



Introduzione all'ingegneria del software



Cosa si intende per ingegneria

La creazione di soluzioni economiche ed efficienti ...

- ... a problemi pratici
- ... applicando conoscenze scientifiche
- ... per costruire oggetti
- ... con benefici anche per la società



Cosa si intende per software

Dal glossario standard dell'IEEE:

L'insieme di programmi, procedure, regole e documentazione associata, e i dati (relativi all'operatività di un sistema di elaborazione)

Essenza del software



- Complessità
- Conformità
- Modificabilità
- Invisibilità

Brooks, Frederick P., "No Silver Bullet: Essence and Accidents of Software Engineering," Computer, Vol. 20, No. 4 (April 1987) 10-19.

http://worrydream.com/refs/Brooks-NoSilverBullet.pdf

No Silver Bullet

Essence and Accidents of Software Engineering

Frederick P. Brooks, Jr. University of North Carolina at Chapel Hill

Fashioning complex conceptual constructs is the essence: accidental tasks arise in representing the constructs in language. Past progress has so reduced the accidental tasks that future progress now depends upon addressing the essence.

terrify more than werewolves, because they transform unexpectedly from the familiar into horrors. For these, to develop, propagate, and exploit these one seeks bullets of silver that can magically lay them to rest.

The familiar software project, at least as royal road, but there is a road. seen by the nontechnical manager, has something of this character; it is usually in- of disease was replacement of demon nocent and straightforward, but is capable theories and humours theories by the germ of becoming a monster of missed schedules, blown budgets, and flawed products. So we hear desperate cries for a silver solutions. It told workers that progress bullet-something to make software costs would be made stepwise, at great effort, drop as rapidly as computer hardware and that a persistent, unremitting care costs do.

But, as we look to the horizon of a cleanliness. So it is with software engidecade hence, we see no silver bullet. neering today. There is no single development, in either technology or in management technique, that by itself promises even one order-ofmagnitude improvement in productivity, in reliability, in simplicity. In this article, I shall try to show why, by examining both the nature of the software problem and the properties of the bullets proposed.

This article was first published in Information Processing '86, ISBN No. 0-444-70077-3, H.-J. Kugker, ed., Elsevier Science Publishers B.V. (North-Holland) @

f all the monsters that fill the throughs-and indeed. I believe such to be nightmares of our folklore, none inconsistent with the nature of software-many encouraging innovations are under way. A disciplined, consistent effort innovations should indeed yield an orderof-magnitude improvement. There is no

> The first step toward the management theory. That very step, the beginning of hope, in itself dashed all hopes of magical would have to be paid to a discipline of

Does it have to be hard?-Essential difficulties

Not only are there no silver bullets now Skepticism is not pessimism, however. in view, the very nature of software makes Although we see no startling break- it unlikely that there will be any-no inventions that will do for software productivity, reliability, and simplicity what electronics, transistors, and large-scale integration did for computer hardware.



Complessità

- «Software entities are more complex for their size than perhaps any other human construct because no two parts are alike. If they are, we make the two similar parts into a subroutine -- open or closed. In this respect, software systems differ profoundly from computers, buildings, or automobiles, where repeated elements abound.» -- Frederick P. Brooks, Jr.
 - componenti tutte differenti
 - esplosione combinatoriale degli stati
 - la complessità cresce esponenzialmente al crescere della dimensione
- La conseguenza è che un prodotto software è difficile da comprendere e da controllare



Conformità

- Il software è un prodotto dell'uomo: non esistono leggi naturali unificanti
- Il software si deve adattare all'ambiente esterno
 - molte interfacce hardware
 - più utenti con profili differenti
 - processi lavorativi predefiniti
- La conformità forzata aggiunge maggiore complessità





- C'è sempre pressione per il cambiamento
 - il software sembra facile da cambiare
 - il software deve adattarsi in corsa a una realtà in movimento
 - se il software soddisfa i bisogni iniziali allora le richieste di estensione aumentano
 - il software di successo sopravvive all'hardware (in generale all'infrastruttura) per cui era stato sviluppato
- Il cambiamento continuo degrada la qualità iniziale del software





- Il software è un'entità invisibile e non può essere catturato completamente da un'unica rappresentazione geometrica
 - flusso di controllo
 - flusso dei dati
 - dipendenze delle variabili
 - sequenze temporali
- La conseguenza è che ci sono problemi di comunicazione

Chi ha inventato il termine «software engineering»?



- Direttrice del Software Engineering Division del MIT Instrumentation Laboratory
- Ha guidato lo sviluppo del software per la missione Apollo
- Apollo 11 fu la missione spaziale che per prima portò gli uomini sulla Luna (1969)



Software during the early days of this project was...not taken as seriously as other engineering disciplines... I fought to bring [it] legitimacy so that it (and those building it) would be given its due respect...I began to use the term 'software engineering' When I first started using this phrase, it was considered to be quite amusing. It was an ongoing joke for a long time. They liked to kid me about my radical ideas."

