

Công nghệ Phần mềm (Software Engineering)



CHƯƠNG 5

Quản lý cấu hình phần mềm

Mục tiêu của bài học

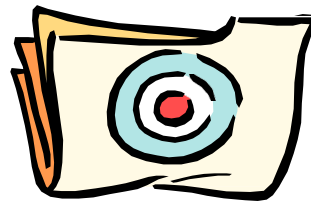
Sinh viên sẽ được trang bị các kiến thức sau:

- Các khái niệm liên quan tới Quản lý cấu hình phần mềm
- Giới thiệu về quy trình Quản lý cấu hình phần mềm
- Một số hoạt động quan trọng như: quản lý phiên bản, quản lý thay đổi

Nội dung

1. Khái niệm quản lý cấu hình phần mềm
2. Quy trình quản lý cấu hình phần mềm
3. Quản lý phiên bản
4. Quản lý thay đổi

1.1 Đặt vấn đề



Quá trình phát triển phần mềm

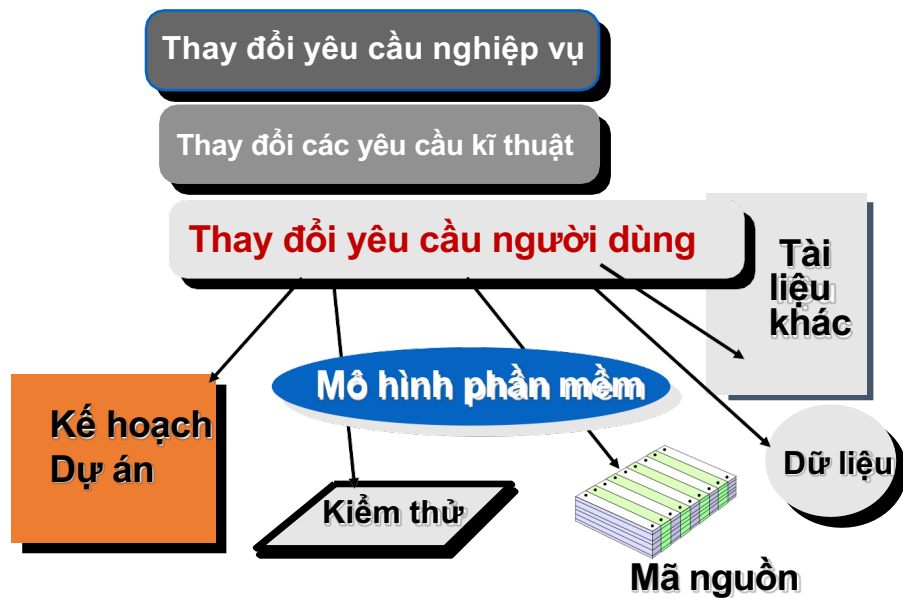
- Lý tưởng:
 - Phần mềm được phát triển từ các yêu cầu ổn định
 - (do việc hướng đến mục tiêu cố định luôn dễ dàng hơn mục tiêu bị thay đổi)
- Thực tế:
 - Các yêu cầu ổn định luôn không tồn tại cho hầu hết các hệ thống thực tế
- Do đó:
 - Một dự án phần mềm hiệu quả cần phải có chiến lược để **giải quyết vấn đề “THAY ĐỔI”**

Sự tiến hóa của phần mềm

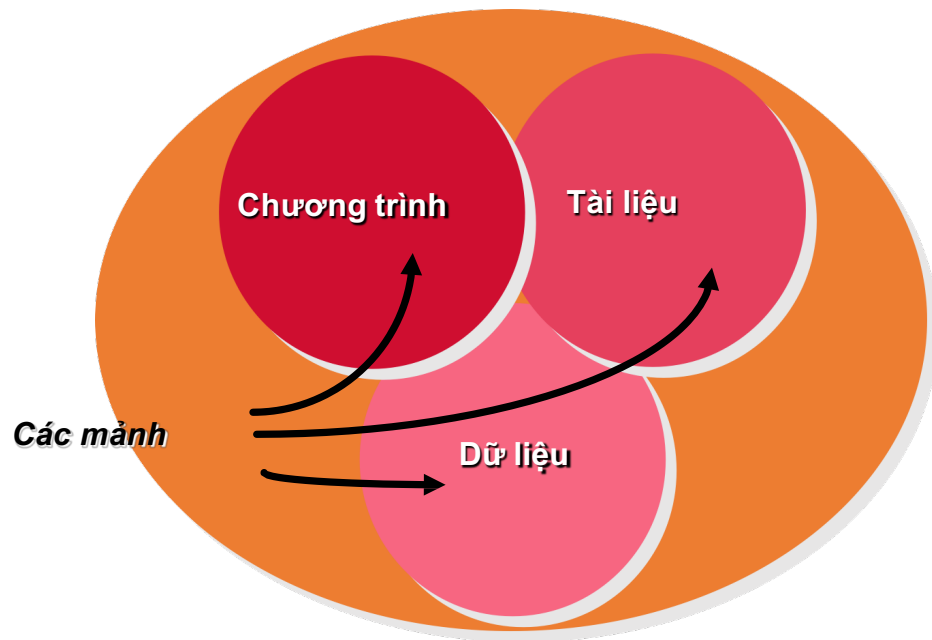
- Phần mềm được phát triển theo thời gian
 - Nhiều yếu tố khác nhau được tạo ra trong suốt thời gian của dự án
 - Có nhiều phiên bản khác nhau
 - Các nhóm làm việc song song để đưa ra sản phẩm cuối cùng
- Hệ thống có thể **thay đổi liên tục**



Những thay đổi là gì?



Cấu hình phần mềm



Vấn đề

- **Nhiều người** phải làm việc trên phần mềm đang thay đổi
 - **Nhiều phiên bản** của phần mềm phải được hỗ trợ:
 - Hệ thống đã phát hành
 - Hệ thống được cấu hình tùy chỉnh (các chức năng khác nhau)
 - Hệ thống đang được phát triển
 - Phần mềm phải chạy trên **các máy và hệ điều hành khác nhau**
- Do đó cần phải có sự quản lý và phối hợp với nhau
- Quản lý cấu hình phần mềm (Software Configuration Management (SCM))
 - Quản lý các hệ thống phần mềm đang phát triển
 - Kiểm soát chi phí liên quan đến việc thực hiện các thay đổi đối với hệ thống

Thay đổi và Kiểm soát

- Nếu những **thay đổi** không được **kiểm soát** - mọi thứ có thể và sẽ vượt khỏi tầm tay
- Vấn đề quản lý thay đổi thậm chí là cần thiết khi nhiều người cùng làm việc trong một dự án
- Nếu không có các chiến lược và cơ chế thích hợp để kiểm soát các thay đổi - người ta không bao giờ có thể khôi phục về bản sao cũ ổn định hơn của phần mềm
 - Do bởi **mọi thay đổi đều dẫn đến rủi ro**

Câu trả lời

- Sự thật:
 - Những thay đổi là không thể tránh khỏi
 - Các thay đổi cần được kiểm soát
 - Các thay đổi cần được quản lý
- Giải pháp
 - Quản lý cấu hình phần mềm
Software Configuration Management (SCM)



1.2 Quản lý cấu hình phần mềm

- Mô tả:
 - Quản lý cấu hình phần mềm - Software Configuration Management (SCM) bao gồm các nguyên tắc và kỹ thuật đánh giá và kiểm soát sự thay đổi đối với các sản phẩm phần mềm trong và sau quá trình kỹ thuật phần mềm.
- Các tiêu chuẩn (được ANSI phê duyệt)
 - IEEE 828: Software Configuration Management Plans
 - IEEE 1042: Guide to Software Configuration Management

1.2 Quản lý cấu hình phần mềm (2)

- Áp dụng một cách tiếp cận nghiêm ngặt để đảm bảo
 - Các chi tiết trong hệ thống phần mềm đều được xác định và theo dõi
 - Các thay đổi với các mục khác nhau được ghi lại và theo dõi
 - Tích hợp thích hợp tất cả các mô-đun khác nhau

