(1) DevOps คืออะไร มีความสัมพันธ์กับการพัฒนาซอฟต์แวร์อย่างไร

DevOps คือ ชุดแนวทางการพัฒนาซอฟต์แวร์ที่รวมการพัฒนาซอฟต์แวร์และการดำเนินงานด้านเทคโนโลยีสารสนเทศ เพื่อลดวงจรชีวิตของการพัฒนาระบบในขณะที่นำเสนอคุณลักษณะการแก้ไขและการอัปเดตบ่อยครั้งเพื่อให้สอดคล้องกับวัตถุประสงค์ทางธุรกิจ

กระบวนการหรือการปฏิบัติ ของ DevOps เกี่ยวข้องกับชุดของกระบวนการทางเทคนิคเช่น การพัฒนาอย่างต่อเนื่อง, การรวมอย่างต่อเนื่อง (CI), การทดสอบอย่างต่อเนื่อง (CI), การปรับใช้อย่างต่อเนื่อง (CD) และการตรวจสอบอย่างต่อเนื่อง

(2) CI/CD คืออะไร มีความสัมพันธ์กับ DevOps อย่างไร

CI (Continuous Integration) คือ กระบวนการรวม source code ของคนในทีมพัฒนาเข้าด้วยกัน และมีการ test ด้วย test script เพื่อให้แน่ใจว่าไม่มี error ในส่วนใดๆ ของโปรแกรม แล้วถึงทำการ commit ไปที่ branch master อีกต่อนึง โดยในการพัฒนานั้น มักใช้ Build Server มาช่วย กล่าวคือจะเริ่มทำการ Integration กันตั้งแต่เมื่อมีการเปลี่ยนแปลง Source Code ที่ Repository กลาง ระบบจะทำการตรวจสอบ Code หลังจากการเปลี่ยนแปลงว่าทำงานร่วมกันได้หรือไม่ตั้งแต่ Compile, Testing

มีความสำพันธ์กับ DevOps ในด้านการแก้ปัญหาในการ deploy code โดยมีการทำงานตั้งแต่ Plan, Code Build, Test, Release, Deploy, Operate, Monitor หรือบางทีเรียกสั้นๆ ว่า Pipeline

(3) หากองค์กรต้องการนำเอา DevOps และ CI/CD เข้ามาเป็นส่วนเสริมในกระบวนการพัฒนาซอฟต์แวร์จะต้องทำอย่างไรบ้าง

1. Plan

การจะพัฒนาโปรแกรมขึ้นมา ต้องมีการวางแผนร่วมกันจากทุกฝ่ายที่เกี่ยวข้องอย่างละเอียดก่อน

โปรแกรมเราจะทำอะไรบ้าง จะพัฒนาด้วยภาษาอะไร Framework อะไร จะรันบน Platform อะไร จะไปลงไว้ที่ไหน จะเก็บข้อมูลอย่างไร จะแก้ปัญหาที่เกิดขึ้น และจะพัฒนา Process การทำงานอย่างไร

2. Create

การสร้างหรือการพัฒนา นี่ก็คือขั้นตอนของการเขียนโปรแกรมขึ้นมา DevOps เข้ามาช่วยทำให้ Developers ทุกคนทำงานบน Environment แบบเดียวกัน ไม่ว่าเขาจะพัฒนาบน OS ใดก็ตาม

