



**DECSAI**

**Departamento de Ciencias de la Computación e I.A.**

Universidad de Granada



SCM

Fernando Berzal, [berzal@acm.org](mailto:berzal@acm.org)

SCM

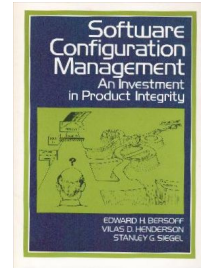


- SCM [Software Configuration Management]
- Terminología
- Control de cambios
- Control de versiones
  - Check-in & check-out
  - Herramientas de control de versiones
- El proceso de gestión de la configuración
- El plan de gestión de la configuración





No matter where you are in the system life cycle, the system will change, and the desire to change it will persist throughout the life cycle.



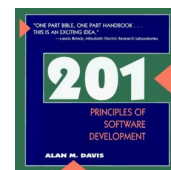
Edward H. Bersoff, Vilas D. Henderson & Stanley G. Siegel:  
**Software Configuration Management:  
An Investment in Product Integrity**

1980



## **Software Configuration Management**

SCM no es sólo disponer de una herramienta que registre quién hizo qué cambio al código o a la documentación y cuándo lo hizo.



SCM también incluye la creación de convenciones sobre el código, políticas y procedimientos que aseguren que todos los "stakeholders" relevantes están involucrados en la realización de cambios sobre el software:

**Conjunto de actividades realizadas para gestionar los cambios a lo largo del ciclo de vida del software.**



# SCM



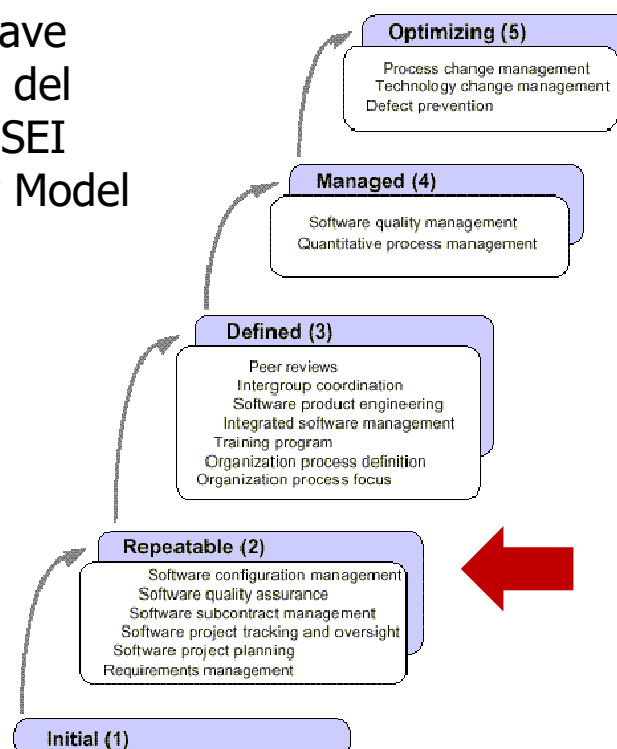
- Cómo informar de un problema con el software.
- Cómo solicitar la incorporación de un nuevo requisito.
- Cómo informar de los cambios solicitados y pedir sus opiniones a los "stakeholders" involucrados.
- Cómo priorizar y planificar las solicitudes de cambios.
- Cómo mantener bajo control las distintas versiones de los productos (i.e. impedir que puedan cambiarse sin seguir los procedimientos apropiados).



# SCM

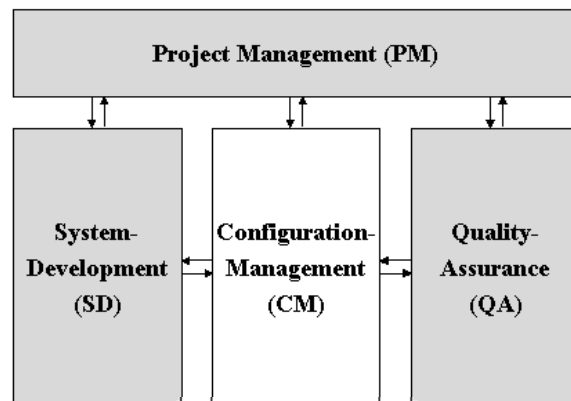


Área de proceso clave [KPA] en el nivel 2 del modelo SW-CMM [SEI Capability Maturity Model for Software].



## Propósito

Mantener la integridad de los productos de un proyecto de desarrollo de software a lo largo de su ciclo de vida.



Aborda 4 tipos de problemas habituales:

- **Comunicación** (el número de vías de comunicación aumenta cuadráticamente con el tamaño del equipo).
- **Cambios** (problemas cuando un programador no es consciente de los últimos cambios realizados sobre un ítem compartido).
- **Actualizaciones** (problemas asociados a la actualización simultánea de ítems compartidos).
- **Mantenimiento** (problemas asociados al mantenimiento de múltiples versiones).





## Beneficios

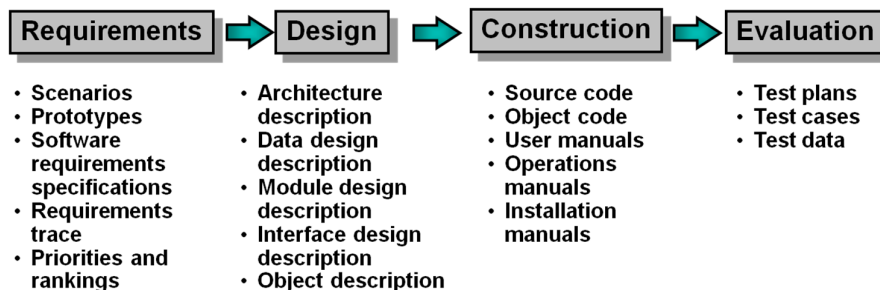
- Se evitan problemas asociados a la realización de cambios y se facilitan las tareas de mantenimiento.
- Aumento de productividad (disminución de trabajo repetido [rework]).
- Automatización de tareas, p.ej. Seguimiento de defectos [defect logging & tracking].



