Modelli di processo per lo sviluppo del software: modelli lineari e modelli iterativi

Prof. Paolo Ciancarini
Corso di Ingegneria del Software
CdL Informatica Università di Bologna

Obiettivi di questa lezione

- Cos'è un **processo software**
- Cos'è un **modello di processo software**
- Modelli lineari
- Modelli iterativi

Nella prossima:

- Modelli agili
- Modelli orientati alla qualità
- Modelli open source

Discussione

- Se dovete creare un sistema di 100 KLOC,
  - Quante *persone* occorrono? Per quanto tempo?
  - Cosa debbono fare?
  - Come le organizzate?
  - Quali **documenti** debbono produrre?

One Person
Very Little Coordination
and Communication
Overhead

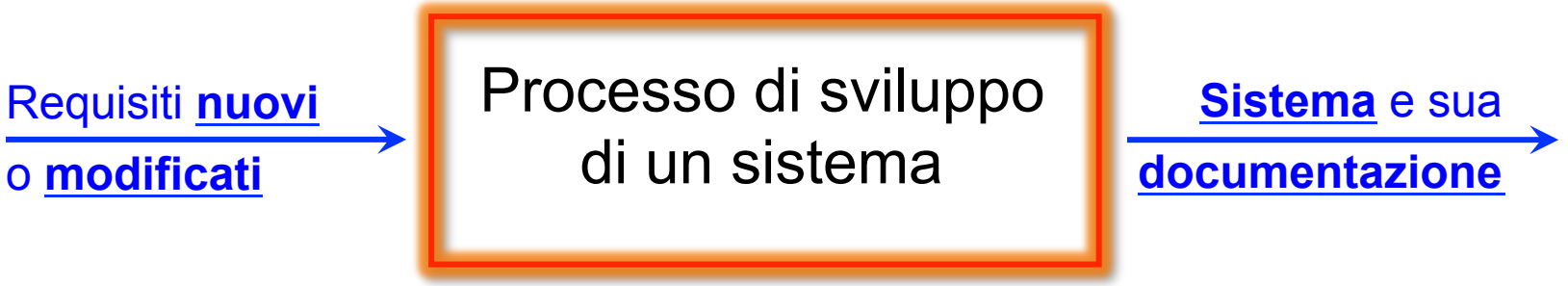
Three People Communicating
and Coordinating Tasks

Il team include più ruoli

- Lo sviluppo in team è molto diverso dallo sviluppo “personale”
- Nel team ci sono persone con esperienze diverse, che ricoprono diversi **ruoli**:
  - Come progettare il prodotto software (architetti)
  - Come costruire il prodotto sw (programmatori)
  - A cosa serve il prodotto sw (esperti di dominio)
  - Come va fatta l’interfaccia utente (progettisti di interfaccia)
  - Come va controllata la qualità del prodotto sw (testatori)
  - Come usare le risorse di progetto (project manager)
  - Come riusare il software esistente (gestori delle configurazioni)

Cos'è un processo di sviluppo

Un processo di sviluppo definisce **Chi** fa **Cosa**, **Quando**, e **Come**, allo scopo di conseguire un certo risultato



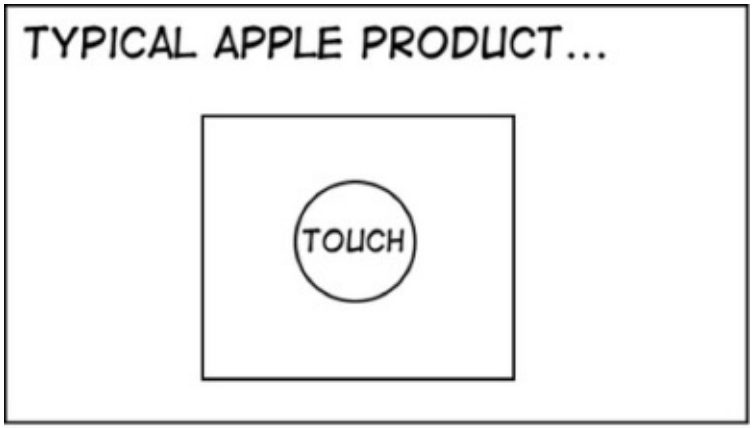
Perché studiare il processo di sviluppo del sw?

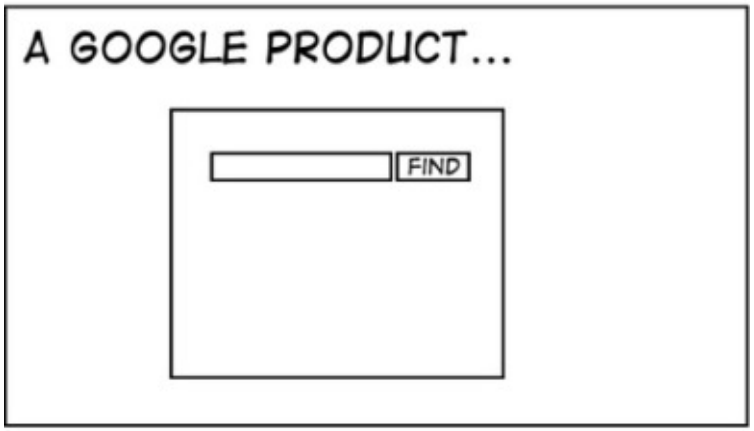
- I sistemi software che costruiamo devono risultare affidabili e sicuri: il processo di sviluppo del software influenza tali **qualità**
- Esistono parecchi modelli di processi software, adatti a prodotti, organizzazioni e mercati **diversi**
- Alcuni **strumenti** sw di sviluppo sono efficaci solo nell'ambito di processi specifici
- Il processo di sviluppo del software impatta l'organizzazione che lo sviluppa
- L'organizzazione che esegue lo sviluppo impatta la struttura del prodotto **(legge di Conway)**

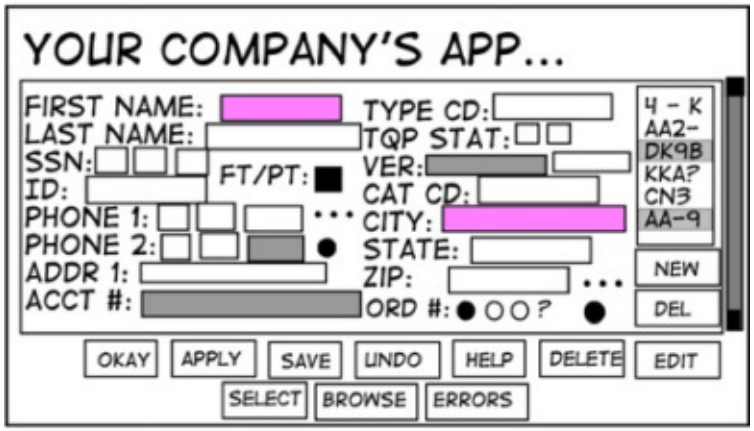
Legge di Conway

Le organizzazioni che progettano sistemi ne progettano la struttura riproducendo le proprie strutture comunicative (es. l'organigramma)

Esempio: se 4 team collaborano a costruire un compilatore, la sua struttura finale sarà su 4 processi in pipeline







STUFFTHATHAPPENS.COM BY ERIC BURKE

Processo software

- Un processo di sviluppo del software (o “processo software”) è un **insieme di attività** che costruiscono un **prodotto** software
- Il prodotto può essere costruito **da zero** o mediante **riuso di software esistente** (asset) che viene modificato o integrato

Programming in the small/large/many

- Programming in-the-small:
  un programmatore, un modulo = edit-compile-debug
- Programming in-the-large:
  costruire software decomposto in più **moduli**,
  su più **versioni**, su più **configurazioni**
- Programming in-the-many:
  costruire **grandi** sistemi sw richiede la cooperazione ed il coordinamento di più sviluppatori, nell'ambito di un ciclo di vita

DeRemer F., Kron, H. 1975. Programming-in-the large versus programming-in-the-small.
In *Proceedings of the international conference on Reliable software*. ACM, New York, NY, USA, 114-121.

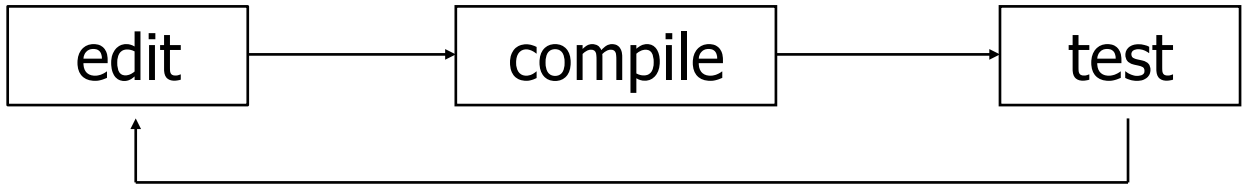
Processo semplice: “programma e debugga”

Tradurre un progetto in un programma e rimuovere gli errori dal programma

- **Programmare è un attività personale**
  di solito non si definisce una sequenza generica di passi “creativi” della programmazione

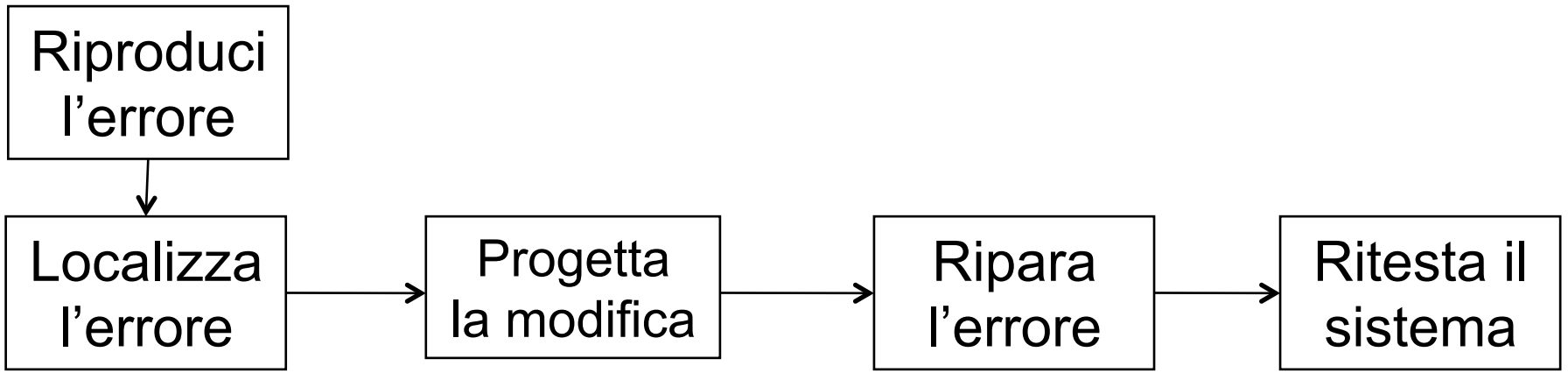
- **Il debugging è una forma di verifica**
  per scoprire e rimuovere errori nel programma

Il processo edit-compile-debug

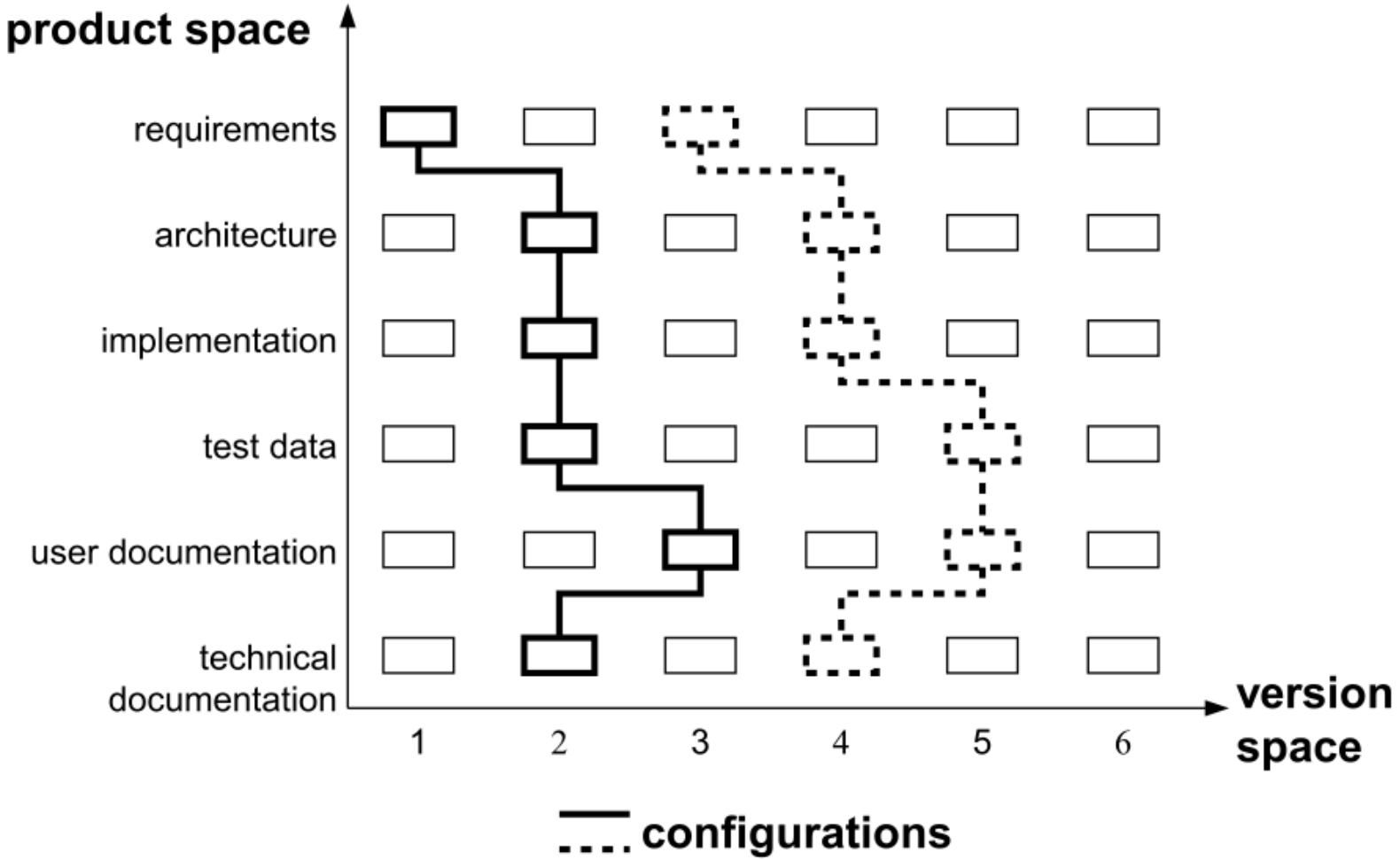


- **Molto veloce, feedback rapido**
- **Disponibilità di molti strumenti**
- **Specializzato per la codifica**
- **Non incoraggia la documentazione**
- **Il processo non scala: in-the-large, in-the-many**
- **Ingestibile durante la manutenzione**
- **Il debugging ha bisogno di un processo specifico**

Il processo del debugging



Programming in-the-large: moduli, versioni, configurazioni



Versioni e configurazioni

- Configuration Item: Un “elemento atomico” (modulo) gestito nel processo di gestione delle configurazioni
  - non solo file sorgenti ma tutti i tipi di documenti
  - in alcuni casi anche hardware

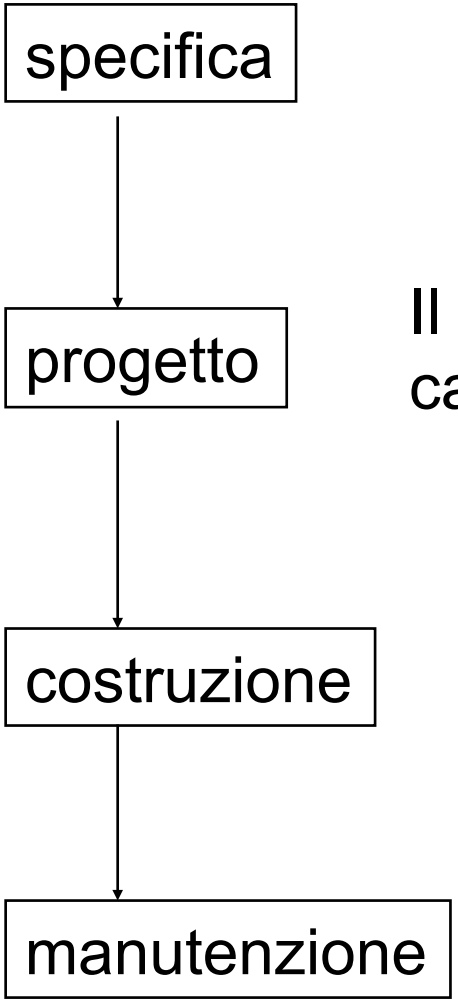
- Versione: stato di un configuration item in un determinato istante di tempo

Programming in-the-large: baseline e release

- Baseline: Una specifica o un prodotto che è stato revisionato e approvato formalmente dal management
  - base per ulteriori sviluppi
  - può essere modificato solo mediante procedure formalmente controllate

- Release: “Promozione” di un baseline resa visibile anche al di fuori del team di sviluppo (per es al cliente)

Segmentare il ciclo di vita



È la fase di stesura dei **requisiti** e di descrizione degli scenari d'uso

Il progetto determina un'**architettura software** capace di **soddisfare** i **requisiti** specificati

La costruzione, o ***codifica***, è una fase complessa che include il ***testing*** e termina con il ***deployment*** del sistema