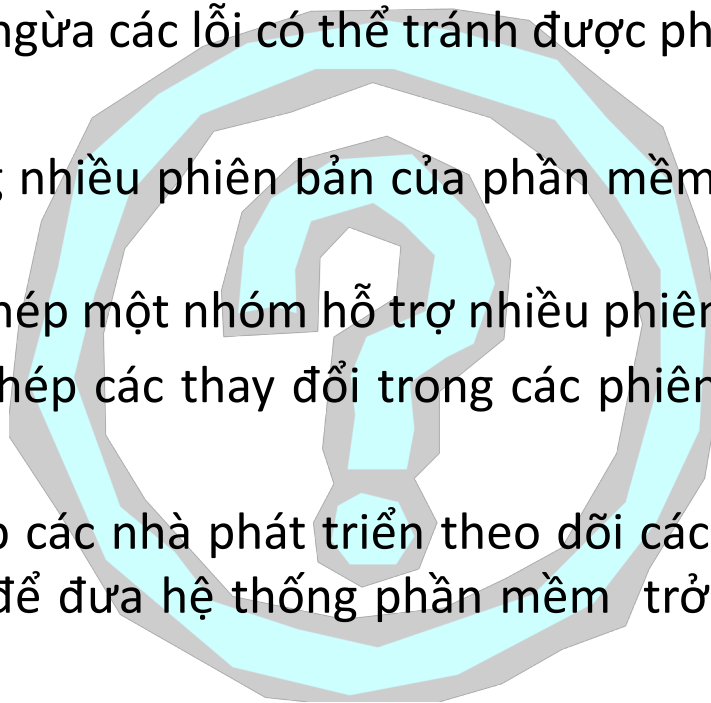
phát hiện - xác định chế độ kiểm soát và theo dõi - tích hợp

1.2 Quản lý cấu hình phần mềm (3)

- SCM có thể giúp xác định tác động của thay đổi cũng như kiểm soát sự phát triển
- Nó có thể theo dõi và kiểm soát các thay đổi trong tất cả các khía cạnh của phát triển phần mềm
 - Yêu cầu
 - : tài liệu đặc tả y/c PM
 - Thiết kế
 - : < trúc - khung: sơ đồ
 - chi tiết: đặc tả chi tiết
 - Mã hóa
 - : mã nguồn
 - Kiểm thử
 - : test-case, test plan, test report
 - Làm tài liệu
 - : dự gó, HDSĐ, HD triển khai

Sự cần thiết của SCM...

- Khi phần mềm phát triển - nhiều tài nguyên hệ thống thay đổi
 - SCM ngăn ngừa các lỗi có thể tránh được phát sinh từ các thay đổi xung đột
- Thông thường nhiều phiên bản của phần mềm được phát hành và cần đến sự hỗ trợ
 - SCM cho phép một nhóm hỗ trợ nhiều phiên bản.
 - SCM cho phép các thay đổi trong các phiên bản tuần tự được truyền bá
- SCM cho phép các nhà phát triển theo dõi các thay đổi và khôi phục bất kỳ thay đổi nào để đưa hệ thống phần mềm trở lại trạng thái an toàn đã biết gần đây nhất



1.3 Các khái niệm trong SCM

- Mục cấu hình (Configuration Item - CI) / Mục cấu hình phần mềm (Software Configuration Item - SCI)
- Đường cơ sở (Baseline)
- Kho lưu trữ SCM (SCM Repository)
- Thư mục SCM (SCM Directory)
- Phiên bản, bản sửa đổi và bản phát hành (Version, Revision and Release)

Mục cấu hình (CI)

“Tập hợp phần cứng, phần mềm hoặc cả hai, được chỉ định để quản lý cấu hình và được coi như một thực thể duy nhất trong quy trình quản lý cấu hình”.

“An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process.”

Mục cấu hình (CI) (2)

- Các mục cấu hình phần mềm không chỉ là các đoạn mã chương trình mà là tất cả các loại tài liệu cho sự phát triển phần mềm, ví dụ:
 - các tệp mã
 - trình điều khiển cho các trường hợp kiểm thử
 - tài liệu phân tích hoặc thiết kế
 - tài liệu hướng dẫn người dùng
 - cấu hình hệ thống (ví dụ: phiên bản trình biên dịch được sử dụng)
- ❖ Trong một số hệ thống, không chỉ phần mềm mà còn tồn tại các mục cấu hình phần cứng (CPU, tần số tốc độ bus)!

Tìm kiếm các mục cấu hình

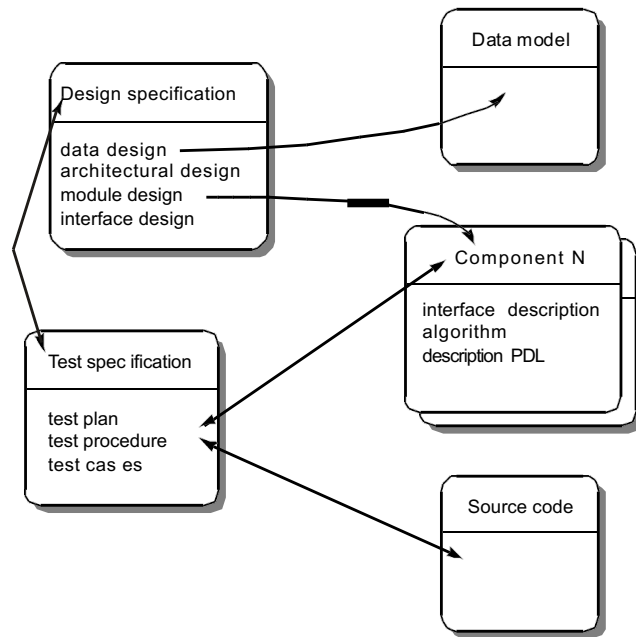
- Các dự án lớn thường tạo ra hàng nghìn thực thể (tệp, tài liệu, ...) phải được *xác định duy nhất*.
- Nhưng không phải tất cả các thực thể đều cần được định cấu hình. Vấn đề:
 - Cái gì: Lựa chọn CI (Nên quản lý những gì?)
 - Khi nào: Khi nào bạn bắt đầu đặt một thực thể dưới sự kiểm soát cấu hình?
- Bắt đầu quá sớm dần đến quá sự “áp đặt”
- Bắt đầu quá muộn dần đến hỗn loạn

Tìm kiếm các mục cấu hình (2)

- Một số thực thể này phải được duy trì trong suốt thời gian tồn tại của phần mềm. Điều này cũng bao gồm giai đoạn khi phần mềm không còn được phát triển nhưng vẫn được sử dụng bởi khách hàng mong đợi sự hỗ trợ thích hợp trong nhiều năm.
- **Một lược đồ đặt tên** thực thể nên được xác định để định danh cho các tài liệu có tên liên quan.
- Lựa chọn các mục cấu hình phù hợp là một kỹ năng cần thực hành
 - Rất giống với mô hình đối tượng
 - Sử dụng các kỹ thuật tương tự như mô hình hóa đối tượng để tìm các CI

Tìm kiếm các mục cấu hình (3)

- Các đối tượng cấu hình phần mềm



Đường cơ sở (Baseline)

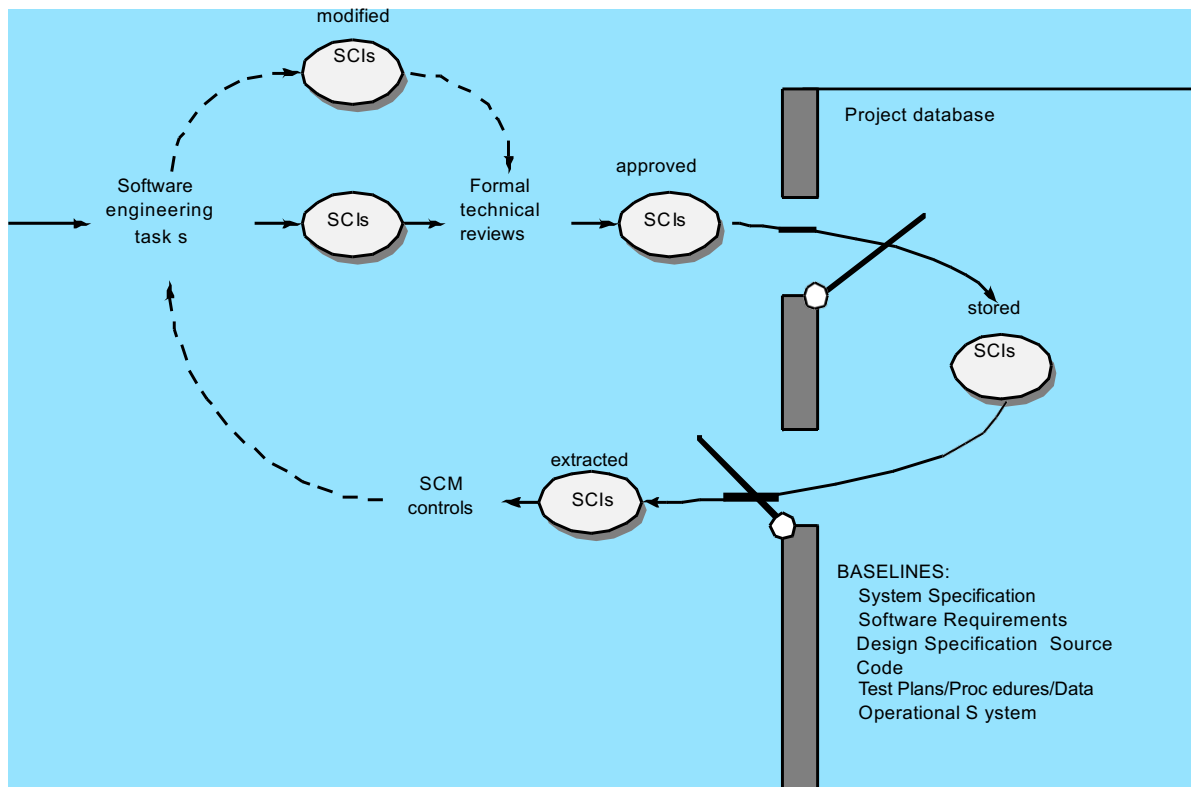
- Định nghĩa:
 - Chuẩn IEEE (IEEE Std. No. 610.12-1990) định nghĩa đường cơ sở (baseline) như sau:
 - Đặc tả kỹ thuật hoặc sản phẩm đã được xem xét và thống nhất chính thức, sau đó được dùng như là một cơ sở để tiếp tục phát triển, và có thể thay đổi chỉ thông qua thủ tục kiểm soát thay đổi chính thức.
 - Một baseline là một mốc quan trọng trong sự phát triển của phần mềm được đánh dấu bằng việc cung cấp một hoặc nhiều mục cấu hình phần mềm và sự chấp thuận của các **SCIs - software configuration items** thu được thông qua đánh giá kỹ thuật chính thức.