## Terminología



**SCI** [Software Configuration Item]:  
 Unidad fundamental en un sistema de gestión de la configuración, p.ej. ficheros de código, planes, modelos, especificaciones, documentos, procesos...



SDP (Software Development Plan), tasks, QA plan, SCM plan, tools, change requests, defect reports, procedures

Configuration items are identified at the start of a project

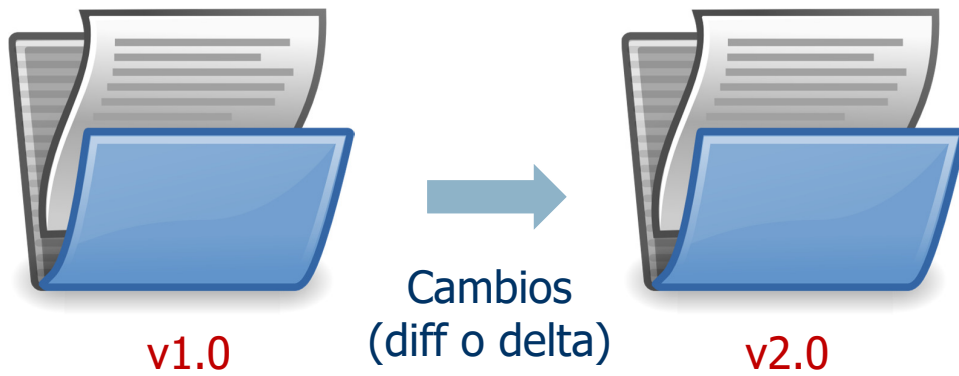


# Terminología



## Versión

Versión específica de un SCI



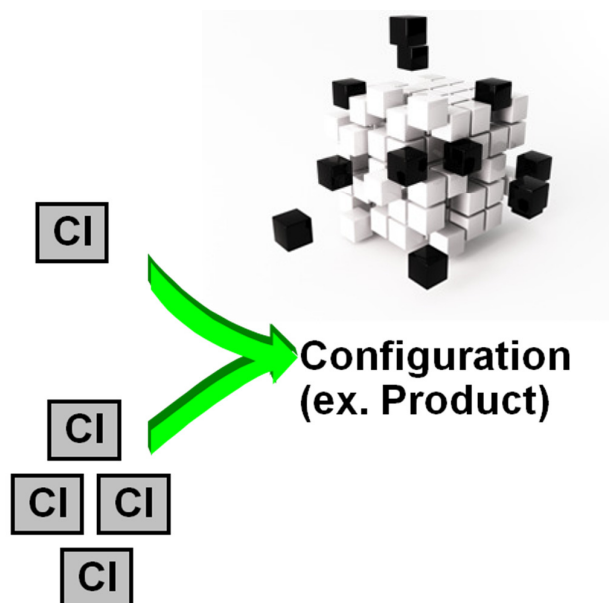
10

# Terminología



## Configuración

Colección de versiones concretas de ítems SCI.



11

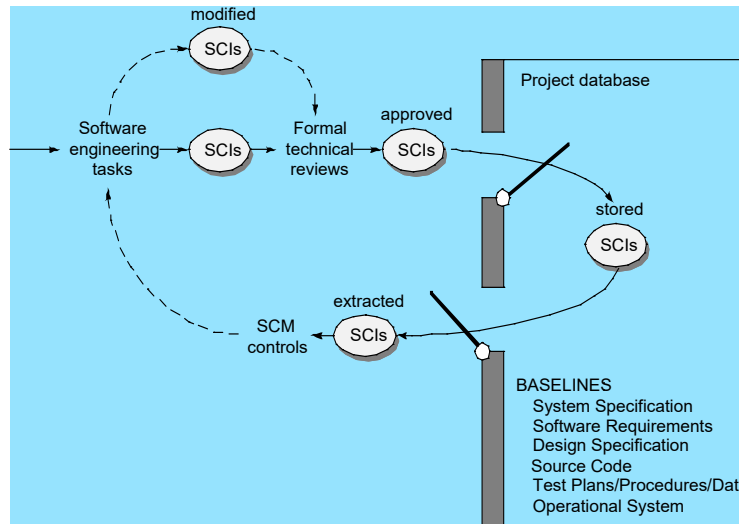


# Terminología



## Baseline ["línea base"]:

Versión aprobada de un producto en un momento determinado que sirve de base para definir cambios.



Roger S. Pressman: "Software Engineering: A Practitioner's Approach" 7/e



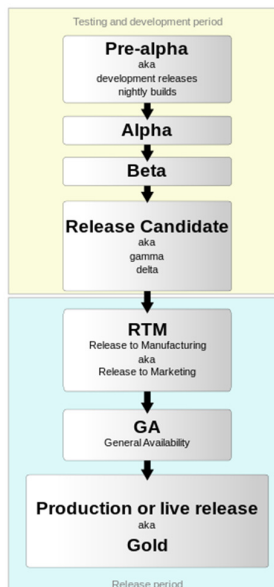
# Terminología



## Release

Distribución formal de una versión aprobada.

p.ej. al equipo de QA, versión alfa/beta, versión para un cliente determinado, RTM [release to manufacturing]...



