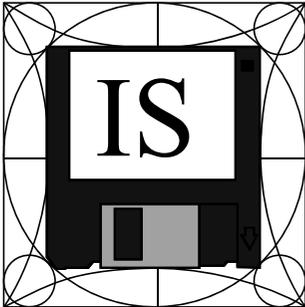




Verifica e validazione: introduzione



Ingegneria del Software
V. Ambriola, G.A. Cignoni,
C. Montanero, L. Semini

Aggiornamenti di: T. Vardanega (UniPD)

Dipartimento di Informatica, Università di Pisa

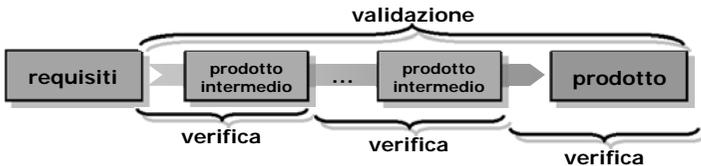
1/28



Verifica e validazione

Verifica e validazione – 2

- ❑ La verifica si occupa di accertare che l'esecuzione delle attività di processi svolti nella fase in esame non abbia introdotto errori nel prodotto
- ❑ La validazione si occupa di accertare che il prodotto realizzato sia conforme alle attese



The diagram illustrates the software development process. It starts with 'requisiti' (requirements), followed by 'prodotto intermedio' (intermediate product), then an ellipsis, then another 'prodotto intermedio', and finally 'prodotto' (product). Brackets below the flow indicate 'verifica' (verification) for each transition. A larger bracket above the flow from the second intermediate product to the final product is labeled 'validazione' (validation).

Dipartimento di Informatica, Università di Pisa

3/28



Verifica e validazione

Verifica e validazione – 1

- ❑ **Software verification**
 - Provides objective evidence that the design outputs of a particular phase of the software development life cycle meet all of the specified requirements for that phase
 - Software verification looks for consistency, completeness, and correctness of the software and its supporting documentation, as it is being developed, and provides support for a subsequent conclusion that software is validated
- ❑ **Software validation**
 - Confirmation by examination and provision of objective evidence that the software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled

Dipartimento di Informatica, Università di Pisa

2/28



Verifica e validazione

Forme di verifica

- ❑ **Analisi statica**
 - Non richiede esecuzione del prodotto SW in alcuna sua parte
 - Essenziale quando il sistema non è ancora completamente disponibile
 - Studia caratteristiche del codice sorgente (talvolta anche del codice oggetto) e della documentazione associata
 - Conformità a regole date, assenza di difetti, presenza di proprietà positive
- ❑ **Analisi dinamica**
 - Richieste esecuzione del programma
 - Viene effettuata tramite prove (*test*) ←
 - Usata sia nella verifica che nella validazione

Dipartimento di Informatica, Università di Pisa

4/28

Verifica e validazione

Verifica e validazione – 2

Dipartimento di Informatica, Università di Pisa

5/28

Verifica e validazione

Analisi dinamica: definizioni

□ Unità

- La più piccola quantità di SW che conviene verificare da sola
 - Tipicamente prodotta da un singolo programmatore
- La sua natura specifica dipende dal linguaggio di programmazione in uso
 - Va sempre intesa in senso architeturale ←
 - Non linee di codice ma entità di strutturazione (procedura, classe, package)

□ Il «modulo» è parte dell'unità

□ Il «componente» integra più unità

Dipartimento di Informatica, Università di Pisa

7/28

Verifica e validazione

Analisi dinamica: ambiente di prova

□ La ripetibilità è requisito essenziale

- Ambiente (HW, stato iniziale, ...)
- Specifica (ingressi richiesti, comportamenti attesi)
- Procedure (esecuzione, analisi dei risultati)

□ Strumenti

- **Driver** componente attiva fittizia per pilotare il test
- **Stub** componente passiva fittizia per simulare una parte del sistema ma non oggetto di test
- **Logger** componente non intrusivo di registrazione dei dati di esecuzione per analisi dei risultati

