Git

คำสั่งที่ใช้บ่อยในงานต่าง ๆ

คำสั่งพื้นฐาน

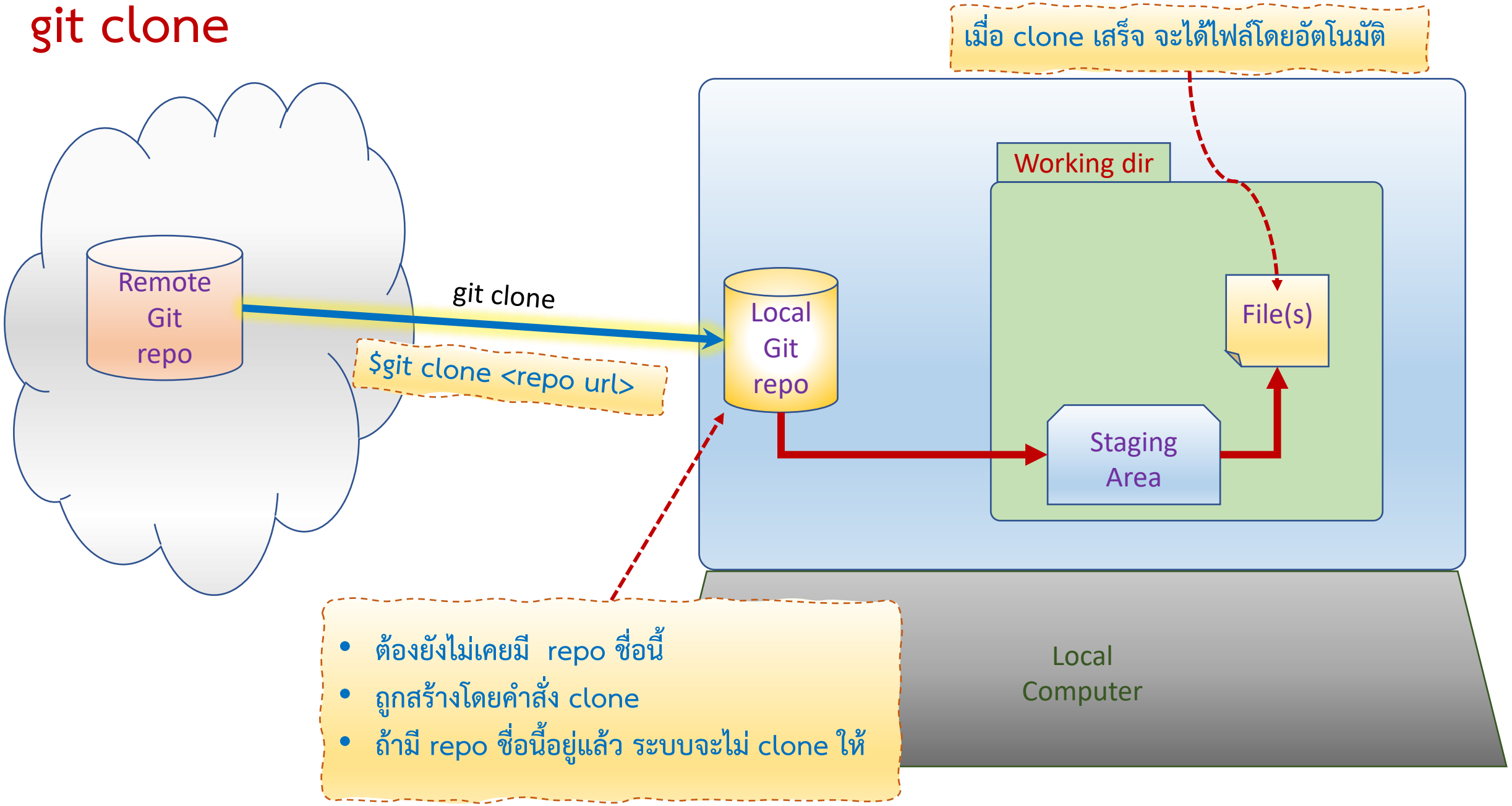
คำสั่ง git ที่ใช้งานบ่อย

- git clone
  - สำเนา remote (origin) repository จาก server มายังเครื่อง local repository แล้วแตกไฟล์ต่างๆ ลงใน working directory ให้พร้อมใช้งาน
- git pull
  - ดึง contents ล่าสุดของ branch ปัจจุบัน จาก remote (origin) repository
- git status
  - ตรวจสอบสถานะของไฟล์ใน git (working directory, staging area, local repo)

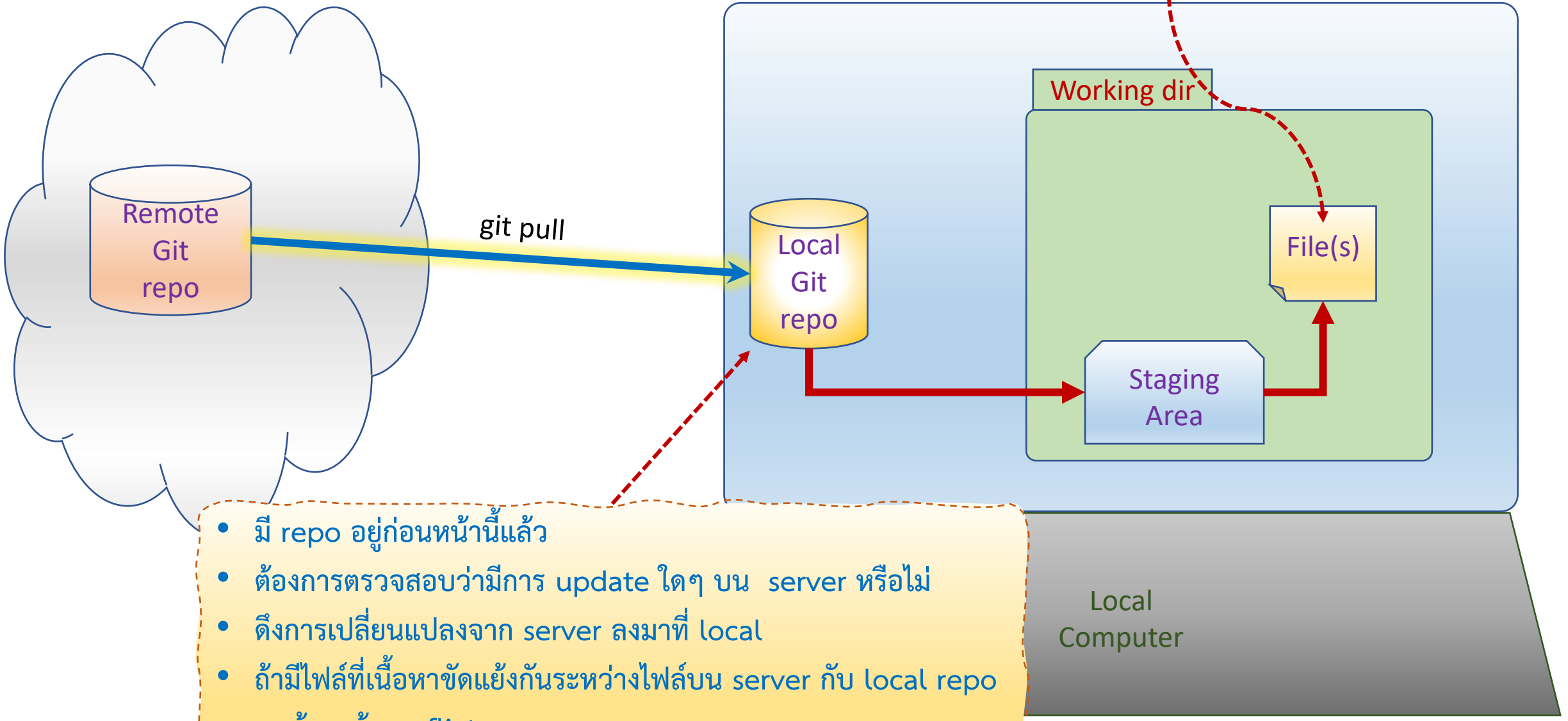
คำสั่ง git ที่ใช้งานบ่อย

- git add <file(s)>
