



Il ciclo di vita del SW



Ingegneria del Software

V. Ambriola, G.A. Cignoni,
C. Montanero, L. Semini

Aggiornamenti : T. Vardanega (UniPD)

Dipartimento di Informatica, Università di Pisa

1/34




Il ciclo di vita del SW

Il concetto di ciclo di vita – 2

- ❑ Lo stazionamento in uno stato di ciclo di vita o in una transizione tra essi viene detta fase
- ❑ Aderire a un modello di ciclo di vita determina vincoli sulla pianificazione e la gestione del progetto
 - La scelta del modello precede e determina la selezione dei metodi e degli strumenti di lavoro
- ❑ Associare un sistema di qualità al modello adottato (e ai processi di sua attuazione) serve a misurare e assicurare conformità e maturità
 - Conformità agli obiettivi del modello (nel rispetto dei vincoli)
 - Maturità (qualità misurata) dei processi che lo attuano

Dipartimento di Informatica, Università di Pisa

3/34




Il ciclo di vita del SW

Il concetto di ciclo di vita – 1

- ❑ **Concezione** → sviluppo → utilizzo → ritiro
 - Gli stati (principali) assunti da un prodotto SW
- ❑ La transizione tra stati avviene tramite l'esecuzione di attività di processi [di ciclo di vita]
 - Processi istanziati da modelli generali (p.es. ISO/IEC 12207)
 - Le attività prevedono ruoli e responsabilità ben determinati
- ❑ Per organizzare le attività di processo implicate
 - Identifichiamo le dipendenze tra ingressi e uscite
 - Fissiamo l'ordinamento nel tempo e i criteri di attivazione (pre-condizioni) e completamento (post-condizioni)

Dipartimento di Informatica, Università di Pisa



2/34




Il ciclo di vita del SW

Modelli e fasi

Questi sono modelli ...



Queste sono fasi ...



Dipartimento di Informatica, Università di Pisa

4/34




Il ciclo di vita del SW

Evoluzione dei modelli [di sviluppo]

- ❑ Dal “*Code-’n-Fix*”, un non-modello
 - Insieme di attività senza organizzazione preordinata
 - Fonte di progetti caotici difficilmente gestibili
- ❑ Ai modelli organizzati
 - Cascata rigide fasi sequenziali
 - Incrementale realizzazione in più passi
 - Evolutivo ripetute iterazioni interne
 - A componenti orientato al riuso
 - Agile altamente dinamico, fatto di brevi cicli iterativi e incrementali

Dipartimento di Informatica, Università di Pisa5/34



Il ciclo di vita del SW

Modello sequenziale (a cascata) – 2

- ❑ Ogni stato di vita (fase) è caratterizzato da pre-condizioni di ingresso e post-condizioni di uscita
 - Il loro soddisfacimento è dimostrato prima tramite prodotti documentali e poi tramite esecuzione del SW
- ❑ Le fasi sono distinte e non sovrapposte nel tempo
- ❑ Modello adatto allo sviluppo di sistemi complessi sul piano organizzativo
 - Le iterazioni costano troppo per essere un buon mezzo di mitigazione dei rischi tramite approssimazioni successive

Dipartimento di Informatica, Università di Pisa7/34




Il ciclo di vita del SW

Modello sequenziale (a cascata) – 1

- ❑ Definito nel 1970 da Winston W. Royce
 - Nella pubblicazione: “*Managing the development of large software systems: concepts and techniques*”
 - Centrato sull’idea di processi ripetibili
- ❑ Successione di fasi rigidamente sequenziali
 - Non ammette ritorno a fasi precedenti
 - Eventi eccezionali fanno ripartire dall’inizio
- ❑ Prodotti
 - Principalmente documenti, fino poi a includere il SW
 - L’emissione e l’approvazione di documenti sono condizione necessaria per l’avvio della fase successiva → modello “*document driven*”

