11.โปรแกรม version control มีประโยชน์อย่างไร

ตอบ

ช่วยให้คุณสามารถย้อนไฟล์บางไฟล์หรือแม้กระทั่งทั้งโปรเจคกลับไปเป็นเวอร์ชั่นเก่าได้ นอกจากนั้นระบบ VCS ยังจะช่วยให้คุณเปรียบเทียบการแก้ไขที่เกิดขึ้นในอดีต ดูว่าใครเป็นคนแก้ไขคนสุดท้ายที่อาจทำให้เกิดปัญหา แก้ไขเมื่อไร ฯลฯ และยังช่วยให้คุณสามารถกู้คืนไฟล์ที่คุณลบหรือทำเสียโดยไม่ตั้งใจได้

12. ข้อได้เปรียบของ distributed version control เมื่อเทียบกับ centralized version control คืออะไร  
ตอบ

distributed version control ที่ใช้ใน BitKeeper และ git นั้น แต่ละคนจะมี copy ของไฟล์ตั้งแต่เริ่มแรกจนท้ายสุด ดังนั้นแต่ละทีมหรือแต่ละคนสามารถ maintain code ได้โดยที่ทุกคนมี copy ของไฟล์ทั้งหมดอยู่ที่ local computer และ change set ของแต่ละ version ที่เกิดขึ้นจากหลายๆ ทีมนั้นสามารถรวมกันได้ ไม่จำเป็นต้องรอ changes copy จาก central repo อีกต่อไป

ดังนั้น distributed version control จึงเร็วกว่า และผู้ใช้งานก็สามารถแก้ได้ทุกที่ โดยไม่จำเป็นจะต้องติดต่อกับ central repo ก็ได้

13.ข้อได้เปรียบของ centralized version control เมื่อเทียบกับ distributed version control คืออะไร

ตอบ

ระบบ Version Control ที่มี Server หลักเป็นศูนย์กลางในการเก็บข้อมูลของโปรเจคทั้งหมด Client แต่ละเครื่องจะเป็นแค่ Working Copy เท่านั้น วิธีทำงานคร่าวๆ เริ่มต้นเราก็สร้างโปรเจคขึ้นมาบน Server หลังจากนั้นแต่ละคนก็มาดาวน์โหลดลงเครื่องของตัวเอง เวลามีการเปลี่ยนแปลงอะไรก็อัพโหลดกลับขึ้นไปบน Server แล้วคนอื่นๆ ก็จะสามารถดาวน์โหลด Update นั้นมายังเครื่องของตัวเองได้

ดังนั้น centralized version control จึงมีการสำรองข้อมูลได้ดีกว่ามีความสามารถที่ทำงานเป็นทีมได้ดี

14.บอกแนวทางในการแก้ไข conflict ที่เกิดขึ้นเมื่อมีการ merge โปรแกรมของผู้พัฒนาหลายๆคนเข้าด้วยกัน

ตอบ

1. ทำการ Merge บ่อย ๆ

ปัญหาใหญ่ ๆ ของ Merge conflict เกิดจากจำนวน source code ที่ชนหรือขัดแย้งกันมากเหลือเกิน

### 2. เล็ก ๆ สิดี

ว่าด้วยเรื่องของ **Single Responsibility Principle (SRP)**  
เป็นหนึ่งในแนวทางการออกแบบระบบงานที่ดี คือ [SOLID](http://www.somkiat.cc/testable-design-part-2-about-solid/)

นั่นคือ ในหนึ่ง class ใน หนึ่ง method นั้น  
ควรจะมีหน้าที่การทำงานเพียงอย่างเดียวเท่านั้น  
หรือในแต่ละ class แต่ละ method ควรมีเหตุผลเดียวในการเปลี่ยนแปลงเท่านั้น

ผลที่ได้ก็คือ นักพัฒนาจะไม่ทำงาน หรือ เปลี่ยนแปลง source code ที่เดียวกันอย่างแน่นอน  
ยกเว้นจะทำงานเดียวกัน หรือ ทำงานด้วยกัน

**ดังนั้น การออกแบบที่ดีมันช่วยให้ทีมทำงานร่วมกันได้อย่างดี**  
ยิ่งแต่ละส่วนการทำงานเล็ก ๆ แล้ว  
ก็ยิ่งทำให้คุณภาพของการออกแบบระบบดี  
รวมทั้ง ลดความเสี่ยงจาก Merge conflict อีกด้วย

### 3. การพูดคุย การสื่อสาร มันสำคัญอย่างมาก

**Communication is a Key!!**

แต่ละคนในทีมพูดคุยกันหรือไม่ ?  
แต่ละคนในทีมรู้หรือไม่ว่า เพื่อน ๆ แต่ละคนทำงานอะไร ?  
รู้หรือไม่ว่า แต่ละคนแก้ไข class อะไรกันอยู่ ?  
รู้หรือไม่ว่า สิ่งที่แก้ไขไปนั้นกระทบใครบ้าง ?