  - เพิ่มไฟล์ <file(s)> เข้าไปยัง staging areas, ใช้ git add . เพื่อเพิ่มไฟล์ทั้งหมดใน directory
- git commit –m “<commit messages>”
  - ส่งไฟล์จาก staging area เข้าไปยัง local repo พร้อมคำอธิบายเหตุผลในการแก้ไขไฟล์
- git push
  - Sync การเปลี่ยนแปลงใน local repo ไปยัง remote repository

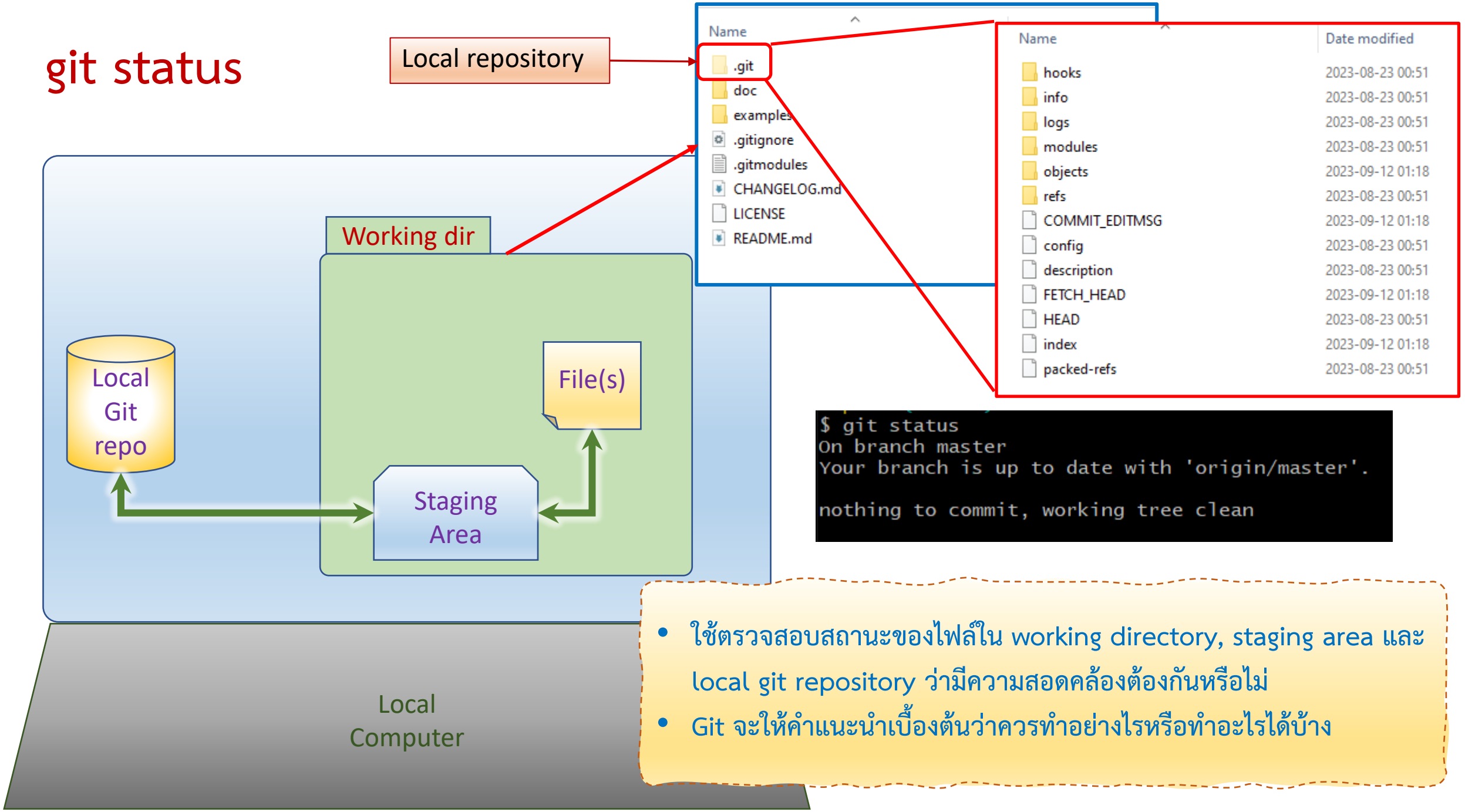
git clone


เมื่อ clone เสร็จ จะได้ไฟล์โดยอัตโนมัติ
Remote
Git
repo
git clone
$git clone <repo url>
Local
Git
repo
Working dir
File(s)
Staging
Area
Local
Computer
ต้องยังไม่เคยมี repo ชื่อนี้
ถูกสร้างโดยคำสั่ง clone
ถ้ามี repo ชื่อนี้อยู่แล้ว ระบบจะไม่ clone ให้

git pull


