Università degli Studi di Roma Tor Vergata

Facoltà di Ingegneria

Corso di Informatica Mobile

SLASH

http://mislash.googlecode.com

Professore:
Vincenzo Grassi

Studenti:
Simone Notargiacomo
"Roscio" Tavernese
Ibrahim Khalili

Anno Accademico 2007-2008

Contents

Abstract **2**

1 Introduzione **3**

1.1 Specifiche problema 3

1.1.1 Logica applicazione 3

1.1.2 Ambiente d'uso 4

1.1.3 Lavoro progettuale 5

2 Architettura **6**

2.1 Sistema 6

2.1.1 Composizione Nodi 6

2.1.2 Composizione Rete 8

2.2 Funzionamento 8

2.2.1 Scenario d'uso 8

Abstract

Nel mondo dell'Informatica si è diffuso da molto tempo il bisogno di sviluppare applicazioni per dispositivi mobili, quali notebook, smartphone, tablet ed altro ancora. In particolare si è raggiunta la necessità di sviluppare applicazioni distribuite per dispositivi mobili, ovvero il *Mobile Computing*. Grazie a questo bisogno sono nate molte nuove tecniche che hanno apportato migliorie alla comunicazione tra dispositivi, come la stipulazione di *SLA* (Service Level Agreement: Contratti basati sul livello di servizio) tra richiedenti e fornitori di servizi. Nel corso di Informatica Mobile è stato richiesto di realizzare un sistema che simuli il comportamento di alcuni richiedenti e fornitori dopo aver stipulato un contratto sulla qualità del servizio. In questa relazione verranno trattate le problematiche incontrate nella progettazione del sistema, ed in particolare tutte le scelte implementative effettuate, nonchè i modelli matematici ed empirici. Inoltre verranno presentati anche dei test di esecuzione con i relativi dati sulle performance; infine si riporteranno le conclusioni a cui si è giunti ed i possibili sviluppi futuri.

Chapter 1

Introduzione

Per comprendere a fondo l'entità di tale progetto deve essere riportata la specifica del problema da risolvere, in quanto consente di entrare nell'ottica del problema.

1.1 Specifiche problema

1.1.1 Logica applicazione

Si richiede di progettare una architettura di supporto al monitoraggio e controllo di SLAoooo (2000) in ambiente (possibilmente) mobile. L'architettura del servizio è basata sulla definizione di un certo numero di componenti "logici": SLA Checker (SC), Context Manager (CM), Resource Monitor (RM). Tali entità interagiscono tra loro unicamente tramite il meccanismo DSM ("tuple space"). Il ruolo di tali entità viene descritto come segue:

SLAchecker (SC): data una coppia fornitore/richiedente servizio, che ha stipulato un SLA, SC ha il compito di controllare il rispetto dei parametri del contratto sia da parte del fornitore che del richiedente, e segnalare eventuali violazioni ad entrambi. A questo scopo, SC raccoglie informazioni fornite da opportuni componenti di tipo Monitor presenti sia sul nodo del fornitore che del richiedente, relative a (per esempio):

- tempo di risposta osservato per una richiesta;
- affidabilità (completamento con successo) di una richiesta;
- intervallo di tempo tra due richieste consecutive.

I dati "grezzi" ricevuti dai componenti di monitoraggio vengono elaborati da SC per calcolare i valori degli indici di interesse.

Context Manager (CM): è un componente associato a un particolare nodo di elaborazione e il suo ruolo è quello di fornire informazioni su vari tipi parametri che caratterizzano il contesto di esecuzione di componenti presenti su quel nodo, p.es.:

- utilizzazione cpu;
- RAM disponibile;
- memoria stabile (disco, o altro) disponibile;
- tipo di rete e banda disponibile;
- energia disponibile.

Resource Monitor (RM): un componente di questo tipo fornisce le informazioni relative a una delle risorse elencate sopra.

1.1.2 Ambiente d'uso

L'ambiente in cui si immagina che il servizio di controllo di SLA venga realizzato è costituito, in generale, da una molteplicità di nodi (fissi o mobili) con vari livelli di disponibilità di risorse interne (memoria, cpu, ecc.), connessi tra loro da infrastrutture di comunicazione di varia qualità. Su tali nodi sono in esecuzione componenti che offrono/richiedono servizi. Ogni volta che una coppia fornitore/richiedente stipula un SLA, il controllo di questo SLA viene affidato a un componente SC.