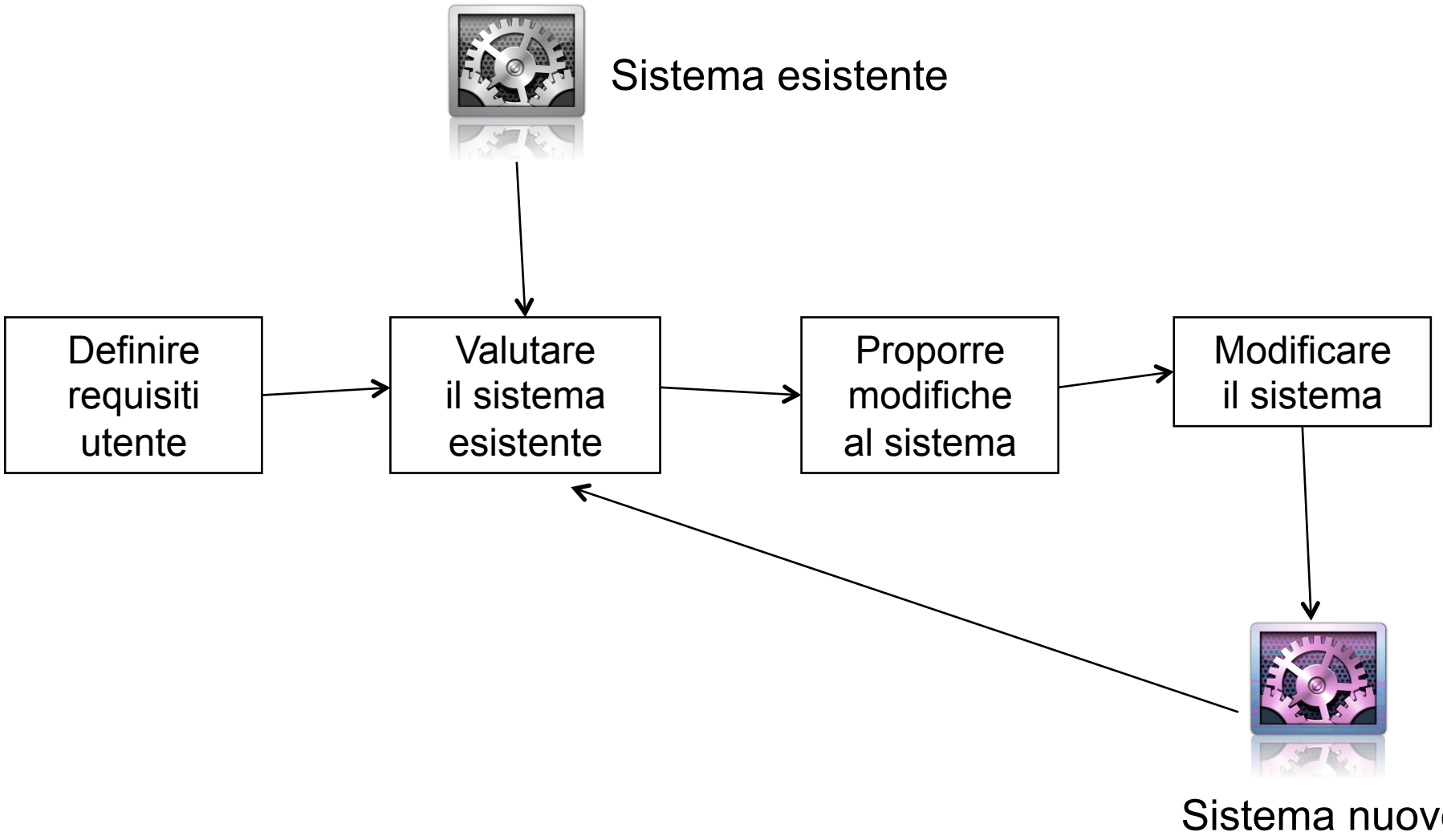
Manutenzione **perfettiva**
Manutenzione **correttiva**
Manutenzione **adattiva**

L’evoluzione del software

Il software è flessibile e può cambiare

- I requisiti cambiano perché cambia l’ambiente della attività, quindi il software che sostiene tali attività deve evolvere
- Anche se si può distinguere lo sviluppo dall’ evoluzione (manutenzione) la demarcazione tra le due fasi è sfumata, perché pochi sistemi sono del tutto nuovi

Esempio


Sistema esistente
Definire requisiti utente
Valutare il sistema esistente
Proporre modifiche al sistema
Modificare il sistema
Sistema nuovo

Modelli di processo

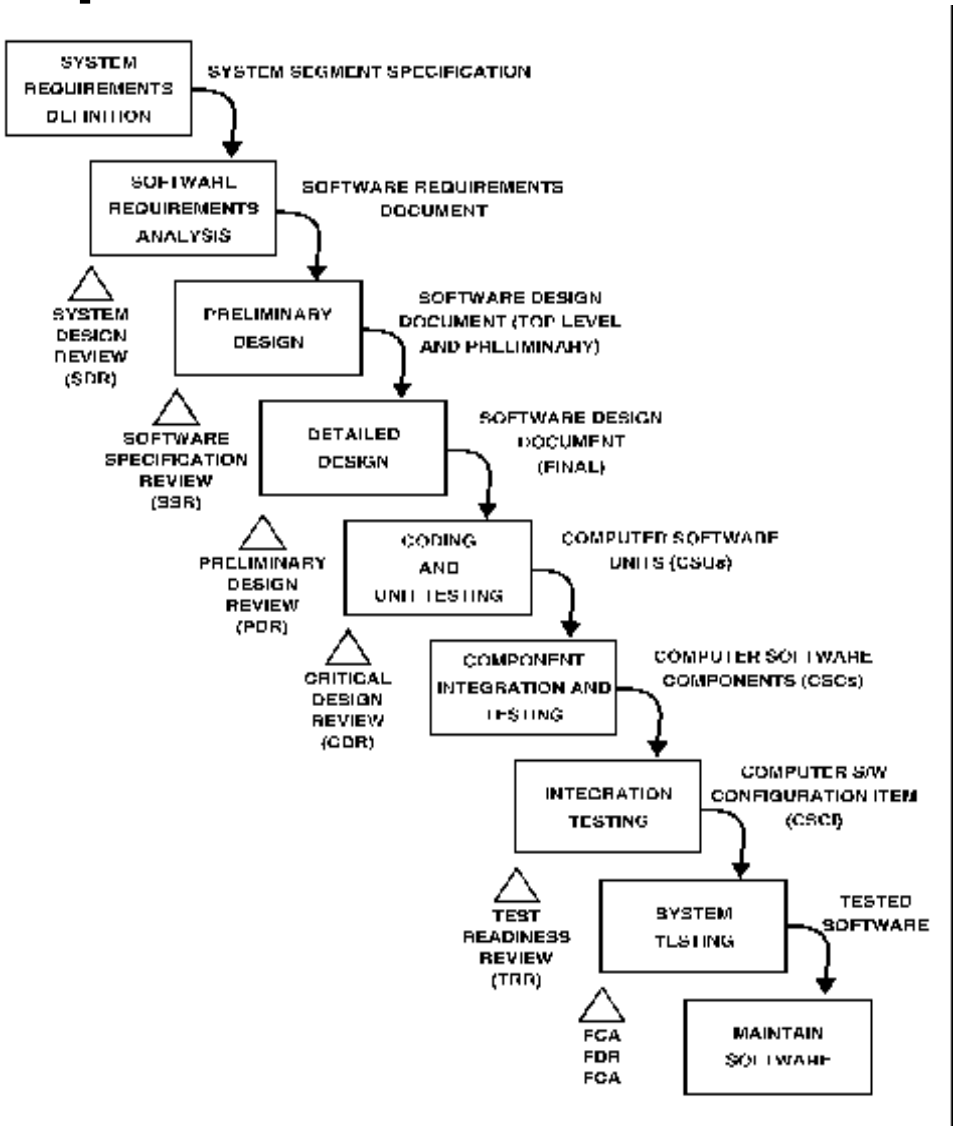
- Un processo sw è un insieme strutturato di attività necessarie per sviluppare un sistema sw:
  - i **ruoli**, le **attività** e i **documenti** da produrre

- Un modello di processo sw è una **rappresentazione** di una **famiglia di processi**
  - Fornisce una descrizione da prospettive particolari
    - per catturare caratteristiche importante dei processi sw
      - utili a diversi scopi, ad esempio per valutarli, criticarli o estenderli

Esempio

- **Modello di processo**: **waterfall**, cioè pianificato lineare

- **Processo**: istanza di un modello waterfall che viene “eseguita” per creare un sistema
  - crea una serie di artefatti come prescritto dal modello



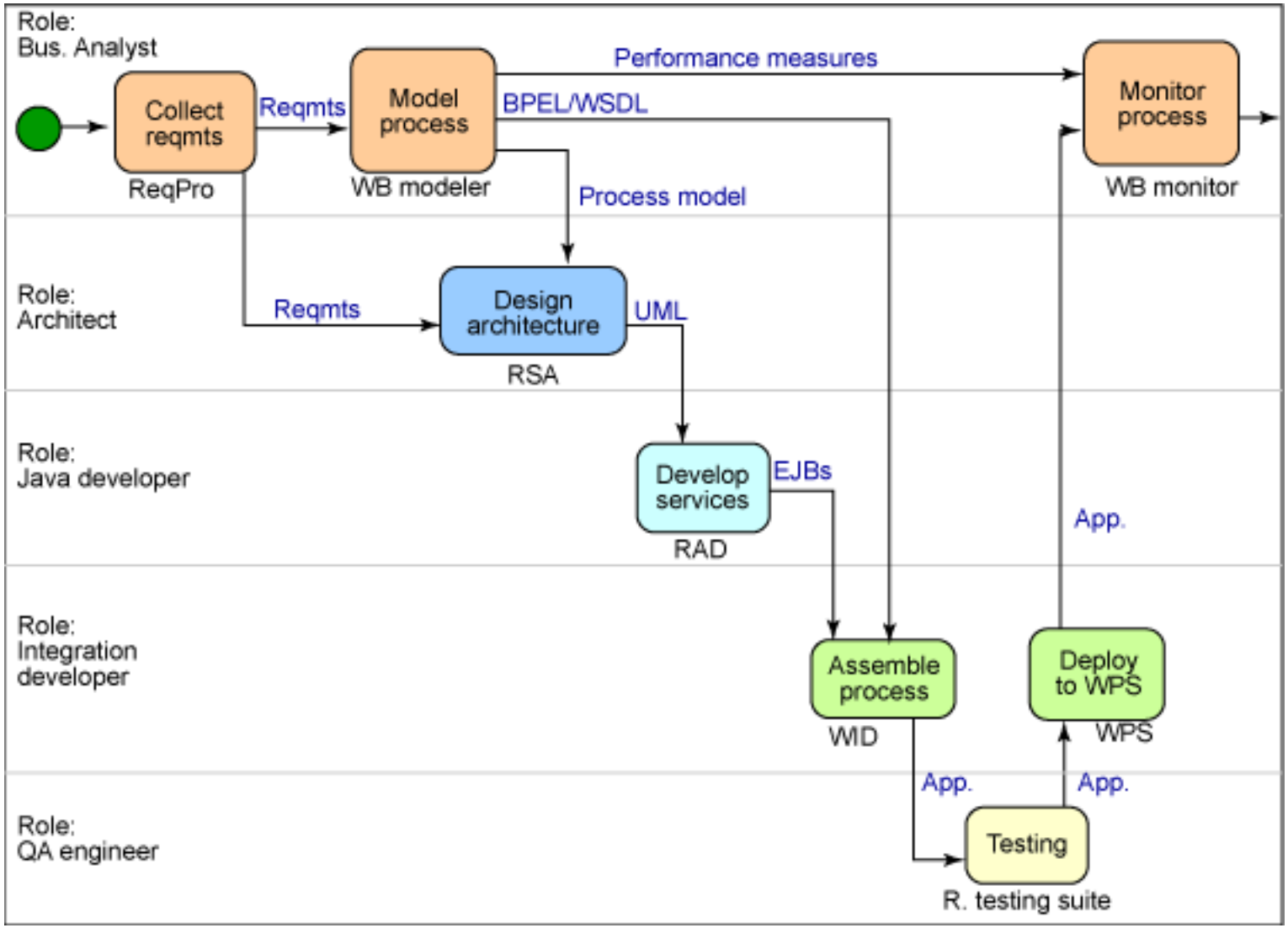
Descrivere un processo

- Occorre descrivere/monitorare le **attività**
- Occorre descrivere/assemblare gli **strumenti**
- Occorre descrivere/assegnare i **ruoli**
- Occorre descrivere/controllare i gli **eventi**
- Occorre descrivere/validare i **documenti**
- Occorre descrivere/verificare i **criteri di qualità**

- *Software processes are software, too*

  L. Osterweil. 1987. Software processes are software too. In Proceedings of the 9th international conference on Software Engineering (ICSE '87). IEEE Computer Society Press, Los Alamitos, CA, USA, 2-13.

Esempio: ruoli e strumenti in un processo a cascata (IBM WebSphere)



Perché descrivere un processo sw

Processo software:

- *L'insieme strutturato di attività, eventi, documenti e procedure necessari per la costruzione di un sistema software*

Benefici della modellazione dei processi sw:

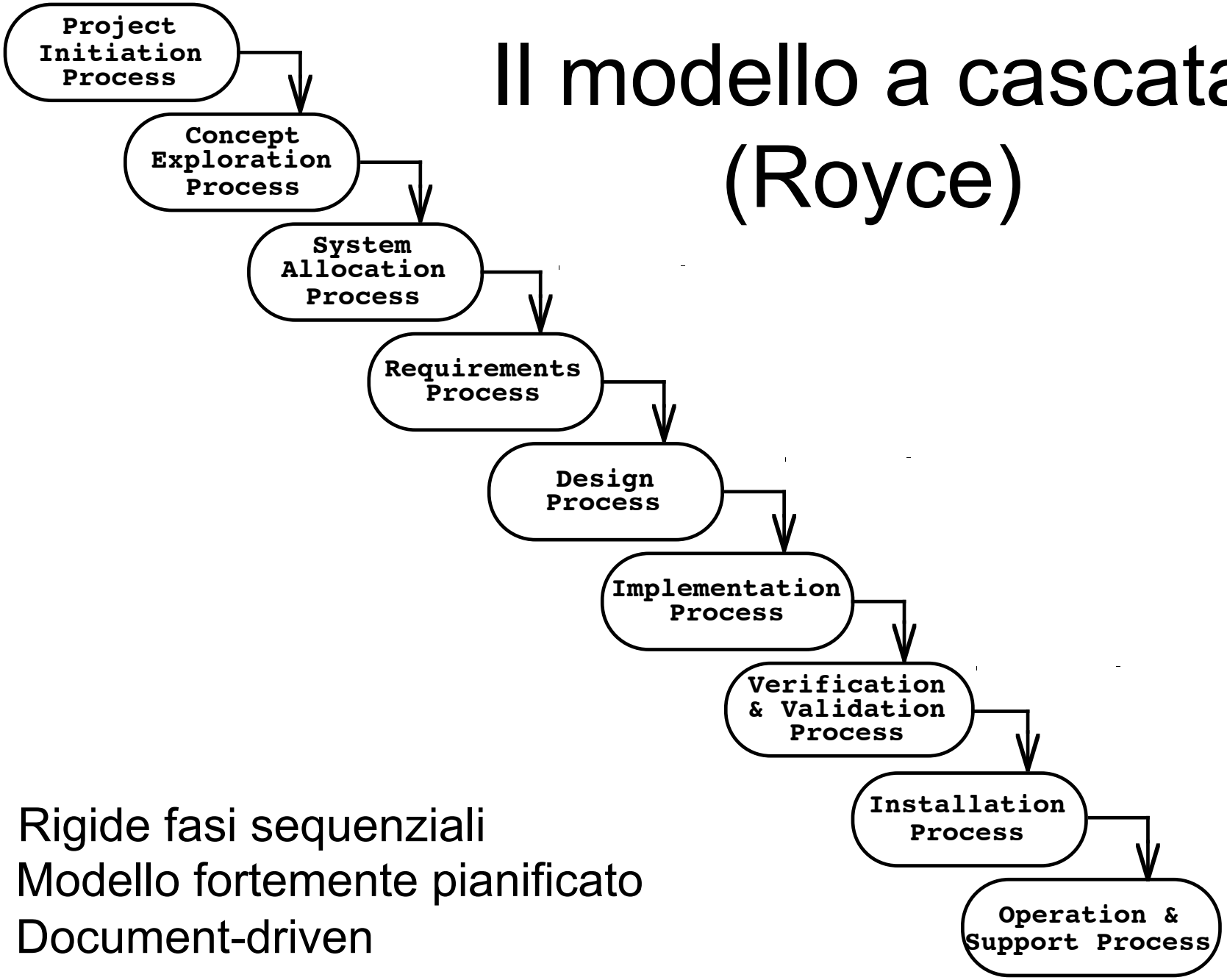
- "Migliora il processo per migliorare il prodotto"
- Miglior coordinamento del team di sviluppo
- Accumulazione di esperienza
- Aderenza agli standard internazionali

Modelli generici di processo sw

- **Modello a cascata (esempio: Waterfall)**
  - Specifica e sviluppo sono separati e distinti
- **Modello iterativo (esempio: UP)**
  - Specifica e sviluppo sono ciclici e sovrapposti
- **Modello agile**
  - Non pianificato, guidato dall'utente e dai test

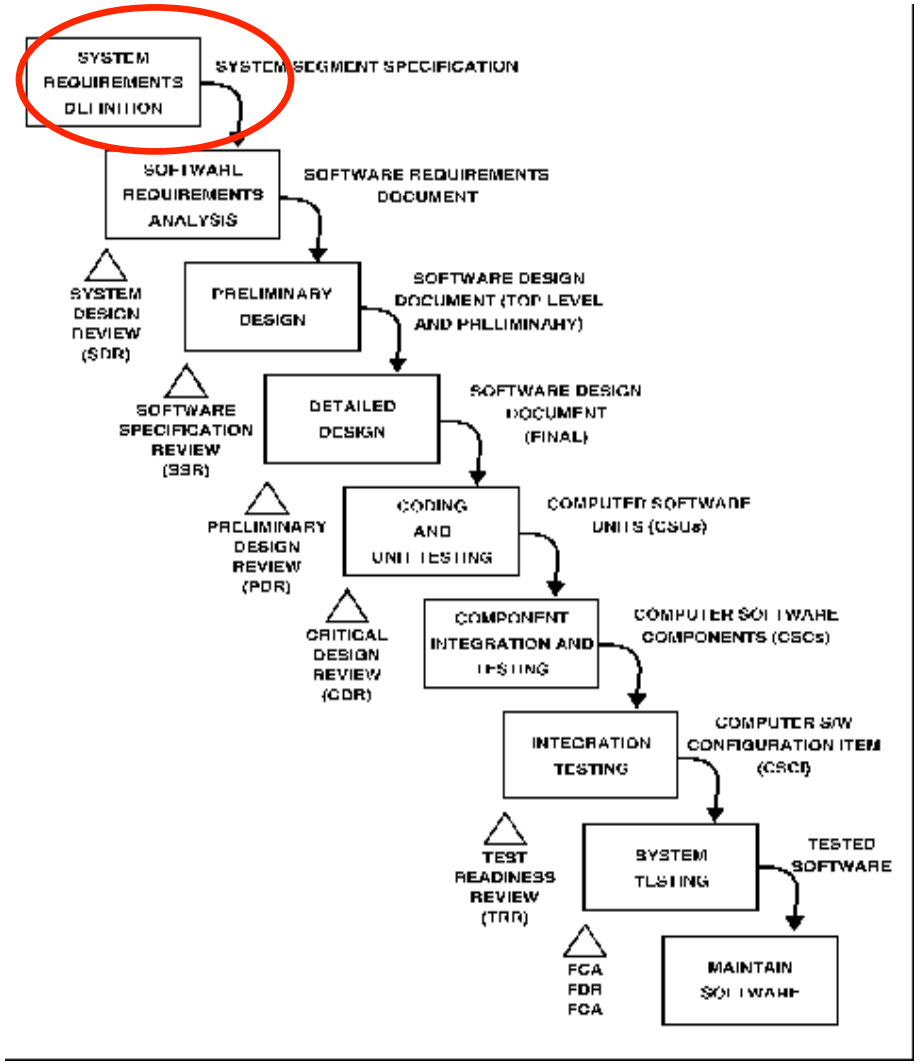
- **Sviluppo formale (esempio: B method)**
  - Il modello matematico di un sistema viene trasformato in un'implementazione

