ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Seconda Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica

PROGETTARE E PROGRAMMARE UN'APPLICAZIONE WEB 2.0 E 3.0 CON AGENTI BDI, COME E PERCHÈ?

Elaborato nel corso di: Sistemi Multiagente LS

Ad opera di:
MATTIA MINOTTI

ANNO ACCADEMICO 2009–2010
SESSIONE I

PAROLE CHIAVE

Web 2.0

Web 3.0

Web Semantico

Agenti

Agenti Intelligenti

Indice

Introduzione

Il seguente elaborato, realizzato per il corso di Sistemi Multiagente LS, si pone lo scopo di motivare, descrivere e spiegare l'utilizzo di un modello di programmazione ad agenti nell'ambito delle applicazioni Web lato client. Esso è da considerarsi come la naturale continuazione della tesi di laurea triennale da me realizzata e dal titolo "Un Modello di Programmazione ad Agenti per Applicazioni Web 2.0 e 3.0"[1] e del successivo articolo scientifico "An Agent-Based Programming Model For Developing Client-Side Concurrent Web 2.0 Applications"[2] presentato presso la conferenza WEBIST di Lisbona, 2009. Rispetto ai precedenti lavori, in questo elaborato si cercherà di focalizzarsi maggiormente sulle motivazioni che hanno portato a proporre l'utilizzo di agenti nell'ambito del Web 2.0 e di agenti intelligenti nell'ambito del Web 3.0 o Web Semantico.

La trattazione sarà suddivisa in tre capitoli. Nel primo si cercherà di riproporre una veloce panoramica sui concetti fondamentali che portano alla definizione di Web 2.0 e 3.0, cercando di fare particolare attenzione a quegli aspetti che porteranno poi a motivare le scelte che verranno fatte. Il secondo capitolo sarà invece dedicato alla discussione più specifica dei pregi e dei difetti che l'utilizzo di un modello di programmazione ad agenti può apportare nello sviluppo di una applicazione Web 2.0. In esso vi sarà anche un confronto con le tecniche oggi più utilizzate in questi ambiti, come AJAX e GWT. Il terzo ed ultimo capitolo sarà infine dedicato alle implicazioni che l'uso di questo modello, ed in particolare una sua estensione che faccia uso di agenti intelligenti, può portare in relazione alla visione ed alle prospettive offerte dal Web 3.0 o Web Semantico.

Capitolo 1

Web 2.0 e 3.0

In questo capitolo si introdurranno i principali concetti legati ai contesti di Web 2.0 e Web 3.0 o Web Semantico. In particolare, si cercherà di focalizzarsi sui possibili problemi che ad oggi si devono affrontare nella programmazione di applicazioni Web di questo tipo utilizzando i modelli e le tecnologie disponibili.

1.1 Web 2.0

1.1.1 Le Definizioni

Il termine "Web 2.0" venne coniato da Dale Dougherty (general manager della divisione Maker Media di O'Reilly Media, Inc.) e Craig Cline (di Media Live International) durante la preparazione di presentazione in una conferenza che avrebbero dovuto tenere congiuntamente. Tale definizione è stata poi ampiamente dibattuta dall'editore Tim O'Reilly, fondatore della O'Reilly Media stessa ed acceso sostenitore del software libero e di numerosi movimenti open source. Egli aprì un primo dibattito sul Web 2.0 tramite il suo blog, dove definì"*il Web 2.0 è la rete intesa come piattaforma, che abbraccia e si estende su tutti i sistemi ad essa connessi; le applicazioni Web 2.0 sono quelle che riescono a sfruttare i vantaggi di una tale piattaforma: offrendo software come un servizio sempre aggiornato che migliora all'aumentare del numero di persone che lo usano (anche detto network effect), utilizzando e rimescolando i dati provenienti da sorgenti di-*

verse, offrendo i propri dati in una forma che ne permetta un facile riutilizzo da parte degli altri, creando network effect attraverso una architettura della partecipazione, andando oltre la metafora della pagina tipica del Web 1.0 in modo tale da offrire agli utenti una esperienza di utilizzo realmente ricca"[3].

Analizzando questa definizione, un primo importantissimo e naturale spunto viene dalla visione del *Web come piattaforma di sviluppo*. Il Web è una piattaforma di sviluppo rete-centrica emergente che offre un valido supporto alla creazione distribuita e collaborativa. Un buon indicatore della diffusione di questa visione rete-centrica è il numero di openAPI che sono ad ora disponibili per gestire il flusso di contenuti tra sistemi differenti. Proprio grazie a queste openAPI è possibile utilizzare l'intera Internet come API stessa per progettare e creare le proprie applicazioni. La potenza e le possibilità di riuso in questo senso sono probabilmente senza limiti. Colossi come Amazon, Ebay, Google, Yahoo, Skype e molti altri mettono a disposizione le proprie openAPI permettendo agli sviluppatori di creare e inventare nuove applicazioni che lavorano su informazioni e dati preesistenti.

Il flusso di contenuti e idee è un'altra delle importanti novità del "Web 2.0". Esso vuole infatti rappresentare quelle tecnologie che permettono alle informazioni di essere suddivise in unità che possono viaggiare liberamente da un sito all'altro, spesso in modi che il produttore non aveva previsto. Questo paradigma permette agli utenti di prendere informazioni da diversi siti simultaneamente e di distribuirle sui propri siti per nuovi scopi. Tutto questo può essere riassunto con l'espressione *decentralizzazione dell'informazione*.

E' poi molto interessante l'ultima parte della definizione data da O'Reilly, ossia la possibilità di *offrire agli utenti una esperienza di utilizzo realmente ricca*. Si comincia quindi a parlare sempre più insistentemente di applicazioni Web in grado di offrire agli utenti esperienze di utilizzo sostanzialmente paragonalibili a quelle che si possono avere con applicazioni desktop. Ovviamente, le applicazioni Web ricche portano con sè un concetto innovativo di Web. Fondamentalmente esse sono difficili da comprendere perchè portano inevitabilmente una serie di principi (drag'n drop, l'aggiornamento in tempo reale dei dati, ecc.) ben noti alle applicazioni desktop tradizionali, ma quasi totalmente sconosciuti ai puri sviluppatori Web. Per parecchio tempo, e

non si può dire che ancora oggi non sia più così, le applicazioni Web sono rimaste legate a regole particolari, quelle dettate dalle caratteristiche dei vari browser. Queste caratteristiche ora hanno raggiunto un livello tale da andare ad abbracciare le funzionalità tipiche delle applicazioni desktop.

Sebbene tutti i concetti descritti finora possano sembrare totalmente o parzialmente rivoluzionari per il Web, è cosa ormai nota che il "Web 2.0", per come lo si intende oggi, non introduce tecnologie sostanzialmente nuove. Anche se può sembrare strano, tutti gli obbiettivi che si pone questo nuovo modo di intendere il Web non sono stati la meta di una tangibile evoluzione tecnologica, bensì il frutto di uno sfruttamento diverso, talvolta oggettivamente migliore, degli strumenti già a disposizione. In effetti, molti degli elementi che caratterizzano il Web 2.0 sono ben noti già da molto tempo, tecnologie a disposizione degli sviluppatori già da fine anni '90. Di seguito, dunque, si procede nella loro breve descrizione.

1.1.2 Le Tecnologie

Come è stato appena detto, si può affermare semplicemente che alcune tecnologie che una volta erano solo concetti ora sono arrivate alla piena maturazione e sono in grado di attribuire al Web 2.0 capacità di immagazzinare, creare e diffondere le informazioni che vanno ben oltre quello che un tempo ci si aspettava dai siti. Di seguito viene presentata una breve lista di tecnologie ed elementi che in generale caratterizzano i siti, le applicazioni e i servizi Web 2.0.

- Gli Standard Web come il DOM, l'XHTML e XML, o ancora i CSS, sono e rimaranno ancora per lungo tempo saldi punti fermi nel mondo Web.