เมื่อ clone เสร็จ จะ update ไฟล์ใน branch ที่ทำงานอยู่โดยอัตโนมัติ
Working dir
Remote Git repo
git pull
Local Git repo
File(s)
Staging Area
Local Computer
มี repo อยู่ก่อนหน้านี้แล้ว
ต้องการตรวจสอบว่ามีการ update ใดๆ บน server หรือไม่
ดึงการเปลี่ยนแปลงจาก server ลงมาที่ local
ถ้ามีไฟล์ที่เนื้อหาขัดแย้งกันระหว่างไฟล์บน server กับ local repo เราต้องแก้ conflict เอง

git status

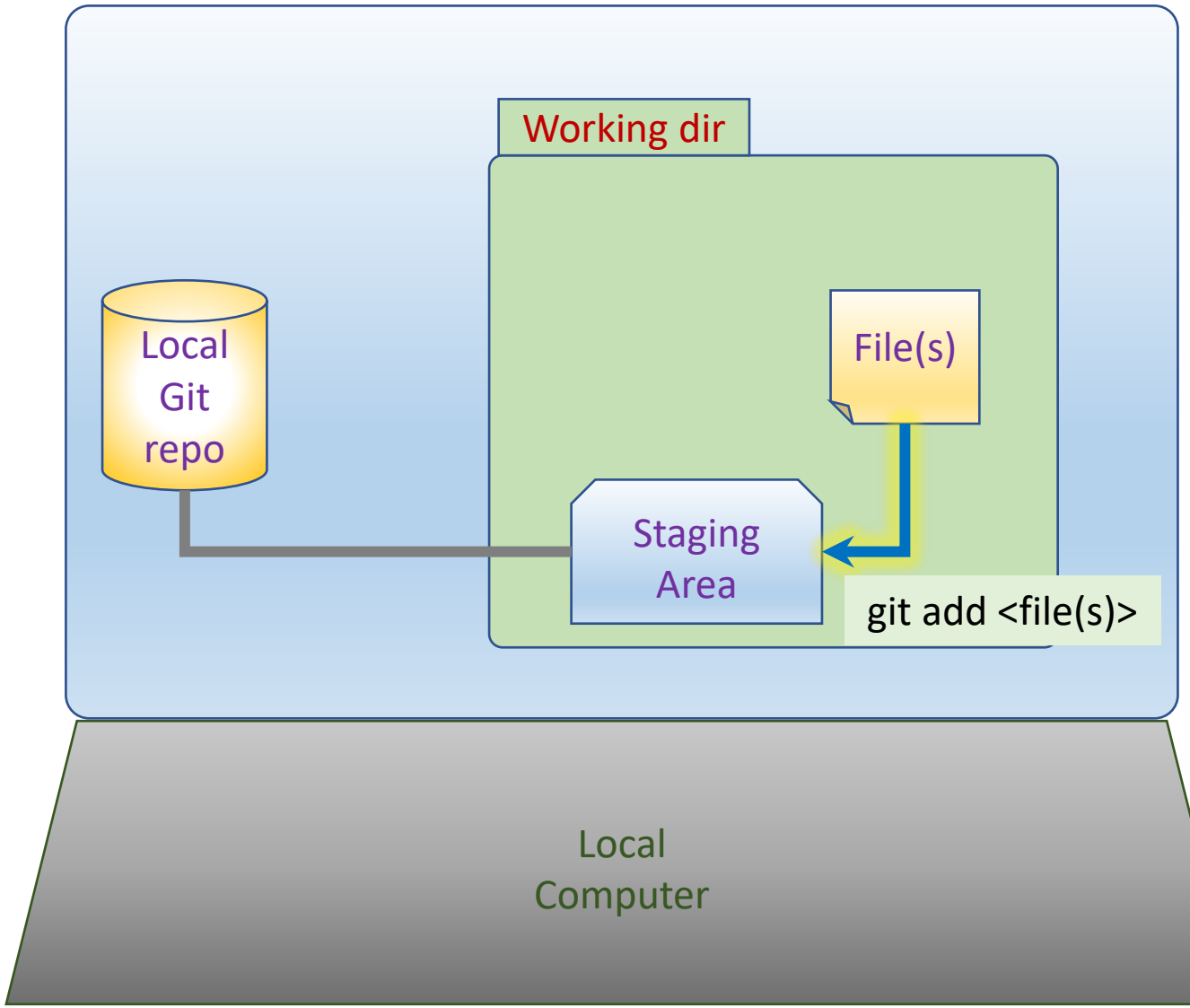



```
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

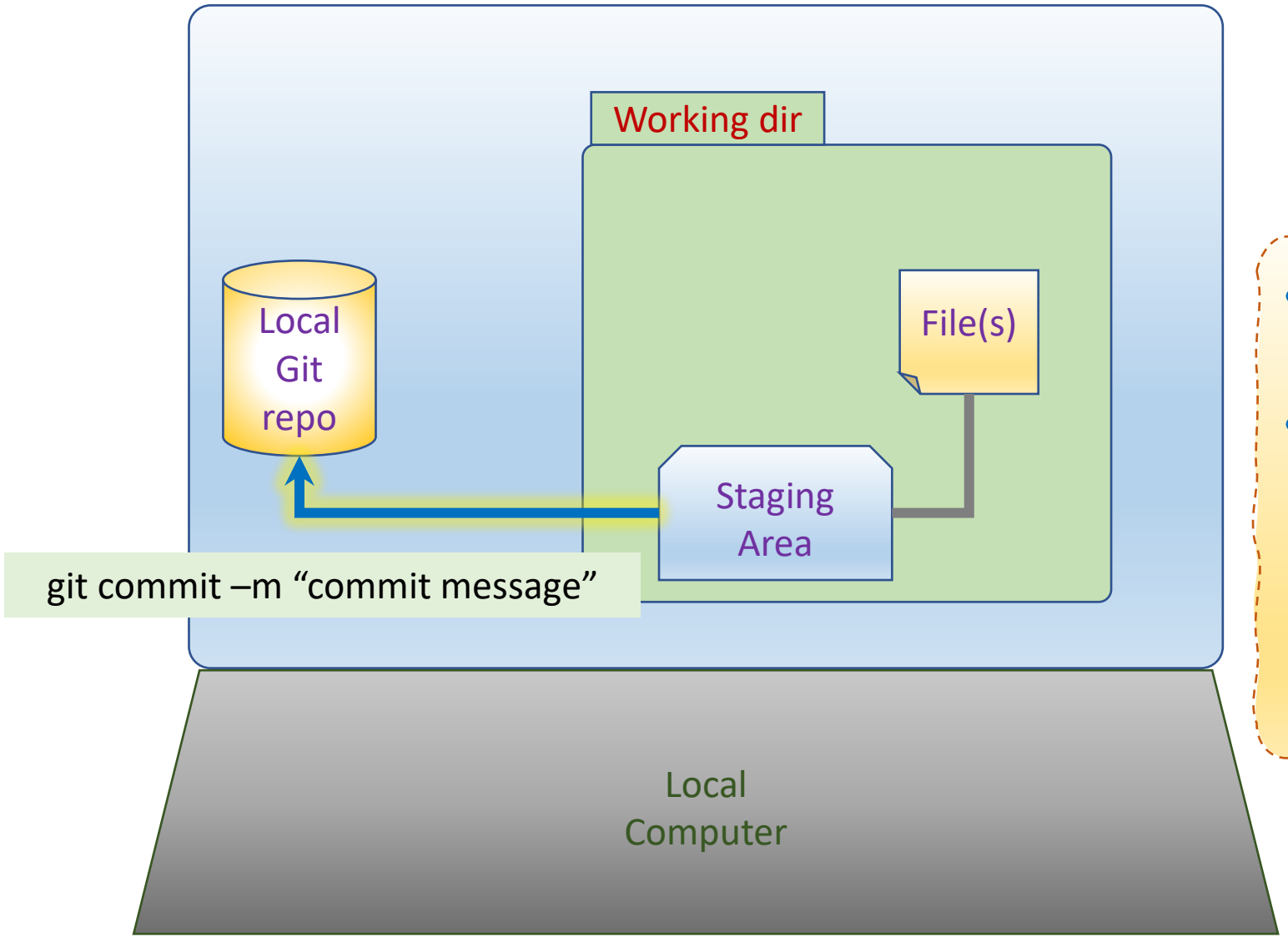
