gitについて

計算機科学コース　中澤篤志

参考：計算機科学実験及び演習のための Git(Yusuke Miyazakiさん)

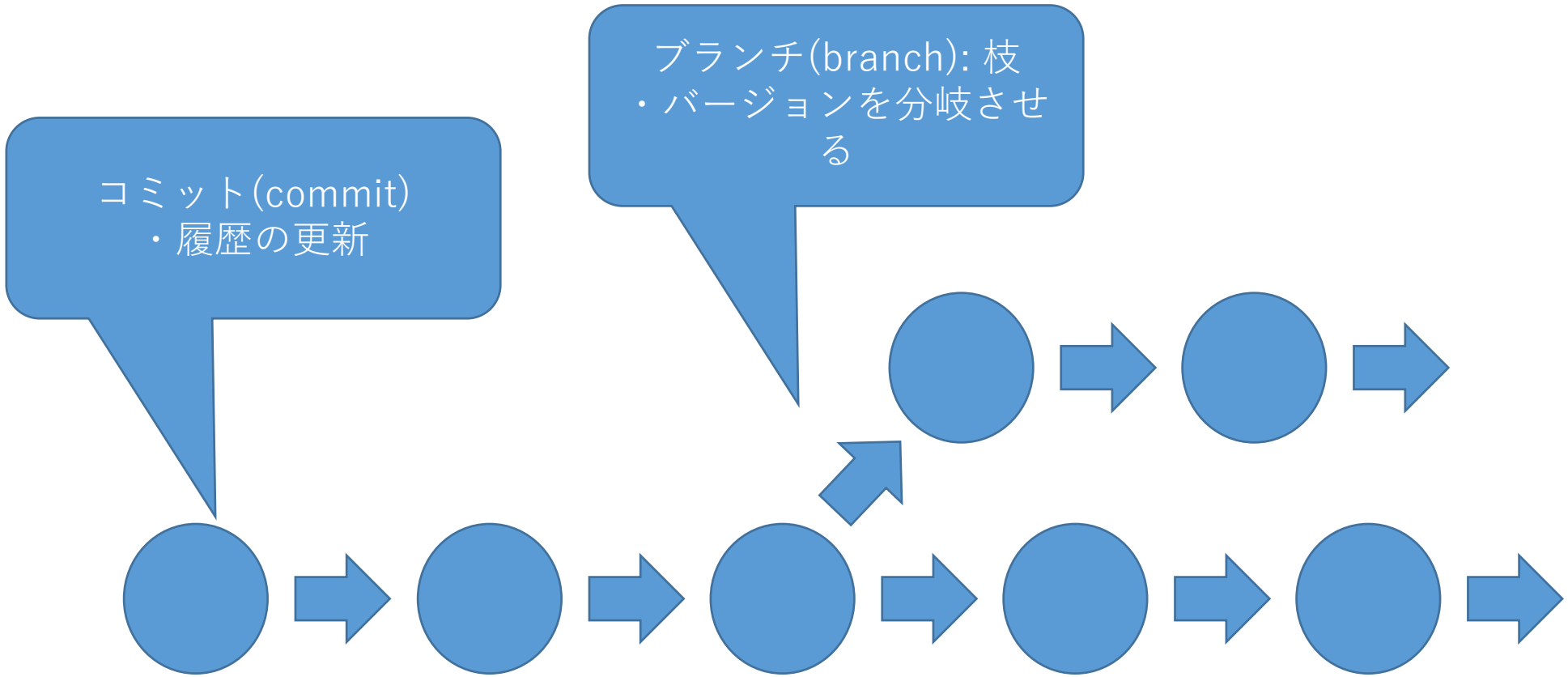
バージョン管理システムとは

- **バージョン管理システム**とは、コンピュータ上で作成、編集されるファイルの変更履歴を管理するためのシステム。特にソフトウェア開発においてソースコードの管理に用いられる。
- バージョン管理システムの最も基本的な機能は、ファイルの作成日時、変更日時、変更点などの履歴を保管することである。これにより、何度も変更を加えた ファイルであっても、過去の状態や変更内容を確認したり、変更前の状態を復元することが容易になる。