- Tecnologie per la condivisione e l'aggregazione delle informazioni mediante RDF, RSS, meglio conosciute come tecnologie per il content syndication. La prima e forse più importante evoluzione verso il Web 2.0 riguarda sicuramente l'aggregazione ed il riutilizzo dei contenuti dei siti Web. In questo senso soprattutto RSS (Really Simple Syndication) è veramente una tecnologia che ha fatto la differenza.

- API per Webservice con REST o SOAP, ma soprattutto openAPI, cioè API pubbliche e a disposizione di chiunque come abbiamo già avuto modo di osservare in precedenza.

- tecniche per la creazione di applicazioni ricche e non invadenti, come per esempio AJAX, senza dimenticarsi di altre importanti tecnologie come Flash, Lazslo e Flex che già da molto tempo sono impegnate in questo senso.

1.1.3 AJAX

AJAX, come acronimo di Asynchronous JavaScript And XML, fu enunciato per la prima volta da Jesse James Garrett, il 18 Febbraio 2005, nel titolo di un post all'interno del suo blog. Esso, però, non si limita a definire una semplice tecnologia, come il suo essere un acronimo può lasciar pensare, ma rappresenta un approccio allo sviluppo di applicazioni Web del tutto nuovo.

Va detto che AJAX sta avendo un grande successo e questo è dovuto principalmente al fatto che offre effettivamente qualcosa di nuovo dal punto di vista dell'utente finale. Inoltre si può ben dire che AJAX abbia dato nuova vita allo sviluppo di applicazioni Web ed abbia spronato la diffusione del "Web 2.0" forse ancor più di molti altri fattori che, come è stato detto in precedenza, caratterizzano questa nuova filosofia di concepire il Web. Offrire agli utenti esperienze ricche tramite applicazioni Web che siano sempre più vicine alle applicazioni desktop non è solo uno degli elementi caratterizzanti il Web 2.0, è in realtà il fulcro stesso di questa innovazione, uno dei principali motivi del suo successo. E siccome AJAX è la principale innovazione in questo ambito, esso ha di conseguenza ricevuto sempre più attenzione da parte di ricercatori e sviluppatori.

Dando uno sguardo al recente passato, le applicazioni Web apparentemente più ricche erano legate principalmente alle tecnologie Adobe-Macromedia Flash o Java Applet. Entrambe purtroppo non sempre interpretabili dai browser e troppo spesso usate a sproposito con il solo scopo di stupire l'utente piuttosto che aggiungere vere e proprie funzonalità. In alternativa a queste tecniche di interazione client/server, nel 1996 venne introdotta una nuova quanto semplice

tecnologia, l'iframe, ossia una sorta di frame inline che ben presto fu supportat dai principali browser Web dell'epoca. Molti sviluppatori sfruttarono quest'ultimo modificando l'attributo sorgente della pagina racchiusa e simulando così un refresh trasparente di una parte di contenuti il che emulava, in maniera veramente poco lineare, un'interazione asincrona. Nel 1998 Microsoft cominciò a sviluppare una tecnologia, chiamata Remote Scripting, con lo scopo di creare una tecnica più elegante per richiamare contenuti differenti ed è in questo periodo, seppur con nome differente, che AJAX venne utilizzato per la prima volta, per poi evolversi in versioni più mature fino a quella attuale.

Il motivo principale per cui il Remote Scripting non ebbe immediato successo è che solo in un secondo momento, precisamente quando Google mostrò cosa era riuscita a fare all'interno dei suoi applicativi senza necessità di Flash Player o Java Virtual Machine, esso suscitò lo stupore degli addetti ai lavori, mettendo in evidenza come queste applicazioni fossero contemporaneamente in grado di fornire una grande compatibilità con i browser Web ed un approccio fin da subito mirato a temi come l'accessibilità e l'usabilità.

Le caratteristiche elencate di seguito sono una guida molto generale, una sorta di percorso e un insieme di regole comunque indicative per stabilire cosa costituisce un'applicazione AJAX.

Interazione uniforme e continua
I siti Web tradizionali seguono uno schema del tipo: inviare un form, aspettare alcuni secondi, far ricaricare la pagina e ripartire con lo stesso ciclo. Questo accade perchè qualunque interazione con il server, perfino una modifica minima nella visualizzazione, richiedono una chiamata al server e un aggiornamento completo della pagina; una sequenza lenta e frustrante. AJAX modifica questo modello in diverse maniere. In primo luogo le interazioni col server possono essere gestite tramite JavaScript, per cui è possibile inviare comandi e scaricare dati nuovi senza dover aggiornare la pagina. In secondo luogo, il codice JavaScript che gira dalla parte del browser può manipolare direttamente la visualizzazione, cosicchè non è necessario caricare dal server una pagina completamente nuova per nascondere un elemento o riarrangiarne la disposizione. In terzo luogo azioni

compiute dall' utente come click del mouse e digitazione possono essere gestite anch'esse tramite JavaScript, cosicchè l' interazione è molto più ricca del modello tradizionale di invio dei moduli. Tutti questi miglioramenti fanno sì che le interazioni AJAX vengano percepite come più rapide e più regolari.

In diretta
Poichè l' interazione browser-server non rappresenta più un ostacolo, è possibile interrogare continuamente il server in cerca di nuove informazioni. Perciò un'applicazione AJAX può essere programmata per mostrare sempre le ultime notizie, chi altri sia online, o per mandare messaggi agli utenti. Il contenuto è "in diretta", o per meglio dire aggiornato continuamente.

Sostegno
Le applicazioni AJAX possono monitorare le azioni dell'utente e dare un sostegno attivo alle attività che esso sta svolgendo. Azioni quali premere un tasto possono portare a chiamare il server, dove il potere di calcolo e le informazioni immagazzinate possono essere sfruttati per produrre contenuto utile in tempi estremamente rapidi. Per esempio, un modulo potrebbe cambiare a seconda dei dati immessi dall'utente, oppure potrebbe apparire un messaggio di errore non appena viene digitato un valore.

Effetti visivi
Le applicazioni AJAX appaiono simili ad applicazioni Web convenzionali, ma tendono ad includere una maggior quantità di animazioni. Non simili a quelle di Flash, spesso piazzate solo per motivi estetici, ma animazioni che fanno passare un messaggio relativo all'evento in corso e a quel che l'utente può fare dopo. Per esempio, un'icona cancellata potrebbe restringersi lentamente e sparire.

Nuovi stili di interazione
AJAX vede aggiornati anche gli stili di interazione. Gli sviluppatori AJAX hanno preso in prestito pure in questo campo concetti che appartengono agli ambienti desktop tradizionali. Il drag and drop per esempio è stato una caratteristica stabile degli ambienti a finestre per

circa vent'anni ma per qualche motivo non è stato mai integrato nel Web. Ora comincia ad affacciarsi anche sul browser e in contesti in cui la sua applicazione è davvero sensata, come la personalizzazione dei portali. Stanno diventando popolari anche altri stili di interazione, quali le combinazioni di tasti per sveltire le attività, e alcuni sviluppatori iniziano a sperimentare con i tasti del mouse: l' utilizzo del doppio click e pure dei tasti destro e centrale.

1.2 Web 3.0

Il "Web 3.0" è un termine a cui corrispondono significati diversi volti a descrivere l'evoluzione dell'utilizzo del Web e l'interazione fra gli innumerevoli percorsi evolutivi possibili. A seguito dell'introduzione del termine "Web 2.0" come descrizione della recente evoluzione del Web, molti studiosi ed esperti nel campo informatico hanno introdotto il termine "Web 3.0" per raccogliere tutte le sue ulteriori ipotesi di evoluzione. In questo senso, i punti di vista su quale sarà il futuro di Internet sono molteplici, da chi vede una vera rivoluzione solo se basata su un nuovo sistema di gestione dell'informazione a chi pensa che la vera evoluzione possa venire dall'introduzione nelle applicazioni Web di una computer graphics finalmente avanzata. C'è tuttora un dibattito piuttosto acceso su cosa debba essere parte integrante di questa evoluzione e cosa no, ma le tematiche più ricorrenti sono sicuramente quelle legate al cosiddetto "Web Semantico", che verrà approfondito con maggiore dettaglio in seguito. Ciò che è certo è che, se per quanto riguarda il Web 2.0 ancora non si è giunti ad una definizione del tutto univoca, tantomeno la sua successiva evoluzione, il Web 3.0, ha dei lineamenti chiari e ben delineati. Ciò che traspare, però, è che a differenza del Web 2.0 questo nuovo passo evolutivo sembra essere ben più importante, persino in grado di cambiare definitivamente la concezione classica del Web e di soddisfare finalmente le nuove esigenze degli utenti finali.