[https://en.wikipedia.org/wiki/Software\\_release\\_life\\_cycle](https://en.wikipedia.org/wiki/Software_release_life_cycle)



# Terminología



## Branch ["rama"]

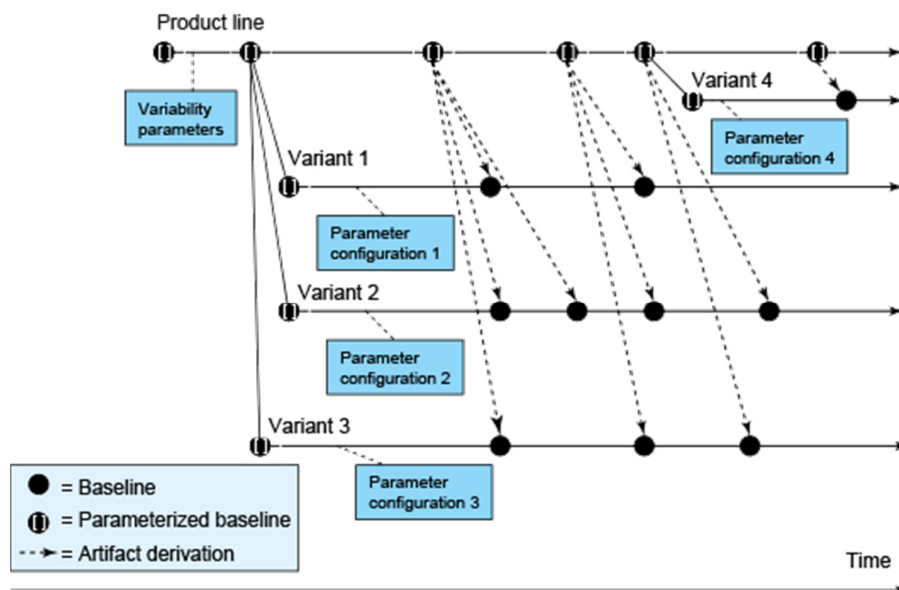
Duplicación de un conjunto de SCI's de forma que las modificaciones puedan realizarse en paralelo.



- **"trunk"** (línea de desarrollo que no es una rama), a.k.a. baseline, mainline o master.
- Conjunto de ramas de un proyecto: árboles de versiones [trees], streams o "codelines"



# Terminología



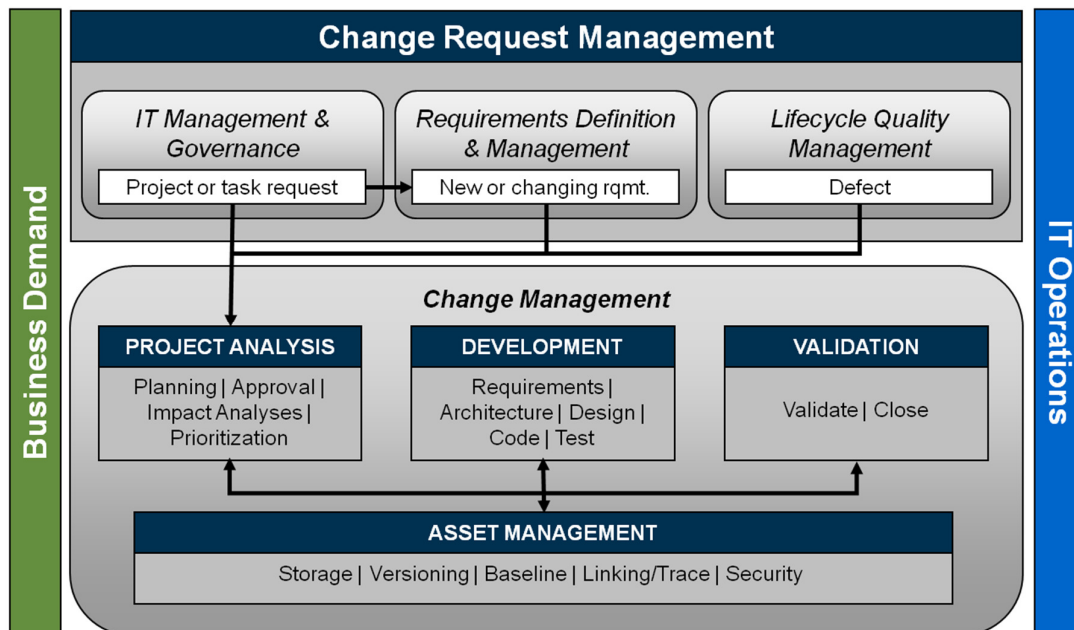
Eran Gery & Joanne L. Scouler:  
"Strategic reuse and product line engineering."  
IBM developerWorks, October 2014

<http://www.ibm.com/developerworks/rational/library/14/strategic-reuse/strategic-reuse-pdf.pdf>





# Control de cambios



Micro Focus StarTeam



# Control de cambios



“Software is never done”...