Nascita ufficiale dell'ingegneria del software



Convegni NATO (1968 Garmisch, 1969 Roma) in risposta alla "crisi del software"

- progetti in ritardo
- progetti che sfondano i costi preventivati
- sistemi non all'altezza dell'affidabilità richiesta

«L'ingegneria del software è sviluppo multi-persona di programmi multi-versione»

Cosa si intende per ingegneria del software



Dal glossario standard dell'IEEE:

«L'approccio sistematico allo sviluppo, all'operatività, alla manutenzione e al ritiro del software»

Un'altra definizione, più completa:

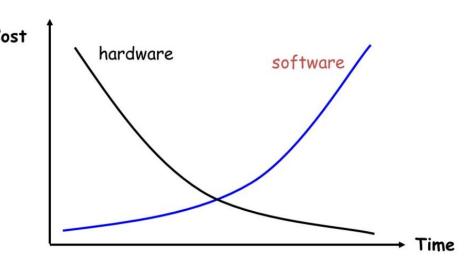
«La disciplina tecnologica e manageriale che riguarda la produzione sistematica e la manutenzione dei prodotti software che vengono sviluppati e modificati entro i costi e i tempi preventivati, e con qualità accettabile» (Fairley, 1985)

R. Fairley, Software Engineering Concepts, McGraw-Hill, New York, 1985



Il fattore costo

 Il rapporto di costo hw/sw si è invertito negli ultimi 50 anni

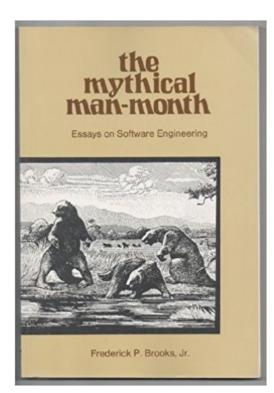


- Il costo del software è un costo di progetto e non di produzione
- Il costo del software è principalmente un costo di sforzo (umano)
 - misurato in [ore | giorni | mesi | anni] persona



Il fattore tempo

- Il "time to market" è il tempo che intercorre dall'idea iniziale al rilascio del prodotto
 - Fattore critico per molti progetti
- Legge di Brooks:
 Aggiungere persone a un progetto già in ritardo ha un effetto contrario alle intenzioni
 - Addestramento
 - Comunicazione intragruppo: n(n-1)/2





- Come definirla?
 - aderenza allo scopo
 - conformità alle specifiche
 - grado di eccellenza
 - la totalità delle caratteristiche di un prodotto software che si basano sulla abilità del prodotto a soddisfare I bisogni espliciti e impliciti (def. Standard ISO)
- Modello di qualità: decomposizione del concetto in attributi fino ad arrivare a misure
 - Standard ISO 9126: funzionalità, affidabilità, efficienza, usabilità, manutenibilità, portabilità





Differenze tra programmazione e ingegneria del software secondo Google



- La programmazione è l'atto immediato di produrre codice
- L'ingegneria del software è l'insieme di politiche, pratiche e strumenti necessari per rendere il codice utile per tutto il tempo necessario e consentire la collaborazione all'interno di un team
 - Tempo

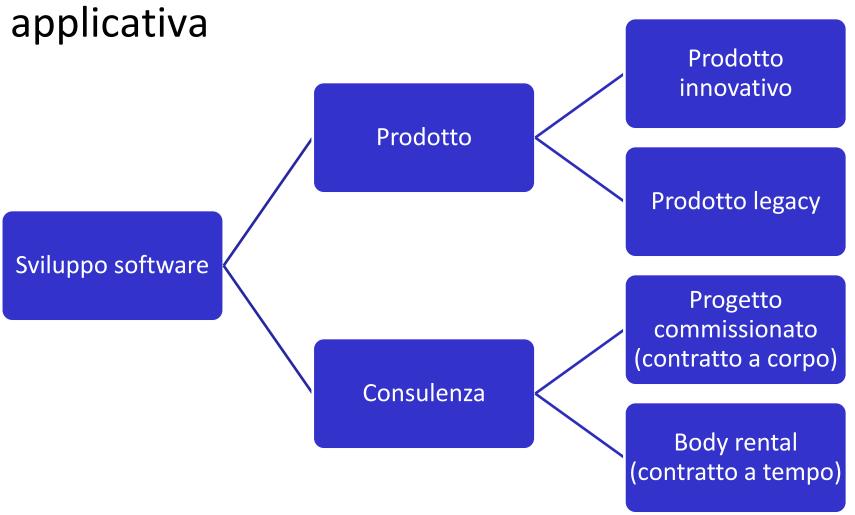
Necessità di preoccuparsi del passare del tempo e dell'eventuale necessità di cambiamento

- Scalabilità
 - Necessità di preoccuparsi della scalabilità, sia del software che produciamo che dell'organizzazione che lo produce
- Decisioni critiche
 - Necessità di prendere decisioni complesse con risultati ad alta posta in gioco, spesso basate su stime imprecise di tempo e crescita

T. Winters, T. Manshreck, H. Wright. Software Engineering at Google. O'Reilly Media. 2020 https://www.oreilly.com/library/view/software-engineering-at/9781492082781/ch01.html



Il ruolo di un ingegnere del software dipende dal tipo di lavoro e dall'area







- Mobile apps
- Web apps
- Information systems (ERP)
- Scientific computing
- Medical devices
- Industrial and process control
- Embedded systems

- Real-time systems
- Big Data
- Defense systems
- System software (OS, DBMS, ...)
- Network software (router, ...)
- Tools (git, ...)
- •



Strategia dell'ingegneria del software

- Separare il processo dal prodotto e concentrarsi sul processo
 - Un processo software descrive quali sono le attività che concorrono a sviluppare un prodotto software e come le attività sono collegate tra loro
- Altre discipline dell'Informatica si concentrano sui prodotti:
 - basi di dati, sistemi operativi, compilatori, reti di calcolatori, sistemi distribuiti, ...