1.2.1 Le Definzioni

Il termine Web 3.0 è apparso per la prima volta agli inizi del 2006 in un articolo[4] di Jeffrey Zeldman, celebre editore e consulente in

ambito informatico, che lo introdusse mentre conduceva una critica verso il Web 2.0 e le sue tecnologie associate. Le seguenti sono invece le più celebri definizioni che il termine, nella sua vastità di significati, ha ricevuto negli ultimi anni:

- Nel Maggio 2006, Tim Berners-Lee, pioniere del World Wild Web, affermò: "*Le persone si chiedono che cosa sia il Web 3.0. Io penso che forse, quando noi otterremo una giusta padronanza della Grafica Vettoriale Scalabile - tutte le increspature e pieghe appaiono oggi nebbiose - nel Web 2.0 ed un completo accesso al Web Sematico integrato ad un grande spazio di dati, noi avremo finalmente accesso ad una incredibile risorsa di dati*".

- Esattamente un anno dopo, al Digital Forum di Seoul, Eric Schmidt, amministratore delegato di Google, fu interpellato per definire il Web 3.0. La sua risposta fu: "*Se devo fare una ipotesi su cosa il Web 3.0 sia, io dico che esso rappresenta un nuovo modo di fare applicazioni. La mia affermazione sta a significare che il Web 3.0 deve essere visto come gruppi di diverse applicazioni che vengono unite insieme. Ci sono alcune caratteristiche che possono descrivere tutto questo: applicazioni relativamente piccole, dati come in una immensa nube, applicazioni che possono essere eseguite su ogni piattaforma, programmi veloci e personalizzabili. Inoltre, le applicazioni dovranno essere intrinsecamente distribuite: esse non dovranno più essere acquistate e memorizzate. Il Web 3.0 è il modello applicativo più innovativo che si sia mai visto nell'informatica*".

- Infine nel Novembre 2006, al Technet Summit, il fondatore e presidente di Yahoo, Jerry Yang, disse: "*Il potere della Rete è giunto ad un livello altissimo tramite le funzionalità assolvibili dalla rete stessa. Negli ultimi quattro anni stiamo vedendo la diffusione di dispositivi e modi di interagire con la rete sempre più ricchi, non solo nel campo dell'hardware come per le game console ed i dispositivi mobili, ma anche a livello software. Non è più necessario essere degli esperti informatici per sviluppare un programma. Stiamo osservando tutto ciò nel Web 2.0 e nel Web 3.0 ed è chiaro che essi, come medium delle comunità, rendono*

> *ormai confusi i confini tra professionisti, semiprofessionisti e consumatori, creando un "network effect" del business e delle applicazioni*".

I precedenti interventi, sebbene assomiglino più a visioni che a vere proprie definizioni, danno un'idea su cosa si intenda per "Web 3.0". Degli aspetti appena citati, però, non tutti stanno trovando lo stesso successo. In particolare, il tema che sembra essere di maggior interesse per i ricercatori, sebbene in realtà non così nuovo, è il Web Semantico. Esso, infatti, vide gli esordi già nel 2001, quando per esempio ancora non si parlava minimamente di Web 3.0, e da allora ha continuato ad essere oggetto di grande interesse per molti studiosi del Web. Essendo poi stato riportato in auge dai dibattiti sul Web 3.0, esso ne rappresenta ora una delle tematiche fondamentali. Si procede dunque ad una sua analisi più approfondita, abbracciando inevitabilmente quella corrente di pensiero che vede in una nuova e sostanziale gestione dell'informazione la vera evoluzione del Web.

1.2.2 Web Semantico

Il Web Semantico è un progetto mirato a creare un mezzo universale per lo scambio di informazioni attraverso la rete attribuendogli un significato (semantica), in modo tale che sia comprensibile dalle macchine. Con questo si intende la trasformazione del World Wide Web in un ambiente dove sia possibile pubblicare non più solo documenti (pagine HTML, file office, immagini, file multimediali, ecc.), ma anche informazioni e dati, in un formato adatto alla interrogazione, interpretazione e, più in generale, alla elaborazione automatica. Dati così descritti possono poi essere elaborati per usi diversi: estrazione di informazioni secondo specifici criteri, riformulazione più o meno parziale per adattarli ad altri formati, visualizzazione in funzione delle capacità del terminale.

Secondo il W3C il Web Semantico altro non è che una estensione del World Wide Web attuale. Le pagine Web attuali sono pensate per essere lette dagli uomini e non dalle macchine. Rendendo le pagine comprensibili per le macchine si portrebbero ottenere numerosi benefici, come far svolgere delle azioni ad applicazioni in modo standardizzato. I benefici potenziali sono rappresentati dalla possibilità dei

computer di sfruttare l'enorme rete di informazioni e servizi presenti sul Web stesso.

Le tecnologie principali sulle quali si basa oggi il Web Semantico sono:

- **XML**, che offre una sintassi per strutturare i documenti ben oltre quel poco che si potrebbe fare con il solo HTML;
- **XML Schema**, vale a dire il linguaggio per regolare la struttura dei documenti XML;
- **RDF**, un semplice data model per far riferimento ad oggetti e a come questi sono relazionati (un modello RDF-based può essere espresso tramite una sintassi XML);
- **RDF Schema**, cioè un vocabolario per descrivere le proprietà e le classi delle risorse RDF, con una semantica per esprimere le gerarchie di tali proprietà e classi;
- **OWL** (Web Ontology Language) il quale estende le possibilità di descrivere le proprietà e le classi.

L'intento fondamentale è quello di accrescere l'usabilità e l'utilità del Web e delle sue risorse interconnesse attraverso: documenti "marcati" con informazioni semantiche machine-readable di vario tipo, vocabolari di metadati comuni (ontologie) e mappe tra vocabolari diversi per permettere a chi crea i documenti di sapere esattamente come marcare i propri documenti, agenti automatizzati per l'esecuzione di compiti per gli utenti del Web Semantico utilizzando i metadati, servizi Web-based per fornire informazioni specificamente agli agenti.

Andando ad analizzare meglio ciò che implica il Web Semantico, si può dire che uno dei primi dibattiti relativi a queste tematiche è, per certi versi, anche il più illuminante. Nel Maggio del 2001, presso la rivista Scientific American Magazine, Sir Tim Berners-Lee, James Hendler e Ora Lassila pubblicarono un celebre articolo[5] in cui tracciarono le linee guida in grado di descrivere i più importanti aspetti di quella che allora era ancora poco più che un'idea, per l'appunto il

