NONE OF LINKS WORK SRY :-(

PROJECT GODEVAC

TA-II, we're gonna make it happen!

About this document

This document is a complilation of ideas and opinions from internal team members and the TA community. Scribes such as myself has translated ideas into more foreseeable designs and explainations.

The workings behind this document are that it is made public and people everybody is allowed to say what they like about anything in it. If there is a positive motion by several people to get something changed it can be, usually you will have to start a thread on the forum to prove this. New suggestions can go in here as long as they are proved popular by the public in the same manner.

If you think you can better explain something written in here email me (ole_v2@dsl.pipex.com)

The purpose of this document is to provide all members, especially new ones, with the nessarily information to start work as soon as possible.

- Godevac Leadership | Ole

Items of Reference:

Art mailing list:
godevac@smartgroups.com
http://www.smartgroups.com/group/group.cfm?GID=2089986

Programming mailing list:
godevacprogramming@smartgroups.com
http://www.smartgroups.com/group/group.cfm?GID=2367170

Offical Website
www.godevac.de.vu

Offical Forums:
http://www.tauniverse.com/forum/forumdisplay.php?s=&forumid=85

Contents

Introduction

Team Members

Name	Position	Email	Msn
Sash	Webmaster	nebuchadneza@gmx.net	
Goliat	Modelling	goliat@darkabyss.de	goliatskipson@uni.de
oLE	Leader, Music, Sound FX	ole_v2@dsl.pipex.com	*
caydr	modeler		
krackers87	modeler	krackers87@hades-combine.com	
me22	programmer/fmod implamentation	me22@fastmail.ca	*
drvali	programmer	drvali@hotmail.com	*
pewee	programmer		
pavlo	Lead programmer	pavlo@liana.com.ua	a2free@hotmail.com
neutron	concept artist/designer	n3utr0n@hotmail.com	*
Eightball Maniac	scribe/plot writing	eightball_maniac@hotmail.com	*
Stinky Sheep	modeler	stinkysheep@hotmail.com	
Yoyobuae	programmer	supersayoyin@hotmail.com	
Xiphias	Temp Scribe	neversnake@cwazy.co.uk	

Policies

- Communication

Team communication works through the TAU forum.

Announcements of importance affecting the entire team or for attention to be drawn to the forum are done through the Smartgroups mailing list. There are to mailing lists, one for the programmings team and one for the art based team. Any art based member can make an announcement using Smartgroups by sending mail godevac@smartgroups.com and any programmer can make announcements by emailing godevacprogramming@smartgroups.com please be aware that everyone in the group will receive such emails.

Smartgroups provides 20megs of webspace for pictures, an additional 20meg for general files, calender and bookmark storage which the entire team shares and has access to. Go to the group home page http://www.smartgroups.com/group/group.cfm?GID=2089986 ant login To access this.

What this document does and how it should be used is at the very top on the first page and if you haven't read it you should. Go now, read!

- Outside world

The Godevac website can be found at www.godevac.de.vu, Sash is the webmaster and responible for new mediacontent while I make the majority of the new postings. Godevac is an open-source project so we try to make as much of our work as possible public.

Godevac has an entirely public forum which anyone can read or post on. 3rd party contribution is allowed and encouraged. Work is published as it is completed, Alphas are released. Components parts should be released separately on occasion.

The idea behind this it so receive public feedback at all stages and allow other people input in more than just a suggestive basis. Allowing anyone who thinks they can help to just submit something and have the chance of it being included.

The member initialization process is relatively simple, as our discussion forum is public anyway all external members of the public have to do is to get to know us and submit something of benefitical to the group. An official member is defined as someone on the Smartgroups member list.

- Rules

There aren't too many rules to Godevac, just to be nice to fellow members. Some members would request that you only use the Smartgroups mailing list occasionally and general discussion is meant to be conducted in the TAU forums.

- Work

This is a voluntry organization so we can't force you to work. Remember you joined because you had a genuine interest and wanted to contribute.

Job assignment is based on an informal agreement between you and me. Its best if you can stick to the agreement but if you need to change it for whatever reason it is your responiblity to email me.

- Quality Control

The team should give criticism as often as possible. Keep it constructive and frequent. Please appreciate that there are people not as good as you are and you'd do well to assist those people.

All work made is published provided:
a) the author would like it published
b) there are no serious technical faults
c) it fits with the Godevac designs and prinicples
d) it is compatible with the rest of the Godevac code
e) it is up to your personal standard
f) you've made as least one attempt at altering it according to criticism you're received.

Despite the number of points there this really isn't difficult to do, if your work does the job and you've tried your best your in; basicaly.