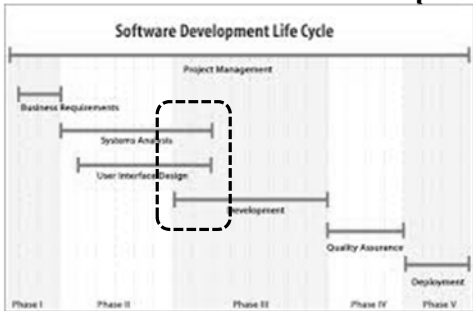
Dipartimento di Informatica, Università di Pisa6/34



Il ciclo di vita del SW


Esempio

- ❑ Questo non è un vero modello sequenziale



The diagram illustrates the Software Development Life Cycle (SDLC) across five phases. Phase I includes Business Requirements. Phase II includes System Analysis and User Interface Design. Phase III includes Development. Phase IV includes Quality Assurance. Phase V includes Deployment. A dashed box highlights the first three phases, indicating that the model is not strictly sequential as it shows overlapping activities.

Dipartimento di Informatica, Università di Pisa8/34




Il ciclo di vita del SW

Modello sequenziale (a cascata) – 3

- ❑ Ogni fase viene definita in termini di
 - Attività previste e prodotti attesi in ingresso e in uscita
 - Contenuti e struttura dei documenti
 - Responsabilità e ruoli coinvolti
 - Scadenze di consegna dei prodotti
- ❑ Le fasi sono durate temporali con dipendenze causali tra loro
 - Entrare, uscire, stazionare in una fase comporta lo svolgimento di azioni specifiche
 - Realizzate come attività erogate da corrispondenti processi

Dipartimento di Informatica, Università di Pisa

9/34




Il ciclo di vita del SW

Correttivi del modello sequenziale

- ❑ Difetto principale: eccessiva rigidità
 - Stretta sequenzialità tra fasi (nessun parallelismo e nessun ritorno)
 - Non ammette modifiche nei requisiti in corso d'opera
 - Richiede molta manutenzione (per garantire economicità)
 - Esprime una visione burocratica e poco realistica del progetto
- ❑ Correttivo 1: con prototipazione
 - Prototipi di tipo "usa e getta"
 - Per capire meglio i requisiti o le soluzioni
 - Strettamente all'interno di singole fasi
- ❑ Correttivo 2: cascata con ritorni
 - Ogni ciclo di ritorno raggruppa sotto-sequenze di fasi

Dipartimento di Informatica, Università di Pisa

11/34



Il ciclo di vita del SW

Schema secondo ISO 12207:1995

Alle attività qui elencate se ne aggiungono molte altre istanziate da altri processi (documentazione, verifica, configurazione, gestione di progetto, ...)

Analisi
 5.3.2 AR sistema
 5.3.4 AR software
 Insieme dei processi primari
 Processo "Sviluppo"
 Attività del processo "Sviluppo"

Progettazione
 5.3.3 DA sistema
 5.3.5 DA software
 5.3.6 DD software

Realizzazione
 5.3.7 Codifica software
 5.3.8 Integrazione software
 5.3.9 Collaudo software
 5.3.10 Integrazione sistema
 5.3.11 Collaudo sistema

Manutenzione
 5.5 Manutenzione
 La manutenzione comporta ritorni sulle fasi precedenti

Legenda
 AR: analisi dei requisiti
 DA: progetto (design) architetturale
 DD: progetto (design) di dettaglio

Dipartimento di Informatica, Università di Pisa

10/34




Il ciclo di vita del SW

Ritorni: iterazione o incremento?

