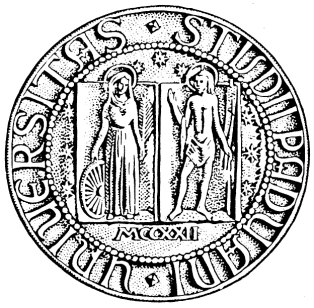

Università degli Studi di Padova

Facoltà di Scienze MM.FF.NN.

Dipartimento di Matematica Pura e Applicata
Laurea Magistrale in Informatica
Corso di Tecnologie open-source

Progetto d'esame

Analisi di progetto open-source:

PariPari

Autore: Daniele Bonaldo

Anno Accademico 2009/2010

Indice

1 Introduzione e visione

Dalla comparsa di internet numerosi nuovi servizi sono diventati disponibili per gli utenti di computer attraverso la rete. L'utente medio non si limita più alla semplice navigazione attraverso siti web, ma usufruisce di numerosi servizi offerti tramite internet, spesso contemporaneamente.

Oltre al World Wide Web, è stato possibile sperimentare nuovi modi di comunicare, prima tramite posta elettronica, poi in tempo reale, tramite chat o sistemi di VoIP, e nuove modalità per condividere files fra vari utenti, come ad esempio l'utilizzo di reti Peer to Peer create allo scopo.

Ovviamente per poter sfruttare tutti questi servizi è generalmente necessario che l'utente utilizzi numerosi programmi, spesso attivi nello stesso momento, la qual cosa può portare ad un utilizzo inefficiente della connessione disponibile, oltre al lavoro richiesto al computer in sé.

Con il diffondersi delle connessioni internet disponibili, si sono andate a creare numerose possibilità di collegarsi da postazioni pubbliche (come ad esempio in locali di ristorazione, centri commerciali, aeroporti, alberghi...), in cui però la libertà concessa all'utente è limitata ed è possibile svolgere solo attività basilari e in cui è raramente possibile installare software necessario per usufruire di tutti i servizi di cui si ha bisogno.

Il progetto PariPari nasce proprio dall'idea di raggruppare all'interno di un unica suite i programmi maggiormente utilizzati in ambito internet, fra cui programmi P2P e di Istant Messaging. Un punto essenziale evidenziato fin dall'inizio è il fatto che tale suite non deve necessitare di installazione, in modo da poter essere utilizzata in qualunque contesto, anche nel caso non si disponga di privilegi tali da permettere l'installazione di software. Altro requisito era quello di poter essere utilizzato indipendentemente dal sistema operativo sottostante e la facilità d'uso da parte dell'utente.

L'obiettivo finale è di creare una rete serverless (attualmente basata su una variante di Kademlia), che sia in grado di garantire l'anonimato dei suoi nodi, che implementi un sistema di crediti più intelligente (e transitivo) di reti come ED2K e che sia multifunzionale fornendo i più comuni servizi disponibili su Internet mantenendoli però raggiungibili e fruibili anche da computer esterni a PariPari.

2 Storia

L'idea di un progetto che soddisfasse i requisiti visti in precedenza venne inizialmente a Paolo Bertasi, all'epoca studente del corso di Laurea in Ingegneria Informatica presso l'Università degli studi di Padova. Nella sua tesi di laurea descriveva la struttura iniziale del progetto e le problematiche affrontate nella sua realizzazione.

Fin da subito ci si è resi conto che per sviluppare un progetto di tale complessità, la cui durata sarebbe stata stimabile in anni, sarebbe stato necessario il lavoro di diverse persone.

Da un progetto individuale quindi si è fin da subito creato un gruppo di ricerca, composto da diversi laureandi che poi sarebbe diventato la comunità che ad oggi prosegue nello sviluppo del progetto.

Ad oggi il progetto è portato avanti da una comunità di oltre 70 studenti divisi in 15 gruppi.

Anche i servizi offerti sono aumentati col tempo e si sono andati ad aggiungere a quelli di base, fra cui il backup distribuito, il supporto per diversi protocolli di istant messaging, il supporto per l'utilizzo di torrent e VoIP.

3 Business Model e licenza

Il progetto ha fra gli altri scopi quello didattico ed è pertanto finanziato dal Dipartimento di Ingegneria informatica (DEI) dell'Università degli studi di Padova.

Tale finanziamento permette di avere a disposizione degli sviluppatori aule per le riunioni, laboratori per lo sviluppo e i server per l'hosting del progetto.

Il DEI si occupa anche della gestione e del finanziamento del sito ufficiale del progetto e della sua wiki.

Utilizzando le risorse fornite dal dipartimento, si è scelto di non utilizzare un Collaborative Development Environments, come ad esempio Sourceforge, in quanto tutti i servizi necessari al progetto vengono forniti, previa richiesta, dall'università.

