



Version Control

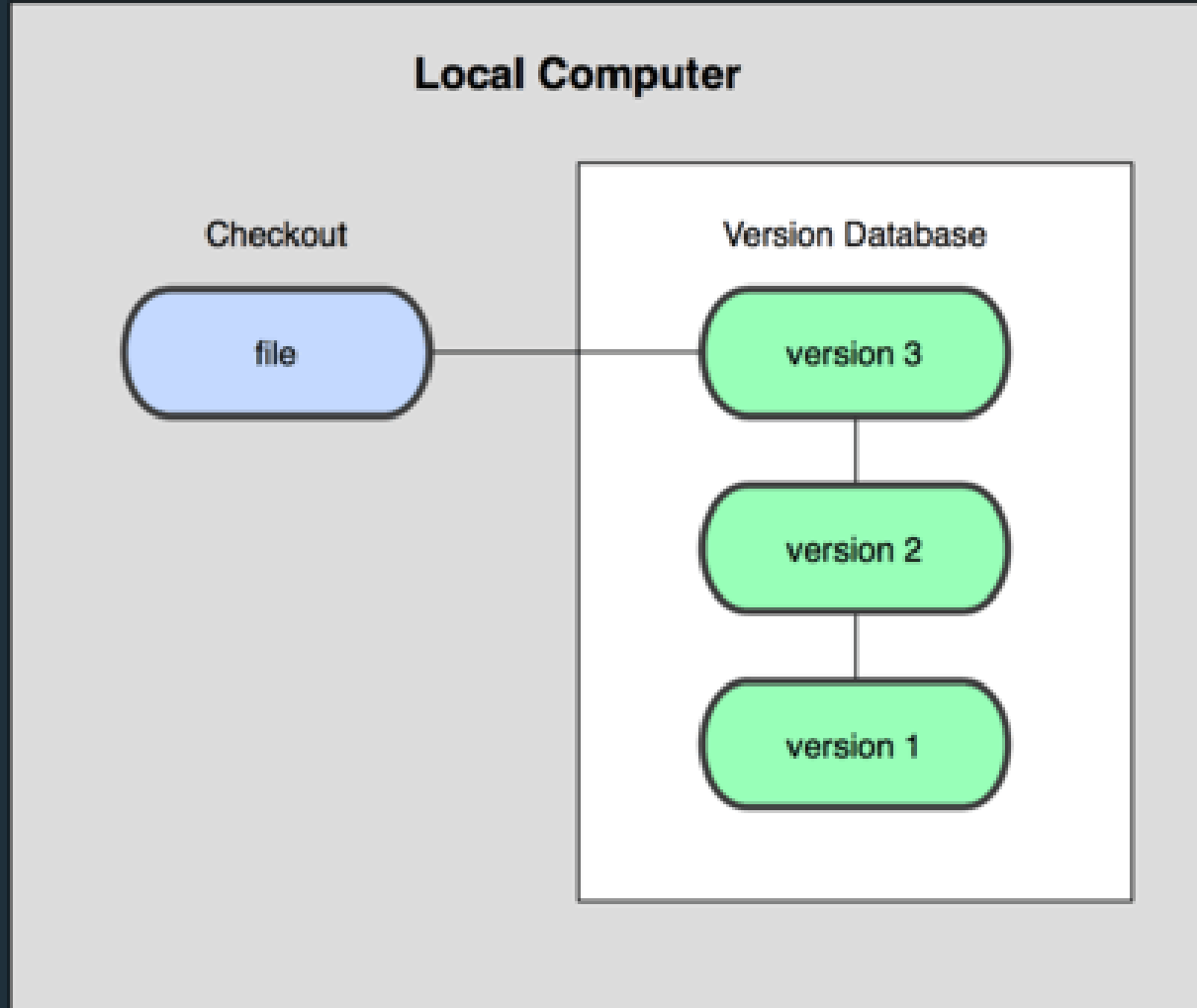
Version Control System(VCS)

- ระบบที่จัดการการเปลี่ยนแปลงที่เกิดขึ้นกับไฟล์หนึ่งหรือหลายไฟล์
- เพื่อที่จะสามารถเรียกเวอร์ชันใดเวอร์ชันหนึ่งกลับมาดูเมื่อไรก็ได้
- version control กับไฟล์ชนิดใดก็ได้

Type Of Version Control

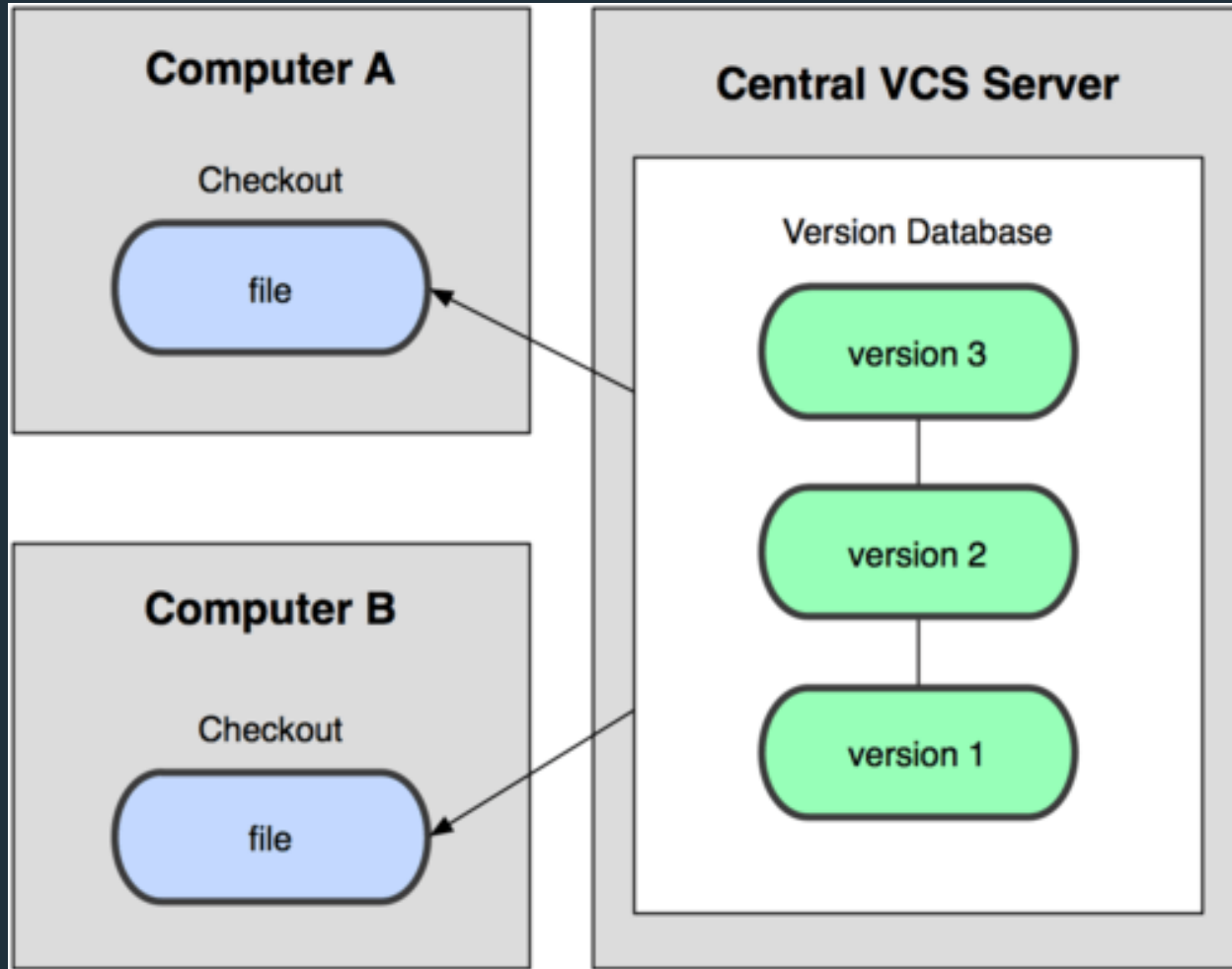
- Local Version Control (LVC)
- Centralized Version Control Systems (CVCSs)
- Distributed Version Control Systems (DVCSs)