- ❑ Non sempre gli *stakeholder* hanno ben chiaro dall'inizio ogni aspetto del sistema richiesto
 - In tal caso bisogna decidere cosa sia meglio fare per aiutarli
 - Per problemi particolarmente complessi conviene prevedere iterazioni
 - Ma le iterazioni possono essere distruttive e eliminare e rimpiazzare lavoro precedente
- ❑ Spesso non conviene rimandare alle fasi finali l'integrazione di tutte le parti del sistema (*big-bang integration*)
 - In tal caso è meglio l'integrazione successiva di piccole parti
 - Questo è un procedimento incrementale
- ❑ Iterazione e incremento coincidono quando la sostituzione raffina ma non ha impatto sul resto

Dipartimento di Informatica, Università di Pisa

12/34




Il ciclo di vita del SW

Vantaggi dei modelli incrementali

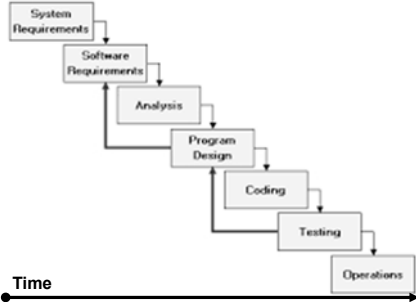
- ❑ **Possono produrre valore a ogni incremento**
 - Un insieme di funzionalità diventa presto disponibile
 - I primi incrementi possono essere il frutto di prototipazione
 - Che aiutano a fissare meglio i requisiti per gli incrementi successivi
- ❑ **Ogni incremento riduce il rischio di fallimento**
 - Senza però azzerarlo a causa dei costi aggiuntivi derivanti dalla caduta nell'iterazione
- ❑ **Le funzionalità più utili sono sviluppate nei primi incrementi**
 - Quindi attraversano più spesso la verifica
 - E per questo diventano via via più stabili

Dipartimento di Informatica, Università di Pisa13/34




Il ciclo di vita del SW

Rischi dei modelli iterativi



Ogni iterazione comporta un ritorno all'indietro nella direzione opposta all'avanzamento del tempo

Dipartimento di Informatica, Università di Pisa15/34




Il ciclo di vita del SW

Vantaggi dei modelli iterativi

- ❑ **Sono applicabili a qualunque modello di ciclo di vita**
 - Con opportuni vincoli
- ❑ **Consentono maggior capacità di adattamento**
 - Evoluzione di problemi, requisiti utente, soluzioni e tecnologie
- ❑ **Ma comportano il rischio di non convergenza**
- ❑ **Soluzione generale**
 - Decomporre la realizzazione del sistema
 - Identificare e trattare prima le parti più critiche
 - Quelle più complesse oppure quelle i cui requisiti vanno maggiormente chiariti
 - Limitando superiormente il numero di iterazioni

