EQDKP PROJECT

Relazione Finale

Corso di Tecnologie Open Source.
Docente: Dott. Ing. Luigi Bellio.

Emanuele Righetto

06/07/2009

Sommario

Introduction

Prima di iniziare la relazione vera e propria è necessario introdurre il panorama in cui l'applicazione è stata sviluppata e per il quale si intende che venga usata.
Da anni ormai si è sviluppata nel mondo una tendenza dei videogiocatori ad utilizzare la rete internet per migliorare la qualità della loro esperienza di gioco.
Tali utenti, detti netgamers, hanno visto la loro possibilità di scelta nel panorama dei videogiochi online crescere rapidamente negli ultimi anni, e spaziare dai sparatutto (i cosidetti FPS), agli strategici (sia in tempo reale che a turni), agli arcade e per finire i cosiddetti MMORPG, ossia i Massive Multiplayer Online Role Play Games.
Quasi tutti questi giochi RPG vengono ambientati in realtà fantasy o mitologiche e addentrano il giocatore in epici scontri tra razze diverse di esseri viventi, dagli orchi alle fate, dai nonmorti agli elfi, dagli umani ai nani e molte ancora.
Nel 2002 la Blizzard Entertainment® ha pubblicato un gioco che sarebbe negli anni diventato senza dubbio (e che è ancora) il più giocato e in definitiva il migliore MMORPG ad oggi presente sul mercato: World of Warcraft.
Il gioco si basa su un background già esistente (Warcraft) di cui fanno parte diversi libri e tre giochi (RTS: real time strategy) già pubblicati dalla stessa software house.
World of Warcraft (da ora chiamato WoW) è un fantasy in terza persona in cui ogni giocatore impersona un eroe di una razza (attualmente sono 10 le razze disponibili) e di una particolare classe (anche queste 10, con varie combinazioni possibili in base alla razza scelta). Snocciolare le peculiarità di questo gioco sarebbe assai lungo e senza dubbio fuori tema rispetto alla relazione in oggetto, per cui verranno date solo alcune nozioni rilevanti e inerenti all'applicazione che poi andremo ad analizzare.
Scopo ultimo (o quantomeno principale) del gioco è quello di acquisire armature, equipaggiamenti ed armi di livello sempre superiore per migliorare le abilità del proprio personaggio (resistenza, stamina, mana, forza, agilità, poteri magici ecc..). Migliore è la qualità complessiva dell'equipaggiamento di un personaggio, maggiore è la sua capacità di affrontare e sconfiggere altri giocatori o altri npc (not-playing character, cioè esseri o mostri gestiti dal computer e non da altri giocatori).
Per acquisire questi oggetti è spesso fondamentale organizzarsi in più giocatori per affrontare dungeon insieme e sconfiggere mostri che da soli non sarebbe possibile affrontare. Attualmente l'organizzazione sociale di WoW si basa sulle gilde, ossia clan di persone che si uniscono sotto uno stesso stemma e creano spesso un rapporto di collaborazione per scambiarsi materiali, oggetti e soprattutto organizzare i cosiddetti "raid" che altro non sono che un gruppo di giocatori che affrontano in maniera coordinata un determinato dungeon. Sconfiggendo mostri (i cosiddetti "boss") all'interno di questi dungeon si ottengono oggetti rari o epici fondamentali per creare le tanto agoniate armature o armi.
Ma a questo punto si crea un non trascurabile problema organizzativo.
Come e a chi, tra i partecipanti al raid, assegnare il bottino di guerra? Spesso un'arma o un oggetto raro servono a diversi giocatori e senza un'adeguata gestione questo diventa un grosso problema ed è stato non poche volte causa di liti e scioglimento di gilde.
È a questo punto che entra in gioco il software che andremo ad analizzare.
Personalmente ho scoperto questa applicazione nel mio ruolo di webmaster della mia gilda, e dopo un periodo di utilizzo come normale utente ho pensato di usare la mia esperienza in programmazione per apportare alcune modifiche e miglioramenti al programma, oltre che sviluppare un piccolo plug-in per esigenze particolari della mia gilda. In questo modo sono entrato a far parte per un periodo del team di sviluppo ufficiale dell'applicazione e ho avuto modo di entrare in contatto per la prima volta con una comunità open.

VISION

Cosa sono i DKP?

