

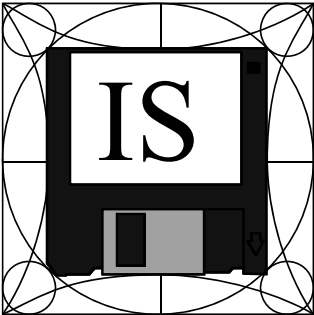


Amministrazione di progetto

Ingegneria del Software

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

Aggiornamenti: T. Vardanega (UniPD)



Dipartimento di Informatica, Università di Pisa1/35



Amministrazione di progetto

Cosa è un servizio

- Mezzo per fornire valore all'utente agevolando il raggiungimento dei suoi obiettivi senza doverne sostenere gli specifici costi e rischi
 - Esempio di risultato desiderato
 - Massima efficacia nel prodotto con massima efficienza nel lavoro
 - Esempio di valore fornito
 - Abbattimento delle attività improduttive
 - Esempio di costi e rischi
 - Ciascun utente sceglie i propri strumenti e segue le proprie regole e procedure

Dipartimento di Informatica, Università di Pisa3/35



Amministrazione di progetto

Amministrare un progetto

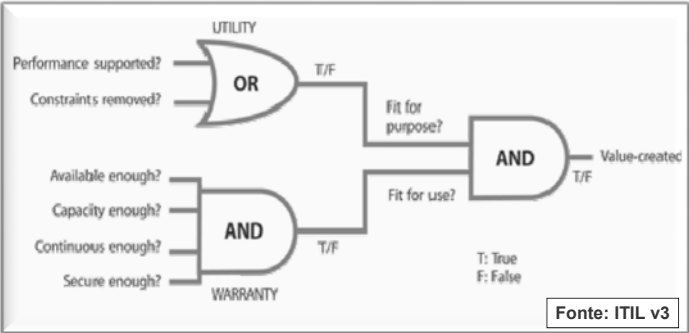
- Equipaggiare, organizzare e gestire l'ambiente di lavoro e di produzione
 - Regole
 - Procedure
 - Strumenti e servizi
 - A supporto di tutti i processi istanziati nel progetto
- L'amministratore non compie scelte gestionali ma attua le scelte tecnologiche concordate con i responsabili aziendali e di progetto

Dipartimento di Informatica, Università di Pisa2/35



Amministrazione di progetto

Il valore del servizio



Fonte: ITIL v3

Dipartimento di Informatica, Università di Pisa4/35



Amministrazione di progetto

Attività di amministrazione

- ❑ Redazione e manutenzione di regole e procedure di lavoro
 - La loro approvazione spetta al responsabile di progetto
- ❑ Reperimento, organizzazione, gestione e manutenzione delle risorse informatiche per l'erogazione dei servizi di supporto
 - Ambiente, infrastruttura, strumenti, prodotti, documenti

Dipartimento di Informatica, Università di Pisa

5/35



Amministrazione di progetto

Disponibilità e diffusione

- ❑ I documenti sono utili solo se sono sempre disponibili
 - Chiaramente identificati
 - Corretti nei contenuti
 - Verificati e approvati
 - Aggiornati, datati e dotati di versione
- ❑ La loro diffusione deve essere controllata
 - I destinatari devono essere chiaramente identificati
 - Ogni documento ha una sua lista di distribuzione
 - L'amministratore gestisce le liste di distribuzione e ne assicura il rispetto

Dipartimento di Informatica, Università di Pisa

7/35



Amministrazione di progetto

Documentazione di progetto

- ❑ Tutto ciò che descrive gli ingressi e le uscite delle attività necessarie al progetto
 - Riguardo al prodotto
 - Riguardo al processo
- ❑ Documenti di sviluppo
 - Documentazione fornita dal cliente
 - Diagrammi di progettazione
 - Codice
 - Piani di qualifica e risultati delle prove
 - Documentazione di accompagnamento del prodotto
- ❑ Documenti di gestione del progetto
 - Documenti contrattuali
 - Piani e consuntivi delle attività
 - Piani di qualità