Đường cơ sở (Baseline) (2)

- Baseline có thể được định nghĩa ở bất kỳ mức chi tiết nào
- Ví dụ:
 - Baseline A: API của một chương trình được xác định hoàn toàn; phần thân của các phương thức trống.
 - Baseline B: Tất cả các phương thức truy cập dữ liệu được thực hiện và kiểm thử; lập trình GUI có thể bắt đầu.
 - Baseline C: GUI được triển khai, giai đoạn thử nghiệm có thể bắt đầu.

Đường cơ sở (Baseline) (3)

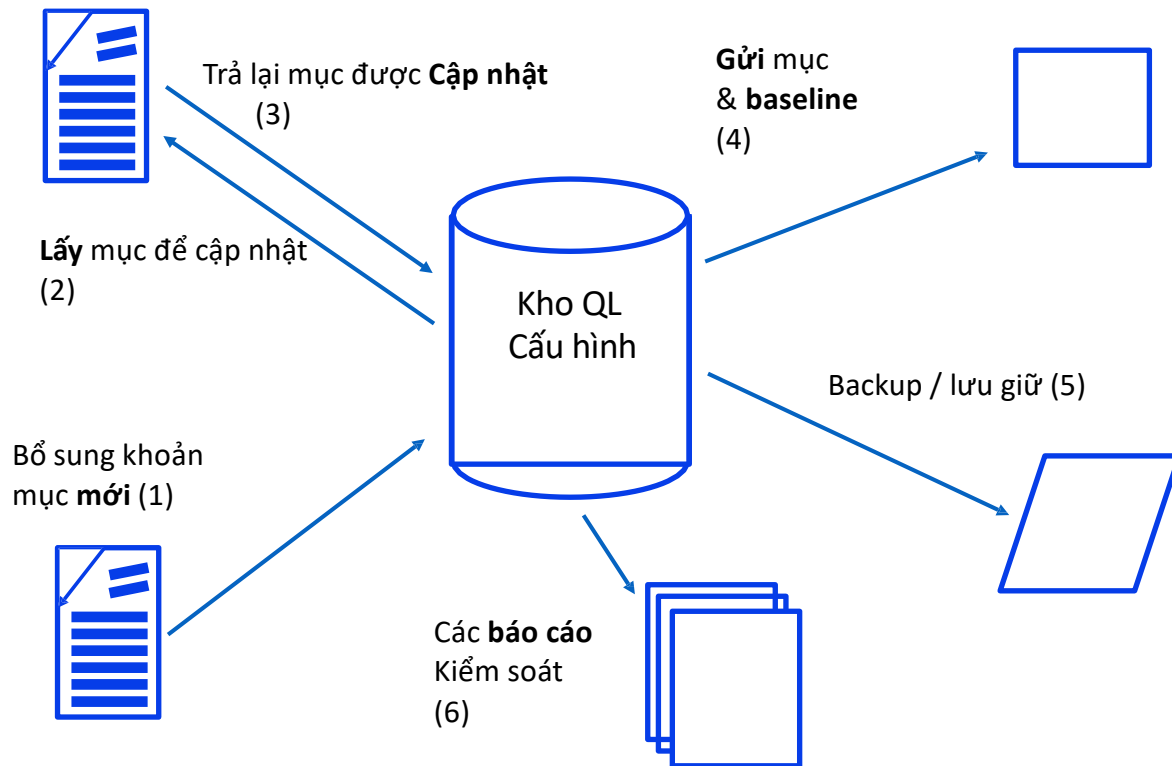
- Các SCI đã xác định đường cơ sở và cơ sở dữ liệu dự án:



Kho lưu trữ SCM (SCM Repository)

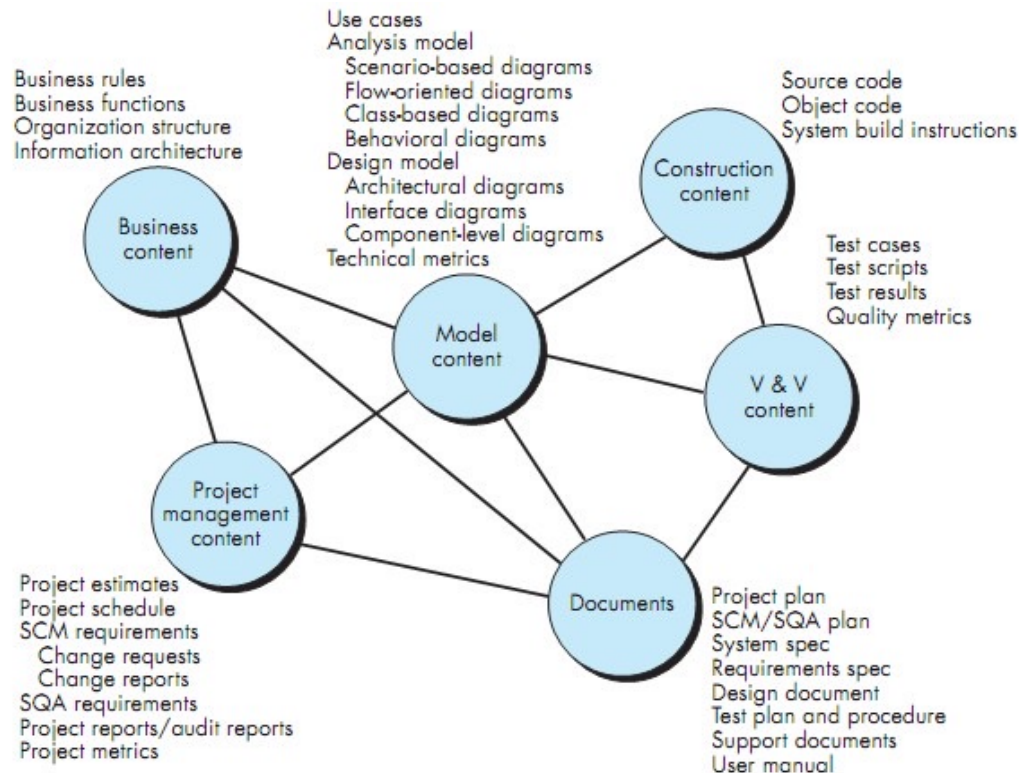
- Kho lưu trữ SCM là một tập các cơ chế hoạt động và cấu trúc dữ liệu cho phép một nhóm phát triển phần mềm có thể quản lý thay đổi, phát triển, bảo trì phần mềm một cách hiệu quả.
- Một kho lưu trữ (repository) dữ liệu có các chức năng sau đây:
 - Toàn vẹn dữ liệu
 - Chia sẻ thông tin
 - Tích hợp công cụ
 - Tích hợp dữ liệu
 - Thực thi phương pháp luận
 - Tiêu chuẩn hoá tài liệu

Kho lưu trữ SCM (SCM Repository) (2)



Kho lưu trữ SCM (SCM Repository) (3)

- Nội dung repository:



Kho lưu trữ SCM (SCM Repository) (4)

- Các đặc trưng của repository:
 - Phiên bản
 - Lưu trữ tất cả các phiên bản để cho phép quản lý các phiên bản đã đóng gói của sản phẩm và cho phép nhà phát triển có thể dùng các phiên bản này để phát triển, bảo trì.
 - Theo dõi sự phụ thuộc và quản lý thay đổi
 - Các kho quản lý lưu trữ mối quan hệ giữa các yếu tố dữ liệu được lưu trữ trong đó.
 - Yêu cầu tìm kiếm
 - Cung cấp để có thể theo dõi tất cả các bản thiết kế và các thành phần xây dựng và các sản phẩm mà kết quả từ một đặc tả cụ thể
 - Quản lý cấu hình
 - Lưu giữ tất cả những cấu hình cụ thể đại diện cho sự quan trọng và các sản phẩm đã đóng gói . Quản lý phiên bản các phiên bản cần thiết, và quản lý liên kết theo dõi phụ thuộc lẫn nhau.
 - Thông tin tác giả
 - Thiết lập đầy đủ các thông tin về thời gian bắt đầu và kết thúc, người thực hiện theo từng phiên bản.