Il contributo da parte degli studenti è volontario e, ovviamente, non retribuito, quindi non c'è da aggiungere una spesa per il pagamento degli sviluppatori.

Il prodotto finale è distribuito tramite il sito del progetto in forma completamente gratuita, e vista la natura aperta al pubblico ed agli sviluppatori della rete è stato deciso di rilasciare il progetto sotto licenza GPLv3.

La scelta di tale licenza permette agli utenti del progetto di modificarlo secondo le loro esigenze, ad esempio creando dei nuovi plugin, e soprattutto permette di condividere tali modifiche.

4 Development Process

Per la dimensione del progetto e soprattutto per la composizione della sua comunità, è stato scelto di svilupparlo tramite la metodologia dell'**Extreme Programming**, adattata per alcuni dettagli al progetto specifico.

Tale metodologia, come tutte le metodologie agili in generale, considera naturali le modifiche in corso d'opera dei requisiti e conseguentemente mira a rendere flessibile lo sviluppo del progetto, rispetto ad una metodologia che preveda la stesura rigida dei requisiti sono nella fase iniziale dello sviluppo.

La flessibilità garantita da una simile tecnica è particolarmente utile in un progetto come PariPari, in cui la comunità è soggetta ad un turn-over continuo dei componenti, salvo casi particolari.

L'Extreme Programming mira a ridurre il costo delle modifiche, che come abbiamo visto sono inevitabili, introducendo una serie di valori, principi e pratiche, ma soprattutto definendo quattro attività base:

- **coding**: il codice è il prodotto più importante ed è la cosa più utilizzata per la comunicazione all'interno del progetto. Il codice dev'essere sempre documentato, ma già da solo il codice è un'ottimo mezzo di comunicazione, in quanto non può essere interpretato che in un modo solo.
- **Testing**: la fase di test è fondamentale per lo sviluppo tramite Extreme Programming, tanto che vengono costruiti i test prima di costruire il codice da testare. In PariPari si segue perfettamente questa filosofia e per uno studente c'è la possibilità di partecipare al progetto anche solo come tester, tanta è l'importanza che si da a questa fase.
- **listening**: ossia prestare attenzione alle richieste di nuove funzionalità ed a quello che viene chiesto dal business. Nel caso di PariPari ci si riferisce alle proposte di nuove funzionalità e servizi da inserire nel progetto.
- **designing**: con un progetto di tali dimensioni, procedere solo con codifica e test senza la dovuta progettazione è improponibile. Per tale ragione i team leaders ed il coordinatore del progetto effettuano delle riunioni regolari in cui viene analizzato lo stato attuale del progetto e viene programmato lo sviluppo futuro.

Per quanto riguarda i valori riconosciuti dall'Extreme Programming, PariPari li favorisce tutti e cinque:

- **communication**: la comunicazione è essenziale e ben diffusa all'interno del progetto, soprattutto all'interno di ogni singolo team.
- **simplicity**: come tecnica di sviluppo iterativo, la semplicità è d'obbligo, in quanto innanzitutto è fondamentale cominciare con la soluzione più semplice ad un problema, posticipando l'eventuale aggiunta di ulteriori funzionalità.
- **feedback**: i feedback vengono considerati moltissimo durante lo sviluppo, sia che vengano dal sistema, quindi attraverso il testing, sia che vengano dal team, per quanto riguarda le previsioni rispetto l'implementazione di una determinata funzionalità.

- **courage**: il refactory del codice è una cosa assolutamente naturale e quindi uno sviluppatore deve essere in grado di valutare quando è necessario buttare del codice per sostituirlo con del codice migliore.
- **respect**: il rispetto è fondamentale all'interno di ogni comunità, tanto più in una comunità come quella di PariPari in cui tutti i componenti sono studenti, spesso compagni di corso. Tale rispetto si traduce nell'obbligo di mantenere alto il livello di ciò che ogni componente produce in modo da non intaccare il lavoro altrui, come potrebbe ad esempio succedere pubblicando del codice che faccia fallire dei tests.

Rispetto alle pratiche proposte dall'Extreme Programming, PariPari ne adotta solo alcune. Non viene adottata ad esempio la pratica del **pair programming**, ossia la tecnica di programmazione in cui due sviluppatori condividono la stessa postazione contemporaneamente.

Viene effettuato il **test driven development**, in cui i test di unità vengono scritti prima di scrivere il codice da testare, in modo da stimolare lo sviluppatore a pensare ai vari modi in cui il codice che sta sviluppando potrebbe fallire, agendo di conseguenza per evitare tali possibilità.

La **continuous integration** garantisce, tramite l'uso di un repository SVN, che ogni sviluppatore stia lavorando sull'ultima versione del codice disponibile.

Tramite il **design improvement** è possibile migliorare la struttura del codice, tramite refactoring, rendendolo più semplice e generico.