- CR [Change Requests]
- Change “triage”
- CCB [Change/Configuration Control Board]  
p.ej. informal en proyectos pequeños  
7 niveles en FAA NAS [National Airspace System]



# Control de cambios



Tipos de cambios:

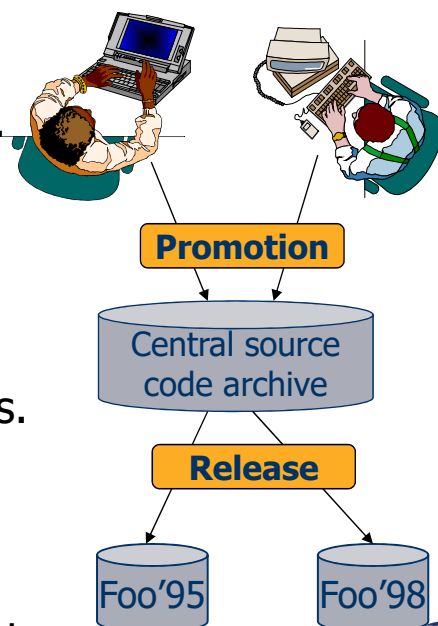
- **Promotion**  
(interno a la organización de desarrollo)
- **Release**  
(visible fuera de la organización de desarrollo)



# Control de cambios



- **Programmer's directory**  
[IEEE Std.: "dynamic library"],  
bajo el control del programador.
- **Master directory**  
[IEEE Std.: "controlled library"]  
contiene la línea base actual,  
sólo con los cambios autorizados.
- **Software repository**  
[IEEE Std.: "static library"],  
archivo de las versiones publicadas.



# Control de versiones



El almacén (a.k.a. repositorio) SCM proporciona el conjunto de mecanismos y estructuras de datos que permiten gestionar los cambios de forma efectiva.

## Funciones

- Integridad de las distintas versiones.
- Gestión de dependencias entre elementos SCI.
- Compartir datos entre los miembros del equipo.
- Integración con las herramientas de desarrollo.
- Aplicación de mecanismos de gestión de cambios (p.ej. aprobación en revisiones técnicas formales).
- Trazabilidad (p.ej. Requisitos).
- Realización de auditorias



# Control de versiones



Un sistema de control de versiones implementa o se integra directamente con:

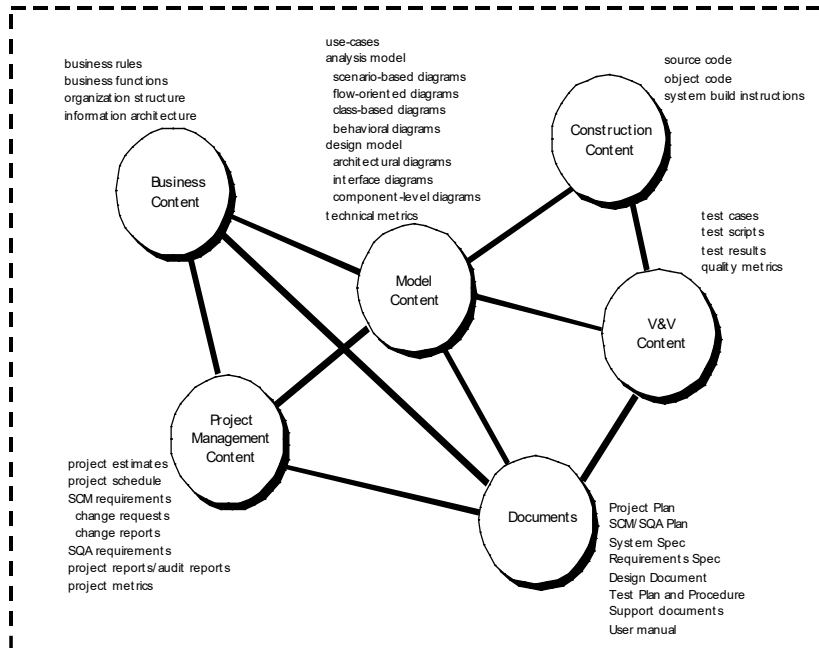
- Una **base de datos** (o repositorio) del proyecto.
- Una herramienta de **gestión de versiones** que maneja todas las versiones de cada SCI.
- Una herramienta de **compilación [make/build]** que permite automatizar la creación de versiones específicas.
- Una herramienta de **seguimiento [issue/bug tracking]** que permite registrar y controlar todos los hechos o problemas asociados a los distintos SCI.



# Control de versiones



## Contenido de la base de datos del proyecto

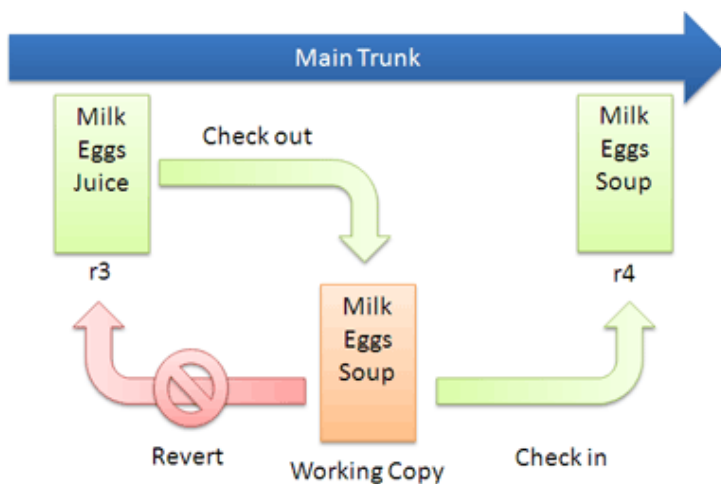


Roger S. Pressman: "Software Engineering: A Practitioner's Approach" 7/e

# Control de versiones



## Check-in & check-out



<http://betterexplained.com/articles/a-visual-guide-to-version-control/>



# Control de versiones



## Lista de comprobación para la realización de check-ins

### Check-in Meeting Checklist

Describe the changes in the check-in

Bug numbers fixed?

Files and databases changed?

Developer's confidence level

Developer's assessment of risks in the code / what most concerned about?

Known issues remaining?

Any areas of code more brittle than others?

Is the code documented?

Is the code testable? Has the developer tested it?

Who code-reviewed prior to check-in? (right choice of reviewer?)

Possible performance issues?