Dipartimento di Informatica, Università di Pisa14/34

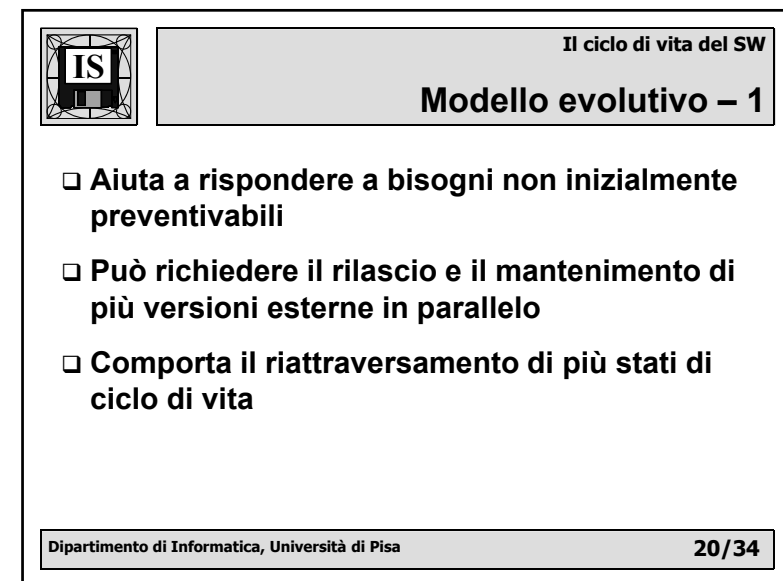
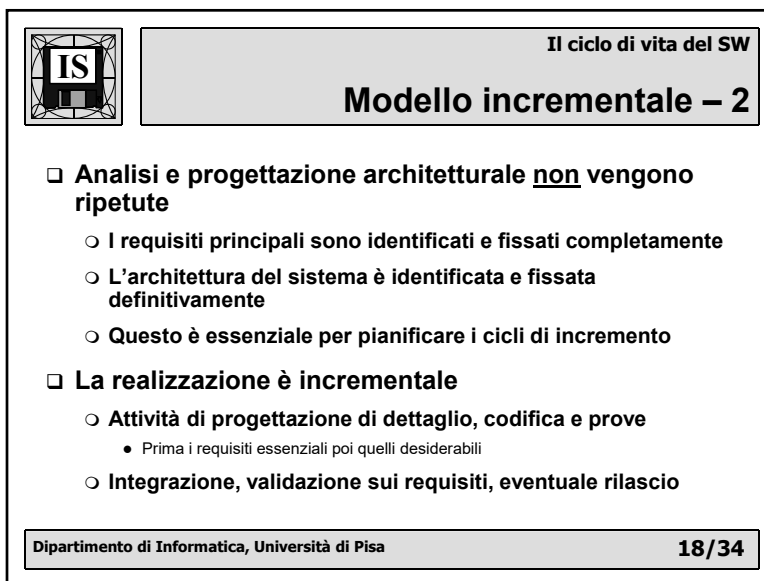
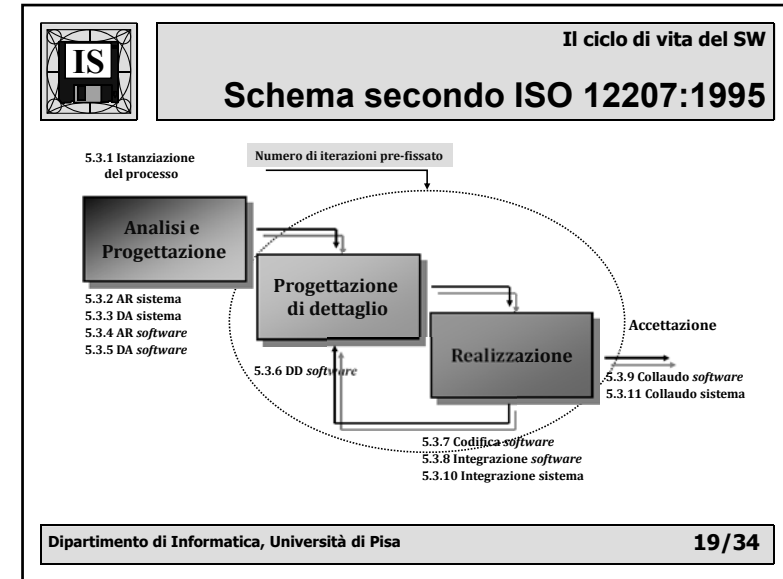
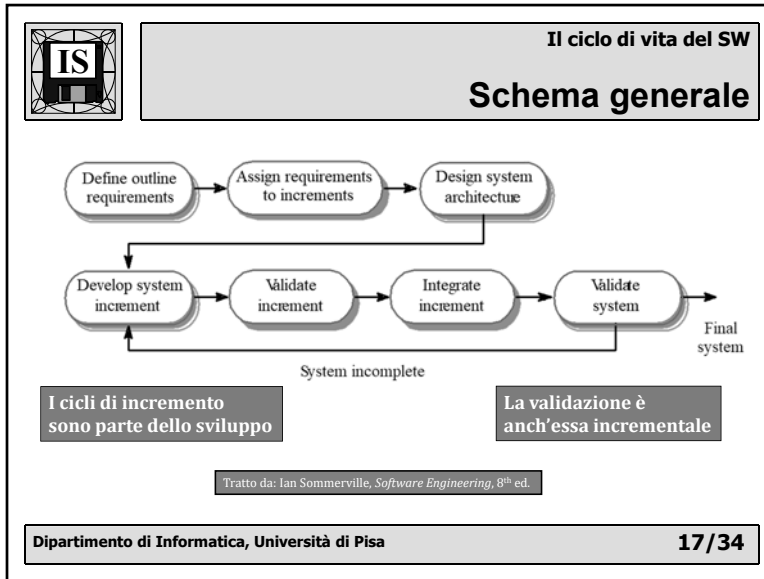