Web Semantico. L'articolo iniziava con la descrizione, molto ad effetto, di una realtà i cui sistemi informatici sono basati su questa nuova evoluzione del Web: "*Il sistema hi-fi sta trasmettendo la canzone dei Beatles 'We Can Work It Out' quando suona il telefono. Mentre Pete risponde, il suo telefono abbassa il volume della musica inviando un segnale a tutti i dispositivi della casa. Dall'altra parte della cornetta c'è Lucy, la sorella, che si trova dal medico di famiglia. 'La mamma ha bisogno di una visita specialistica e successivamente di una serie di sedute di riabilitazione.' gli comunica la ragazza. 'Ne deve eseguire due a settimana circa. Sto avviando il mio agente per fissare gli appuntamenti.' Pete, senza esitare, le dice di poter condividere l'onere di accompagnare la madre alle visite. Una volta riagganciato il telefono, ancora nell'ufficio del dottore, Lucy instruisce il suo agente semantico grazie al suo Web browser integrato nel dispositivo di telefonia mobile. Esso, immediatamente, si occupa di recuperare le informazioni relative al trattamento prescritto alla madre di Lucy dall'agente del medico ed a cercare una lista dei servizi (ad un massimo di 20km di distanza dalla casa della madre e compatibili con la sua assicurazione personale) in grado di fornire le dovute sedute di riabilitazione e che abbiano una qualità almeno ottima o molto buona. Successivamente, esso cerca di trovare degli orari possibili per ogni appuntamento settimanale in ognuno di questi servizi (interrogando i loro agenti), compatibilmente agli impegni di Lucy e Pete, che trova nei loro relativi agenti. In pochi minuti l'agente semantico presenta a Lucy un primo piano, tuttavia uno dei servizi da lui scelti si trova presso l'University Hospital, dall'altra parte dlla città. Secondo Pete è troppo lontano da raggiungere, in particolar modo all'orario di pranzo in cui l'appuntamento è stato fissato. Egli, perciò, instruisce il proprio agente con nuove restrizioni sul luogo e sugli orari in cui fissare gli appuntamenti e ne avvia una nuova ricerca. L'agente di Pete, assistito da quello di Lucy che ora ha completa fiducia nel suo, compie una nuova scansione ed in pochi secondi presenta un nuovo piano. Questa volta le cliniche interessate e gli orai stabiliti sono ottimi, tuttavia l'agente di Pete comunica anche due note: la prima è che alcuni appuntamenti in questo casa andrebbero a coincidere con impegni di scarsa importanza nell'agenda di Pete e la seconda è che il piano assicurativo della madre non è perfettamente compatibile con uno dei servizi, con la possibilità di ot-*

tenere ulteriori dettagli. Dopo aver chiesto il consenso a Lucy tramite una seconda telefonata, Pete decide di ignorare quest'ultima nota e di spostare i suoi impegni minori. Egli, infine, conferma le prenotazioni e gli appuntamenti tramite il suo agente".

Il seguito dell'articolo cerca poi di delineare gli aspetti chiave del Web Semantico, tenendo l'esempio precedente come possibile applicazione, o ancora meglio visione futura. Si riportano i concetti principali che i tre ricercatori individuarono:

Espressione del Significato
Oggi la maggioranza dei contenuti del Web sono scritti per essere letti da persone e non da macchine. Un software può operare un parsing adattativo delle pagine, basandosi su layout predefiniti, ma in generale esso non è in grado di processarne la semantica. Il Web Semantico, invece, introduce una struttura al contenuto informativo delle pagine Web, creando un ambiente in cui gli agenti, entità software autonome che verranno meglio trattate in seguito, che navigano da pagina a pagina possono estrapolare direttamente informazioni sofisticate per gli utenti finali. Il Web Semantico non è separato dal concetto attuale di Web, ma ne è una estensione, dove le informazioni sono legate ad un significato ben definito e disponibile sia per macchine che per utenti, che possono dunque cooperare nel loro lavoro.

Rappresentazione della Conoscenza
Perchè il Web Semantico funzioni, le macchine devono avere a disposizione una collezione strutturata di informazioni ed un elenco di regole deduttive in modo da poter condurre ragionamenti autonomi. Il concetto di "rappresentazione della conoscenza" non è nuovo in ambito informatico, ma esso ha sempre fatto perno sulla centralizzazione. Occorre che ognuno condivida la stessa definizione di ogni concetto perchè si possa pensare di costruire un sistema che si basi su di essi. Tale prerogativa di centralizzazione è però diventata ben presto un limite. Lo scopo del Web Semantico è quindi diventato quello di fornire un linguaggio in grado di esprimere sia dati che regole di ragionamento in grado di integrare altri dati ed altre regole importandoli nel Web da rappresentazioni della conoscenza esterne e preesistenti. In questa ottica le tecnologie che per prime si sono

fatte strada sono state sicuramente XML ed RDF. RDF (Resource Description Framework) è un framework per la descrizione della conoscenza nel Web. Esso si basa su tre principi chiave: qualunque cosa può essere identificata da un Universal Resource Identifier (URI), utilizzare il linguaggio meno espressivo per definire qualunque cosa, qualunque cosa può dire qualunque cosa su qualunque cosa. L'unità base per rappresentare un'informazione in RDF è lo statement, ossia è una tripla del tipo: Soggetto Predicato Oggetto, dove il soggetto è una risorsa, il predicato è una proprietà e l'oggetto è un valore. Sebbene RDF permetta potenzialmente di descrivere gran parte della conoscenza sul Web, particolare attenzione va posta sul concetto di URI. Esso diventa infatti fondamentale qualora si vogliano eliminare definitivamente le ambiguità e definire univocamente il significato semantico di una parola che, in diversi contesti e situazioni, potrebbe riferire a concetti diversi.

Ontologie
In filosofia, una ontologia è una teoria sulla natura dell'esistenza. I ricercatori Web hanno successivamente adottato questo termine utilizzandolo per definire ogni documento o file che definisce univocamente le relazioni tra diversi termini. Il tipo più comune di ontologie Web è caratterizzato da una tassonomia e da regole deduttive. La tassonomia definisce classi di elementi e le relazioni tra di essi, mentre le regole deduttive devono permettere ad un programma di "capire" informazioni inespresse tramite inferenze sulle informazioni invece disponibili indotte dalle regole stesse. L'importanza delle ontologie interviene qualora una macchina voglia manipolare informazioni provenienti da basi di dati diversi. Se un termine non avesse la propria definizione in ogni base di dati, collegabile a quella memorizzata in un'altra base di dati in modo da poterne effettuare un matching, sarebbe impossibile per una macchina capire se due termini di due basi di dati diverse fanno riferimento al medesimo concetto o meno.

Agenti
Il reale potere del Web Semantico si percepisce appieno solo in presenza di applicazioni che, ottenendo informazioni da diverse risorse del Web, processano i contenuti e scambiano i risultati della loro

computazione con altre applicazioni. In quest'ottica, evidentemente orientata ad un paradigma ad agenti, il Web Semantico diviene fondamentale qualora esso permetta a due agenti, non creati appositamente per interagire, di scambiarsi informazioni grazie al trasferimento dei dati e della loro semantica. E' perciò possibile prevedere la creazione di una cosiddetta "catena del valore", nella quale informazioni disassemblate vengono passate da un agente all'altro, ognuno dei quali ne accresce il contenuto, fino a costruire il prodotto finito che è stato richiesto dall'utente finale.

E' inoltre interessante come il Web Semantico possa persino oltrepassare le barriere del Web stesso per sfociare nel mondo reale. Un URI, per definizione, può riferire qualsiasi cosa, è perciò credibile che dispositivi fisici, dal televisore al telefono cellulare, dall'impianto hi-fi al sistema di illuminazione, possano essere identificati tramite URI e quindi essero oggetto di interazione all'interno di applicazioni Web evolute.

Evoluzione della Conoscenza

Come conclude Sir Tim Barners-Lee nel suo articolo: "*Il Web Semantico non è "solo" un modo per raggiungere i singoli scopi sopradescritti, esso può assistere l'evoluzione della conoscenza umana nel suo complesso.*".

Le nuove conoscenze nascono sempre da piccole comunità e necessitano di molto tempo per diffondersi in tutto il mondo. Un piccolo gruppo può innovarsi velocemente e con efficienza, ma questo produce una subcultura i cui concetti non vengono capiti dagl'altri. Il mondo procede sempre su due estremi, con la tendenza di partire da una idea personale, fino ad arrivare ad una diffusione planetaria grazie al passare di molto tempo. La necessità di unire le conoscenze sviluppate da diversi piccole realtà è tangibile ed il Web Semantico cerca proprio di inserirsi in questo contesto, fornendo strumenti e linguaggi in grado di permettere a tutti, in maniera il più semplice possibile, di fornire ed usufruire contemporaneamente della conoscenza collettiva.

