Amministrazione di progetto

Ingegneria del Software

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

Aggiornamenti: T. Vardanega (UniPD)

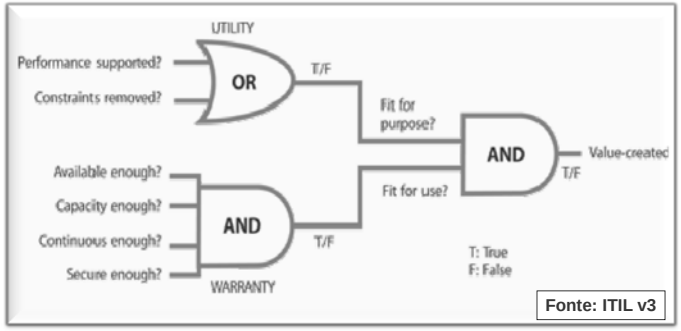
Amministrare un progetto

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

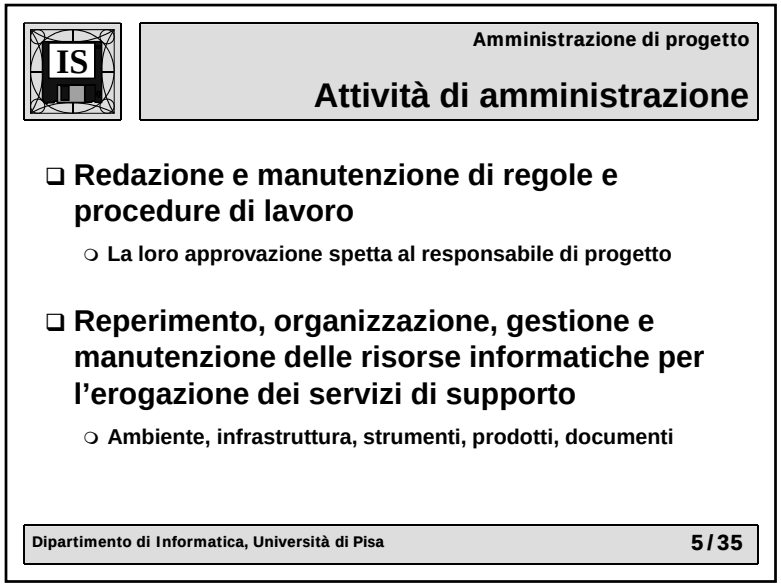
Cosa è un servizio

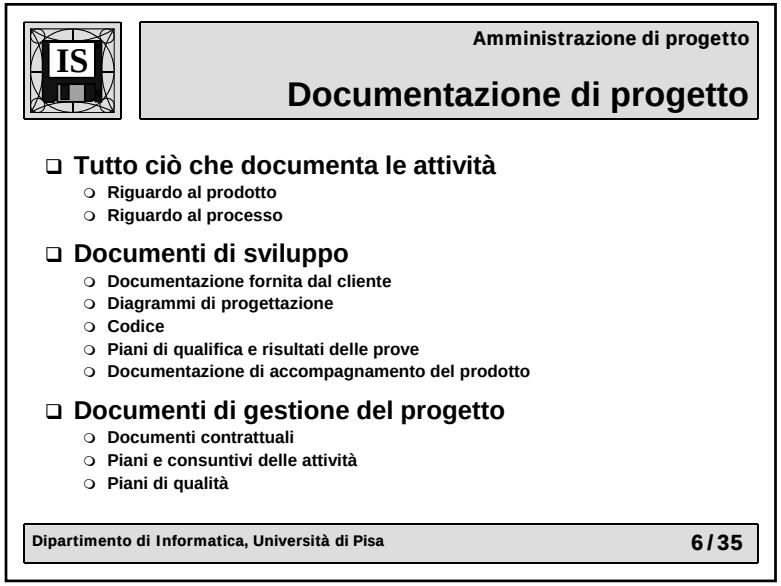
- Mezzo per fornire valore all'utente agevolando il raggiungimento dei suoi obiettivi senza doverne sostenere gli specifici costi e rischi
  - Esempio di risultato desiderato
    - Massima efficacia nel prodotto con massima efficienza nel lavoro
  - Esempio di valore fornito
    - Abbattimento delle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

