**Version Control System**

2016/06/03: Tìm hiểu các hệ thông quả lý version phổ biến hiện nay

1. Vì sao cần quản lý source

Dẫn truyện:

“ Có hai anh chàng lập trình viên đang hợp tác với nhau cùng xây dựng một phần mềm cho khách hàng. Dĩ nhiên hai anh chàng này sẽ làm việc với nhau bằng cách mỗi người tự viết code một ít và gửi cho nhau qua email hoặc một phương thức gửi dữ liệu thông thường nào đó. Mỗi người sau khi nhận được code của nhau sẽ tiến hành tự xem và sửa lại, sau đó cùng nhau ráp nối vào phần mềm mà họ cần làm. Tuy nhiên, một hôm anh A bỗng nhiên vô tình viết code sai vào phần mềm dẫn tới sản phẩm bị lỗi mà trước đó cả hai anh đều không hề sao lưu lại do phần mềm quá lớn. Và thế là hai anh lại cùng nhau xây dựng lại tính năng bị lỗi và tiếp tục gửi cho nhau như vậy. ”

Thế bạn có thấy vấn đề gì xảy ra ở câu chuyện trên không? Mình thì mình thấy có hai vấn đề như sau:

1. Việc tự code riêng và gửi cho nhau qua email rất mất thời gian của nhau. Giá như anh A có thể chủ động xem những thay đổi của anh B từ xa và tiến hành gộp trực tiếp những thay đổi của anh B vào sản phẩm.
2. Việc sửa code mà không sao lưu khiến cho họ phải viết lại code từ đầu khi phần mềm bị lỗi.

Và hai vấn đề trên đều là 2 vấn đề thường xuyên gặp phải trong quá trình làm việc nhóm với nhau. Vì sao khi làm việc ta luôn cần backup lại các file mã nguồn đã viết trước đó ? Nếu không có bản backup thì sao ?

Có thể trong quá trình làm việc với các project vừa và nhỏ, có thời gian hoạt động không lâu dài thì việc sử dụng các hệ thống quản lý version là một việc khiến cho bạn mất them thời gian trong quá trình phát triển sản phẩm. Tuy nhiên nếu hệ thống phát triển lâu dài thì việc quản lý version là việc rất quan trọng, đơn giản là nếu nhiều người trong team cùng sửa 1 file vào cùng 1 thời điểm thì sao ? Không lẽ người này đợi người kia làm xong rồi mới sửa. Hoặc là nếu 1 người sửa và sinh ra lỗi cho phần của người khác thì sao ? Rất nhiều vấn đề nảy sinh khi làm việc nhóm, vì vậy mà việc quản lý version là rất quan trong trong khi làm việc nhóm.

Như vậy tổng quan thì hệ thống quản lý version sẽ giúp ta xác định , phục hồi lại trạng thái tại 1 thời điểm nào đó trong quá khứ , giúp làm việc nhóm hiệu quả hơn.

Tham khảo: <http://www.atmarkit.co.jp/ait/articles/1305/20/news015.html>

「集中管理方式」と「分散管理方式」

　バージョン管理システムとは、ファイルに対して「誰が」「いつ」「何を変更したか」というような情報を記録することで、過去のある時点の状態を復元したり変更内容の差分を表示できるようにするシステムのことです。バージョン管理システムは大きく2つに分けると、「集中管理方式」「分散管理方式」があります。

　過去には集中管理方式の「CVS」「Subversion」が多く利用されていましたが、複数人での分散開発の容易さやパフォーマンスに優れた分散管理方式の「Git」「Mercurial」などがスタンダードになりつつあります。

1. SVN

+ Hướng dẫn cài đặt và sử dụng Tortoise SVN <http://o7planning.org/web/fe/default/vi/document/32313/huong-dan-cai-dat-va-su-dung-tortoise-svn>

+ Hướng dẫn cài đặt và quản lý Visual SVN Server　<http://o7planning.org/web/fe/default/vi/document/19481/huong-dan-cai-dat-va-quan-ly-visual-svn-server>

+ Cài đặt Subversive vào Eclipse

<http://o7planning.org/web/fe/default/vi/document/4975/cai-dat-subversive-vao-eclipse>

+ Sử dụng SVN trong Eclipse

<http://o7planning.org/web/fe/default/vi/document/19490/lap-trinh-java-theo-nhom-su-dung-eclipse-va-svn>

1. Git