Il codice deve essere scritto attenendosi ad uno standard e nel caso di PariPari si adotta *"The Code Conventions for the Java Programming Language"* consigliata da Sun. L'uso di uno standard nella stesura del codice è descritto dalla pratica detta **coding standard**.

Non viene adottata la pratica del **collective code ownership**. Questa pratica prevede che ogni sviluppatore sia autorizzato a modificare il codice di qualunque parte del progetto, mentre nel caso di PariPari uno sviluppatore è responsabile ed autorizzato a modificare solo ed esclusivamente il codice prodotto da sé stesso.

5 La Comunità

La comunità che presiede lo sviluppo di PariPari è composta da studenti dei Dipartimento di Ingegneria Informatica dell'Università degli studi di Padova. A seconda del livello di impegno messo a disposizione, lo studente può svolgere incarichi di testing, sviluppo, fino a gestione di un team, incaricato di portare avanti la realizzazione di un modulo del sistema.

Al momento gli studenti sono divisi in 15 teams, ognuno diretto da un team leader. I vari team leader rispondono infine al Prof. Paolo Bertasi.

5.1 Leadership

Carisma: i team leader sono generalmente studenti più anziani rispetto agli studenti che svolgono compito di sviluppatore o tester. Possono essere dei laureandi o degli studenti del corso di laurea magistrale.

Esperienza: essendo studenti più anziani, i team leader hanno una maggiore esperienza riguardo gli argomenti trattati dal team loro assegnato, senza considerare che il periodo passato da loro all'interno della comunità è maggiore rispetto a quello di un tester o di uno sviluppatore semplice.

Responsabilità: ogni team leader è responsabile di quanto prodotto dal suo gruppo, i cui componenti dovranno rendere conto a lui. I vari team leader poi rispondono al Prof. Bertasi, che si occupa della coordinazione globale, e si riuniscono periodicamente per tracciare il punto della situazione relativo allo sviluppo globale.

Disponibilità: i vari team leader hanno il compito di aiutare i nuovi arrivati nella comunità per eventuali dubbi e richieste di aiuto, dimostrando quindi anche doti di pazienza ed abilità comunicativa.

5.2 Precondizioni favorevoli

Linguaggio di programmazione scelto: il linguaggio scelto per scrivere il client è Java, per permettere la massima penetrazione della rete. Java, come è noto, rende possibile la portabilità del programma su tutte le principali piattaforme senza la necessita di ri-compilare il codice sorgente. Java, inoltre, grazie alla tecnologia Java Web Start, rende trasparente all'utente l'uso della rete. Infatti, grazie all'integrazione col browser, sarà possibile scaricare, installare, e tenere aggiornato il software in modo pressoché automatico. L'uso di Java, tuttavia, comporta una certa perdita di performance. Le operazioni più costose dal punto di vista computazionale risultano essere quelle di crittografia, ma la riduzione nelle prestazioni non influisce molto sui tempi di risposta del programma, dato che la maggior parte delle cifrazioni avviene su stream di byte dalle dimensioni molto contenute. Nonostante quindi un leggero calo nelle prestazioni, l'uso di Java promette all'utente finale, ed allo sviluppatore, un significativo incremento nella facilità d'uso e di gestione del client.

Qualità del codice sorgente iniziale: scritto per la tesi e sviluppato inizialmente da poche persone, ma fin da subito molto accurato e soprattutto documentato.

Richiesta del prodotto: l'insieme di servizi che il progetto finale dovrà garantire sono quelli maggiormente utilizzati dalla tipologia di persone che forma la comunità (studenti universitari). Questo fatto può portare altri studenti ad essere interessati all'aiutare nella realizzazione del progetto.

Innovatività: esistono alcuni programmi che permettono di usufruire di servizi di chat e file sharing, ma nessuno che permetta l'utilizzo di tutti i protocolli inclusi in PariPari nella stessa applicazione.

5.3 Promote comunity building

Modularità: uno dei punti di forza del progetto è proprio la sua modularità: alcuni studenti possono essere attirati dallo sviluppo di un singolo modulo, senza che abbiano la necessità di conoscere nel dettaglio l'intero sistema.

Riunioni: oltre alla comunicazione via email o chat, periodicamente vengono svolte delle riunioni fra i membri di uno stesso gruppo, oltre a quelle riservate ai team leaders. Tali riunioni sono attuabili durante il periodo delle lezioni, ma di difficile praticabilità durante le sessioni d'esame o nei periodi di vacanza, in cui molti studenti fuori sede risultano non reperibili fisicamente.

Task list: fra una release e la successiva viene sempre stilata una lista, per ogni gruppo, contenente le attività da svolgere e le funzioni da implementare. In tal modo uno studente interessato ad inserirsi nella comunità può vedere a cosa stanno lavorando i vari gruppi e quindi orientarsi verso l'argomento che più lo attira.