DKP, acronimo per Dragon Kill Points, è un concetto ideato originariamente da Alan Thott della gilda Afterlife. Questi punti vengono assegnati ad ogni membro della gilda ogni qualvolta questo partecipa ad un raid.
Il valore corrente di DKP di ogni player riflette la sua priorità al momento dell'assegnazione dei loot. Nel momento in cui un player riceve un oggetto, esso perde una certa quantità di punti che rappresenta il valore stimato dell'oggetto ricevuto.
Questo sistema di punteggi permette un confronto obiettivo tra diversi membri della gilda al momento dell'assegnazione dei loot, tenendo conto dell'attendance (cioè la partecipazione del player alle attività della gilda) e dell'history dell'assegnazione di tutti gli oggetti precedenti nelle attività della gilda.

Cos'è EQdkp?

EQdkp è un sistema DKP Open Source creato per far fronte ad una necessità contingente. Molte gilde (di diversi MMORPG) vogliono utilizzare un sistema di assegnazione di punti per i loro raid, ma non hanno la volontà o il tempo o la capacità di creare un loro programma per la gestione di questi punti e le rispettive classifiche.
L'obiettivo principale di EQdkp è di rendere più facile per i guild leaders la decisione su come distribuire i loot (i bottini di guerra di cui parlavo prima) ed allo stesso tempo avere un sistema facile e veloce da mantenere.

Il progetto originario si è sviluppato sotto forma di una applicazione closed-couce proprietaria per un'unica gilda (EQ appunto, gilda americana), ma ci si è resi conto che un cambio di strategia ed un rilascio sotto forma di progetto open-source avrebbe portato (come è stato) al più rapido sviluppo di un'applicazione completa per la partecipazione di contributors di molte altre gilde, e al contempo alla possibilità per ogni gilda di personalizzare la propria installazione modificando una porzione relativamente piccola di codice sorgente, senza intaccare la struttura generale dell'applicazione.
L'applicazione, sviluppata in linguaggio PHP e supportata da una connessione ad un database MySQL, fornisce un front-end web per la gestione delle partecipazioni ai raid dei membri della gilda, delle assegnazioni dei loot, delle classifiche e sostanzialmente dell'history delle attività rilevanti della gilda.
Inoltre, tramite quello che è il principale plug-in (divenuto ora parte integrante del progetto) fornisce un calendario delle attività future programmate dalla gilda e permette a tutti i membri autorizzati di confermare o meno la loro presenza a tali attività, permettendo agli officer (lo staff dirigente della gilda) di organizzare in anticipo setup e composizione dei raid e di comunicarlo al resto della gilda attraverso il sito, senza quindi la necessità per un player di dover entrare in gioco per conoscere il suo status (cosa che molti non possono fare pur stando davanti al computer, soprattutto chi lavora).
Altri plug-in permettono di visualizzare diverse statistiche o dati rilevanti, come news inserite dagli amministratori, il progress della gilda, il costo prestabilito degli item che possono essere assegnati ai player ed eventuali precedenze (rispetto a vari fattori come classe per personaggio, grado all'interno della gilda, percentuale di attendance ecc..).
La natura modulare dell'applicativo permette inoltre due personalizzazioni sicuramente apprezzate con un relativamente basso sforzo di sviluppo: la localizzazione nella lingua preferita dell'utente (le lingue già presenti sono inglese, francese e tedesco) e la personalizzazione del tema grafico (sia tramite CSS che tramite modifiche alle pagine html che fanno da template per il PHP).

MARKET

Come la maggior parte dei progetti open - source, parte dei contributi economici per lo sviluppo del programma vengono da piccole donazioni di privati, e data la natura relativamente piccola del progetto (il team di sviluppo, nel periodo in cui io ne facevo parte, era composto da 4 core developer e 3 external developers) tutti i contributors lavorano gratuitamente.
Lo stesso vale per quelli che vengono chiamati "casual contributors" cioè utenti dell'applicazione che possono sviluppare alcune modifiche (anche non molto rilevanti, ma comunque sempre utili) in base alla loro esperienza d'uso.