1.1.3 Lavoro progettuale

Si richiede di progettare e realizzare, utilizzando la piattaforma JADE (`http://jade.tilab.com`), l'architettura indicata nella sezione precedente. In particolare, occorre definire una localizzazione dei componenti e organizzazione del modello DSM (basato sulla realizzazione di uno o più "tuple space") che sia adeguata alla esecuzione del servizio di controllo SLA in un ambiente possibilmente mobile, caratterizzato da possibile scarsità di risorse per i componenti in esecuzione su determinati nodi. Il livello di adeguatezza andrà valutato rispetto alla capacità di ottimizzare misure di prestazione quali:

- traffico generato su rete;
- consumo di energia da parte di nodi mobili;
- carico computazionale/di memorizzazione per nodi mobili;

tenendo anche conto del fatto che il contesto (disponibilità di risorse) in cui opera il servizio di controllo SLA può variare nel tempo, per esempio per effetto della mobilità di alcuni nodi.

Chapter 2

Architettura

Nella specifica del problema (rif. 1) è stato riportato il funzionamento del sistema da realizzare ei relativi componenti neccessari. Per adempire alle richieste della specifica si è deciso di sviluppare l'applicazione dando priorità ai punti fondamentali, dopodichè sono stati trattati gli aspetti secondari come la taratura dei parametri ei meccanismi di richiesta dei servizi. Nella prossima sezione verrà riportata l'architettura del sistema.

2.1 Sistema

2.1.1 Composizione Nodi

Tutta l'applicazione è stata portata avanti considerando la presenza di molteplici nodi richiedenti e fornitori, per rendere possibile una simulazione più reale e complessa. Inizialmente sono state prese delle decisioni inerenti la composizione generale di ogni nodo della nostra rete, in particolare si è deciso di supportate nodi di due tipi:

wired: nodo fisso collegato tramite cavo;

wireless: nodo mobile collegato tramite "etere".

Effettuando tale scelta si sono scatenate un'altra serie di decisioni attinenti l'hardware dei nodi, ovvero l'energia, la memoria ram, il disco e il carico

della cpu. Tali componenti sono molto dipendenti dal tipo di collegamento del nodo, in quanto un link di tipo wireless ha bisogno di maggiore calcoli e quindi un utilizzo di energia maggiore.

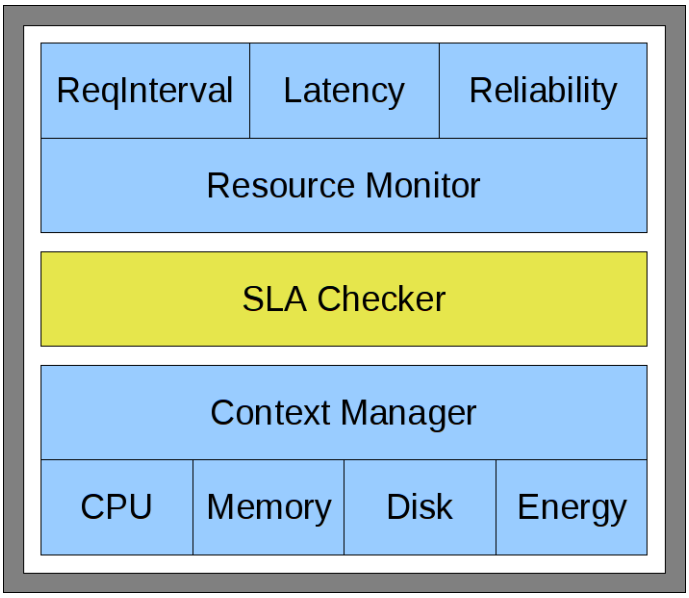

Figure 2.1: Componenti Nodo

Dalla figura 2.1 si può notare come è strutturato un singolo nodo, in particolare si può notare come i componenti hardware principali (cpu, memory, disk, energy) siano in stretto contatto con il *Context Manager*. Componenti quali: latency, reqInterval e reliability sono a contatto con il *Resource Monitor* il quali si occupa appunto di monitorarli. Il componente fondamentale, ovvero lo *SLA Checker* (o *SC*), si trova in genere a contatto con il *CM* e il *RM*. Tutte queste parti elencate saranno chiamate d'ora in poi *Agenti*, considerando che ci troviamo in ambiente di programmazione ad agenti (ovvero *Jade*).

2.1.2 Composizione Rete

La rete che si è deciso di creare è formata da tanti di questi nodi, sia richiedente che fornitore. Nello specifico ogni nodo fornitore si occupa di fornire un servizio generico, il quale viene registrato nelle pagine gialle del sistema. Ogni richiedente, invece, può richiedere il servizio ad uno qualsiasi dei fornitori disponibili. Questa operazione è stata resa possibile per fare in modo che la simulazione si attenesse ad un tipico caso reale. Per quanto riguarda lo *SLA Checker* si è progettato il sistema considerando che tale agente dovesse *migrare* da un nodo all'altro in base alle condizioni del contesto. Un esempio della rete in questione con 3 nodi può essere visto in figura 2.2. Si può notare che lo *SC* si trova solo su uno dei nodi della rete in quanto deve migrare poter migrare fra di loro.

