นายมีชัย หนูพิศ รหัสนักศึกษา 613020594-0 Section 1

DevOps และ CI/CD

1. DevOps คืออะไร มีความสัมพันธ์กับการพัฒนาซอฟต์แวร์อย่างไร

(Anon n.d.) DevOps คือการนำเอาแนวความคิดเชิงวัฒนธรรม แนวทางปฏิบัติ และเครื่องมือต่าง ๆ มาผสมผสานกันและประยุกต์ใช้ร่วมกัน เพื่อเพิ่มความสามารถในองค์กร ทำให้การส่งมอบซอฟต์แวร์และบริการรวดเร็วขึ้น ด้วยการพัฒนาและปรับปรุงผลิตภัณฑ์ต่าง ๆ ให้รวดเร็วกว่ากระบวนการพัฒนาและจัดการกับซอฟต์แวร์แบบเดิม ซึ่งความรวดเร็วนี้ทำให้องค์กรสามารถให้บริการกับลูกค้าได้ดีมากกว่าเดิม และสามารถต่อสู้กับองค์กรอื่น ๆ ในตลาดได้อย่างมีประสิทธิภาพ

(Gene et al. 2016) (Gene, Kevin, and George 2013) DevOps มีความสำคัญในการช่วยให้การพัฒนาซอฟต์แวร์และการส่งมอบซอฟต์แวร์เป็นไปได้อย่างรวดเร็วและมีประสิทธิภาพมากขึ้น เพราะ DevOps เกิดขึ้นจากการแก้ปัญหาของหน่วยงานไอที ที่มีความล่าช้าในการส่งมอบซอฟต์แวร์ให้แก่ลูกค้า โดยจุดเริ่มต้นของปัญหาเกิดจาก 2 เป้าหมายสำคัญของหน่วยงานไอทีที่จะต้องทำให้แก่บริษัทหรือองค์กร ได้แก่ 1) ความต้องการที่จะพัฒนาระบบให้ทันการเปลี่ยนแปลงของตลาดอย่างรวดเร็ว เพื่อคงความสามารถในการแข่งขัน 2) ความต้องการในการให้บริการระบบที่มีความเสถียร เชื่อถือได้และมีความปลอดภัย ซึ่งเป้าหมายแรกผู้ที่รับผิดชอบคือฝ่ายการพัฒนา (Development) และเป้าหมายที่สองผู้รับผิดชอบคือฝ่ายการดำเนินงาน (Operations) ซึ่งปัญหาทั้งหมดก็คือ ความยากที่หน่วยงานไอทีจะต้องทำงานให้บรรลุเป้าหมายทั้งสองข้อในเวลาเดียวกันได้ เพราะการเปลี่ยนแปลงซอฟต์แวร์ในแต่ละครั้งมีโอกาสที่ส่งผลอันตรายต่อซอฟต์แวร์ได้ ถึงแม้จะเป็นการเปลี่ยนแปลงเพียงเล็กน้อยก็ตาม ซึ่งบริษัทหรือองค์กรมีทางออกสองทางได้แก่ 1) การเพิ่มเวลาในการพัฒนาและทดสอบซอฟต์แวร์ จะทำให้ระบบมีความปลอดภัยและมีความเสถียรมากขึ้น 2) การเพิ่มความเร็วในการพัฒนาและทดสอบซอฟต์แวร์ให้สามารถส่งมอบได้เร็วที่สุด ซึ่งมีโอกาสที่จะเกิดปัญหาในภายหลังและส่งผลต่อความเสถียรของซอฟต์แวร์ ดังนั้นการที่บริษัทหรือองค์กรมีความล่าช้าในการส่งมอบซอฟต์แวร์ให้แก่ลูกค้า เกิดจากการมีเป้าหมายที่แตกต่างกันของฝ่ายการพัฒนาและฝ่ายการดำเนินงาน

หลักการสำคัญ 3 ข้อ ของ DevOps ที่จะช่วยแก้ไขปัญหาสำหรับฝ่ายการพัฒนาและฝ่ายการดำเนินงานที่มีเป้าหมายที่แตกต่างกัน

