



AEG: Applicazione esplorativa di Gigaspaces

Redatto Da:

Alessandro Cossutti

Sommario

AEG: Applicazione esplorativa di Gigaspaces.....	1
1 Introduzione al Documento.....	2
1.1 Novità Introdotte Rispetto alla Precedente Emissione.....	2
1.2 Scopo e Campo di Applicazione del Documento.....	2
1.3 Termini e Definizioni.....	2
2 Ipotesi di Progetto.....	2
3 Campo di Applicazione.....	3
4 Contesto.....	3
5 Obiettivo.....	3
6 Modalità di Esecuzione.....	3
6.1 GigaSpaces Grid Computing.....	3
6.2 Drools Rule Engine.....	4
6.3 Postgres RDBMS.....	4
6.4 HTTP / Servlet Container.....	5
6.5 Ipotesi di Realizzazione.....	5

1 Introduzione al Documento

1.1 Novità Introdotte Rispetto alla Precedente Emissione

Versione/Release:	1.0	Data Versione/Release:	10/11/08
Descrizione Modifiche:	Inizio Stesura Documentazione		
Motivazioni:			

1.2 Scopo e Campo di Applicazione del Documento

Specifica di capitolato d'appalto nell'ambito dell'insegnamento "Ingegneria del Software" del Corso di Laurea in Informatica dell'Università di Padova.

1.3 Termini e Definizioni

Lista omessa. I termini e le definizioni salienti usati nel presente documento sono associati a URL di riferimento e approfondimento.

2 Ipotesi di Progetto

Il progetto proposto si basa sulla possibilità di impiegare a scopo formativo e di esplorazione di fattibilità i concetti di Grid Computing (si veda: Gigaspaces all'URL <http://www.gigaspaces.com>) e di Inferenza (si veda: Drools all'URL <http://www.jboss.org/drools/>) applicati a un caso reale.

3 Campo di Applicazione

Il progetto è applicato alle tematiche di gestione delle risorse umane e più precisamente alla verifica delle timbrature, letture IN/OUT dei tesserini magnetici dei dipendenti, per la applicare le corrette politiche di retribuzione.

4 Contesto

Un importante ente pubblico offre come servizio la gestione delle Risorse Umane (HR) per una rete di entità a livello nazionale.

Questo tipo di gestione e la sua varietà richiede che il sistema sia in grado di fornire una esatta situazione del personale con al massimo 24H di ritardo.

Attualmente il sistema eroga questo servizio per circa 6.000 dipendenti e la mole di calcolo da esso richiesta è impegnativa.

5 Obiettivo

La soluzione da esplorare si propone di predisporre una nuova architettura di sistema dove l'impatto del codice da governare non sia più le migliaia di linee di codice COBOL, ma una semplice interfaccia tramite cui, attraverso semplici specifiche dichiarative, si possa esprimere la modalità di valutazione delle timbrature.

Poiché inoltre ogni singola realtà territoriale può applicare sue proprie deroghe contrattuali, il sistema dovrà essere anche in grado di gestire specificità locali.

La massa di dati da elaborare e l'esigenza di una valutazione giornaliera dello stato dei singoli dipendenti obbliga infine il sistema a continue elaborazioni circa il loro stato.

6 Modalità di Esecuzione

Il progetto si baserà su una serie di componenti Open-Source, ognuna specializzata per soddisfare una specifica area del problema.

