


バージョン管理の基本 ～ 集中型と分散型

鈴木耕介

2014/3/24


- 1 はじめに
- 2 目的
- 3 バージョン管理システムとは
- 4 集中型と分散型の違い
- 5 まとめ



ヤバイにゃ・・・
外部設計書間違って更新しちゃったにゃ



おう！ 早く直せよコラ！！



すんません！
直せません！
ごめん寝！

- ・ 誰しもあるとおもいます

- 今日はそんな悩みを解決してくれるバージョン管理システムについてお話します

- 1 はじめに
- 2 目的
- 3 バージョン管理システムとは
- 4 集中型と分散型の違い
- 5 まとめ

- ① バージョン管理システムの基礎を理解する

- ① バージョン管理システムの基礎を理解する
- ② 集中型と分散型の違いを理解する

- 1 はじめに
- 2 目的
- 3 バージョン管理システムとは
- 4 集中型と分散型の違い
- 5 まとめ

① バージョン管理システム (VCS:Version Control System)

① バージョン管理システム (VCS:Version Control System)

- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること

① バージョン管理システム (VCS:Version Control System)

- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること
- 主にソースコードの管理で使用されている

① バージョン管理システム (VCS:Version Control System)

- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること
- 主にソースコードの管理で使用されている

② 代表的なバージョン管理ツールはこれ

① バージョン管理システム (VCS:Version Control System)

- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること
- 主にソースコードの管理で使われている

② 代表的なバージョン管理ツールはこれ

- CVS
 - VCS の火付け役。お客さんはコレ使ってる

① バージョン管理システム (VCS:Version Control System)

- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること
- 主にソースコードの管理で使用されている

② 代表的なバージョン管理ツールはこれ

- CVS
 - VCS の火付け役。お客さんはコレ使ってる
- Subversion
 - CVS の後継機的位置。WEB チームはコレ使ってる

① バージョン管理システム (VCS:Version Control System)

- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること
- 主にソースコードの管理で使用されている

② 代表的なバージョン管理ツールはこれ

- CVS
 - VCS の火付け役。お客さんはコレ使ってる
- Subversion
 - CVS の後継機的位置。WEB チームはコレ使ってる
- Git
 - 従来のツールとは違うポジション。最近の主流はコレ

① バージョン管理システム (VCS:Version Control System)

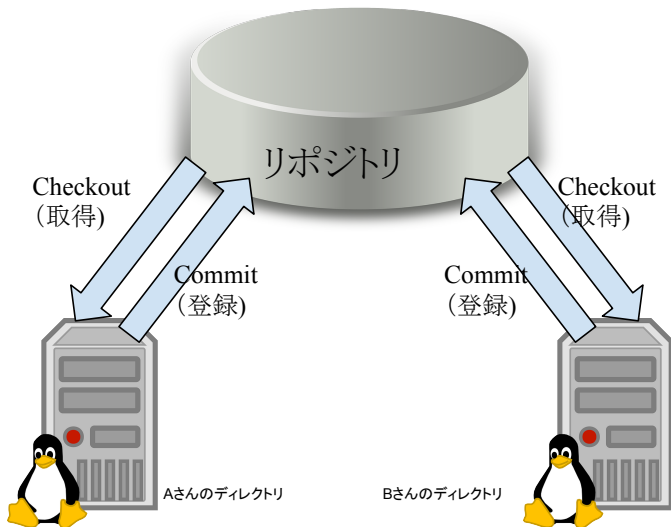
- ファイルの作成日時・変更履歴・更新日時をリポジトリに保管し、管理すること
- 主にソースコードの管理で使用されている


② 代表的なバージョン管理ツールはこれ

- CVS
 - VCS の火付け役。お客さんはコレ使ってる [集中型]
- Subversion
 - CVS の後継機的位置。WEB チームはコレ使ってる [集中型]
- Git
 - 従来のツールとは違うポジション。最近の主流はコレ [分散型]

- 1 はじめに
- 2 目的
- 3 バージョン管理システムとは
- 4 集中型と分散型の違い**
- 5 まとめ

仕組み(集中型)



A man in a blue t-shirt is using an ATM. He is holding a card and inserting it into the machine. The ATM is grey and has a screen and a keypad. The background is red.

