

Modelli per Sistemi Distribuiti e Cooperativi

Introduzione al Corso 2013-14

Docente

- Alessandro Bianchi
 - Dipartimento di Informatica – V piano
 - Tel. 080 544 2283
 - E-mail alessandro.bianchi@uniba.it
 - Orario di ricevimento:
 - mercoledì 15:30 - 17:30
 - URL <http://www.di.uniba.it/~bianchi/>

Il Corso

- Orario
 - Lunedì 16.30 - 18.30 aula ICD – Piano 2
 - Martedì 14.30 – 16.30 aula ICD – Piano 2
- Crediti
 - 4 (T1) + 2 (T2) = 6

Il Contesto (1)

- Caratteristica emergente nei sistemi informatici complessi è quella della **distribuzione** e **cooperazione** tra diversi **agenti computazionali** allo scopo di fornire servizi sempre più *raffinati*
 - La realizzazione di sistemi informatici complessi si sta configurando come **integrazione di moduli indipendenti**, spesso **distribuiti**, ciascuno capace di offrire servizi computazionali in **cooperazione** con gli altri

Il Contesto (2)

- Modifica di paradigma: dalla **computazione** alla **comunicazione**
 - Necessità di **capire** e **modellare** opportunamente il comportamento di sistemi **di comunicazione**, analogamente alla comprensione che si è sviluppata riguardo i sistemi **di computazione**
- La comunicazione è vista come parte integrante della computazione

Il Contesto (3)

- Il corso si concentrerà su sistemi informatici
 - Complessi, sia per struttura che per comportamento
 - Critici, rispetto a qualche obiettivo degli utenti
 - **Distribuiti** e **cooperativi**

Esempi di Sistemi Distribuiti Cooperativi

- Sistemi Grid
 - Insieme di risorse computazionali distribuite
- Mobile Ad-hoc NETWORKS – MANET
 - Reti wireless di computer che **NON** necessitano di infrastruttura fisica fissa
- Sistemi per il file sharing
 - BitTorrent
- Protocolli di sicurezza distribuiti

Motivazioni

- Necessità di trattare **formalmente** la comunicazione
 - tra i processi diversi di una stessa applicazione
 - all'interno di un processo
- Il formalismo è necessario per poter garantire la **Qualità del Servizio (QoS)**
- Formalismi per
 - Specificare
 - Analizzare
 - Verificare

Modellazione (1)

- La realizzazione di sw richiede successive attività di modellizzazione per passare da una rappresentazione del problema da livelli di astrazione più alti a livelli più bassi
- Esistono diversi tipi di modellizzazioni
 - Informali
 - Semiformali
 - Formali

Modellazione (2)

- I modelli non **devono** e non **possono** rappresentare tutto il sistema
 - **Separazione degli aspetti** il più possibile ortogonali fra loro
- Per ogni aspetto di interesse si definisce un modello che:
 - lo rappresenti come concetto “chiave”
 - che astragga da altri aspetti meno importanti

Modellazione (3)

- Per risolvere un problema estremamente complesso
 - lo si divide in diversi livelli di astrazione, affrontati in sequenza, ad esempio: top-down, bottom-up, o combinati
- La scelta del modello è essenziale per poter affrontare problemi complessi e per la qualità della realizzazione
 - Eventuali failure possono avere effetti disastrosi
 - La sicurezza deve essere garantita
 - ...

Modellazione (4)

- È spesso necessario adottare modellizzazioni che favoriscano la validazione dell'applicazione rispetto ai parametri di qualità desiderati

Modellazione (5)

- Nel corso, la modellazione formale ha l'obiettivo di
 - favorire l'analisi di proprietà computazionalmente interessanti
 - NON è orientata allo sviluppo del sistema

Analisi di proprietà computazionalmente interessanti

- L'adozione di modelli formali facilita l'analisi di proprietà particolarmente interessanti dal punto di vista computazionale
 - Deadlock/Livelock
 - Starvation/Liveness
 - Reachability
 - Reversibility
 - ...

Svantaggio della formalizzazione

- Adottare metodi di sviluppo formali:
 - è difficile
 - è impegnativo
 - richiede molto tempo
 - richiede elevate competenze

Obiettivi

- **Fornire**
 - conoscenze culturali su metodi, modelli e tecniche per la specifica, l'analisi e la valutazione della comunicazione
 - capacità di applicare i concetti appresi nella risoluzione dei problemi
- **Stimolare**
 - analisi critica delle conoscenze acquisite

Prerequisiti e Caratteristiche Richieste

- Conoscenze di base fornite nei corsi di Informatica triennale
- Capacità di astrazione e formalizzazione
- Desiderio di applicare le conoscenze per indagare fenomeni che si presentano in pratica

Programma Preliminare

- Presentazione
 - Contesto, problemi, obiettivi
 - Proprietà di interesse
 - Esempi di sistemi distribuiti cooperativi complessi: MANET, Grid
- Reti di Petri
- Abstract State Machine (ASM) e Distributed ASM (DASM)
- Altri formalismi:
 - Comunicazione tra processi sequenziali, mediante il CSP di Hoare
 - Introduzione ai linguaggi GO e ERLANG
- Applicazioni esemplificative

Valutazione (1)

- Scopo della valutazione
 - Verificare
 - l'apprendimento dei concetti
 - le capacità di applicarli per risolvere problemi specifici

Valutazione (2)

- 2 modalità:
 - Tradizionale – Prova Orale sull'intero programma svolto
 - Tesina di approfondimento
 - può consistere in un seminario monografico su un argomento trattato nel corso, oppure nello svolgimento di un caso di studio proposto dagli studenti

Bibliografia (1)

- **Testi**
 - E. Börger, R. Stärk, Abstract State Machine, Springer 2003
 - C.A.R. Hoare, Communicating Sequential Processes, Prentice Hall International, 1985 (disponibile all'indirizzo <http://www.usingcsp.com./cspbook.pdf>)
 - R. David, H. Alia, Discrete, Continuous, and Hybrid Petri Nets, Springer 2003.
 - AA.VV. The Go Programming Language, <http://golang.org/>
 - AA.VV. Erlang Programming Language, <http://www.erlang.org>

Bibliografia (2)

- **Lucidi del corso**
 - disponibili a partire dal sito http://www.di.uniba.it/~bianchi/didattica/2013_14/mod_sist_dist_coop/index.htm
- **Ulteriori riferimenti**
 - Articoli e lucidi citati / distribuiti durante le lezioni