Dipartimento di Informatica, Università di Pisa

6/35



Amministrazione di progetto

Ambiente di lavoro

- ❑ Quanto serve ai processi di produzione
 - Processi di sviluppo e di supporto
 - L'ambiente è fatto da persone, ruoli e procedure, infrastruttura
- ❑ La sua qualità determina la produttività
 - Influisce sulla qualità del processo e sulla qualità del prodotto
- ❑ L'ambiente di lavoro deve essere
 - Completo: offre tutto il necessario per svolgere le attività previste
 - Ordinato: è facile trovare ciò che vi si cerca
 - Aggiornato: il materiale obsoleto non deve causare intralcio

Dipartimento di Informatica, Università di Pisa

8/35



Amministrazione di progetto

Supporto di processi – 1

❑ Gestione di progetto

○ Pianificazione, stima e controllo dei costi

- Tramite *issue tracking*, derivato dall'ambito del Service Management

○ Allocazione e gestione delle risorse

- Redazione e consultazione di diagrammi di Gantt e PERT (p.es., <http://www.ganttproject.biz/>)

○ Strumenti collaborativi di controllo gestionale e di qualità e di coordinamento attività

- Assembla (<http://www.assembla.com>), per *task and issue management*
- Maven (<http://maven.apache.org>)
- Jira (<http://www.atlassian.com/software/jira>)

❑ Gestione documentale

○ TWiki (<http://www.twiki.org>)

○ Google Docs (<http://docs.google.com>)

○ Versionamento e configurazione (ne riparliamo tra poco ...)

Dipartimento di Informatica, Università di Pisa

9/35



Amministrazione di progetto

Configurazione – 1

❑ Un prodotto SW è l'unione di parti distinte unite insieme secondo regole rigorose

○ Specifiche, progetti, programmi, dati di verifica, manualistica

❑ Le regole di configurazione vanno pianificate

○ Le responsabilità di configurazione vanno assegnate

❑ La gestione di configurazione va automatizzata

○ Servono strumenti adatti

- Configuration Management
- Build

Dipartimento di Informatica, Università di Pisa

11/35



Amministrazione di progetto

Supporto di processi – 2

❑ Analisi e progettazione

○ Analisi, gestione e tracciamento dei requisiti

- eRequirements (<http://erequirements.com/app>)

○ Supporto alle metodologie

- UML (p.es., <http://www.eclipse.org/papyrus/>, <http://www.modelio.org/>)

❑ Codifica e integrazione

○ Ambienti integrati di sviluppo (p.es., <http://www.eclipse.org/>)

○ Strumenti di integrazione continua (*continuous integration*)

- Hudson (<http://hudson.dev.java.net>)
- CruiseControl (<http://cruisecontrol.sourceforge.net/index.html>)
- Merlin ToolChain (<http://merlintoolchain.sourceforge.net>)

○ Misurazione e analisi del codice prima dell'integrazione

○ Generazione ed esecuzione automatica delle prove prima dell'integrazione

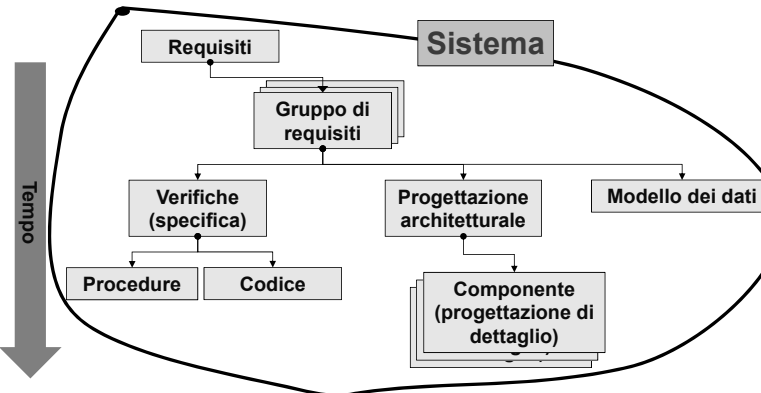
Dipartimento di Informatica, Università di Pisa

10/35



Amministrazione di progetto

Configurazione – 2



```
graph TD
    R[Requisiti] --> GR[Gruppo di requisiti]
    GR --> V[Verifiche (specifica)]
    GR --> PA[Progettazione architetturale]
    V --> P[Procedure]
    V --> C[Codice]
    PA --> MD[Modello dei dati]
    PA --> CP[Componente (progettazione di dettaglio)]
    MD --> S[Sistema]
    CP --> S
    S --> R
```