Il modello a cascata (Royce)



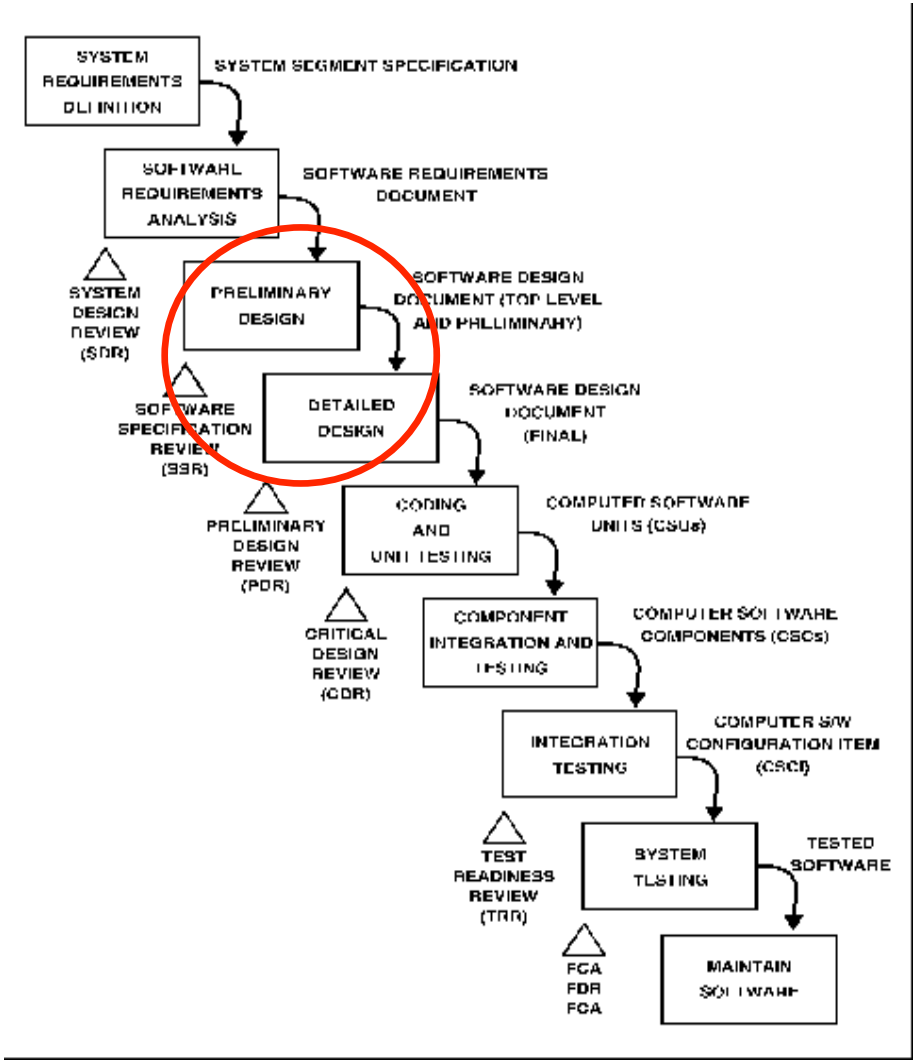
- Rigide fasi sequenziali
- Modello fortemente pianificato
- Document-driven

Analisi dei requisiti



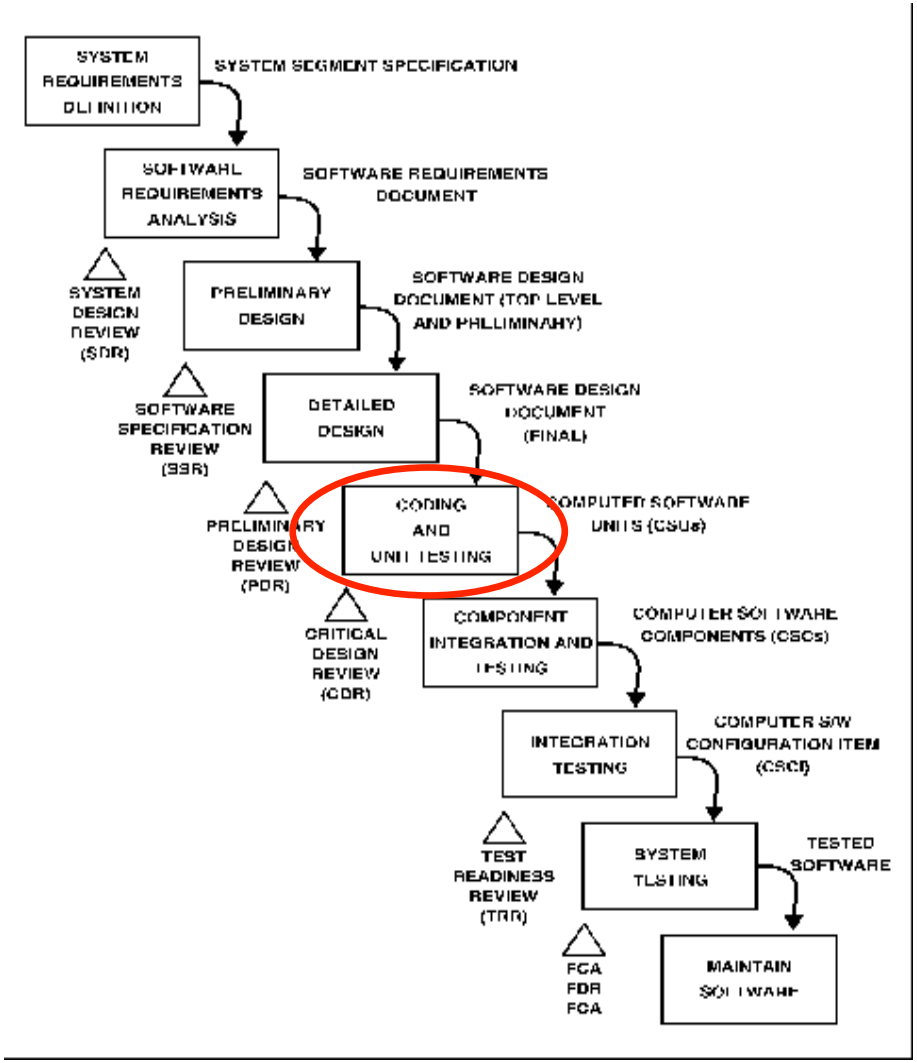
- Utenti e sviluppatori sw
- Successo se:
  - il sw soddisfa i requisiti
  - i requisiti soddisfano i bisogni come percepiti dall'utente
  - I bisogni percepiti dall'utente riflettono i bisogni reali
- Prodotto di questa fase (*deliverable*):
  - documento su cosa il sistema deve fare (non come)

Progetto



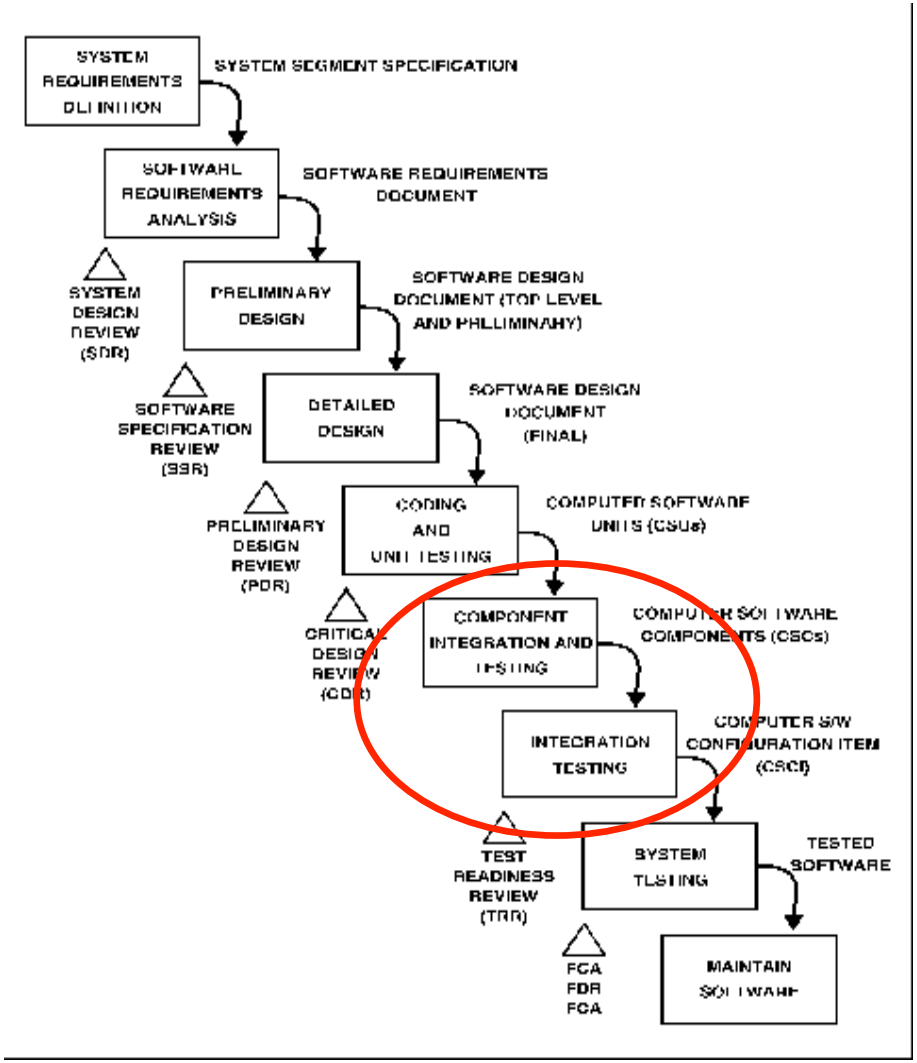
- I progettisti sw usano i requisiti per determinare l'architettura del sistema
- Diversi approcci e metodologie
  - Strumenti
  - Linguaggi di programmazione
  - "librerie"
- Risultato di questa fase:
  - documento di progetto del sistema che identifica
    - tutti i moduli
    - le loro interfacce

Codifica



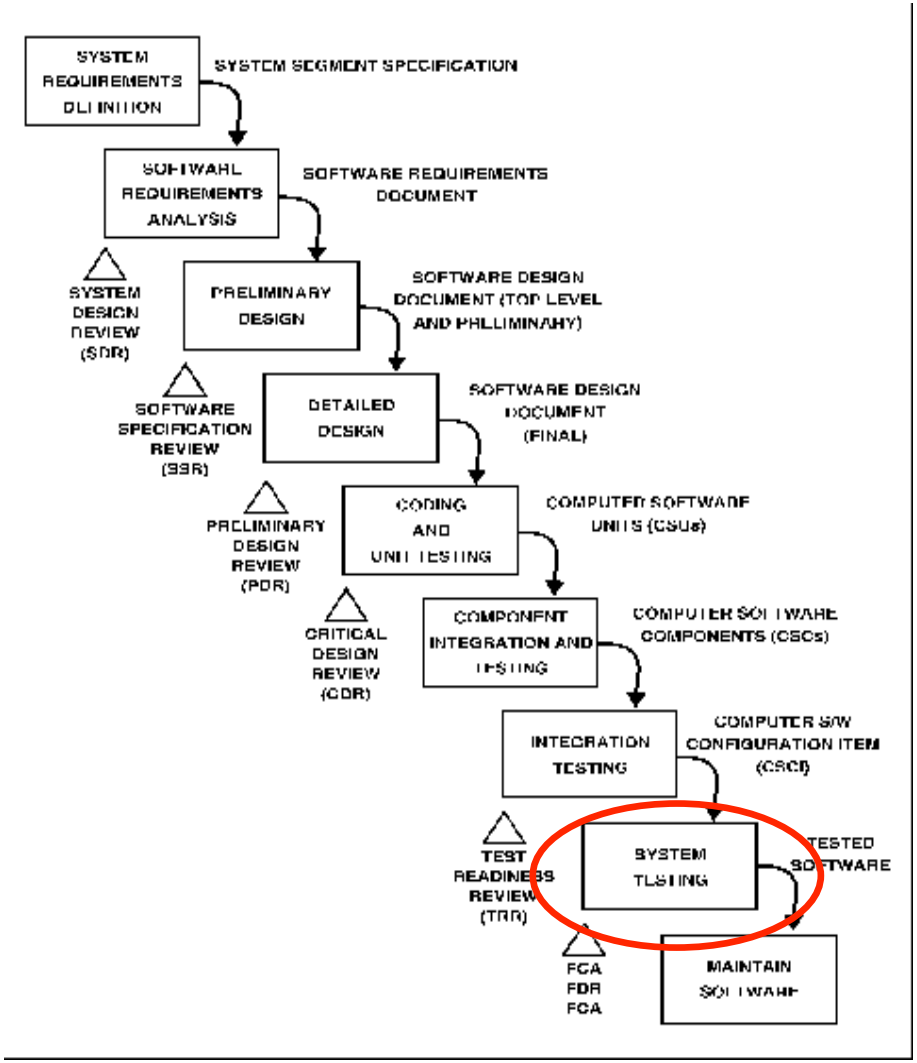
- I singoli moduli sono *implementati* secondo le loro specifiche
- Si usa qualche *linguaggio di programmazione*
- Il singolo modulo è *testato* rispetto alla propria specifica
- Risultato di questa fase:
  - Codice dei moduli
  - Test dei singoli moduli

Integrazione



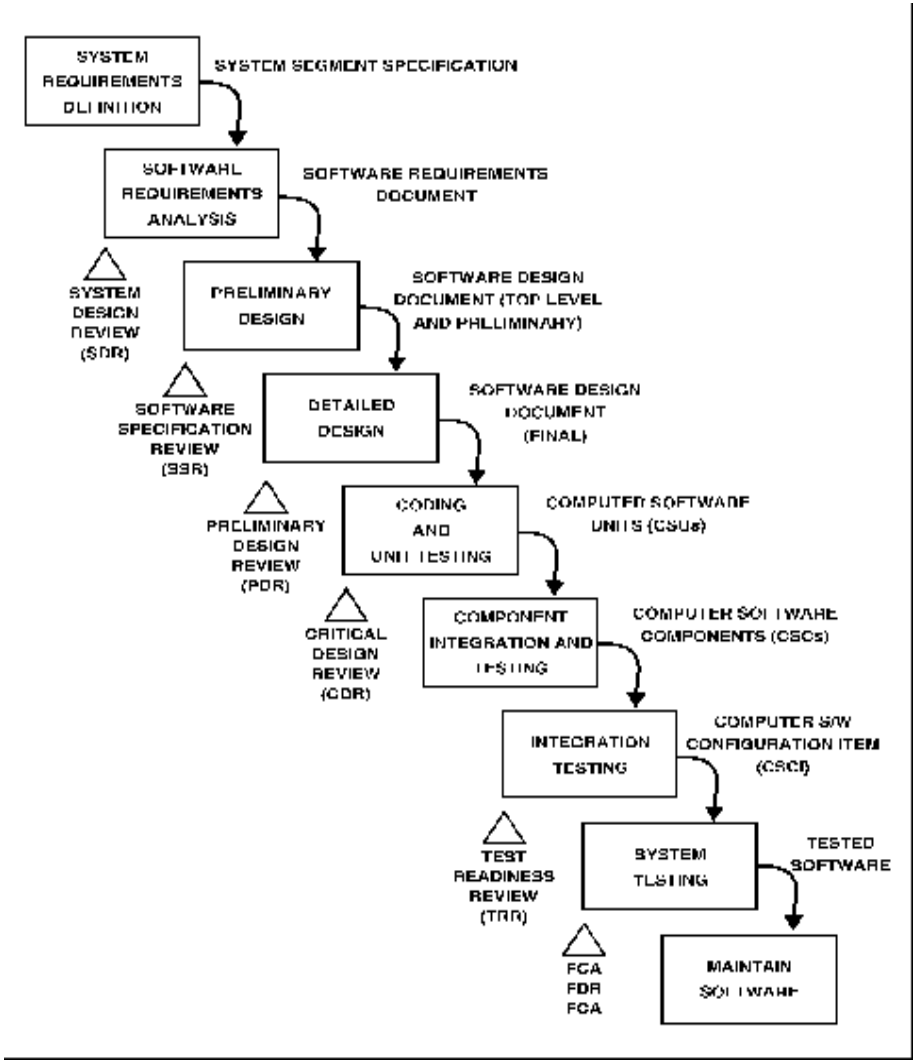
- I moduli sono integrati tra loro
- Testing delle interazioni inter-modulo
- Testing preliminare di sistema
- Risultato di questa fase:
  - il sistema completamente implementato
  - Documento dei test di integrazione

Test del sistema



- Il sistema è testato nella sua globalità e nel suo ambiente d'uso (con gli utenti)
- Accettazione (validazione)
- Risultato di questa fase:
  - un sistema completamente testato, pronto per essere *deployed*

Modello a cascata: aspetti positivi e negativi?