- ใช้ตรวจสอบสถานะของไฟล์ใน working directory, staging area และ local git repository ว่ามีความสอดคล้องต้องกันหรือไม่
- Git จะให้คำแนะนำเบื้องต้นว่าควรทำอย่างไรหรือทำอะไรได้บ้าง

git add



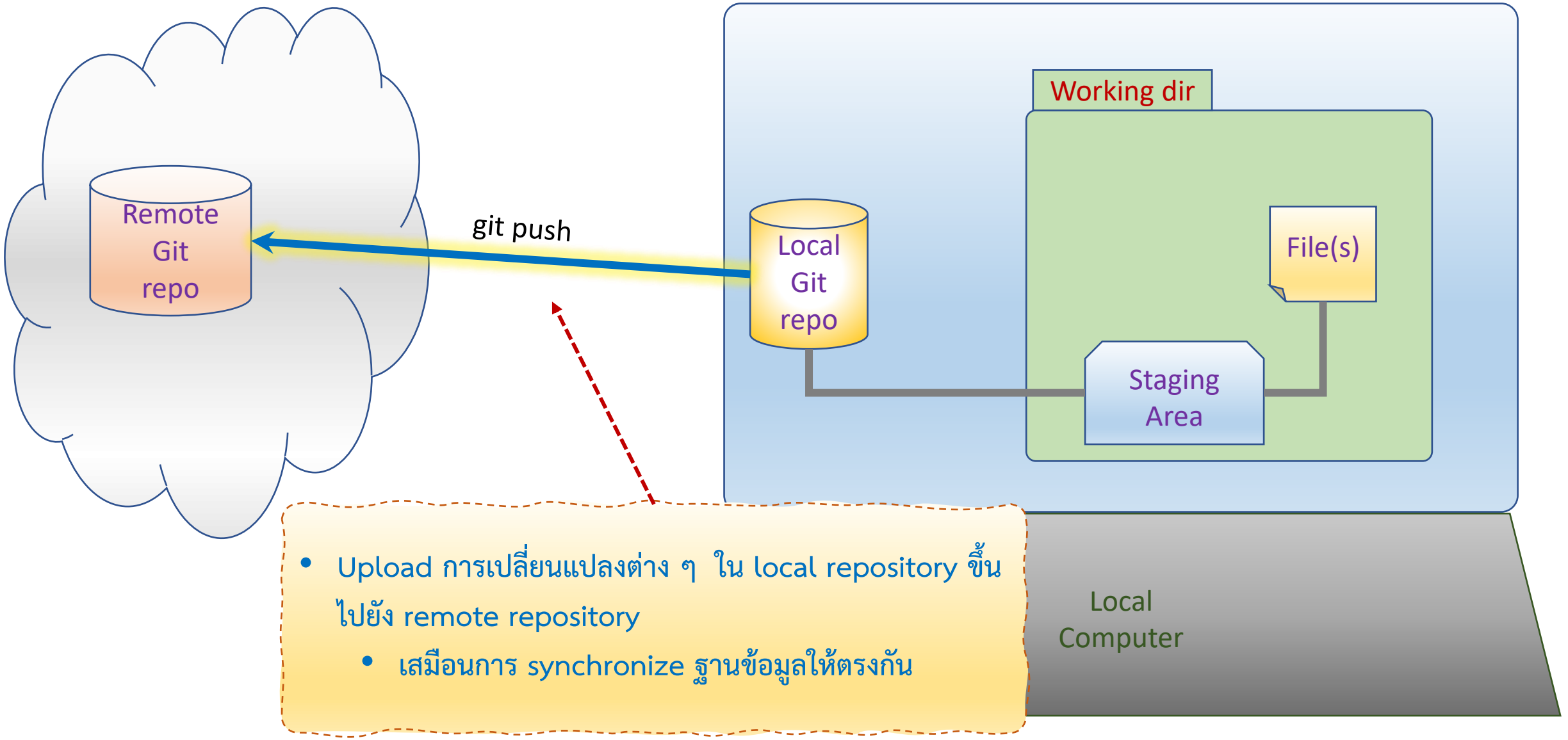
- ใช้เพิ่มไฟล์ที่จะให้ git ติดตามการเปลี่ยนแปลง
- Git แค่ทำการ tracking แต่จะไม่ commit
  - ถ้ามีการเปลี่ยนแปลงเนื้อหาโดยที่ไม่ commit เนื้อหานั้นก็จะไม่ถูกบันทึกใน local repository
  - ถ้าต้องการ add ไฟล์ทั้งหมด ใช้ git add .

git commit



- ใช้บันทึกการเปลี่ยนแปลงไฟล์ที่ถูกติดตามโดย git ลงใน local repository
- ต้องระบุเหตุผลในการ commit ทุกครั้ง
  - ให้ทีมงานเข้าใจว่าเราทำอะไรลงไป