Local Version Control



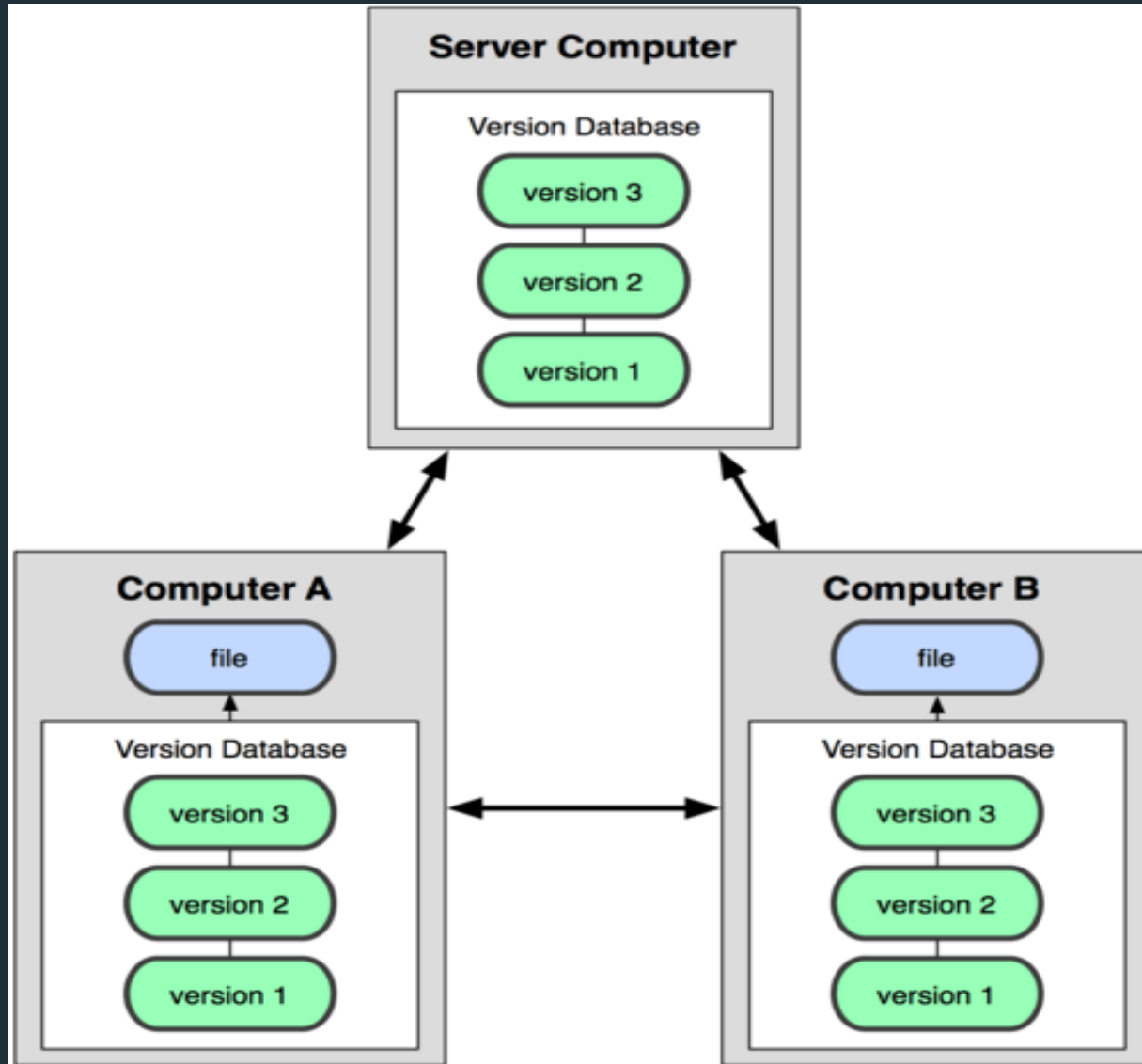
- การคัดลอกไฟล์ไปไว้ในไดเรกทอรีใหม่
- เป็นวิธีที่เกิดข้อผิดพลาดได้ง่าย

Centralized Version Control



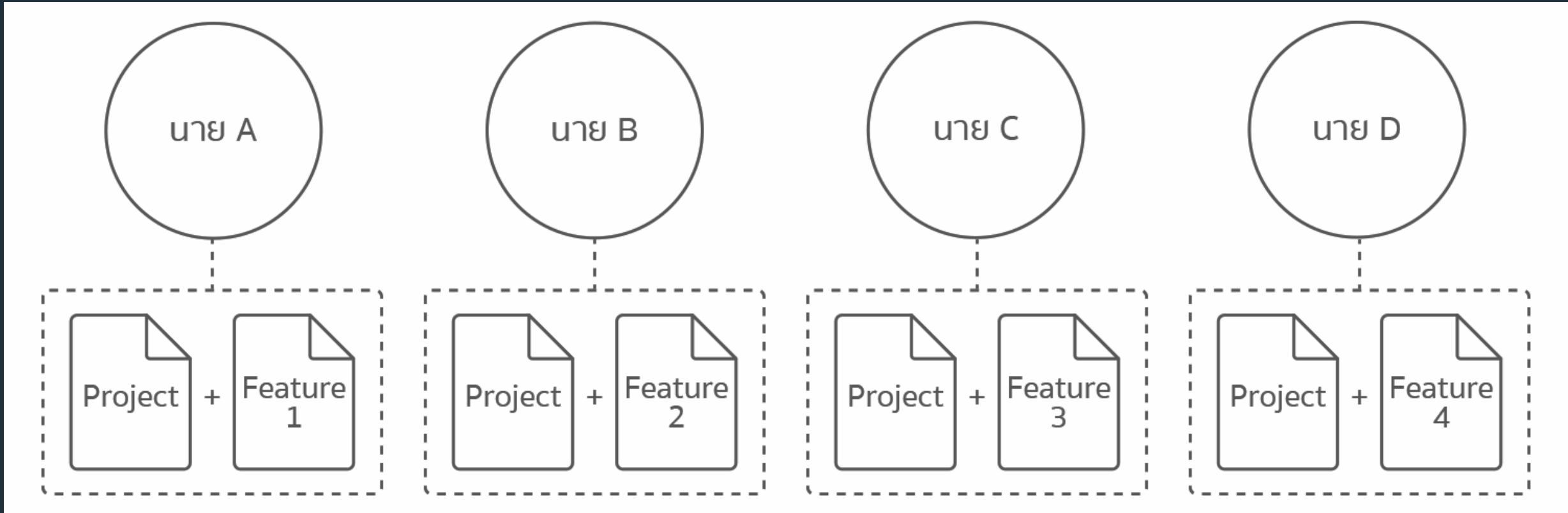
- แบบรวมศูนย์ เช่น CVS, SVN
- มี Server กลางที่เก็บไฟล์ทั้งหมดไว้ในที่เดียวและผู้ใช้หลาย ๆ คนสามารถต่อเข้ามาเพื่อดึงไฟล์จากศูนย์กลางนี้ไปแก้ไขได้
- ถ้า Server ล่ม ก็จะไม่มีการสามารถใช้งานได้ หรือ หากเสีย ก็จะไม่สามารถนำข้อมูลกลับมาได้

Distributed Version Control



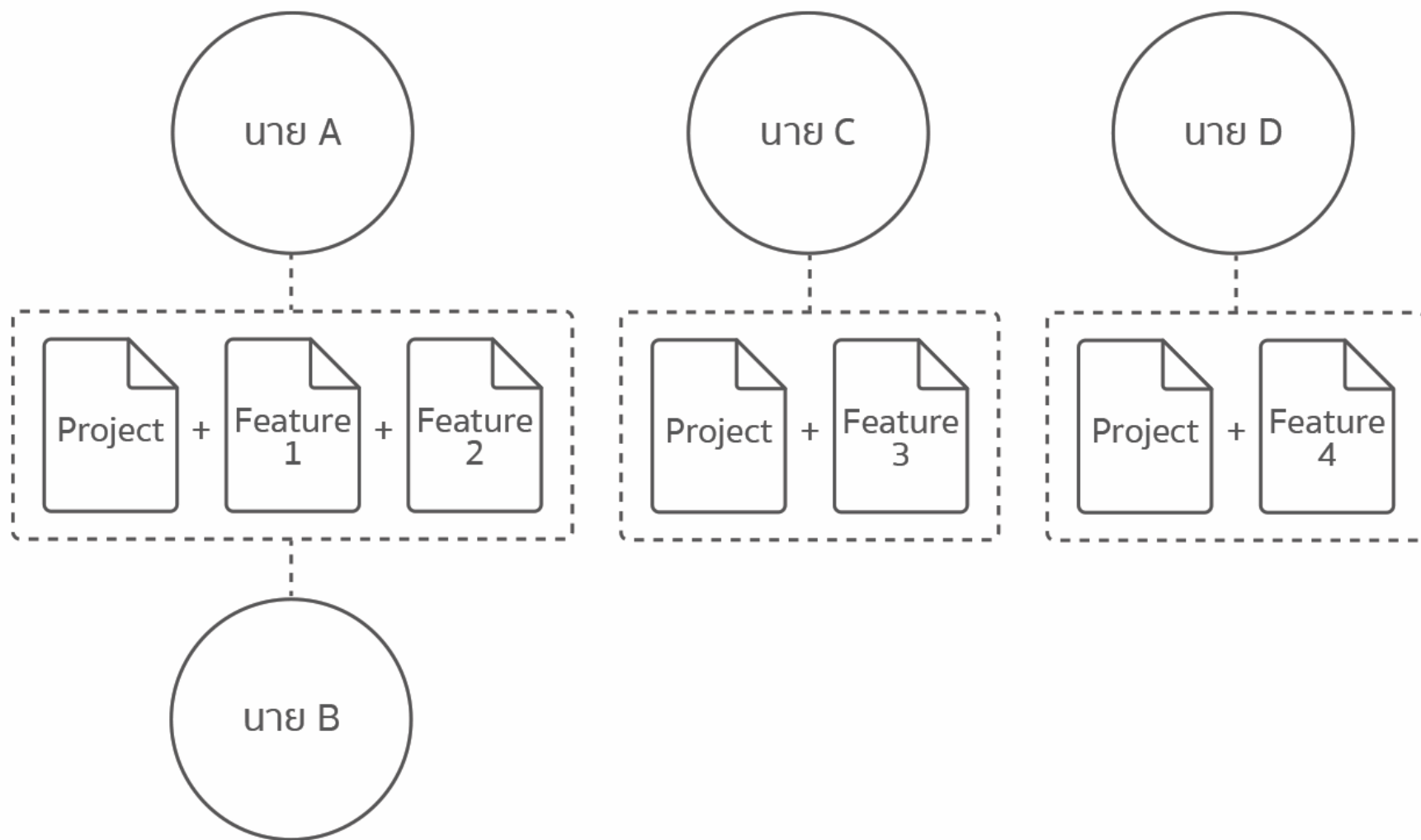
- แบบกระจายศูนย์ เช่น Git
- Copy มาทั้ง Repository
- Server เสีย Client ก็ยังทำงานบน Local ได้
- ทุกครั้งที่ Checkout คือ การ Backup file

Why Version Control ?