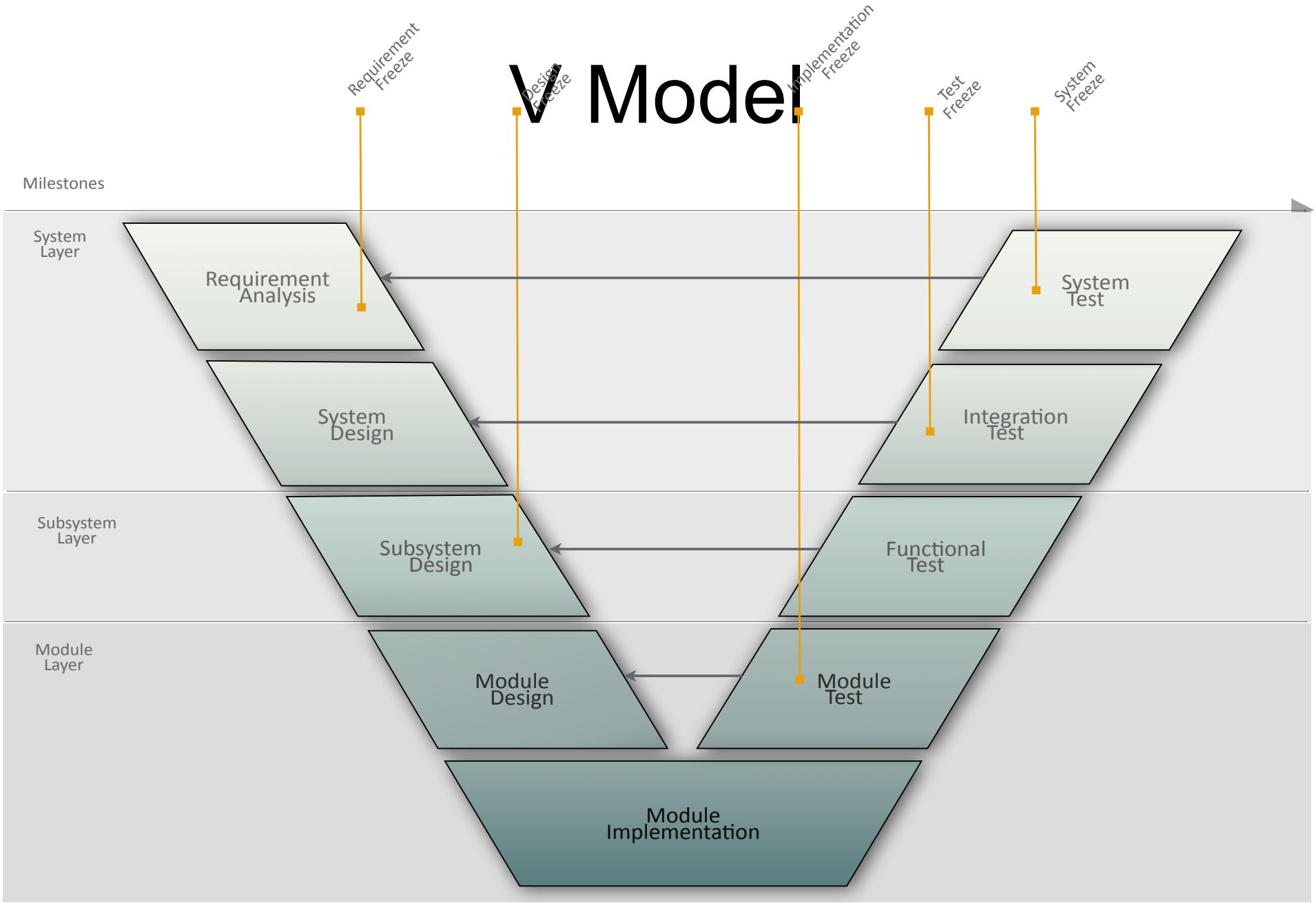


- Pianificato
- Molto dettagliato
- Molto rigido
- Orientato alla documentazione
- Orientato agli standard
- Adatto solo per organizzazioni gerarchizzate (burocrazie)
- Coinvolge il cliente solo alla fine del processo

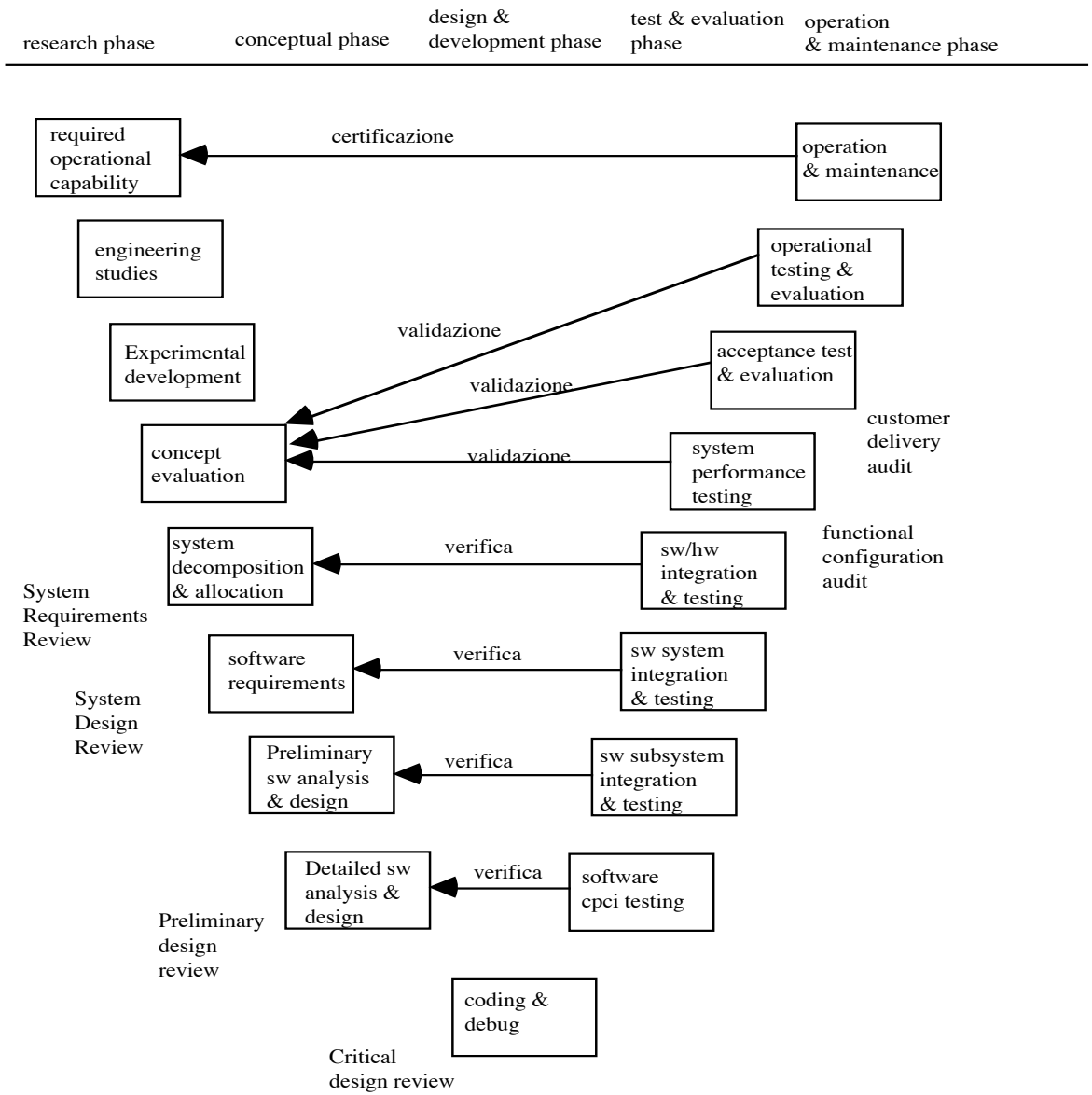
Modello a cascata: aspetti positivi e negativi?

- Si adatta bene a progetti con requisiti stabili e ben definiti
- Problemi:
  - Il cliente deve sapere definire i requisiti
  - Versione funzionante del software solo alla fine
  - Difficili modifiche “in corsa”
  - Fasi fortemente collegate tra loro e bloccanti

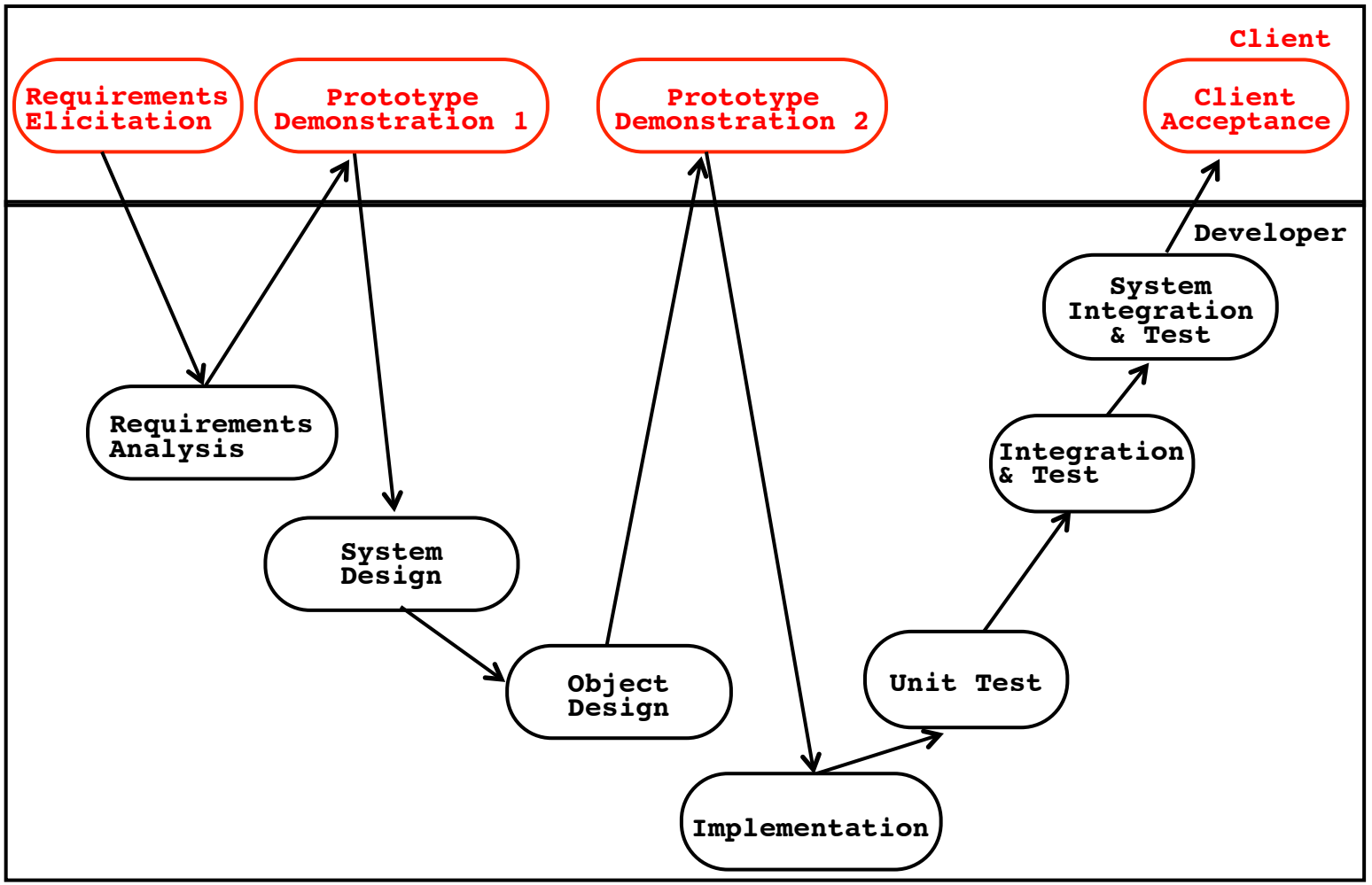
V Model


Requirement Freeze
Design Freeze
Implementation Freeze
Test Freeze
System Freeze
Milestones
System Layer
Subsystem Layer
Module Layer
Requirement Analysis
System Design
Subsystem Design
Module Design
Module Implementation
Module Test
Functional Test
Integration Test
System Test

Modello a cascata, versione a V

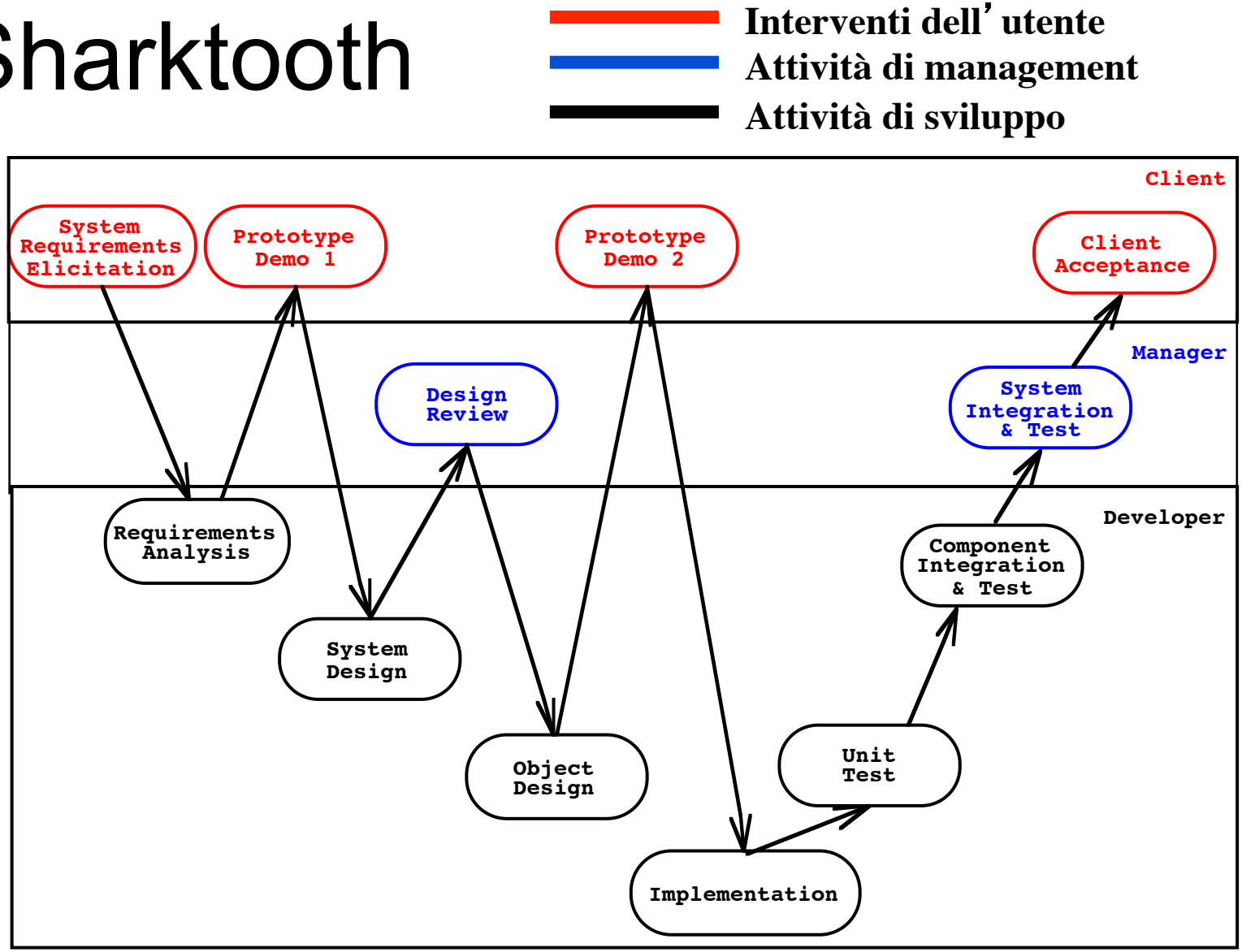


Modello Sawtooth



Modello waterfall con prototipazione (revolutionary and evolutionary)

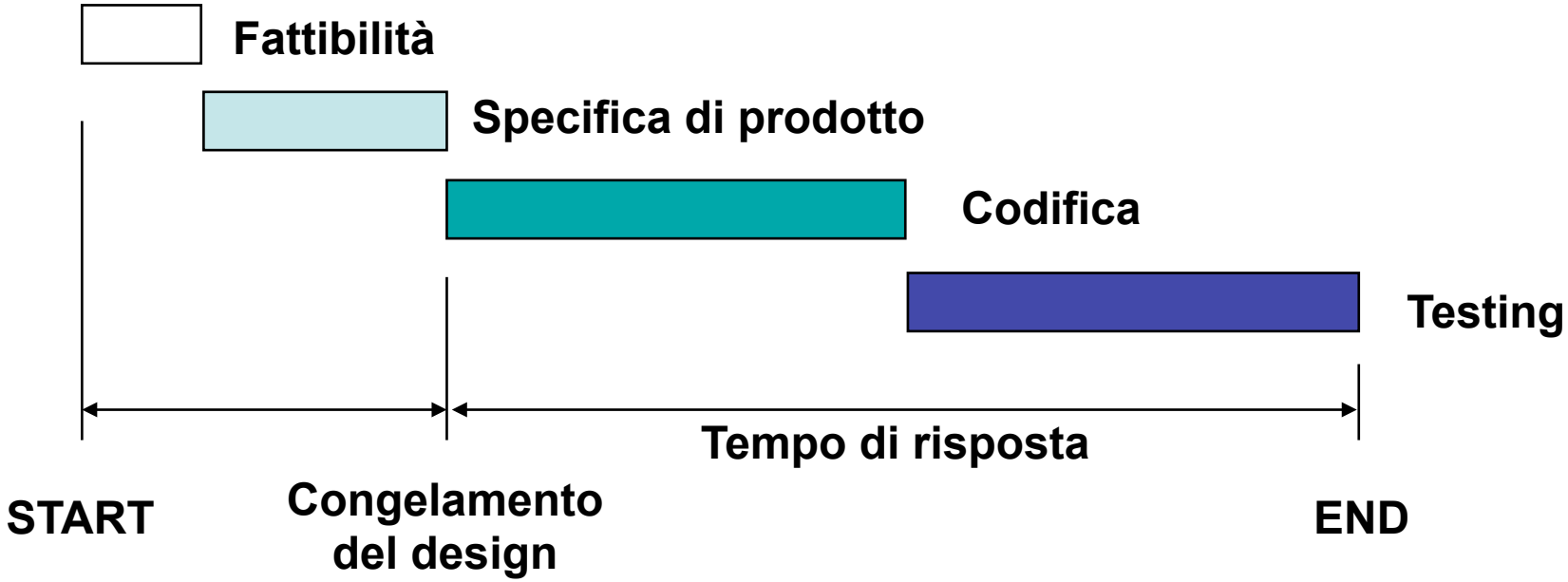
Modello Sharktooth



Waterfall con prototipazione, usabile per outsourcing

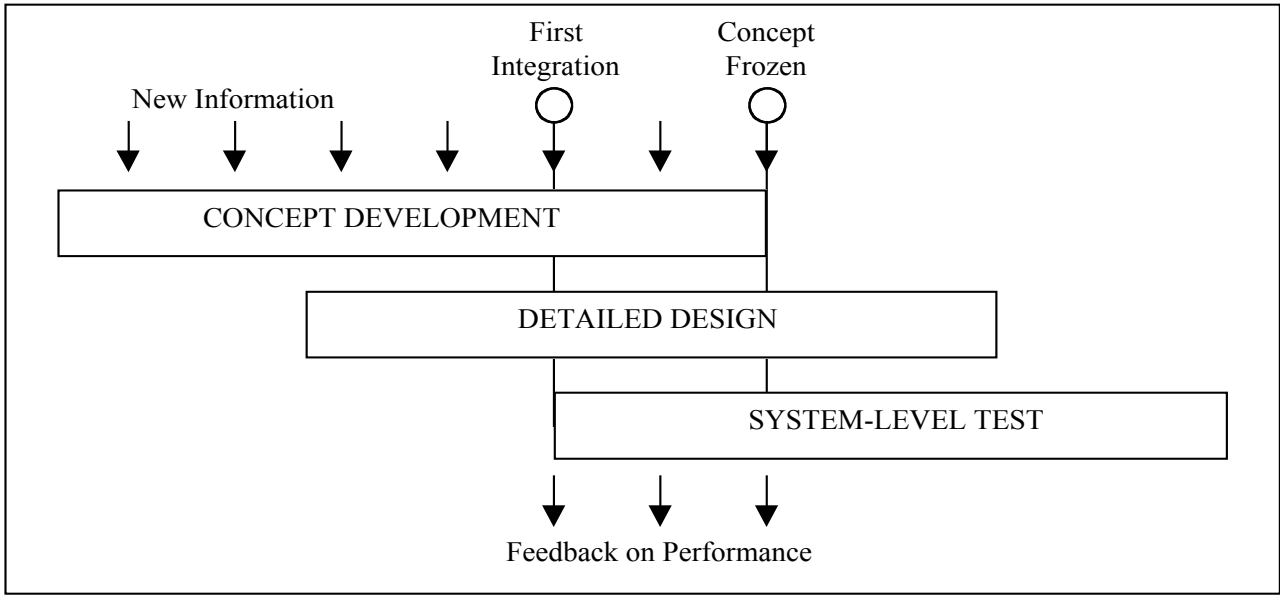
Problema dei processi lineari

Modello di sviluppo rigidamente sequenziale



Problema: il processo non risponde ai cambiamenti di mercato che siano più rapidi della sua esecuzione