銀行のATMみたいなものです

集中型の特徴 (弱点)

- ① 手元に最新版 (現在の状況) しかない

集中型の特徴 (弱点)

- ① 手元に最新版 (現在の状況) しかない
- ② 変更履歴を問い合わせるには中央リポジトリに問い合わせる必要がある。

もしもリポジトリと
接続ができない状態になってし
まったら・・・？

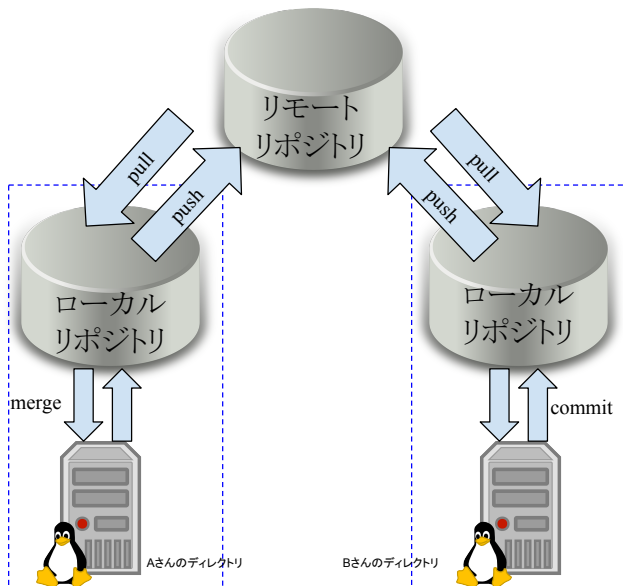
残高がわからないじゃない！



- ① 常に中央リポジトリの接続している必要があります

- ① 常に中央リポジトリの接続している必要があります
- ② 当然中央リポジトリがお亡くなりになったら **OUT**

仕組み(分散型)





各々が自分専用の銀行を所持している
ようなもんです

- ① 一時的作業の履歴管理が容易

分散型の特徴

- ① 一時的作業の履歴管理が容易
- ② オフライン環境でも開発を進められる

- ① はじめに
- ② 目的
- ③ バージョン管理システムとは
- ④ 集中型と分散型の違い
- ⑤ まとめ

最後に分散型の
全般的なメリデメをご紹介します

強弱まとめ (分散型視点)

① 強み

② 弱み

強弱まとめ (分散型視点)

① 強み

- 一時的作業の履歴管理が容易
- 柔軟なワークフロー
- パフォーマンスとスケーラビリティ
- オフラインによる開発
- 障害に強い

② 弱み

強弱まとめ (分散型視点)

① 強み

- 一時的作業の履歴管理が容易
- 柔軟なワークフロー
- パフォーマンスとスケーラビリティ
- オフラインによる開発
- 障害に強い

② 弱み

- 管理が煩雑
- ロックができない (楽観的ロック)
- 細かいアクセス制御ができない
- 有識者が少ない



偉い人曰く、お金と時間に余裕がある
ならGitに変えることもアリだと

- ① text 入門 git Travis Swicegood オーム社
- ② ガチで5分で分かる分散型バージョン管理システム Git
<http://www.atmarkit.co.jp/ait/articles/1307/05/news028.html>
- ③ 分散バージョン管理 Git/Mercurial/Bazaar 徹底比較
<http://www.atmarkit.co.jp/ait/articles/0901/14/news133.html>

- ① スイマセンまだ Git の勉強始めたばかりなのです。
- ② Github も合わせて勉強しています。
- ③ ちなみにこのスライドは tex ファイルで作成してみました。