Gameplay: Basics

The choices I've made here are designed to preserve or expand upon the features of TA that we loved whilst fixing some of the disliked features and adding some. If you have an idea for this section please read the rejects and opens before you submit it, otherwise you are wasting your time.

Two Teams

Godevac will have two teams. TA had 2 teams and that offered more than enough scope.

The two teams are Idus and Maul, The names are designed to sound that little bit darker than Arm and Core. Both teams will appear to be robotic.

Metal and Energy

Construction and the Metal/Energy system are the same as in OTA you do not need to read on if you are familiar with it.

The construction of any unit requires an amount of metal and energy. Typically units built for better durability have a more expensive metal cost while units that require more complex manufacture methods will have greater energy costs.

The purposes of the metal and energy system is to challenge the player to maintain a balance between the 2 resources. Building a unit causes a continuous and unvaring drain on metal and energy that is spread out over the time that the unit takes to be built.

Metal can be obtained by reclaimation and or deposit extraction. Reclaimation is the process of converting metalitic material into an easily storable form, it doesn't take very long to perform and can provide large bursts of metal but is not reliable because it is dependent on large quanities or metal to be lying around for the taking and you giving the orders to reclaim them. The choice method of obtaining metal is desposit extraction which is where structures are built over metal despoits in the ground, these structures (metal extractors) bore into the despoits and provide an undieing and reliable source of metal.

This way a unit costing 10,000 metal can be built even when your stores are low because your consistant income is offset against your consistant outgoingings.

Energy production works in very similar way and can be offset in the same fashion. Organic material can be reclaimed for energy and Solar panel and tidal generators (and many more desides) can be used to provide constant sources of energy income.

Godevac is going to stick to the standard production methods used by TA. Construction units will build building and factories and labs will build mobile units. Constructions units have nathlathe lasers which reclaim, repair and bulid units.

PROJECT GODEVAC

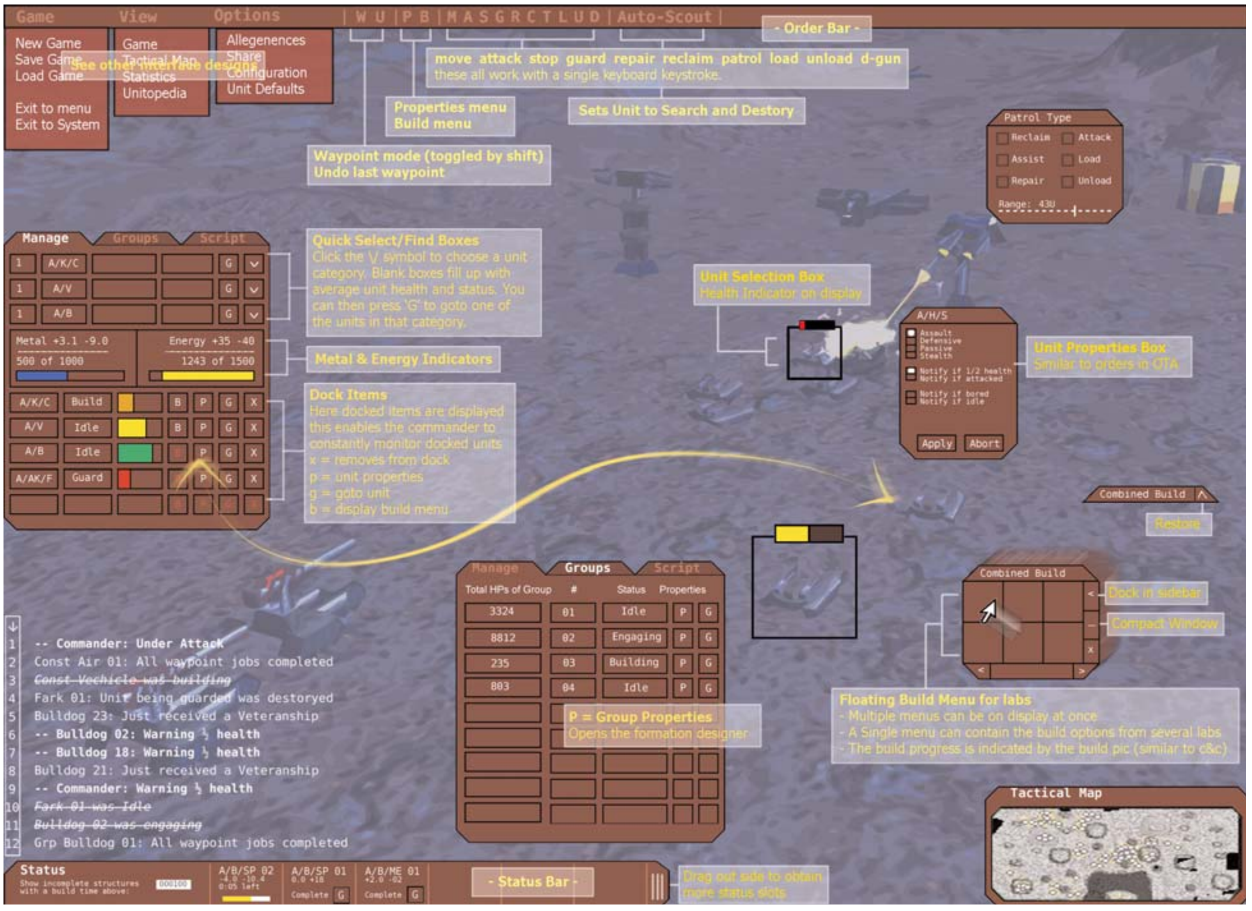
TA-II, we're gonna make it happen!