  - ไว้ช่วยเตือนความจำ หรือค้นหาการเปลี่ยนแปลง
  - อาจจะ tag ด้วยข้อความที่ค้นหาได้ง่าย

git push



- Upload การเปลี่ยนแปลงต่าง ๆ ใน local repository ขึ้นไปยัง remote repository
  - เสมือนการ synchronize ฐานข้อมูลให้ตรงกัน

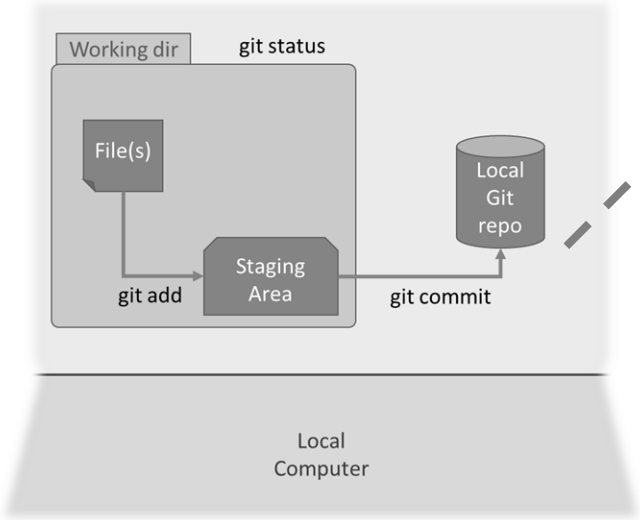
Git workflow ที่ใช้งานบ่อย

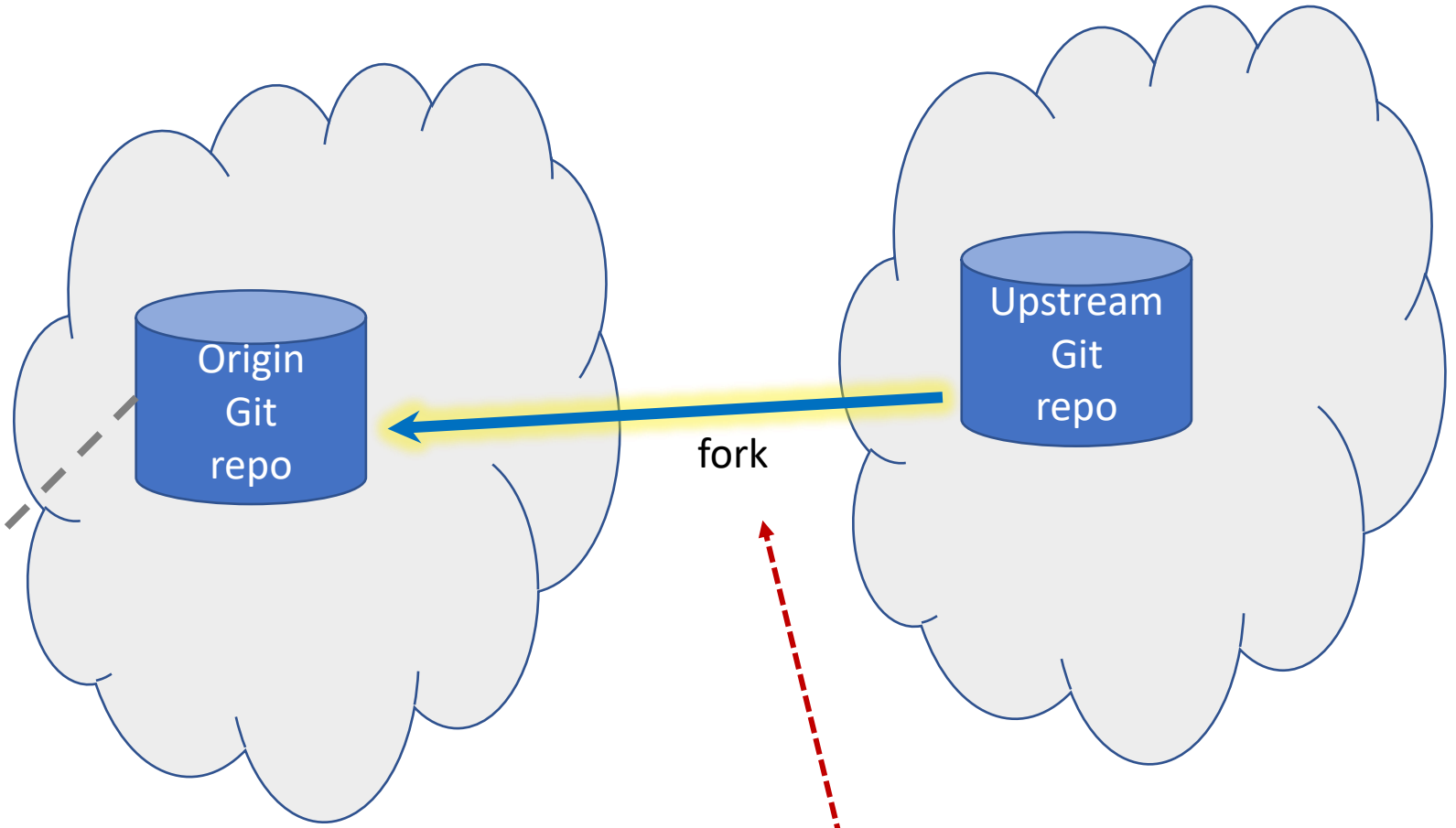
Git workflow ที่ใช้งานบ่อย ๆ

- Fork
  - Clone repository จาก user หรือ organization มาไว้ใน git account ส่วนตัว (บน server)
    - Repository ต้นทางเรียกว่า upstream repository, Repo ส่วนตัวเรียกว่า remote repository
