Strumenti per gli informatici: i processi per lo sviluppo software

Storia dell'Informatica
a.a. 2024/25

contenuti

- storia dei processi software
- principali modelli
- movimento agile
- Scrum e HMR

processi software

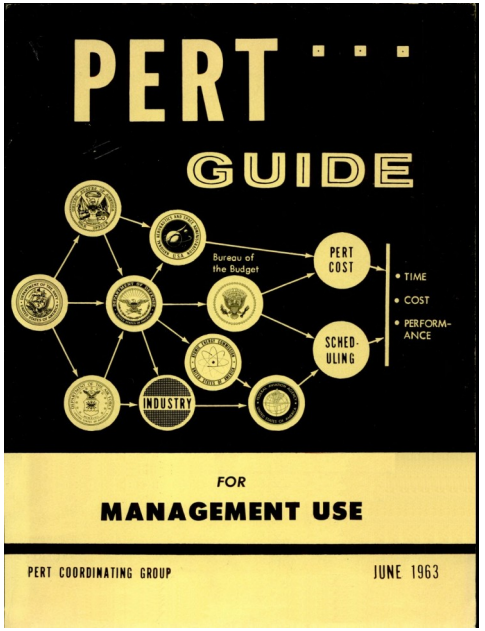
- Definizione di modelli pratici per pianificazione, controllo, analisi e ottimizzazione dello sviluppo di prodotti software
- Perché è utile "modellare sistemi":
  - Analisi
  - Documentazione
  - Valutazione
  - Simulazione
  - Controllo
- I processi organizzativi sono sistemi

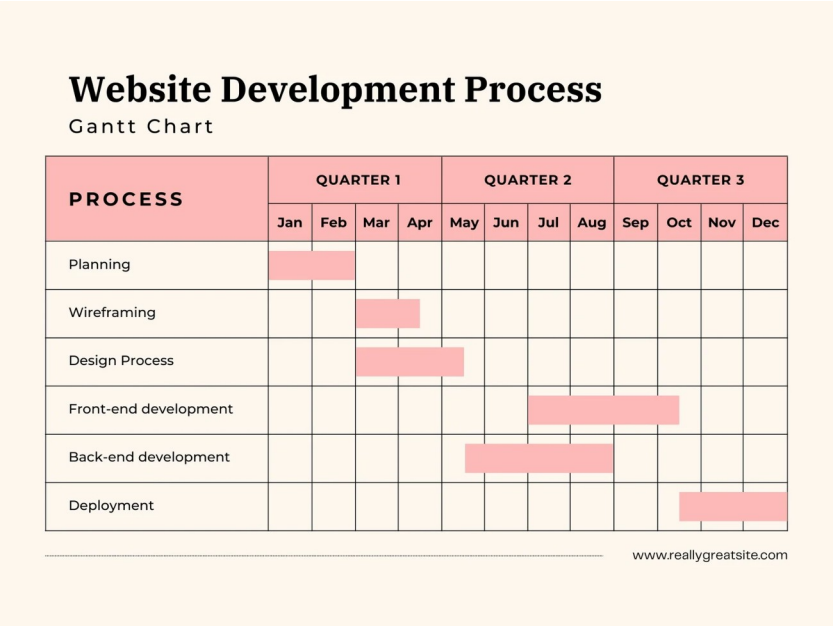
modellazione di processi, inizi

- 1911, Frederick Winslow Taylor: scienza della gestione con *The Principles of Scientific Management*
- 1919, Henry Laurence Gantt: grafici per la pianificazione delle attività con *Organizing for Work*

Scientismo, tecnocrazia,
... taylorismo, fordismo

formalismi e grafici





- 1958, PERT charts: per le dipendenze tra attività, spesso utilizzati coi Gantt

processi per il software

- Negli anni ‘40 e ‘50 le cose si specializzano
- Esempi di modellazione di prodotto e di processo
- 1947, ENIAC/EDVAC, US
  - Goldstine e Von Neumann:
    Progettazione, uso dei flow chart per progettare
- 1949, ESDAC, UK
  - Wilkes e Wheeler:
    Gestione dello sviluppo per parti
    Architettura di prodotto con subroutine
    Istruzione di macchina per la chiamata a subroutine

crisi del software degli anni ‘60

- Hw dedicato VS software su macchine universali
- Le prestazioni hw rendono possibile sw più complessi, con tutti i problemi del caso
- Software come strumenti di sviluppo (sistemi operativi, compilatori, linguaggi…)
  - Si vende solo l’hw e lo sviluppo di applicazioni è un servizio
  - Gli strumenti sembrano inadeguati per la nuova dimensione di complessità delle applicazioni