Interface and Script

Interface Preview Picture

This an example of the ingame interface, a lot of the point and click actions will be identicial to that of OTA this outlines most of the new. The big arrow in the centre of the image indicates a mouse dragging action.

Interface items to be added to this section
- Statistic View
- Unitopedia View
- Allegencences Dialouge
- Share Dialouge
- Configuration Dialouge
- Unit Defaults Dialouge
- Formation Editor
- Working model made in VB where you'll be able to click things are try stuff out.



move attack stop guard repair reclaim patrol load unload d-gun
these all work with a single keyboard keystroke.

Properties menu
Build menu

Sets Unit to Search and Destory

Waypoint mode (toggled by shift)
Undo last waypoint

Quick Select/Find Boxes
Click the V symbol to choose a unit category. Blank boxes fill up with average unit health and status. You can then press 'G' to goto one of the units in that category.

Metal & Energy Indicators

Dock Items
Here docked items are displayed this enables the commander to constantly monitor docked units
x = removes from dock
p = unit properties
g = goto unit
b = display build menu

Unit Selection Box
Health Indicator on display

Unit Properties Box
Similar to orders in OTA

Total HPs of Group	#	Status	Properties	
3324	01	Idle	P	G
8812	02	Engaging	P	G
235	03	Building	P	G
803	04	Idle	P	G

P = Group Properties
Opens the formation designer

Floating Build Menu for labs
- Multiple menus can be on display at once
- A Single menu can contain the build options from several labs
- The build progress is indicated by the build pic (similar to c&c)

```
1  -- Commander: Under Attack
2  Const Air 01: All waypoint jobs completed
3  ~~Const Vechicle was building~~
4  Fark 01: Unit being guarded was destoryed
5  Bulldog 23: Just received a Veteranship
6  -- Bulldog 02: Warning ½ health
7  -- Bulldog 18: Warning ½ health
8  Bulldog 21: Just received a Veteranship
9  -- Commander: Warning ½ health
10 ~~Fark 01 was Idle~~
11 ~~Bulldog 02 was engaging~~
12 Grp Bulldog 01: All waypoint jobs completed
```

Drag out side to obtain more status slots

Modifications going to be made to this design:
- Metal and Energy on order bar instead of manage
- Make all windows compactable
- Resizeable corners
- Unit pics in the combined build menu for realism
- Replace cyptic a/h/s stuff with real names.
- New menu 'Floaters' allows you to toggle floating windows on and off and create copies.
- F1throught 4 movers.
- F5 compact and uncompact
- F6 display/hide menu for selected unti.
- Snapping to edges

Script

Scripting (in this case) is the act of giving a unit commands. Just like waypoint but with the critical difference that it is code driven.

From now on the word maploc refers to an XY location on the map e.g. 355:657
',' separates commands in groups of commands.
';' terminates a command.

Interpreter error messages:
1. Unknown command
2. Bad Syntax (correct syntax is displayed)
3. Incorrect commas or bracket placement
4. Unit can't perform that command
5. Map location is outside of map
6. Unit does not exist
7. Ambigious unit name
8. Not enough data. (no opperand)
9. You cannot specify a map location here
9. You cannot specify a unit name
10. You must specify a side
11. That side does not exist
12. Can't build there.

Run-time error messages:
1. I Can't get there
2. My target location is blocked
3. Hostile action prevented me from performing
4. Cannot build any more units, you've hit the maximum
5. Cannot load any more units.
6. No units to unload
7. Unit no longer exists.
8. That side no longer exists.

Commands: (I thought I'd use pink :P)

Long-hand: patrol...until
Short hand: pat...utl

Purpose:
The patrol command is used automate movement, the unit follows a series of movment points continously repeating any other command as its required in the game environment. As you would expect the maplocs can be filled in with clicks. Think of patrol as a loop.

