11. Version control คือ ระบบที่จัดเก็บการเปลี่ยนแปลงที่เกิดขึ้นกับไฟล์หนึ่งหรือหลายไฟล์เพื่อที่คุณสามารถเรียกเวอร์ชั่นใดเวอร์ชั่นหนึ่งกลับมาดูเมื่อไรก็ได้ ช่วยให้คุณสามารถย้อนไฟล์บางไฟล์หรือแม้กระทั่งทั้งโปรเจคกลับไปเป็นเวอร์ชั่นเก่าได้ นอกจากนั้นระบบ VCS ยังจะช่วยให้คุณเปรียบเทียบการแก้ไขที่เกิดขึ้นในอดีต ดูว่าใครเป็นคนแก้ไขคนสุดท้ายที่อาจทำให้เกิดปัญหา แก้ไขเมื่อไร ฯลฯ และยังช่วยให้คุณสามารถกู้คืนไฟล์ที่คุณลบหรือทำเสียโดยไม่ตั้งใจได้อย่างง่ายดาย

12.-13 [ระบบ Version Control Systems แบบรวมศูนย์](https://git-scm.com/book/th/v1/%E0%B9%80%E0%B8%A3%E0%B8%B4%E0%B9%88%E0%B8%A1%E0%B8%95%E0%B9%89%E0%B8%99%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99-%E0%B9%80%E0%B8%81%E0%B8%B5%E0%B9%88%E0%B8%A2%E0%B8%A7%E0%B8%81%E0%B8%B1%E0%B8%9A-Version-Control#ระบบ-Version-Control-Systems-แบบรวมศูนย์)

Centralized Version Control Systems (CVCSs) หรือระบบ Version Control Systems แบบรวมศูนย์ มีเซิร์ฟเวอร์กลางที่เก็บไฟล์ทั้งหมดไว้ในที่เดียวและผู้ใช้หลาย ๆ คนสามารถต่อเข้ามาเพื่อดึงไฟล์จากศูนย์กลางนี้ไปแก้ไขได้



ระบบ version control แบบรวมศูนย์

การทำงานแบบนี้มีประโยชน์เหนือ local VCS ในหลายด้าน เช่น ทุกคนสามารถรู้ได้ว่าคนอื่นในโปรเจคกำลังทำอะไร ผู้ควบคุมระบบสามารถควบคุมได้อย่างละเอียดว่าใครสามารถแก้ไขอะไรได้บ้าง การจัดการแบบรวมศูนย์ในที่เดียวทำได้ง่ายกว่าการจัดการฐานข้อมูลใน client แต่ละเครื่องเยอะ

แต่ระบบแบบนี้ก็มีจุดอ่อนเหมือนกัน ตรงที่การรวมศูนย์ทำให้มันเป็นจุดอ่อนจุดเดียวที่จะล่มได้เหมือนกันเพราะทุกอย่างรวมกันอยู่ที่เซิร์ฟเวอร์ที่เดียว ถ้าเซิร์ฟเวอร์นั้นล่มซักชั่วโมงนึง หมายความว่าในชั่วโมงนั้นไม่มีใครสามารถทำงานร่วมกันหรือบันทึกการเปลี่ยนแปลงงานที่กำลังทำอยู่ไปที่เซิร์ฟเวอร์ได้เลย หรือถ้าฮาร์ดดิสก์ของเซิร์ฟเวอร์เกิดเสียขึ้นมาและไม่มีการสำรองข้อมูลเอาไว้ คุณก็จะสูญเสียข้อมูลประวัติและทุกอย่างที่มี จะเหลือก็แค่ก๊อปปี้ของงานบนเครื่องแต่ละเครื่องเท่านั้นเอง

[ระบบ Version Control Systems แบบกระจายศูนย์](https://git-scm.com/book/th/v1/%E0%B9%80%E0%B8%A3%E0%B8%B4%E0%B9%88%E0%B8%A1%E0%B8%95%E0%B9%89%E0%B8%99%E0%B9%83%E0%B8%8A%E0%B9%89%E0%B8%87%E0%B8%B2%E0%B8%99-%E0%B9%80%E0%B8%81%E0%B8%B5%E0%B9%88%E0%B8%A2%E0%B8%A7%E0%B8%81%E0%B8%B1%E0%B8%9A-Version-Control#ระบบ-Version-Control-Systems-แบบกระจายศูนย์)