Capitolo 2

Web 2.0 e Agenti

In questo capitolo si cercherà di motivare più nello specifico l'introduzione di un modello ad agenti per applicazioni Web 2.0, attraverso un'analisi delle tecnologie attualmente utilizzate e successivamente alla discussione dei vantaggi che ci potrebbe apportare sia per gli utenti che per gli sviluppatori di software.

2.0.3 Web 2.0: Quale modello di programmazione?

Per prima cosa, occorre capire quale sia il modello di programmazione ad oggi introdotto nelle applicazioni Web 2.0. Per farlo, occorre certamente analizzare la tecnologia che ha un ruolo centrale all'interno di questa attuale evoluzione del Web, ossia AJAX.

Sebbene stia trovando una grande diffusione, AJAX è in realtà un agglomerato di tecnologie eterogenee e molto spesso ancora perfezionabili. Inoltre, sfruttando l'interazione di tecniche di programmazione e linguaggi nati per scopi diversi e poi riuniti in un secondo momento per uno scopo comune, spesso l'utilizzo integrato di tutte queste tecnologie non è semplice e lineare. Esse manifestano a volte alcuni problemi e impongono spesso importanti vincoli a quanto è possibile realizzare.

In particolare è da sottolineare come AJAX fondi le proprie basi su JavaScript e questo se in alcuni ambiti è un vantaggio, in altri è un grosso limite. JavaScript, al momento, fornisce un supporto quasi

nullo al multithreading, è un linguaggio debolmente tipizzato ed è anche debolmente orientato agli oggetti.

Inoltre, ad oggi, vi è un disaccoppiamento tra il linguaggio usato per implementare l'applicazione Web lato server e lato client, dove in gran parte si usa appunto JavaScript, il che rende impossibile la replicazione di logiche comuni ad entrambe le parti. Applicazioni che devono eseguire la stessa logica computazionale sia lato client che lato server vanno solitamente riscritte in due linguaggi di programmazione distinti.

Alcuni di questi problemi, sicuramente già noti da tempo, in realtà sono già stati affrontati ed in parte risolti. E' importante far notare, ad esempio, come Google con il progetto GWT (Google Web Toolkit)[1] abbia creato un framework per sviluppare applicazioni Web in AJAX ma esulando dall'utilizzo di JavaScript. Con GWT, è possibile sviluppare ed effettuare il debug di applicazioni AJAX in linguaggio Java, usando un tools di sviluppo a propria scelta tra quelli tipici di Java. Un apposito compilatore si occupa poi di trasformare i sorgenti Java in applicazioni composte da pagine HTML e script JavaScript.

GWT, in particolare, permette di utilizzare quindi un unico linguaggio di programmazione fortemente orientato al paradigma ad oggetti, Java per l'appunto, per sviluppare l'intera applicazione lato client, con i conseguenti vantaggi in termini di decomposizione ed incapsulamento. Tuttavia, pur offrendo un ambiente di sviluppo più agevole e totalmente basato su Java, GWT non modifica in realtà il modello computazionale dell'applicazione Web che, in definitiva, si riduce sempre ad essere una normale applicazione basata su AJAX, con i relativi limiti.

Inoltre, se questi sono problemi più dal punto di vista tecnologico, ad essi se ne affiancano altri più a livello concettuale, che riguardano il modello computazionale e di interazione. Problemi che GWT ad esempio non risolve affatto.

La prima considerazione che si può fare a riguardo è sul meccanismo di comunicazione messo a disposizione da AJAX. L'invio di richieste asincrone è sicuramente di estrema utilità, ma la gestione di risposte ricevute tramite callback non è certamente agevole e non fornisce un'astrazione in grado di soddisfare quelli che sono oggi i requisiti

[1] http://code.google.com/Webtoolkit/

per un modello di applicazioni complesse. Un tale meccanismo non è sufficiente a descrivere protocolli di comunicazione articolati: per applicazioni di una certa semplicità sembra che i mezzi messi a disposizione da AJAX siano sufficienti, ma non appena i sistemi si fanno un pò più complessi, queste tecnologie mostrano immediatamente i loro limiti.

Inoltre AJAX si basa su linguaggi eterogenei, non tutti ad oggetti, che di conseguenza non forniscono astrazioni in grado di gestire in maniera semplice ma completa problemi di coordinazione e comunicazione tipici delle applicazioni distribuite. Se, come si è visto, l'obbiettivo deve essere quello di fornire un modello computazionale in grado di supportare lo sviluppo di applicazioni Web del tutto simili ad applicazioni desktop distribuite, occorre che tale modello fornisca astrazioni e meccanismi davvero orientati, seguendo la filosofia del Web 2.0, alla collaborazione ed alla cooperazione delle entità in gioco. AJAX, in realtà, non definisce alcun modello innovativo in grado di aiutare lo sviluppatore nella realizzazione di applicazioni di nuova generazione basate sul Web.

In questo senso, basandosi sul linguaggio Java, nemmeno GWT può fornire meccanismi utili alla risoluzione di problemi di concorrenza, collaborazione e comunicazione asincrona che, con l'aumento della complessità delle applicazioni Web che si intende sviluppare, diventa in realtà di grande importanza e centralità.

In realtà, è il paradigma Object Oriented stesso a vacillare davanti a tematiche di questo tipo. Se, in effetti, nell'ingegneria del software per applicazioni desktp (che esulano dall'ambito Web) si è sentita nel tempo l'esigenza di introdurre nuovi paradigmi e modelli che portassero a quella che è oggi la programmazione ad agenti, è evidente che vedere le future applicazioni Web sempre più simili ad applicazioni desktop eseguite all'interno del browser, significa dover prevedere che tale tipo di programmazione venga estesa anche a questo tipo di applicazioni. Le motivazioni per cui il paradigma Object Oriented diventerebbe ad un certo punto inadeguato, sono dunque le stesse che hanno portato alla creazione di quello ad agenti per applicazioni desktop, su tutte l'aumento della loro complessità e la loro crescente connotazione distribuita.

2.0.4 Perchè Agenti lato Client?

Dopo aver descritto in maniera molto critica l'attuale modello di programmazione introdotto dal Web 2.0, si procede ora nell'elencare le motivazioni vere e proprie che portano a proporre l'introduzione di un modello ad agenti, contestualmente ai problemi sopra elencati che esso potrebbe risolvere.

- **Un nuovo modello di programmazione.** Le applicazioni Web 2.0 sono innanzitutto applicazioni distribuite che fanno largo uso di comunicazione asincrona tra client e server, per mettere a disposizione dell'utente un'interfaccia viva e reattiva, in grado di permettere l'implementazione di servizi basati sulla cooperazione e sulla coordinazione. Partendo da questa definizione, frutto delle ampie discussioni precedenti, è evidente che il modello di programmazione previsto dal Web 2.0 possa essere notevolmente migliorato con l'introduzione di un paradigma ad agenti, in cui la coordinazione, la cooperazione e l'interazione asincrona sono viste come astrazioni di prima classe. Esso permetterebbe non solo di rappresentare questi concetti dietro ad entità ben definite, gli agenti appunto, ai fini implementativi, ma aiuterebbe il programmatore nell'analisi e nel progetto di applicazioni complesse, grazie all'utilizzo di metodologie create appositamente per la programmazione ad agenti.

- **Carico computazionale lato client.** A questo punto, si può fare una considerazione molto generale sulle odierne tendenze nello sviluppo di applicazioni Web. Superati molti dei problemi di velocità di comunicazione tra client e server che hanno caratterizzato per decenni sia Internet che il Web, assistiamo oggi ad una parziale inversione di tendenza per quanto riguarda il bilanciamento del carico computazionale tra client e server. Se un tempo era necessario minimizzare sia il carico computazionale lato client che la mole di dati scambiati con il server a causa della lentezza della rete e della limitatezza di risorse, oggi questi vincoli sono in parte scomparsi. E' quindi logico pensare che con il tempo ci si orienti verso un maggior bilanciamento del carico computazionale tra le due controparti e che

dunque la complessità delle applicazioni Web lato client aumenti considerevolmente.