Syntax:
patrol(maploc1,maploc2,etc)(command1,command2,etc)(range) until(condition);

Example:
patrol(24543:5432,25642:5464,27642:5464)(repair,reclaim)(500) until (me<50%);

This would cause the unit patrol from in a triangluar fashion stopping to repair any repairables and reclaim any reclaimables within a 500 point radius of its path. It will not repair or reclaim outside its range even it moves outside the range after its started. As long as its health is greater than 50% it will patrol.

patrol(load)(2000);

This would cause the unit to stand still and execute a load command to anything loadable within 2000. Notice there is no until, which is allowed.

Long-hand: attack
Short-hand: atk

Purpose: Well there wouldn't be much point in Godevac without attacking now would there? The unit and side can be automatically filled in by clicking a unit to attack.

Syntax:
attack(side name or color,unit);

Example:
attack(blue,ak4);
Will cause unit to attack one ak from the blue team

attack(ole,commander1);
Will cause unit to attack the commander of the team called ole. nooo!

Long-hand: move
Short-hand: mov

Purpose: Moves units about, only units that can move mind. In the case of factories it performs a park function, where all the units built by the faction go to the maploc, this was in OTA. Also as you can see you can string multiple maplocs together with this command.

Syntax:
move(maploc or unitname);

Example:
move(24543:5432);
move unit to 24543:5432.

move(commander1);
unit will move to the commander on own side.

Long-hand: stop
Short-hand: stop

Purpose:
Terminates any command that is executing.

Syntax:
stop;

Example:
stop;
simple and classic, but pretty useless in a script.

Long-hand: guard..until
Short-hand: grd..utl

Purpose:
Make any unit follow and assist another. If the unit has weapons and is set to assault mode it may attempt to engage units attacking the guardee.

Syntax:
guard(unitname)until(condition);

Example:
guard(commander1)until(commander1=100%);
Unit will follow and assist the comander as long as the commander's health is not 100%.

guard(peewee13);
As long as Godevac doesn't crash peewee13 will be guarded.

guard(conkbot3)until(conkbot=idle);
As long as construction kbot is not idle it will be guarded.

guard(me);
illegal command you can't guard yourself.

Long-hand: set
Short-hand: set

Purpose:
Sets the properties and behaviour of a unit.

Syntax:
set(item = value);

Example:
set(notify_bored = 1);
Unit will from now on notify if it gets bored.

Items and their values:
notify_bored = boolean;
notify_idle = boolean;
only a single binary '1' to share between these: assualt,defensive,passive,stealth. only for units with weapons.
boredom_time = integer; sets the amount of time it takes a unit to get bored.

Long-hand: load
Short-hand: ld

Purpose:
Orders transport units to load units to be transported. You can load units from any team.

Syntax:
load(teamname or color,mobile unitname);

Example:
load(blue,jethro3);
Unit will load the jethro from the blue team.

load(245:145);
illegal command

load(commander1);
illegal command. No side specified

Long-hand: unload
Short-hand: uld

Purpose:
Orders transport units to unload units to a location.

Syntax:
unload(maploc);

Example:
unload(2424:1235);
Unloads the next unit in the stack to 2424:1235.

unload(commander1);
illegal command. a location is specified not a unit.

Long-hand: repair
Short-hand: rep

Purpose:
Repairs damaged units.

Syntax:
repair(unitname);

Example:
repair(bobmech1);
Repairs bobmech1 on your own side.

repair(red,bobmech1);
illegal command.

Long-hand: reclaim

Short-hand: rec

Purpose:
Absorb the lovely metal or energy goodness from things.

Syntax:
reclaim(teamname or color,unitname);
reclaim(featurename);

Example:
reclaim(green,kbotlab2);
Reclaims the Kbotlab on the green side

reclaim(423);
Reclaims feature number 423.

reclaim(blue,423);
illegal command.

Long-hand: assist
Short-hand: ast

Purpose:
Orders a construction unit to assist with building an incomplete unit. The command terminates once the unit has been built. You can assist allys with building.

Syntax:
assist(side,incomplete unitname);

Example:
assist(blue,fido5);
Assists with the construction of fido5.

Long-hand: build
Short-hand: bld

Purpose:
Orders a construction unit or factory to start building a new unit. No number need be specified in the unit to build.

Syntax:
construction unit: build(unit to build,maploc);
factory: build(unit to build);

Example:
build(fusgen,253:546);
Construction unit builds a fusion generator at maploc 253:546.

build(ak);
factory build an ak.

build(ak,2465:134);
illegal command

Long-hand: wait..until

Short-hand: wt..utl

Purpose:
As long as the condition is not met or minutes have not expired the unit stands idle and doesn't notify of idleness or boredom during this period.