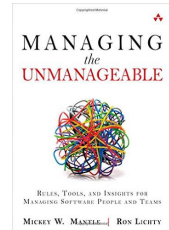
Any hidden secrets to how it really works?

Projected impacts on other programmers' work?

Requests/notes to QA?

Notes to Documentation Team?

Notes to Build Team?



# Control de versiones



What makes a good developer a great developer is willingness to take risks.

You can measure a developer's willingness to take risks by the frequency they check into source control.

Source control is their safety net.

- Ike Ellis: "Continuous Integration", Silicon Valley Code Camp 2013





# Control de versiones



## Herramientas de control de versiones

Sistemas locales (usuarios únicos)

- **SCCS** [Source Code Control System],  
Mark Rochkind, Bell Labs, 1972
- **RCS** [Revision Control System],  
Walter F. Tichy, Purdue University, 1982
- **PVCS** [Polytron Version Control System]  
Serena Software, 1985



# Control de versiones



## Herramientas de control de versiones

Sistemas cliente/servidor

- **CVS** [Concurrent Versioning System],  
Dick Grune, VU University Amsterdam, 1986
- **Subversion** a.k.a. **SVN**  
CollabNet, 2000-2004 (proyecto Apache desde 2010)
- Herramientas comerciales:  
Rational ClearCase (1992), Perforce (1995), StarTeam (1995), Surround SCM (2002), AccuRev SCM (2002), Microsoft Team Foundation Server (2005)...



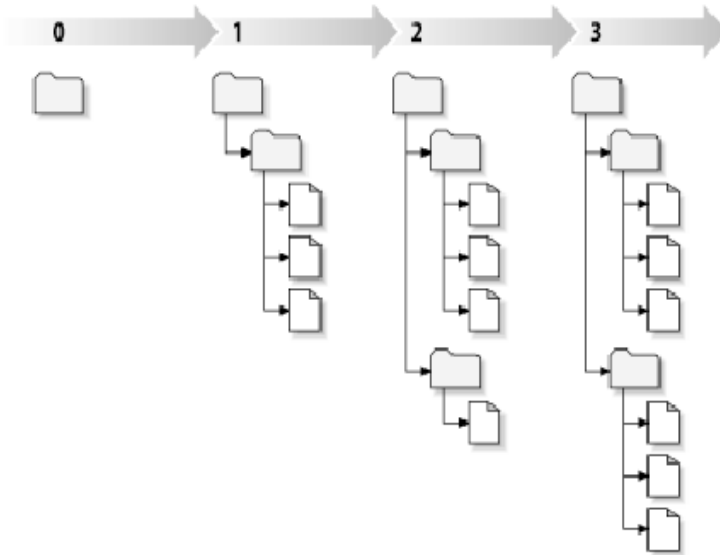


# Control de versiones



## Subversion

Global revision number

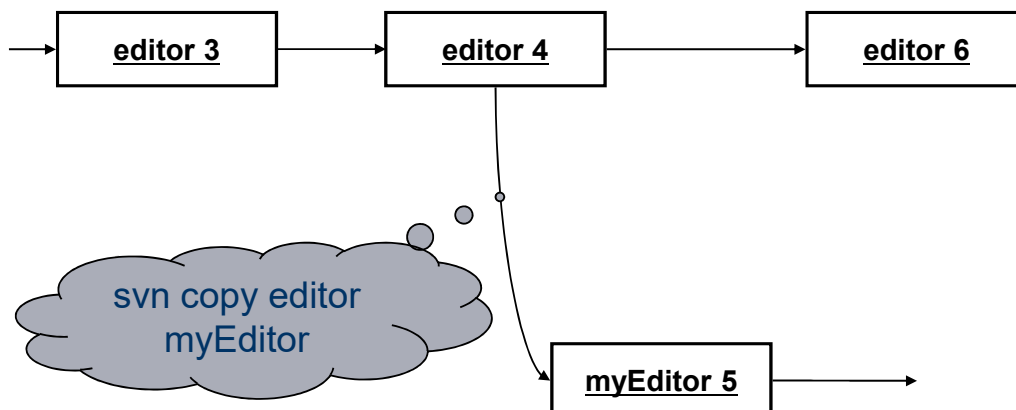


# Control de versiones



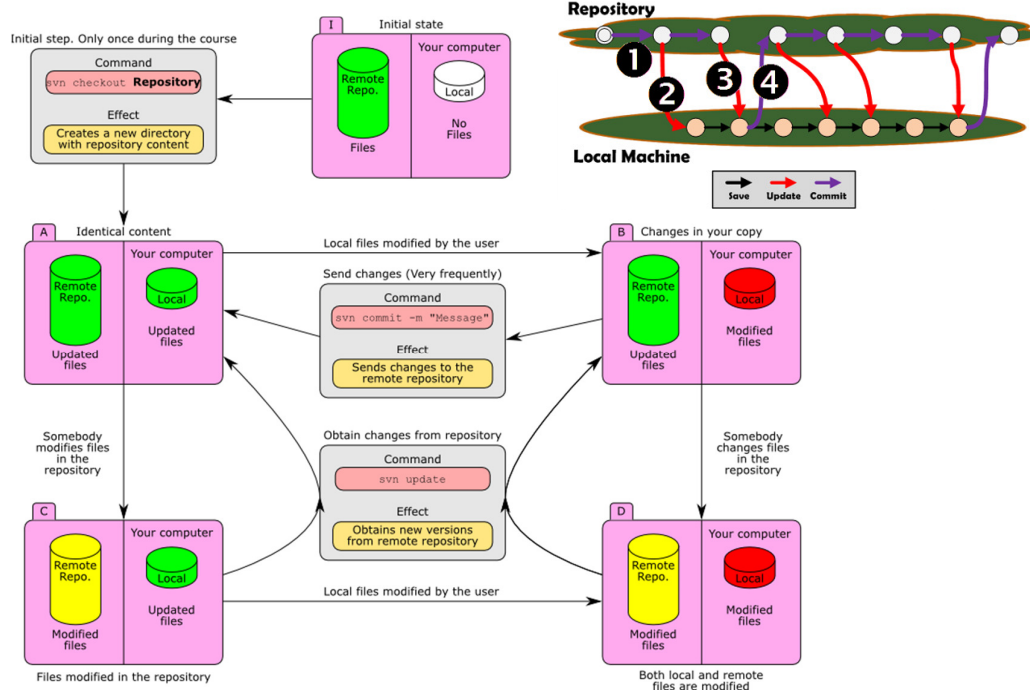
## Subversion

Branching (copia)





