Metodi Formali dell'Informatica aa 2001/2002

Docente Maura Cerioli
Studio 331 (terzo piano lato monte)
Tel. 010 353 6731
cerioli@disi.unige.it

Orario

	Lunedi	Martedi	Mercoledi	Giovedi	Venerdi
11-12	MFI	MFI		MFI	MFI
12-13	MFI	MFI		MFI	MFI
14-15			MFI		
15-16			MFI		

Proposta 1: "saltare" il quarto d'ora intermedio (facendo 90 minuti di lezione senza interruzioni) e finire alle 12.45 lu, ma, gio, ve, iniziare alle 14.30 me.
Proposta 2: Scambiare con Basi di Dati le ore del venerdi e fare dalle 9 alle 11 (volendo 9.30-11.00 senza quarto d'ora).
Soggetta a verifica con gli altri corsi, perché per farlo dobbiamo scambiare anche SE e MG...

Orario finale

Lunedi	Martedi	Mercoledi	Giovedi	Venerdi
11.15-12.45	11.15-12.45	14.30-16.00	11.15-12.45	11.15-12.45

Spiegazioni: nei quarti d'ora adiacenti le lezioni + ora di ricevimento da fissare su vostra proposta
…questo è il primo esempio di come mi aspetto interazione dalla platea e non desidero imporre decisioni "dall'alto"
(il che non vuol dire che assegneremo assieme i voti decidendoli democraticamente per alzata di mano)

Svolgimento del corso e degli esami

Problema: con un corso da 3 unità compresso in un semestre, come facilitare la sopravvivenza dello studente medio?
1 mancano i tempi fisiologici di maturazione della materia
2 assenze anche brevi = uno o più argomenti persi
3 prove intermedie come e dove?

Possibile soluzione 1: avviare più argomenti in parallelo
- risolve ragionevolmente bene 1 e 2, ma non 3
Esami a fine corso (al massimo si può fare un compitino verso Natale)

Possibile soluzione 2: un solo tema alla volta con un compitino alla fine di ogni argomento (che può voler dire da una settimana a 20 giorni a seconda del soggetto)
- risolve ragionevolmente bene 3, ma non 1, né 2
Esami = scrutinio dei compitini (eg, media pesata di n su n+1)

Possibile soluzione 3: …se avete suggerimenti sono i benvenuti

MTI e ASD alg-comp-calc

Chi deve sostenere l'esame di MTI di Scienze dell'Informazione, oppure l'esame di ASD (algoritmi, computabilità e calcolabilità)
Deve mutuare da questo corso di MFI la parte di specifiche e sostenere un esame "ridotto" con me.
Deve mutuare la parte di calcolabilità e complessità da Algoritmi con Moggi e sostenere l'esame della parte corrispondente (secondo semestre, credo) con lui.
Si fa la media dei voti e si registra.

Metodi Formali =?

Si usa nell'accezione di fondamenti rigorosi alle discipline per lo sviluppo di software

migliore migliore

Anche con questa restrizione ci sono molti metodi che ricadono nella definizione (categorie, grafi, metriche…) noi ci esamineremo solo casi per cui sono sufficienti:

- Induzione (usando solo i risultati visti ad LP)
- Linguaggio insiemistico algebrico (idem)
- Logica

Che bisogno ha lo sviluppo di Software di basi rigorose?

Il *real programmer* produce quotidianamente software senza bisogno di metodi, strumenti di supporto e men che meno di rigore e il prodotto finale è ottimo, in tempo, economico e facile da mantenere

Alcuni dati (USA '98) sulle speranze di successo di un progetto di medio-grande dimensione (10^6/10^8 righe di codice)

31% annullati = $\$81*10^9$
40% mai operativi/usati
94% riprogettati =$\$105*10^9$
54% delle *original features* modificati

..e del resto vi chiedereste che bisogno ha la progettazione di un ponte di basi matematiche?

Il parallelo regge fino a un certo punto:

Si può fare anche senza basi matematiche ma con si ottiene…

- Diminuzione dei costi
- Progettazione corrispondente alle aspettative del cliente
- Sicurezza dei risultati

Però…

- Non esiste una realtà fisica della produzione del software (una buona metodologia può migliorare il mondo di applicazione)
- Siamo nell'equivalente dell'era romana, cioè all'inizio dello studio della materia
- Non esiste metodologia consolidata e il ritardo nell'impatto dei metodi formali è circa 20 anni (a meno di forzare la mano con tecnologie e metodologie integrate)

Argomenti del corso

Tema centrale: strumenti per la comprensione esatta di programmi

Suddivisione in ore e ordine di presentazione sono soggetti alle decisioni da prendere

Semantica di linguaggi di programmazione

Stabilisce in modo rigoroso che cosa "fa" un programma in maniera indipendente dalla macchina (mai avuto problemi col C?)

Fornisce una base per decidere se un compilatore/interprete è corretto (contrapposto alla validazione per testing)

Mette in luce pericoli (goto) e dipendenze fra le parti di un programma (riusabilità del codice)

Su linguaggi "didattici" permette di studiare in isolamento l'impatto di una feature

Casi particolari che tratteremo: linguaggi imperativi, nucleo applicativo, modelli e linguaggi per la concorrenza

Specifiche

Introduzione di un linguaggio logico con interpretazione nello stesso dominio della semantica del linguaggio di programmazione; permette di asserire proprietà e verificarne la validità in un dato punto del programma.

Specifica di programmi (cruciale in casi safety critical, forse feasable per componenti)

Contratto utente/programmatore

Per i casi effettivi (=calcolabili) debugging intelligente "per tutti i casi possibili", come contrapposto a testing

Guida allo sviluppo di tests per tutti e soli i casi rilevanti

Tratteremo due casi: asserzioni per linguaggi imperativi e specifiche algebriche di tipi di dato