Il ciclo di vita del SW

Modello incrementale – 1

- ❑ **Prevede rilasci multipli e successivi**
 - Ciascuno realizza un incremento di funzionalità
- ❑ **I requisiti sono classificati e trattati in base alla loro importanza strategica**
 - I primi incrementi puntano a soddisfare i requisiti più importanti sul piano strategico
 - Così i requisiti importanti diventano presto stabili
 - E quelli meno importanti hanno più tempo per stabilizzarsi e armonizzarsi con lo stato del sistema

Dipartimento di Informatica, Università di Pisa16/34



Il ciclo di vita del SW

Modello evolutivo – 2

- ❑ **Analisi preliminare**
 - Per identificare i requisiti di massima
 - Per definire l'architettura di massima
 - Per pianificare i passi di analisi e realizzazione evolutiva
- ❑ **Analisi e realizzazione di una evoluzione**
 - Per raffinamento ed estensione dell'analisi
 - Per progettazione, codifica, prove e integrazione
- ❑ **Rilascio di prototipi sempre più completi**
 - Fino all'accettazione finale

Dipartimento di Informatica, Università di Pisa

21/34

Il ciclo di vita del SW

Schema secondo ISO 12207:1995

Dipartimento di Informatica, Università di Pisa

23/34

Il ciclo di vita del SW

Schema generale

Dipartimento di Informatica, Università di Pisa

22/34


Il ciclo di vita del SW

Modello a componenti

- ❑ **Molto di quello che ci serve fare è già stato fatto e molto di quello che faremo ci potrà servire ancora**
 - L'analisi dei requisiti viene adattata alle possibilità di riuso
- ❑ **Massima attenzione al riuso sistematico di componenti presistenti proprie oppure di terzi ("off-the-shelf")**

Dipartimento di Informatica, Università di Pisa

24/34




Il ciclo di vita del SW

Metodi agili – 1

- ❑ **Nascono alla fine degli '90 come reazione alla eccessiva rigidità dei modelli allora prevalenti**
 - <http://agilemanifesto.org/>
- ❑ **Si basano su quattro principi fondanti**
 - *Individuals and interactions over processes and tools*
 - L'eccessiva rigidità ostacola l'emergere del valore
 - *Working software over comprehensive documentation*
 - La documentazione non sempre corrisponde a SW funzionante
 - *Customer collaboration over contract negotiation*
 - L'interazione con gli stakeholder va incentivata e non ingessata
 - *Responding to change over following a plan*
 - La capacità di adattamento al cambiare delle situazioni è importante

Dipartimento di Informatica, Università di Pisa

25/34




Il ciclo di vita del SW

Metodi agili – 2

- ❑ **L'idea base è il concetto di “user story”**
 - Un compito significativo che l'utente vuole svolgere con il SW richiesto
- ❑ **Ogni “user story” è definita da**
 - Un documento di descrizione del problema individuato
 - La minuta delle conversazioni con gli *stakeholder* effettuate per discutere il problema e comprenderlo insieme
 - La strategia da usare per confermare che il SW realizzato soddisfi gli obiettivi del problema

Dipartimento di Informatica, Università di Pisa

27/34




Il ciclo di vita del SW

Osservazioni

- ❑ **Il SW senza documentazione è un costo e non un valore**
 - Commentare il codice non basta → serve motivare e spiegare le scelte realizzative
- ❑ **Senza un piano non si possono valutare rischi e avanzamenti**
 - La sola misurazione di consuntivo non può bastare
- ❑ **Cambiare si può ma con consapevolezza del rapporto costo/benefici**

