**นางสาววิภาดา ศิลาราช 613020598-2 sec.1 No.35**

(1) DevOps คืออะไร มีความสัมพันธ์กับการพัฒนาซอฟต์แวร์อย่างไร

**DevOps** เป็นคำที่ย่อมาจาก Development Operations เป็นศาสตร์ที่ช่วยเชื่อมกระบวนการพัฒนาโปรแกรม การส่งเอาขึ้นไป Deploy ให้ผู้ใช้เข้าถึงได้อย่างมีประสิทธิภาพ มีคุณภาพ ไปจนถึงการติดตามสถานะการทำงานของโปรแกรมเพื่อนำกลับมาพัฒนาปรับปรุงกระบวนการพัฒนาโปรแกรม DevOps Engineer เปรียบได้กับ System Admin ในสมัยก่อน ที่ทำหน้าที่เอาโปรแกรมขึ้นไปรันบน Server ให้ได้ แล้วคอยดูไม่ให้มันล่ม แต่พัฒนาร่างขึ้นมาทำเรื่อง Automation ด้วย คือทำให้กระบวนการทั้งหมดที่ต้องทำมือในสมัยก่อน ทำงานโดยอัตโนมัติได้ นอกจากนั้นยังลงไปช่วยนักพัฒนาทำงานด้วย ในส่วนของการวางสภาพแวดล้อมในการพัฒนาโปรแกรมที่ดี

**ความสัมพันธ์กับการพัฒนาซอฟต์แวร์** หลังจากการแพร่หลายของ Docker ที่ทำให้ผู้คนเข้าถึงโลก Container ได้ง่ายขึ้น การทำให้เกิดความมั่นใจว่าโปรแกรมของเราจะไปรันที่ไหนก็ได้โดยได้ผลลัพธ์เหมือนกันแน่นอนเป็นเรื่องสำคัญมาก มันกลายเป็นการยกระดับมาตรฐานการทำงานของวงการ Software Engineering ไปเลย นอกจากเรื่องความเป็นกระแสแล้ว ก็เป็นเพราะว่ามีเครื่องมือดีๆ ในวงการ DevOps พัฒนาขึ้นมาหลายตัว ตั้งแต่ Docker, Kubernetes, CI/CD Platforms ก็เกิดขึ้นหลายตัว ช่วยให้นักพัฒนาทั่วโลกเข้าถึงเครื่องมือเจ๋งๆ เหล่านี้ได้อย่างทั่วถึง นอกจากนั้นการทำ DevOps ยังเพิ่ม Productivity ให้กับทีม Engineers ได้อย่างมาก (เมื่อ Implement อย่างถูกวิธีในเวลาที่เหมาะสม) ตั้งแต่เรื่องการนำ Docker มาใช้ในทีมพัฒนา ช่วยให้โปรแกรมที่รันทำงานเหมือนกันทุกคนบนทุก Platform ช่วยให้ทุกคนอุ่นใจได้ว่าโปรแกรมที่รันจะไม่ได้ใช้งานได้แค่บนเครื่องของเขาคนเดียวแล้วไปพังบนเครื่องอื่น

(2) CI/CD คืออะไร มีความสัมพันธ์กับ DevOps อย่างไร

**CI/CD** คือ กระบวนการหนึ่งที่จะมาช่วยในพัฒนา Software ให้มีประสิทธิภาพมากขึ้น ทั้งในแง่ของระยะเวลาการพัฒนาและคุณภาพของ Software

Continuous Integration (CI) คือ กระบวนท่าที่ใช้สำหรับการรวมร่างซอฟแวร์ที่มีการพัฒนาแยกส่วนกันอย่างอัตโนมัติ อาจจะโดยหนึ่งหรือหลายนักพัฒนาก็ตามที สุดท้ายแล้วซอฟแวร์ที่พัฒนาชิ้นเล็กๆ ที่พัฒนาขึ้นมาจะต้องนำมารวมกันเป็นชิ้นใหญ่หนึ่งชิ้น จะทำอย่างไรให้มั่นใจได้ว่า ไม่มีชิ้นส่วนใดที่จะส่งผลให้ชิ้นส่วนอื่นๆ พังเสียหาย โดยเฉพาะอย่างยิ่งเมื่อเป็นการพัฒนาโดยโปรแกรมเมอร์ร้อยพ่อพันแม่มาพัฒนาร่วมกัน ยิ่งมีความเสี่ยงที่จะมีนักพัฒนาบางคนอาจจะพลาดพลั่งมือปล่อยบั๊กออกไป โจทย์ก็คือ ทำอย่างไรให้เราป้องกันสิ่งเหล่านั้นได้ คำตอบก็คือ ทำระบบที่คอยทดสอบความเข้ากันได้ของแต่ละชิ้นส่วนโดยอัตโนมัตินั่นเอง และนั่นก็คือการทำ CI

