 First Connection.

After a player receives a password, play may begin. A few things are different on this first connection since the player has no information in the game's database of islands and sectors.

When a player connects for the first time, the game updates the player's record in the database with his "date of birth" from the standpoint of the game.

The game engine looks through the file of islands, and finds one that is unoccupied. The game then chooses a random sector of flat land, and gives the player the default number of civilians in that sector, designates it as a Capitol, and supplies the sector with the default starting units of movement time.

3.5.2 Exceptional conditions

It is possible the between early sign up and a couple of calendar days later when a player might actually be ready to begin, that all islands are occupied. In that case, the game en-

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

gine will select from the islands with the minimum number of players (quite likely, 1 player) on the island.

3.5.3 Player's initial view of the game

The game engine controls the view of the world. After the game successfully establishes a player's initial presence on some island, the player is sent a description of the island. This information is the same "packet" that one would get regardless of the island that one is on.

In the initial state, the new player gets a sector by sector map of the island showing elevation, the one sector that is the new player's capitol, and an indication of what nations occupy other sectors on the island. A player may not see what type of sector an enemy sector is unless he owns an adjacent sector.

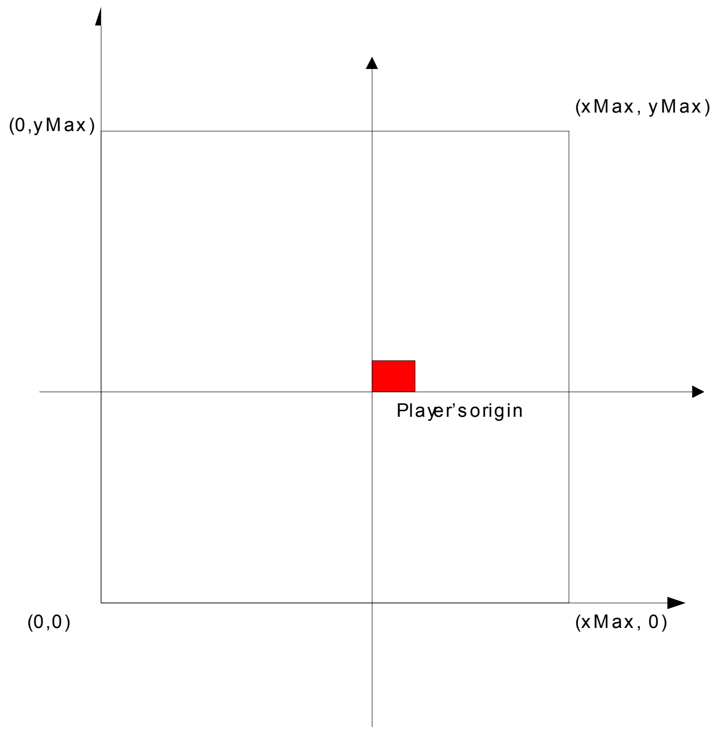

Figure: Initial view of the world.

The initial description of the island tells the player nothing about its location in the world, and in fact, the lower left corner of the player's home island becomes the "centre of the world" from his point of view. The drawing above shows how a player's coordinate system would be significantly different from the absolute coordinate system of the world.

3.6 Inter-player and Game-to-Player Communication

3.6.1 Players contacting other players.

One of the few clues offered to the player is that a new player knows he is "player number 12 of 30," for example.

A player can choose to send a message to any other player by player number. When this request is received by the game engine, the player's registered email address is looked up, and a standard email message is composed to on send.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

3.6.2 Cooperative Play

The notions of alliance, treachery, deceit, and team play are all a recognised part of Empire. Empire allows for players to transfer planes, fixed materiel, people, land sectors, and ships to another players. Arrangements to do so are made via the game's communication channels. The transfer is executed by appropriate actions by the sending and receiving players in the game.

3.6.3 Game contacting players.

The game sends out a newsletter summarising recent game events to the active and deceased players. Reading this newsletter is a valuable way to determine which players are at war with each other, and which players occupy which islands.

Players may wish to note that if they are not engaging in combat, their activities will go unreported in the game's newsletter.

3.7 Passage of Time

3.7.1 At first play.

The parameters of the game will give each new player a number of movement time units to start the game. Typically, this might be 100 units. This gives a new player a considerable amount "to do" at the beginning of the game.