Dipartimento di Informatica, Università di Pisa

26/34



Il ciclo di vita del SW

Metodi agili – 3

- ❑ **Migliori assunti base**
 - Suddividere il lavoro in piccoli incrementi a valore aggiunto che possono anche essere sviluppati indipendentemente
 - Sviluppare tali incrementi in una sequenza continua dall'analisi all'integrazione
- ❑ **Obiettivi strategici**
 - Poter costantemente dimostrare al cliente quanto è stato fatto
 - Verificare l'avanzamento tramite progresso reale
 - Dare agli sviluppatori la soddisfazione del risultato
 - Assicurare che l'intero prodotto SW è ben integrato e verificato
- ❑ **Esempi**
 - Scrum (caos che nasconde organizzazione), Kanban (*just-in-time*), Scrumban

Dipartimento di Informatica, Università di Pisa

28/34

Il ciclo di vita del software

Il ciclo di vita del SW

Scrum – 1

- **Product Backlog**
Requisiti e funzionalità del prodotto
- **Sprint Backlog**
Insieme di storie del prossimo sprint

- **Sprint**
Fase operativa di sviluppo
Durata media 2 - 4 settimane
Prodotto potenzialmente vendibile

Dipartimento di Informatica, Università di Pisa

29/34

Il ciclo di vita del SW

Il ciclo di vita secondo SEMAT / 1

Identified
Solution Needed
Value Established
Viable
Addressed
Benefit Accrued

Stakeholders

Recognized
Represented
Involved
In Agreement
Satisfied for Deployment
Satisfied in Use

www.ivarjacobson.com/semat

Sheet 1 of 2

Dipartimento di Informatica, Università di Pisa

31/34

Il ciclo di vita del SW

Scrum – 2

- **Sprint Planning**
Pianificazione dello sprint
- **Sprint Review**
Controllo prodotti dello sprint

- **Daily Scrum**
Controllo giornaliero avanzamento
- **Sprint Retrospective**
Controllo qualità sullo sprint

Dipartimento di Informatica, Università di Pisa

30/34

Il ciclo di vita del SW

Il ciclo di vita secondo SEMAT / 2

Architecture Selected
Demonstrable
Useable
Ready
Operational
Retired

Team


Seeded
Formed
Collaborating
Performing
Adjourned

www.ivarjacobson.com/semat

Sheet 2 of 2

Dipartimento di Informatica, Università di Pisa

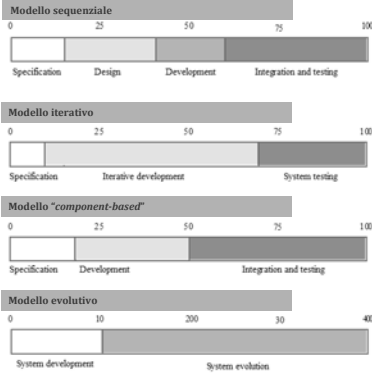
32/34



Il ciclo di vita del SW


Ripartizione dei costi nei modelli

- ❑ Applicazioni “normali”
 - ~ 60% → realizzazione
 - ~ 40% → qualifica
- ❑ I costi complessivi variano al variare del dominio e del tipo di sistema
- ❑ La ripartizione dei costi sulle fasi varia al variare del modello e del dominio
 - Sistemi critici: > 60% qualifica



Tratto da: Ian Sommerville, *Software Engineering*, 8th ed.

Dipartimento di Informatica, Università di Pisa**33/34**



Il ciclo di vita del SW

Riferimenti

- ❑ W.W. Royce, “Managing the development of large software systems: concepts and techniques”, Atti della conferenza “Wescon '70”, agosto 1970
- ❑ B.W. Bohem, “A spiral model of software development and enhancement”, IEEE Software, maggio 1998
- ❑ Center for Software Engineering, http://sunset.usc.edu/research/spiral_model
- ❑ ISO/IEC TR 15271:1998, Information Technology – Guide for ISO/IEC 12207
- ❑ Scrum: http://www.scrumalliance.org/learn_about_scrum

Dipartimento di Informatica, Università di Pisa**34/34**