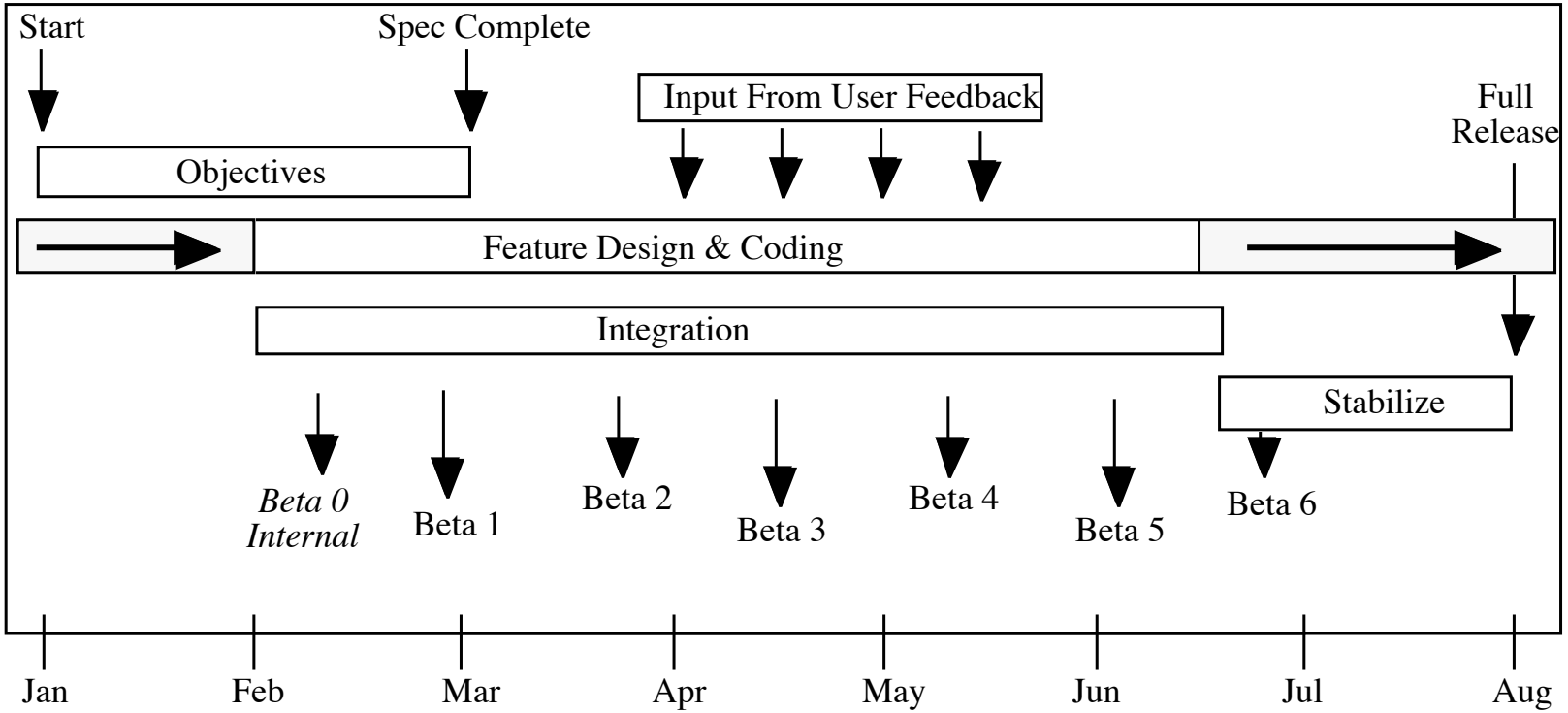
Un modello più flessibile: fasi sovrapposte



- **Caratteristiche**
  - Basato su **apprendimento** e **adattamento** vs *pianificazione* ed *esecuzione*
  - Processo iterativo

Esempio

Progetto Netscape's Navigator 3.0



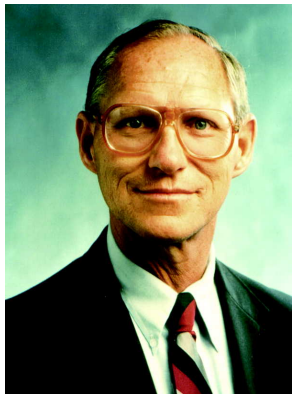
>50% di nuovo codice sviluppato dopo il primo rilascio beta!

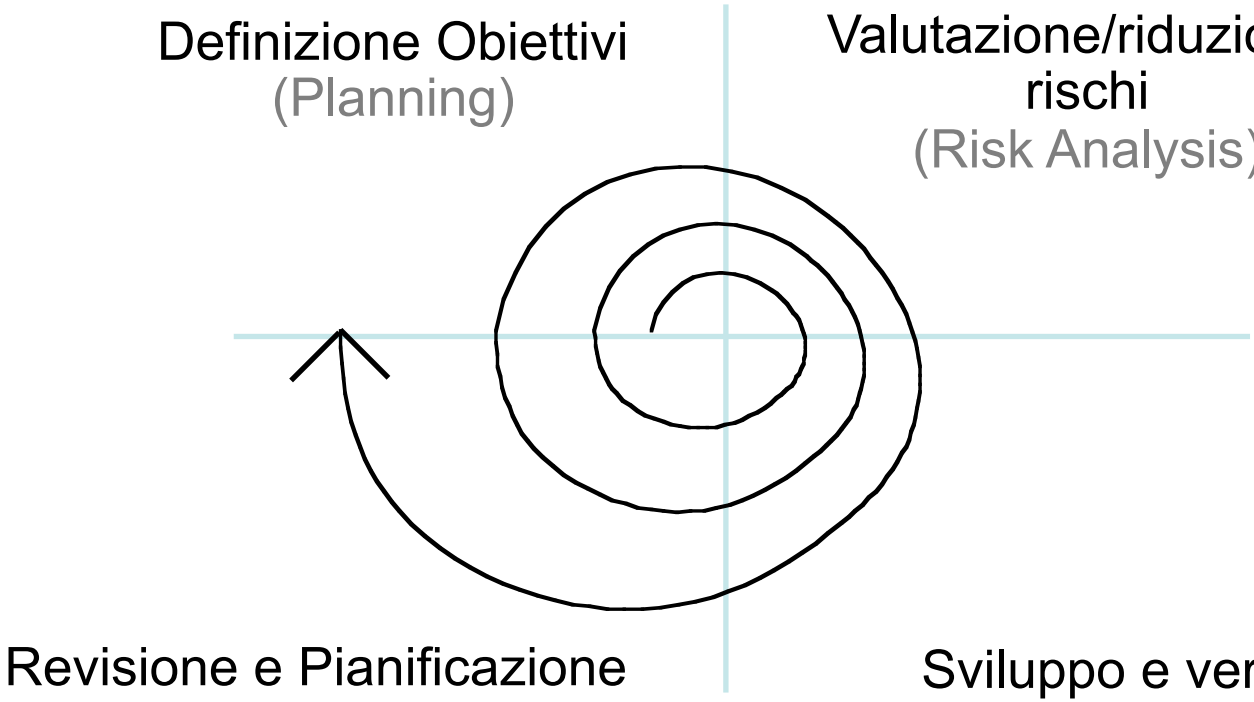
Modello a spirale

- Modello **iterativo** in cui **i rischi vengono valutati continuamente ed esplicitamente**

- Il processo viene modellato da una spirale (e non da una sequenza di attività)
- Ogni spira della spirale è un **round** del processo e prevede diverse attività, divise in quattro fasi:
  - Planning, Risk Analysis, Engineering, Evaluation

Modello a spirale (Boehm)





I settori del modello a spirale

I. Definizione dell'obiettivo
   - Ogni round identifica i propri obiettivi

II. Valutazione e riduzione dei rischi
   - Messa in priorità dei rischi
   - Ogni rischio deve essere affrontato

III. Sviluppo e validazione
   - Il modello di sviluppo può essere generico
   - Ogni round include sviluppo e validazione

IV. Pianificazione
   - Revisione del progetto e pianificazione del suo futuro

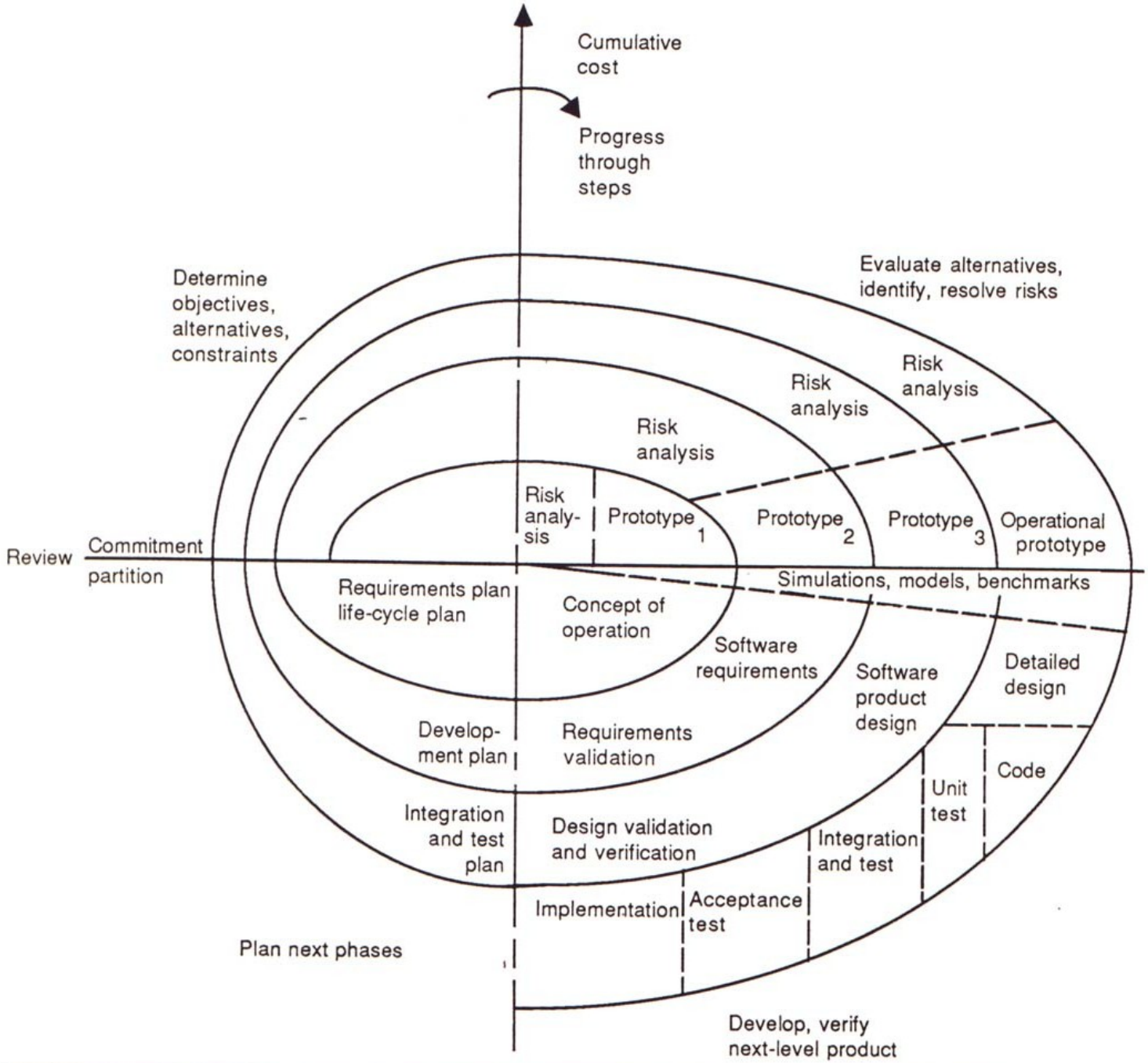
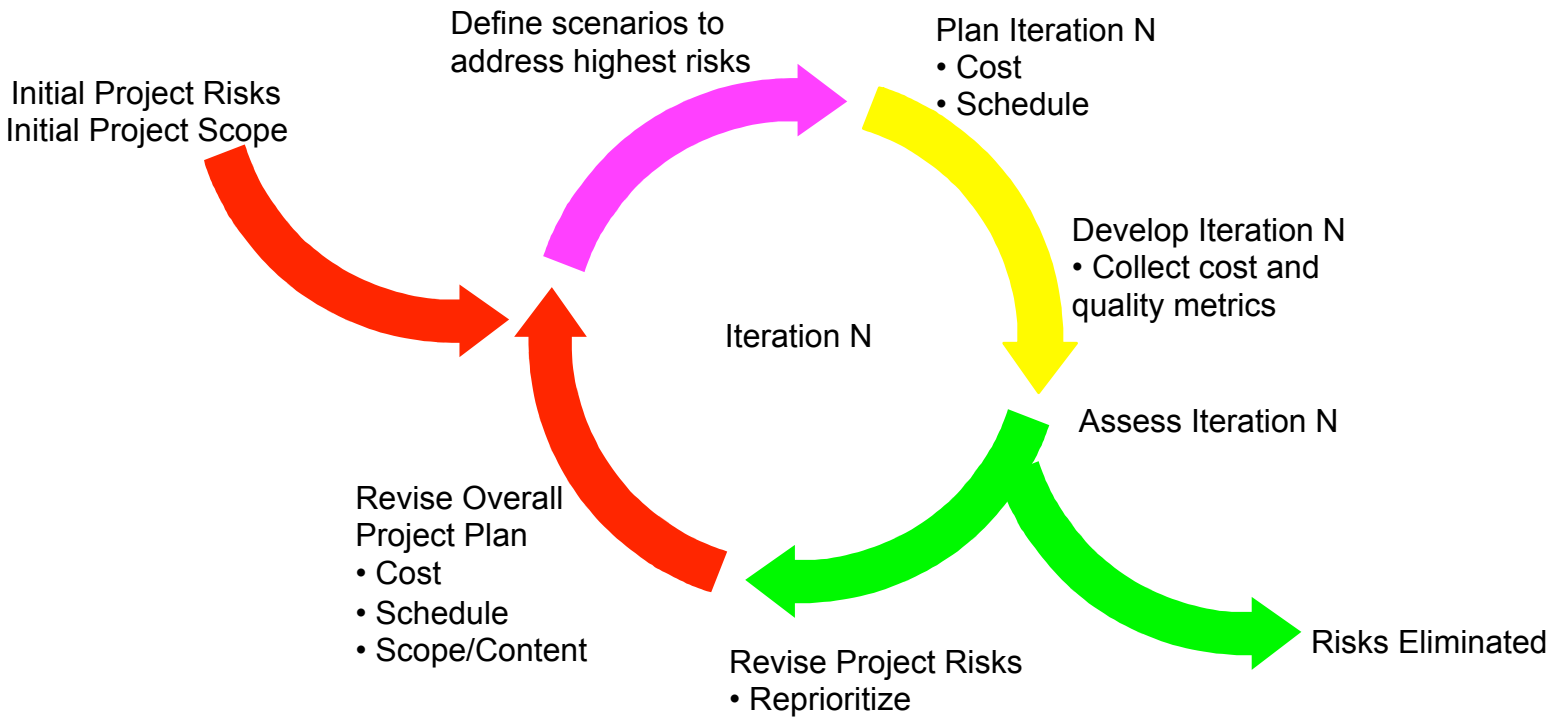


Figure 2. Spiral model of the software process.

