World of Padman
Competition Mod

Uwe Koch
Oleg Kogut
Andreas Schneider

13. Juli 2007

Inhaltsverzeichnis

1 Motivation

1.1 World of Padman

World of Padman (im weiteren WoP) war ursprünglich eine Quake 3 Mod welche auf den Comics von Andreas „ENTE“ Endres [1] basiert. Mit der Freigabe der Quake 3 Engine am 19. August 2005 begann die Fortführung als Open Source Projekt und seit Anfang April 2007 ist die erste Version des Spiel erhältlich.

Homepage: **http://padworld.myexp.de**

Abbildung 1: World of Padman

1.2 Competition Mod

Durch seinen Fun-Charakter ist WoP nur eingeschränkt für eSports Turniere geeignet. Aus diesem Grund entschlossen wir uns eine Modifikation zu erstellen, die alle wichtigen Elemente beinhaltet, die für ein professionelles WoP-Turnier notwendig sind. Die Erstellung dieser Modifikation erfolgte im Rahmen des Seminars „Game Modding“ an der Universität Leipzig [2].

1.3 Aufgabenverteilung

Uwe Scoreboard Details, Teamoverlay, Chat Tokens, Waffeneffekte, Dokumentation, Votes

Oleg Show Netsettings, AutoAction, Competition Befehle, denyfollow/speclock, Crosshairs

Andreas Enemy Colors, mutespec, Referee Status ,Enemymodel, Dokumentation, Präsentationen

[1] http://www.enteswelt.de
[2] http://bis.informatik.uni-leipzig.de/de/Lehre/0607/SS/GameModding

1.4 Werkzeuge

Quake3 / WoP bietet eine umfangreiche Schnittstelle, in der Modifikationen am C-Code vorgenommen werden können. Als Werkzeuge setzten wir die Entwicklungsumgebung Visual C++ 2005 Express [3] ein. Für grafische Änderungen an den Modellen und der Spielgrafik wurden Grafikprogramme wie GIMP [4] und Paint.NET [5] verwendet.

Zum Abstimmen des Teams nutzten wir ein Entwickler-Wiki, sowie das Angebot von csie.org, welches einen kostenlosen Server mit der Versionsverwaltung Subversion bietet. [6].

Entwicklerwiki: http://wop-pgm.wik.is

SVN-Repository: https://opensvn.csie.org/wopcompmod

1.5 Fortschritt

Einige der Features die wir implementieren wollten, waren schon vorhanden und mussten nur noch aktiviert werden, so zum Beispiel das Teamoverlay. In diesem Fall haben wir neue Erweiterungen geplant und umgesetzt (variable Grösse, Anzeige der Spray Ammo). Das Unterbrechen des Spiels über den Befehl `timeout` konnte nur unvollständig implementiert werden, clientseitig kam es zu Problemen die wir im gegebenen Zeitrahmen nicht lösen konnten. Anstatt wie geplant die Variable `cg_DrawcrosshairTeamStatus` einzuführen entschieden wir uns, die Anzeige der Statuswerte von `cg_drawCrosshairNames` abhängig zu machen. Die Implementierung der Variable `cg_drawCrosshairColor` entfiel aus Zeitgründen. Alle weiteren Features konnten wir wie geplant umsetzen.

[3] http://msdn.microsoft.com/vstudio/express/downloads/
[4] http://www.gimp.org
[5] http://www.getpaint.net
[6] https://opensvn.csie.org

2 Installation

2.1 Download

Die aktuelle Version findet man unter folgender URL. Alle benötigten Dateien befinden sich in der ZIP.

http://www.soclose.de/misc/wop/comp/compmod.html

2.2 Installation

Entpacken Sie die ZIP-Datei in Ihren WorldOfPadman Ordner. Es sollte ein neuer Ordner *compmod* entstanden sein. Starten sie das Spiel wie gewohnt und wählen sie im Menü Mods den Eintrag compmod.

Alternativ können sie das Spiel auch mit dem Kommandozeilenparameter `+set fs_game compmod` starten.