Tuttavia i team leader hanno attivato un interessante canale alternativo per racimolare fondi per il progetto.
L'idea è di offrire al webmaster di una gilda che voglia usare il sistema la possibilità di usufruire di un host web con preinstallato sia il sistema base che la quasi totalità di plug-ins, in modo da risparmiare diversi problemi ad un webmaster poco esperto: la ricerca di un domain name e di un host che supporti PHP e MySQL, l'installazione sia dell'applicazione core che di tutti i plug-ins e la configurazione iniziale di tutto il sistema.
Tale offerta è disponibile sia in versione Lite che in versione Plus. Le due installazioni si differenziano per il numero di funzionalità e plug-ins installati (ovviamente maggiori nella Plus).
Il prezzo rispetto ad un normale hosting non è eccessivamente superiore e ultimamente sono state sviluppate soluzioni di hosting più complete e appetibili che comprendono anche server TeamSpeak (un sistema di voice chat largamente utilizzato negli mmorpg) e un forum.
L'idea di maggiorare di poco il costo di un normale host ma far trovare un applicativo già funzionante e che non richiede nessuna competenza informatica per essere utilizzato viene apprezzata da un numero sempre maggiore di utenti.

STORY

L'applicazione non presenta (o comunque non rende nota) una storia particolarmente complessa.
L'idea iniziale di sviluppare questo progetto è stata dello stesso ideatore dei DKP, Alan Thott (che giocava a Everquest) per un utilizzo privato della propria gilda. Due considerazioni importanti, cioè la considerazione sul numero di utenti del gioco e la nascita di nuovi mmorpg concorrenti, hanno portato il team di sviluppo a decidere di rilasciare il codice sorgente sotto forma di progetto open - source (garantendosi quindi nuovi contributors sia da altre gilde del gioco che da altri giochi) e di parametrizzare alcuni parametri del gioco in modo da renderlo compatibile con diversi giochi.
Attualmente il programma è infatti compatibile con EverQuest, EverQuest 2, World of Warcraft, Vanguard, e Dark Age of Camelot. La sua natura modulare permette tuttavia di aggiungere altri giochi in maniera relativamente facile e veloce senza dover riscrivere porzioni significative di codice.
È tuttavia con World of Warcraft che l'applicazione ha trovato il suo maggior successo ed utilizzo, fatto dovuto sicuramente all'elevato numero di players di questo gioco (stime recenti attestano 11 milioni di account attivi nel mondo).
Non c'è da stupirsi quindi che attualmente la maggior parte degli sviluppatori (sia dell'applicazione core che dei plug-ins) stiano sviluppando features fortemente legate a WoW.

È mia opinione che l'applicativo abbia raggiunto la sua piena maturità già da almeno un anno e che il lavoro attuale sia soprattutto di bug-fixing e di ottimizzazioni di alcune funzioni comunque presenti.
È interessante tuttavia notare come la parte "non ufficiale" della community di sviluppo stia lavorando soprattutto sull'integrazione del software con altri applicativi, soprattutto forum e board applications, per permettere all'utente finale (presumibilmente un webmaster e quindi in sequenza gli utenti del sito) di creare un sito completo per la gilda che includa un forum, un sistema dkp, ed un portale di accesso il più possibile integrati sia dal punto di vista della grafica che dal punto di vista della gestione degli utenti e delle informazioni (un solo utente per forum e dkp ed una singola autenticazione, per esempio, oppure visualizzazione nel forum di dati riguardanti i DKP, o altro ancora).
Questa sembra in effetti essere la prossima sfida che il sistema potrebbe decidere di intraprendere in maniere ufficiale.

License

Il progetto, come già detto nasce sotto forma di closed - source, ma ben presto capisce che può essere vantaggioso sia per il progetto che per la comunità dei giocatori di MMORPG trasformare il progetto in open - source.
Attualmente il progetto viene rilasciato sotto licenza GNU GPL v2
L'adozione di questa licenza permette a chiunque voglia (e ovviamente ne sia in grado) di acquisire ed utilizzare liberamente il software in questione, anche senza accettare espressamente la licenza.
È tuttavia necessario accettare i termini della licenza per essere autorizzati a modificare e ridistribuire copie (modificate) del programma, e in questo caso il programma modificato deve essere rilasciato sotto la medesima licenza del codice originale oppure con sotto una delle licenza con essa compatibili. Inoltre, in caso di modifiche parziali a file sorgenti originali, è necessario mantenere integro il riferimento al detentore dei diritti legali originali e apporre specifiche indicazioni sull'autore e la data delle modifiche.
Inoltre viene richiesto che al momento della redistribuzione di applicazioni modificate (a partire da codice sotto questa licenza) venga offerta sempre la possibilità di scaricare il codice sorgente del programma, e opzionalmente anche in versione binaria.