...

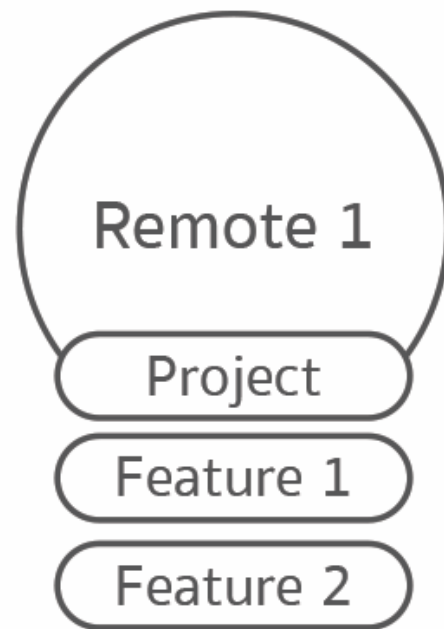




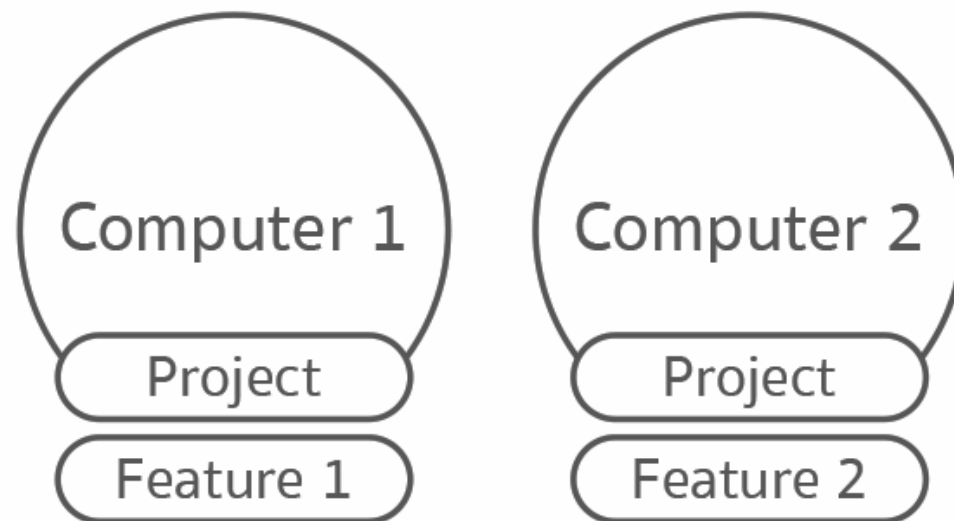
git



Server

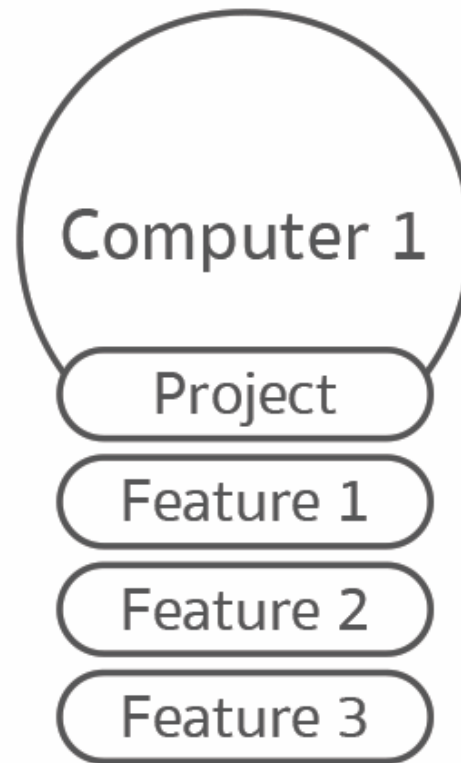


Client



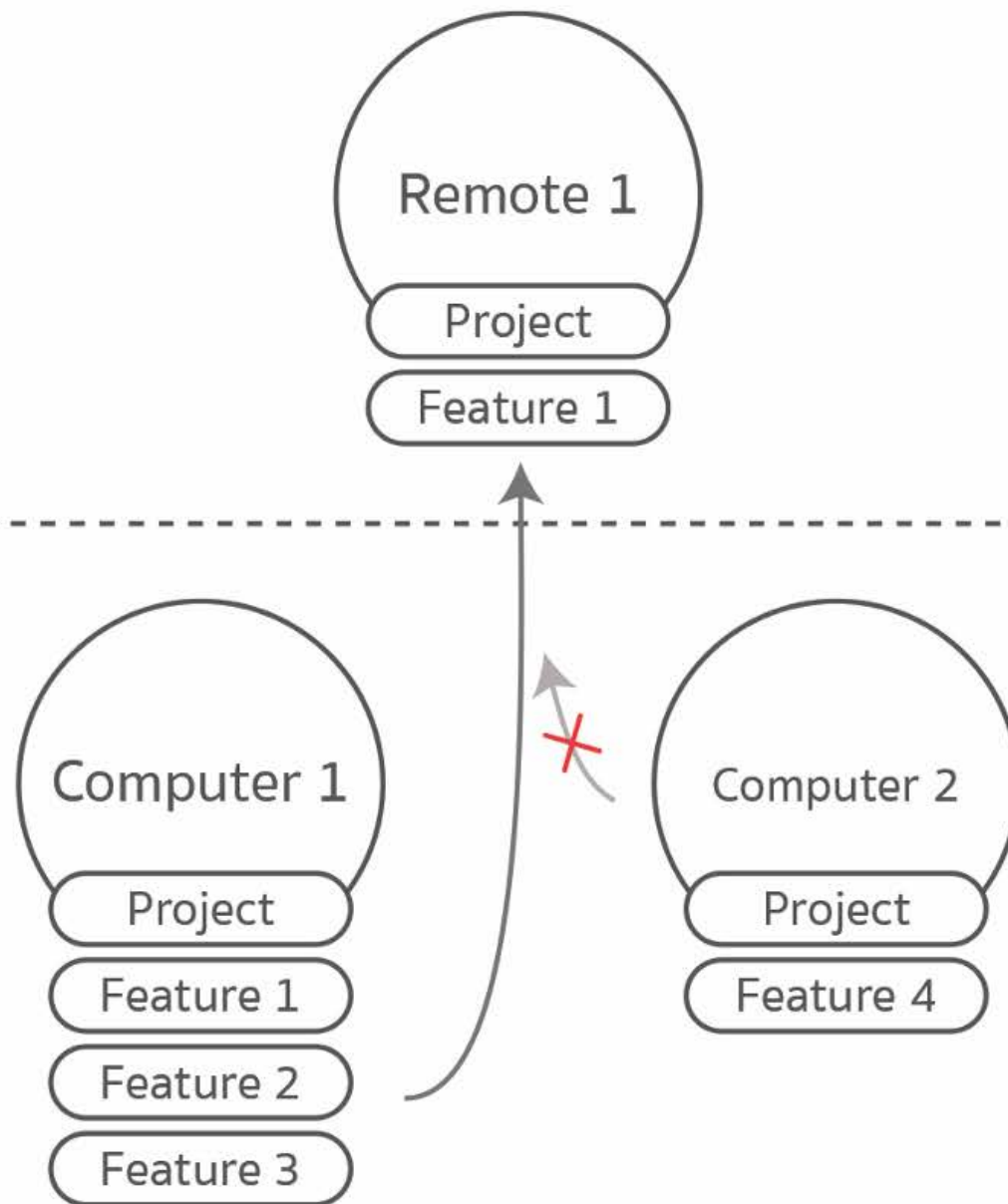