3. Verify

การตรวจสอบอยู่อย่างสม่ำเสมอว่าโปรแกรมที่พัฒนาขึ้นมา ไม่ว่าจะเพิ่มฟีเจอร์อะไรเข้ามาจะไม่ไปกระทบหรือไปก่อบั๊กทำให้การใช้งานเดิมพังเสียหายนั้นเป็นเรื่องสำคัญมาก โดยเฉพาะโปรแกรมขนาดใหญ่ที่ร่วมกันทำกับทีมงานจำนวนมาก เราจะมั่นใจได้อย่างไรว่า Code ที่ Developer X ส่งขึ้นมาจะเข้ากันได้ดีกับการทำงานเดิม วิธีการที่จะรู้ได้เร็วที่สุดคือ เราจะต้องมีการทำ Testing อย่างครอบคลุมในทุกๆ Components ของโปรแกรม แล้วทำให้มันรันโดยอัตโนมัติทุกครั้งก่อนจะปล่อยโปรแกรมออกไป เมื่อใดที่ Test Fail ก็จะรู้ได้ทันทีว่ามีปัญหาเกิดขึ้นแล้ว ให้หยุดการส่งงานที่มีปัญหาไปก่อน สามารถเข้าไปหาสาเหตุและแก้ไขปัญหาได้อย่างทันท่วงที

4. Package

การจะส่งโปรแกรมขึ้นไปรันบน Server เพื่อความสะดวกรวดเร็ว จะต้องมีการเอาโปรแกรมไปใส่ใน Technology ที่ออกแบบมาเพื่อนำโปรแกรมไปรันได้อย่างราบลื่น

5. Release

เมื่อมี engine พร้อมแล้ว ก็พร้อมสำหรับการนำโปรแกรมไปรันบน Deployment Platform ที่ต้องการ ตอนนี้กำลังทำของ Development หรือทำหรือ Production อยู่ จะต้องเลือกไป Deploy ให้ถูกที่

6. Configure

โปรแกรมที่พัฒนานั้น **จำเป็นอย่างมาก** ที่จะต้องตั้งค่าได้ ตัวอย่างการตั้งค่าที่ควรมีเลยก็เช่น การเลือก Database ตั้งค่า Email เป็นต้น

ยิ่งถ้ามี Service แยกอื่นๆ ที่ต้องใช้แล้ว ยิ่งสำคัญ เพราะ Development Server และ Production Server ควรอยู่แยกกันอย่างเด็ดขาด เราจะต้องหาทางสร้าง engine มา 1 อัน แล้วเราจะใช้มัน Deploy ลงบน Environments ที่ต่างกันให้ได้

วิธีการที่จะทำได้ก็คือการทำให้โปรแกรมของเราตั้งค่าได้ คือโปรแกรมที่ไปรันเป็นตัวเดียวกัน แต่เปิดด้วยการตั้งค่าที่แตกต่างกัน ทำให้ได้การทำงานที่แตกต่างกัน โดยปรกติจะใช้ไฟล์ .env และ Environment Variable ในการตั้งค่ากันเป็นมาตรฐานที่ใช้กันบนทุก Platform

7. Monitor

การตรวจสอบว่าโปรแกรมมันขึ้นไปแล้วทำงานได้ปกติ มีสุขภาพแข็งแรงดี การดู Log ว่าโปรแกรมเราทำงานมีปัญหาตรงไหน มีความสำคัญอย่างมากในการแก้ไขบั๊ก ยิ่งในตอนนี้ที่มักจะออกแบบให้โปรแกรมมันสามารถทำให้ Horizontal Scale ได้ (การสร้างเครื่องขึ้นมาหลายๆ เครื่องให้มาแบ่ง Load ทำงานแยกกันได้) จำเป็นอย่างยิ่งที่จะต้องมี Centralized Logging System รวบรวม Log เอามาไว้ในที่เดียว เพราะการ Remote เข้าไปอ่าน Log ในแต่ละเครื่องเป็นไปไม่ได้อีกแล้ว อีกส่วนหนึ่งที่สำคัญไม่แพ้กันคือการทำ Monitoring ดูโหลดต่างๆ ของระบบ ตอนนี้มีกี่เครื่องเปิดขึ้นมาใช้งาน เครื่องล่มหรือเปล่า ให้บริการได้ทันที่ลูกค้าเข้ามาใช้หรือไม่ เรื่องเหล่านี้ล้วนมีเครื่องมือออกมาแก้โจทย์เหล่านี้แล้วทั้งสิ้น เพียงแต่ต้อง Implement โปรแกรมให้ไปรองรับการใช้งานกับเครื่องมือเหล่านั้น