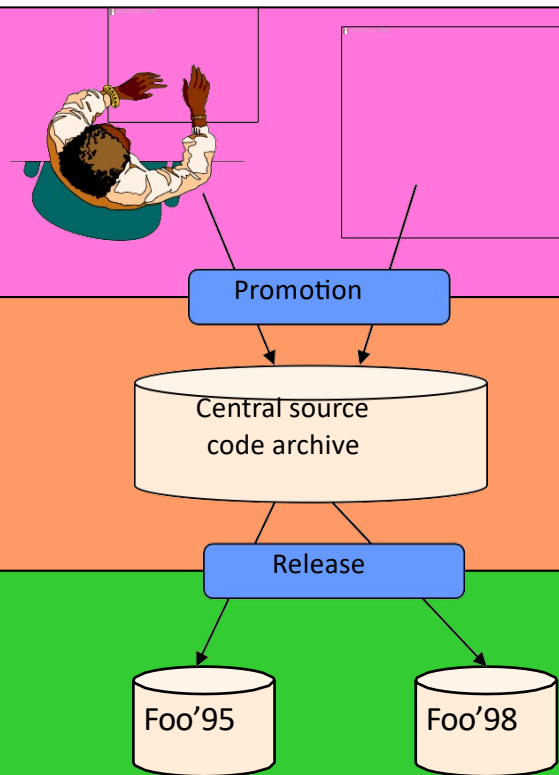


Thư mục SCM (SCM Directory)

- Programmer's Directory (IEEE: Dynamic Library)
 - Thư viện để chứa các thực thể phần mềm mới được tạo hoặc sửa đổi. Không gian làm việc của lập trình viên chỉ do lập trình viên kiểm soát.
- Master Directory (IEEE: Controlled Library)
 - Quản lý (các) baseline và kiểm soát các thay đổi được thực hiện đối với chúng. Mục nhập được kiểm soát, thường sau khi được xác minh. Các thay đổi phải được cho phép.
- Software Repository (IEEE: Static Library)
 - Lưu trữ cho các baseline khác nhau được phát hành để sử dụng chung. Các bản sao của các baseline này có thể được cung cấp cho các tổ chức yêu cầu.

Thư mục SCM (SCM Directory) (2)

- Programmer's Directory
 - (IEEE Std: "Dynamic Library")
 - Completely under control of one programmer.
- Master Directory
 - (IEEE Std: "Controlled Library")
 - Central directory of all promotions.
- Software Repository
 - (IEEE Std: "Static Library")
 - Externally released baselines.



Version vs. Revision vs. Release

- Phiên bản (Version): *thay đổi lớn* *1.0*
 - Một bản phát hành *ban đầu* hoặc tái phát hành một mục cấu hình được liên kết với một bản biên dịch hoặc biên dịch lại *hoàn chỉnh* của mục đó. Các phiên bản khác nhau có chức năng khác nhau.
- Bản sửa đổi (Revision): *thay đổi nhỏ* *1.0.156*
 - *Thay đổi* thành phiên bản chỉ sửa các lỗi trong thiết kế / mã, nhưng không ảnh hưởng đến chức năng đã được lập thành tài liệu. *thường số phiên lẻ*
- Bản phát hành (Release):
 - Việc *phân phối chính thức* phiên bản đã được phê duyệt.

* Các thành phần của 1 hệ thống quản lý cấu hình

- Các y tố cấu phần
- Các y tố xử lý (tiến trình) : quy trình
- Các y tố x dụng
- Các y tố con nợ

Các thành phần của một hệ thống quản lý cấu hình

- *Các yếu tố Cấu phần*—Một tập công cụ trong một hệ thống quản lý tập tin (ví dụ như dữ liệu) cho phép truy cập và quản lý mỗi mục cấu hình phần mềm.
- *Các yếu tố Xử lý*—Một tập các thủ tục và nhiệm vụ xác định một phương pháp hiệu quả quản lý sự thay đổi (và các hoạt động liên quan) cho tất cả các cử tri liên quan về quản lý, kỹ thuật và sử dụng phần mềm máy tính.
- *Các yếu tố Xây dựng* —Một tập hợp các công cụ tự động hóa việc xây dựng các phần mềm bằng cách đảm bảo rằng các thiết lập thích hợp của các thành phần được phê duyệt (tức là, đúng phiên bản) đã được lắp ráp.
- *Các yếu tố Con người* —để thực thi SCM hiệu quả, nhóm nghiên cứu phần mềm sử dụng một tập hợp các công cụ và tính năng quá trình (bao gồm các yếu tố khác SCM).

Nội dung

1. Khái niệm quản lý cấu hình phần mềm
- 2. Quy trình quản lý cấu hình phần mềm**
3. Quản lý phiên bản
4. Quản lý thay đổi

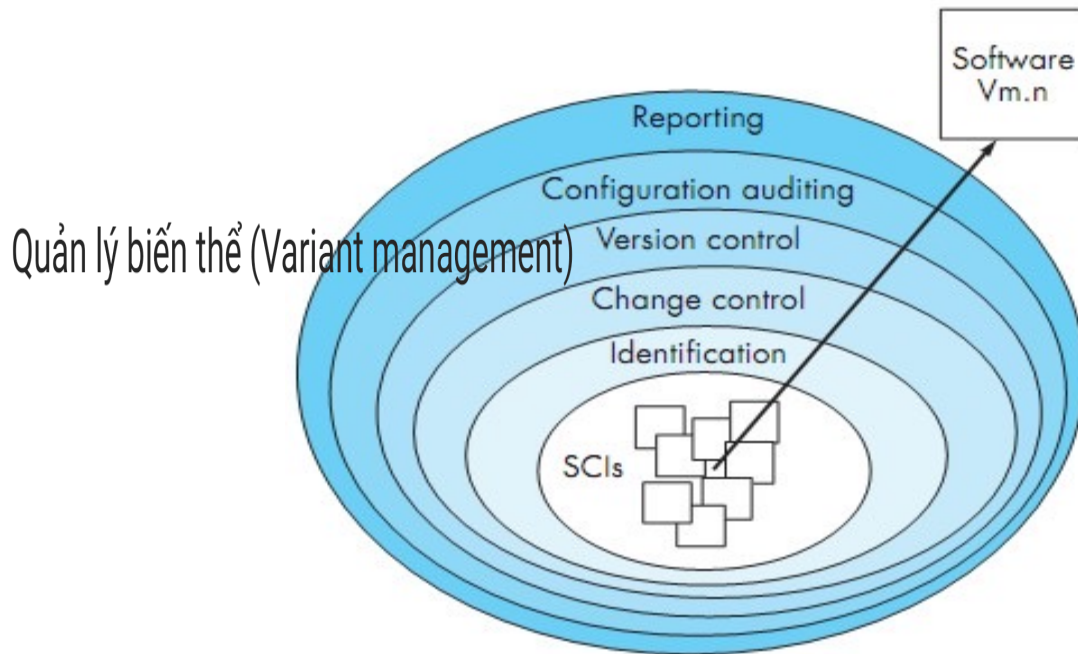
2.1 Quy trình SCM

Giải quyết các câu hỏi sau đây ...