要は

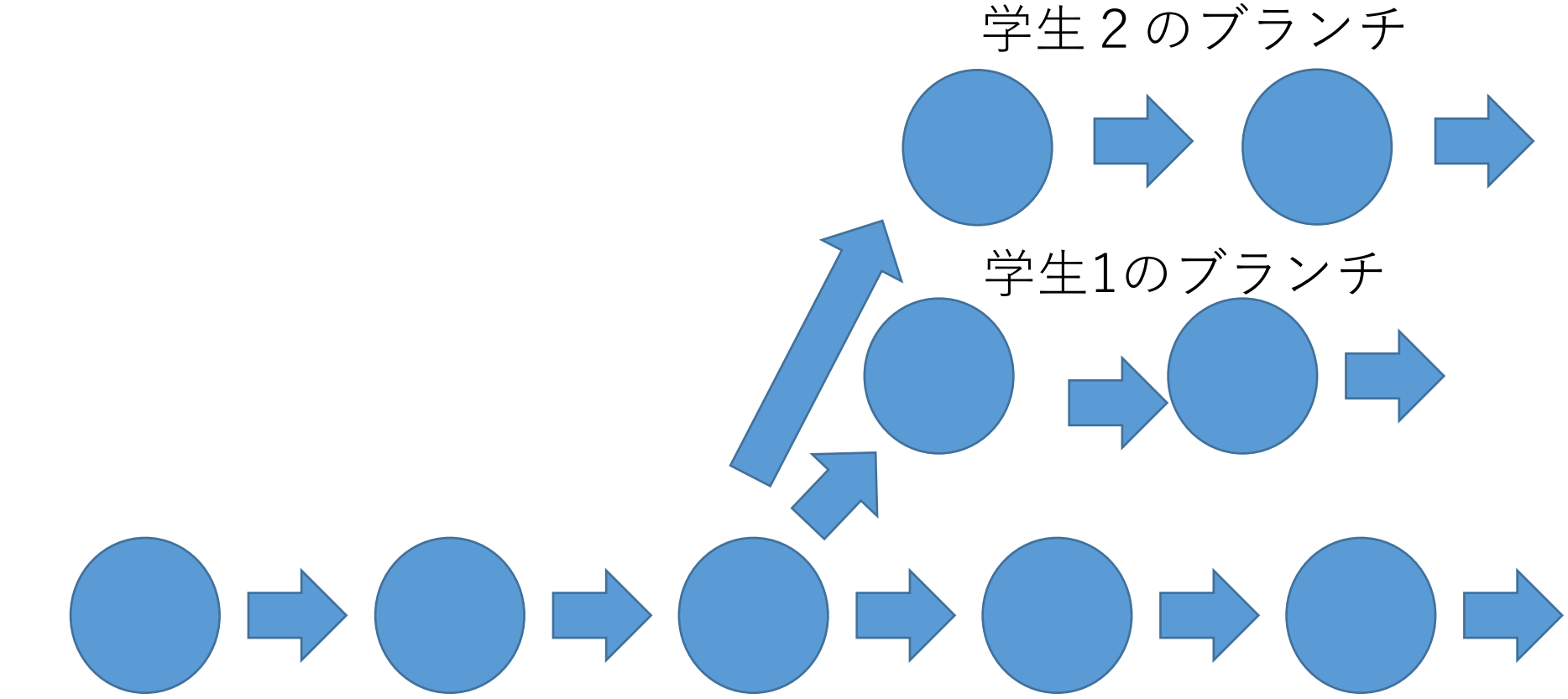
- プログラムのファイルを管理できる
- プログラム変更点などを管理できる
  - 誰が変更したか？を記録
  - 誤った変更を元に戻せる
  - 何を変更したかを確認できる
- Gitは現在最もよく用いられているバージョン管理システムの１つ