Syntax:
wait(minutes)until(condition)

Example:
wait(5);
As long as unit hasn't been waiting five minutes it waits.

wait until(me < 100%);
As long as it isn't damaged it waits.

wait;
waits forever. This is the say as having no code in the script.

wait(10)until(metallevel<10%);
As long as unit hasn't been waiting ten minunte or metal level drops below 10%, unit waits.

Long-hand: if
Short-hand: if

Purpose:
If is the only command that cannot be performed using the game interface. It executes other orders dependent on a condition.

Syntax:
if(condition)command;
condition are made of (unit operator value)

Example:
if(blue,commander1<20%)guard(commander1);
if the commander's health is below 20% at point of if's execution guard the commander.

if(metallevel>1000)build(fusion,432:562);
if the variable metallevel is greater than 1000 at point of if's execution guard build a fusion.

Long-hand: loopif
Short-hand: lpif

Purpose:
Same as an if command only the condition is continuously checked while the unit is idle.

Operators:
First I have better just say that we are going to be using these operators:
< - less than

> - greater than
= - equal to
!= - not equal to
% - indicates a reference to health

there'll be no need for + - * / operators and bracket won't be allowed as they are purely for syntaxal purposes.

Global Environment Variables:
metallevel - current amount of metal in storage
energylevel - current amount of metal in storage
metalin - current amount production
metalout - current metal drain
energyin - current amount production
energyout - current metal drain

Thinggies that return a value:

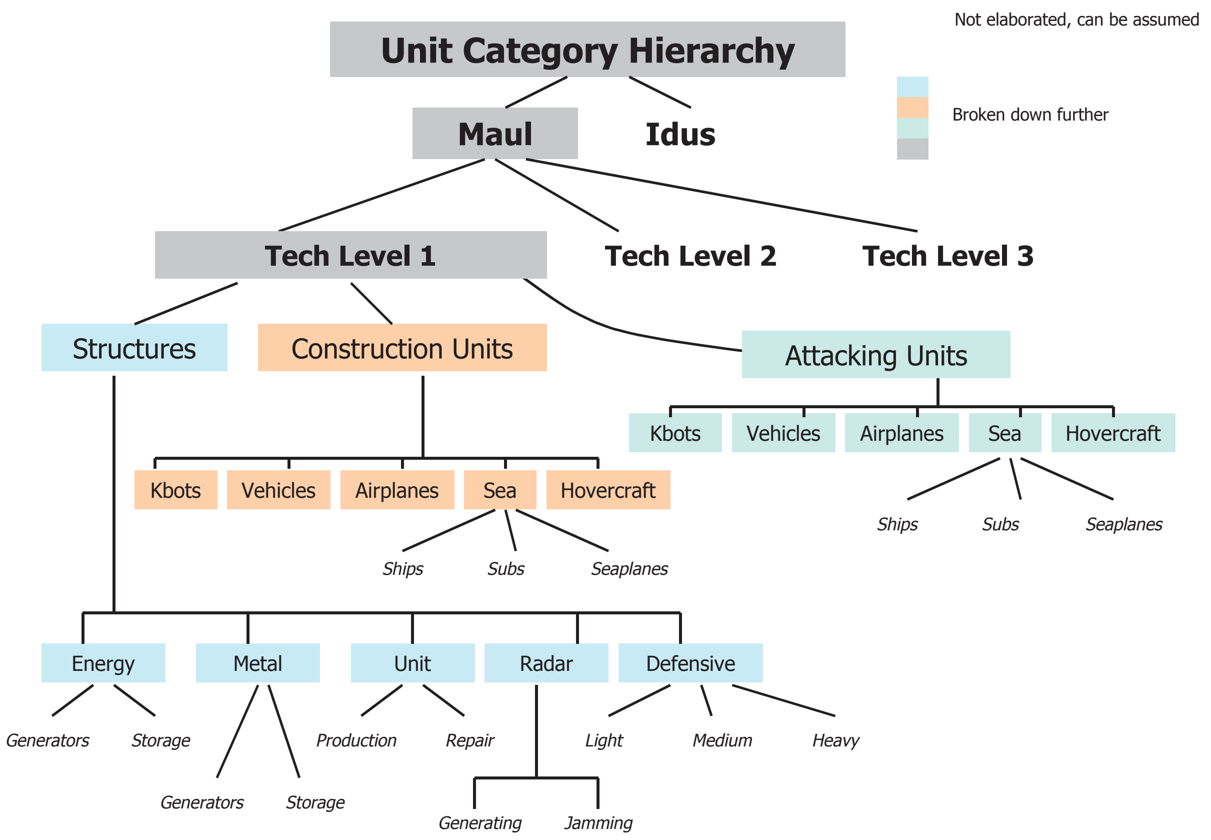
me	- unit being scripted
damaged	- see if command below
underattack	- ditto
idle	- ditto
bored	- ditto

PROJECT GODEVAC

TA-II, we're gonna make it happen!