Server

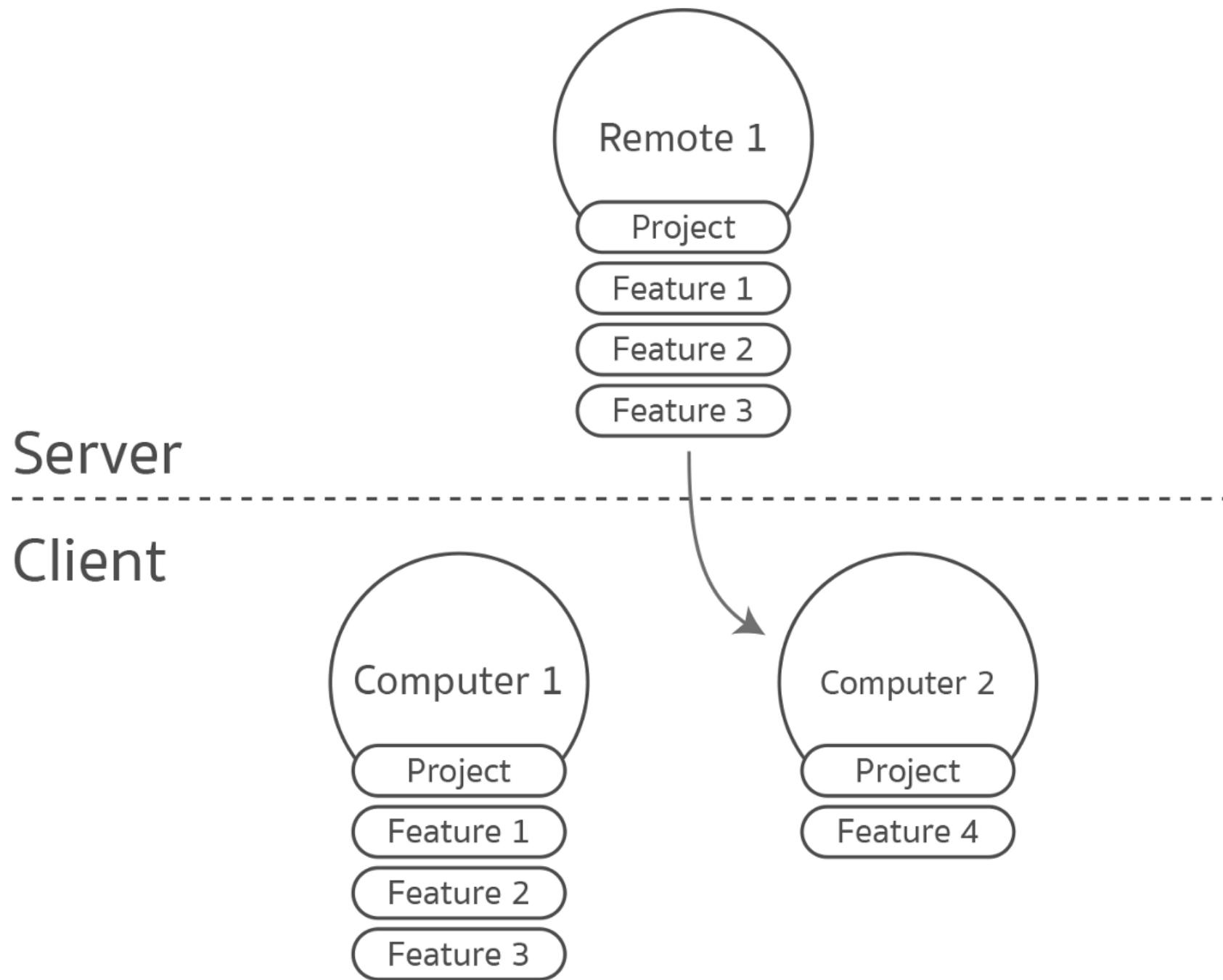
Client



Server

Client









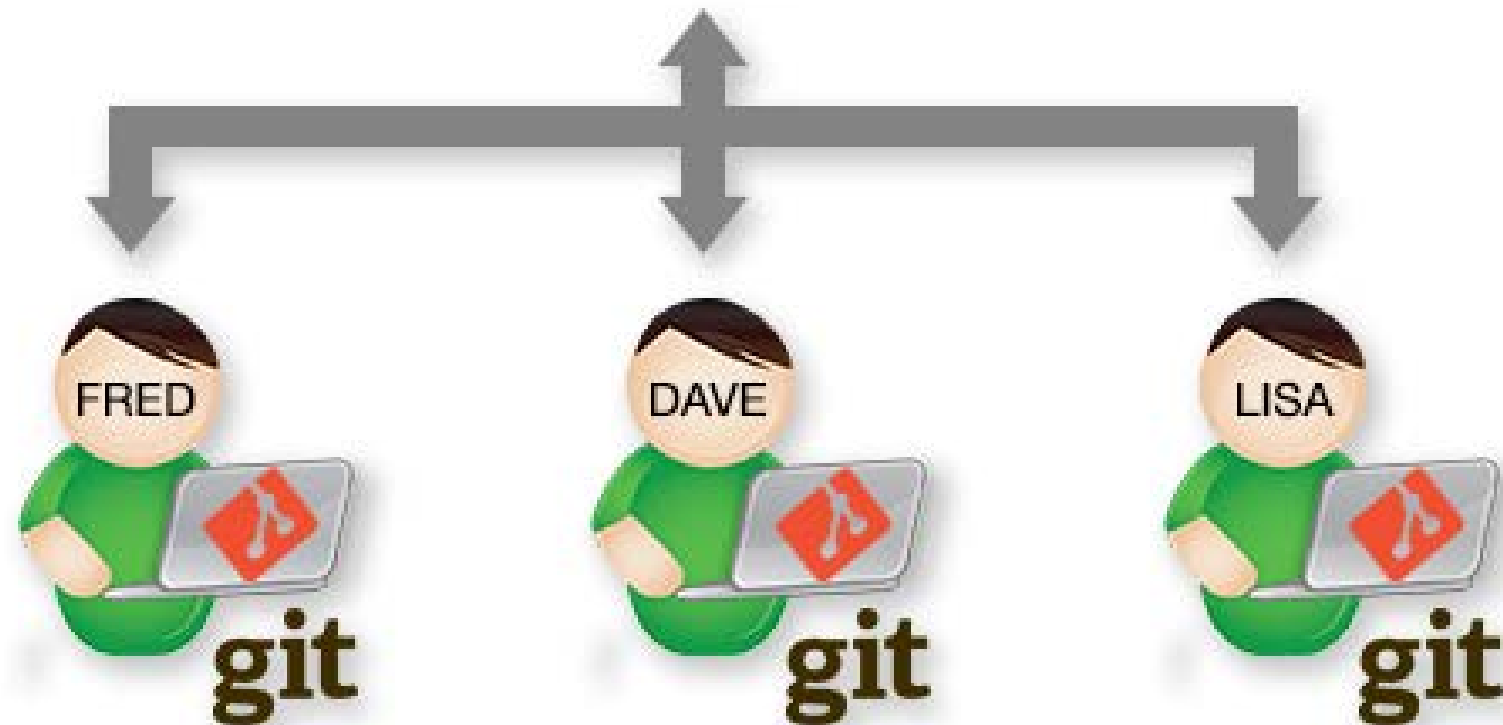
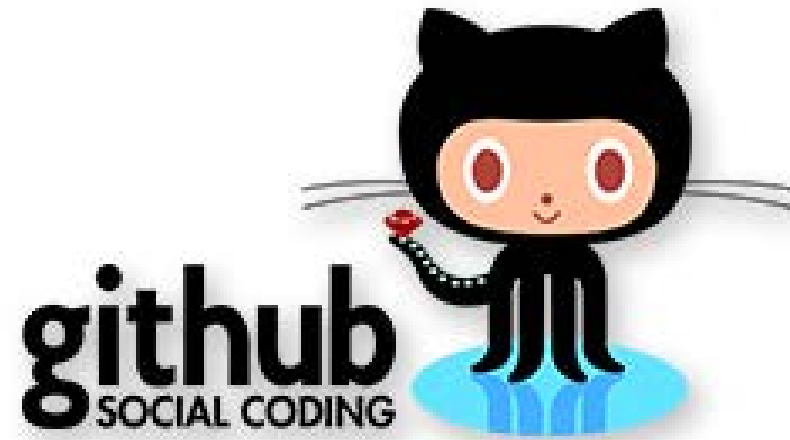