Gitの管理イメージ



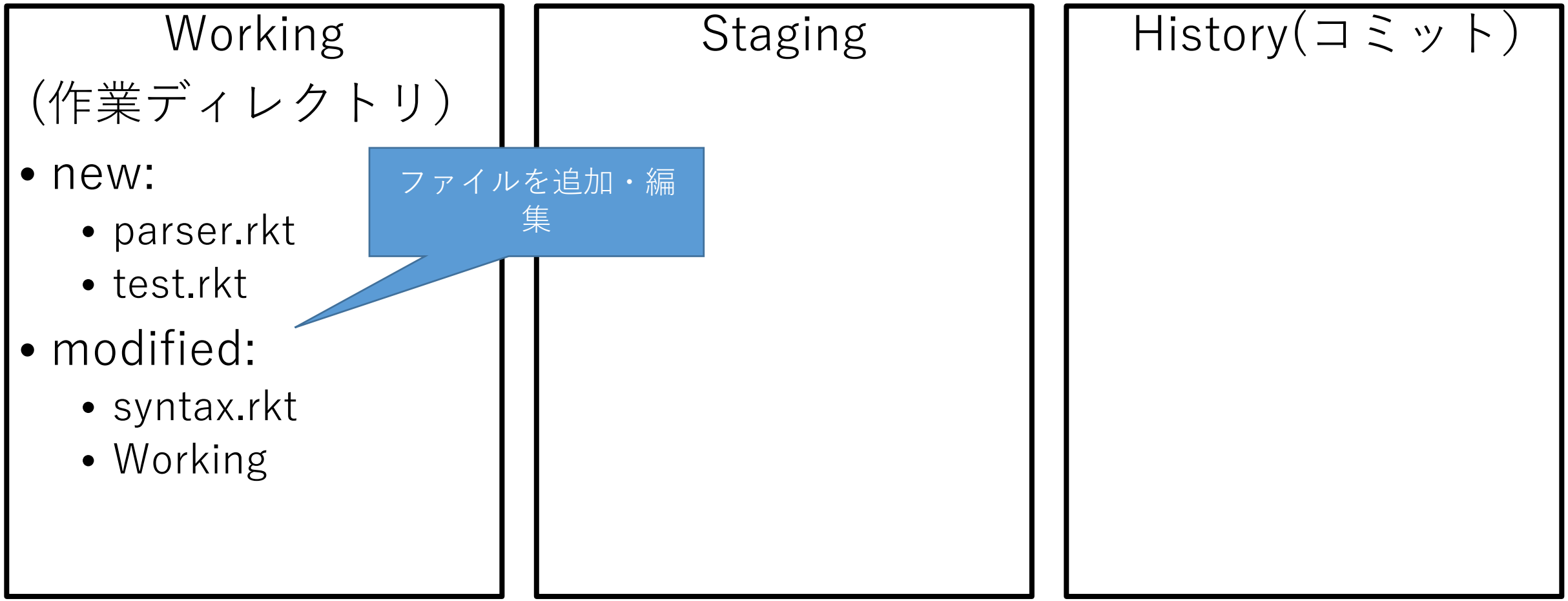
世代管理

Branchの使い方の概念（本実験ではあまり使わないが覚えておく）

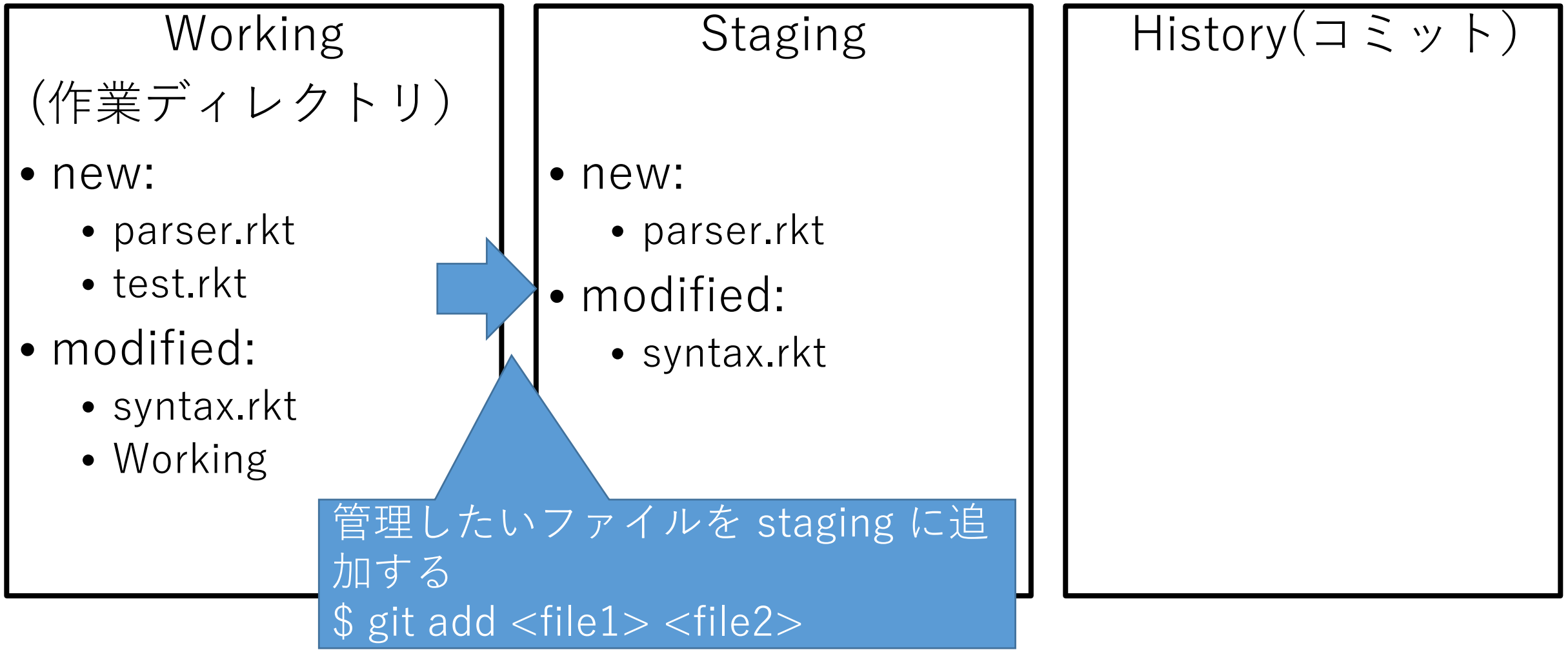


世代管理

管理の流れ


Working
(作業ディレクトリ)
• new:
• parser.rkt
• test.rkt
• modified:
• syntax.rkt
• Working
ファイルを追加・編集
Staging
History(コミット)

管理の流れ（２）


Working
（作業ディレクトリ）
• new:
• parser.rkt
• test.rkt
• modified:
• syntax.rkt
• Working
Staging
• new:
• parser.rkt
• modified:
• syntax.rkt
History(コミット)
管理したいファイルを staging に追加する
$ git add <file1> <file2>

管理の流れ（３）

Working
（作業ディレクトリ）

- new:
  - parser.rkt
  - test.rkt
- modified:
  - syntax.rkt
  - Working

Staging

- new:
  - parser.rkt
- modified:
  - syntax.rkt

History(コミット)

- message
- hash
- new:
  - parser.rkt
- modified:
  - syntax.rkt

Staging の内容をコミット（履歴に登録）