- Một nhóm phần mềm **xác định các yếu** tố rời rạc của cấu hình phần mềm như thế nào? *What*
- Một tổ chức quản lý nhiều phiên bản tồn tại của một chương trình (và tài liệu của nó) **theo** cách mà sẽ cho phép thay đổi một cách hiệu quả thế nào? *How*
- Một tổ chức **điều khiển những thay đổi** trước và sau khi phần mềm được đóng gói bàn giao cho khách hàng như thế nào? *When*
- **Ai có trách nhiệm phê duyệt** và xếp hạng những thay đổi? *Who*
- Làm thế nào chúng ta có thể **đảm bảo rằng những thay đổi** đã được thực hiện đúng cách? *How*
- Cơ chế nào được sử dụng để **đánh giá những thay đổi** được thực hiện bởi người khác? *Which*

2.1 Quy trình SCM (2)

- 5 nhiệm vụ của SCM: identification, version control, change control, configuration auditing, reporting



2.1 Quy trình SCM (3)

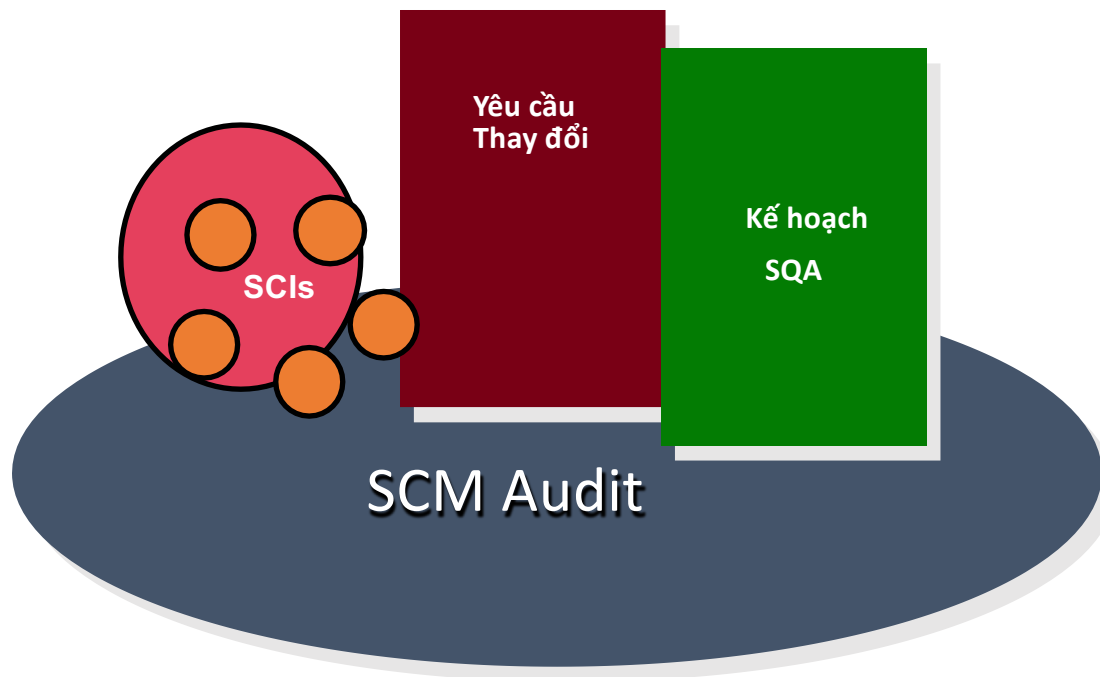
- Cung cấp một kho chứa an toàn đối với các kết quả bàn giao
- Cho phép việc kiểm soát và tiết lộ có nguyên tắc các kết quả bàn giao thông qua vòng đời của nó, với đầy đủ các dấu tích lịch sử, đảm bảo phiên bản đúng và cập nhật, đã được kiểm tra và phát hành
- Kiểm soát thay đổi của các kết quả bàn giao, đảm bảo các kết quả này được lưu theo đúng thứ tự
- Cung cấp việc lập báo cáo về hiện trạng của các kết quả bàn giao và những thay đổi của chúng

2.2 Các hoạt động trong SCM

- Nhận dạng mục cấu hình (Configuration item identification)
 - mô hình hóa hệ thống như một tập hợp các thành phần đang phát triển
- Quản lý tăng trưởng (Promotion management)
 - là việc tạo ra các phiên bản cho các nhà phát triển khác
- Quản lý phát hành (Release management)
 - là việc tạo ra các phiên bản cho khách hàng và người dùng
- Quản lý nhánh (Branch management)
 - là quản lý của sự phát triển đồng thời
- Quản lý biến thể (Variant management)
 - là việc quản lý các phiên bản dự định cùng tồn tại
- Quản lý thay đổi (Change management)
 - là việc xử lý, phê duyệt và theo dõi các yêu cầu thay đổi

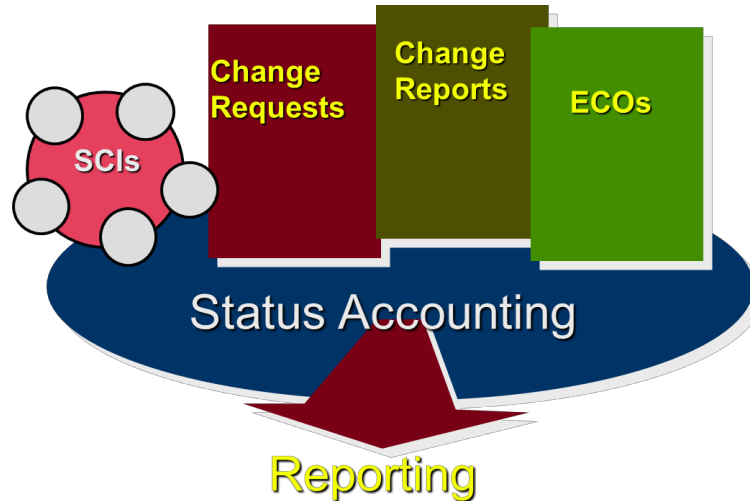
2.2 Các hoạt động trong SCM (2)

- Kiểm toán cấu hình (Configuration audit)
 - Đánh giá cấu hình phần mềm bổ sung cho việc xét duyệt kỹ thuật



2.2 Các hoạt động trong SCM (2)

- Báo cáo trạng thái (Status reporting)
 - Trả lời cho các câu hỏi: (1) What happened? (2) Who did it? (3) When did it happen? (4) What else will be affected?



2.3 Các vai trò trong SCM

- Người quản lý cấu hình
 - Chịu trách nhiệm xác định các mục cấu hình (configuration items – CI). Người quản lý cấu hình cũng có thể chịu trách nhiệm xác định các thủ tục để tạo các sự tăng trưởng và các bản phát hành.
- Thành viên ban kiểm soát thay đổi
 - Chịu trách nhiệm phê duyệt hoặc từ chối các yêu cầu thay đổi
- Lập trình viên
 - Tạo các thay đổi được kích hoạt bởi các yêu cầu. Nhà phát triển kiểm tra các thay đổi và giải quyết xung đột
- Kiểm soát viên
 - Chịu trách nhiệm về việc lựa chọn và đánh giá các thay đổi để phát hành và đảm bảo tính nhất quán và đầy đủ của bản phát hành này

Nội dung

1. Khái niệm quản lý cấu hình phần mềm
2. Quy trình quản lý cấu hình phần mềm

3. Quản lý phiên bản

4. Quản lý thay đổi

3.1 Khái niệm quản lý phiên bản

- **Quản lý phiên bản** (Version management hay kiểm soát phiên bản – Version control) là quá trình theo dõi các phiên bản khác nhau của các thành phần phần mềm hoặc các mục cấu hình và hệ thống mà các thành phần này được sử dụng.
- Nó cũng liên quan đến việc đảm bảo rằng các thay đổi do các nhà phát triển khác nhau thực hiện đối với các phiên bản này không ảnh hưởng lẫn nhau.
- Quản lý phiên bản gắn liền với **quản lý baselines**, chỉ định các phiên bản thành phần được bao gồm trong hệ thống cụ thể

3.1 Khái niệm quản lý phiên bản (2)

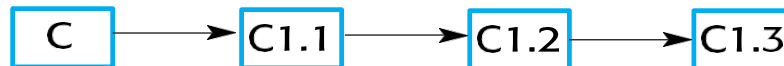
- Ví dụ: Codeline (A)



Codeline (B)



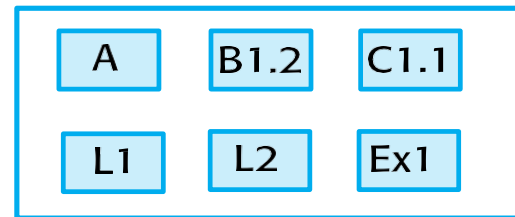
Codeline (C)



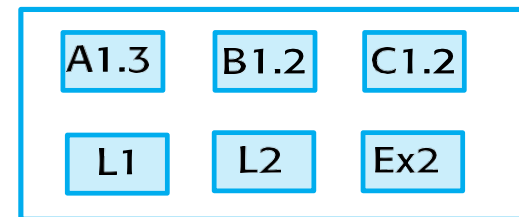
Libraries and external components



Baseline - V1



Baseline - V2



Mainline

- Codeline là một chuỗi các phiên bản mã nguồn với các phiên bản mới hơn trong chuỗi bắt nguồn từ các phiên bản trước đó.