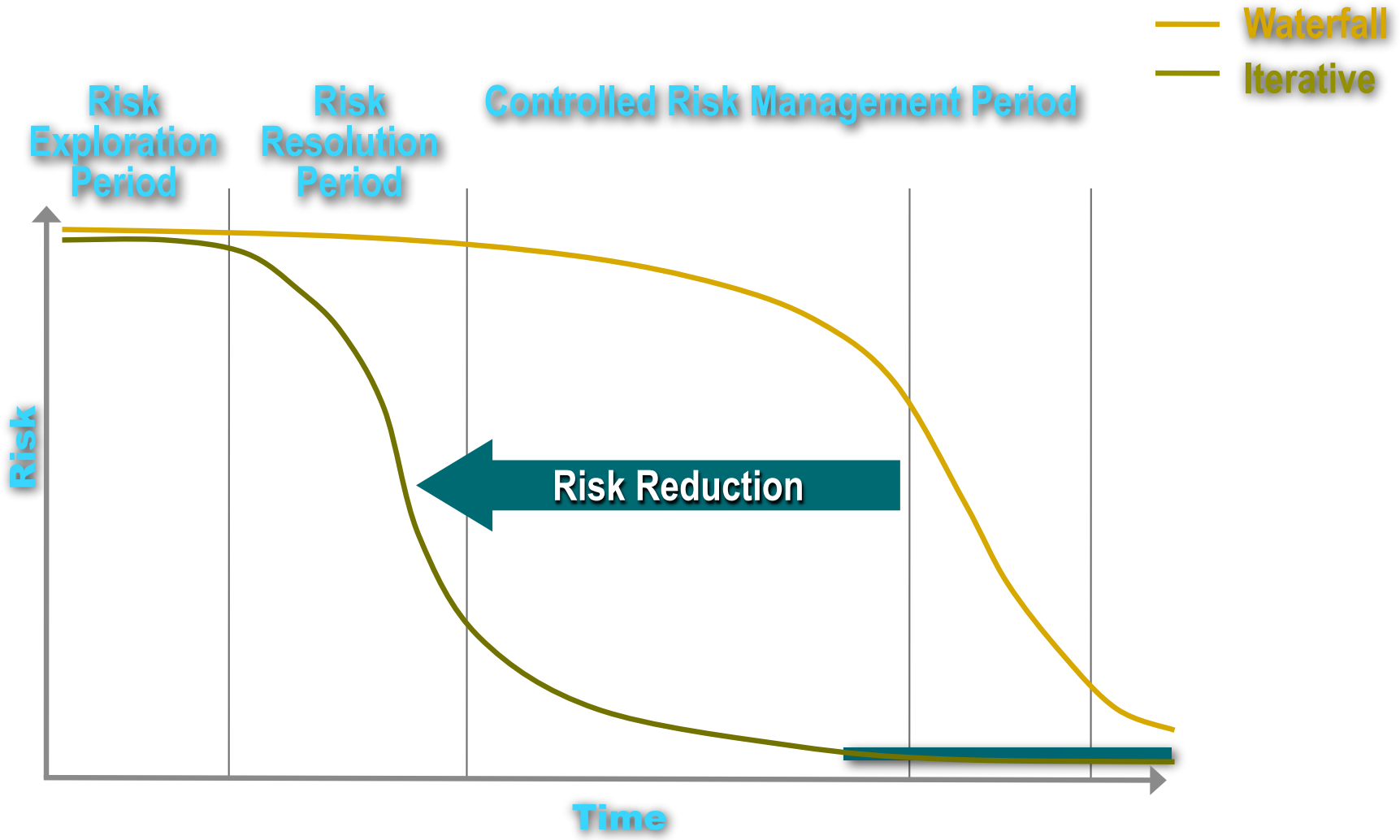
Modello a spirale

- Adatto se requisiti instabili
- Non lineare ma **pianificato**
- Flessibile
- Valuta il rischio
- Può supportare diversi modelli
- Richiede il coinvolgimento del cliente
- Difficile valutare i rischi
- Costoso: ROI (Return On Investment)?

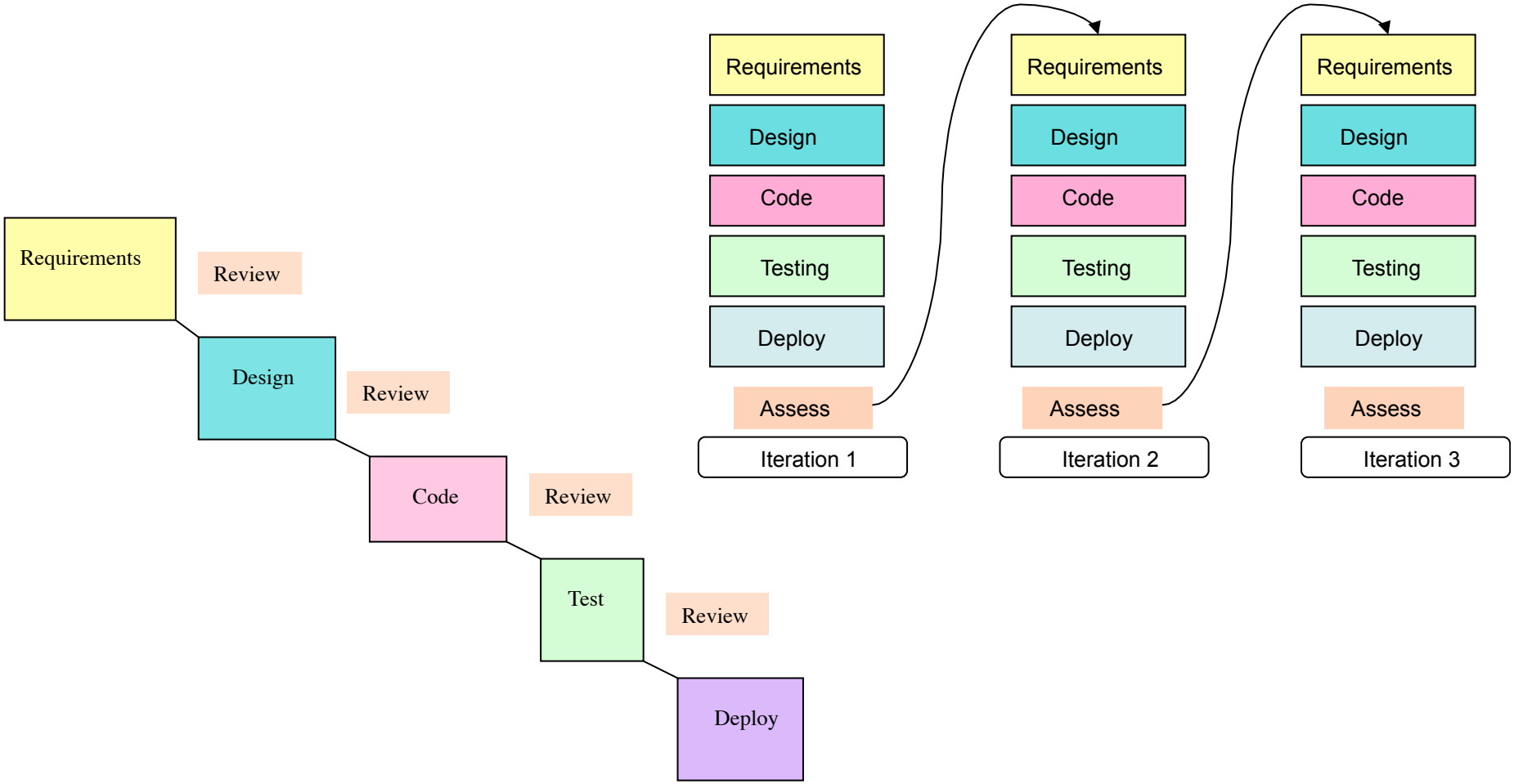
I processi iterativi diminuiscono i rischi



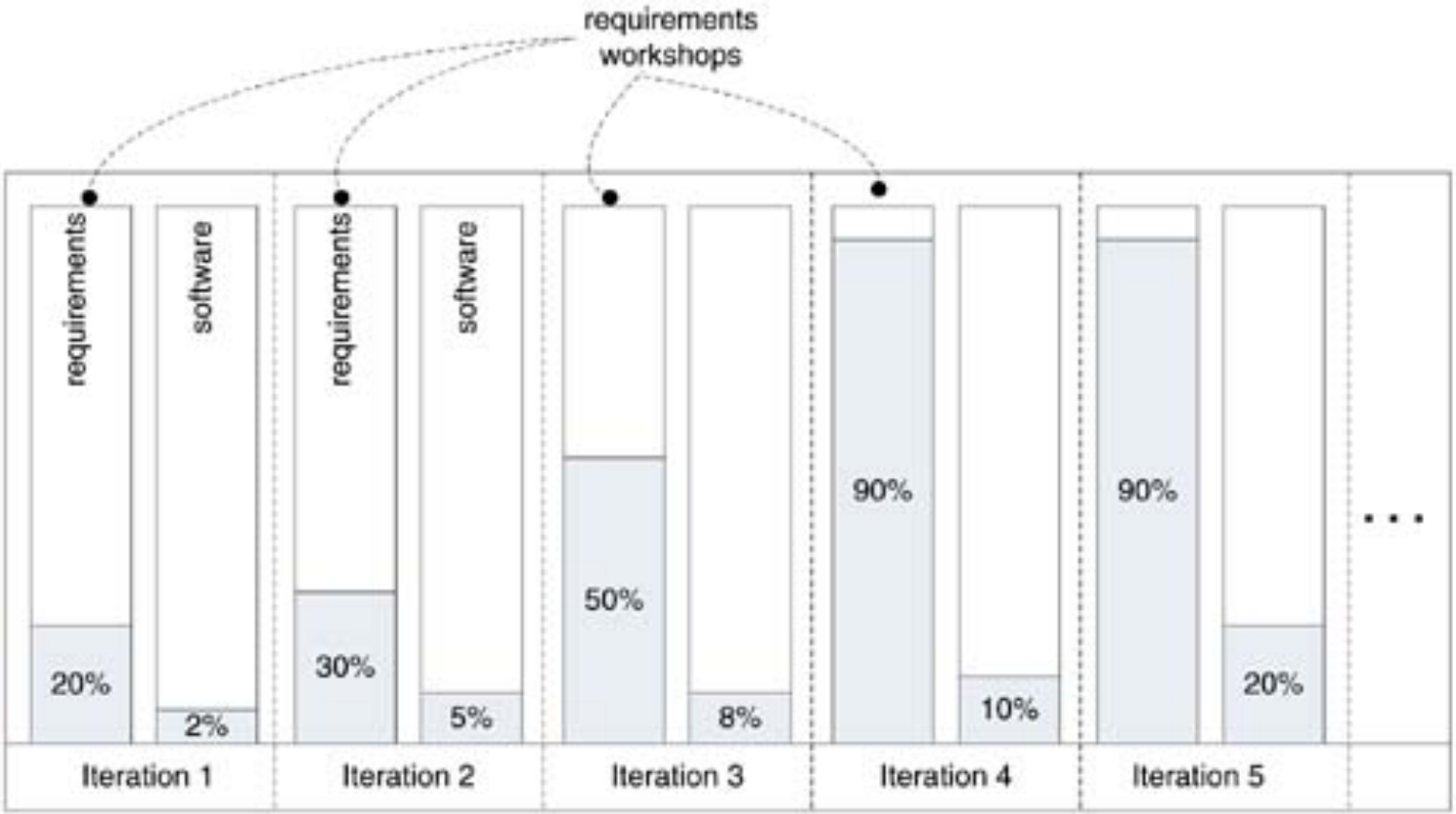
Cascata vs Iterativi



Cascata vs iterativi: freezing artifacts?



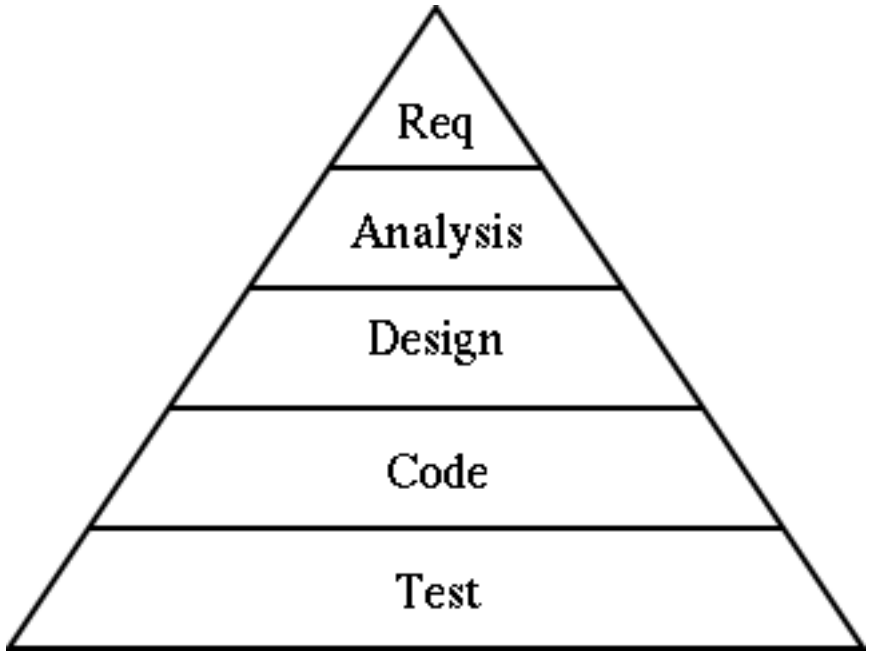
Iterazioni



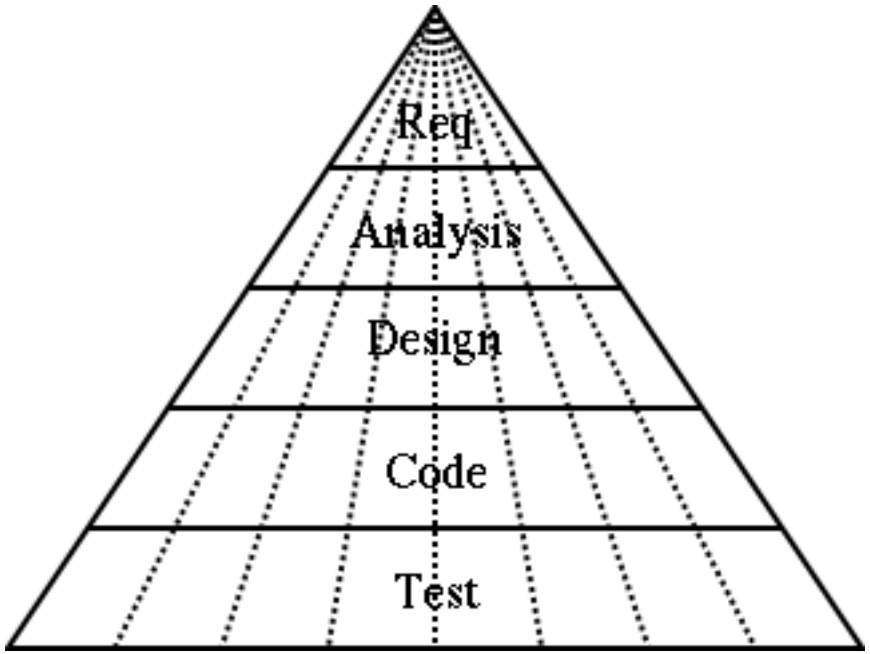
Imagine this will ultimately be a 20-iteration project.

In evolutionary iterative development, the requirements evolve over a set of the early iterations, through a series of requirements workshops (for example). Perhaps after four iterations and workshops, 90% of the requirements are defined and refined. Nevertheless, only 10% of the software is built.

Cascata vs iterativi: sforzo

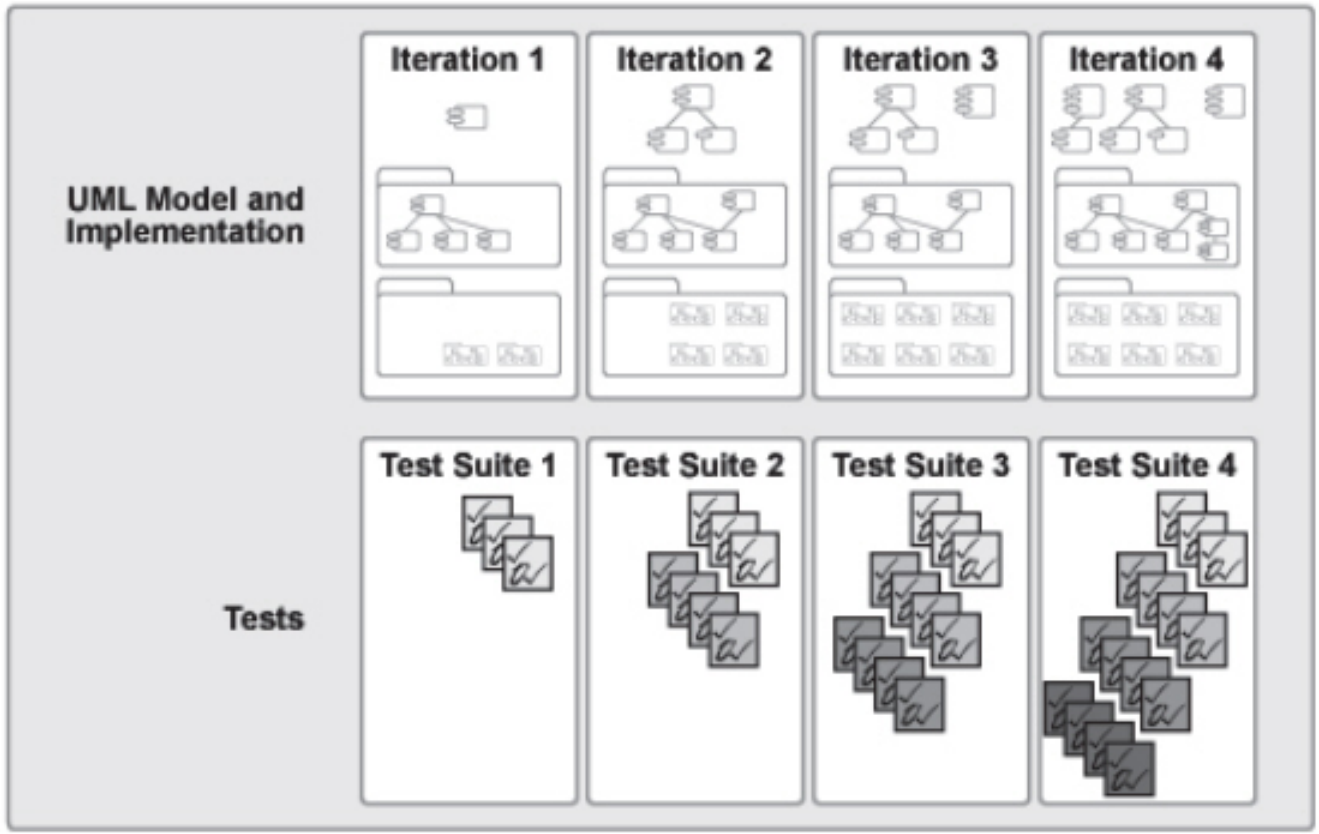


Aumento dello sforzo nelle fasi di un processo a cascata



Segmentazione dello sforzo nelle fasi di un processo iterativo

Iterativi: testing incrementale



Processi iterativi

- RUP
- Open UP
- Varianti RUP e MSF
- Synch and stabilize

RUP

- Modello di processo di tipo iterativo e incrementale, diviso in quattro fasi
  - Inception
  - Elaboration
  - Construction
  - Transition
- Articolato su diverse discipline (workflows)
- Supportato da strumenti proprietari IBM (Rational Rose)