- Pull request
  - แบ่งปัน (Contribute) งานจาก remote repository ไปยัง upstream repository
- Branch
  - ในกรณีที่เราต้องการแก้ไขงาน โดยที่ยังไม่ต้องการ release ใน mainstream (อยู่ระหว่างการพัฒนา) ก็สามารถแยกสาขาของไฟล์ออกมาทำงาน แล้วค่อยนำกลับไปรวมเมื่อพัฒนาเสร็จ

fork

- ในขณะที่ fork จะยังไม่เกิด local repository
- ต้องทำการ clone จาก remote repo ลงมาที่เครื่อง local computer

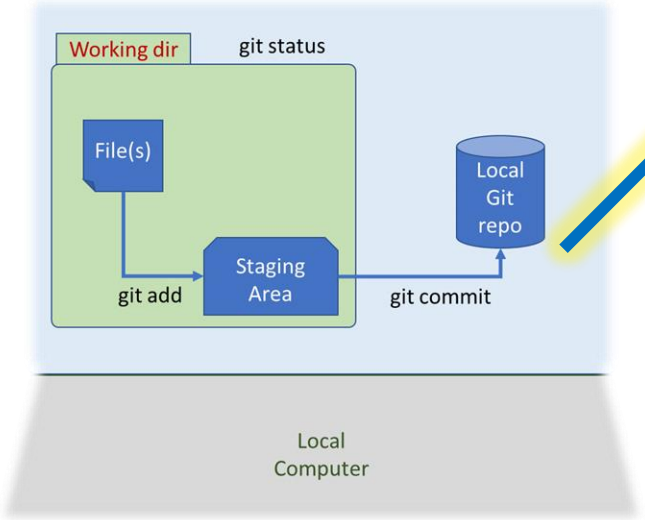


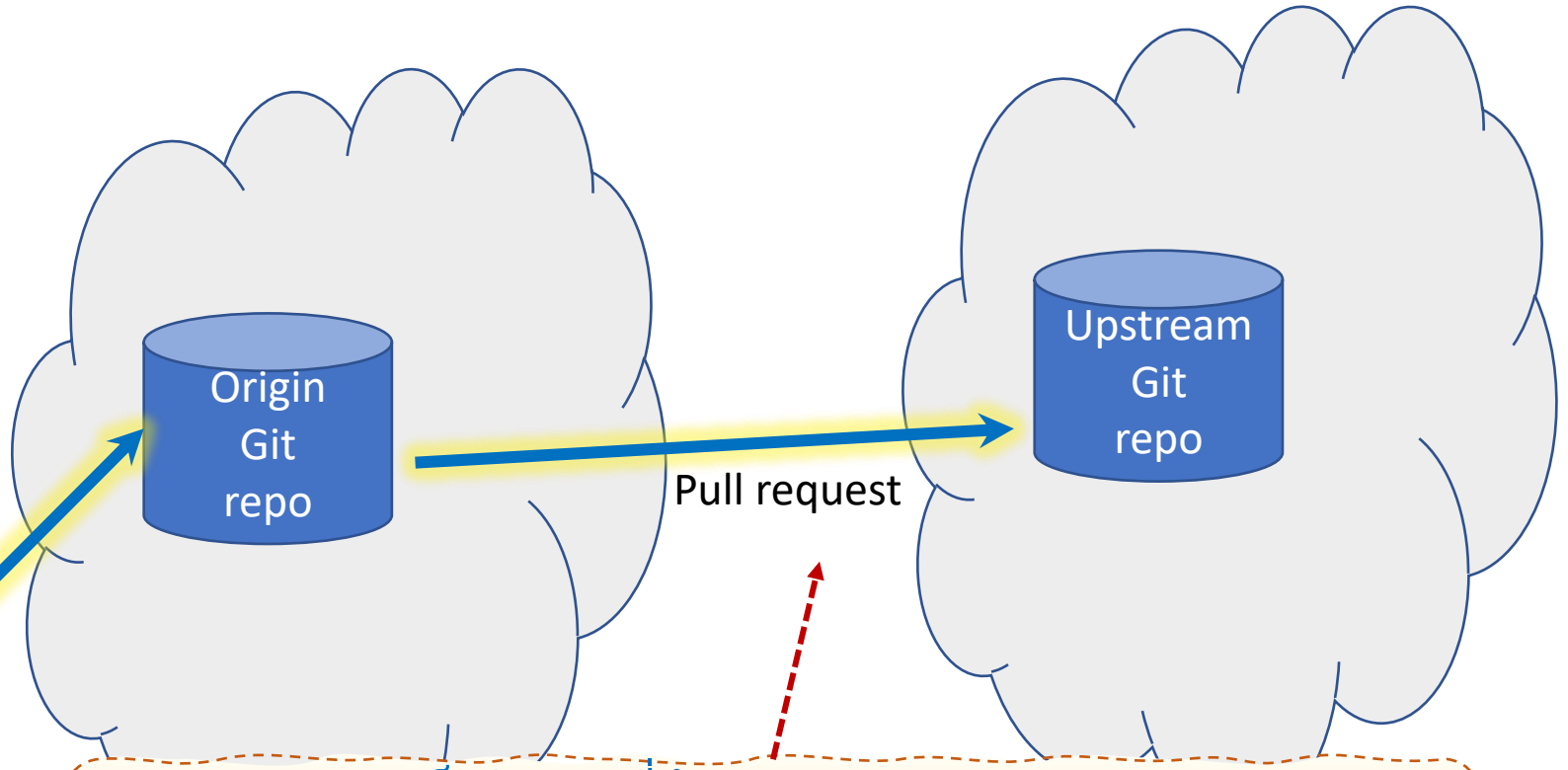


- Fork เป็นกิจกรรมที่ทำบน git server
- เป็นการ clone จาก upstream repository มายัง remote repository