Dipartimento di Informatica, Università di Pisa

6/28

Verifica e validazione

Stub e driver

Unità U composta dai moduli M1, M2, M3

La direzione degli archi indica la relazione d'uso

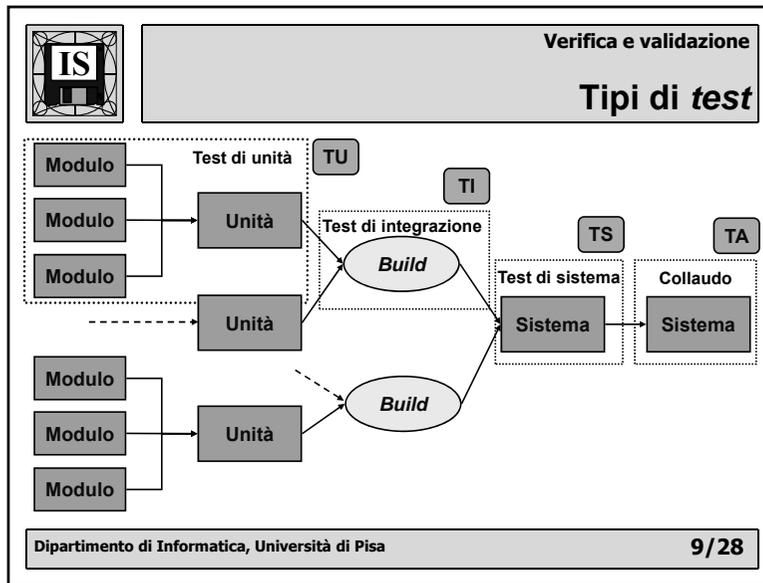
Test di unità su U

Con driver

Con stub

Dipartimento di Informatica, Università di Pisa

8/28



Verifica e validazione

Test di integrazione

- ❑ Per la costruzione e la verifica incrementale del sistema
 - Componenti sviluppati in parallelo e verificati incrementalmente
 - In condizioni ottimali l'integrazione è priva di problemi
- ❑ Quali problemi rileva
 - Errori residui nella realizzazione dei componenti
 - Modifica delle interfacce o cambiamenti nei requisiti
 - Riutilizzo di componenti dal comportamento oscuro o inadatto
 - Integrazione con altre applicazioni non ben conosciute

Dipartimento di Informatica, Università di Pisa 11/28

Verifica e validazione

Test di unità

- ❑ Attività di analisi dinamica
 - Con il supporto di attività mirate di analisi statica
 - Per determinare limiti di iterazioni, flusso di programma, valori di variabili, ecc.
 - Si svolge con il massimo grado di parallelismo
- ❑ Responsabilità
 - Dello stesso programmatore per le unità più semplici
 - Di un verificatore indipendente (meglio un automa) altrimenti
- ❑ Obiettivi
 - Verificare la correttezza del codice «*as implemented*»

Dipartimento di Informatica, Università di Pisa 10/28

Verifica e validazione

Test di sistema e collaudo

- ❑ Validazione
 - Test di sistema come attività interna del fornitore
 - Per accertare la copertura dei requisiti SW
 - Collaudo come attività supervisionata dal committente
 - Per dimostrazione di conformità del prodotto sulla base di casi di prova specificati nel o implicati dal contratto
- ❑ Implicazioni contrattuali
 - Il collaudo è attività formale
 - Al collaudo segue il rilascio del prodotto (con eventuale garanzia) e la fine della commessa (con eventuale manutenzione)

Dipartimento di Informatica, Università di Pisa 12/28



Verifica e validazione
Test di regressione

- ❑ L'insieme di *test* (TU, TI) necessari ad accertare che la modifica di una parte P di S non causi errori in P o nelle altre parti di S che hanno relazioni con P
 - Ripetizione di *test* già previsti ed effettuati per ogni parte coinvolta
- ❑ Modifiche effettuate per aggiunta, correzione o rimozione, non devono pregiudicare le funzionalità già verificate
 - Il rischio aumenta all'aumentare dell'accoppiamento e al diminuire dell'incapsulazione

Dipartimento di Informatica, Università di Pisa

13/28



Verifica e validazione
Metodi di lettura

- ❑ **Inspection e Walkthrough**
- ❑ **Metodi pratici**
 - Basati sulla lettura della documentazione sul prodotto
 - Di efficacia dipendente dall'esperienza dei verificatori
 - Nell'organizzare le attività di verifica
 - Nel documentare le attività svolte e i risultati ottenuti
- ❑ **Modalità relativamente complementari**

Dipartimento di Informatica, Università di Pisa

15/28



Verifica e validazione
Forme di analisi statica

- ❑ Non richiedono esecuzione di parti del sistema SW
- ❑ Applicano a ogni prodotto di processo (non solo SW)
 - Per tutti i processi attivati nel progetto
- ❑ **Metodi di lettura (*desk check*)**
 - Impiegati solo per prodotti semplici
- ❑ **Metodi formali**
 - Basati sulla prova assistita di proprietà
 - La cui dimostrazione dinamica può essere eccessivamente onerosa
 - Verifica di equivalenza o generazione automatica

Dipartimento di Informatica, Università di Pisa

14/28



Verifica e validazione
Inspection

- ❑ **Obiettivi**
 - Rivelare la presenza di difetti
 - Eseguire una lettura mirata [del prodotto in esame]
- ❑ **Agenti**
 - Verificatori distinti e separati dai programmatori
- ❑ **Strategia**
 - Focalizzare la ricerca su presupposti
 - *Error guessing*

Dipartimento di Informatica, Università di Pisa

16/28



Verifica e validazione
Attività di *inspection*

- Fase 1: pianificazione
- Fase 2: definizione della lista di controllo ←
- Fase 3: lettura
- Fase 4: correzione dei difetti
- In ogni fase
 - Documentazione come rapporto delle attività svolte