## Subversion



## Herramientas de control de versiones

### Sistemas distribuidos

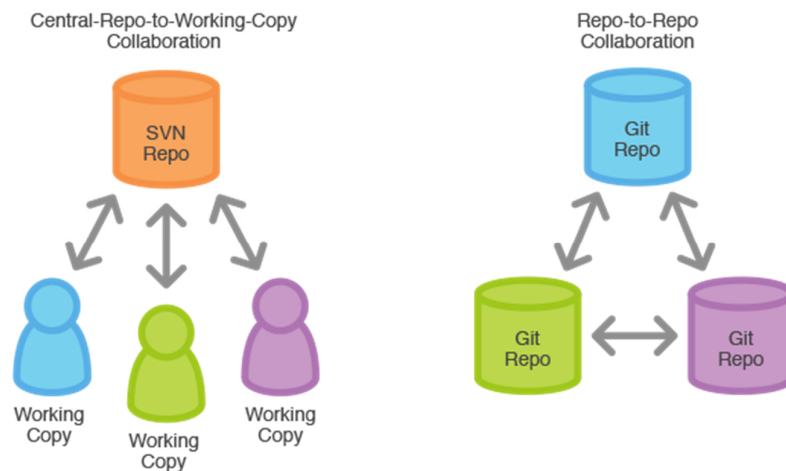
- **GNU arch**  
Thomas Lord, 2001
- **Git**  
Linus Torvalds et al., 2005 (Linux kernel)
- **Mercurial a.k.a. hg**  
Matt Mackall, 2005 (Python)
- Herramientas comerciales: Code Co-op (1997), BitKeeper (1998), Plastic SCM (2006)...



# Control de versiones



## De Subversion (C/S) a Git (P2P)



<https://www.atlassian.com/git/tutorials/>

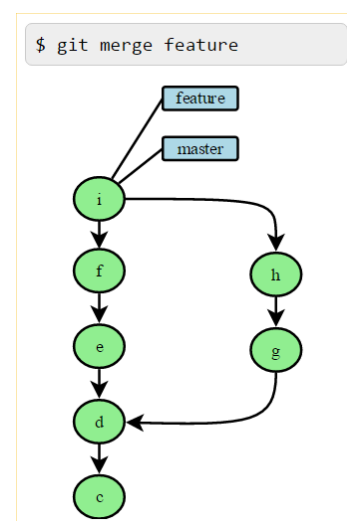
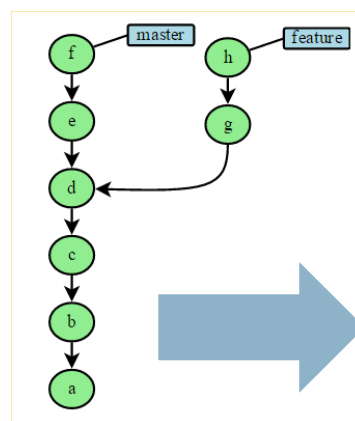
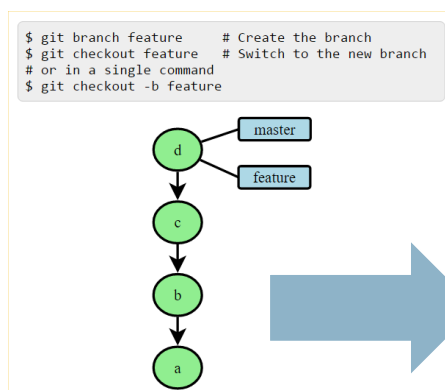