Per rendersi conto di ciò, basti pensare che inizialmente il Web inteso come ipertesto virtuale per lo scambio di informazioni tramite la rete, non prevedeva affatto un qualsiasi tipo di computazione lato client ed invece ora disponiamo di innumerevoli tecnologie quali le Applet, JavaScript, AJAX e quant'altro.

Detto questo, viene da sè che l'aumento della complessità di questi sistemi è assolutamente compatibile con la scelta di introdurre un modello ad agenti e ne costituisce anzi uno dei motivi principali.

- **Omogeneità del linguaggio tra Server e Client e migrazione di codice.** Si è citato in precedenza il problema di avere diversi linguaggi di programmazione per lo sviluppo della parte server e di quella client di una applicazione Web. Ciò implica l'impossibilità di poter riutilizzare, per la parte client o per quella server, il codice scritto per quella opposta, a meno di una conversione dello stesso. Inoltre preclude totalmente la possibilità di veder migrare intere porzioni di codice da server a client, o viceversa, in modo ad esempio di equilibrare il carico computazionale tra le due parti.

  Se il primo problema ha trovato una parziale soluzione nel già discusso framework GWT, per il secondo il sistema di Google non offre alcun aiuto in quanto esso in realtà genera pur sempre codice JavaScript e HTML.

  Introdurre un modello ad agenti in questo contesto, invece, potrebbe permettere la creazione di applicazioni Web in cui il codice, sottoforma di agenti, può migrare da server a client e viceversa, risolvendo tutti i problemi dovuti alla distribuzione ed alla comunicazione grazie alle astrazioni fornite da tale modello.

- **Network Effect.** Riprendendo la definizione di Web 2.0 data da Tim O'Reilly[3] e citata all'inizio del capitolo precedente, in essa si parla del cosiddetto network effect, ossia del fenomeno per cui il software migliora all'aumentare del numero di utenti che

lo utilizza. Ciò è possibile grazie alla condivisione, da parte di ogni utente che usufruisce di un servizio, di nuove informazioni e dati che contribuiscono a migliorare progressivamente il servizio stesso. In questo senso, la possibilità di avere agenti software in grado di automatizzare la storicizzazione e l'organizzazione delle informazioni, nonchè la ricerca ed il filtraggio dei dati, è sicuramente una prospettiva allettante. Inoltre, contrariamente ai punti precedenti in cui si faceva riferimento al concetto di agente soprattutto in termini di concorrenza, collaborazione, cooperazione e comunicazione, in questo caso potrebbe essere utile estendere il discorso alle astrazioni di agente intelligente o agente BDI (Beliefs Desires Intentions). Esse, infatti, vengono spesso citate negli ambiti che riguardano l'apprendimento, la ricerca e l'estrazione di informazioni, rivelandosi particolarmente adatta a risolvere problemi di questo tipo.

Queste argomentazioni, comunque, sono strettamente legate al contesto del Web Semantico e verranno quindi riprese ed ampliate nel capitolo successivo. Per ora, è importante sottolineare come, anche facendo riferimento solamente alla prima definizione di Web 2.0, l'introduzione di un modello ad agenti per applicazioni Web lato client potrebbe risolvere molti problemi ed offrire scenari interessanti per il futuro, sicuramente in linea con la filosofia introdotta da questa evoluzione del Web.

Capitolo 3

Web Semantico ed Agenti

In questo capitolo, analogamente a quanto fatto precedentemente per il Web 2.0, si discuteranno le motivazioni che possono giustificare l'introduzione di un modello ad agenti intelligenti nell'ambito del Web Semantico.

3.0.5 Verso gli Agenti

Iniziando con un'analisi di carattere generale, mirata a decidere se l'introduzione di un modello ad agenti sia più o meno utile nell'ottica della diffusione del Web Semantico, si può far riferimento all'articolo[5] già citato nel primo capitolo, nel quale il Web Semantico viene definito da Berners-Lee, Hendler e Lassila. In esso, gli autori spiegano come "*[...] Il Web Semantico diviene fondamentale qualora esso permetta a due agenti, non creati appositamente per interagire, di scambiarsi informazioni grazie al trasferimento dei dati e della loro semantica*".

E' evidente che il concetto stesso di Web Semantico è strettamente legato a quello di paradigma ad agenti. La vera potenzialità di questa futura evoluzione del Web, infatti, può essere sfruttata solo da applicazioni che sono in grado di utilizzare, comunicando e cooperando tra di loro, l'insieme di informazioni messe a disposizione da esso. Volendo semplificare il concetto, il Web Semantico può essere pensato come uno strumento, o meglio una piattaforma, creata appositamente per sistemi ad agenti.

Introdurre questo tipo di modello di programmazione, quindi, sarebbe un notevole passo in avanti verso quello che sembra essere il futuro del Web. Da non trascurare, inoltre, è l'impegno che il consorzio W3C sta dedicando a queste tematiche, in particolar modo in un ambito in cui le standardizzazioni sono la chiave per la diffusione nel mainstream delle nuove tecnologie.

Di seguito, dunque, si prosegue nel discutere come questo nuovo modello di programmazione per applicazioni Web potrebbe dimostrarsi utile in due dei maggiori ambiti di applicazione del Web Semantico, ossia l'Information Learn and Retrieving e la cosiddetta "Internet of Things".

3.0.6 Agenti per l'Information Learn and Retrieving

La crescita esponenziale delle informazioni disponibili on-line, ha reso necessaria l'introduzione di strumenti efficienti per la ricerca e l'estrazione di dati[15]. In un mondo ideale, l'utente dovrebbe essere in grado di recuperare istantaneamente informazioni di cui necessita in un determinato momento, filtrandole in base alle sue richieste, ai suoi interessi ed al contesto. Per fare ciò, occorrerebbe disporre di un sistema software adattativo ed intelligente, in grado di capire gli interessi dell'utente e costruire progressivamente un modello delle sue preferenze, da utilizzare al momento del recupero e dell'estrazione delle informazioni.

A questo scopo, il paradigma ad agenti è stato utilizzato per sviluppare diversi strumenti ed applicazioni, molte delle quali esulano dei più recenti concetti relativi al Web Semantico e fanno uso di tecniche di intelligenza artificiale per analizzare i semplici documenti testuali presenti sulla rete. Alla base di ognuna di queste applicazioni, però, c'è sempre il concetto di agente intelligente e di agente goal-based. Un esempio chiarificativo, in questo senso, è dato dal sistema WAWA [1], sviluppato dall'Università del Wisconsin a partire dal 2001. Esso si compone principalmente di due agenti BDI (Beliefs Desires Intentions), che tramite reti neurali multilivello feed-forward e reti

[1] WAWA: Wisconsin Adaptive Web Assistant

neurali knowledge-based, si occupano l'uno del ritrovamento e l'altro dell'estrazione delle informazioni ricercate su una base di conoscenza testuale. Esso mostra molto bene come il paradigma ad agenti sia utile alla risoluzione di questo tipo di problemi e, soprattutto, come le sue astrazioni siano veramente adatta a rappresentare gli elementi, le entità ed i meccanismi di questo tipo di sistemi.

Esulando ancora dai concetti di Web Semantico, un altro ambito in cui la programmazione ad agenti è stata introdotta nel contesto Web è l'indicizzazione dei documenti on-line da parte degli spider-Web [2] dei grandi motori di ricerca. Il più famoso di questi è sicuramente Google-Bot, l'agente realizzato da Google per indicizzare i documenti presenti sulla rete tramite l'analisi del loro contenuto e soprattutto dei metadati specificati dai loro creatori. Esso è un chiaro esempio di agente autonomo eseguito in un ambiente, il Web, dall'elevata dinamicità e, dato il suo successo, costituisce un caso di successo dell'applicazione del paradigma ad agenti a questo contesto.