Continuous Delivery (CD) คือ การทำให้ซอฟแวร์ของเราส่งออกไปถึงลูกค้าโดยอัตโนมัติอย่างมีคุณภาพ เป็นขั้นตอนที่ทำต่อหลังจาก CI กล่าวคือ หลังจากที่ทดสอบระบบที่พัฒนาเพิ่มขึ้นมาแล้วว่าสามารถเข้ากันได้ดีไม่มีปัญหา ก็ถึงเวลาส่งซอฟแวร์ดังกล่าวขึ้นไปใช้งานจริง ซึ่งถ้าไม่มีระบบอัตโนมัติก็ต้องมีอัพโหลดขึ้นไปบนเซิร์ฟเวอร์เอง พอเริ่มเห็นว่ามันเป็นงานเดิมๆ ซ้ำๆ ก็เริ่มมีคนทำให้มันเป็นอัตโนมัติ แล้วก็เกิดเป็น CD ขึ้นมา นอกจากนั้น ซอฟแวร์ปัจจุบันบางทีมีการวางบนเครื่องจำนวนมาก หรืออาจจะเป็น Micro Services ที่ใช้ Container จำนวนมาก การทำ CD จะช่วยให้กระบวนการเหล่านั้นง่ายขึ้นเป็นอันมาก ดังนั้น CI/CD รวมกันจึงเป็นกระบวนทดสอบคุณภาพซอฟแวร์ของซอฟแวร์และส่งขึ้นสู่สาธาณะชนโดยอัตโนมัตินั่นเอง

**ความสัมพันธ์กับ DevOps**

CI/CD เป็น platform ตัวนึงที่อยู่ใน Devops ในการพัฒนา software ให้มีประสิทธิภาพยิ่งขึ้น

สำหรับแนวคิด และ Tools ในการทำงานแบบ DevOps (CI/CD) ไม่มีแบบไหนผิด แบบไหนถูก ขึ้นอยู่กับการเอามาประยุกต์ใช้งาน ให้เหมาะสมกับงานของเรา องค์กรของเรา เพื่อทำให้งาน เกิดประสิทธิภาพสูงสุด ตอบโจทย์ผู้ใช้งาน product ของเราให้ดีที่สุด และที่สำคัญ ต้อง Monitor ได้ ต้องมี Dashboard เอาไว้ Tracking Metric ต่างๆ ได้

(3) หากองค์กรต้องการนำเอา DevOps และ CI/CD เข้ามาเป็นส่วนเสริมในกระบวนการพัฒนาซอฟต์แวร์จะต้องทำอย่างไรบ้าง

มีวิธีการทำดังนี้

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 เวลา Developer ทำงานบนคอมพิวเตอร์คนละ OS กัน มักจะมีปัญหา “ทำไมโปรแกรมรันบนเครื่องเธอไม่ได้ล่ะ บนเครื่องฉันก็รันได้อยู่นะ!” ซึ่งมันไม่ควรจะเกิดขึ้นแล้วในยุคนี้ที่เรามี Docker ที่ช่วยให้ทุกคนพัฒนาโปรแกรมอยู่บนพื้นฐาน Environment เดียวกันเป๊ะ ตั้งแต่ขั้นตอนการพัฒนาไปจนถึงการส่งโปรแกรมขึ้น Production

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 ซึ่งในปัจจุบันนี้เราใช้ Docker กัน เวลาจะใช้ เราจะสร้าง Docker Image ที่เอาตัวโปรแกรมพร้อมสำหรับการรันใส่อยู่ในนั้น แล้วเอา Docker Image นี้ไป Deploy ที่ไหนก็ได้สามารถรัน Docker ได้ โปรแกรมก็จะรันขึ้นมาพร้อมใช้งาน

5. Release เมื่อมี Docker Image 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 Docker Image มา 1 อัน แล้วเราจะใช้มัน Deploy ลงบน Environments ที่ต่างกันให้ได้ วิธีการที่จะทำได้ก็คือการทำให้โปรแกรมของเราตั้งค่าได้เนี่ยแหละ คือโปรแกรมที่ไปรันเป็นตัวเดียวกัน แต่เปิดด้วยการตั้งค่าที่แตกต่างกัน ทำให้ได้การทำงานที่แตกต่างกัน โดยปรกติจะใช้ไฟล์ .env และ Environment Variable ในการตั้งค่ากันเป็นมาตรฐานที่ใช้กันบนทุก Platform

7. Monitor สุดท้ายที่ขาดไม่ได้เลยคือการตรวจสอบว่าโปรแกรมมันขึ้นไปแล้วทำงานได้ปรกติ 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

**อ้างอิง**

**DevOps CI/CD คืออะไร?**. (2562). [ออนไลน์]. เข้าถึงได้จาก : <https://ton.packetlove.com/blog/life-style/devops-ci-cd.html>. (วันที่สืบค้นข้อมูล 24 กุมภาพันธ์ 2564).

**DevOps คืออะไร นำมาประโยชน์ได้ยังไง และตัวอย่างการทำ DevOps ที่ Credit OK**. (2562). [ออนไลน์].

เข้าถึงได้จาก : <https://www.spicydog.org/blog/introduction-to-devops-and-the-practical-use-cases-at-credit-ok/>. (วันที่สืบค้นข้อมูล 24 กุมภาพันธ์ 2564).

**เมื่อหัดทำ CI/CD ครั้งแรก**. (2561). [ออนไลน์]. เข้าถึงได้จาก : <https://medium.com/@thraithep.a>. (วันที่สืบค้นข้อมูล 24 กุมภาพันธ์ 2564).