Rules

This section in time will come to explain the reasoning and structure behind a single game of Godevac as well as an overview of unit roles. See the first in this section a Unit Hierarchy Document, yes I know it looks a bit crappy, unfortunately Freehand was playing up that day so I had to use Powerpoint.



Game Modes

This section will explain the working used for multiplayer, skirmish and campign.

<u>Single Player Campaign</u>

blah

<u>Skirmish and Mutliplayer</u>

Begin by specifing your team or player name, this cannot have any commas or semi-colon in it (because of the use of commas and semi-colons in the script).

In skirmish mode you will be able to use many teams, perhaps 20. Each team can then be assigned to a controler, such as an AI or human player. The point is that any single controller can be assigned to many teams and therefore control bases on several areas of the map.

The more teams a controller is assigned to the larger inital metal and energy storage he is given but all his teams will share off that single storage.

Plot

Game plot - who Idus and Maul are, where they came from, why they are so pissed off with each other. etc. Eightball Maniac will be writing this a bit later.

Rejects and Opens

-Opens

Should a third race be included later?

- Rejects

Metal desposits run out after a period of time.

PROJECT GODEVAC

TA-II, we're gonna make it happen!

Program: Engine

Introduction

Following me22 telling me that the programming team has no idea what to do I have produced a early incomplete engine section. This section will explain what will be required from the engine.

Please comment on the information here because its quite obviously far from perfect.

Maps

Renderer

OpenGL. agreed. There a debate going on right now about whether we'll making it 2.1D or 3D. My opinion is to go with full 3D, however if someone here know how to make a 2.1D straight away then that is what must be done.

Besides this I think everyone is in agreement that Terragen made maps are the only way forward, so its going to have to be capable of:
- Large, high resolution, 24-bit base texture
- Colored lighting from moving or static objects affecting map surface
- Smoke particle systems
- Water reflection system
- Deformable terrain
- Destorable and animating features (ideally in 3D)
- Separate heightmap file

Editor/Map definition file

As the maps are going to be created in Terragen all the editor will need to do is:
- Initial processing
- Set map properties (water height type stuff)
- Provide tools for making a single player campaign
- Heightmap editing tools
- Choose allowed and disallowed units.
- Specify unit scripting

Scalability

I think the entirely of the engine code should be designed on the basis that there will be large volumes of data to process. So that 0xFFFF unique weapons and units can be made.

Units

Unit and Weapon Definition Files

See here

These should work as a ascii editable files. Unless that is going to reduce performance.

When a unit is built it is given a name similar to 'Bertha1' this is important because it will be referred to in the scripts that the user writes. Its up to you programmers to decide if each unit is automatically allocated a number and a string name or just a string name.

Collisions

Basic rectangular collision masks, low processing requirement good for lots of units. Units won't collide as much though we hope because of better path finding.

Animation/Explosions

As much the same as OTA as possible, perhaps wreakages could burn a bit more.

Orders and Movement

- Acceleration/Deceleration rates and max speed
- Units react to map slopes
- Mounted cannons move
- Projectiles rise and fall
- Lines and symbols are printed on the game to indicate orders. Building plans appear as transparent buildings.

Statistical Processing

- Different build rates
- Different metal/energy drains and supply rates
- Different weapon strengths, fire rates.

File System

```
Godevac
        Engine
        Interface Skins
        Units
                Animations
                3DOs
                Unitpics
        Maps
```

- Music
- Weapons
- Features
- Docs
- Source
- Sounds
  - Weapons
  - Units
  - Ambience
  - Interface

Format Support

Audio

FMOD will do nicely

Visual

In thinking PNG (lossless compression) for map
3D0 if we want to use existing TA units. MDL has quite a following too.

Compression

The large map will have to be compressed although I agree with me22 that this should come later, for now we can rar everything up for internet transmision.

Interface

In-game

- Skinned interface
- Buttons, pointers,
- transparent lines and indicators
- Animating symbols
- Build Pics

Game Launching Menus

- Standard windows interface for specifying a game.

Rejects and Opens

Rejects

to be filled

Opens

2.1D or fully 3D maps?
What else is wrong with this lot?