2.3 Dateien

Die Datei *vm.pk3* enthält die Objektcode Dateien für die Module Server, Client und User Interface im Format qvm(Quake Virtual Machine). In der Datei *wop-comp_media.pk3* befinden sich alle Mediendateien wie Shader, Skins, Crosshairs, Waffeneffekte.

2.4 Die Konsole

Zur Eingabe von Befehlen und zum Ändern von Variablen dient die Konsole. Geöffnet und geschlossen wird sie mit der Taste ^ (Zirkumflex). Befehle müssen in der Form `/BEFEHLSNAME` eingegeben werden, Variablen entsprechend `/VARIABLENNAME WERT`. Alle Eingaben die nicht mit einem Slash beginnen, werden als Chat interpretiert.

3 Features

3.1 Enemy Models und Enemy Colors

Beschreibung: Die Konsolenvariable `cg_enemyModel` ermöglicht es dem Spieler, dem gegnerischen Team ein einheitliches Model zuzuweisen, so dass sich Freund und Feind sofort visuell und akustisch unterscheiden lassen. Der Variable können Werte der Form *modelname/skinname* zugewiesen werden, z.B.: `cg_enemyModel monsterpad/default`

Da die normalen Skins der Modelle nicht besonders gut zu erkennen sind, gibt es in der CompMod für jedes Model einen sogenannten Glowskin, die den Gegner farbig leuchten lassen. Die Farbe wird über die Konsolenvariable `cg_enemyColors` gesteuert, sie kann folgende Werte annehmen: white, yellow, cyan, magenta, green, red, blue, black

Implementierung: Der Glowskin besteht aus farblich neutralen (grauen) Texturen, die mit einem Shader belegt sind. Diesem Shader können zur Laufzeit bestimmte Parameter zugewiesen werden. In diesem Fall ist dies ein Vektor, der die Rot, Grün und Blauanteile der Farbe des Shaders bestimmt. Der Wert des Vektors wird aus `cg_enemyColors` abgeleitet.

Abbildung 2: Glow Skin

3.2 Chat Tokens

Beschreibung: Chat Tokens sind Platzhalter, die in Chat Strings eingefügt werden können und dynamisch durch Inventar- oder Spielstatusvariablen ersetzt werden. Mögliche Werte sind:

- #i - Name des Items auf das der Spieler zielt
- #l - Map Location (Ort) des Spielers
- #h - Health und Armor des Spielers

- #s - Spray Ammo des Spielers

Wenn ein Spieler z.B. seinem Team mitteilen will, wo er sich befindet und wieviel Energie er hat, kann er eine Taste folgendermassen belegen:
`bind TASTE say_team "‘I’m at #l with #h."’`

Im Chat wird dann je nach Situation z.B. folgendes dargestellt: „I’m at the bird’s house with 80/150“

Implementierung: Jeder Chat String durchläuft `G_Say()`. Dort wird `ReplaceChatTokens()` eingefügt, um den String zu parsen, den Platzhalter zu entfernen und die entsprechende Information einzufügen. Anschliessend verfährt `G_Say()` mit dem String wie gehabt, er wird an alle Adressaten gesandt.

3.3 Waffeneffekte

Beschreibung: Bei zwei Waffen wurde die Sichtbarkeit der Projektile verbessert: die Waffe Splasher erzeugt einen durchgehenden Strahl dessen Lebensdauer durch die Konsolenvariable `cg_splasherTrailTime` gesetzt werden kann. Die Waffe BubbleG erhält vergrösserte und leuchtende Projektile, dieser Effekt lässt sich über die Konsolenvariable `cg_altBubbleg` an- und abschalten.

Implementierung:

BubbleG: Jedes Projektilmodel wird entlang der Raumdimensionen mit dem Faktor 3 skaliert, allerdings nur Clientseitig, so das die Kollisionserkennung auf dem Server unbeeinflusst bleibt. Eine Lichtquelle wird am Ort des Projektils eingefügt.

Splasher: Bei jedem Splasher Schuss wird ein Event des bisher unbenutzen Typs `EV_SHOTGUN` an die Clients gesendet, die entsprechende Handlerfunktion erzeugt das Schweifobjekt und lässt es mit der Zeit transparenter werden.