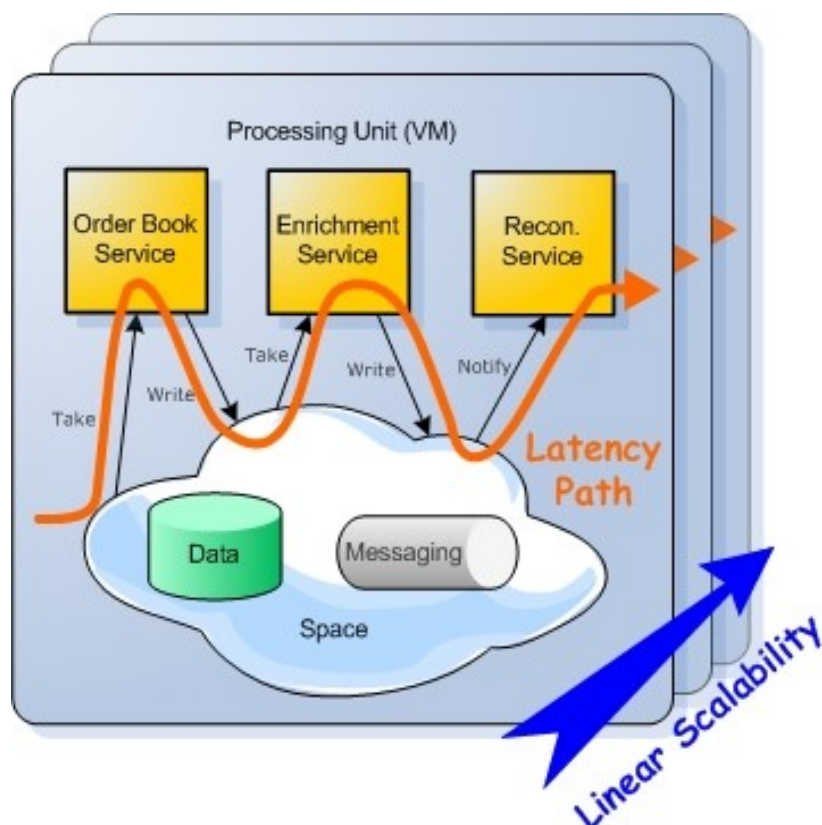
Il progetto realizzato sarà esonibile come caso d'uso del concetto e della tecnologia JavaSpaces presso la comunità OpenSpace.org oppure all'interno del sito GigaSpaces.com.

Costituirà titolo a fini della valutazione della proposta e del prodotto il prevedere e l'effettuare tale esposizione.

6.1 GigaSpaces Grid Computing

Le funzionalità del *grid* sono sfruttate per gestire la grossa massa dati in elaborazione e l'esigenza di gestione dei picchi di calcolo, soprattutto a fine mese.

Il modello funzionale completo che verrà impiegato è quello di seguito rappresentato:



Per qualsiasi ulteriore approfondimento la fonte più affidabile e completa è il wiki presente all'URL
<http://www.gigaspaces.com/wiki/display/XAP66/6.6+Documentation+Home>

6.2 Drools Rule Engine

Drools è un *Rule Engine* dichiarativo basato su algoritmo di RETE per la valutazione delle timbrature dei dipendenti.

Questo motore verrà impiegato per sostituire le attuali regole di verifica basate su un imprecisato numero di clausole IF-THEN-ELSE.

Il Motore di Regole impiegato Drools (si veda all'URL <http://www.jboss.org/drools/documentation.html>) consente di esprimere *pattern* di regole attraverso un linguaggio semplice e attraverso delle *sentence*.

Questa piattaforma impiega l'Algoritmo di RETE III e si basa sul concetto di Fatti & Regole. Inizialmente questo algoritmo fu impiegato per soddisfare le esigenze tipiche dell'intelligenza artificiale e poi come elemento piattaforma di linguaggi come il Prolog. Si veda all'URL: http://en.wikipedia.org/wiki/Formal_grammar

6.3 Postgres RDBMS

Postgres è il RDBMS dove, in via temporanea, verranno caricati i dati delle timbrature dei clienti per costituire la base di calcolo. In esercizio reale, il RDBMS di riferimento è Oracle.

L'RDBMS Postgres è stato scelto per la sua facilità di impiego e la sua

somiglianza da un punto di vista sintattico con il linguaggio SQL supportato da Oracle.

Postgres è attualmente uno dei progetti Open più accreditati come RDBMS per applicazioni del tipo in oggetto. Informazioni dettagliate sono reperibili all'URL <http://www.postgresql.org/>

6.4 HTTP / Servlet Container

Le applicazioni di amministrazione di Drools sono realizzate in linguaggio JSP/DHTML e quindi richiedono necessariamente la presenza di un HTTP *Server* con capacità di *Servlet Container*. Attualmente i prodotti Open tra i più affermati e dotati di tali capacità sono il Tomcat di Apache o Jetty di Codehouse.

La fase di realizzazione del progetto dovrà scegliere la tecnologia ottimale in relazione allo stato e alle necessità.