git



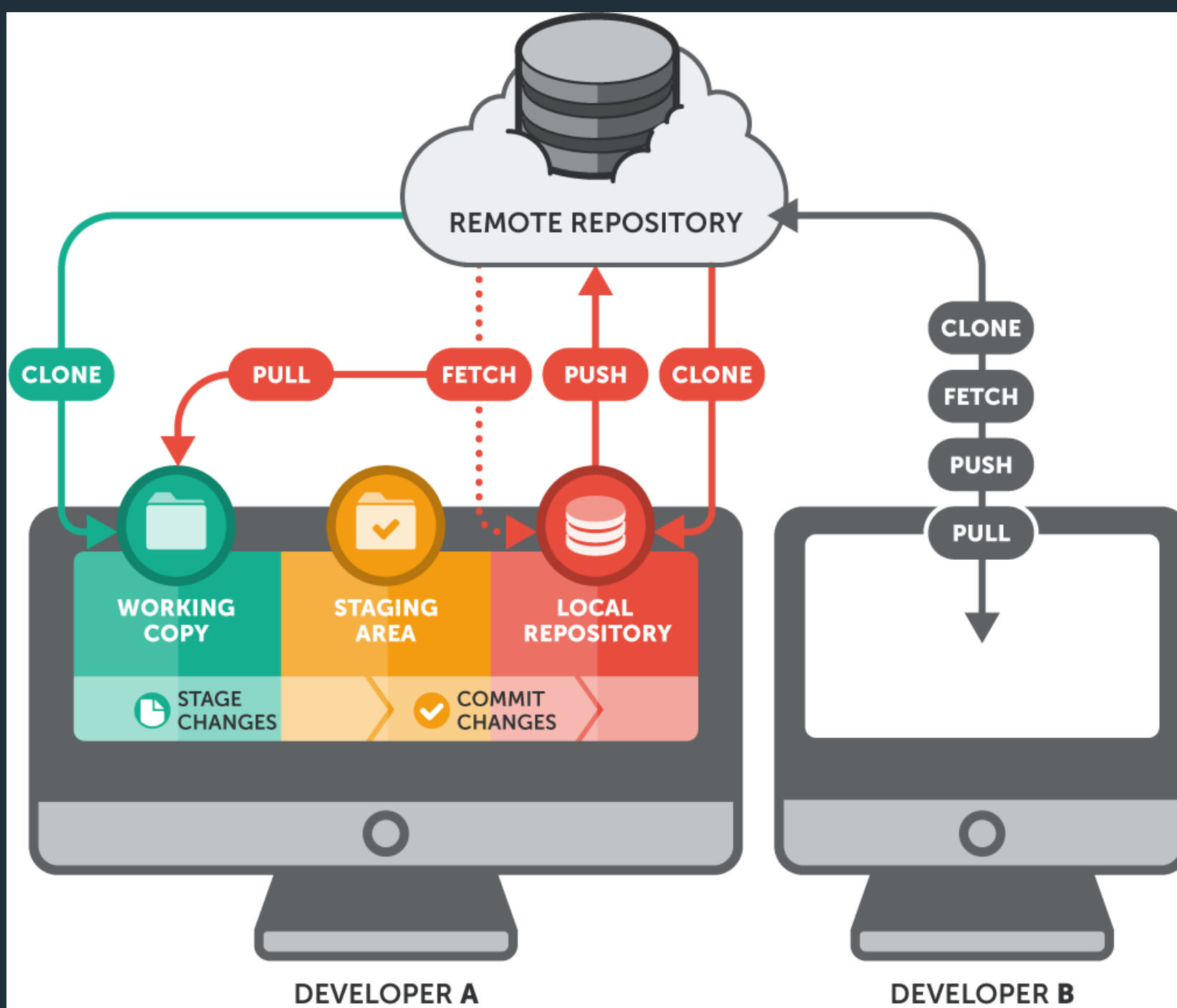
github
SOCIAL CODING

Whats the difference
between Git & Github ?



Keywords of Git

- Repository
- Clone
- Commit
- Unstaged
- Staged
- Push
- Pull
- Fetch
- Conflict
- Merge Commit



Repository

- Folder ที่เก็บ Project ของงานนั้นๆ
- 1 Repository สามารถเก็บ Project เท่าไรก็ได้ตามที่ต้องการ
- ส่วนใหญ่ก็นิยมเก็บ Project 1 ตัวต่อ 1 Repository

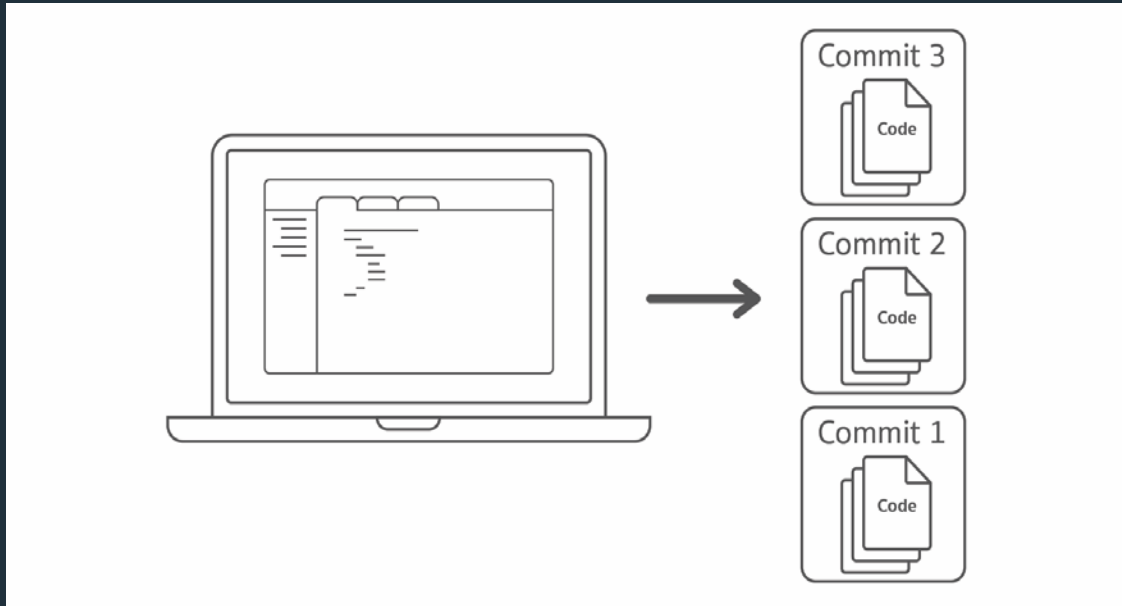


Clone

- คือ การ Sync Code มาลงเครื่องของเรา
- เรียกว่า *Clone* Repository
- หรือการ Copy Repository จาก Remote มาลงเครื่อง



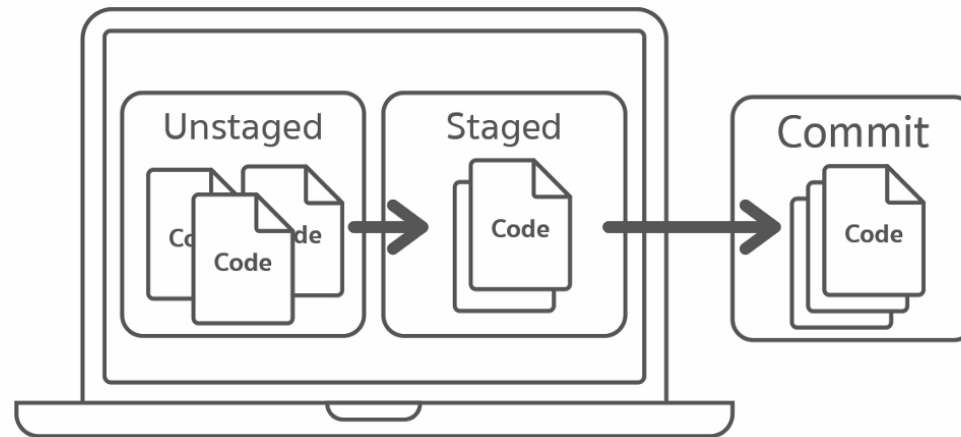
Commit



- การ Backup เก็บไว้ใน VCS จะเรียกกันว่า *Commit*
- การ Commit จะสามารถเลือกได้ว่าจะเอาไฟล์ไหนบ้าง (ไม่จำเป็นต้องเลือกทุกไฟล์)
- แต่ละ Commit จะไม่มีไฟล์ข้อมูลฉบับเต็ม
- แต่ละครั้งที่ทำการ Commit จะจำแนกว่ามีตรงไหนของข้อมูลที่ถูกเปลี่ยนแปลงไป
- สามารถย้อนดู History ได้ว่ามีการแก้ไขอะไรบ้าง ทำให้รู้ว่าใน Commit นั้นๆ แต่ละไฟล์มีข้อมูลเป็นอย่างไร