Movement time units accrue to the Capitol sector of the player's home island as time passes, and at the rate that is associated with the particular game.

3.7.2 Movement vs. Update time.

As a player uses his movement time units to shuffle materiel, attack neighbours, etc., these units of time are then shifted to the update side of the ledger of the Capitol sector on a one-for-one basis. For example:

Suppose a new player starts with 100 units of movement time in a game where one tick of game's clock is 15 minutes, or as one would more likely term it, "96 clock ticks per day." During the first hour of play, the player distributes some of his population across sectors near to the Capitol, using 45 units of movement time in doing so.

The player's time accrual in the Capitol sector is now 59 (55 remaining + 4 accrued as time passes) units of movement time available, and 45 units of update time.

The following table shows common game activities and the number of movement time units expended in doing so.

Game activity	Movement Cost
Move into unoccupied adjacent sector	1
Attack an occupied adjacent sector	2
Move troops or materiel into mountain sectors.	5

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Move troops or materiel into or through a partially efficient sector.	1 * (1-efficiency) I.e., 0.5 units for a 50% efficient sector, 0.1 for a 90% efficient sector, and so forth.

Figure: Common game actions and associated movement time expenditures.

3.7.3 A more complete explanation of Update Time

When spent, update time causes the population to grow at the rate prescribed for the game. In most cases the population multiplier is $(1.02)^{\text{update-ticks}}$. The following table shows the resultant population for a sector of 100 civilians after the given units of update time are used.

Update Units	Population after Update
10	121
20	148
30	181
50	269
100	724

Figure: Population change with updates of varying sizes.

The expenditure of update time also causes the efficiency of each occupied sector on the island to increase by 1% if it is not already at 100% efficiency. Sectors that have just been occupied or captured from an enemy are set to 1% efficient, and thus take 100 units of time passing before they can reach maximum efficiency. Sectors that have been damaged by combat might be at any level of efficiency below 100%. These sectors are "repaired" by the expenditure of update time.

Once a factory sector is at 100% efficiency and has 100 or more civilians living there to operate the factory, it begins to produce production time units – one for each unit of update time that is expended. These production time units, or "production points," are used to build whatever the appropriate wares are for that type of factory.

Sectors designated as ore mines begin to produce ore once their sectors have reached 100% efficiency and have 100 or more civilians in residence. The game "magically" distributes this ore to all 100% efficient factory sectors where it is manufactured into appropriate wares. On any island, a player must have enough mines to produce a bit of excess ore since this excess ore is used as fuel for the ships.

3.7.4 Limits on time accrual and expenditure

Each game has a fixed limit of how many units of both movement and update time can accrue in a Capitol sector before some of it must be used. If a Capitol sector is "full," time passes in the game without any changes being made to the Capitol sector. Time accrues on a player-island by player-island basis; there is no facility for transferring movement or update time from one island to another, nor from one player to another.

Since the Capitol sector "holds" the movement time, if the Capitol is lost to an opponent's attack, the player loses all ability to execute commands that expend movement time until a new Capitol sector is designated.

One situation that arises frequently is that a player has completely occupied a home island, and there is no great need to use movement time to actually move anything about.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

A player can always transfer movement time directly to the update side of the ledger without actually moving anything.

3.7.5 Game activities that consume neither movement nor update time

Not every activity requires movement time. Activities that require no movement time may be made anytime during the game, even when a player has no movement time or when his Capitol has been captured. These include:

- Firing artillery pieces.
- Loading docked ships with materiel and people.
- Building factory wares from accrued production units.
- Doing anything with ships, whether they are docked or not.

3.7.6 Efficiency and land operations

If a sector is less than 100% efficient, it is easier for an enemy to invade and conquer. From an economic standpoint, low efficiency means that factory sectors that are less than 100% efficient do not produce new wares.

3.8 Ship time

3.8.1 At creation

Each new ship is supplied with one day of movement time at creation. This block of time is equivalent to the number of sectors it can move in one day. The grant of initial time allows a player to undock a ship after it is built, and to move it some distance away from the shore.