If we can't make our own engine I am making enquires into an existing engine not well recognised, it was used for Rage Software's Incoming if you've heard of that, it features tanks, airplanes and vehicles on a fully 3D textures landscape and was famed for its explosions.

Otherwise the Unreal Warfare engine will be up to the job of RTS, if we opt for this it would mean the death of Godevac and everybody moving over to the Unreal Annihilation Project

PROJECT GODEVAC TA-II, we're gonna make it happen!

Unit Definitions

me22 made these unit definition files.

```
/***********************************

Godevac Unit Definition File Description
Maintainer: me22 <me22@fastmail.ca>
Revision 1.0: 16/03/2004

***********************************

— Key —

/* Comment */: no nesting; only between semicolon and next tag
"string": double quotes MUST be included
integer: possible relational operators for constraints
float: constraints possible here too
boolean: 1=true, 0=false
One|of|these|tags: Single Flag
Any&Number&of&these&flags: Whitespace separated flags

Note: Name = Value pairs are separated with ;
Note: a line can begin and end with arbitrary whitespace
Note: whitespace is any amount of a common whitespace character, including \n
Note: file radical is filename without extension
Note: filename radical is same for unit, model, script, and etc files,
which is what is mentioned in the built_by files.
Note: all distances in meters, all times in seconds

***********************************/

Info_Version = Lowest Compatible Version >= 0;
Info_Revision = File Version >= 0; /* For Mod Overloading */
Info_Name = "Unit Name";
Info_Desc = "Unit Description";
Info_Level = Build Tree Tier >= 0;
/* 0: Com Buildable
1: Lvl 1 Factory Buildable
2: Lvl 1 Builder Buildable
n: etc */
Info_Type = RESOURCE|FACTORY|BUILDER|ATTACK|SENSOR
Y|SPECIAL;

Info_Role = PASSIVE|ANTI|LR|ARTY|AA|DIRECT|MELEE;
Info_Group = GENERIC|KBOT|TANK|AIR|SEA|COMBO;
Info_Power = WEAK|LOW|MEDIUM|HIGH|INSANE; /* Generic strength for type.
For example Weapon Damage, Radar Range, or Build Speed. */
/* All the above stuff can be used by the AI and to make
the build menus coherant w/o GUIs or download tdfs */

Model_X = X bounding box dimention;
Model_Y = Y bounding box dimention;
```

```
Model_Footprint = “TA-Style footprint string”; /* format TBA */

Cost_Metal = Metal Cost >= 0;
Cost_Energy = Energy Cost >= 0;
Cost_Time = Build Time With 100-Strength Builder > 0;

Move_Speed = Maximum Speed >= 0;
Move_Accel = Time To Full Speed From Rest > 0;
Move_Brake = Time To Stop From Full Speed > 0;
Move_Turn = Time To Complete A 360 Degree Rotation > 0;
Move_Leash = Movement Freedom >= 0;
Move_Climb = 0 <= Maximum Traversable Upward Slope >= 256;
Move_Slide = -256 <= Maximum Traversable Downward Slope >= 0;
/* So you can let units jump off cliffs if if you want */
/* Building locations must satisfy both criteria */
Move_Low = -256 <= Deepest Possible Location <= 256;
Move_High = -256 <= Hightest Possible Location <= 256;
/* 0 is water level; Move_High for planes/hovers is the
flight altitude above ground level */
/* If Move_Low is 256, the plane wont land */
/* Only planes can be built below Stats_Low, although not
below water unless Stats_Low < 0 */

Stats_HP = Hitpoints > 0;
Stats_Sight = LoS Radius > 0;
Stats_Build = 0 <= Build Rate <= 500;
Stats_Rader = Radar Radius >= 0;
Stats_RaderJam = Radar Jamming Radius >= 0;
Stats_Sonar = Sonar Radius >= 0;
Stats_SonarJam = Sonar Jamming Radius >= 0;
Stats_Cloak = Cloak Radius >= 0;
Stats_Stealth = Stealth Radius >= 0;
/* If 0 is allowed, then 0 means that the Unit does not have the feature */

Resources_Tidal = 0 <= Tidal Efficiency <= 100;
Resources_Wind = 0 <= Wind Efficiency <= 100;
Resources_Mex = 0 <= Extractor Efficiency <= 100;

Resources_MStore = Metal Storage >= 0;
Resources_EStore = Energy Storage >= 0;
Resources_MIdle = Metal Use When Idle;
Resources_EIdle = Energy Use When Idle;
Resources_MActive = Metal Use When Active;