# Control de versiones



## Git

### Branching & merging



<https://illustrated-git.readthedocs.org/en/latest/>

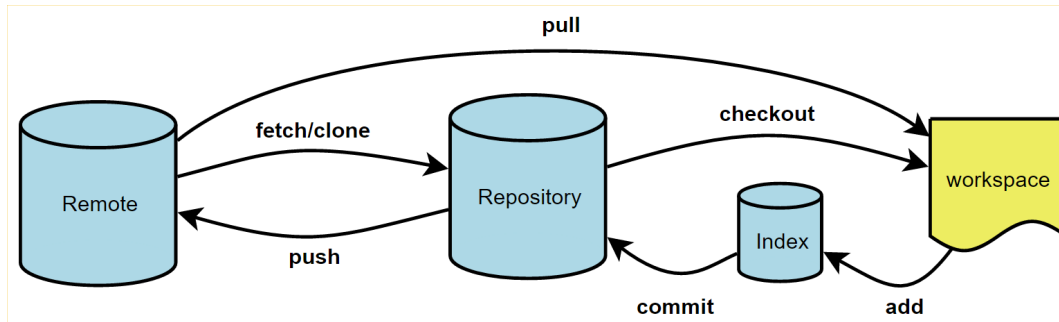


# Control de versiones



## Git

### Remote repositories



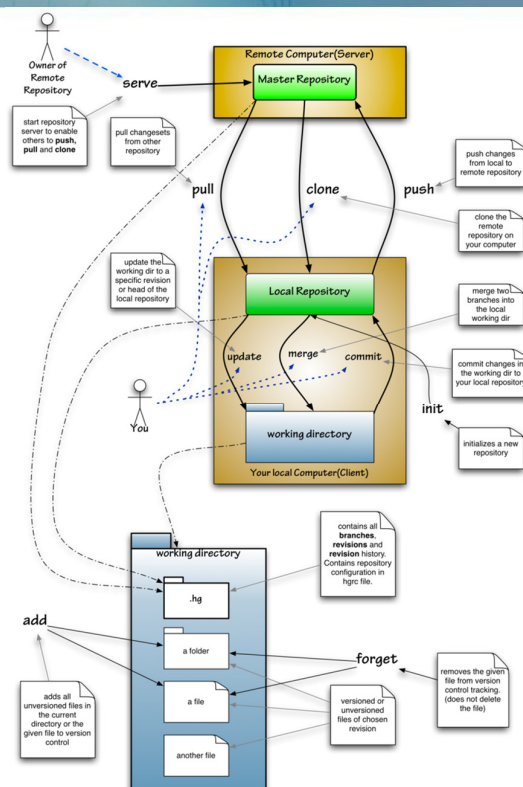
<https://illustrated-git.readthedocs.org/en/latest/>



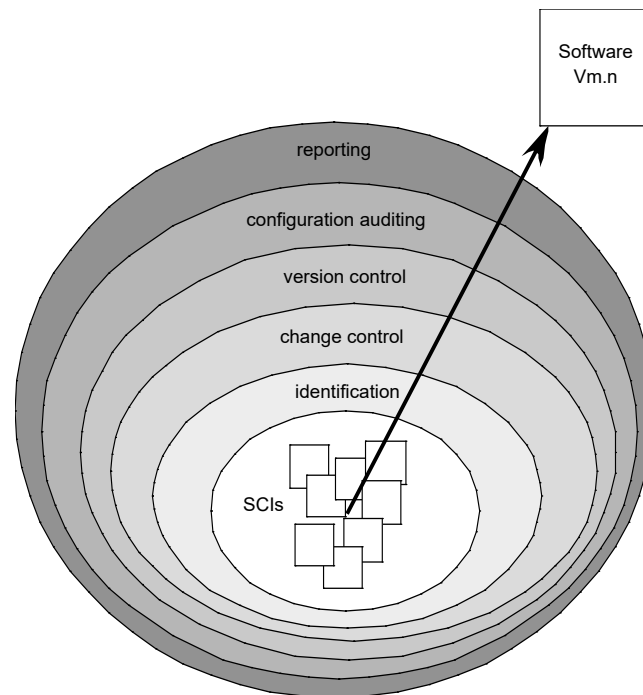
# Control de versiones



## Mercurial



# El proceso SCM



Roger S. Pressman: "Software Engineering: A Practitioner's Approach" 7/e

# Plan SCM



Generalmente, las decisiones acerca de cómo resolver las cuestiones de SCM se recopilan en un documento:

## SCMP

### Software Configuration Management Plan

Este documento se elabora al comienzo del proyecto y, normalmente, se aprueba más o menos cuando se aprueba la especificación de requisitos del sistema.





## Estándar IEEE 828-2005 Software Configuration Management Plan

### 1. Introduction

(describes the plan's purpose, scope of application, key terms, and references)

### 2. SCM management (WHO?)

(identifies the responsibilities and authorities for the planned SCM activities)

### 3. SCM activities (WHAT?)

(identifies all activities to be performed)



## Estándar IEEE 828-2005 Software Configuration Management Plan

### 4. SCM schedule (WHEN?)

(establishes required coordination of SCM activities with other activities in the project)

### 5. SCM resources (HOW?)

(identifies tools and physical and human resources required for the execution of the Plan)

### 6. SCM plan maintenance

(identifies how the Plan will be kept current while in effect)







## SCMP Section 3: SCM activities

- 3.1 Configuration identification  
(identification, naming conventions & acquisition)
- 3.2. Configuration control  
(request, evaluation, approval & implementation)
- 3.3. Configuration status accounting  
(metrics to be tracked and reported)
- 3.4. Configuration evaluation and reviews  
(audit on CI before release: objective, schedule, procedures, participants, approval criteria...)



## SCMP Section 3: SCM activities

- 3.5. Interface control  
(coordination of changes to CIs with changes to interfacing items outside of the scope of the Plan)
- 3.6. Subcontractor/vendor control  
(incorporation of items developed outside the project environment into the project CIs)
- 3.7. Release Management and Delivery  
(description of the formal control of build, release and delivery of software products)



# Plan SCM



## SCMP following the IEEE 828-2005 standard

Define configuration items

Define promote / release policies

Define activities and responsibilities

Set up configuration management system



# Comentarios finales



"The success of SCM is not primarily determined by tool sophistication, but by the attitude and cooperation of the people involved."