Pull request

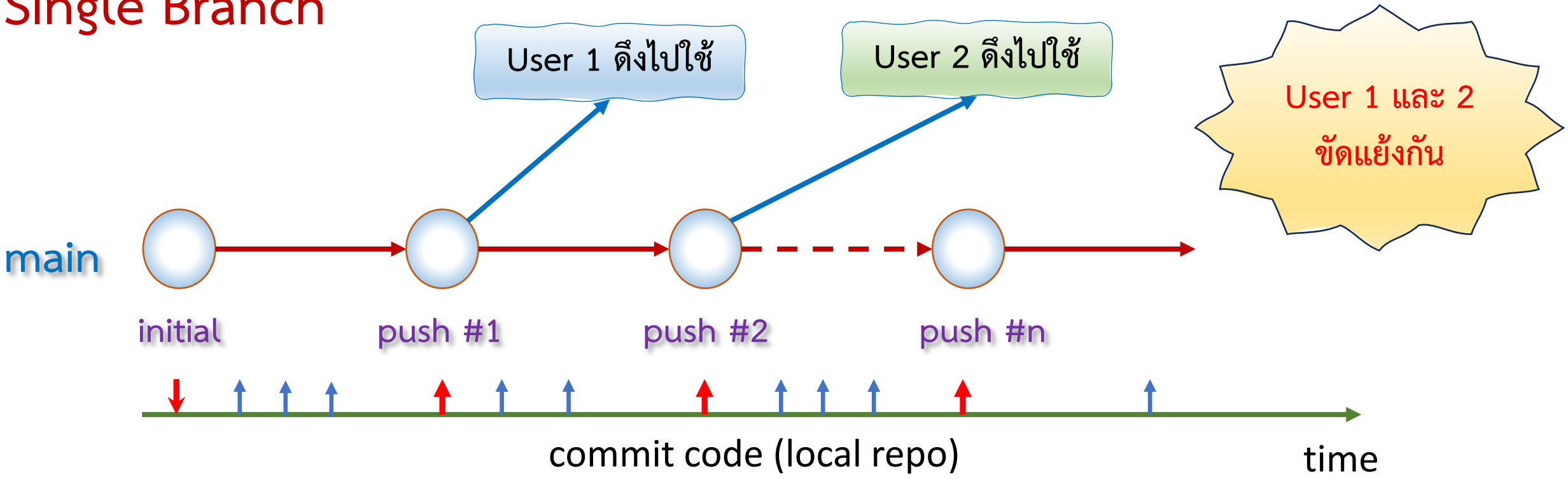
- ก่อน pull request ต้องแน่ใจว่าได้ push งานที่ถูกต้องขึ้นไปบน remote repository
- ต้องแน่ใจว่าเป็น code ที่ไม่สร้างปัญหาให้ส่วนรวม





- Pull request เป็นกิจกรรมที่ทำบน git server
- เป็นการ contributes การเปลี่ยนแปลงจาก remote repository ไปยัง upstream repository
  - เราไม่สามารถ push ขึ้นไปที่ upstream repository ได้โดยตรงเช่นเดียวที่ทำกับ remote repo เนื่องจาก git ไม่ให้สิทธิ์ในระดับเดียวกับเจ้าของ upstream repo

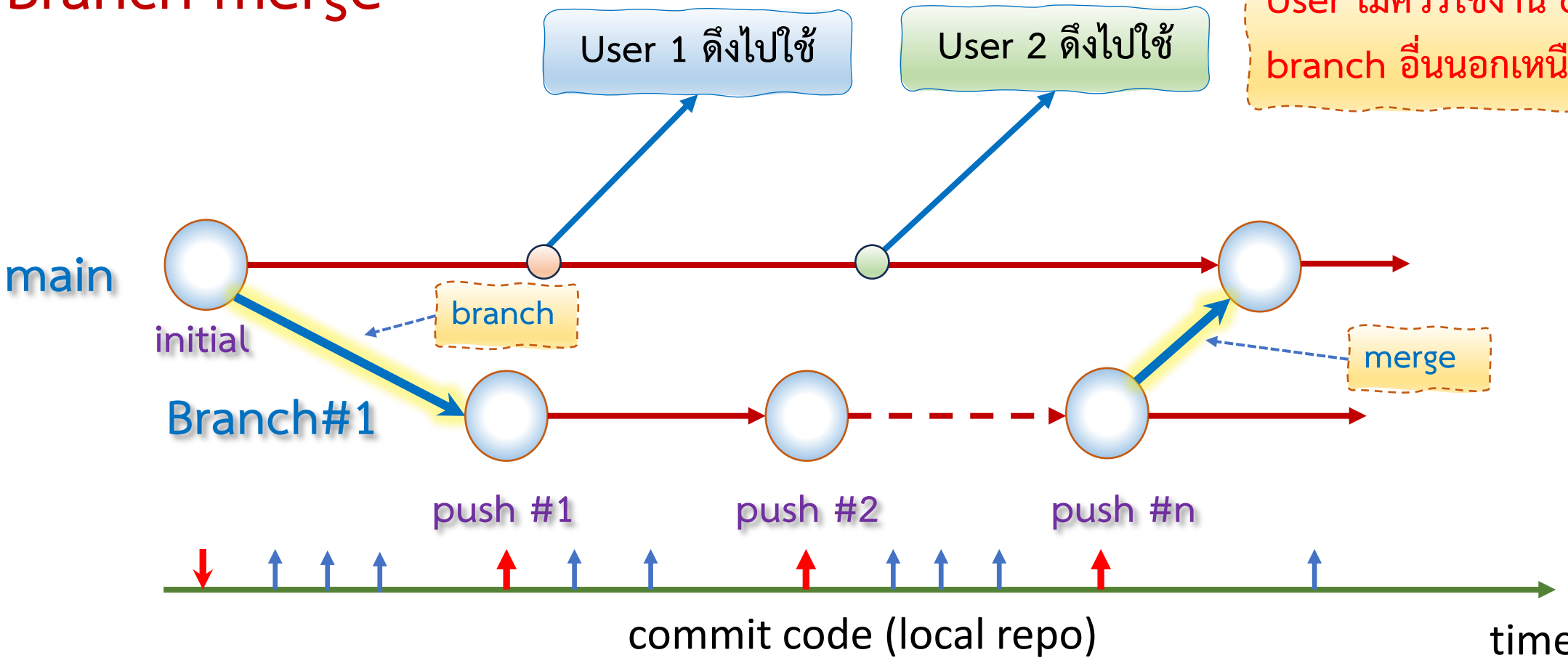
Single Branch



- โดยปกติ เราสามารถทำงานกับ branch ‘main’ เพียงอย่างเดียวก็ได้
- แต่ถ้าเป็น incremental development จะทำให้ end user ถูกรบกวนจากการเปลี่ยนแปลงบ่อยเกินไป
  - อาจจะเป็นคนที่นำ source code ที่อยู่ระหว่างการพัฒนาไปใช้ แล้วทำให้เกิดข้อร้องเรียนเข้ามาบ่อยครั้ง

Branch-merge

User ไม่ควรใช้งาน code จาก branch อื่นนอกเหนือจาก main



- เมื่อแยก branch ออกมาจาก main จะช่วยให้ developer มีอิสระในการ change, commit, push code โดยที่การเปลี่ยนแปลงเหล่านั้นจะไม่กระทบต่อ code ใน branch main
  - User ที่ดึง code จาก main ไปใช้จะได้ code เดียวกันเสมอ
- เมื่อพร้อมแจกจ่าย code ไปยัง main ให้ทำการ merge