$ git commit -m "<message>"

リポジトリの作成

- バージョン管理を始めるために, 履歴を格納するGit リポジトリを作成する
- 最初に一度だけ行えば良い

- プロジェクトのルートディレクトリで

```
git init
```

- 該当するディレクトリ内に .git という隠しフォルダができる

状態の確認

- 現在の Git リポジトリの状態を確認するには

  `git status`

- Working Dir. で追加・変更された内容やStaging に追加された内容が表示される

管理（１）

- 機能の追加やバグの修正などきりの良い時点でコミットし状態を保存しておく

- 新しくバージョン管理下に置くファイルや変更したファイルを Staging に追加する

  - **`git add <file1> <file2> …`**
  - **`git add .`**（すべての新規作成あるいは変更されたファイルをadd)

管理(2)

- Staging の内容をコミットして履歴に登録する

`git commit -m "<commit message>“`

- コミット時にはメッセージを付加する
- 変更内容を書いておくと後で分かりやすい
  - 例1: パーサーを実装
  - 例2: 意味解析で〜〜〜になるバグを修正

履歴の閲覧1

- 履歴を閲覧する

```
git log
```

- コミットの一覧がハッシュ・メッセージ・日時などとともに表示される

- それぞれのコミットはハッシュ (SHA-1) で一意に特定できる
  - 例: 5a00b5712a039bfea1e8055206ab697e3081247d