Quest'ultima parte è, per l'applicativo esaminato, superflua vista la natura del programma (PHP ed SQL) scritto in linguaggi interpretati e quindi in assenza di un atto di compilazione vero e proprio.

Menzione parte meritano i plug-ins. Il sistema permette infatti lo sviluppo di plug-ins esterni da aggiungere all'applicazione. La licenza GPL richiede che l'intero programma sia rilasciato sotto GPL. Essa permette comunque a questi moduli di essere rilasciati sotto una licenza compatibile con GPL, anche in forma meno stretta (per esempio LGPL).

Per mia esperienza personale, tutti i plug-ins ufficiali e non ufficiali di cui sono a conoscenza sono stati rilasciati sotto GPL.

Business Model

L'analisi del Business Model non è semplice da eseguire soprattutto dall'esterno.
Il team di sviluppo non rilascia dichiarazioni in tal senso e non sembra essere particolarmente interessato a formalizzare la maggior parte degli aspetti che riguardano questo punto.
La mia esperienza personale all'interno del progetto, come external developer (la spiegazione della suddivisione dei ruoli verrà spiegata in seguito), non mi ha dato in questo senso particolare indicazioni che già non si potessero dedurre dall'esterno del team.
È possibile cerca di raggruppare alcuni concetti chiave seguendo lo schema di Business Model proposto da Osterwalder e studiato a lezione.

Infrastructure:
- *Core Capabilities*: la natura relativamente piccola del progetto e del team non richiedono a mio avviso grosse capacità organizzative per gestire il processo di sviluppo.
- *Partner Network*: non è chiaro, ma sembra che siano state strette alcune alleanze con alcune piccole società di hosting web per fornire un servizio di host in cui sia già presente e configurato il sistema.
- *Value configuration*: come spiegato nel capitolo "Vision" la creazione di un sistema non banale di gestione di punteggi e classifiche non è alla portata di tutti sia per motivi tecnici che per mancanza di tempo (ricordiamoci che siamo all'interno di una situazione ludica che già di per se tende ad assorbire molto del tempo libero dell'utente). Questo applicativo permette in modo relativamente facile e veloce di installare e configurare sistema stabile, testato, largamente utilizzato nell'ambito e che richiede un tempo minimo di manutenzione.

Offering:
- *Value Proposition*: EQdkp è stato in assoluto il primo sistema di gestione automatica dei DKP utilizzabile per (quasi) tutti i più importanti MMORPG basati su raiding guild. Questo fattore ha spronato la maggior parte delle gilde di alto livello ad adottare questo sistema preferendolo a metodi manuali e dispendiosi (a livello di tempo) e di conseguenza ha indotto molte gilde di medio livello ad uniformarsi "imitando" le top guild.
Il modello di applicazione e la logica di distribuzioni di punti che stanno alla base di questo progetto gli hanno permesso di resistere alla concorrenza di altri applicativi che sono stati sviluppati in seguito (alcune volte tentando un'imitazione, altre stravolgendo completamente alcune idee) tanto che oggi si è imposto fortemente e senza dubbio il più diffuso e completo.
Inoltre è iniziata recentemente lo sviluppo di alcuni "hack" (modifiche al codice) che permettono al sistema di interagire e in alcuni casi anche di integrarsi in maniera efficacie con altri applicativi affini, soprattutto con forum e portali. Questo fatto è facilitato dalla natura modulare del progetto e dall'uso di tecnologie (PHP e MySQL largamente condivise in questo ambito) e porta l'utente finale ad avere un portale web altamente personalizzabile e integrato tra tutte le sue componenti.

Customers:
- *Target Customer*: l'utente "del progetto" è solitamente un membro di alto profilo della gilda, spesso con nozioni di informatica (o quantomeno di web) che si assume l'incarico di approntare un sito web per i suoi compagni e di installare in tale sito il sistema di gestione di DKP. Successivamente vi è un altro utente, l'utente finale, che è colui che utilizza praticamente l'applicazione consultando le classifiche e comparando i dati che vengono messi a disposizione.