GUI: l'interfaccia utente è semplice ed intuitiva ed il client non necessita di installazione per essere eseguito, grazie all'utilizzo di Java Web Start.

6 Strumenti di Sviluppo

6.1 Information management software

Questi sono i principali strumenti usati per la gestione della documentazione e delle informazioni riguardanti il progetto:

- **Sito web**: www.paripari.it, da cui si può scaricare l'ultima release del progetto
- **Wiki**: www.paripari.it/mediawiki/index.php/Main_Page, la wiki del progetto, in cui uno sviluppatore può trovare le informazioni di cui necessita. Utilizzata anche per condividere i verbali di incontri e la programmazione per le releases future. Al suo interno sono presenti numerosi **HOWTOs**, importantissimi per uno studente appena entrato nella comunità di PariPari.
- **Gruppo su Google Groups**: groups.google.it/group/paripari, il gruppo creato utilizzando il servizio Groups di Google. Iscrivendosi a tale gruppo lo sviluppatore si può iscrivere a diverse mailing lists a seconda del team di cui fa parte.
- **Bug tracking**: www.paripari.it/bugzilla, è il portale basato su Bugzilla in cui possono essere inserite notifiche di bug e da cui possono essere poi monitorate le operazioni svolte per la gestione di tali problemi. L'inserimento di un nuovo bug può essere effettuato solo da parte di un tester approvato o da parte di uno sviluppatore. Nella fase di inserimento di una notifica di bug è necessario indicare in che area rientra l'anomalia, in modo che ne venga informato il gruppo responsabile.

6.2 Strumenti di versionamento

Per il versionamento del codice viene utilizzato un repository basato su Subversion.

Il repository SVN è diviso in cinque cartelle principali:

- ***Trunk***, che contiene il codice dell'ultima versione, anche se non ancora rilasciata sotto forma di release. Il codice dei plugin qui presente è quello dell'ultima versione stabile.
- ***Branches***, che contiene una copia di parte del progetto con una durata limitata. Viene utilizzata nel caso si necessiti di effettuare di un certo spessore modifiche nel Trunk, in questo modo è possibile testare a fondo la correttezza di tali modifiche prima di fondere il branch nel Trunk finale.
- ***Tags***, che contiene tutte le releases di ogni plugin “salvate” nei giorni in cui vengono pubblicate le releases del progetto: quando un plugin è abbastanza stabile da poter essere pubblicato in una release, viene effettuato uno snapshot della relativa sottocartella in Trunk e salvato in Tags. Su tale copia non viene più effettuato sviluppo ma viene mantenuta solo a scopo di bugfix.
- ***Jars***, che contiene i jar prodotti da ogni team per il plugin a loro assegnato. Tali jar sono salvati esclusivamente in versione stabile in modo che possano essere usati dagli altri teams.
- ***Extras***, che contiene del materiale vario di utilità allo sviluppatore, come la documentazione interna al progetto ed i verbali delle riunioni.

All'interno di ogni macro cartella, poi, il repository è diviso per moduli e per ogni modulo sono presenti quattro cartelle: **src**, **test**, **jar** e **lib**.

- In *src* è contenuto il codice sorgente del plugin
- in *test* sono contenuti i files necessari per i test
- in *jar* si trova la cartella *META-INF* che contiene il file *descriptor.xml* ed altri files necessari alla compilazione
- in *lib* sono presenti tutte le librerie da cui il plugin dipende.

6.3 Development tools

IDE: la maggior parte della scrittura del codice avviene utilizzando Eclipse, come IDE per lo sviluppo. La scelta di tale IDE, uno dei più diffusi per la scrittura di codice in Java, ha permesso di utilizzare parecchi plugin scritti per tale IDE che facilitano l'integrazione con il repositry SVN, il refactory del codice e l'attività di build automation.

Build automation: come tool di build automation si è scelto di utilizzare Ant, che si integra naturalmente con Eclipse e permette di automatizzare una serie di attività svolte durante la fase di build prima e di deploy successivamente.

Per gli studenti appena entrati nella comunità e che non hanno familiarità con Eclipse o Ant, nella wiki di PariPari sono presenti degli HOWTOs appositi che illustrano come configurare tali tools.

7 Riferimenti

- E. Peserico, P. Bertasi *Progettazione e realizzazione in Java di una rete Peer to Peer anonima e multifunzionale*. Padova, 2005.
- http://www.paripari.it/ , il sito internet del progetto
- http://www.paripari.it/mediawiki/index.php/Main_Page , la wiki
- http://groups.google.it/group/paripari , il gruppo su Google Groups
- http://java.sun.com/docs/codeconv/ The Code Conventions for the Java Programming Language