Resources_EActive = Energy Use When Active;

Weapon_Primary = “Main Weapon File Radical”; /* Script ID 1 */
Weapon_Special = “Special Weapon File Radical”; /* Script ID 0 */
Weapon_2 < Script Weapon ID < 10 = “Additional Weapon File Radicals”;
/* Primary determines attack range, chase targets, etc
2-9 must be specifically activated by a script
Special is the DGun-style button that never auto-attacks */

FX_Killed = “On-Killed Weapon File Radical”;
FX_SelfD = “On-Self-Destruct Weapon File Radical”;
```

```
FX_Corpse = "Corpse File Radical";
FX_Sound = "Sound Class Definition File Radical";

Notify_Idle = Notify When Out Of Orders;
Notify_Success = Notify Upon Successfully Acheiving A Waypoint;
Notify_Failure = Notify If Waypoint Impossible or Interrupted;
Notify_Critical = Notify Once Health Reaches "Red" Zone;
Notify_Attacked = Notify After Taking Damage;

/**************************************

— Notes on things NOT included —

Side (int): If you can build it, I don't care what side it thinks it is
Fixed (bool): If you can't move, I'll assume you're fixed
Canbuild (bool): If your Stats_Build is 0, I'll assume you can't build
canstealth (bool): Uses Stats_Stealth, see note above
UnitNumber (int): Dynamically allocated
Behavior (string array): Inferred from Info_* tags
Notify.bored (bool): Ambiguous
shootme (bool): Illogical. +shootme will be replaced with something that
lets you pick categories for your units to auto-attack
cruisealt (int): Incorporated into Move_High
nochasecategory (filename): Inferred by weapon information and Info_* Tags
stealthcost (int): Resources_(M|E)Idle
stealthcost.moving (int): Resources_(M|E)Active
```

```
/ ****************************************************
****************************

                Godevac Unit Definition File Examples
                 Maintainer: me22 <me22@fastmail.ca>
                      Revision 1.0: 20/03/2004

  ****************************************************
****************************/

/***** units/arm_mexx.info *****/

/* This one defines everything explicitly */

Info_Version = 0;
Info_Revision = 0;
Info_Name = "Mexx";
Info_Desc = "Metal Extractor";
Info_Level = 0;
Info_Type = RESOURCE;
Info_Role = PASSIVE;
Info_Group = RESOURCE;
Info_Power = WEAK;
```

```
Model_X = 32;
Model_Y = 32;
Model_Footprint = "(Dunno Yet)"; /* format TBA */

Cost_Metal = 10;
Cost_Energy = 1000;
Cost_Time = 3.7;

Move_Speed = 0;
Move_Accel = 0;
Move_Brake = 0;
Move_Turn = 0;
Move_Leash = 0;
Move_Climb = 32;
Move_Slide = 32;
Move_Low = 0;
Move_High = 256;

Stats_HP = 100;
Stats_Sight = 300;
Stats_Build = 0;
Stats_Rader = 0;
Stats_RaderJam = 0;
Stats_Sonar = 0;
Stats_SonarJam = 0;
Stats_Cloak = 0
Stats_Stealth = 0;

Resources_Tidal = 0;
Resources_Wind = 0;
Resources_Mex = 10;

Resources_MStore = 100;
Resources_EStore = 0;
Resources_MIdle = 0;
Resources_EIdle = 0;
Resources_MActive = 0; /* Metal Comes From
Resources_Mex */
Resources_EActive = 3;

/* No Weapons, So Commented Out *
Weapon_Primary = "Main Weapon File Radical"; /* Script
ID 1 */
Weapon_Special = "Special Weapon File Radical"; /*
Script ID 0 */
Weapon_{2 < Script Weapon ID < 10** = "Additional
Weapon File Radicals";
*/

FX_Killed = "small_die";
FX_SelfD = "small_selfd";
FX_Corpse = "arm_mexx";
FX_Sound = "mexx";

/***** units/arm_bertha.info *****/

/* This one only includes those tags that need something
other than default */

Info_Version = 0;
Info_Revision = 0;
Info_Name = "Big Bertha";
Info_Desc = "Long-Range Plasma Cannon";
Info_Level = 6;
Info_Type = ATTACK;
Info_Role = LR;
Info_Group = GENERIC;
Info_Power = HIGH;

Model_X = 32;
Model_Y = 32;
Model_Footprint = "(Dunno Yet)";

Cost_Metal = 2222;
Cost_Energy = 567000;
Cost_Time = 62.7;

Move_Climb = 8;
Move_Slide = 8;
Move_Low = 0;
Move_High = 256;

Stats_HP = 0;
Stats_Sight = 400;

Weapon_Primary = "arm_lrpc"; /* Script ID 1 */

FX_Killed = "big_die";
FX_SelfD = "big_selfd";
FX_Corpse = "arm_bertha";
FX_Sound = "lrpc";
```