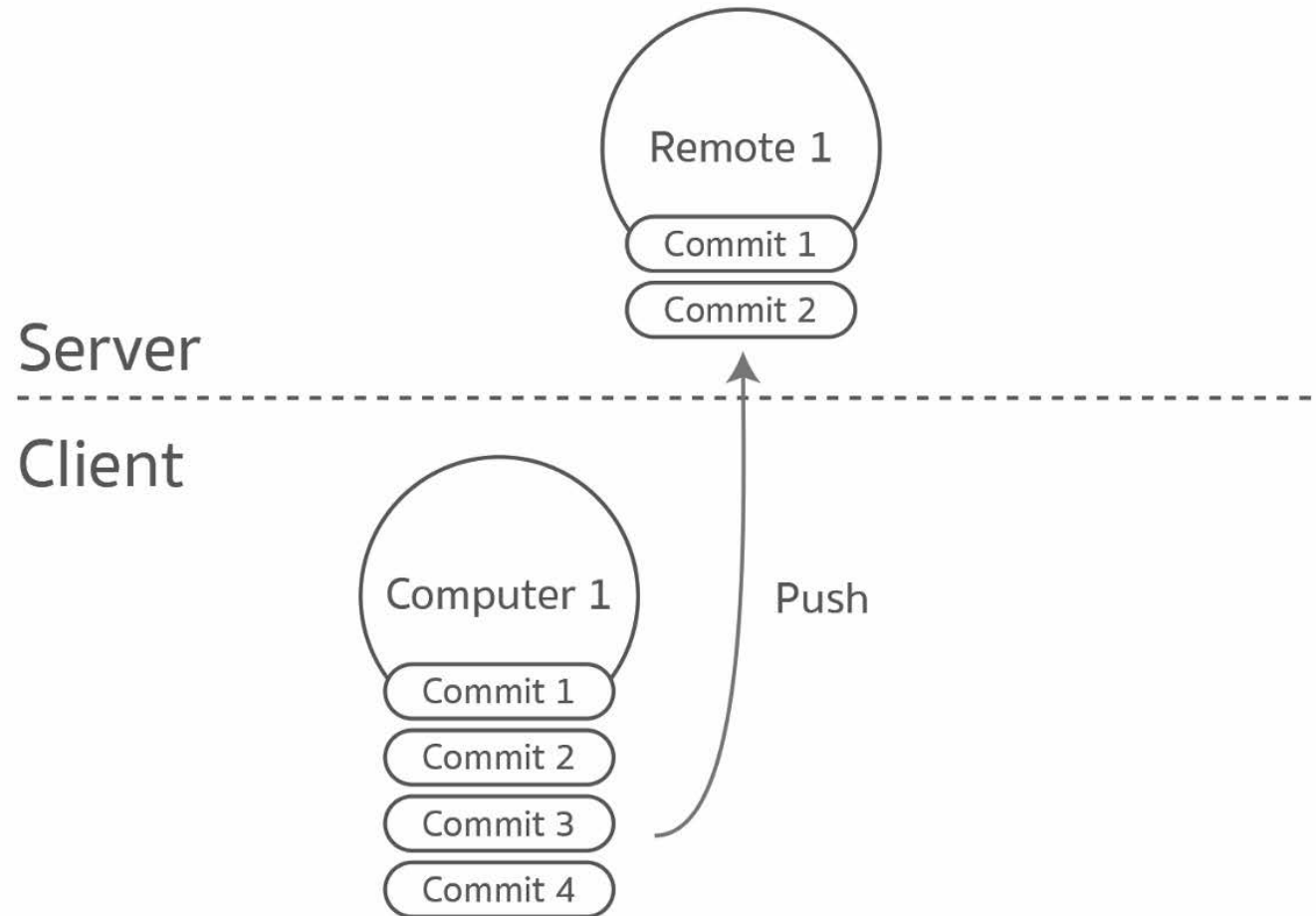
Unstaged & Staged

- ไฟล์ที่ถูกแก้ไขจะอยู่ในสถานะ *Unstaged*
- จะต้องเลือกไฟล์ที่ต้องการเพื่อย้ายเข้าสู่สถานะ *Staged* ก่อน
ถึงจะทำการ Commit ได้
- Unstaged และ Staged ทำให้เราสามารถเลือกเฉพาะบางไฟล์
สำหรับ Commit ได้



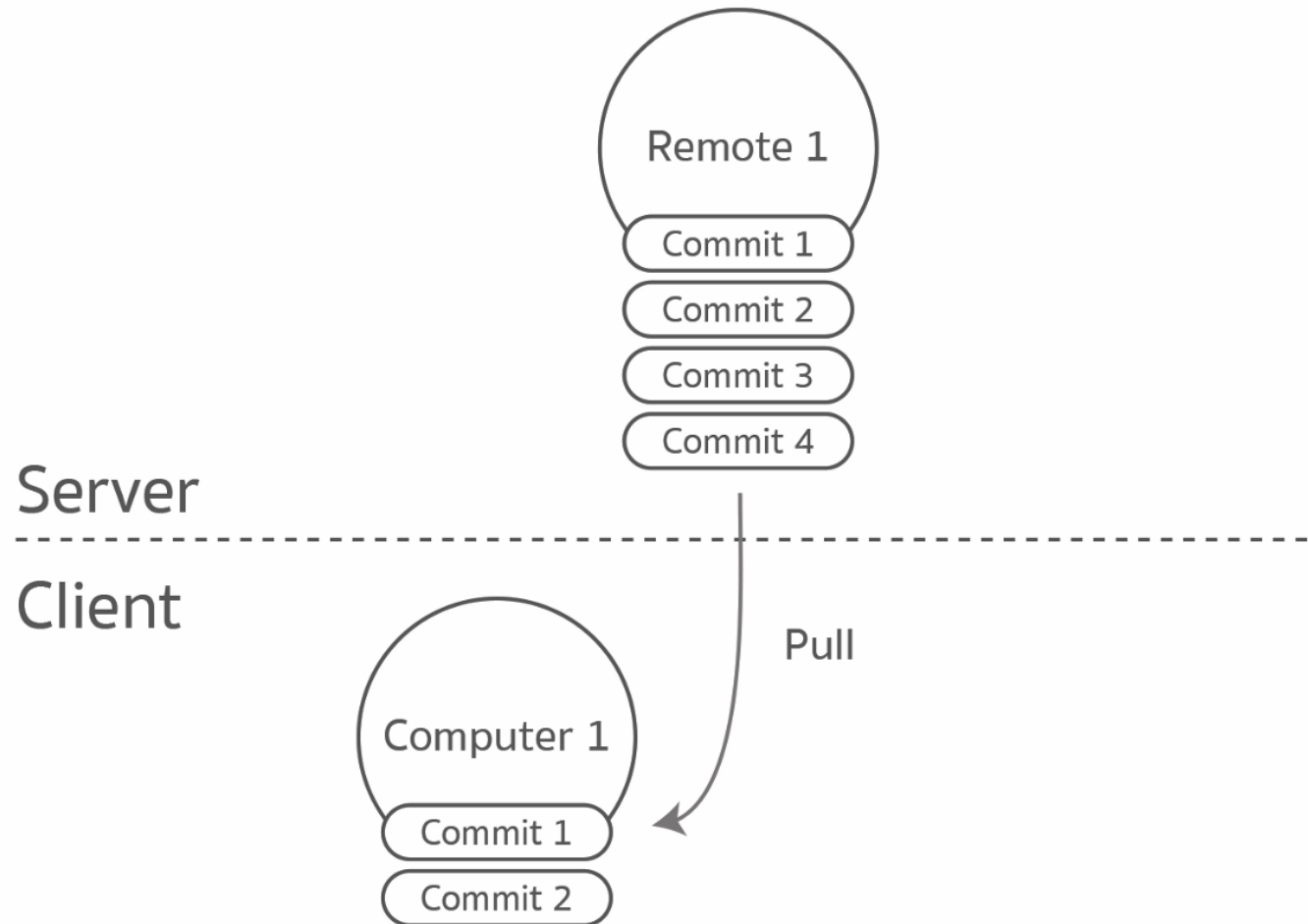
Push

- เวลาที่มี Commit อยู่ในเครื่องและต้องการจะ Sync ขึ้นไปเก็บไว้ใน Remote จะเรียกขั้นตอนนี้ว่า *Push*



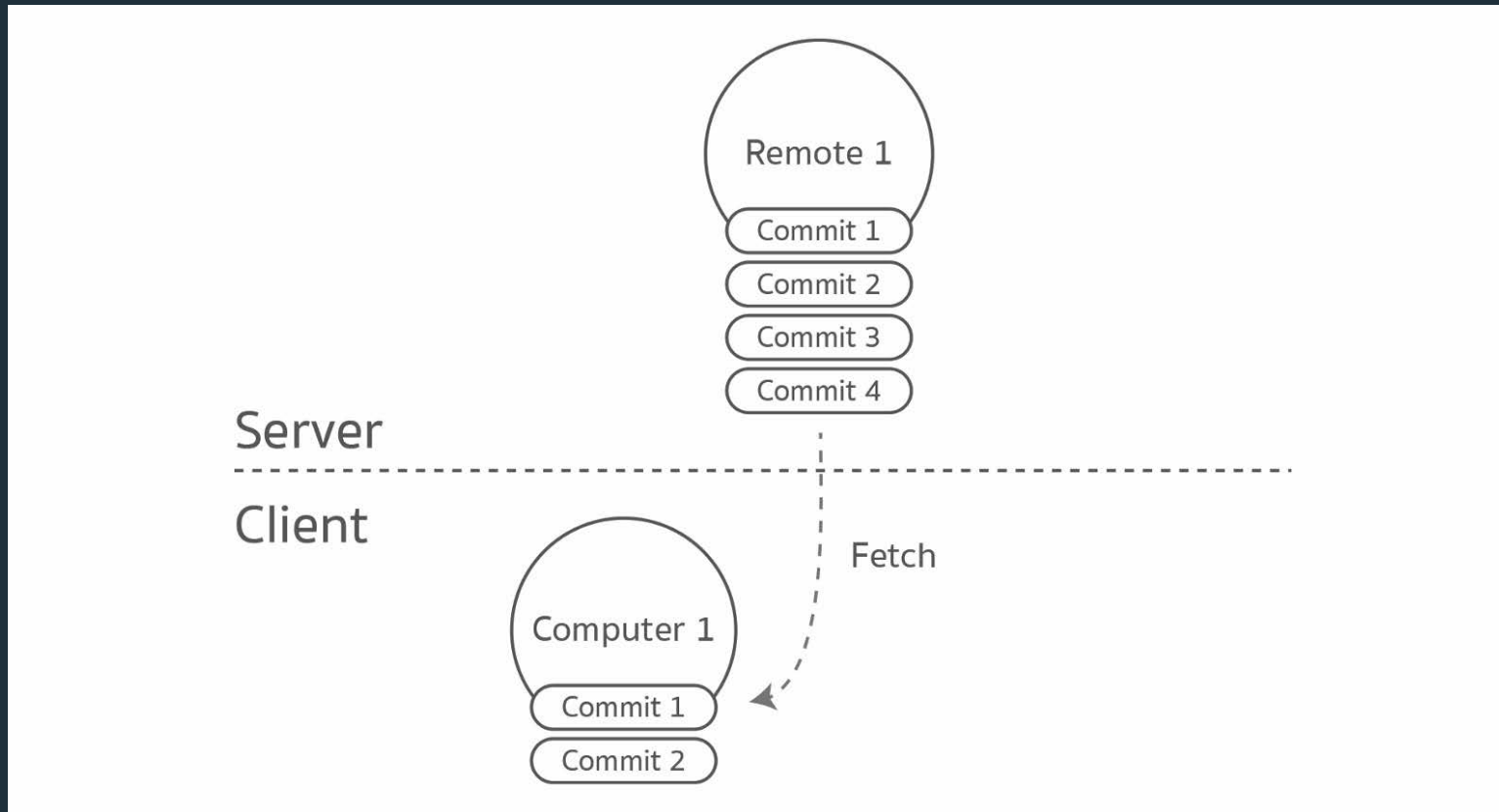
Pull

- เวลา Sync จาก Remote เพื่อดึงข้อมูล Commit ใหม่ ๆ ลงมาเก็บไว้ในเครื่องจะเรียกขั้นตอนนี้ว่า *Pull*