1968, conferenza di Garmisch

- Sponsorizzata dalla NATO (grande committente)
- Definito il termine "ingegneria del software"
- Problemi:
  - Inadeguatezza (non conformità tra richiesta e prodotto, "non è la cosa che si voleva")
  - Difetti (del prodotto, "è quello che si voleva, ma non funziona")
- Conseguenze:
  - difficoltà nel rispettare le specifiche
  - ritardi (problema per il committente)
  - inefficienze (problema per il produttore)
  - aumento dei costi (colpisce entrambi)

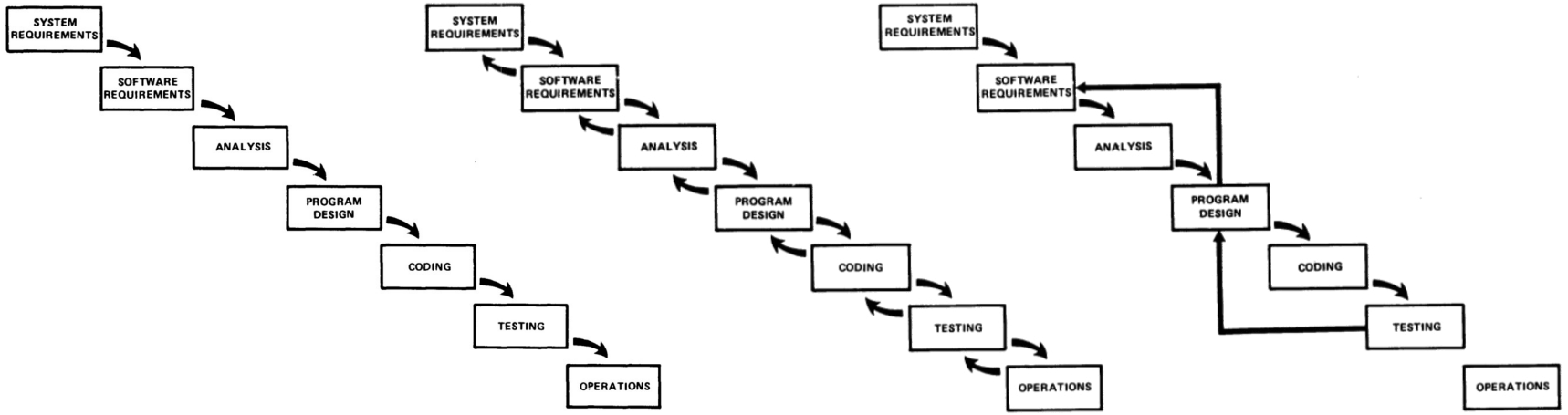
l'informatico è un'anomalia

- Nei processi di produzione industriale:
  - C'è chi progetta e chi realizza, due categorie diverse
  - Le procedure sono soprattutto per i secondi
- Nello sviluppo sw sono tutti ingegneri:
  - Le attività? Risoluzione di problemi
  - Le persone? Istruite e indipendenti
  - Tendenzialmente tutti bravi, orgogliosi e indisciplinati: *cowboy coding*, invece di pianificazione e consapevolezza della necessità di iterare per correggere gli errori