Infine, si arriva alla tanto discussa evoluzione del Web, ossia il Web Semantico. Esso sta cambiando sicuramente il modo di pensare l'information learn and retreaving, con l'introduzione sia di nuove astrazioni che di nuove tecnologie. In particolare, il sub-topic riguardante i metodi detti Linked Data, descritti da Tim Berners-Lee in una nota architetturale per il W3C[12], sembra essere il punto fondamentale in grado di costituire il passo evolutivo tra la base di conoscenza basata su documenti testuali fino ad ora messa a disposizione dal Web e quella fondata sui concetti di risorsa, classe, URI ed ontologia, sulla quale lo stesso Berners-Lee fece riferimento al momento della definizione del Web Semantico[5]. Il termine Linked Data, più precisamente, è usato per descrivere un metodo per l'esposizione, la condivisione e la connnessione di informazioni via URI dereferenziabili sul Web. I principi fondamentali individuati dallo stesso Berners-Lee nella definizione di Linked Data, portano alle seguenti linee guida:

- Usare gli URI per identificare ogni cosa.

[2]Uno spider-Web, o crawler, è un componente software che si occupa di analizzare una rete o un database in maniera metodica ed automatizzata al fine di produrre informazioni di supporto per l'attività di un motore di ricerca.

- Usare URI HTTP in modo che ogni cosa sia recuperabile e riferibile da persone e agenti software.
- Fornire informazioni strutturate (metadata) sulle risorse che vengono dereferenziate da un URI.
- Includere links ad altri URI inerenti al contesto durante la definizione dei dati, in modo da migliorare la ricerca di informazioni correlate all'interno del Web.

Queste linee guida hanno dato vita ad un intero gruppo di lavoro all'interno del W3C Semantic Web Education and Outreach. Esso si propone lo scopo di estendere l'attuale piattaforma del Web creando una serie di dataset, scritti in linguaggio RDF e in linea con i principi di Linked Data. Il primo di questi dataset si chiama DBpedia[**?**]. Ottenuto dall'estrazione di dati dalla famosa enciclopedia libera Wikipedia, esso contiene circa 2,18 milioni di concetti e 218 milioni di triplette RDF.

In questo contesto, Daniel J. Lewis, ricercatore in computer science di IBM, ha cercato di studiare come e dove potesse inserirsi la programmazione ad agenti. Egli ha identificato tre tipi di agenti che sarebbero in grado di supportare lo sviluppo, l'utilizzo ed il mantenimento di una tale base di conoscenza[10] orientata al Web Semantico:

- Il primo tipo di agente è quello che si occupa di ricercare ed analizzare documenti sul Web al fine di estrarre informazioni e creare dataset che rispettino il principio di Linked Data (come DBpedia ad esempio), basati sui linguaggi RDF ed OWL per descrivere rispettivamente risorse ed ontologie. Tali agenti sarebbero fondamentali per il processo di conversione dell'attuale base di conoscenza del Web, che come visto è già in corso.
- Una seconda classe di agenti sarebbe quella degli utilizzatori del Web Semantico. Agenti BDI in grado di usufruire della nuova base di conoscenza al fine di supportare le attività dell'uomo sul Web, la più comune delle quali è sicuramente la ricerca delle informazioni. Si tratta di agenti fondamentalmente goal-based, in grado di sfruttare la semantica delle informazioni al fine di applicare processi di intelligenza artificiale per produrre risultati migliori di quelli producibili fino ad oggi.

- L'ultimo tipo, invece, è quello di agenti in grado di occuparsi del mantenimento e della creazione di nuovi dataset, sempre orientati al Web Semantico. Anche in questo caso di tratta di agenti intelligenti, in grado di creare nuova conoscenza, sia grazie all'analisi delle informazioni esistenti e della loro semantica, sia grazie al monitoraggio dei dataset correnti in modo da mantenerli aggiornati e funzionali.

E' evidente, sia dagli esempi iniziali che esulano dai concetti di Web Semantico, sia da quest'ultima analisi, che il paradigma ad agenti è sicuramente adatto al contento dell'information learn and retrieving. Le astrazioni di agenti autonomi ed agenti intelligenti sono già state utilizzate con successo in questi ambiti e dunque c'è da aspettarsi che la loro diffusione di consolidi sempre di più, confermando la tesi per cui le basi stesse di Web Semantico poggiano sul concetto di agenti software.

3.0.7 Agenti e Web Semantico per i Web Services: Internet of Things

Quando si parla di "Internet of Things" si intende un'estensione del concetto attuale di internet in cui più e più dispositivi (sensori, computer, pda, telefoni mobili, ...) sono connessi tra di loro ed accessibili proprio come oggi è possibile fare con i contenuti del Web. Secondo il professor D. Fensel dell'università di Innsbruck, ogni device potrebbe potenzialmente essere in grado di fare uso ed offrire servizi su Internet[14]. In quest'ottica, egli afferma che è prevedibile un progressivo aumento dei Web Services disponibili sulla rete, fino a raggiungere un numero equivalente al numero di pagine Web attualmente pubblicate. In un prossimo futuro, quindi, è possibile che si veda la creazione di una Internet of Services che poggi direttamente sulla Internet of Things. Questa visione richiede sicuramente nuove tecnologie che permettano l'introduzione di una elevata automazione, in modo da poter rendere il Web scalabile. La semantica ha già dimostrato, in numerosi contesti, di essere un approccio utile a raggiungere l'automazione dei processi, grazie soprattutto al suo supporto nativo per la composizione, la ricerca e la mediazione. Per questo motivo, uno

degli ambiti applicativi a cui pare essere destinato il Web Semantico è proprio quello della definizione, della ricerca e del mantenimento dei Web Services.

Il Web Semantico, in realtà, è già stato applicato in maniera diffusa nell'ambito dei Web Services, in quanto esso porta una serie di indiscutibili benefici[3] quali:

- Integrazione più veloce tra le applicazioni.
- Riutilizzo di dati e metadati.
- Ricerche diffuse su diverse applicazioni o basi di dati.
- Robustezza.
- Capacità evolutiva.

Esso, tuttavia, pare ora diventare una necessità vera e propria per la diffusione su larga scala di una architettura basata su Web Services, proprio per motivi di scalabilità. Per fare in modo che i servizi siano raggiungibili da utenti ed applicazioni, occorre avere un sistema in grado di fornire meccanismi di ricerca tanto potenti quanto più grande è il numero di questi servizi. In pratica, all'aumentare del numero di Web Services presenti sulla rete, per la loro ricerca valgono tutti i discorsi realtivi alla ricerca ed all'estrazione di informazioni fatti nella sezione precedente.

Anche per questo motivo, quindi, l'introduzione di un modello di programmazione ad agenti per applicazioni Web 3.0 pare essere una scelta dovuta. Gli agenti software si inserirebbero perfettamente in un'architettura basata su Web Services e permetterebbero di costruire applicazioni in grado di integrare e comporre servizi diversi usufruendo di informazioni definite da dati semantici.

[3] http://www.w3.org/2003/Talks/0521-www-keynote-tbl/slide12-0.html

Capitolo 4

Dalle motivazioni al modello

Nei precedenti due capitoli sono stati illustrati i motivi principali che hanno portato a pensare all'introduzione di un modello di programmazione ad agenti per applicazioni web 2.0. Ora, molto brevemente, si procede nel fornire una serie di linee guida che possono essere utili per definire questo modello. Ora, dunque, si procede nell'individuare quali possono essere i punti critici ed i problemi da affrontare durante tale processo e, successivamente, nella realizzazione di un prototipo di applicazione. Nel farlo, si far riferimento alla sperimentazione documentata nella precedente tesi[1], durante la quale si è cercato di modellare ed implementare un tale prototipo, basandosi sul metamodello ad Agenti&Artefatti[9], sul framework ad agenti simpA e sulla piattaforma CARtAgO.