3.8.2 Time accrual

Each ship accrues the ability to move a certain number of sectors per ship-day. Generally, this factor is set so that an average speed ship (22 sectors/ship-day) can travel between islands in about 24 hours. For many games, a ship-day is a day.

A ship cannot accrue more than one day's worth of movement time. If a ship is "on course" at maximum speed, it accrues no time because the time is being spent as it accrues.

3.8.3 Fuel Consumption

All ships consume fuel whether or not they are moving. The data for fuel consumption are contained in the game's database. If a ship runs out of fuel while it is at sea, it begins to rust. When a ship is fully rusted it is sunk, and the record of the ship is removed from the database.

3.8.4 Efficiency and ships

Efficiency of less than 100% affects most functions of the ship. For example, a ship's maximum speed at any one time is equal to the maximum speed for that class of ship times the efficiency. Efficiency affects the efficacy of the ship's guns, but not their range.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

To repair a ship (i.e., restore it to 100% efficiency), it must be brought to a dock (ship factory sector) owned by the player, and there it must be refurbished using the production time units accrued in that sector.

3.9 Factory operation

3.9.1 Ore mines

Without ore, there can be no wares made, and there would be no fuel for ships. The simplification provided by this one universal substance from which everything is fabricated is important to the playability of Empire. With each expenditure of update time, the ore mines pump an endless supply of ore from the ground, and it is instantaneously transferred to factories that need it.

Each sector has a characteristic called its “sample rate.” The sample rate is a randomly generated characteristic that tells for this sector how many units of ore come up from the ground per update time unit expended. Clearly, players should designate sectors as mines where the sectors have a high sample rate.

For there to be some ore left over for the ship’s fuel, there must be a small excess of ore produced per island beyond the minimum amount necessary to operate the factories.

3.9.2 Explosive, Artillery and Plane Factories

These three factory types have in common that a player need not do any specific building of wares. These factories instantaneously change production time units into the wares they produce. In the case of explosive and artillery factories, the explosives and artillery pieces are added to the inventory of things in that sector up to the maximum concentration per sector.

Plane factories are slightly different. Plane factories produce a plane each time their accrued production time hits the cost of a plane, but the planes themselves appear as the contents of the airport sector nearest them on the same island. If a player has no airport sector on the island, the production time units increase with no planes being produced.

3.9.3 Ship Factories (Docks)

As we have seen, creating a navy involves a bit more of a selection process on the part of the player. As a result, these sectors accumulate production time units until the player issues a command to build a particular kind of ship at that sector.

When ships are produced, they are empty. They are created with a location the same as the sector that produced them. For any materiel to be place on a ship, it must either be docked (and filled from the associated land sector), or it must be in the same sector at sea with another ship that is capable of tending it.

3.10 Automatic actions provided by the game

One might ask, “What takes place while a player is not logged on?” The answer is that the Empire game engine provides a number of automatic actions consistent with sane play. Here are some examples:

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Ships that are in motion remain in motion and on course unless they run out of fuel or encounter land sectors. Movement time accumulates in the capitol sectors on all islands for all players at the appropriate rate.

If a sector that contains artillery pieces, explosives, and military is attacked, that sector will return fire on the appropriate target. In other words, if the sector is bombed, the flak is directed at the offending aircraft; if it is shelled from a ship or another land sector, then that ship or that sector is the target.

An aircraft carrier will attempt to defend any ship within the flight radius of its aircraft. Likewise, cruisers will attempt to shoot down aircraft that attack any ship within its guns' range. See the section concerning sea combat for more information.

3.11 Combat

As one might expect for a war game, a tremendous amount of Empire is concerned with combat, combat resolution, and the mechanisms of letting the players know the outcome.

3.11.1 Basic principles

A player must be logged on to engage in combat. This does not seem like an unnecessary restriction on play since it would seem that the "uses and gratifications model" of play for Empire would indicate that most players would want to be actively connected to execute their own military plan.

Outcomes of combat are determined by sensible objective rules, with each rule having only a small uncertainty window.

Additionally, the game does not prevent one from attacking one's own sectors or ships. As in all cases in the real world, "friendly fire" incidents are generally devastating.

3.11.2 Concepts governing land combat

The following is a table of the basic rules that are invoked to resolve combat and a list of the scenarios that are governed by the rule.