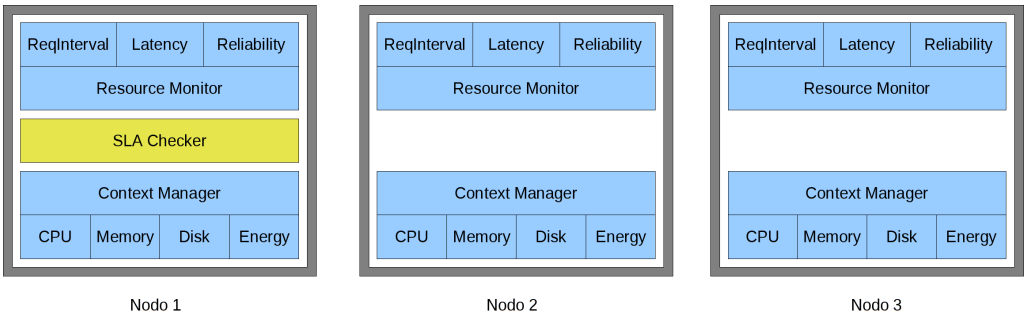

Figure 2.2: Esempio di rete con 3 nodi

2.2 Funzionamento

Per spiegare il funzionamento del sistema in questione si è deciso di definire uno scenario d'uso e quindi descrivere il comportamento dei relativi nodi.

2.2.1 Scenario d'uso

Un tipico scenario d'uso potrebbe essere una rete con 3 nodi di cui:

- 1 nodo fornitore;
- 2 nodi richiedenti;

in particolare ci troviamo in una situazione in cui ognuno dei due nodi richiede un servizio che si attenga al contratto prestabilito con il nodo fornitore. Ogni contratto contiene le seguenti informazioni:

Publisher: fornitore del servizio;

Subscriber: richiedente del servizio;

ReqInterval: intervallo di tempo tra le richieste del richiedente;

Latency: tempo impiegato per espletare il servizio;

Reliability: affidabilità di servizio da parte del fornitore.

Tali informazioni servono per fare in modo che sia il fornitore che il richiedente facciamo il possibile per attenersi ai valori specificati nel contratto. Su uno dei nodi del sistema è presente lo *SLAChecker* che si occupa di controllare la validità di tutti i contratti stipulati. Nel caso in cui le condizioni di un contratto vengono violate da uno dei due nodi allora lo SC informerà immediatamente entrambi i nodi di tale condizione. Tutti i componenti dei nodi sono fortemente dipendenti dalla presenza dello SC, in quanto comporta un aumento oneroso in termini di calcoli. A causa delle risorse limitate di alcuni nodi (come l'energia) è necessario effettuare un controllo sullo stato attuale dei componenti dei nodi per evitare di sovraccaricarlo. In caso di necessità lo SC migra su un nodo con condizioni di carico migliori. La selezione del nodo migliore viene fatta utilizzando una specifica politica di selezione (vedi cap. XXX). Un esempio di migrazione può essere visto nelle due figure seguenti, in cui nel primo caso lo SC si trova sul nodo 1, mentre nel secondo caso lo SC è migrato sul nodo 2 a causa di scarsità di risorse sul nodo 1.

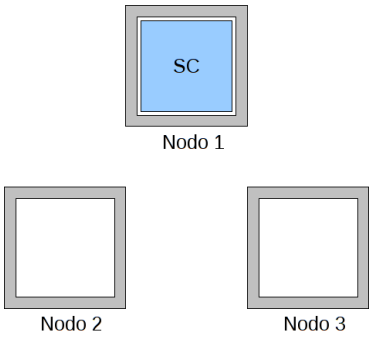

Figure 2.3: Esempio di scenario

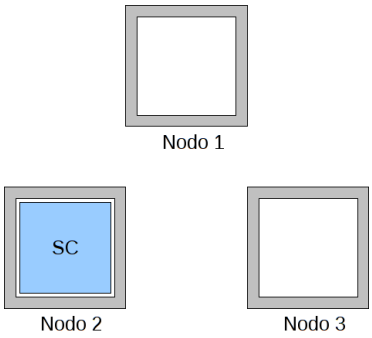

Figure 2.4: Esempio di scenario

List of Tables

List of Figures

2.1 Componenti Nodo 7
2.2 Esempio di rete con 3 nodi 8
2.3 Esempio di scenario 10
2.4 Esempio di scenario 10

Index

Agenti, 7

CM, 7
Context Manager, 7

Jade, 7

migrare, 8
Mobile Computing, 2

Resource Monitor, 7
RM, 7

SC, 7–9
SLA, 2
SLA Checker, 7, 8
SLAChecker, 9

Bibliography

oooo, *test* (test, 2000).