3.2 Hệ thống kiểm soát phiên bản

- Hệ thống quản lý phiên bản kết hợp các thủ tục và các công cụ để quản lý các phiên bản khác nhau của các đối tượng cấu hình được tạo ra trong quá trình phần mềm
- Một hệ thống kiểm soát phiên bản thực hiện hoặc được tích hợp trực tiếp với bốn tính năng chính:
 - **Một cơ sở dữ liệu dự án (kho)** lưu trữ tất cả các đối tượng cấu hình có liên quan. *repo*
 - **Quản lý phiên bản** cho phép lưu trữ tất cả các phiên bản của đối tượng cấu hình (hoặc cho phép bất kỳ phiên bản được xây dựng bằng cách sử dụng sự khác biệt so với phiên bản trước đây).
 - **Make facility** cho phép các kỹ sư phần mềm để thu thập tất cả các đối tượng cấu hình có liên quan và xây dựng một phiên bản đặc biệt của phần mềm. *tiền tích*
 - **Theo dõi vấn đề** (còn được gọi là *bug tracking*) cho phép các nhóm ghi lại và theo dõi tình trạng của tất cả các vấn đề nổi bật liên quan đến từng đối tượng cấu hình.



3.2 Hệ thống kiểm soát phiên bản (2)

- Có hai loại hệ thống điều khiển phiên bản hiện đại
 - **Hệ thống tập trung**, nơi có một kho lưu trữ chính duy nhất duy trì tất cả các phiên bản của các thành phần phần mềm đang được phát triển. **Subversion** là một ví dụ về hệ thống kiểm soát phiên bản tập trung.
 - **Hệ thống phân tán**, nơi tồn tại nhiều phiên bản của kho thành phần cùng một lúc. **Git** là một ví dụ về hệ thống kiểm soát phiên bản phân tán.

3.2 Hệ thống kiểm soát phiên bản (3)

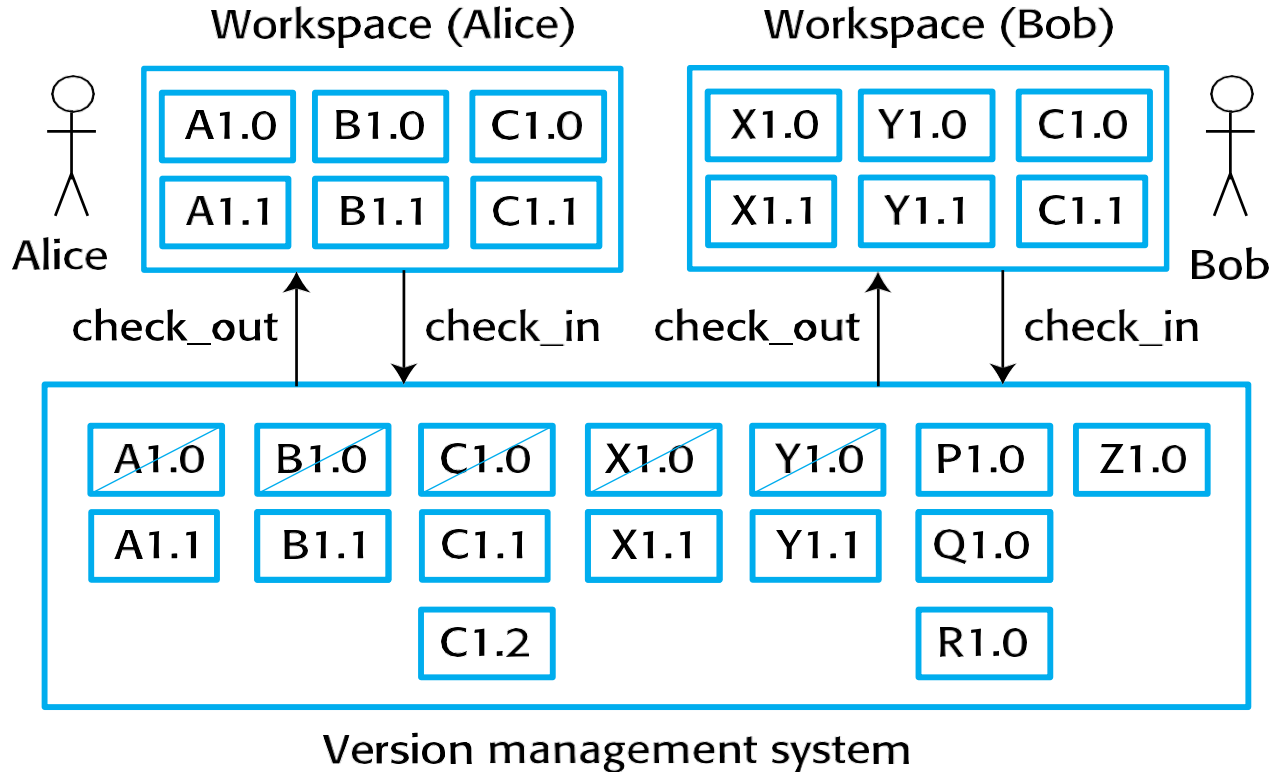
- Khái niệm về *public repository* và *private workspaces*
 - Để hỗ trợ phát triển độc lập mà không bị can thiệp, hệ thống kiểm soát phiên bản sử dụng khái niệm kho lưu trữ dự án và không gian làm việc riêng.
 - **Kho lưu trữ dự án duy trì phiên bản 'chính'** của tất cả các thành phần. Nó được sử dụng để tạo đường cơ sở cho việc xây dựng hệ thống.
 - Khi sửa đổi các thành phần, nhà phát triển sao chép (**check- out**) những thành phần này từ kho lưu trữ vào không gian làm việc của họ và làm việc trên các bản sao này.
 - Khi họ hoàn thành các thay đổi của mình, các thành phần đã thay đổi sẽ được trả lại (**checked-in**) vào kho lưu trữ.

Kiểm soát phiên bản tập trung

- Các nhà phát triển *check-out* các thành phần hoặc thư mục của các thành phần từ kho lưu trữ dự án vào không gian làm việc riêng tư của họ và làm việc trên các bản sao này.
- Khi các thay đổi của họ hoàn tất, họ *check-in* các thành phần trở lại kho lưu trữ.
- Nếu nhiều người đang làm việc trên một thành phần cùng một lúc, mỗi người *check-out* nó từ kho lưu trữ → hệ thống sẽ cảnh báo rằng nó đã được người khác *check-out*

Kiểm soát phiên bản tập trung (2)

- Ví dụ:



Kiểm soát phiên bản phân tán

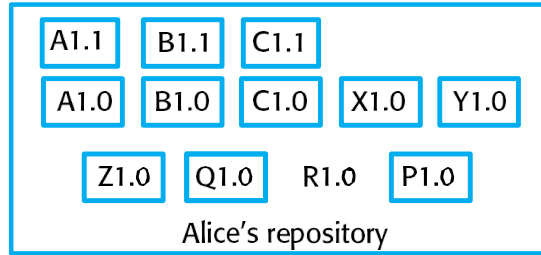
- Kho lưu trữ '**master**' được tạo trên máy chủ duy trì mã nguồn do nhóm phát triển tạo ra.
- Thay vì *check-out* các tệp cần dùng, một nhà phát triển tạo một bản sao của kho lưu trữ dự án được tải xuống và cài đặt trên máy tính. *git clone*
- Các nhà phát triển làm việc trên các tệp được yêu cầu và duy trì các phiên bản mới trên kho lưu trữ *local* riêng trên máy tính của mình.
- Khi các thay đổi được thực hiện, nhà phát triển '**commit**' những thay đổi này và cập nhật kho lưu trữ riêng. Sau đó, có thể '**push**' những thay đổi này vào kho dự án.

Kiểm soát phiên bản phân tán (2)

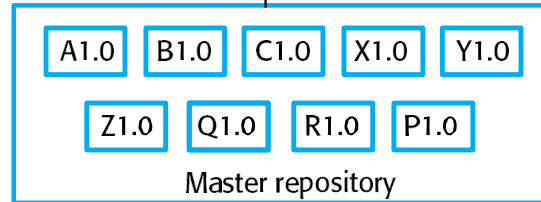
- Ví dụ:

Repository cloning

Alice

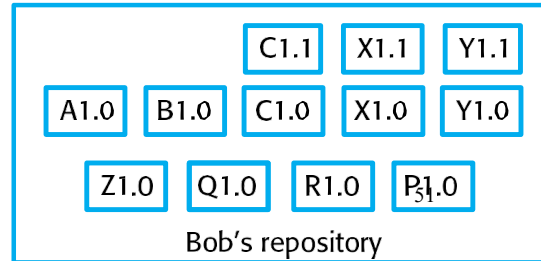


clone



clone

Bob

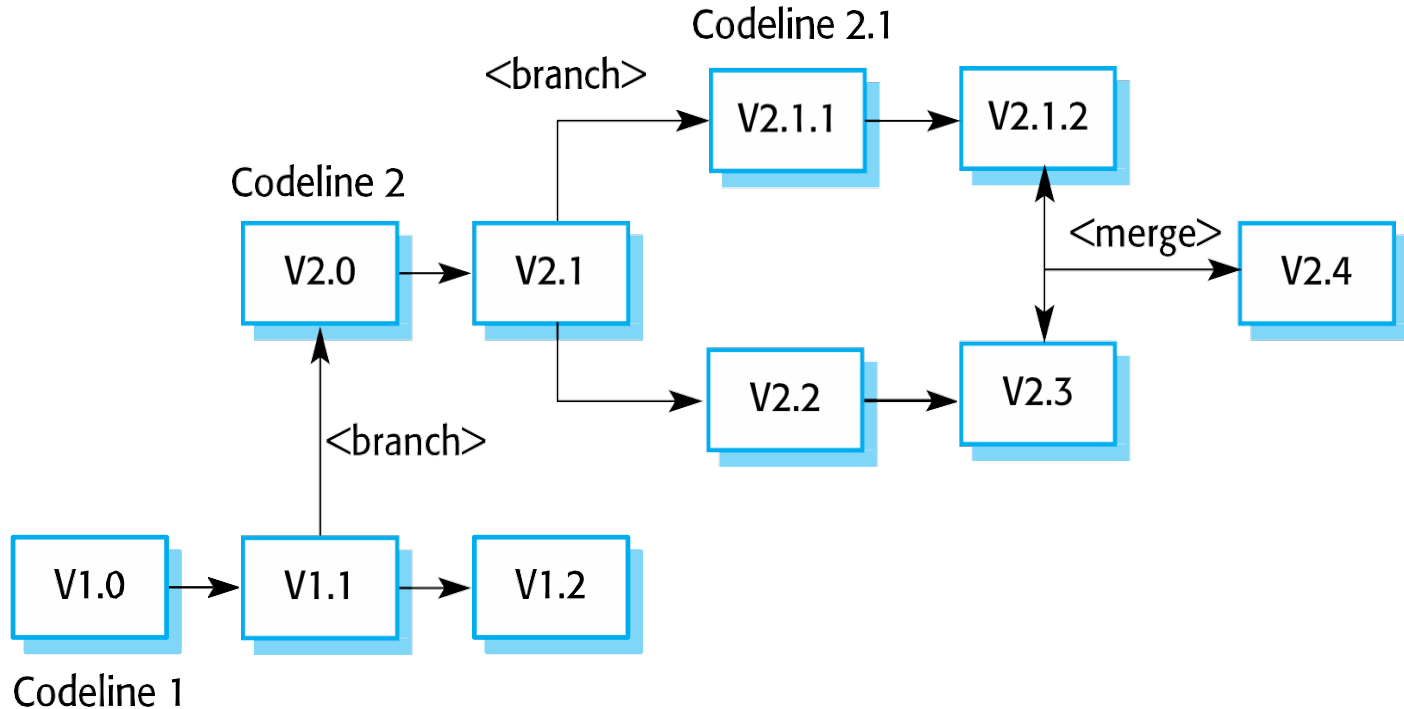


Kiểm soát phiên bản phân tán (3)

- Phân nhánh và hợp nhất (branching & merging):
 - Thay vì một chuỗi phiên bản tuyến tính phản ánh các thay đổi đối với thành phần theo thời gian, có thể có một số chuỗi độc lập.
 - Điều này là bình thường trong quá trình phát triển hệ thống, nơi các nhà phát triển khác nhau làm việc độc lập trên các phiên bản khác nhau của mã nguồn và vì vậy hãy thay đổi nó theo những cách khác nhau.
 - Ở một số giai đoạn, có thể cần hợp nhất các nhánh dòng mã để tạo phiên bản mới của thành phần bao gồm tất cả các thay đổi đã được thực hiện.
 - Nếu các thay đổi được thực hiện liên quan đến các phần khác nhau của mã, các phiên bản thành phần có thể được hợp nhất tự động bằng cách kết hợp các delta áp dụng cho mã.

Kiểm soát phiên bản phân tán (4)

- Phân nhánh và hợp nhất (branching & merging):



Kiểm soát phiên bản phân tán (5)

- Lợi ích:
 - Cung cấp một cơ chế sao lưu cho kho lưu trữ.
 - Nếu kho lưu trữ bị hỏng, công việc có thể tiếp tục và kho lưu trữ dự án có thể được khôi phục từ các bản sao cục bộ.
 - Cho phép hoạt động ngoại tuyến để các nhà phát triển có thể thực hiện các thay đổi nếu không có kết nối mạng.
 - Các nhà phát triển có thể biên dịch và kiểm tra toàn bộ hệ thống trên máy cục bộ và kiểm tra những thay đổi đã thực hiện.

3.3 Đánh số phiên bản

- Định danh/đánh số phiên bản
 - Một trong những hoạt động nền tảng của quản lý cấu hình.
 - Mục đích của định danh là để xác định tính duy nhất của một mục cấu hình (CI), cũng như mối quan hệ của nó với các CI khác.
 - Nó bao gồm việc mô tả tên, đánh số, đánh dấu đặc trưng, giúp nhận biết và phân biệt một CI với các CI hay thành phần khác.
- Một số phương pháp gán số hiệu phiên bản được dùng phổ biến:
 - Đánh số phiên bản bằng các con số (Sequence-based identifiers)
 - Đánh số hiệu phiên bản dựa theo mức độ ổn định của sản phẩm (Stage-based identifiers)

ý giai đoạn

3.3 Đánh số phiên bản (2)

- Đánh số phiên bản bằng các con số
 - Sử dụng các con số (đôi khi kết hợp thêm các chữ cái) để gán số hiệu cho các phiên bản
 - Công thức đánh số hiệu phiên bản:
major.minor.[build [.revision]] hoặc
major.minor [maintenance[.build]]
- Ý nghĩa các số major, minor, build, revision
 - major: Chuỗi phiên bản chính
 - minor: Chuỗi phiên bản phụ *đánh dấu cho n thay đổi nhỏ*
 - build: Chuỗi phiên bản cấu tạo. Đánh dấu sự khác nhau trong cùng 1 phiên bản phụ, 2 chữ số
 - revision: Lần sửa đổi, đánh dấu lần sửa đổi của mã nguồn

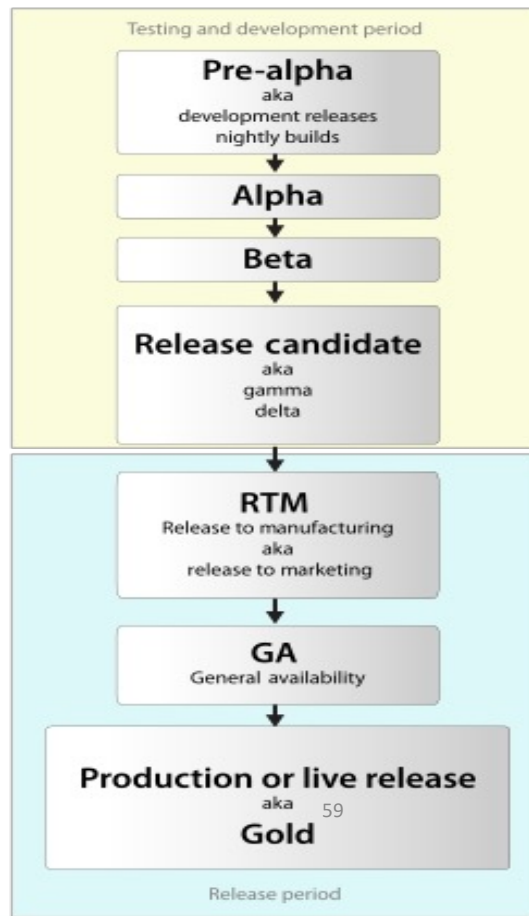
Ví dụ: mô hình định danh 3 chữ số

MUE.0.0.1:Release	Alpha test release
MUE.1.0.0:Release	First major release
MUE.1.2.1:Release	Second minor release with bug fixes
MUE.2.0.3:Release	Second major release with three series of bug fixes

3.3 Đánh số phiên bản (3)

- Đánh số hiệu phiên bản dựa theo mức độ ổn định của sản phẩm, sử dụng các tên gọi:
 - Closebeta: Phiên bản thử nghiệm hạn chế
 - Openbeta: Phiên bản thử nghiệm diện rộng
 - Release Candidate (RC): phiên bản ứng viên
 - Official version: phiên bản chính thức
 - ...