4.0.8 Modello e Prototipo

Evitando di inoltrarsi nella discussione di problemi implementativi e tecnici, dovuti all'integrazione di diverse tecnologie al fine di offrire un ambiente ad agenti in grado di essere eseguito all'interno di un browser, si procede dunque nel definire i seguenti punti:

- **Individuare gli elementi esistenti.** La prima cosa da fare, nella definizione di un modello di questo tipo, è quella di individuare gli elementi indispensabili al funzionamento di un'applicazione Web lato client. Ci si riferisce ad esempio ad una entità che permetta la comunicazione tramite il protocollo

HTTP/HTTPS, oppure alla pagina Web visualizzata all'utente. Le funzionalità incapsulate logicamente da tali elementi devono essere in qualche modo prese in considerazione e devono comparire nel modello, in modo da fornire all'applicazione le stesse potenzialità date oggi dalle tecnologie esistenti.

- **Possibili nuovi elementi.** L'introduzione di un nuovo modello di programmazione potrebbe essere l'occasione per estendere l'attuale insieme di funzionalità offerte dalle odierne applicazioni Web, inserendo nuovi elementi in grado di dare al programmatore strumenti per costruire applicazioni sempre più simili ad applicazioni desktop, in totale assenso con la filosofia del Web 2.0. Mi riferisco ad esempio a elementi in grado di interfacciarsi con applicazioni esterne al browser, oppure ad entità semipermanenti sul client che offrano funzionalità ricorrenti e quindi riutilizzabili da diverse applicazioni. In tal senso, si pensi alle moderne applicazioni di e-commerce, nelle quali troviamo frequentemente diversi shopping cart con funzionalità pressochè identiche. Potrebbe aver senso pensare ad un elemento, persistente sul client, che implementi queste funzionalità e le renda disponibili a tutte le applicazioni che ne necessitano, senza doverne effettuare il download ad ogni esecuzione. Ovviamente, l'inserimento di questo tipo di elementi, deve sottostare ai vincoli tecnologici che sono imposti dall'architettura REST del Web e dall'integrazione con i moderni browser.

- **Mapping sulle astrazioni del meta-modello.** Una volta definiti e descritti tutti questi elementi, occorre mappare ognuni di essi sulle astrazioni del meta-modello che si intende utilizzare. Nell'ambito della programmazione ad agenti, quindi, è necessario individuare quale di essi è modellabile come una entità attiva e reattiva, un agente, e quali invece sono modellabili sull'astrazione di ambiente. Nel secondo caso, qualora si facesse ad esempio riferimento ad un meta-modello ad Agenti&Artefatti[9], come in effetti è stato fatto nella tesi di cui si è parlato in precedenza, si potrebbe quindi effettuare un mapping di ogni elemento che rappresenti uno strumento passivo, utilizzato da uno o più agenti, sull'astrazione di artefatto.

Fatto ciò e ottenuto un primo modello, in cui tutti gli strumenti necessari all'esecuzione di una applicazione Web lato client sono rappresentati come astrazioni tipiche della programmazione ad agenti, si può quindi procedere nella progettazione di un prototipo di applicazione. Per fare questo, ancora una volta, si può seguire lo stesso iter appena descritto e quindi: individuare gli elementi costitutivi dell'applicazione e conseguentemente mapparli sulle astrazioni del meta-modello.

Capitolo 5

Conclusioni

Lo scopo di questo elaborato era dimostrare che l'introduzione di un modello di programmazione ad agenti nell'ambito delle applicazioni Web 2.0 e 3.0 trova solide fondamenta teoriche nelle definizioni e nelle applicazioni di queste due attuali e future evoluzioni del Web.

Dopo una prima analisi dello stato dell'arte per quanto riguarda modelli computazionali e di interazione del Web 2.0, si è cercato di sottolineare quali potrebbero essere i vantaggi che questo nuovo approccio potrebbe apportare ad ogni fase dello sviluppo, a partire dall'analisi, per passare al progetto ed infine all'implementazione di applicazioni Web. Nel secondo capitolo, è stato descritto come il paradigma ad agenti possa essere utilizzato per costruire un modello in grado di supportare lo sviluppo di applicazioni in cui la cooperazione, la coordinazione e la comunicazione asincrona, aspetti fondamentali del Web 2.0, siano rappresentati come entità di prima classe.

Successivamente, nel terzo capitolo, si sono analizzati due dei principali ambiti in cui il modello in discussione potrebbe apportare nuovi ed importanti vantaggi, ossia l'information learn and retrieving ed il Web semantico applicato ai Web services. Anche in questo caso, l'introduzione di un modello di programmazione ad agenti è sembrata una scelta appropriata e di notevole interesse.

In ognuna di queste sezioni, inoltre, è emerso, sia a livello teorico che pratico, come la tesi che è stata più volte sostenuta lungo tutto l'elaborato abbia fondamenta solide sui concetti fondamentali di Web 2.0 e Web Semantico. A questo punto, accertato il fatto che esso si basa su

solide motivazioni, si possono prospettare lavori futuri sull'argomento in termini di esperimenti pratici ed implementativi. Potrebbe essere interessante, ad esempio, cercare di sviluppare un prototipo di applicazione Web basandosi su un modello ad agenti, magari poggiando su diverse tecnologie in modo da confrontarne pregi e difetti.

Bibliografia

[1] M. Minotti: *Un Modello di Programmazione ad Agenti per Applicazioni Web 2.0 e 3.0*, 24 Luglio 2008

[2] M. Minotti, A. Ricci, G. Piancastelli: *An Agent-Based Programming Model For Developing Client-Side Concurrent Web 2.0 Applications*, WEBIST, Marzo 2009

[3] Tim O'Reilly: *What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software*, 30 Settembre 2005

[4] Jeffrey Zeldman: *Web 3.0*, pubblicato nel n.210 della rivista A List Apart

[5] Tim Berners-Lee, James Hendler and Ora Lassila: *The Semantic Web*, pubblicato nella rivista Scientific American Magazine il 17 Maggio 2001

[6] N.Jennings: *An agent-based approach for building complex software systems.* Communication of ACM, 44:35-41, Aprile 2001

[7] M.Wooldridge: *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, 1:27-73, 1999.

[8] M. Wooldridge, N. R. Jennings: *Intelligent agents: Theory and practice* Knowledge Engineering Review, 1995.

[9] A. Omicini, A. Ricci, M. Viroli: *Agens Faber: Toward a theory of artefacts for MAS.* Electronic Notes in Theoretical Computer Sciences, 150(3):21-36,29 May 2006.

[10] Daniel J Lewis: *Intelligent agents and the Semantic Web*, IBM Library, 21 Oct 2008.

[11] Charles J. Petrie: *Agent-Based Engineering, the Web, and Intelligence*, IEEE Expert, December 1996.

[12] Tim Barners-Lee: *Linked Data: Design Issues*, W3C architecture note, 27-07-2006.

[13] James Hendler: *Agents and the Semantic Web*, IEEE Intelligent Systems Journal, April 2001.

[14] Dieter A. Fensel: *The Potencial and Limitations of Semantics Applied to the Future Internet*, WEBIST, March 2009.

[15] Tina Eliassi-Rad, Jude Shavlik: *Intelligent Web Agents that Learn to Retrieve and Extract Information*, Intelligent exploration of the web, 2003.

Sono inoltre stati utilizzati in maniera più generica:

Libri

- Brett McLaughlin: *Head Rush Ajax*, OReilly 2006.
- Dave Crane, Eric Pascarello, Darren James: *Ajax in Action*, Manning 2006.
- Thomas Erl: *Service-Oriented Architecture*.

Tesi

- Roy Thomas Fielding: *Architectural Styles and the Design of Network-based Software Architectures*, University of California 2000
- Tina Eliassi-Rad: *Building Intelligent Agents that Learn to Retrieve and Extract Information*, University of Wisconsin, 2001.
- Lorenzo Cavina: *Progettazione e Sviluppo di Applicazioni Web di ultima generazione con AJAX*, Cesena 2005

- Matteo Rossi:*AJAX: da JavaScript a Google Web Toolkit*, Cesena 2007

Siti Web

- http://www.sun.com
- http://www.w3.org
- http://www.w3schools.com
- http://wikipedia.org
- http://www.openajax.it
- http://ajax.asp.net
- http://code.google.com/webtoolkit