Il valore del servizio

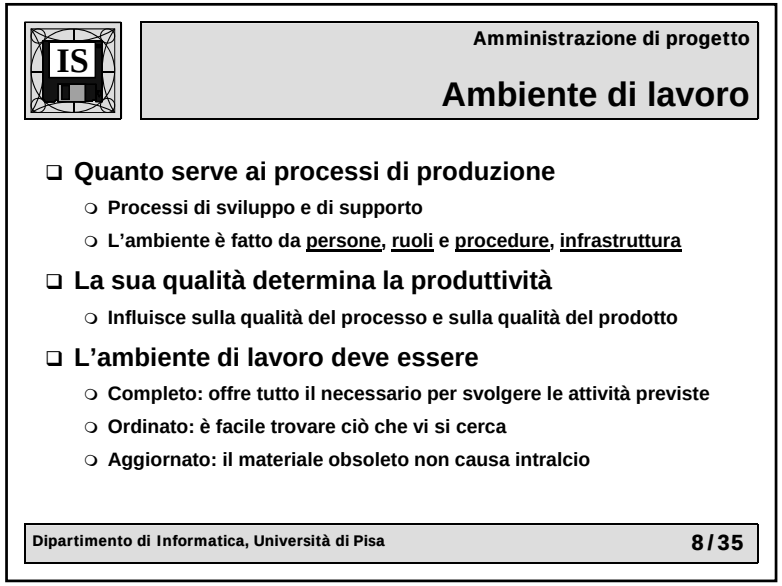


Fonte: ITIL v3


IS
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


IS
Amministrazione di progetto
Documentazione di progetto
Tutto ciò che documenta le attività
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


IS
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


IS
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 causa intralcio
Dipartimento di Informatica, Università di Pisa
8 / 35

Infrastruttura

- **Risorse HW**
  - ***Server***
    - Archivi logici centralizzati di prodotti («*repository*») sempre aggiornati e accessibili
  - **Rete**
    - Sempre operativa, protetta, accessibile agli autorizzati anche da remoto
  - **Postazioni di lavoro e dispositivi di utilità**
    - Per assicurare la massima produttività individuale
  - **Archivi fisici (documenti cartacei e altro materiale)**
- **Risorse SW**
  - **Ambienti di sviluppo, prova, studio, gestione e documentazione**
- **L'infrastruttura offre servizi posti sotto la responsabilità dell'amministratore**
  - **Funzione aziendale più spesso che ruolo a progetto**

Supporto di processi – 1

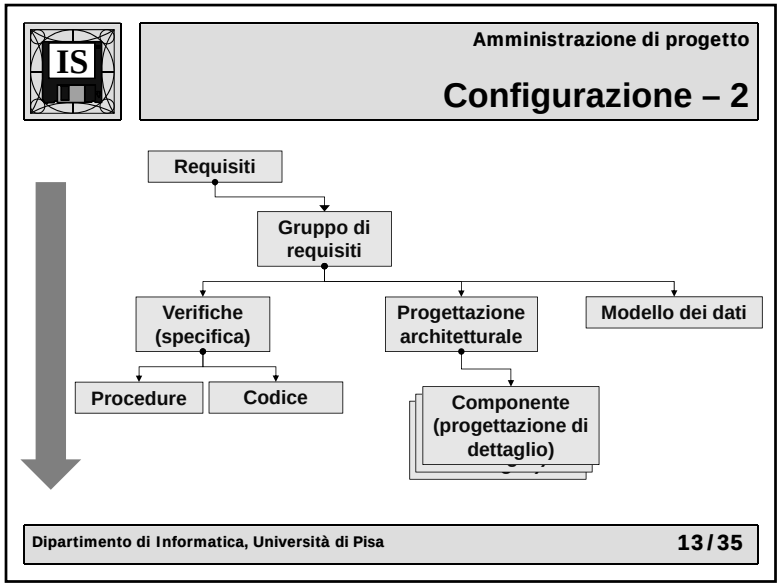
- **Gestione di progetto**
  - **Pianificazione, stima e controllo dei costi**
  - **Allocazione e gestione delle risorse**
    - Redazione e consultazione di diagrammi di Gantt e PERT (p.es., http://www.ganttproject.biz/)
  - **Strumenti collaborativi di controllo gestionale e di qualità e di coordinamento attività**
    - Assembla (http://www.assembla.com)
    - 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 ...)**