Rule	**Scenarios Governed**
Efficiency of both the attacking forces and the defending forces figures into result	All except bombing runs. Individual aircraft are construed to be 100% efficient.
Defending forces have a 3:2 advantage.	Infantry and artillery ground combat. The assumption here is that the defending forces can see the point of origin of the attack.
Last military occupant of a sector cannot be eliminated by bombing or artillery.	All. Forces players to make a ground assault to capture territory.
Civilians are removed as collateral damage.	The ratio of military to civilians killed varies on the type of assault, but some civilians are always killed.
Efficiency always drops to less than 100% after an attack.	Ships and sectors.
Accumulated production in factory sectors is wiped out by any attack.	Factory sectors.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Materiel in a sector can only be captured by a ground assault. Artillery bombardment and bombing tends to destroy materiel.	Sectors.
Mountainous terrain is difficult.	All. It is difficult to attack a mountain sector; it is likewise difficult for a player to move materiel into a mountainous sector.

Rules governing basic combat.

3.11.3 Concepts governing sea combat

Sea combat is quite a bit different from land combat for two reasons: [1] the granularity level of a ship is somewhat smaller than that of a land sector, and [2] there are many different kinds of ships, and they interact in different ways.

3.11.3.1 Measuring distance on the ocean

A ship's position is measured by the game as a floating point pair of coordinates, quite unlike measurements on the land. A ship is considered to be "in" the sector named by the integer portion of this pair of floating point numbers. An unlimited number of ships of any number of players will "fit" into a water sector.

Ships of the same nation that are in the same sector can "tend" each other, meaning that the player can move materiel from one ship to another one in the same sector. The most common use of this feature is refuelling, although ships can be loaded with other materiel while at sea.

Finally, when ships are travelling across the open ocean and reach a land sector of an island they halt. However, they do not "dock" at the land sector, even if the sector in question is a sector owned by the same player that owns the ship. Thus, a ship whose location coordinates put it in a land sector may either be docked or undocked. A ship that is docked may unload its cargo; a ship that is undocked may not.

3.11.3.2 Fleet organisation, and mutual defence

The game contains a mechanism to allow a player to group several ships to simplify navigation. For example, a player might want to designate several ships to travel together to a neighbouring island for the purpose of an invasion.

[1] Ships that are within each other's gun range will mutually defend each other if attacked by land based artillery or the guns of other players' ships.

[2] Ships that are within the flight radius of carrier based planes will be defended by that carrier if the ships are attacked from the air.

[3] Ships within five sectors of a depth charge capable ship will be defended from submarine attack if the attacking submarine is also within five sectors of the destroyer.

3.11.3.3 Ship to ship shelling

Ships may fire at other ships within range. The result depends on the distance to the target, and the quality of the guns. The quality of the guns is dependent on the type of ship as well as its efficiency.

There is no such thing as a free shot at a ship unless it is unarmed. It is assumed that ships that are close enough to each other to exchange shells "exchange fire" with one of

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

the ships simply being the instigator. The factors of distance and gun quality also affect the return fire.

3.11.3.4 Shelling between ships and land

When ships shell a land sector, if they are with range of the sector's return fire all the guns in that sector return fire simultaneously. Some sectors are clearly much more "powerful" than others, and ships have no way of seeing into a sector to determine the consequences of shelling it ahead of time.

3.11.3.5 Air attack and flak at sea

It is common for ships to be attacked at sea by aircraft, either land based or from another player's carrier. The attack is simple from the player's point of view, and all that is required is that the source (carrier or airport land sector) have planes, pilots (military population), and explosives (to be used as bombs).

Defence is more complex. The game first determines if the target of the attack is within the flight radius of a friendly aircraft carrier. If so, that ship's planes come up to meet the intruding aircraft in numbers at most equal to the number of incoming aircraft. If any incoming planes survive, then they are hit with one round of flak from every friendly ship within flak range. Only after penetrating those two defences do the incoming aircraft drop bombs on the ship that is the subject of the attack.

3.11.3.6 Submarines and torpedos

Submarines, when submerged or on the surface, may fire torpedoes. Torpedoes are quite destructive, so players have a strong incentive to use them where possible. When a submerged submarine shoots a torpedo, its position is immediately known to all ships within five sectors, whether or not they are sonar capable.