ดังนั้น ถ้าคุณไม่รู้ ก็ควรพูดคุยกันซะเพื่อให้รู้  
หรือบางครั้งต้องแก้ไข class เดียวกันอยู่ตลอดเวลา  
ทำไมไม่มานั่งทำงานด้วยกันไปเลยล่ะ !!

### 4. [Mob programming](http://mobprogramming.org/)

เป็นวิธีการที่ทรงประสิทธิภาพอย่างมาก  
และเชื่อได้เลยว่า แก้ไขปัญหา Merge conflict ได้ 100%  
เนื่องจากทุกคนมานั่งทำงานด้วยกัน  
ใช้เครื่องทำงานเดียวกัน  
ดังนั้น ไม่มีทางที่ source code จะขัดแย้งกัน

แต่มันมีค่าใช้จ่ายที่สูงมาก ๆ  
ดังนั้นจึงเป็นวิธีการสุดท้ายที่จะทำนั่นเอง

15.บอกแนวทางในการลด conflict ที่เกิดขึ้นจากการ merge โปรแกรมของผู้พัฒนาหลายๆคนเข้าด้วยกัน

ตอบ

1. ทำการ Merge บ่อย ๆ
2. พูดคุยกันในที
3. Mob programming

16.Git คือ อะไร แตกต่างจาก GitHub อย่างไร

ตอบ **Git** คือ Version Control ตัวหนึ่ง ซึ่งเป็นระบบที่มีหน้าที่ในการจัดเก็บการเปลี่ยนแปลงของไฟล์ในโปรเจ็คเรา มีการ backup code ให้เรา สามารถที่จะเรียกดูหรือย้อนกลับไปดูเวอร์ชั่นต่างๆของโปรเจ็คที่ใด เวลาใดก็ได้ หรือแม้แต่ดูว่าไฟล์นั้นๆใครเป็นคนเพิ่มหรือแก้ไข หรือว่าจะดูว่าไฟล์นั้นๆถูกเขียนโดยใครบ้างก็สามารถทำได้ ฉะนั้น Version Control ก็เหมาะอย่างยิ่งสำหรับนักพัฒนาไม่ว่าจะเป็นคนเดียวโดยเฉพาะอย่างยิ่งจะมีประสิทธิภาพมากหากเป็นการพัฒนาเป็นทีม

โดยต่างจาก GitHub ที่ Github คือ เว็บที่ให้บริการพื้นที่จัดเก็บโครงการโอเพ่นซอร์สด้วยระบบควบคุมเวอร์ชันแบบ **Git** โดยมีจุดประสงค์หลักคือ ทำให้การแบ่งปันและพัฒนาโครงการต่างๆด้วยกันเป็นไปได้ง่ายๆ

17.จุดประสงค์หลักในการ branch คืออะไร

ตอบ เป็น feature ที่ช่วยให้นักพัฒนาสามารถที่จะทำงานได้สะดวกขึ้น ยกตัวอย่างเช่น เรามีโค๊ดที่ดีอยู่แล้ว แต่อยากจะทดลองอะไรนิดๆหน่อย หรือแก้ไขอะไรก็ตาม ไม่ให้กระทบกับตัวงานหลัก ก็เพียงแค่สร้าง branch ใหม่ขึ้นมา เมื่อแก้ไขหรือทำอะไรเสร็จแล้ว ก็ค่อยเซฟกลับมาที่ master เหมือนเดิม

18.Fast forward merge คืออะไรและทำไมการ push ไปที่ remote repo จึงควรจะต้อง merge แบบนี้

ตอบ

If Master has not diverged, instead of creating a new commit, git will just point master to the latest commit of the feature branch. This is a “fast forward.”

19.หน้าที่หลักของคำสั่ง git pull คืออะไร

ตอบ

จริงๆแล้ว git pull ก็คือรวมโค๊ดจาก remote มายัง local โดยที่เราไม่สามารถรู้ได้เลยว่าจะรวมโค๊ดอะไรบ้าง รู้แค่หลังจาก pull เสร็จแล้วนั่นเอง ซึ่งจริงๆแล้ว git pull มันก็คือการทำ git fetch และต่อด้วยgit merge อัตโนมัตินั่นเอง

20.แผนภาพด้านล่างนี้ต้องการสื้อความหมายอะไร

ตอบ