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/modeling/mdt/papyrus/)
- **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 statica del codice prima dell'integrazione**
  - **Generazione ed esecuzione automatica delle prove prima dell'integrazione**

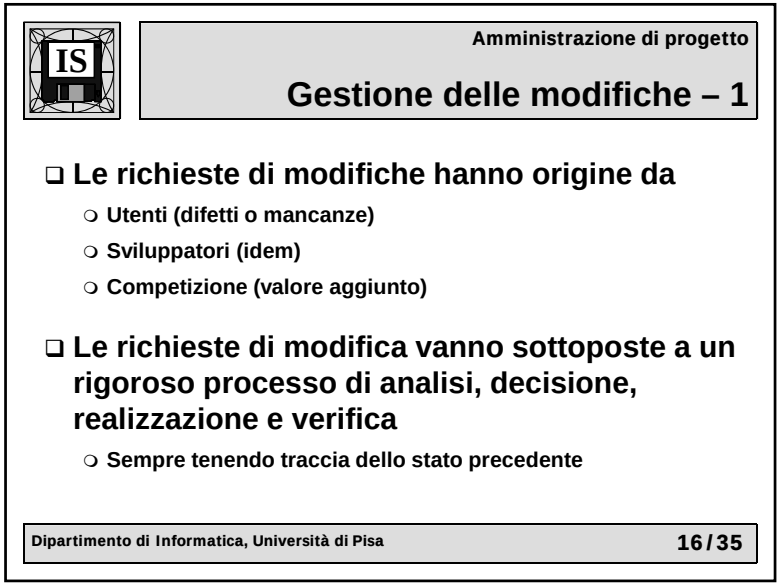
Configurazione – 1

- **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*


IS
Amministrazione di progetto
Configurazione – 2
Requisiti
Gruppo di requisiti
Verifiche (specifica)
Progettazione architetturale
Modello dei dati
Procedure
Codice
Componente (progettazione di dettaglio)
Dipartimento di Informatica, Università di Pisa
13 / 35