modelli storici

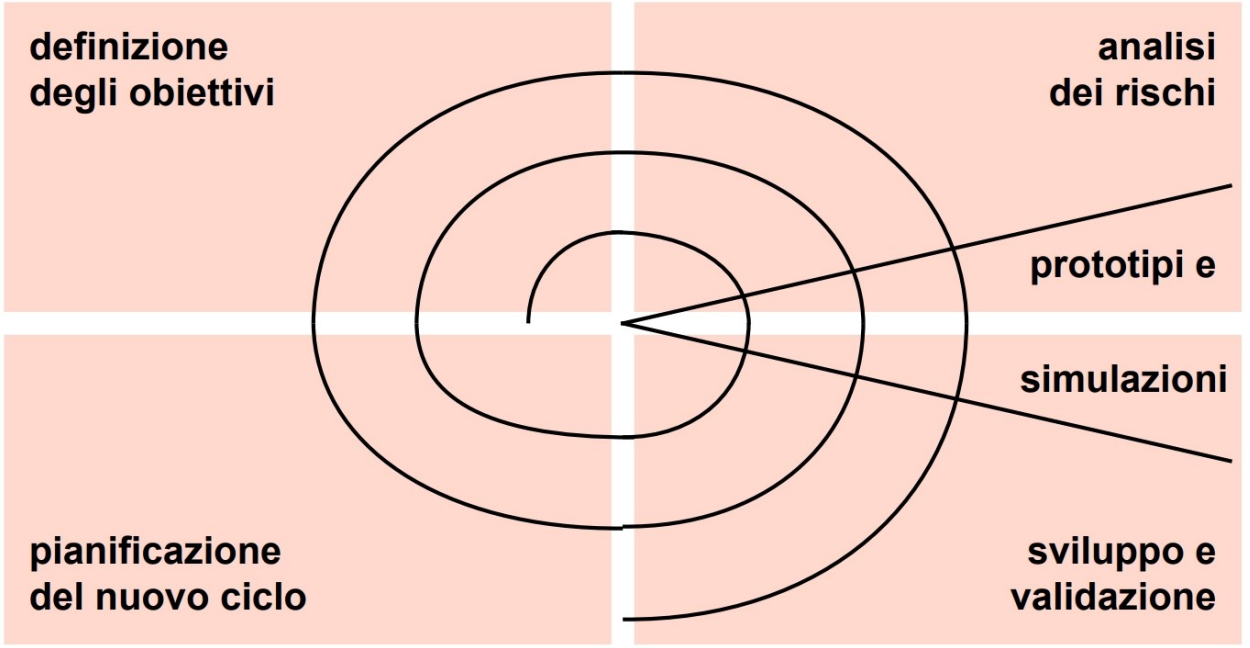
L'importanza dei processi software è stata compresa subito, ma non sempre sono state comprese le proposte

- 1970, Winston W. Royce e il modello a cascata:
  - *Managing the development of large software systems: concepts and techniques* (IEEE WESCON 1970).



modelli storici

- 1986, Barry Boehm e il modello a spirale:
  - A spiral model of software development and enhancement, SIGSOFT Softw. Eng. Notes 11, 4 (August 1986)
  - Ha colto il ciclo di vita e delle versioni del prodotto:

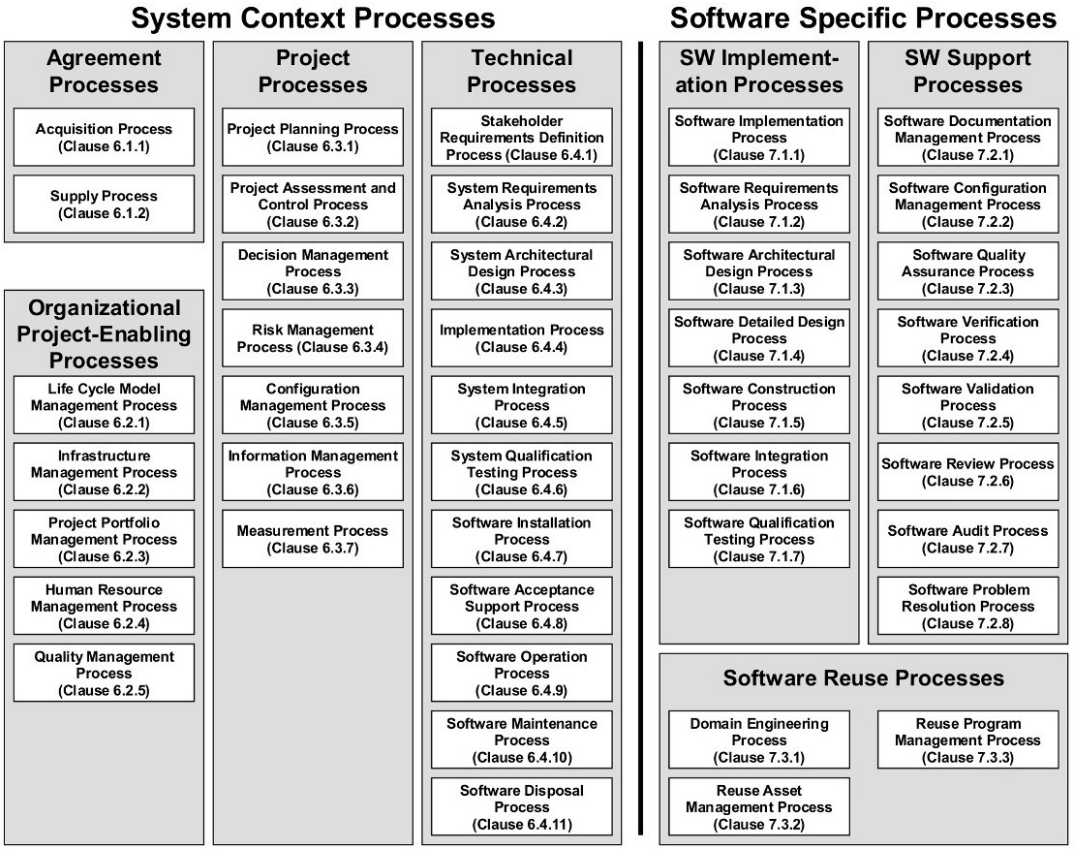


un esempio di successo

- 1987, IBM Cleanroom
  - iterativo incrementale
  - specifiche formali
  - separazione di responsabilità
  - prove con validazione statistica
  - driver, sistemi operativi, sw mission critical
  - ripreso e formalizzato dal SEI 1996 $\rightarrow$ '05
    - usato da NASA, DoD STARS, Ericsson...
    - produttività da 1.7$\times$ a 4.6$\times$

standard importanti

- ISO/IEC 12207 (1995): Systems and software engineering, Software life cycle processes
- '95 → '02 → '04 → '08 → '17



impatto storico

La discussione sui processi software ha promosso:

- Evoluzione dei processi software: da una gestione poco strutturata a modelli definiti e standardizzati.
- Obiettivi: migliorare l'efficienza, ridurre i rischi e garantire la qualità.
- Influenza duratura: principi e buone pratiche stabili e consolidati, che nel tempo trovano formulazioni diverse

problema: troppi standard

Generalmente orientati alla documentazione, esempi (alcuni!):

- 1983, IEEE 839 Std. Software Configuration Management Plans (’90, ’98, ’05)
- 1984, IEEE 830 Std. Software Requirements (’93, ’98, ’09)
- 1984, IEEE 730 Std. Software Quality Assurance Processes (’89, ’98, ’02, ’14)
- 1987, IEEE 1016 Std. Software Design Descriptions (’93, ’98, ’09)

problema: troppi standard

Alcune aree particolarmente gettonate:

- 1983, IEEE 829 Std. Software and System Test Documentation ('98, '08)
  - oggi diventato
    - ISO/IEC/IEEE 29119:2013-16 parti 1-5: concetti, processi,
  - documentazione, tecniche, più una (!?) tecnica (key[word] driven testing)
  - ISO/IEC 20246:2017 per la verifica statica
  - ISO/IEC 33063:2015 per la valutazione dei processi di prova

nel frattempo nell'industria

- Total (= company wide) Quality Control/Management,
  - in ogni processo, a tutti i livelli, continuamente
  - crisi economica, mercato diverso...
    if Japan can, why can't we?
- Mizuno/Shewhart/Deming, dai '50 agli '80, dagli USA al Giappone e ritorno
  - Ciclo di Deming:
    plan $\rightarrow$ do $\rightarrow$ check/study $\rightarrow$ act/adjust
    - vari modelli che vendono l'idea del Ciclo di Deming per avere il controllo della qualità prodotta

tecniche made in Japan

- Kaizen:
  - miglioramento continuo, come atteggiamento
  - coinvolgimento delle persone nelle attività di Deming
- Kanban:
  - Cartellone per la gestione del progresso delle attività
- Muda-mura-muri:
  - futile-irregolare-difficile, sprechi di cui vergognarsi
- 5s (five s, da non confondere con 6σ):
  - seiri-seiton-seisō-seiketsu-shitsuke:
    ordina-mantieni-pulisci-regola-sostieni