6.5 Ipotesi di Realizzazione

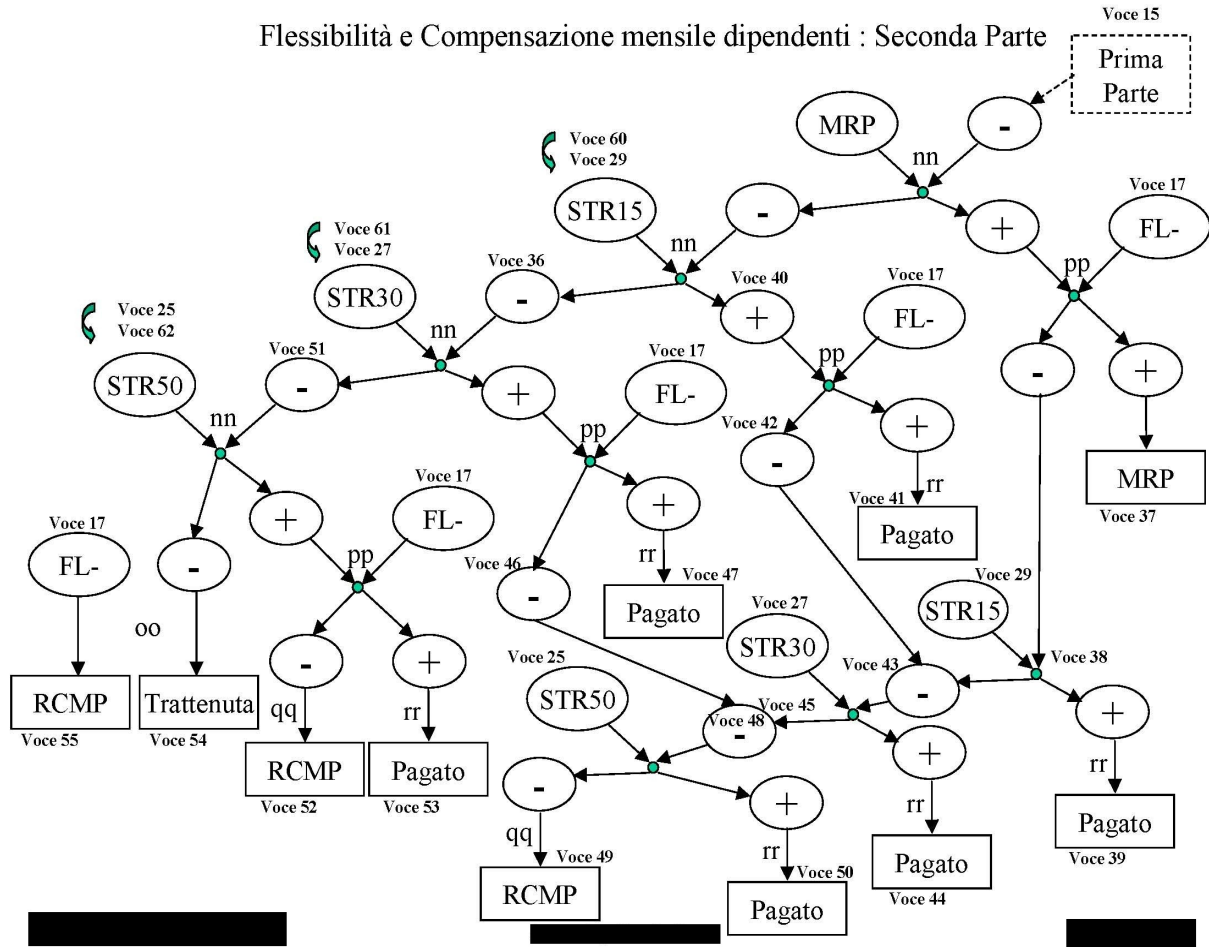
Una ipotetica architettura di soluzione potrebbe basarsi sull'acquisizione da parte del Cliente delle varie Reti di Petri che rappresentano i vari casi di test per le quadrature giornaliere/mensili. Tali oggetti dovrebbero essere trasformati poi in *Rules* all'interno di Drools.

Il Grid viene impiegato per contenere l'esecuzione del Drools fornendo le funzionalità di Data Caching dei dati (si tratta di diversi milioni di *record*) e parallelizzare il processo di quadratura.

Il processo di quadratura dovrà poi generare eventi di segnalazione anomalie in caso di Rules con non vengono soddisfatte.

Di seguito forniamo un esempio di una Rete di Petri per la Valutazione della compensazione in Minuti:

Flessibilità e Compensazione mensile dipendenti : Seconda Parte



Il flusso logico di condizioni da validare, espresso dalla Rete di Petri riportata in figura, dovrà essere trasformato in un insieme di *Rules* grossomodo rappresentabili secondo il seguente esempio:

```

when
    an honest Politician exists
then
    logically assert Hope
when
    Hope exists
then
    print "Hurrah!!! Democracy Lives"
when
    Hope does not exist
then
    print "Democracy is Doomed"
    
```