IS
Amministrazione 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
14 / 35


IS
Amministrazione 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
15 / 35


IS
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
16 / 35

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

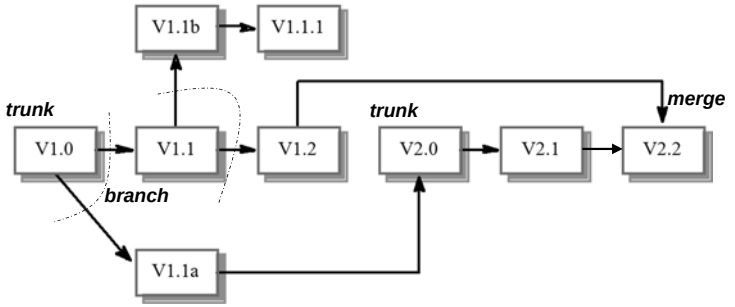
Controllo di versione – 1

- 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*

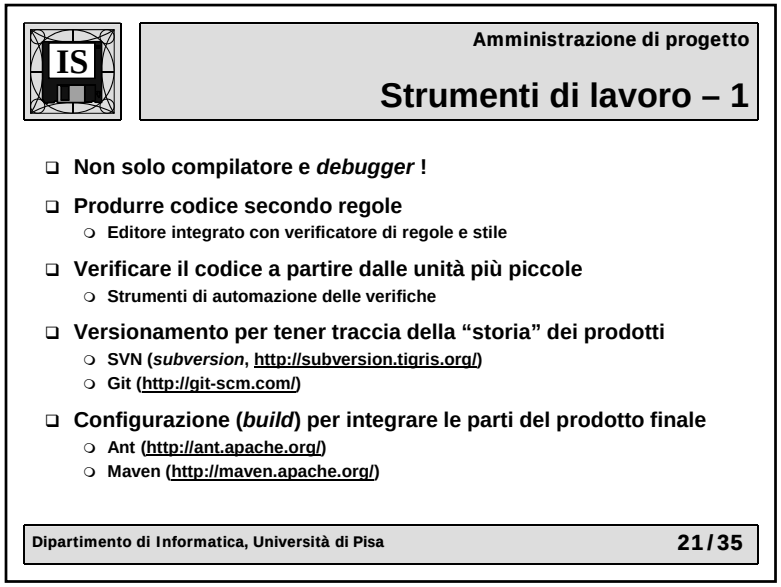
Controllo di versione – 2

- 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

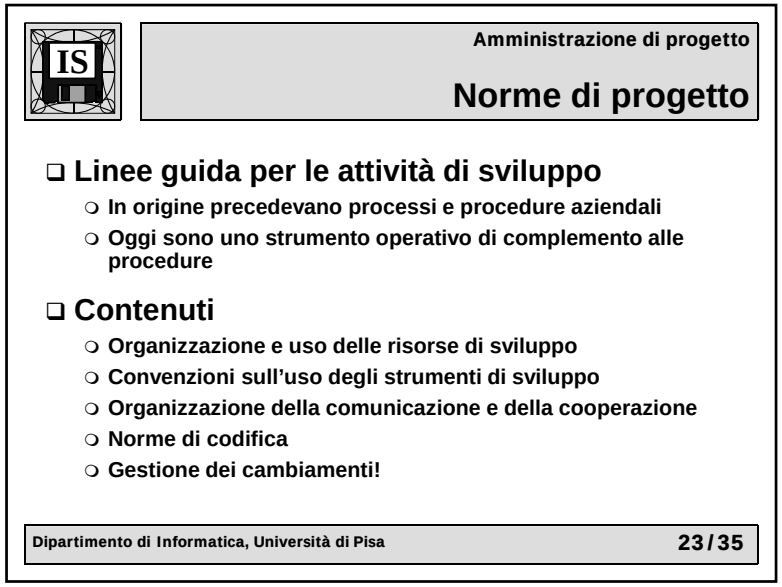
Controllo di versione – 3



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


IS
Amministrazione di progetto
Strumenti di lavoro – 1
Non solo compilatore e *debugger* !
Produrre codice secondo regole
Editore integrato con verificatore di regole e stile
Verificare il codice a partire dalle unità più piccole
Strumenti di automazione delle verifiche
Versionamento per tener traccia della "storia" dei prodotti
SVN (*subversion*, http://subversion.tigris.org/)
Git (http://git-scm.com/)
Configurazione (*build*) per integrare le parti del prodotto finale
Ant (http://ant.apache.org/)
Maven (http://maven.apache.org/)
Dipartimento di Informatica, Università di Pisa
21 / 35


IS
Amministrazione di progetto
Strumenti di lavoro – 2
Dal sito del corso ...
Amministrazione di progetto (parte 2)
Per approfondire #12: Uno strumento di gestione progetto
Per approfondire #13: Strumenti di collaborazione
Per approfondire #14: Strumenti di gestione del ciclo di vita del *software* (Ambiente Java, **Guida**)
Per approfondire #15: Strumenti di gestione del ciclo di vita del *software* [8MB] (Ambiente Microsoft)
Dipartimento di Informatica, Università di Pisa
22 / 35