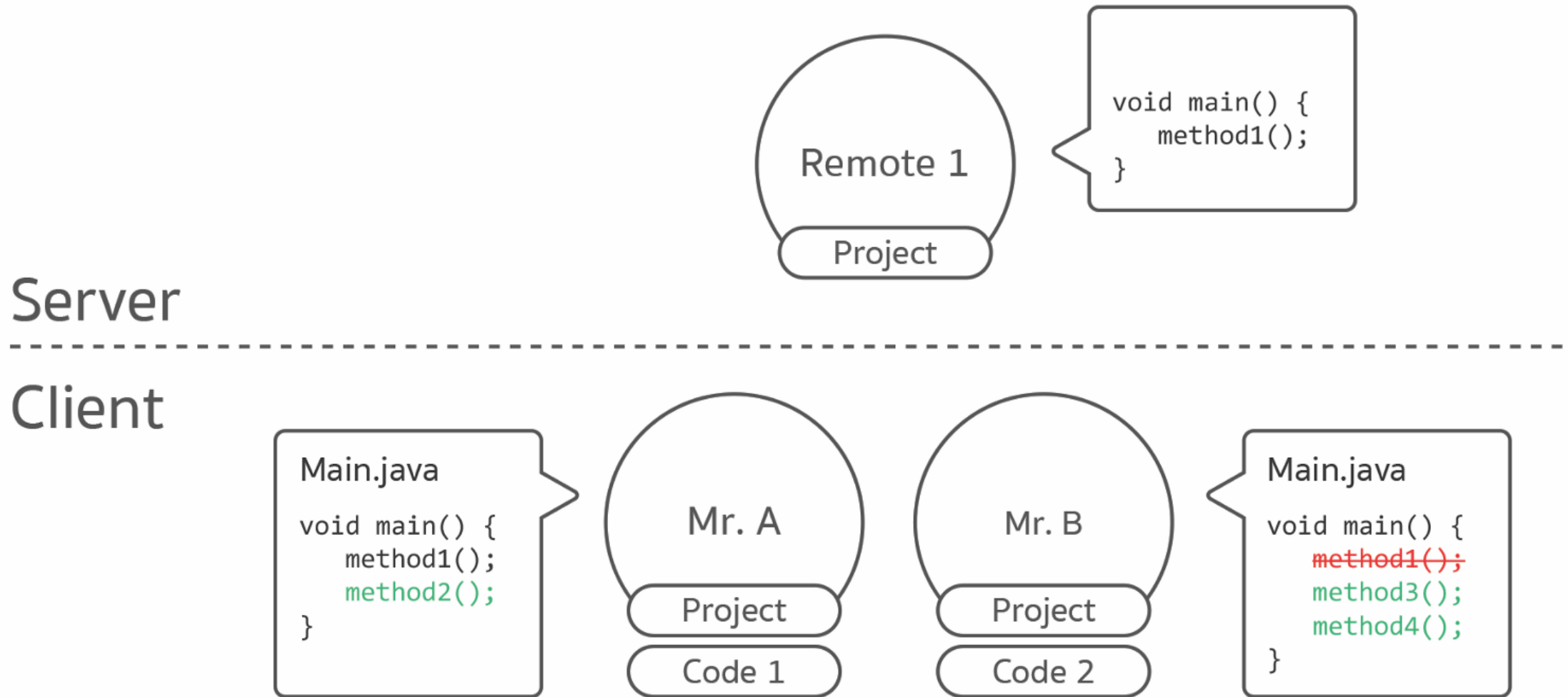


Fetch

- บางครั้ง อาจจะไม่ต้องการ Pull ข้อมูลลงมาเก็บไว้ในเครื่องทันที แค่อยากเช็คสถานะของ Remote ว่ามีใคร Push ข้อมูลใหม่ขึ้นไปที่ Remote หรือไม่ เราเรียกรูปแบบนี้ว่า *Fetch*
- การ Fetch ทำให้สามารถอัปเดตและดู History ทั้งหมดที่อยู่บน Remote ได้ โดยไม่ต้องดึงข้อมูลลงมา
- การ Fetch จะถูกเรียกทุกครั้งที่ทำกร Pull