- *Distribution Channel*: il canale di distribuzione unico è intenet. Il progetto è sia scaricabile in versione .zip direttamente dal sito (l'ultima release) sia recuperabile attraverso l'svn messo a disposizione (ne parleremo più avanti)
- *Customer Relationship*: la leadership del progetto ha predisposto un forum nel quale chiunque (previa registrazione) possono esprimere opinioni, idee, segnalare problemi e fornire risposte a domande poste da altri utenti. Il forum è diviso in sezioni, tra le quali due sono moderate e nelle quali il team di sviluppo interviene più spesso per rispondere agli utenti

Finances:
- *Cost Structure*: Il progetto è open - source e il team di sviluppo lavora gratuitamente. Inoltre tutti gli strumenti di sviluppo e comunicazione sono stati approntati con applicazioni open (e quindi sostanzialmente senza spesa) o appoggiandosi a CDE (Collaborative Development Environment) gratuiti (in particolare google: http://code.google.com)
- *Revenue*: Come già spiegato nel capitolo "Market", come per la maggior parte dei progetti open anche questo si basa su donazioni che servono soprattutto a sostenere i costi di hosting del sito e del forum.
Tuttavia si è recentemente aperta una strada interessante nel recupero di fondi che prevede l'alleanza con alcune società di hosting, per fornire all'utente un servizio ancora più evoluto: uno spazio web (con relativo domain name di secondo livello) in cui è già installato e configurato il sistema base con tutti i plug-ins più importanti. La versione Plus di questa offerta presenta addirittura plug-ins aggiuntivi non disponibili per la comunità e sviluppati appositamente. Questo da un lato permette veramente a chiunque di attrezzare un sito rapidamente e senza alcuna necessità di competenza tecnica e dall'altra di ricevere assistenza pagando un sovrapprezzo modesto su quello che sarebbe il costo del singolo hosting.

Development Process

Non ho avuto modo di lavorare all'interno del team nelle prime fasi di sviluppo del software ne di discutere questa questione con gli altri membri del team.
Cercherò di fare un'analisi del processo di sviluppo attuale del progetto, considerando però che si tratta di un'applicazione già stabile e matura, che richiede più che altro attività di bug-fixing e ottimizzazione.
Il modello che più si avvicina a quello utilizzato dai dev (tra quelli analizzati durante il corso) è il *Rapid Application Development*, che è riconoscibile da alcuni elementi:
- Tentativo di ridurre alcuni rischi inerenti al progetto dividendolo in parti più piccole, strategia adottata attraverso la modularizzazione della grafica, dei plug-ins e del supporto per lingue e giochi diversi.
- Prorotyping: ogni modifica funzionante al sistema viene caricata nel sistema di versionamento e verificata dagli altri dev. Se la modifica convince e non si rilevano anomalie viene confermata altrimenti si riparte da una versione precedente del modulo.
- Attenzione principalmente nel seguire gli interessi dell'utenza e minore importanza data all'eccellenza della programmazione e della progettazione. Questo si evince da alcuni aspetti di progettazione (il database non è in forma normale, il codice spesso non è ottimizzato..) ma anche dalla continua attività di aggiornamento del progetto rispetto a patch (o espansioni) o modifiche ai giochi supportati, per adeguarsi rapidamente al cambiamento delle esigenze degli utenti.
- Produzione di documentazione necessaria per l'eventuale sviluppo futuro (anche di altri utenti) di plug-ins compatibili e funzionanti.

- Alta partecipazione degli utenti nelle fasi di "analisi dei requisiti" (vedremo in seguito) e di test dell'applicazione, con considerazione durante tutto il processo di sviluppo delle segnalazioni.