The game engine resolves this type of combat by considering the number of depth charging friendly ships within the radius of the attack, the type of ship being attacked, and the distance of the attacking submarine. The submarine is attacked with depth charges whether or not the torpedo hits the mark.

A submarine that is stationary and submerged is construed to be resting on the bottom of the ocean, and it is therefore invisible to enemy sonar. If depth charges are dropped in the sector where a submarine is resting on the bottom, they will be less effective than if the position of the submarine were known.

3.11.3.7 Transfer of ships at sea.

While playing earlier versions of the game, a flaw was discovered in the transfer of materiel. Originally, the transfer mechanism was executed solely on the "from" side, with no corresponding actions of the part of the recipient. Clever players discovered that one could do the following: [1] Sail a ship just within the flight radius of an enemy carrier. [2] Re-flag the ship to the enemy nation. [3] Begin attacking the newly transferred ship with one's own planes.

This manoeuvre would cause the enemy carrier to begin defending this ship. Repeated attacks would gradually deplete the planes of the opponent's carrier, thus leaving the entire fleet open to attack without the resource of air defence.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Thus, any transfer of materiel now requires acceptance by the recipient. While not so necessary for land based materiel (there is no similar “back door” to land play), the transfer-accept paradigm has been found to be useful for standardisation.

3.12 User Interface Requirements

A game as complex as Empire must have a substantial user interface component. In fact, the level of effort to produce a satisfactory user interface for Empire is around three times the level of effort for the server piece. Fortunately, some stepping stones along the way reduce the need to have a “big bang” delivery of the user interface. These stepping stones are even more important when one considers the difficulty in testing a system as complex as Empire.

3.12.1 Complexity & and the Two-man Cockpit Problem

Modern passenger jets are designed to be flyable with a two man crew. Older planes used at least three people. The aircraft industry discovered that the cockpit user interface had to change when few pairs of eyes were looking at the “dials and gauges.” Empire is similarly complex, and the instrumentation necessary to play it falls into these interacting categories:

Economics: “Are there enough mines on the island to keep the factories supplied? What is the balance of production between planes, ships, artillery, explosives, population growth?”

Geographic: “What does my map of the world look like? What is the detailed view of a particular island I occupy? Where are my ships?”

Military: “What is my troop strength? What level of damage will be inflicted by an attack? What can I know about adjacent sectors occupied by the enemy?”

Forecasting and projection: “When will my ships arrive at the new island? How long will it be until my factories are productive? When will I have enough production to build an aircraft carrier?”

Redistribution and movement: “I want to distribute my artillery and explosives around the island rather than having them remain in the factories. I want to load a new ship with supplies. I want to bring my battleship close enough to shell the island.”

3.12.2 Data Representation

The primary challenge in the Empire UI is scale. As a Commander-in-Chief, a player needs a view equivalent to that of a world map. As a player takes on the role of a General, there is a need for a view of perhaps 96 sectors at a time (12 wide and 8 high on normal 4:3 displays computer displays) to resolve intense combat to a useful level.

There is also a need to have an “island scale” view of the world, and to provide a player with the ability to rapidly switch views between several occupied islands. Naval scenarios and activities are similarly scaled. Experiments conducted by DEC and Adobe have shown that rather than having continuously scaleable views, most users of visually intensive software would prefer to have a short but useful list of fixed scales. This approach would seem to be applicable to Empire.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

The secondary challenge is dimension. For example: it would be useful for a player to toggle between economic specifics and combat details in order to avoid visual clutter. The UI challenge in this case is making sure there are enough visual clues that a player knows what view is currently being displayed. Extensive reliance on colour could be a problem, since 10% of the primary audience (male) has compromised colour vision.

The fat client allows a degree of freedom for the designer that is absent in web based games. The entire processing power of the client CPU may be brought to bear on rendering the images without requiring network traffic. The amount of CPU power necessary to convey to a user that he has been attacked by player #n's cruiser at sector (x, y) of island z is negligible compared with the effort to render that information visually.

3.12.3 Target Platforms for the Client

The only reasonable conclusion about the **fat client** side of Empire is that players will either have, or have access to, a Windows PC. Since gamers are likely to have up-to-date PCs, we will further assume that the client be developed for Windows 2000/XP and later editions of Windows.