3.4 Team Overlay

Die vorhandene Team Statusanzeige enthält nun ein zusätzliches Feld mit der Anzahl an Spray Munition, ein taktisch wichtiger Wert im Spieltyp „Spray your Color“. In anderen Spieltypen ändert sich die Anzeige nicht. Eine Konsolenvariable `cg_teamoverlaySize` wurde eingeführt, mit der sich die gesamte Statusanzeige skalieren lässt, zulässig sind Werte zwischen 8 und 20.

3.5 Show Netsettings

Das Client Kommando `players` erzeugt in der Konsole eine Liste mit Verbindungsdetails für alle Spieler. Folgende Werte werden ausgegeben: Client Slot, Snaps, Rate, Handicap, Name

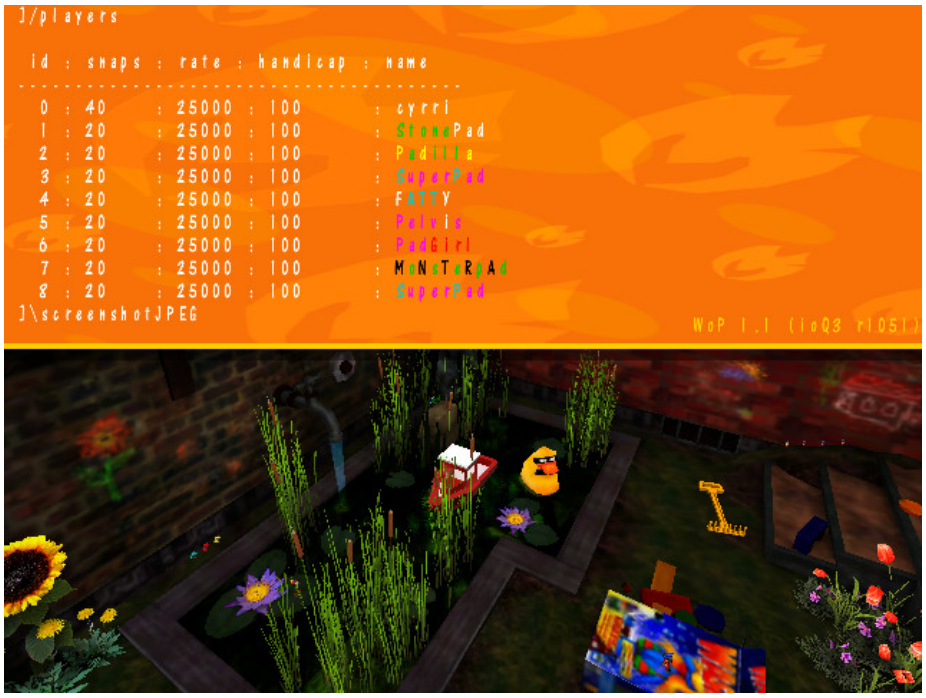

Abbildung 3: Netsettings

3.6 Votes

Es wurden folgende neue Abstimmungen eingeführt um

- Referees zu wählen, siehe Referee Status
- Items aus der Map zu entfernen, beispielsweise weil sie zu mächtig sind oder um das Spiel taktisch zu variiren
- Alle Zuschauer stumm zu schalten, so dass die Teamchat Nachrichten der Spieler nicht mit Zuschauerkommentaren gemischt werden

Details finden sich in der Befehlsübersicht.

3.7 Crosshairs

Dem Spieler stehen drei neue Crosshairs zur Auswahl. Diese können entweder über das Menü (Setup -> Options -> Crosshair) oder direkt per Konsolenvariable `cg_drawCrosshair` gewählt werden.