Ogni ciclo di rilascio prevede, da quando sono riuscito a capire, alcune fasi che però non sono strettamente pianificate e possono subire variazioni anche profonde per decisione dei dev, fatto comunque facilmente gestibile visto l'esiguo numero di persone coinvolte.
- *Analisi dei Requisiti:* durante questa fase vi erano due attività principali: individuare nuovi bug (o catalogare le segnalazioni di bug esistenti) e individuare nuove feature da apportare. Questa seconda fase era diretta conseguenza delle segnalazioni e delle richieste pervenute dagli utenti, soprattutto attraverso il forum, che chiedevano patch al programma o funzioni specifiche (come la parametrizzazione di alcuni indici delle classifiche). Se queste richieste venivano considerate di largo interesse il team metteva in programma la loro inclusione nella prossima release prevista.
- *Analisi degli aggiornamenti*: questa fase si è resa necessaria dal fatto che i giochi supportati ricevono, con scadenze più o meno lunghe, patch o espansioni dalle software house e che questo implica talvolta modifiche e aggiunte nelle strutture gestionali del programma. Solitamente si cerca di pianificare l'uscita delle major release poco tempo dopo (due o quattro settimane) l'uscita di espansioni importanti in cui si prevedono questi cambiamenti. Altre beta release possono essere rilasciate nel frattempo.
- *Progettazione*: questa fase era quasi assente. Il team di sviluppo decideva l'assegnazione dei vari lavori da fare (individuati nelle fasi precedenti) tra i vari membri ed ognuno decideva autonomamente come eseguire le modifiche. In caso di sovrapposizioni tra moduli o possibili dipendenze tra lavori, semplicemente si decideva di aspettare che il modulo più importante fosse completato e successivamente si iniziava a lavorare sui moduli dipendenti.
- *Programmazione*: questa era la fase che occupava la maggior parte del tempo. Ogni dev implementava le funzioni o le modifiche stabilite ed eseguiva un commit nell'svn. In caso di commit possibilmente critici per modifiche che potevano influenzare altri moduli si provvedeva a creare un branch ed eseguire li il commit. Successivamente i dev dei moduli dipendenti interessati tentavano un commit su quel branch. Se il commit portava ad uno stato consistente il branch veniva riportato nel trunk principale, altrimenti i dev coinvolti si riunivano per trovare una soluzione (sostanzialmente individuare il punto problematico e risolvere insieme quella parte di codice).
- *Test*: una volta che il ciclo era a buon punto ogni dev prendeva in carico una parte di programma da testare (diversa da quella su cui aveva lavorato precedentemente) e procedeva ai test sulla sua copia locale.
I test simulavano sessioni di uso del programma in varie condizioni per individuare bug sfuggiti durante la programmazione.
In caso di errore veniva segnalato il bug chiesto al responsabile del modulo di indagare e correggere il problema.
- *Public Test*: quando il dev team riteneva che il progetto fosse sufficientemente a prova di bug veniva rilasciata una versione beta del progetto. A questo punto la comunità si offriva di testarlo nei loro server (attratta soprattutto dalle nuove features).
Durante questa fase era possibile che gli utenti segnalassero bug o anomalie non rilevate precedentemente e i dev raccoglievano tali segnalazioni ripetendo la fase di test e correzione.
Dopo alcuni mesi l'applicazione veniva rilasciata sotto forma di major release (anche se spesso questo non coincideva con un cambio di numerazione della versione).
Il tempo medio di un ciclo di sviluppo era di un anno (cercando sempre di prestare attenzioni ad eventuali patch ai giochi, come spiegato prima). La scelta di questo periodo

di tempo è dettata dalla considerazione che spesso l'avanzamento di versione comporta per gli utenti (admin dei vari siti) un lavoro non banale ed alcune volte rischiava di causare una sospensione del servizio (del sito) di alcuni giorni in caso di errori.
Ricordiamoci che gli utenti diretti dell'applicazione sono i webmaster delle gilde di un gioco, che dedicano gratuitamente il loro tempo libero per i loro compagni di gioco.
Una frequenza maggiore avrebbe quindi portato troppo lavoro da parte degli utenti, una maggiore rischiava di non essere sufficientemente aggiornata e di restare indietro rispetto alle esigenze dei giocatori.