Dipartimento di Informatica, Università di Pisa

12/35

IS 2015 - Ingegneria del Software

3



Amministrare di progetto

Gestione di configurazione – 1

❑ Obiettivi

- Mettere in sicurezza le *baseline* che consolidano gli stati di avanzamento del processo di sviluppo
- Prevenire sovrascritture accidentali su parti
- Consentire ritorno a configurazioni precedenti
- Permettere il recupero da perdite accidentali

❑ Attività

- Identificazione di configurazione
- Controllo di *baseline*
- Gestione delle modifiche
- Controllo di versione

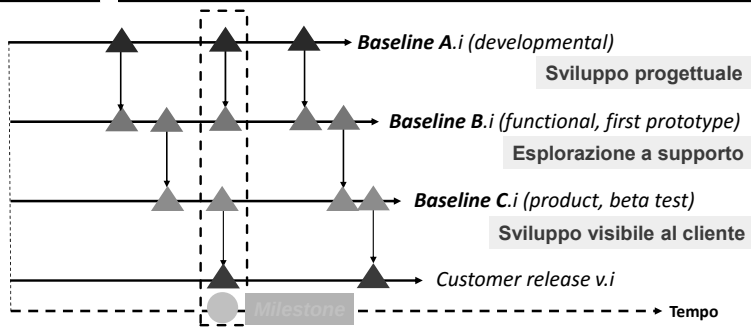
Dipartimento di Informatica, Università di Pisa

13/35



Amministrare di progetto

Evoluzione di *baseline*



La successione di *baseline* (per ogni asse orizzontale e per le sue correlazioni verticali) va gestita da processi dedicati
Ogni *milestone* di calendario è associata a uno specifico insieme di *baseline*

Dipartimento di Informatica, Università di Pisa

15/35



Amministrare di progetto

Gestione di configurazione – 2

❑ Identificazione di configurazione

- Quali parti (*configuration item*, CI) compongono il prodotto
- Ogni CI ha una identità unica
 - ID, nome, data, autore, registro delle modifiche, stato corrente

❑ Controllo di *baseline*

- L'insieme di CI consolidato a una specifica *milestone*
 - Base verificata, approvata e garantita per la prosecuzione dello sviluppo
- L'esistenza di *baseline* ben identificate permette
 - Riproducibilità
 - Tracciabilità
 - Analisi e confronto

Dipartimento di Informatica, Università di Pisa

14/35



Amministrare di progetto

Buone qualità di *milestone*

1. Specifiche per obiettivi
2. Delimitate per ampiezza e ambizioni
3. Incrementali per contenuti
4. Coerenti con e rilevanti per la strategia di progetto
5. Misurabili per quantità di impegno necessario
6. Traducibili in compiti assegnabili
7. Raggiungibili
8. Puntuali rispetto alle esigenze di calendario
9. Dimostrabili agli *stakeholder*

Dipartimento di Informatica, Università di Pisa

16/35



Amministrazione di progetto

Gestione delle modifiche – 1

- ❑ Le richieste di modifiche hanno origine da
 - Utenti (difetti o mancanze)
 - Sviluppatori (idem)
 - Competizione (valore aggiunto)
- ❑ Le richieste di modifica vanno sottoposte a un rigoroso processo di analisi, decisione, realizzazione e verifica
 - Sempre tenendo traccia dello stato precedente

Dipartimento di Informatica, Università di Pisa

17/35



Amministrazione di progetto

Controllo di versione – 1

- ❑ Si appoggia su un *repository*
 - DB centralizzato nel quale risiedono - individualmente – tutti i CI di ogni *baseline* nella loro storia completa
- ❑ Permette a ciascuno di lavorare su vecchi e nuovi CI senza rischio di sovrascritture accidentali
 - *Check-out*
- ❑ E di condividere il lavorato nello spazio comune
 - *Check-in*
- ❑ Verifica la bontà di ogni modifica di *baseline*
 - *Build*