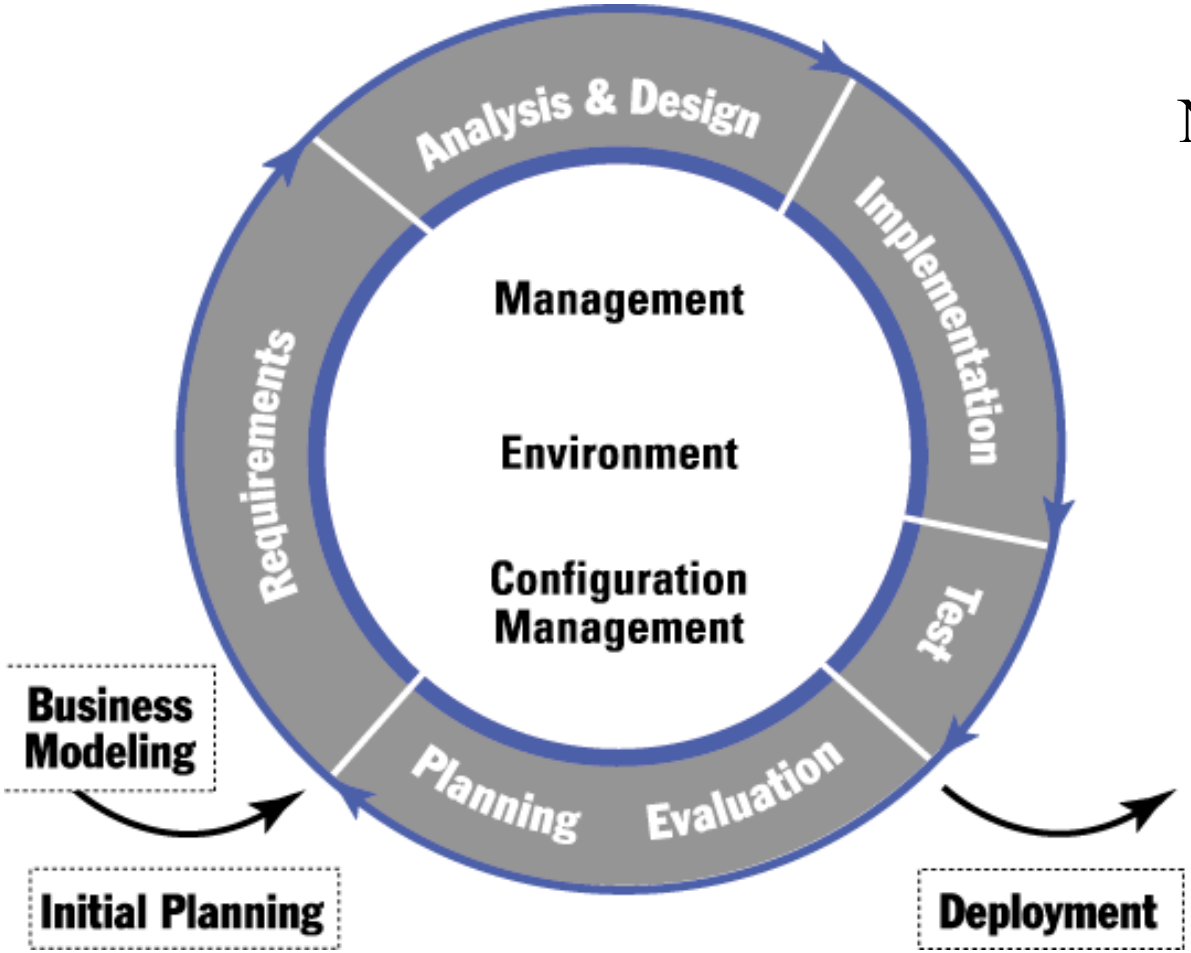
RUP: piccola storia

- Jacobson propose all'inizio degli anni '90 un metodo per progettazione ad oggetti chiamato Objectory
- Nel 1996 Objectory venne scelto da Rational come processo di riferimento col nome USDP Unified Sw Development Process
- In seguito Rational registrò questo processo col nome di RUP™ (TM = trade mark cioè marchio depositato)
- Nel 2001 IBM comprò Rational e continua a supportare RUP™
- I processi che si ispirano a RUP™ di solito si chiamano UP (per es.: Open UP oppure Enterprise UP)

Dunque

Objectory = USDP = RUP™ = UP

Il RUP è un processo iterativo

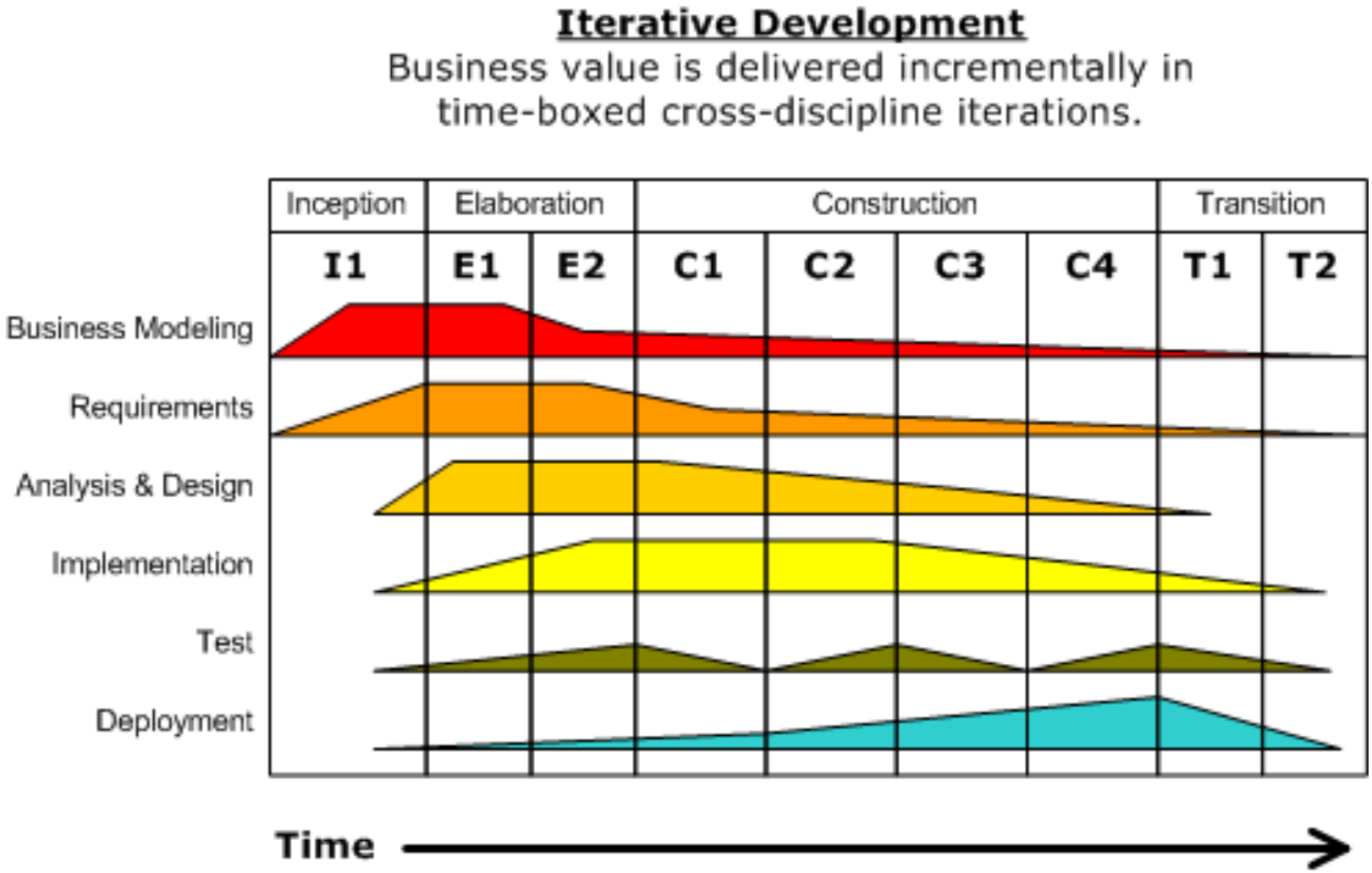


Nel tempo…

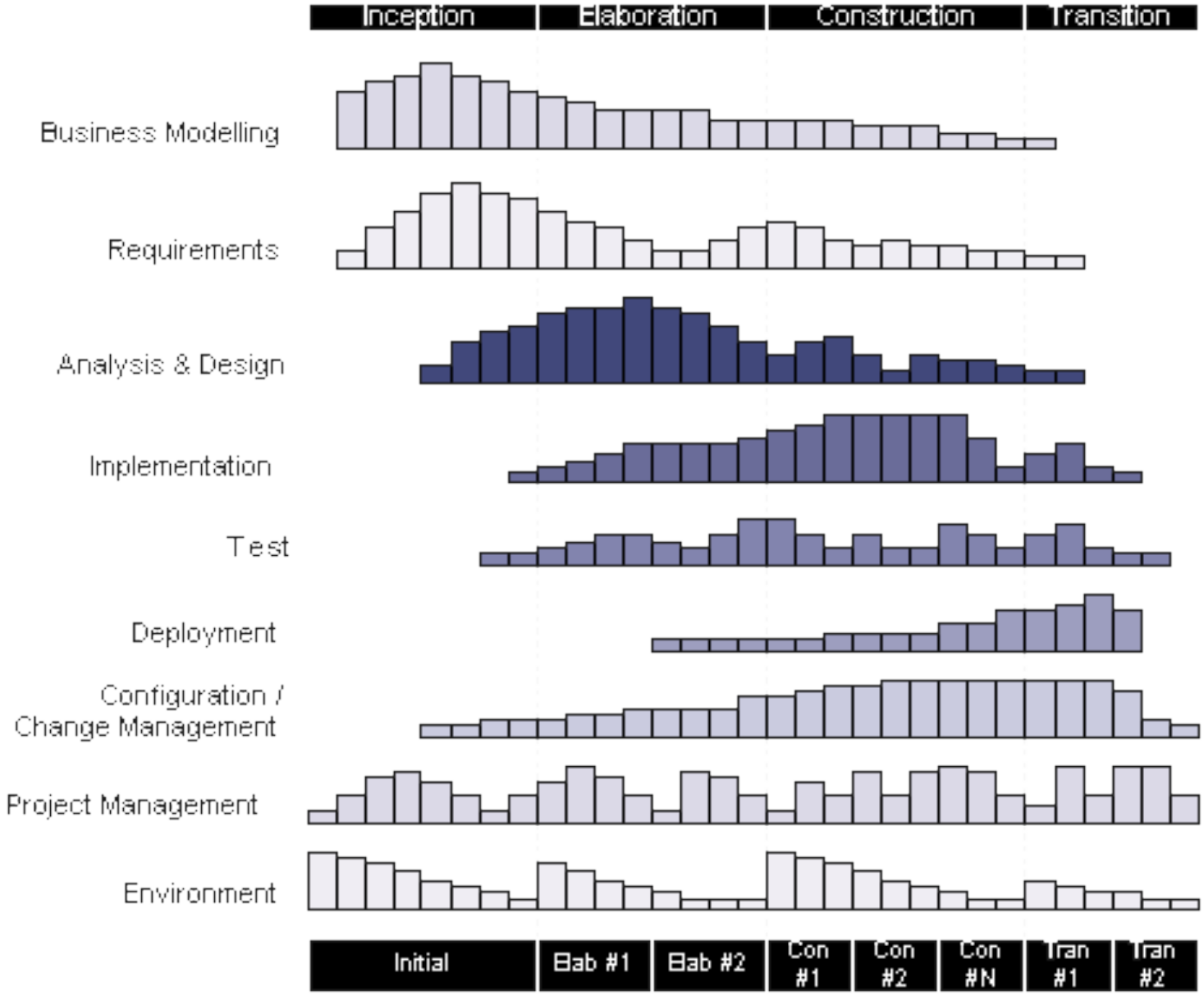
Iterazione: sequenza di *attività* con un *piano* prestabilito e dei *criteri* di valutazione, che termina con un *rilascio* eseguibile

`http://www-306.ibm.com/software/rational/sw-library/`

RUP: visione grafica



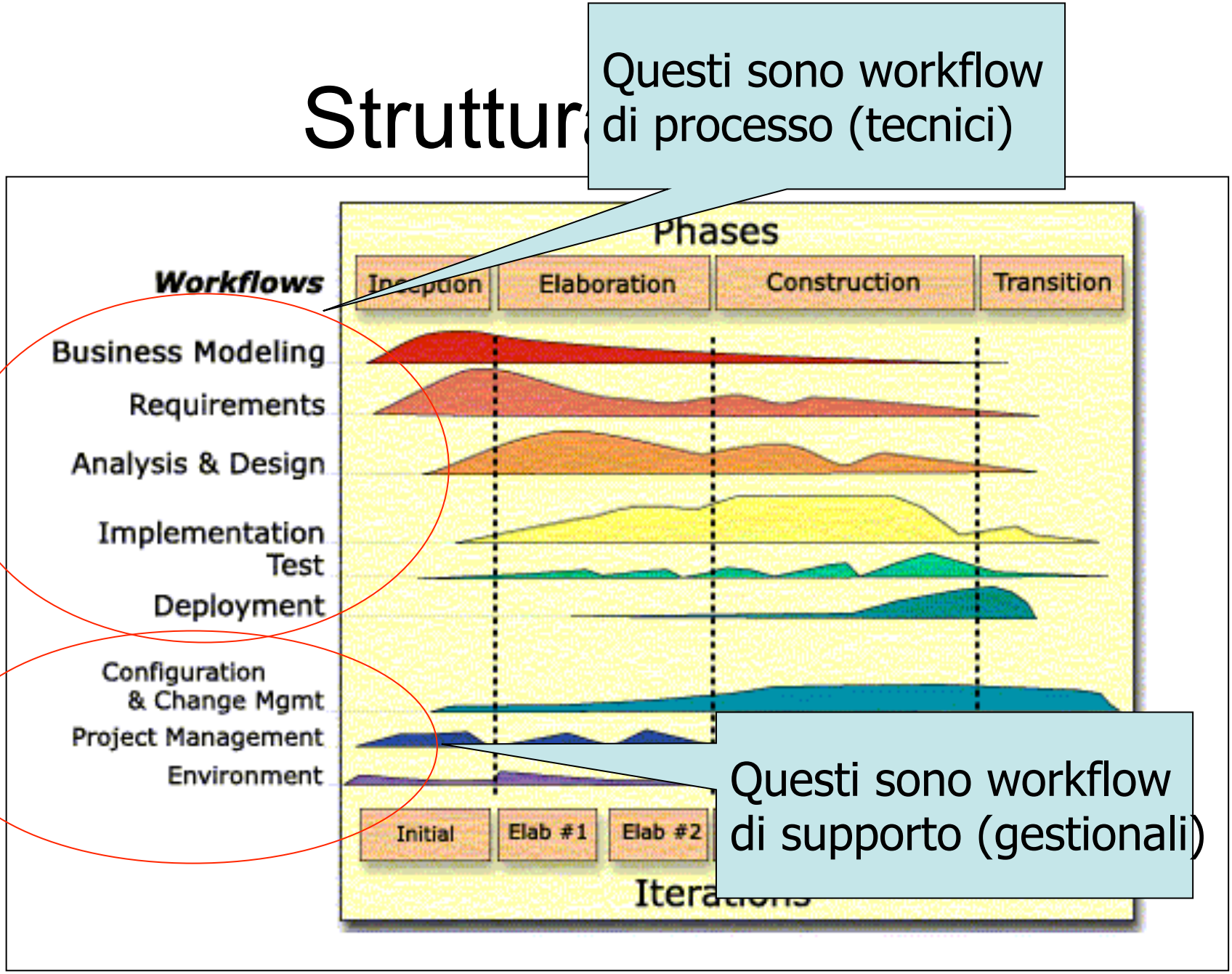
RUP: visione grafica 2



Il RUP ha due prospettive

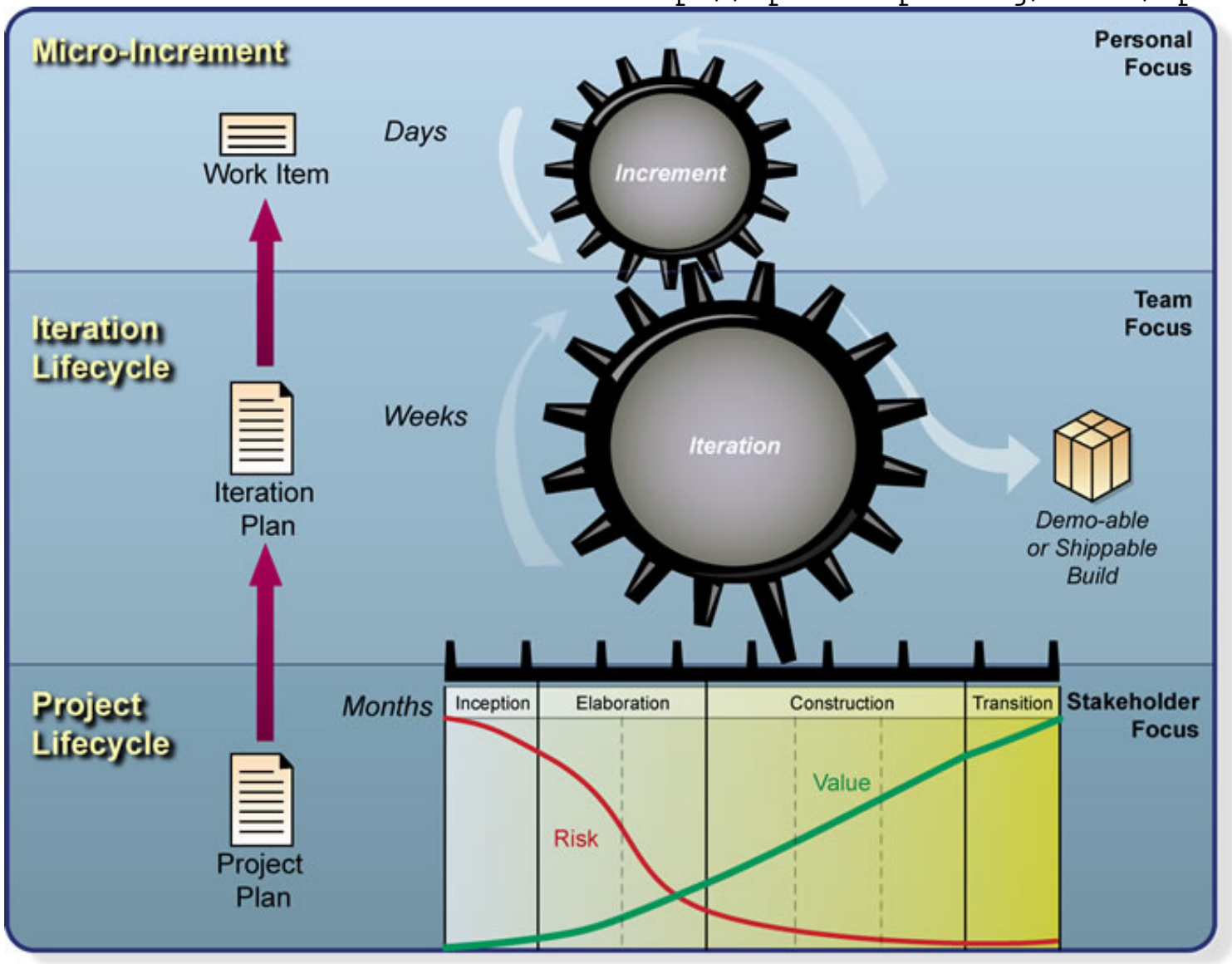
- Il processo RUP integra due diverse prospettive:
  - Una ***prospettiva tecnica***, che tratta gli aspetti qualitativi, ingegneristici e di metodo di progettazione
  - Una ***prospettiva gestionale***, che tratta gli aspetti finanziari, strategici, commerciali e umani
- Le due prospettive sono rispettivamente articolate su sei e tre "**core workflow**"