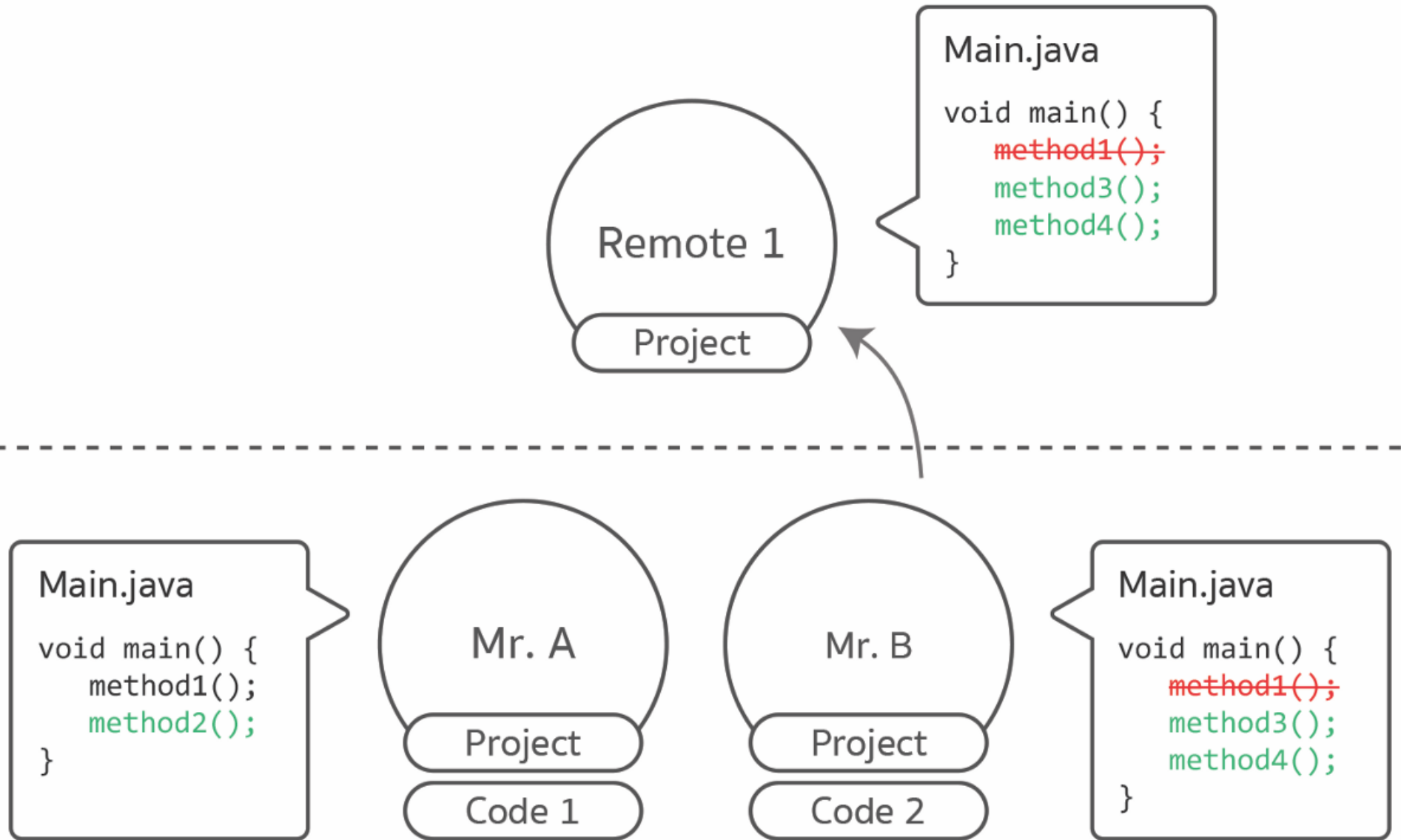


Merge Commit



Server

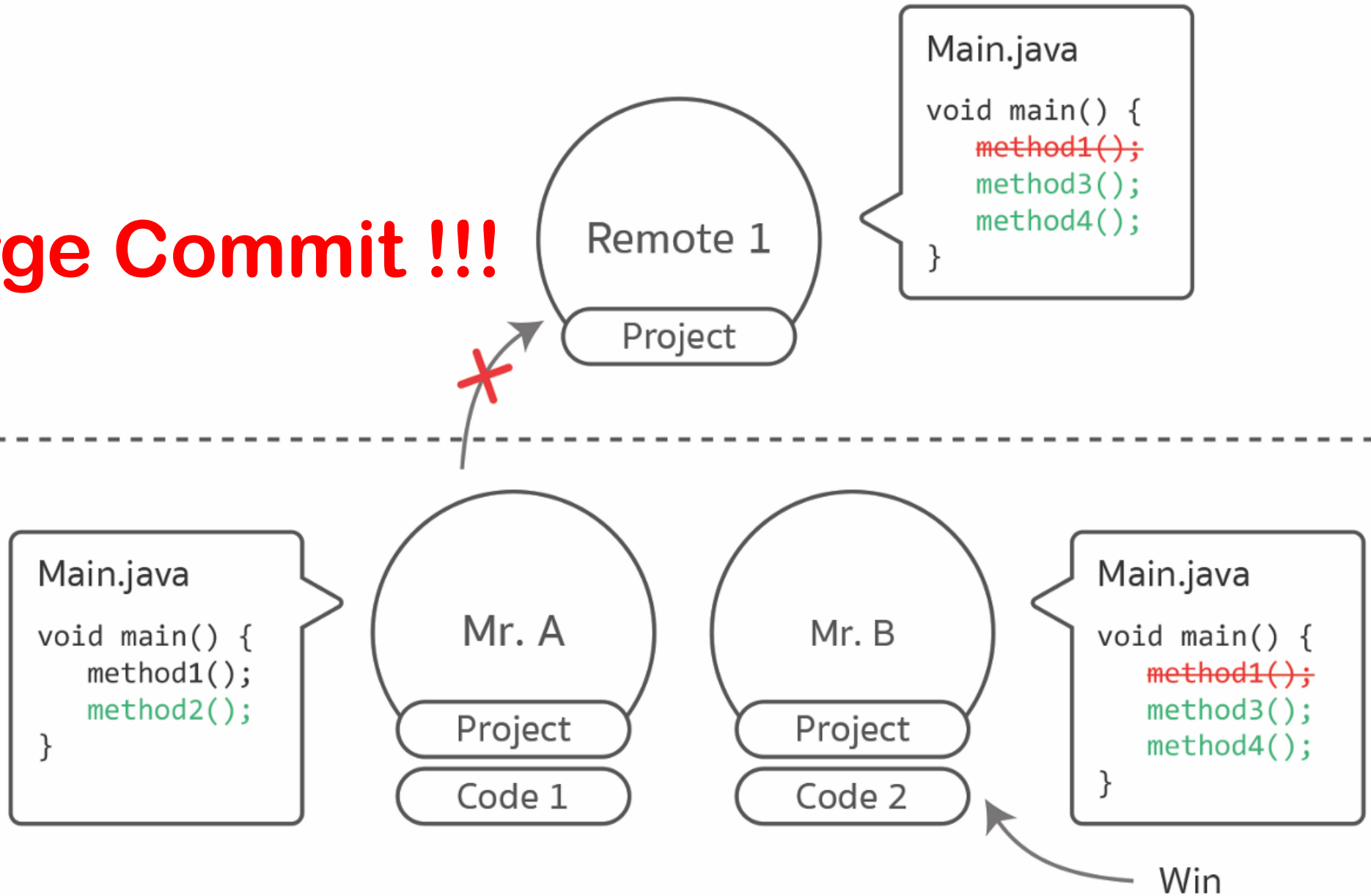
Client



Merge Commit !!!

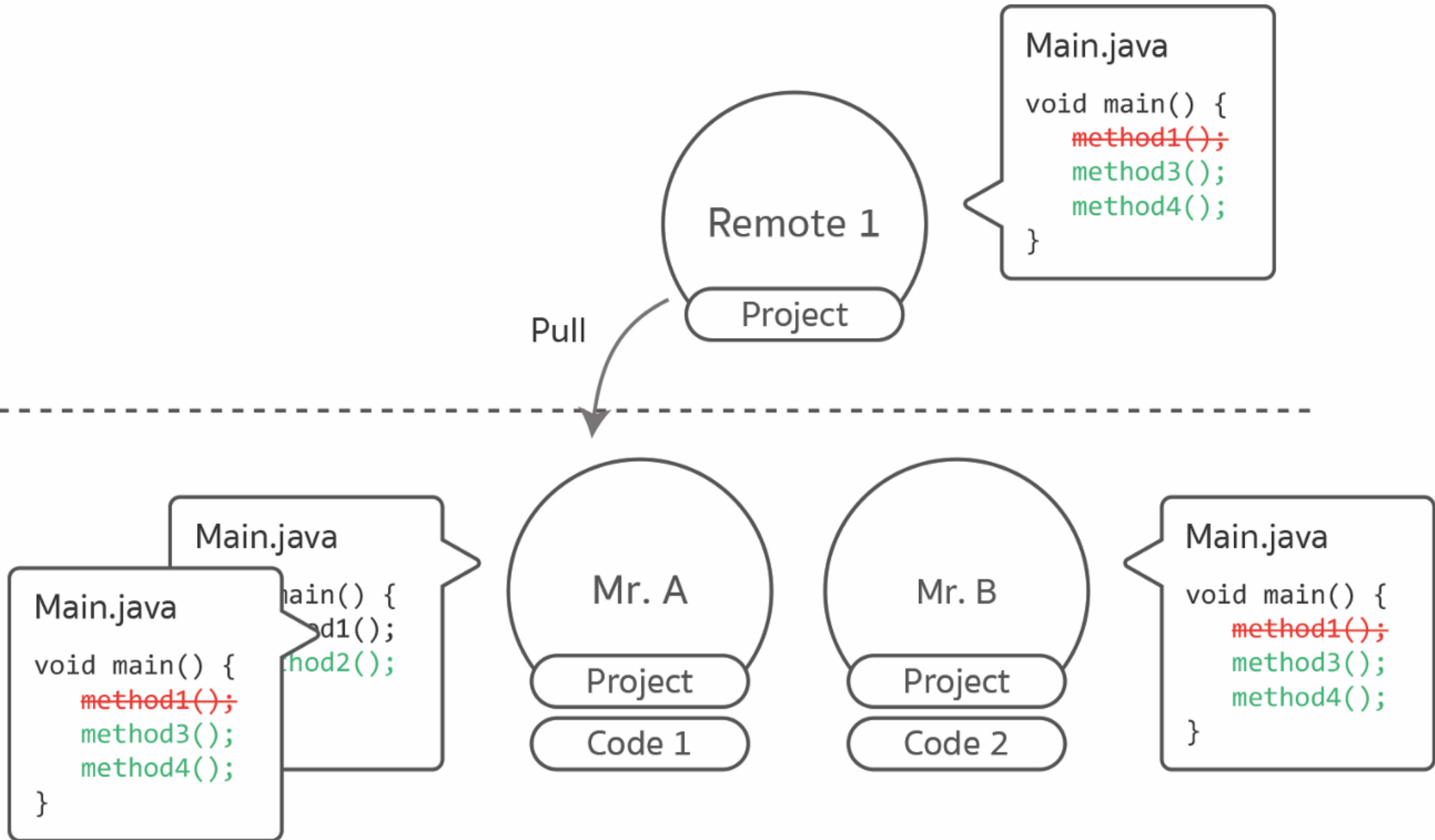
Server

Client



Server

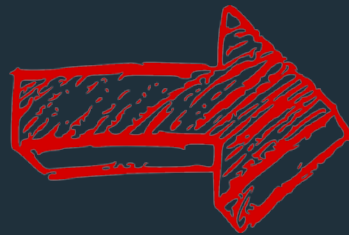
Client




```
// โค้ดเดิม
void main() {
    method1();
}
```

```
// โค้ดที่นาย A แก้ไข
void main() {
    method2();
    method1();
}
```

```
// โค้ดที่นาย B แก้ไข
void main() {
    method1();
    method3();
}
```



```
void main() {
    method2();    // โค้ดของนาย A
    method1();
    method3();    // โค้ดของนาย B
}
```

- กรณีที่โค้ดสามารถ Merge รวมกันได้โดยไม่มีปัญหา เมื่อ Merge เสร็จ ก็จะกลายเป็น Commit ตัวหนึ่ง สำหรับ Push ขึ้น Remote

แต่ทว่าโค้ดที่นาย A กับนาย B
เขียนมันไม่ใช่แบบนี้ นะสิ !!!

Conflict

Main.java

```
void main() {  
    method1();  
    method3();  
    method4();  
}
```

B

Main.java

```
void main() {  
    method1();  
    method2();  
}
```

A

```

void main() {
<<<<<<< HEAD
    method1();
    method2();
=====
    method3();
    method4();
>>>>>>> 322d39a003e4d9...
}

```

- โค้ดที่อยู่ระหว่าง <<< และ === คือโค้ดของนาย A
- โค้ดที่อยู่ระหว่าง === และ >>> คือโค้ดของนาย B
ส่วนตัวเลขต่อท้ายคือหมายเลขของ Commit ที่ทำการ Merge



```

void main() {
    method2();
    method3();
    method4();
}

```

Question ?



Homework ?

Homework

1. Email : Name.2Surname@gmail.com : Chalach.mo@gmail.com
2. Join Facebook Group : R e a c t 2 / 2 5 6 0
3. Register Github
 - Clone : <https://github.com/Kzis/React-Course.git>
 - Commit your file for introduce yourself(World or PDF Only)
 - Push to github
4. Install Program (VSCode , Cmder , SourecTree , Node.js , Git)

Reference

- <https://blog.nextzy.me/มาเรียนรู้-git-แบบง่ายๆกันเถอะ-427398e62f82>
- <https://www.atlassian.com/git/tutorials/what-is-git>