COMMUNITY

La comunità che questo progetto open - source ha creato è composta quasi solo esclusivamente dai webmaster delle gilde più organizzate nell'ambito degli MMORPG citati prima.
La specificità dell'applicazione è infatti incentrata sugli aspetti portanti di questi giochi di ruolo e non credo che sarebbe portabile rispetto ad ambiti diversi da quello ludico.
Io personalmente ho fatto parte per alcuni mesi del team di sviluppo del progetto e ho quindi avuto modo di conoscere un po' meglio l'organizzazione interna della community:
- *Core Developers*: al momento della mia collaborazione con il progetto era composta da 4 sviluppatori, due dei quali erano parte del gruppo originario che per primo iniziò a scrivere il codice, mentre due erano subentrati in seguito ma facevano comunque parte del progetto da molto tempo. Si occupavano dello sviluppo del core del programma, cioè l'infrastruttura di autenticazione e amministrazione e la gestione base dei punteggi e dell'history dei raid. Questa parte era abbastanza consolidata e funzionante e ormai il lavoro principale si traduceva in
- Bug - fixing: soprattutto bug segnalati dagli utenti riguardanti SQL Injection e protezione contro cross-site scripting.
- Miglioramento delle funzionalità di installazione.
- Adattamento del sistema alle feature rilasciate negli ambienti usati (PHP e MySQL)
- Supporto di nuovi giochi
- Supporto di modifiche inerenti ad espansioni per giochi già presenti.
- *External Developers*: io personalmente rientravo in questa categoria assieme ad altri due developers. Si occupano di sviluppare o mantenere i plug-ins ufficiali del progetto. In questo caso ogni developer aveva in carico uno o più plug-ins (in relazione al tipo di plug-in e al lavoro da svolgere) e si occupava di:
- Bug - fixing di bug riportati dagli utenti o dagli altri developers (in alcuni casi errori gravi sui plug-ins compromettevano il funzionamento di tutto il sistema o parte importante di esso).
- Sviluppo di nuove features del plug-in o miglioramento di quelle esistenti.
- Adattamento del plug-in (quando necessario) a cambiamenti nel core che potevano riflettersi sulla funzionalità del plug-in stesso.
Inoltre poteva succedere che un plug-in sviluppato esternamente al progetto (per esempio da qualche utente per necessità particolari) venisse segnalato dalla community come particolarmente interessante o utile. In questo caso il plug-in veniva inserito tra quelli supportati direttamente e al programmatore veniva chiesto di entrare nel team (come external) per sviluppare o mantenere la sua creazione.
- *Casual Contributos*: questa era la parte più grossa della community, ma al contempo gestita come se fosse semplicemente un utente del programma stesso.
Non erano inseriti nel processo di sviluppo, non erano consultati per cambiamenti nel progetto e non avevano accesso agli strumenti di comunicazione interni usati tra le altre

due categorie. In pratica si occupavano di:
- Sviluppare Temi e Modifiche grafiche per l'applicazione
- Sviluppare plug-ins non supportati ufficialmente (ed eventualmente proporli per il supporto) ma dovevano gestire autonomamente il rilascio (solitamente tramite il forum principale) e l'eventuale processo di feedback e bug - fixing.
- Proporre (cioè sviluppare e rilasciare) modifiche a parti del programma, solitamente nel front-end di visualizzazione grafica e non nell'infrastruttura generale del programma
- Sviluppare patch che permettessero il "bridge" dell'applicazione con altri sistemi informatici affini, in particolar modo forum e portlet. L'obiettivo di queste patch era di permettere un'interconnessione non banale tra i sistemi forum più diffusi e l'EQdkp, uniformando per esempio la gestione degli utenti e dei permessi, la registrazione degli stessi e la gestione delle sessioni autenticate, nonché un'integrazione grafica che facesse sembrare il risultato finale il più simile possibile ad un'applicazione unica e coerente.
Nell'ultimo periodo questa ultima tendenza è stata spinta parecchio dalla community e dall'utenza tanto che il team di sviluppo ufficiale sta valutando se inserire questa feature tra quelle supportate direttamente.