The **thin client** is a program that actually runs on the game's server and handles text only. Given that at some level the data is moving back and forth via XML, any platform could make use of the thin client interface.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

4 Catalog of Modules

4.1 COORDINATES

4.2 DIRECTFILE

4.3 GAMETIME

4.4 GENESIS

Genesis is high level function that can only be invoked from the server's direct interface to the game engine. Many of the game's parameters are configurable within bounds. The Genesis function collects these parameters from the user, and then generates the pseudo-random details of a particular game.

4.4.1 Data files

Genesis creates all the data files in a directory chosen by the user. These files are detailed below:

File	***Contents***
globals.empire	A plain text file containing pairs of global symbol names (eg: “clockticksperday”) and values.
players.empire	A binary file with as many valid records as there are players. Each record contains the persistent information about a single player. Record #0 is empty. Records #1 through #n are created when the game is built. As players join the game, their information is stored in canonical order of first logon.
islands.empire	A binary file with one record for each island in the game. Record #0 is empty. The persistent information (absolute location and size of the island, etc.) is stored one record per island. All the records are created when the game is built.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

File	***Contents***
sectors.empire	A binary file with one record for each land sector. The file is organized in blocks of records by island, with enough records in a block to accommodate the largest possible island. As a consequence, not every record in the file is associated with an actual sector. For example: Record #0 contains the information about sector (0,0) of Island #1. Record #1 contains information about sector (1,0) of Island #1. Etc. If the maximum dimension of an island is 40, then the data for the sectors of Island #2 begins at record 1600 (40*40) regardless of the actual size of Island #1. All of the records in this file are created by the Genesis function.
ships.empire	This file is built with only one record (Record #0) when the game is created. Records are added as players build ships. Each record contains the current state of a single ship.

4.4.2 Terraforming

After Genesis collects the information and builds the data files, it calls a routine named Terraform(). Terraform creates a useful topography on each island. In almost all cases, the terraforming will create a somewhat ragged edge to island, with mountain sectors being more likely near the center than the edge.

4.5 GLOBALS

4.5.1 Constructor

Fairly obviously, GLOBALS is intended to be a singleton pattern object. The actual GLOBALS object is built in the main() routine. Other functions that need access to the global data should create a SINGLETON<GLOBALS> object, which is simply a smart pointer to the real data.

For safety, the constructor invokes a function named CreateMe() to instantiate the data.

4.5.2 Reading the persistent data from disc

ReadData() opens a file name provided as an argument. This file name should be some flavor of “globals.empire”. This data is stored in a map<>, so that one can look up the value of some global symbol by name.

4.5.3 In core tables

The inherent characteristics of the various ships are stored in a map<> entity. This associative array is populated with static data when an instance of the game engine is created.

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

The frequently used data about islands and their locations is stored in

4.5.4 Interface to retrieve global data

GLOBALS offers several ways to retrieve the data. To retrieve the numeric value of a symbol use either of constructs below, depending on which is the most convenient notation. The function v() and the overloaded operator() are identical in function.

```
SINGLETON<GLOBALS> g;
double val1 = g->v("circumference");
double val2 = (*g)("circumference");
```

To retrieve the string value of a symbol use:

```
SINGLETON<GLOBALS> g;
string strval = g->s("circumference");
```

4.5.5 Saving the data

At game shutdown, or when changes are made to the persistent data, one should call the function WriteData(). It stores a file in the format expected by ReadData().

4.5.6 Inspecting the data

Like most modules in Empire, GLOBALS contains a function named Dump(). Dump() writes to an ostream that you must provide as an argument.

4.6 ISLAND

4.7 MATRIX

4.8 PARSER

4.9 PLAYER

4.10 PROMPT

4.11 SCANNER

4.12 SECTOR

4.13 SHIP

4.14 SINGLETON

4.15 STOPWATCH

4.16 UTILITIES

This is the end of the document

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.

Serial Number	Date of last Revision	Number of Pages
46	2004/12/09 16:57	29

All information in this document in proprietary and confidential. Bright-Crayon, LLC is incorporated in the State of Delaware, and is registered as a foreign corporation in the Commonwealth of Virginia.