Ingegneria del software per la sicurezza

Lezione 12 di Sicurezza dei sistemi informatici 1

Docente: Giuseppe Scollo

Università di Catania, sede di Comiso (RG)
Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di Studi in Informatica applicata, AA 2006-7

Indice

revisione e ispezione del software

nella lezione precedente si è visto come l'ingegneria del software fornisca **principi** di progettazione di qualità e **modelli** del processo produttivo, utili al conseguimento di qualità dei prodotti software, incluse le qualità di sicurezza

vediamo adesso alcune **tecniche di controllo**, affermatesi in buone pratiche di ingegneria del software, per assicurare che il software possegga le qualità desiderate; nella prossima lezione il quadro si completa con **tecniche di gestione** per il mantenimento della qualità del software attraverso la sua evoluzione

le **revisioni del software**, sia durante il suo sviluppo che a seguito della rilevazione di difetti, sono una pratica già diffusa negli anni '70, che si è in seguito evoluta nella pratica delle **ispezioni formali**, periodicamente reiterate, ciascuna delle quali consta essenzialmente di due fasi:

fase preparatoria:
- selezione e formazione del gruppo
- attribuzione di precisi **ruoli** agli ispettori,con riferimento al successivo incontro, fra i seguenti: Moderatore, Produttore, Verbalizzatore, Ispettore (ruolo attribuito a tutti, indipendentemente dall'attribuzione o meno di altri ruoli)
- distribuzione dei documenti oggetto dell'ispezione, assieme ad istruzioni relative alla stessa
- ispezione individuale dei documenti da parte di ciascun ispettore

incontro (verbalizzato):
- discussione
- conclusioni
- decisioni

verifica di correttezza del software

solo le tecniche di verifica formale di correttezza del software sono in grado di garantire che esso soddisfi i requisiti (formali) con la **precisione** e il **rigore** della matematica

il loro uso è **necessario**, e in taluni casi reso obbligatorio da norme di legge o da vincoli contrattuali, per lo sviluppo di software cosiddetto **safety critical**, cioè dalla cui correttezza dipende l'incolumità di vite umane, o la sicurezza di grandi impianti, o il riparo da rischi di grande danno economico

ostacoli alla diffusione della verifica formale nella produzione industriale del software:
- insufficiente formazione e bagaglio culturale dei produttori
- incidenza significativa sui costi e sui tempi di sviluppo
- proliferazione di tecniche di specifica e verifica formale
- limitata diffusione di strumenti di supporto alla specifica e verifica formale

per prodotti software di grande complessità, i principi di **astrazione** e **modularità** si applicano tanto allo sviluppo del software quanto alla sua verifica di correttezza, affinché quest'ultima risulti **fattibile**: ritorneremo più avanti su questo punto

collaudo del software

"il collaudo del software può solo rivelare la presenza di errori, non la loro assenza" (E.W. Dijkstra)

nonostante questo limite inerente, il collaudo del software è l'alternativa pratica alla verifica formale

ambiti del collaudo:

- di **unità**
- di **integrazione**
- **funzionale**
- di **sistema**

a ciascun ambito sono appropriate distinte strategie di collaudo

- di particolare interesse ai fini della sicurezza sono le strategie di collaudo di sistema

metodi di collaudo:

- **black-box**
- **white-box**

il **collaudo di regressione** si applica a seguito di **modifiche** al software, per scoprire se queste hanno avuto influenza su funzionalità preesistenti del software che non dovrebbero esserne affette

sviluppo del software basato su modelli

modelli del software:

- descrizioni, a diversi livelli di astrazione, di aspetti significativi del software:
  - requisiti
  - architettura
  - dinamica
  - etc.

spesso espressi da collezioni di **diagrammi**

- notazione standard: Unified Modeling Language (**UML**)
- documentazione presso il sito dell'Object Management Group (OMG)

vantaggi dello sviluppo del software basato su modelli:

- riconoscimento di **tratti comuni**, ad es. pattern, a sistemi diversi
- supporto ad **analisi**, valutazione di alternative, **decisioni**
- **riusabilità** per implementazioni differenti
- **documentazione** ad un livello di astrazione appropriato

sviluppo del software con metodi formali

metodi formali possono essere usati nel processo produttivo del software per:

- la specifica di **requisiti**
- la formalizzazione dell'**architettura** e di **modelli dettagliati** del software
- la (parziale o totale) **automazione dello sviluppo del software** dai modelli
- la derivazione dei **casi di collaudo** dalla specifica dei requisiti
- la **verifica formale** di correttezza

è già stata mostrata la **maturità industriale** di numerosi metodi formali, si consulti ad es. www.fmeurope.org

progressi ulteriori sono da attendersi dallo sviluppo di:

- **standard** e **metodologie** d'uso
- **strumenti** di supporto agli standard e alle metodologie
- **integrazione** di strumenti di supporto all'uso di metodi diversi nelle diverse fasi del processo produttivo

riferimenti

Scollo (2006) :

Introduzione al collaudo del software
Note del corso di Ingegneria del software 1, Comiso (RG)

Scollo (2007) :

Modellazione di architetture software
Note del corso di Ingegneria del software 2, Comiso (RG)

Object Management Group (OMG) :

UML® Resource Page, www.uml.org

J. Warmer & A. Kleppe (2005) :

OCL Center, Object Constraint Language resource site, www.klasse.nl/ocl

Formal Methods Europe :

www.fmeurope.org

CoFI :

The Common Framework Initiative for algebraic specification and development
www.cofi.info