Distributed Version Control Systems (DVCSs) (เช่น Git, Mercurial, Bazaar หรือ Darcs) แต่ละคนไม่เพียงได้ก๊อปปี้ล่าสุดของไฟล์เท่านั้น แต่ได้ทั้งก๊อปปี้ของ repository เลย หมายความว่าถึงแม้ว่าเซิร์ฟเวอร์จะเสีย client ก็ยังสามารถทำงานร่วมกันได้ต่อไป และ repository เหล่านี้ของ client ยังสามารถถูกก๊อปปี้กลับไปที่เซิร์ฟเวอร์เพื่อกูข้อมูลกลับคืนก็ได้ การ checkout แต่ละครั้งคือการทำสำรองข้อมูลทั้งหมดแบบเต็ม ๆ นั่นเอง



นอกจากนั้นระบบเหล่านี้ยังทำงานกับหลาย ๆ repository ได้อย่างดี ทำให้คุณสามารถทำงานกับคนหลายกลุ่มซึ่งทำงานในรูปแบบต่างกันในโปรเจคเดียวกันได้อย่างง่ายดาย เนื่องจากระบบเหล่านี้สนับสนุนการทำงานได้หลากหลายรูปแบบ ซึ่งอาจทำได้ยากในระบบแบบรวมศูนย์

16.git คือ เป็นระบบจัดการซอร์ส (source code management หรือบางที่ก็เรียกว่า version control system) ตัวหนึ่งที่ได้รับความนิยมจากนักพัฒนาซอฟต์แวร์ในปัจจุบัน เพียงแค่หน้าที่หลักของมันในการติดตามการเปลี่ยนแปลงและเรียกคืนไฟล์ตามช่วงเวลาต่างๆ ก็ทำให้การทำงานโดยใช้ Git บริหารไฟล์มีประสิทธิภาพมากขึ้นหลายเท่าตัว

Github คือ เว็บที่ให้บริการพื้นที่จัดเก็บโครงการโอเพ่นซอร์สด้วยระบบควบคุมเวอร์ชันแบบ [Git](http://git-scm.com/) โดยมีจุดประสงค์หลักคือ ทำให้การแบ่งปันและพัฒนาโครงการต่างๆด้วยกันเป็นไปได้ง่ายๆ

17.เป็นเรื่องปกติที่ stable app จะต้องมีการแก้บั๊กหลัง roll out และในเวลาเดียวกัน เราก็อาจจะต้องเพิ่ม features ใหม่ๆที่อาจจะทำให้ app ไม่ stable สิ่งที่นิยมทำใน revisioning control คือการแยก branch ใหม่เพื่อให้เราเริ่มโค้ด features ใหม่ได้ ส่วน app ที่ stable ก็จะอยู่ใน master branch เพื่อให้งานดำเนินต่อไปได้โดยไม่ต้องรอแก้บั๊กใน master ให้เสร็จก่อน ใน Git เราใช้คำสั่ง branch เพื่อแตก branch ใหม่ออกมา

Branch ใหม่ที่แยกออกมาก็จะมี history เป็นของมันเอง เราสามารถ checkout และ commit สิ่งใหม่ๆใน branch นี้ได้ นั่นคือ ก็เหมือนใน local repository หนี่งๆ เราสามารถมี sub repository ย่อยๆนั่นเอง เพียงแต่เราเรียก sub repository พวกนี้ว่า branch

พอถีงเวลาที่เราเสร็จการแก้ master branch แล้ว เราสามารถใช้คำสั่ง merge เพื่อรวมงานของ master และ new features branch เข้าด้วยกัน ไม่ต่างจากการ merge remote และ local repository

19.ใช้เข้าถึงการเปลี่ยนแปลงจาก remote มายัง local และรวมเข้าด้วยกัน ซึ่งมีหน้าที่เท่ากับ fetch +merge