IS
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


IS
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

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?**

Convenzioni sui nomi

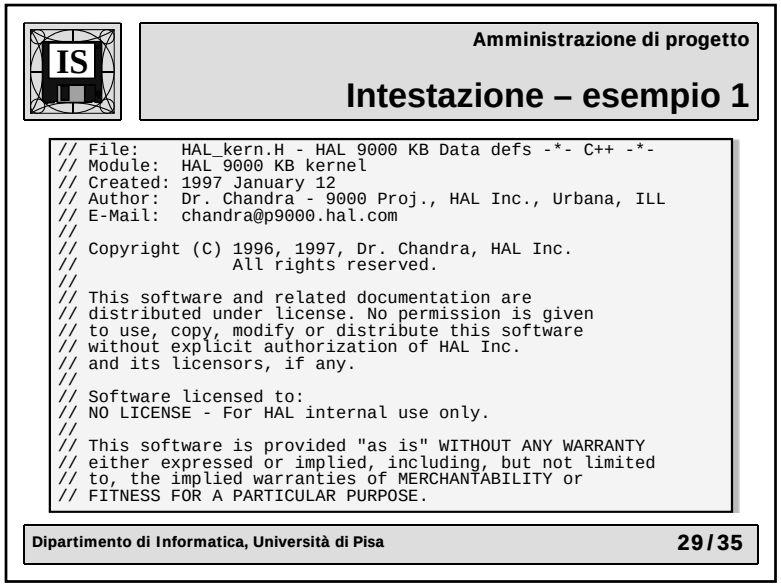
- **Nel codice**
  - **Tipi, costanti, variabili, funzioni, ...**
    - P.es., le norme **Javadoc** (vedi: http://java.sun.com/j2se/javadoc/)
- **Nel progetto**
  - **Strutturazione in moduli, *file*, *directory*, ...**
- **Aspetti pratici**
  - **Conflitti logici all'interno o all'esterno del codice**
  - **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)