-- **Alexis Leon:**

"A Guide to Software Configuration Management"

EJEMPLO

Check-ins limitados al final del proyecto:  
Sólo para actualizaciones críticas

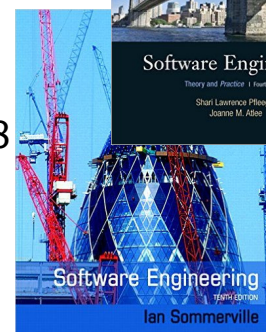
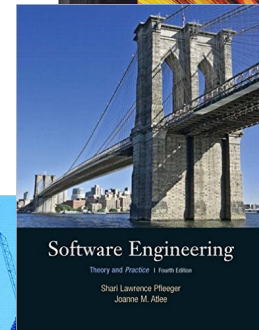
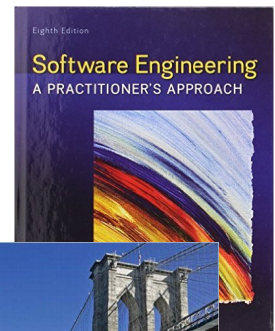


# Bibliografía



## Libros de texto

- Roger S. Pressman:  
**Software Engineering: A Practitioner's Approach**  
McGraw-Hill, 8th edition, 2014. ISBN 0078022126
- Shari Lawrence Pfleeger & Hoanne M. Atlee:  
**Software Engineering: Theory and Practice**  
Prentice Hall, 4th edition, 2009. ISBN 0136061699
- Ian Sommerville:  
**Software Engineering**  
Pearson, 10th edition, 2015. ISBN 0133943038

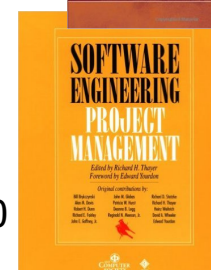
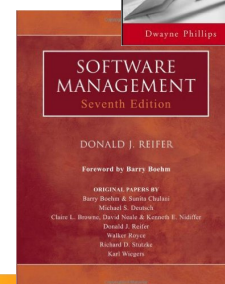
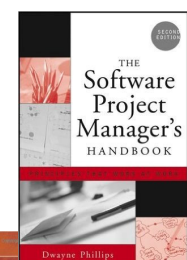


# Bibliografía



## Lecturas recomendadas

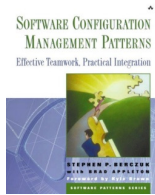
- Dwayne Phillips:  
**The Software Project Manager's Handbook: Principles That Work at Work**  
Wiley / IEEE Computer Society, 2nd edition, 2004  
ISBN 0471674206
- Donald J. Reifer (editor):  
**Software Management**  
Wiley / IEEE Computer Society, 7th edition, 2006  
ISBN 0471775622
- Richard H. Thayer (editor):  
**Software Engineering Project Management**  
Wiley / IEEE Computer Society, 2nd edition, 2000  
ISBN 0818680008



# Bibliografía complementaria



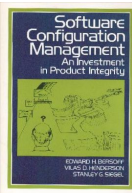
## SCM



Stephen P. Berczuk & Brad Appleton:  
**Software Configuration Management Patterns:  
Effective Teamwork, Practical Integration**  
Addison-Wesley Professional, 2002. ISBN 0201741172



Alexis Leon:  
**A Guide to Software Configuration Management**  
Artech House, 2000. ISBN 1580530729



Edward H. Bersoff, Vilas D. Henderson & Stanley G. Siegel:  
**Software Configuration Management:  
An Investment in Product Integrity**  
Prentice-Hall, 1980. ISBN 0138217696



46

# Bibliografía complementaria



## Estándares

- **IEEE 828-2012**  
IEEE Standard for Configuration Management  
in Systems and Software Engineering  
DOI: [10.1109/IEEESTD.2012.6170935](https://doi.org/10.1109/IEEESTD.2012.6170935)  
(Software Configuration Management Plans)
- **ANSI/IEEE 1042-1987**  
IEEE Guide to Software Configuration Management  
DOI: [10.1109/IEEESTD.1988.94582](https://doi.org/10.1109/IEEESTD.1988.94582)  
(practices compatible with ANSI/IEEE Std 828-1983)



47

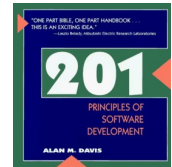
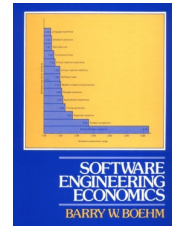
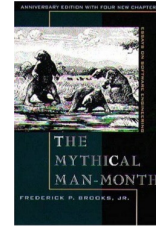


# Bibliografía complementaria



## Clásicos

- Frederick P. Brooks, Jr.:  
**The Mythical Man-Month: Essays on Software Engineering**  
Addison-Wesley, 1995. ISBN 0201835959
- Alan M. Davis:  
**201 Principles of Software Development**  
McGraw-Hill, 1995. ISBN 0070158401
- Barry W. Boehm:  
**Software Engineering Economics**  
Prentice-Hall PTR, 1991. ISBN 0138221227
- **Manager's Handbook for Software Development**  
NASA Software Engineering Laboratory, SEL-84-101, rev.1, 1990.
- **Software Engineering Laboratory (SEL) Relationships, Models, and Management Rules**  
NASA Software Engineering Laboratory, SEL-91-001, 1991.

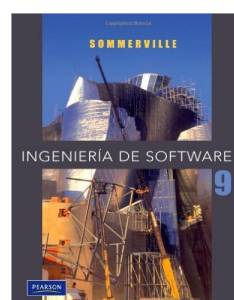


# Bibliografía



## Bibliografía en castellano

- Roger S. Pressman:  
**Ingeniería de Software: Un enfoque práctico**  
McGraw-Hill, 7ª edición, 2010. ISBN 6071503140
- Ian Sommerville:  
**Ingeniería de Software**  
Pearson, 9ª edición, 2012. ISBN 6073206038





Buscar información sobre herramientas concretas que faciliten la realización de las siguientes tareas:

- Control de versiones
- Compilación [daily builds]
- Integración continua
- Pruebas automatizadas (de unidad y de aceptación)
- Seguimiento de defectos [bug tracking]

Para cada categoría, identifique al menos 5 herramientas que se utilicen actualmente.

