Introduction to Configuration Management

Leonardo Gresta Paulino Murta
leomurta@ic.uff.br
From the beginning...

- 50s
  - CM used in the production of military air planes, weapons, and space rockets
- 60s e 70s
  - Born of SCM (S = Software)
  - Still focus on military applications
- 80s e 90s
  - Civil uses of CM (MIL → EIA, IEEE, ISO, etc.)
  - First international standards on CM
  - Assimilation by non-military organizations
Some standards

- **AFSCM 375-1 (1962)**
  - Only consider hardware
- **MIL Std 483 (1971)**
  - Starts to consider software
- **DOD Std 2167A (1985)**
  - Consolidates previous standards
  - Widely used in governmental contracts
  - Deprecated in 2000
  - Substitutes MIL Std 973
  - Compatible to ISO 12207
Some standards

  - Discusses about the conception of CM plan
  - Main reference for non-military organizations

- **IEEE Std 1042 (1987)**
  - Guide to apply IEEE Std 828
  - Very complete and detailed standard about CM
  - Reaffirmed in 1993
  - Deprecated in 2000

  - Based on MIL Std 973
  - Satisfies ISO 9000
Some standards

- **ISO 12207**
  - CM process based on
    - IEEE Std 828
    - IEEE Std 1042
    - ISO 10007

- **CMMI**
  - Process area at level 2

- **MPS.BR**
  - Process at level F
Some books

• “Software Configuration Management Handbook”
  – Alexis Leon (2004)

• “Software Configuration Management”

• “Configuration Management Principles and Practice”
  – Anne Hass (2002)

• “Software Configuration Management Patterns: Effective Teamwork, Practical Integration”
Some websites

- http://www.cmcrossroads.com
  - CM community on the web
- http://www.bradapp.net/acme
  - Branching patterns
- http://dblp.uni-trier.de/db/conf/scm
  - Main CM conference
  - From 1988 to 2005
- https://dblp.uni-trier.de/db/conf/msr/
  - Conference on repository mining
  - Since 2004
- https://dblp.org/db/conf/csmr/index.html
  - Software analytics (SANER = CSMR + WCRE)
  - Since 1997
- https://dblp.uni-trier.de/db/conf/icsm/
  - Software Maintenance
  - Since 1993
Anti-definitions

- CM is not (only) version control
- CM is not content management (take a look on product data management)
- CM is not backup
- CM does not avoid changes
- CM is not only for large and complex systems
- CM is not only for large and distributed teams
Definitions

• Wayne Babich
  – “Configuration management is the art of identifying, organizing, and controlling modifications to the software being built by a programming team. The goal is to maximize productivity by minimizing mistakes.”

• Susan Dart
  – “Software CM is a discipline for controlling the evolution of software systems.”

• Steve McConnell
  – “Configuration management is the practice of handling changes systematically so that a system can maintain its integrity over time.”

• Walter Tichy
  – “Software configuration management (SCM) is the discipline of controlling the evolution of complex software systems.”
Definitions

- Jacky Estublier
  - “It is the discipline that enable us to keep evolving software products under control, and thus contributes to satisfying quality and delay constraints.”
- IEEE Std 610
  - “A discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report changes processing and implementation status, and verify compliance with specified requirements.”
Definitions

- ISO 10007
  - “Technical and organizational activities comprising: configuration identification; configuration control; configuration status accounting; configuration auditing.”

- ISO 12207
  - “The Configuration Management Process is a process of applying administrative and technical procedures throughout the software life cycle to: identify, define, and baseline software items in a system; control modifications and releases of the items; record and report the status of the items and modification requests; ensure the completeness, consistency, and correctness of the items; and control storage, handling, and delivery of the items.”
Definitions

• CMMI
  – “The purpose of Configuration Management (CM) is to establish and maintain the integrity of work products using configuration identification, configuration control, configuration status accounting, and configuration audits.”

• MPS.BR
  – “The purpose of Configuration Management process is establish and maintains the integrity of all work products of a process or project and make them available to all stakeholders”
CM focus

- Control the evolution of
  - Project work products
  - Process work products

- Source of information to other processes
  - Historical data for measurement and analysis
  - Bug reports for causal analysis
Perspectives

Processes (management perspective):
- Identification
- Control
- Status accounting
- Auditing
- Release

Systems (development perspective):
- Issue tracking
- Version control
- Build management
Challenges

• Complex projects in terms of size, sophistication, and technologies
• Huge teams, usually distributed geographically
• Execution requirements for different hardware and software platforms
• Support for different languages and cultures
• Different flavors to manage cost/benefit demands (e.g., desktop, standard, professional, and enterprise)
Challenges

• Increasing speed in communication among clients about defects in the product
• Need to reduce the development time to fix defects to keep the company reputation and comply to Service Level Agreement (SLA) contracts

• Necessity to deal with the inevitable chaos related to the creative activity of software development, using methods and tools to maximize productivity and minimize defects
Problems due to the absence of CM

- Source-code loss
- Libraries fail unexpectedly
- Difficult to understand what happened with a program or its parts
- Difficult to establish when changes were made, why, and by whom
Problems due to the absence of CM

- Requirements already elicited suddenly disappear
- Requirements already implemented disappear from the source-code
- Impossibility to match the executable program version with its source-code version
Evolution of CM techniques

- Small systems, developed by only one person, are less dependent of CM
- When the system size and development team grow, communication problems arise

[Leon, 2000] Communications breakdown problem
Evolution of CM techniques

- Software artifacts are shared by multiple developers
- Changes made by other developers are not well communicated, leading to version incompatibilities

[Leon, 2000] Shared data problem
Evolution of CM techniques

- A naïve solution consists on creating multiple copies of the shared artifacts
- This may lead to lack of control over copies
- Leading to rework to implement the same requirements and fix the same defects in different copies

[Leon, 2000] Multiple maintenance problem
Evolution of CM techniques

• Shared repository as a solution
• Back to the shared data problem
• Work loss due to lack of concurrency control over the shared repository

[Leon, 2000] Simultaneous update problem
Evolution of CM techniques

• Concurrency control over shared repository
• Methods and tools to support the whole CM process
• Productivity gain due to the enforcement of discipline together with reduction of rework
• Side effect may be the increasing bureaucracy required by the CM process
CM Benefits

• Increasing organizational memory
• Development more dependent to the process (less dependency to individuals)
• Control over the software development and its evolution
• Traceability within different abstraction levels of the same entity (requirement, analysis, design, code, and executable program)
• Documentation about the software evolution
Introduction to Configuration Management

Leonardo Gresta Paulino Murta
leomurta@ic.uff.br