Dipartimento di Informatica, Università di Pisa

19/35



Amministrazione di progetto

Gestione delle modifiche – 2

- ❑ Ogni richiesta/proposta di modifica va inoltrata in modo formale
 - *Change request*
 - Autore, motivo, urgenza
 - Stima di fattibilità, valutazione di impatto, stima di costo
 - Decisione del responsabile
- ❑ Di ogni richiesta di modifica bisogna tenere traccia
 - *Issue tracking o ticketing*
 - Per esempio con Bugzilla
 - Stato corrente ed eventuale esito chiusura

Dipartimento di Informatica, Università di Pisa

18/35



Amministrazione di progetto

Controllo di versione – 2

- ❑ Versione
 - Istanza di prodotto funzionalmente distinta dalle altre
- ❑ Variante
 - Istanza di prodotto funzionalmente identica ad altre ma diversa per caratteristiche non funzionali
- ❑ Rilascio (*release*)
 - Istanza di prodotto resa disponibile a utenti esterni
- ❑ Tutte vanno identificate, pianificate e gestite
 - Identificazione per numero, caratteristiche, modifiche

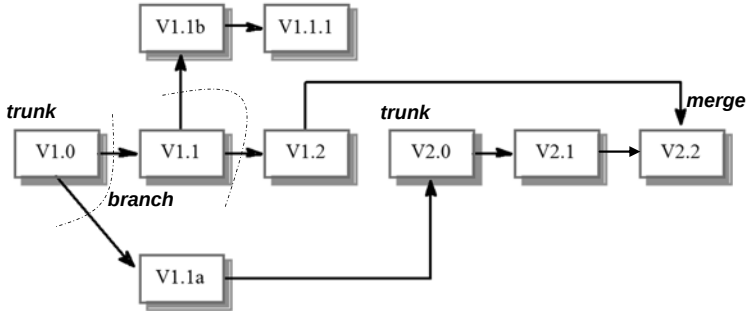
Dipartimento di Informatica, Università di Pisa

20/35



Amministrazione di progetto

Controllo di versione – 3



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

Dipartimento di Informatica, Università di Pisa

21/35



Amministrazione di progetto

Norme di progetto

- ❑ **Linee guida per le attività di sviluppo**
 - In origine precedevano processi e procedure aziendali
 - Oggi sono uno strumento operativo di complemento alle procedure
- ❑ **Contenuti**
 - Organizzazione e uso delle risorse di sviluppo
 - Convenzioni sull'uso degli strumenti di sviluppo
 - Organizzazione della comunicazione e della cooperazione
 - Norme di codifica
 - Gestione dei cambiamenti!

Dipartimento di Informatica, Università di Pisa

23/35



Amministrazione di progetto

Strumenti di lavoro

- ❑ **Produrre codice secondo regole**
 - Editore integrato con il linguaggio e con l'ambiente
- ❑ **Verificare il codice a partire dalle unità più piccole**
 - Strumenti di verifica automatica
- ❑ **Configurazione (build) per integrare le parti nel prodotto finale**
- ❑ **Versionamento per tener traccia della “storia” di ciascun CI**
- ❑ **Pianificazione**
- ❑ **Assegnazione e gestione dei compiti**

Temi della flipped classroom del 2/11/2015

Dipartimento di Informatica, Università di Pisa

22/35



Amministrazione di progetto

Organizzazione di una norma

- ❑ **Regole**
 - Convenzioni di cui si riconosce necessità e convenienza
 - Ne è richiesto (prima) e accertato (poi) il rispetto
- ❑ **Raccomandazioni**
 - Prassi desiderabile
 - Inviti e suggerimenti senza verifica di rispetto
- ❑ **Il contesto definisce la portata della norma**
 - Non tutto può essere regolato
 - Troppe regole sono di difficile attuazione e verifica

Dipartimento di Informatica, Università di Pisa

24/35



Amministrazione di progetto

Obiettivi delle norme di codifica

- ❑ Leggibilità come forma di prevenzione
 - Verificabilità
 - Manutenibilità
 - Portabilità
- ❑ Come è “scritto” il codice?
- ❑ È comprensibile a distanza di tempo?
- ❑ È comprensibile a chi non lo ha prodotto?