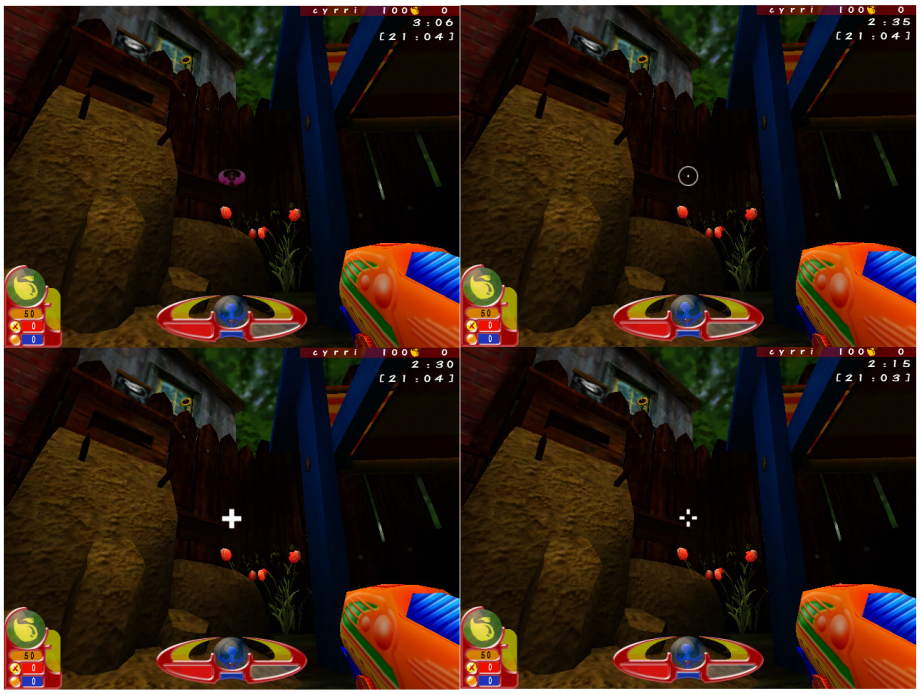

Abbildung 4: Crosshairs

3.8 Referee Status

Beschreibung: Referees sind eine Zwischenstufe zwischen dem Betreiber des Servers, der das Recht hat jegliche Einstellung des Servers zu verändern, und den Spielern, die nur sehr wenig an der Konfiguration des Servers ändern könnnen. Ein Spieler kann nur per Abstimmung zum Referee ernannt werden. Wird der Spieler gewählt, so stehen ihm von nun an eine Menge von Befehlen zur Verfügung, beispielsweise zum Entfernen von Spielern, Beenden der Aufwärmphase und so weiter. Eine vollständige Liste findet sich in der Befehlsübersicht.

Implementierung: Als Basis diente die shrubbot Implementierung von Tony J. White für das Spiel Tremulous [7]. Shrubbot war ursprünglich eine Administratoren - Mod für das Spiel Enemy Territory. Administratoren haben dort verschiedene Rechteklassen und werden anhand ihrer eindeutigen UserID automatisch erkannt. Da es bei World of Padman keine UserIDs gibt, steht dem Client keine effektive Methode zur Verfügung, sich zu authentifizieren. Daher entschieden wir uns, nur die schwachen Administratorrechte des shrubbot zu übernehmen und an die Stelle des automatisch erkannten Administrators einen gewählten Referee zu setzen.

3.9 Speclock

Der Befehl `speclock` wurde eingeführt, mit dem das Verfolgen des Spielers durch Spectators verhindert werden kann.

3.10 Scoreboard

Das Scoreboard enthält in allen Team Spieltypen je eine Abschlusszeile pro Team. Diese ist farblich hervorgehoben und enthält den Punktestand des gesamten Teams und den durchschnittlichen Ping des Teams.

3.11 AutoScreenshots, AutosStats

Nach Abschluss des Spiels lässt sich sowohl das Erstellen eines Screenshots des Scoreboards als auch die Ausgabe der Scoreboardwerte in eine Datei automatisieren. Dazu dienen die Konsolenvariablen `cg_autoScreenshot` und `cg_autoStats`. Die erstellten Dateien finden sich in den Ordnern *screenshots* und *stats* und folgen dem Namensschema „JahrMonatTag_StundeMinuteSekunde_Spielername_Mapname_Gametype.jpg“ bzw „*.txt“.