altri standard

- A un certo punto, invece di imporre un processo, ci si mette a valutare i processi
- 1988-93, CMM Capability Maturity Model, CMU-SEI, Watts Humphrey (ex IBM)
  - Capability (essere capaci di fare qualcosa)
  - Maturity (quanto bene lo fai)
- SPICE
  - Capability → attività di ISO 12207, quali dei pezzettini di processo utilizzi nella tua azienda
  - Maturity → quanto li segui bene (procedure definite, misure, oggetto di valutazione e riflessione)
- 15504 → 33020

manifesto agile

- Principi base e differenze rispetto ai processi tradizionali:
  - Rifiuto dei troppi standard, al punto di sentire la necessità di un ritorno alla “semplicità”
  - informatici come anomalia (coinvolti e non guidati)
  - influenza culturale delle proposte giapponesi mirate a recuperare la dimensione personale/artigianale del lavoro

tecniche e metodi agili

- Iterazioni, dividere il lavoro in brevi cicli temporali (sprint di Scrum)
- Incrementi funzionali, rilascio graduale di componenti funzionali giù pronte all'uso, utili e testate
- Pair programming, collaborazione e condivisione tra i membri del team di sviluppo (eXtreme Programming)
- Adattamento, revisione sistematica e periodica delle priorità e del processo in base alle necessità
- Qualità, verifiche e validazioni continue

pregi e difetti

- Limitazioni dei processi agili:
  - Difficoltà nella pianificazione a lungo termine
  - Funziona meglio con team di piccole dimensioni (una decina di persone) e richiede molta collaborazione
  - Può essere difficile coordinare numerosi team indipendenti senza una pianificazione centralizzata
- Scenari in cui è più efficace:
  - Team piccoli e multidisciplinari
  - Progetti altamente iterativi, che richiedono molta sperimentazione e rilascio frequente di versioni funzionanti del prodotto

metodi e caratteristiche

- Scrum: lavoro diviso in cicli brevi (sprint)
- Kanban: controllo del flusso di lavoro come un ciclo continuo
- Lean Development: iterazioni rapide, riduzione delle attività non essenziali, priorità a generare valore per il cliente
- DevOps: automazione dei processi, monitoraggio e feedback continui

Scrum

- Nome deriva dalla mischia del rugby (scrum), per rappresentare uno sforzo collettivo verso un obiettivo condiviso
  - 1986, Takeuchi e Nonaka, *The New New Product Development Game*, Harvard Business Review
- 1995, paper di presentazione in un workshop in concomitanza con OOPSLA '95 di Austin (Schwaber e Sutherland)

Scrum

- Consigliato per piccoli team
- Caratteristiche:
  - Divisione in sprint e gestione del backlog
  - Momenti codificati:
    - Review
    - Refinement
    - Planning
    - Retrospective
  - Ruoli:
    - Scrum Master
    - Product Owner
    - Team di sviluppo

Scrum in grande

- Scaled Agile Framework (SAFe)
  - Coordinamento di grandi team usando gerarchie
- Large-Scale Scrum (LeSS)
  - Estensione di Scrum per team multipli
  - Un solo Backlog e Product Owner per tutti i team
  - Sprint sincronizzati
- Nexus
  - Per team fino a 9 gruppi Scrum
  - Un Integration Team per gestire le dipendenze tra i team, la coerenza tecnica e l'integrazione del codice

anche HMR usa Scrum

- Adattamento di Scrum al progetto MR-VR
- Gestione di progetto rilassata
- Documentazione con html strutturati, (per semplicità)
- Analisi dei risultati (con script python)

riferimenti

- Schwaber, Ken. 1997. *"SCRUM Development Process."* In Business Object Design and Implementation, edited by Jeff Sutherland, Charles Casanave, John Miller, Patricia Patel, and Glenn Hollowell, 117–134. London: Springer
- V. Ambriola, G., Cignoni, R. Conradi, M.L. Jaccheri. 2003. *"A Framework for Comparing Software Processes off the Shelf (SPOTS)"*, atti del Software Process Simulation Modeling, PROSIM2003 Workshop – International Conference on Software Engineering (ICSE), Portland Oregon