Dipartimento di Informatica, Università di Pisa

25/35



Amministrazione di progetto

Indentazione del codice

- ❑ Obiettivi
 - Programmazione strutturata
 - Evidenziare visivamente la struttura di un programma
- ❑ Aspetti da non sottovalutare
 - Lunghezza delle linee
 - Ampiezza dell’indentazione
 - Posizione degli fine linea nei blocchi
 - Posizione degli fine linea nelle espressioni
- ❑ Evitare guerre ideologiche sugli stili

Dipartimento di Informatica, Università di Pisa

27/35



Amministrazione di progetto

Convenzioni sui nomi

- ❑ Nel codice
 - Tipi, costanti, variabili, funzioni, ...
 - P.es., le norme Javadoc (vedi: <http://java.sun.com/2se/javadoc/>)
- ❑ Nel progetto
 - Strutturazione in moduli, file, directory, ...
- ❑ Aspetti pratici
 - Conflitti logici all’interno o all’esterno del codice
 - Abbreviazioni, per comodità o per necessità
 - Limiti intrinseci del linguaggio
 - P.es., identificazione forte o debole dei tipi (Java ↔ C)
 - Limiti degli strumenti
 - P.es., lunghezza massima dei nomi di file (p.es.: Windows 95-98)

Dipartimento di Informatica, Università di Pisa

26/35



Amministrazione di progetto

Intestazione del codice

- ❑ Obiettivi
 - Identificazione e collocamento di una unità (modulo, file)
 - Storia e responsabilità delle modifiche
- ❑ Contenuti

○ Dati dell’unità	tipo, contenuto, posizione
○ Responsabilità	autore, reparto, organizzazione
○ Copyright / copyleft	licenze, visibilità
○ Avvertenze	limiti di uso e di garanzia
○ Registro modifiche	storia, spiegazione, <u>versione</u>

Dipartimento di Informatica, Università di Pisa

28/35



Amministrazione di progetto

Intestazione – esempio 1

```
// File:      HAL_kern.H - HAL 9000 KB Data defs -*- C++ -*-
// Module:    HAL 9000 KB kernel
// Created:   1997 January 12
// Author:    Dr. Chandra - 9000 Proj., HAL Inc., Urbana, ILL
// E-Mail:    chandra@p9000.hal.com
//
// Copyright (C) 1996, 1997, Dr. Chandra, HAL Inc.
//           All rights reserved.
//
// This software and related documentation are
// distributed under license. No permission is given
// to use, copy, modify or distribute this software
// without explicit authorization of HAL Inc.
// and its licensors, if any.
//
// Software licensed to:
// NO LICENSE - For HAL internal use only.
//
// This software is provided "as is" WITHOUT ANY WARRANTY
// either expressed or implied, including, but not limited
// to, the implied warranties of MERCHANTABILITY or
// FITNESS FOR A PARTICULAR PURPOSE.
```

Dipartimento di Informatica, Università di Pisa

29/35



Amministrazione di progetto

Disciplina di programmazione

- ❑ Serve una strategia forte per costringere i programmatori a lavorare come si conviene
- ❑ Prescrizioni tipiche
 - Compilazione senza errori fatali o potenziali (*warning*)
 - Uso chiaro e coerente dei costrutti del linguaggio
 - Uso di un sottoinsieme appropriato del linguaggio
 - I costrutti di maggiore robustezza, verificabilità, leggibilità
 - Non necessariamente quelli di maggiore potenza espressa e velocità

Dipartimento di Informatica, Università di Pisa

31/35



Amministrazione di progetto

Intestazione – esempio 2

```
// BANKSEC project (IST 6087)
//
// BANKSEC-TOOLS/AUTH/RBAC/USER_ROLE
//
// Object: currentRole
// Author: N. Perwaiz
// Creation date: 10th November 2002
//
// © Lancaster University 2002
//
// Modification history
// Version      ModifierDate      Change      Reason
// 1.0         J. Jones      1/12/2002    Add header    Submitted to CM
// 1.1         N. Perwaiz    9/4/2003    New field     Change req. R07/02
```

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

Dipartimento di Informatica, Università di Pisa

30/35



Amministrazione di progetto

Leggibilità del codice

- ❑ Il codice illeggibile è disarmante e irritante
- ❑ Modificarlo costa tempo ed è rischioso
- ❑ La leggibilità facilita le attività di ispezione
- ❑ Il codice è una risorsa
- ❑ Il primo (l'ultimo) posto dove guardare

Dipartimento di Informatica, Università di Pisa

32/35