- オプションを付加するともっと色々見れる

履歴の閲覧2

- 特定のコミットの変更内容を見る

```
git show <commit>
```

最後のコミットまで戻る

- 変更に問題あった場合、最後のコミットまで戻れる

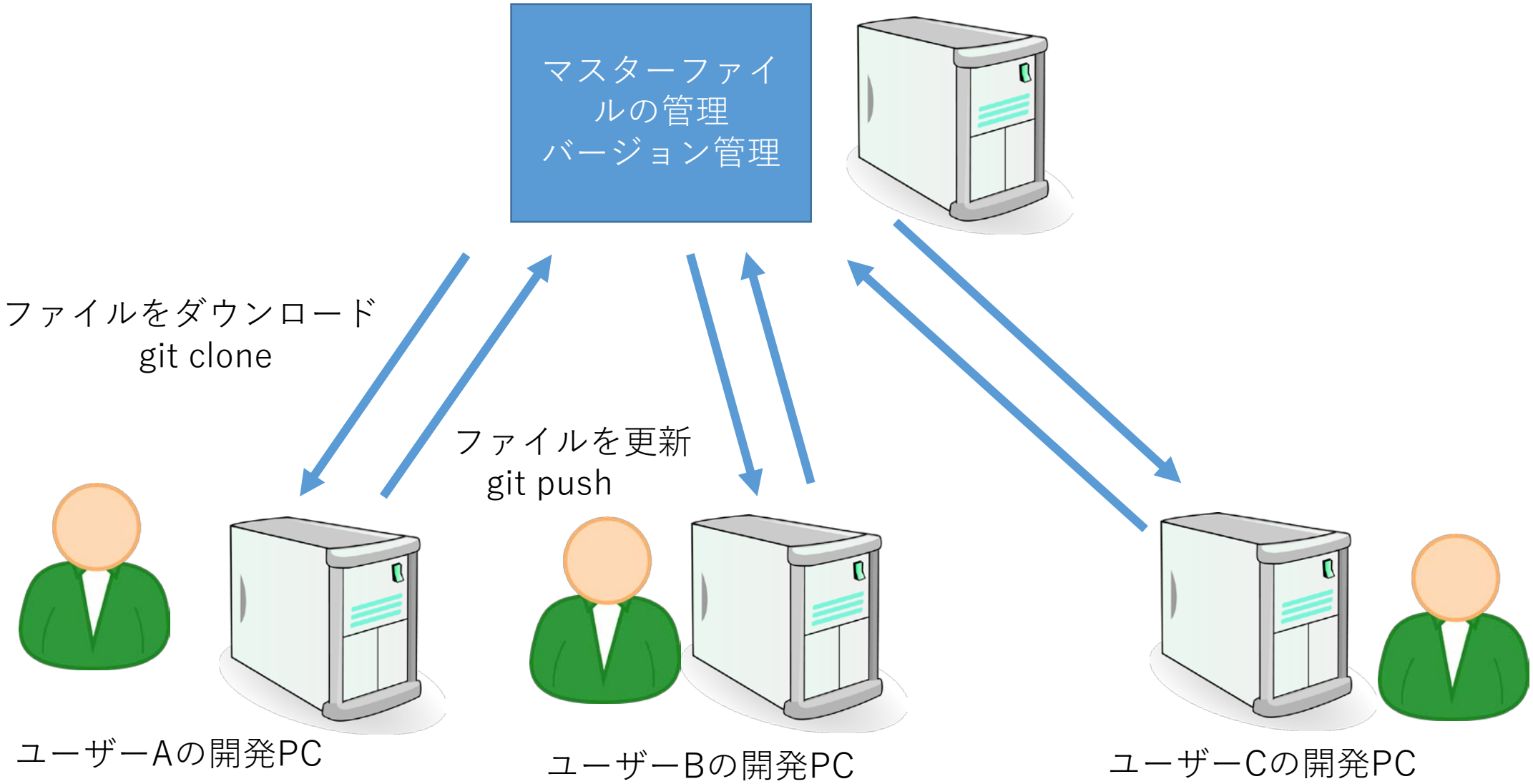
```
git checkout <file or dir>
```

gitとリモートサーバ

- gitはリモートサーバに対して、ファイルを更新・ダウンロードを行うことができる。
- 分散開発（多人数でファイルを管理する）のために有効
- githubなどの商用サービスもある

- 通信プロトコル：ssh (secure socket shell), http, httpsなどが利用できる

gitとリモートサーバ



リモート管理用コマンド

- git clone <url>　　既存のリモートレポジトリを複製する
- git remote add <name> <url>
  リモートレポジトリ(URL)に名前をつける
- git checkout <branch>
  リモートリポジトリのブランチからローカルリポジトリのブランチを作成
- git push <repository> <refspec>
  リモートリポジトリにブランチを作成

今回の学生実験では

- 計算機室端末からのみアクセスできるgitサーバよりファイルをダウンロード(git clone)し、その後は手元でgitによる管理を行う。
- ダウンロード元は、
  - http://10.223.1.242/~le2soft/2017marioAI.zip

手元の端末にダウンロードする

- git clone http://10.223.1.242/~le2soft/2017marioAI.zip
- unzip 2017marioAI.zip
- 2017marioAI.git フォルダができるので、その中で

```
git init
git add .
git commit –m “first commit”
```

- とし、その後は適宜add, commitしていくこと

※注意：パス名に日本語があるとEclipseの設定・コンパイルに失敗するので（例えば“ダウンロード”や”デスクトップ” フォルダ），自分のホームフォルダの下でやるとよい.