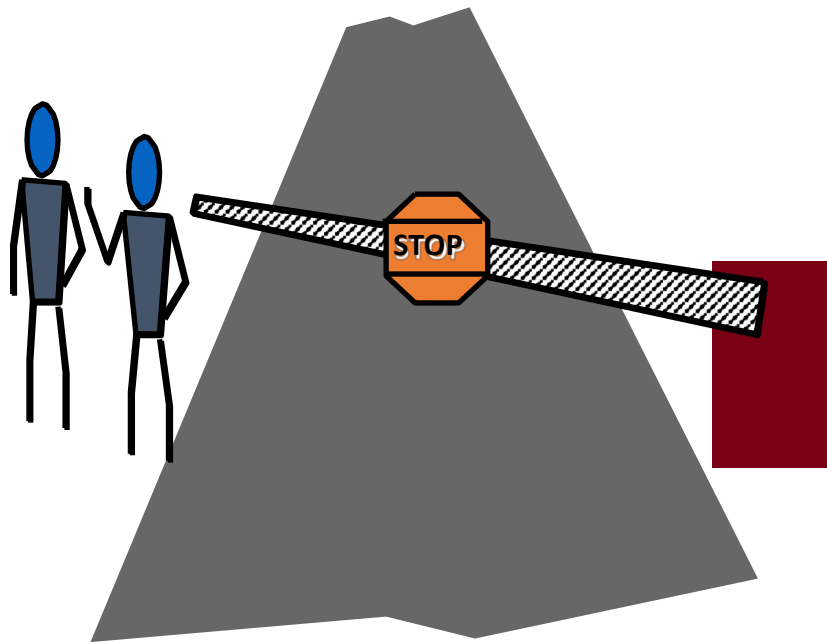
Ví dụ: mô hình tên phiên bản theo độ ổn định của sản phẩm



Nội dung

1. Khái niệm quản lý cấu hình phần mềm
2. Quy trình quản lý cấu hình phần mềm
3. Quản lý phiên bản
- 4. Quản lý thay đổi**

Quan lý thay đổi



61

Quan lý thay đổi (2)

xử lý / phân tích
phê duyệt
↓
theo dõi

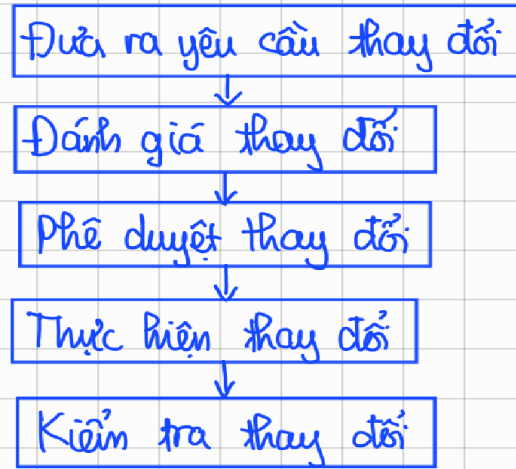
- Các yêu cầu của tổ chức thay đổi trong suốt thời gian tồn tại của hệ thống, các lỗi phải được sửa chữa và các hệ thống phải thích ứng với những thay đổi trong môi trường của chúng.
- **Quản lý thay đổi** nhằm đảm bảo rằng sự phát triển của hệ thống là một quá trình được quản lý và ưu tiên dành cho những thay đổi cấp bách nhất và tiết kiệm chi phí.
- Quá trình quản lý thay đổi liên quan đến việc **phân tích chi phí và lợi ích** của những thay đổi được đề xuất, **phê duyệt** những thay đổi đáng giá và **theo dõi** những thành phần nào trong hệ thống đã được thay đổi.

Quan lý thay đổi (3)

- Mức độ phức tạp của quy trình quản lý thay đổi thay đổi theo dự án.
- Các dự án nhỏ có thể thực hiện các yêu cầu thay đổi một cách không chính thức và nhanh chóng trong khi các dự án phức tạp yêu cầu các biểu mẫu yêu cầu thay đổi chi tiết và sự chấp thuận chính thức của một người quản lý khác.

Quy trình quản lý thay đổi

- Thay đổi được yêu cầu (có thể được thực hiện bởi bất kỳ ai bao gồm cả người dùng và nhà phát triển)
- Yêu cầu thay đổi được đánh giá dựa trên các mục tiêu của dự án
- Sau khi đánh giá, thay đổi được chấp nhận hoặc bị từ chối
- Nếu nó được chấp nhận, thay đổi được chỉ định cho người phát triển và được triển khai
- Thay đổi đã thực hiện được kiểm tra.



Quy trình quản lý thay đổi (2)

Nhu cầu thay đổi được công nhận



Thay đổi yêu cầu từ người dùng



Đánh giá từ nhà phát triển



Báo cáo thay đổi được tạo ra



Quyền kiểm soát thay đổi quyết định



Yêu cầu được xếp hàng
đợi để hành
động

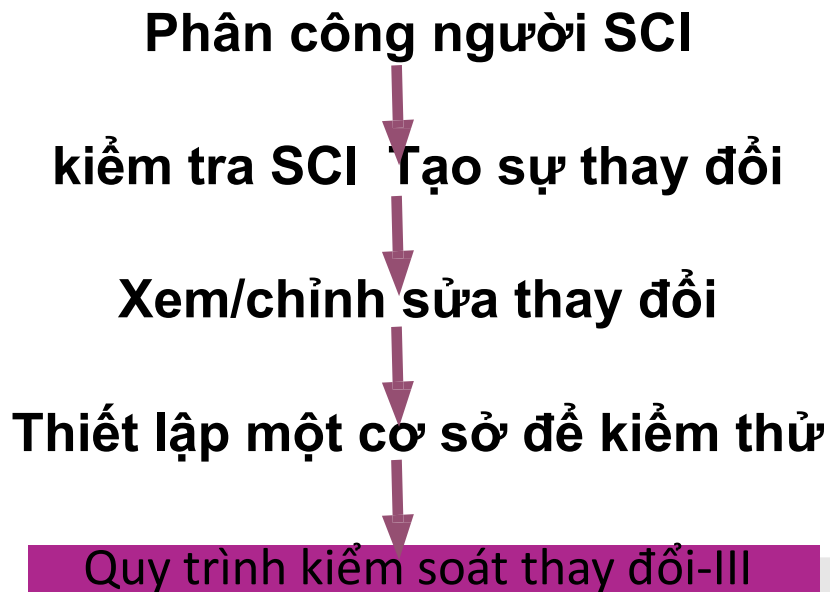


Quy trình kiểm soát thay đổi-I

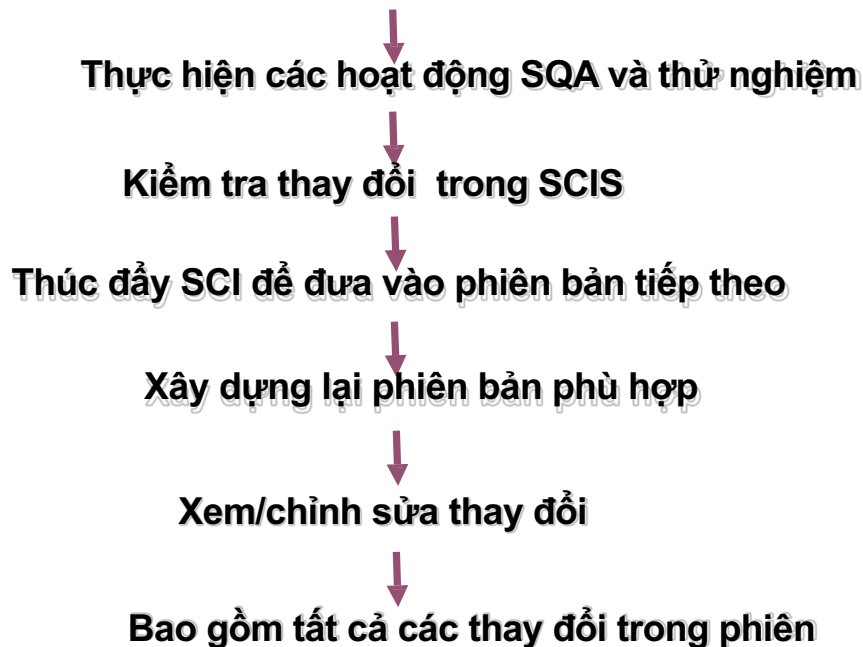


yêu cầu thay đổi bị từ chối
người dùng được thông báo

Quy trình quản lý thay đổi (3)



Quy trình quản lý thay đổi (3)



TỔNG KẾT

- Hệ thống phần mềm liên tục thay đổi trong quá trình phát triển và sử dụng.
- Quản lý cấu hình phần mềm là một phần cơ bản của kế hoạch quản lý dự án để quản lý các hệ thống phần mềm đang phát triển và điều phối các thay đổi đối với chúng.
- Quản lý cấu hình liên quan đến các chính sách, quy trình và công cụ để quản lý các hệ thống phần mềm đang thay đổi.
- SCM cần thiết vì rất dễ mất dấu những thay đổi và các phiên bản thành phần đã được tích hợp vào mỗi phiên bản hệ thống.
- SCM rất cần thiết cho các dự án nhóm để kiểm soát các thay đổi do các nhà phát triển khác nhau thực hiện