Struttur

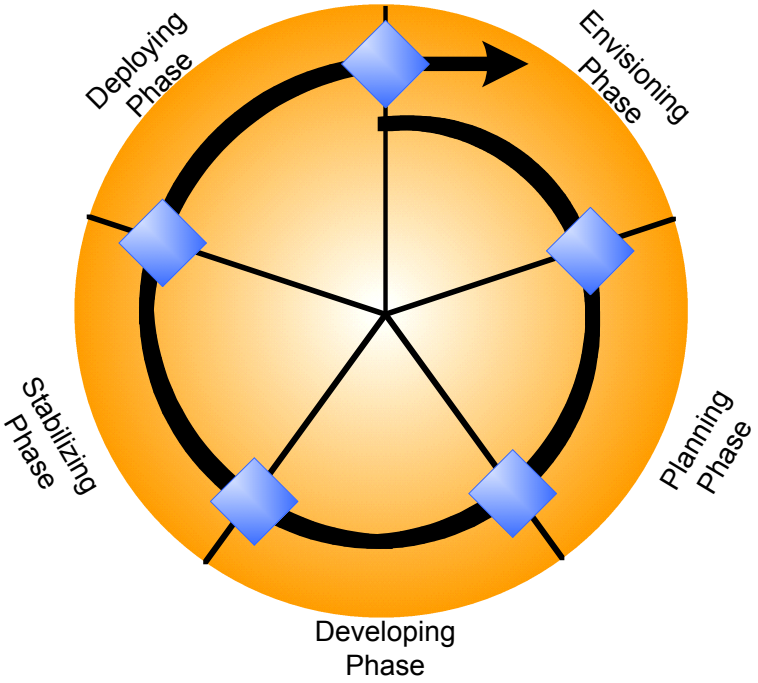

Questi sono workflow di processo (tecnici)
Phases
Workflows
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration & Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Iterations
Questi sono workflow di supporto (gestionali)

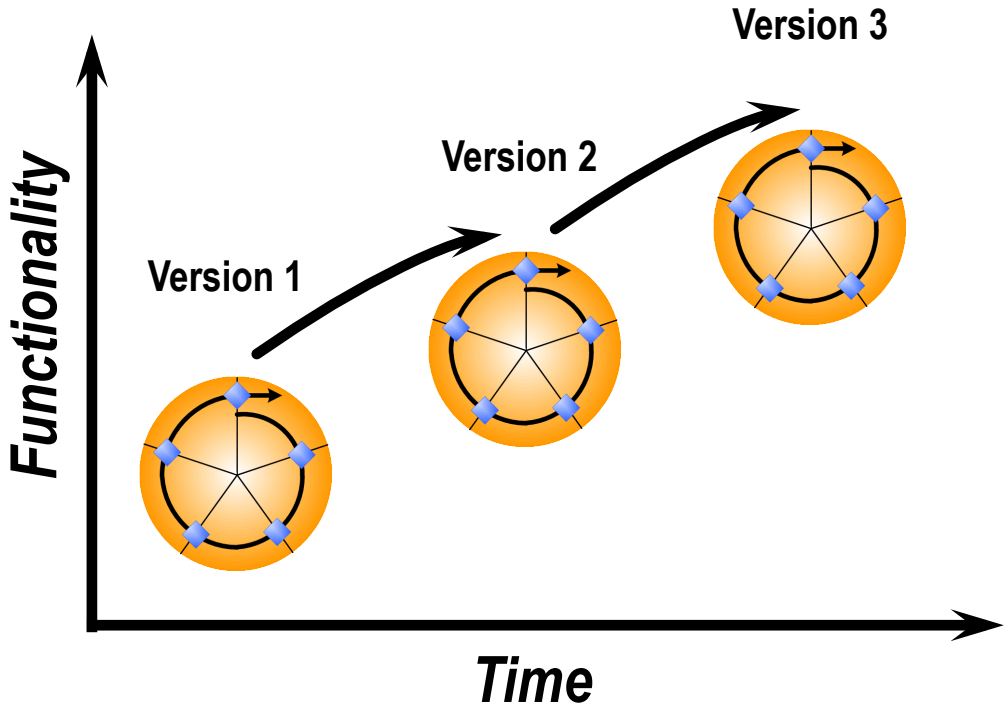
Open UP

http://epf.eclipse.org/wikis/openup/



Anche MSF è un processo iterativo e incrementale

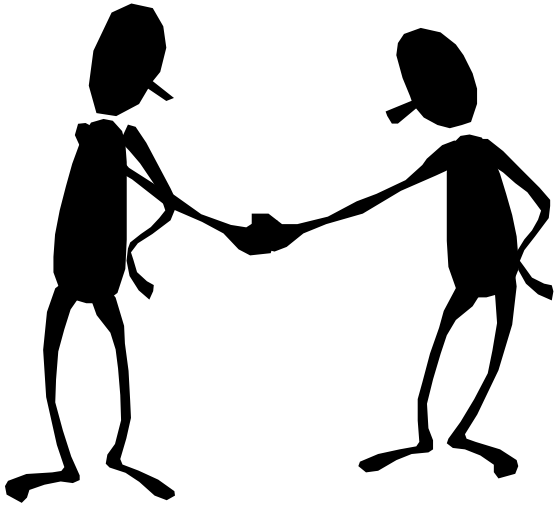




MSF: Microsoft Solutions Framework

Microsoft: Synch-and-Stabilize

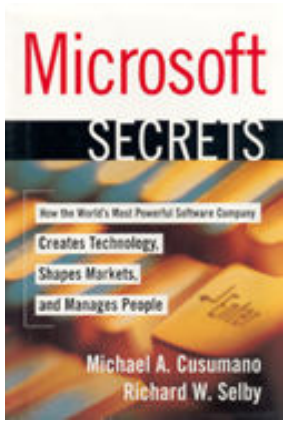
- Sincronizzare di continuo gli sviluppi paralleli
- Stabilizzare e incrementare periodicamente il prodotto (non tutto alla fine)
- Processo noto anche sotto i nomi:
  - milestone process
  - daily build process
  - nightly build process
  - zero-defect process



Il processo interno in Microsoft

- Pianificazione
  - Documento programmatico
  - Specifica
  - Team management
- Sviluppo
  - 3-4 Sottoprogetti
- Stabilizzazione
  - Collaudo interno
  - Collaudo esterno
  - Golden master

Michael Cusumano



Pianificazione

- Documento di “visione” a cura del manager di prodotto
- Definisce gli obiettivi del nuovo prodotto o delle nuove funzioni (“*features*”)
- Definisce le priorità delle funzioni da implementare in base ai bisogni degli utenti
- Pianificare lo sviluppo con “buffer time”
- Documenti tipici:
  - **Specifica** di ciascuna “feature”
  - Pianificazione e definizione dei **team di progetto**
    - 1 program manager
    - 3-8 developers
    - 3-8 testers (1:1 ratio with developers)

Sviluppo

- Lo sviluppo delle ***features*** è suddiviso in 3-4 **sottoprogetti** (da 2-4 mesi ciascuno) usando specifiche funzionali opportunamente partizionate e messe in priorità
- Sottoprogetto: progetto, codifica, debugging
  - Si inizia con le funzioni prioritarie e col codice condiviso
  - L'insieme di "feature" può cambiare del 30% o più durante lo sviluppo

Sviluppo di un sottoprogetto

- Ciascun **sottoprogetto** esegue **in autonomia** il ciclo completo di sviluppo, integrazione di feature, testing e debugging
- I **testers** vengono accoppiati agli **sviluppatori**
- Le **squadre** si **sincronizzano** costruendo il prodotto e correggendo gli errori su **base quotidiana** e **settimanale**
- Il **prodotto** si **stabilizza** alla fine del sottoprogetto

Stabilizzazione

- Testing interno del prodotto completo
- Testing esterno: varie tipologie
  - Siti beta
  - ISVs (Independent Sw Vendors)
  - OEMs (Original Equipment Manifacturers)
  - Utenti finali
- Preparazione della release

Approccio alla qualità in Microsoft

Il principio base seguito da MS per perseguire la qualità dei propri prodotti è: "***Eseguire ogni attività in parallelo, con frequenti sincronizzazioni***"

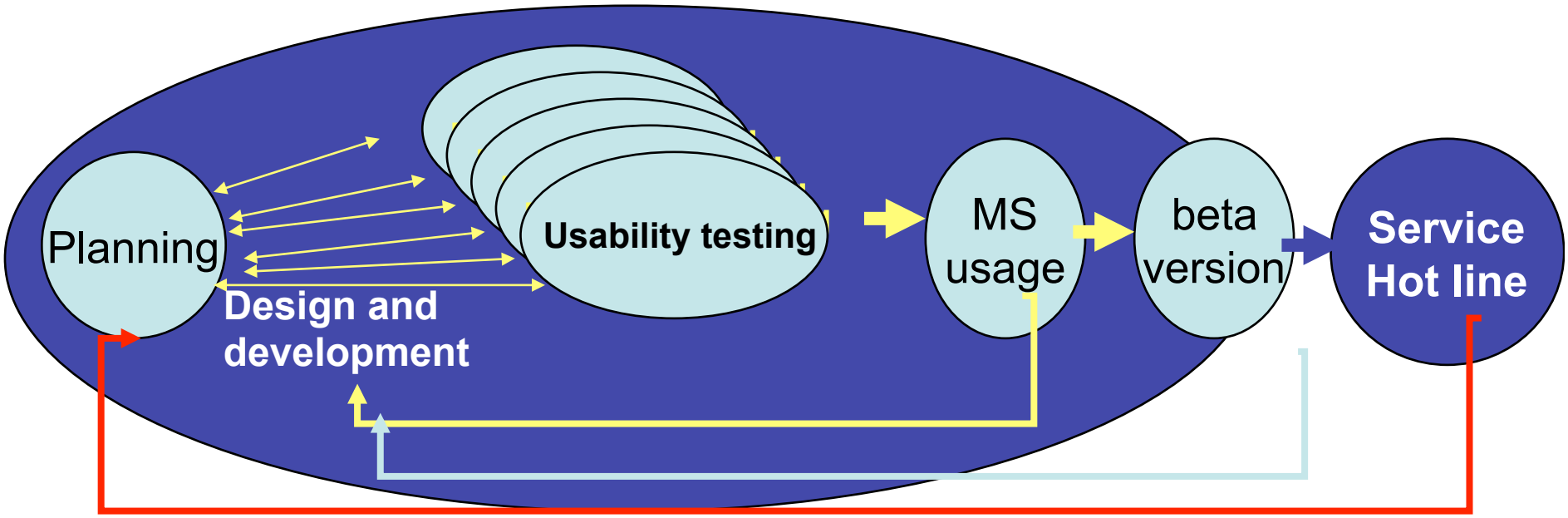
- Dividere i grandi progetti in più cicli di traguardi interni, con *buffer* e *nessuna manutenzione* separata del prodotto
- Utilizzare come guida un *documento programmatico* e un *documento di specifica* delle funzionalità del prodotto
- Fondare la scelta delle funzionalità e la scala delle priorità *sui dati e le attività dell'utente*
- Sviluppare un'architettura modulare e orizzontale, *in cui la struttura del prodotto sia rispecchiata nella struttura del progetto*
- Controllare un progetto impegnando i singoli in compiti di breve portata e "bloccando" le risorse del progetto

Come Microsoft coinvolge gli utenti

Gli utenti sono coinvolti durante lo sviluppo

- ‘pianificazione basata su attività’
- test di usabilità
- uso interno della nuova applicazione da parte di personale Microsoft
- linee di supporto alla clientela

Come Microsoft coinvolge gli utenti



I principi chiave di Synch&Stab

(per definire i prodotti e modellare i processi)

1. Dividere i grandi progetti in più iterazioni con buffer time appropriato (20-50%)
2. Usare un “vision statement” e classificare le specifiche delle funzioni per guidare il progetto
3. Selezione delle caratteristiche principali e ordinamento di priorità basato su dati d’uso
4. Architettura modulare
5. Gestione basata su piccoli compiti individuali e risorse di progetto prestabilite

Conclusioni

- Processi waterfall: pianificati, rigidi
- Processi iterativi: pianificati, flessibili
- Esistono molte varianti
- Ogni organizzazione si definisce il modello che preferisce

Autotest

- Cos’è un processo software?
- Quali sono le fasi tipiche del processo di sviluppo?
- Quali sono le principali differenze tra processi lineari e processi iterativi?
- Quali sono i rischi principali che incontra chi sviluppa software?

Letture raccomandate

- L. Osterwail, Software processes are software too, ICSE 1987
- M.Cusumano, The business of software

Riferimenti

- Capitolo 9 del SWEBOK: “Software engineering process”
- Boehm, A spiral model of software development and enhancement, *IEEE Computer* 21:5(61-72), May 1988
- Cusumano, How Microsoft builds software, *CACM* 1997

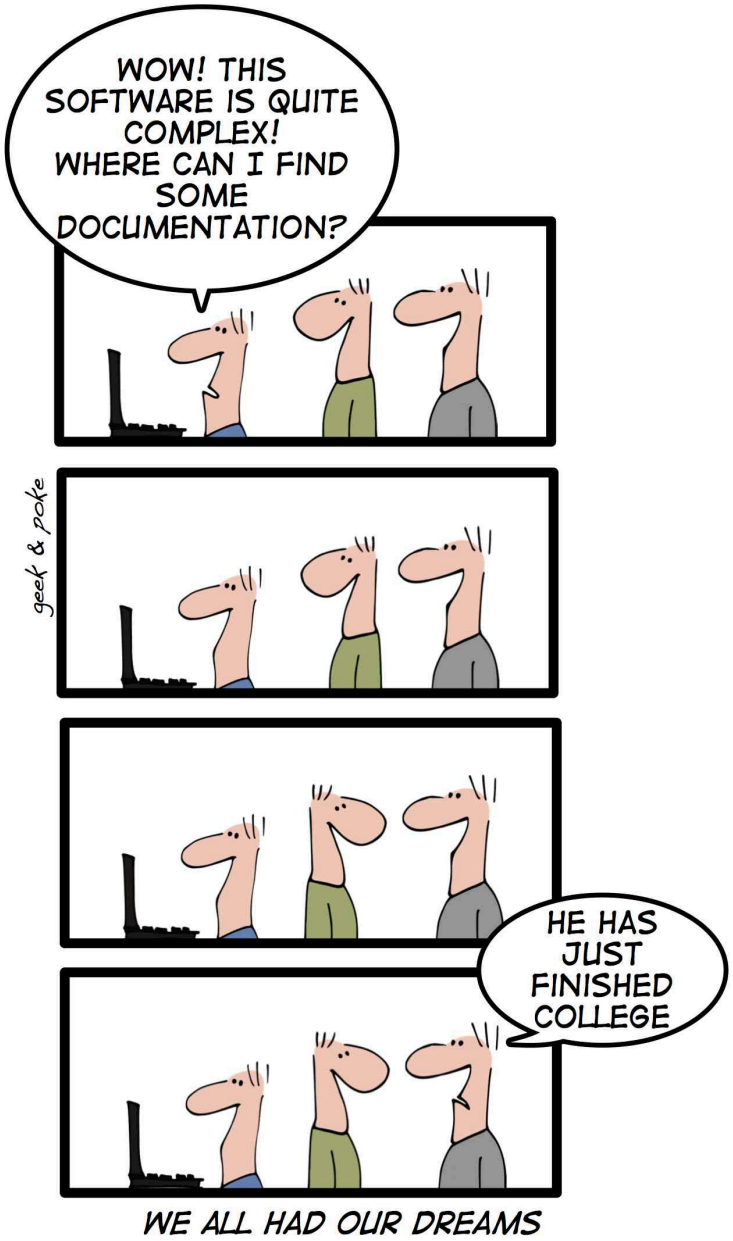
Siti

- Personal Sw Process e Team Sw Process (SEI)
  `www.sei.cmu.edu/tsp/introducing.html`
- Microsoft Solutions Framework
  `www.microsoft.com/technet/itsolutions/msf/default.mspx`
- Adaptable Process Model
  `www.rspa.com/apm/`
- Cucumber `cukes.info/`

Pubblicazioni di ricerca

- International Conference on Software and System Process
- Journal of Software Maintenance and Evolution: research and practice

Domande?


WOW! THIS SOFTWARE IS QUITE COMPLEX! WHERE CAN I FIND SOME DOCUMENTATION?
geek & poke
HE HAS JUST FINISHED COLLEGE
WE ALL HAD OUR DREAMS