Solche Screenshots des Abschluss Scoreboards werden üblicherweise bei Turnieren als Ergebnismeldung des Spieles verlangt, daher ist es sinnvoll sie automatisch erstellen zu lassen.

[7] http://et.tjw.org/etpub/docs/#shrubbot

Abbildung 5: AutoScreenshot

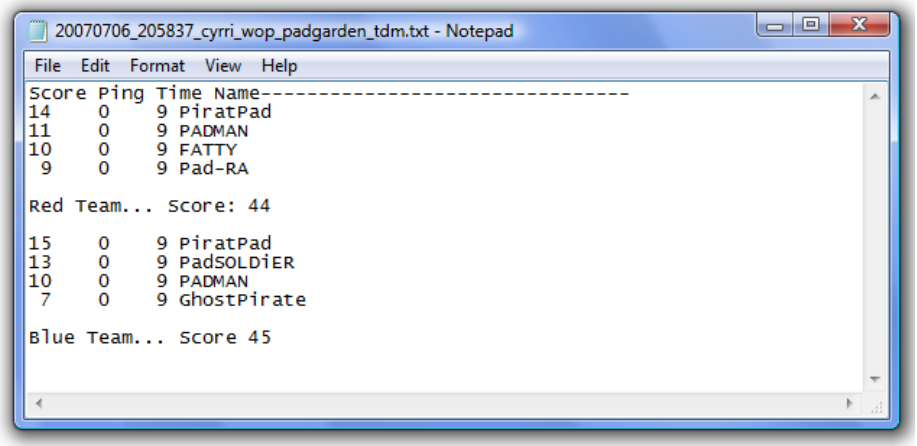

Abbildung 6: AutoStats

3.12 Health und Armor Anzeige

Der aktuelle Health und Armor Status können direkt über dem Crosshair eingeblendet werden. Zielt man mit dem Crosshair auf andere Spieler, so kann man sich deren Namen anzeigen lassen. Dies lässt sich über die Konsolenvariable `cg_drawCrosshairNames` an- und abschalten. In der Competition Mod werden zusätzlich zum Namen noch die Werte für Health und Armor angezeigt, so dass man die wichtigsten Statuswerte seiner Mitspieler sofort im Blick hat.

Abbildung 7: Health-, Armoranzeige über Crosshair

4 Befehlsübersicht

4.1 Konsolenvariablen

Befehl	Parameter	Beschreibung
cg_teamoverlaySize	8 - 20	Skalieren der Statusanzeige
cg_splasherTrailTime	X	Waffe Splasher, Lebensdauer des Strahls X Millisekunden
cg_altBubbleg	1 0	Waffe BubbleG erhält vergrösserte und leuchtende Projektile
cg_drawCrosshair	1 - 12	Verändern des Crosshair
cg_enemyModel	padman/default padman/glow monsterpad/default monsterpadglow fatpad/default fatpadglow padlilly/default padlillyglow padgirl/default padgirlglow piratepad/default piratepad/glow	gegnerischem Team ein einheitliches Model zuweisen
cg_enemyColors	white yellow cyan magenta green red blue black	Farbe des Glowskin

4.2 Client commands

Befehl	Parameter	Beschreibung
players		erzeugt in der Konsole eine Liste mit Verbindungsdetails für alle Spieler. Folgende Werte werden ausgegeben: Client Slot, Snaps, Rate, Handicap, Name
speclock		erlaubt/verbietet es Zuschauern, dem Spieler zuzuschauen
callvote referee	*playername* *slotID*	erhebt den bezeichneten Spieler zum Spielleiter (siehe Referee Commands)
callvote disable	imperius betty	schaltet das bezeichnete Item (auch Waffen) ab, so dass diese nicht mehr aufgesammelt werden können
callvote g_mutespecs	0 1	setzt die Server Variable g_mutespecs, mit der Zuschauer stumm geschaltet werden können