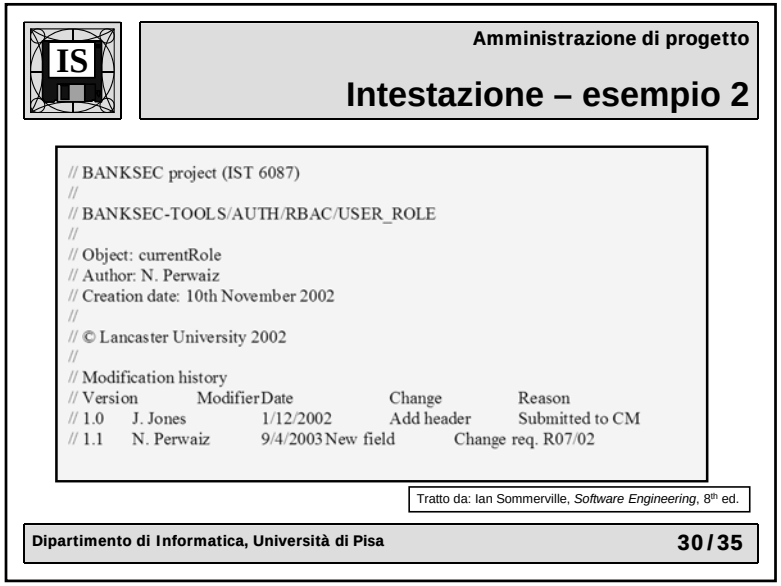
Indentazione del codice

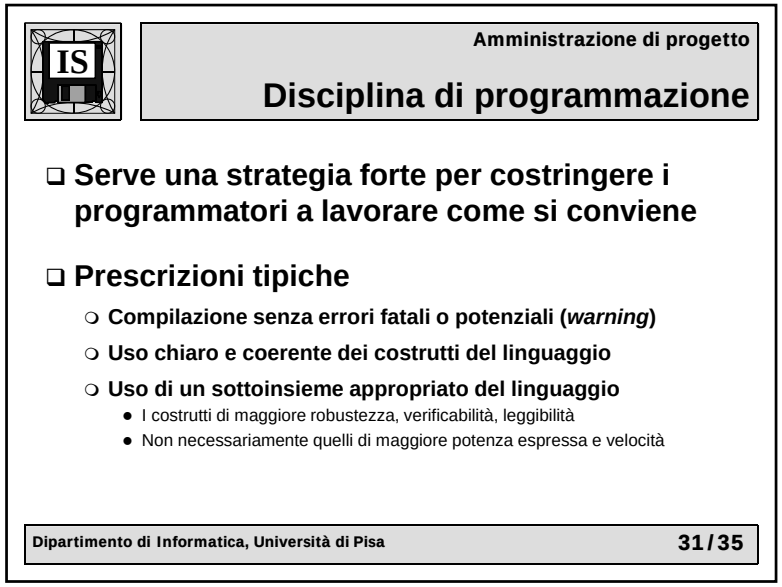
- **Obiettivi**
  - **Programmazione strutturata**
  - **Evidenziare visivamente la struttura di un programma**
- **Aspetti da non sottovalutare**
  - **Lunghezza delle linee**
  - **Ampiezza dell'indentazione**
  - **Posizione degli fine linea nei blocchi**
  - **Posizione degli fine linea nelle espressioni**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

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, versione**


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

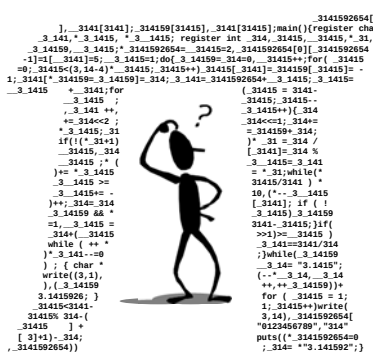

Amministrazione di progetto
Intestazione – esempio 2
// BANKSEC project (IST 6087)
//
// BANKSEC-TOOLS/AUTH/RBAC/USER_ROLE
//
// Object: currentRole
// Author: N. Perwaiz
// Creation date: 10th November 2002
//
// © Lancaster University 2002
//
// Modification history
// Version Modifier Date Change 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, 8th ed.
Dipartimento di Informatica, Università di Pisa
30 / 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
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
USE THE CODE, LUKE!
Dipartimento di Informatica, Università di Pisa
32 / 35

Norme di codifica

- **IOCCC Roemer.c**
  - **I**nternational
  - **O**bfuscated
  - **C**
  - **C**ode
  - **C**ontest



Notazione ungherese ☺

- `lpfnWndProc` **(un campo di struttura)**
  - `l` *long* **(32-bit integer)**
  - `p` *pointer* **[to a]**
  - `fn` *function* **[handling messages directed to]**
  - `Wnd` **[a]** *window*
  - `Proc` *procedure*
  - **Un vettore di puntatori a descrittori di finestre, indicizzato sul numero di finestre**

- **Un'idea di Charles Simonyi @ Microsoft**

Riferimenti

- **V. Ambriola, G.A. Cignoni, "Laboratorio di progettazione", Jackson Libri, 1996**
- **"Programming in C++ – Rules and Recommendations", Ellemtel TSL (Svezia), 1992**
- **C. Simonyi, M. Heller, "The Hungarian Revolution", Byte, agosto 1991**
- **F. Lanubile et al., "Collaboration Tools for Global Software Engineering", IEEE Software, 27:2, 2010, 52-55**
- **J. Portillo Rodriguez et al., "Technologies and Tools for Distributed Teams", IEEE Software, 27:5, 2010, 10-14**