Information Management Tools

Come la maggior parte dei progetti open anche questo si avvale di una serie di strumenti per la gestione delle informazioni e del codice.
- ***Web Site***: il sito principale che contiene le news del progetto e i link a tutte le altre aree del progetto predisposte per la comunicazione.
- ***Mailing List***: il team di sviluppo (core e external developers solamente) hanno accesso ad una mailing list per comunicare direttamente tra di loro, non moderata ma con la richiesta di rispettare alcune regole interne sul tag dei titoli delle mail e sulla formattazione delle stesse. Questo strumento non è accessibile ne in lettura ne in scrittura dal resto della community.
- ***Forum***: è la fonte di interazione principale tra i membri di tutta la community. È moderato direttamente dai developers ma solitamente non vengono intraprese azioni di censura o ban a meno di comportamenti gravi (cosa che non mi è mai capitato di vedere). Il forum è diviso in diverse sezioni (una accessibile solo ai dev) per incanalare le discussioni in base all'argomento.
Tra le varie sezioni spuntano la sezione news in cui i dev postano le ultime novità sviluppate, le sezioni di supporto per i futuri amministratori dell'applicazione (sia per la fase di installazione che per la gestione di routine), una Kwnoledge Base, sezioni per le discussioni sui plug-in interni ed una generale per i plug-in esterni ecc..
Il forum è complessivamente abbastanza attivo, solitamente questioni di una certa importanza riguardanti aspetti critici ricevono risposte sia dalla community che dai dev all'interno della giornata, ma per contrapposizione alcuni topic riguardanti funzioni secondarie o richieste personalizzazioni possono essere lasciati in standby anche per alcune settimane. È molto raro che i dev intervengano su questioni riguardanti modifiche non supportate direttamente.
Questo strumento viene utilizzato anche dai casual contributors come mezzo per distribuire o proporre le loro modifiche e gestire i feedback. All'interno di un thread (che può essere evidenziato da qualche dev in caso di modifiche importanti o interessanti) viene rilasciato un archivio contenente il codice modificato o le istruzioni per eseguire le modifiche e la comunità può rispondere sullo stesso topic con annotazioni, richieste di modifiche o segnalazioni di bug. Spesso in questi casi sono gli stessi utenti che trovano soluzioni per i bug o le modifiche richieste, esaminando e riportando porzioni di codice.

- ***Subversion***: strumento largamente diffuso nell'ambito del controllo del versionamento del codice, è accessibile pubblicamente in sola lettura (sia tramite client svn che direttamente dal browser web) mentre è accessibile in scrittura tramite autenticazione che viene attivata nel momento in cui si entra a far parte del team di sviluppo. Attualmente solo il codice viene mantenuto versionato.
- ***Bug Tracker***: chiamato anche Issue Tracker dal CDE di google, viene usato per segnalare bug e gestirne lo stato. In realtà questo strumento, benché pubblico, viene usato quasi esclusivamente dai dev. Il resto della comunità preferisce utilizzare, forse per pigrizia o forse per inesperienza, il forum per segnalare errori o anomalie.
- ***Area Wiki***: messa anch'essa a disposizione da google, attualmente è scarsamente popolata ma fornisce comunque alcune indicazioni importanti sulle modifiche alle API e sulla creazione di plug-ins compatibili e funzionanti con il programma. Potrebbe essere ampliata notevolmente a mio giudizio ma i dev (forse giustamente) convinti che siano veramente pochi gli utenti che cercano di modificare o anche solo capire il funzionamento del programma, e in questo caso spesso dopo alcune fasi di prova entrano a far parte del team di sviluppo. Non intendono per tanto (almeno per ora) ampliare ulteriormente questo strumento.

Development Management Tools

Non è noto se la prima scrittura del codice, con il relativo carico di funzioni background da scrivere (connessione e gestione del database, gestione delle sessioni... del motore dei template...), sia stata effettuata con l'ausilio di strumenti di sviluppo.
Come esperienza personale io non mi sono avvalso nel mio lavoro di strumenti di sviluppo, se non ovviamente il set di programmi richiesto per far funzionare l'applicazione (PHP, MySQL, un server web e nella fattispecie Apache2).
Sempre personalmente, ho utilizzato un editor HTML per velocizzare alcune modifiche ai template grafici e testarle senza essere costretto a refreshare la cache del programma e ricaricare le pagine.
All'interno del team di sviluppo non c'era alcuna direttiva sull'uso di particolari strumenti.
L'unico strumento realmente condiviso è un Collaborative Development Environment, fornito da Google (Google Code) che mette a disposizione di sviluppatori un set di strumenti di coordinazione: una home page del progetto (a cui in ogni caso si rimandava alla home del sito principale) un repository svn, un bug tracker, una download area, un'area wiki.

Riferimenti e Sitografia

-World of Warcraft Europe: http://www.wow-europe.com
-EQdkp Home: http://eqdkp.com/
-EQdkp Forums: http://forums.eqdkp.com/
-EQdkp ChangeLog: http://eqdkp.com/?p=changelog
-EQdkp CDE (Google Code): http://code.google.com/p/eqdkp/