4.3 Referee commands

Befehl	Parameter	Beschreibung
!help		zeigt alle verfügbaren Referee Commands an
!kick	*Playername*	entfernt den bezeichneten Spieler vom Server
!putteam	*Playername*	weist den bezeichneten Spieler einem Team zu
!map	*Mapname*	beendet das Spiel und lädt die bezeichnete Map
!mute	*Mapname*	schaltet den bezeichneten Spieler stumm
!unmute	*Mapname*	hebt die Wirkung von !mute auf
!allready		versetzt alle Spieler in den „Bereit“ Zustand
!time		gibt die Serverzeit aus
!nextmap		beendet das laufende Match und starten die nächste Map entsprechend der Server config
!restart		Neustart des laufenden Matches
!lock	*Team*	verbietet es Zuschauern, dem bezeichneten Team beizutreten
!unlock	*Team*	hebt die Wirkung von !lock auf
!cancelvote		ein laufender Vote wird abgelehnt und beendet
!passvote		ein laufender Vote wird angenommen und beendet

5 Code Änderungen

Bei den folgenden Quelltexten handelt es sich lediglich um ausgewählte Beispiele. Insgesamt wurden ca. 2500 Zeilen Code in ca. 30 Dateien geändert bzw. neu geschrieben. Eine Übersicht der kompletten Änderungen findet sich unter:

http://www.andesign.de/wop/changes.htm

5.1 Implemetierung Speclock

wop-gamecode/code/game/g_cmds.c

```
void Cmd_Speclock_f(gentity_t* ent){
 int i;
 if(ent->client->speclock){
  ent->client->speclock = qfalse;
  trap_SendServerCommand(ent-g_entities,
  va("print \" Following you is now ^2disabled^7 for
      spectators.\n\"") );
  // find all client that spec ent atm
  for ( i = 0 ; i < level.maxclients ; i++ ) {
   if ( level.clients[i].pers.connected != CON_CONNECTED
      )
    continue;

   if( level.clients[i].sess.sessionTeam !=
      TEAM_SPECTATOR )
    continue;

   //ent->client->ps.clientNum
   if( level.clients[i].sess.spectatorClient == ent -
      g_entities){
    //ent->client->sess.spectatorClient = -1;
    StopFollowing( &g_entities[ level.clients[i].ps.
       clientNum ]  );
   }
  }
  return;
 }
 ent->client->speclock = qtrue;
 trap_SendServerCommand(ent-g_entities,
 va("print \" Following you is now ^2enabled^7 for
    spectators.\n\"") );
}
```

5.2 Implemetierung AutoScreenshot

wop-gamecode/code/cgame/cg_servercmds.c

```
static void CG_AutoScreenshot( void){
         char gametype[30];
       char uniqueStr[1024];
           qtime_t         now;
           char* nowstr;

           // gametype to string
           if( cgs.gametype == GT_FFA)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "ffa");
           else if( cgs.gametype == GT_TOURNAMENT)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "duel");
           else if( cgs.gametype == GT_SINGLE_PLAYER)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "solo");
           else if( cgs.gametype == GT_SPRAYFFA)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "sycffa");
           else if( cgs.gametype == GT_LPS)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "lps");
           else if( cgs.gametype == GT_TEAM)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "tdm");
           else if( cgs.gametype == GT_CTF)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "ctf");
           else if( cgs.gametype == GT_SPRAY)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "syctdm");
           else if( cgs.gametype == GT_BALLOON)
                   Com_sprintf( gametype, sizeof(gametype
                      ), "bb");
           else Com_sprintf( gametype, sizeof(gametype),
              "unknownGT");

           // time to string
           trap_RealTime(&now);
           nowstr = va( "%04d%02d%02d_%02d%02d%02d",
```

```
                                                1900 + now.
                                                   tm_year,
                                                1 + now.tm_mon
                                                   ,
                                                now.tm_mday,
                                                now.tm_hour,
                                                now.tm_min,
                                                now.tm_sec );

        Com_sprintf(uniqueStr, sizeof(uniqueStr), "%s_
           %s_%s_%s", nowstr, cgs.clientinfo[ cg.snap
           ->ps.clientNum ].name, cgs.shortmapname,
           gametype);


        trap_SendConsoleCommand( va("screenshotJPEG %s
           ", uniqueStr) );

        if( cg_autoStats.integer ){
                CG_WriteStats( uniqueStr );
        }
}
```