Dipartimento di Informatica, Università di Pisa17/28



Verifica e validazione
Attività di *walkthrough*

- Fase 1: pianificazione
- Fase 2: lettura
- Fase 3: discussione ←
- Fase 4: correzione dei difetti
- In ogni fase
 - Documentazione come rapporto delle attività svolte

Dipartimento di Informatica, Università di Pisa19/28



Verifica e validazione
Walkthrough

- Obiettivo**
 - Rivelare la presenza di difetti
 - Eseguire una lettura critica [del prodotto in esame]
 - A largo spettro
 - Senza l'assunzione di presupposti
- Agenti**
 - Gruppi misti ispettori/sviluppatori ma con ruoli ben distinti
- Strategia (per il codice)**
 - Percorrerlo simulandone possibili esecuzioni

Dipartimento di Informatica, Università di Pisa18/28



Verifica e validazione
Inspection contro walkthrough

- Affinità**
 - Controlli basati su *desk check*
 - Programmatori e verificatori su fronti opposti
 - Documentazione formale
- Differenze**
 - *Inspection* basato su (errori) presupposti
 - *Walkthrough* richiede maggiore attenzione
 - *Walkthrough* più collaborativo
 - *Inspection* più rapido

Dipartimento di Informatica, Università di Pisa20/28



Verifica e validazione
Attività di *quality assurance*

- Serve a raccogliere evidenza oggettiva e di qualità tempestivamente**
 - A fronte di specifiche metriche e di obiettivi definiti
 - Per dare evidenza oggettiva
 - Per controllo (interno) e accertamento (esterno)

- ISO/IEC 9126 come riferimento**
 - Quali strumenti per quali caratteristiche?
 - La qualità in uso è valutata a posteriori

Dipartimento di Informatica, Università di Pisa**21/28**



Verifica e validazione
Affidabilità

- Dimostrabile tramite combinazione di prove e analisi statica**
 - Analisi statica come attività preliminare
 - Prove a completamento

- Liste di controllo rispetto ai relativi requisiti**
 - Robustezza
 - Capacità di ripristino e recupero da errori
 - Adesione alle norme e alle prescrizioni

- Valutazione empirica di maturità**

Dipartimento di Informatica, Università di Pisa**23/28**



Verifica e validazione
Funzionalità

- Dimostrabile tramite prove**
 - Analisi statica come attività preliminare

- Liste di controllo rispetto ai relativi requisiti**
 - Completezza ed economicità
 - Tutte le funzionalità richieste per tutti e soli i componenti necessari
 - Interoperabilità
 - Accertata la compatibilità tra le soluzioni realizzative adottate
 - Sicurezza del prodotto e dei suoi componenti
 - Adesione alle norme e alle prescrizioni

- Valutazione empirica di accuratezza**

Dipartimento di Informatica, Università di Pisa**22/28**



Verifica e validazione
Usabilità

- Le prove sono imprescindibili**
 - Analisi statica come attività complementare

- Liste di controllo rispetto ai manuali d'uso**
 - Comprensibilità
 - Apprendibilità
 - Adesione a norme e prescrizioni

- Questionari sottomessi agli utenti**
 - Facilità e piacevolezza d'uso

Dipartimento di Informatica, Università di Pisa**24/28**



Verifica e validazione
Efficienza

- Le prove sono necessarie**
 - Analisi statica come attività complementare
- Liste di controllo rispetto alle norme di codifica**
 - Quelle che puntano all'efficienza nel tempo e nello spazio
- Margini di miglioramento e confidenza**
 - L'efficienza provata fornisce confidenza nel prodotto
 - L'analisi statica fornisce indicazioni specifiche sui margini di miglioramento prestazionale

Dipartimento di Informatica, Università di Pisa**25/28**



Verifica e validazione
Portabilità

- Analisi statica come strumento ideale**
- Liste di controllo rispetto a specifiche norme di codifica**
 - Adattabilità
- Prove come strumento complementare**
 - Facilità d'installazione e di sostituzione
 - Compatibilità ambientale

Dipartimento di Informatica, Università di Pisa**27/28**



Verifica e validazione
Manutenibilità

- Analisi statica come strumento ideale**
- Liste di controllo**
 - Rispetto a specifiche norme di codifica
 - Analizzabilità
 - Modificabilità
 - Rispetto alle prove per accertarne
 - Ripetibilità
 - Verificabilità
- Prove di stabilità**

Dipartimento di Informatica, Università di Pisa**26/28**



Verifica e validazione
Riferimenti

- Standard for Software Component Testing, British Computer Society SIGIST, 1997**
- M.E. Fagan, Advances in Software Inspection, *IEEE Transaction on Software Engineering*, luglio 1986**
- G.A. Cignoni, P. De Risi, "Il test e la qualità del software", Il Sole 24 Ore, 1998**

Dipartimento di Informatica, Università di Pisa**28/28**