1.1 หลักการของ flow

คือการทำให้การไหลของข้อมูลเป็นไปได้อย่างลื่นไหล โดยการปรับปรุง flow การทำงานหรือการส่งต่องานระหว่างฝ่ายต่าง ๆ จากซ้ายไปขวา เริ่มตั้งแต่ฝ่ายธุรกิจ (Business) ทำการรวบรวมความต้องการจากลูกค้า (Requirement) เพื่อส่งไปยังฝ่ายการพัฒนา แล้วฝ่ายการพัฒนาส่งต่อให้แก่ฝ่ายการดำเนินงานและดำเนินงานส่งให้แก่ลูกค้า ซึ่งต้องทำการลดขนาดของชิ้นงานให้เป็นงานย่อย ๆ แล้วทำการปรับใช้ (Deploy) ให้มากขึ้น อีกทั้งต้องไม่ส่งงานที่มีข้อบกพร่อง (Defect) ให้ฝ่ายถัดไป เพราะจะทำให้เกิดการแก้ไข ต้องทำการส่งงานกลับไปกลับมาระหว่างฝ่ายแล้วส่งผลให้เกิดความล่าช้า และต้องมีวิธีปฏิบัติเช่น Continuous Integration (CI) ความสามารถในการสร้างหรือการจำลอง Productive Environment ตามความต้องการและการจำกัดงานในกระบวนการ (work in process)

1.2 หลักการของ feedback

คือการสำรวจและการแก้ไขกับปัญหาที่เกิดขึ้นระหว่างการส่งต่อข้อมูลระหว่างฝ่ายต่าง ๆรวมถึงการสร้างวิธีการป้องกันเพื่อไม่ให้เกิดปัญหาเดิมซ้ำ ๆ ด้วยการทำให้ข้อมูลถูกต้องและมีคุณภาพตั้งแต่แหล่งที่มาหรือการสร้างองค์ความรู้ภายในองค์กร อีกทั้งต้องมีการทำการทดสอบแบบอัตโนมัติ (Automate Test) การแก้ไขปัญหาแบบทันทีเมื่อเกิดปัญหา การตรวจสอบซอฟต์แวร์เพื่อให้ผู้ที่เกี่ยวข้องสามารถเข้าไปดูได้ (Monitoring) และการแจ้งเตือนเมื่อซอฟต์แวร์เกิดปัญหา

1.3 หลักการของการทดลองและเรียนรู้อย่างต่อเนื่อง

เป็นการสร้างวัฒนธรรมในองค์กรเพื่อ 1) สนับสนุนการทดลองสิ่งใหม่ ๆ เพื่อปรับปรุงกระบวนการทำงาน ซึ่งจะทำให้เกิดการเรียนรู้จากทั้งความสำเร็จและความล้มเหลว 2) สร้างความเข้าใจในการปฏิบัติอย่างต่อเนื่องเป็นหนทางสู่ความเชี่ยวชาญและความสำเร็จ การทำการทดลองนำสิ่งใหม่ ๆ มาปรับใช้เพื่อปรับปรุงการทำงานแบบเดิม จะมีประสิทธิภาพมากขึ้นเมื่อนำมาประยุกต์ใช้กับหลักการของ feedback เพราะเมื่อเกิดปัญหาจากการทดลองจะสามารถแก้ไขปัญหาได้อย่างรวดเร็ว ส่งผลให้องค์กรหรือบริษัทเรียนรู้ได้อย่างรวดเร็วมากกว่าองค์กรหรือบริษัทอื่น ๆ ดั่งคำพูดที่ว่า “Fail fast win big” ซึ่งองค์กรหรือบริษัทสามารถทำให้เกิดขึ้นได้ ด้วยการสร้างวัฒนธรรมความกล้าที่จะเสี่ยงในการเปลี่ยนแปลง และสร้างความไว้วางใจซึ่งกันและกัน รวมไปถึงการออกแบบกระบวนการทำงานที่สามารถนำความรู้หรือความผิดพลาดจากการทำงานของพนักงานแต่ละบุคคล ไปกระจายให้พนักงานคนอื่น ๆ ได้รับรู้ เพื่อให้กลายเป็นองค์ความรู้ของบริษัทในที่สุด

2. CI/CD คืออะไร มีความสัมพันธ์กับ DevOps อย่างไร

(Puisungsoen 2014) Continuous Integration (CI) คือ กระบวนการในการรวบรวมซอฟต์แวร์แบบอัตโนมัติ ซึ่งซอฟต์แวร์ถูกสร้างขึ้นจากหลาย ๆ นักพัฒนา ซึ่งการรวบรวมซอฟต์แวร์เข้าด้วยกันโดยไม่ให้ซอฟต์แวร์ตัวใดส่งผลให้ตัวอื่น ๆ พังเสียหายต้องมีการทดสอบ (Testing) ความเข้ากันได้ของซอฟต์แวร์แต่ละชิ้น โดยเริ่มตั้งแต่การทำ Unit Testing ที่สร้างจากทีมนักพัฒนาและใช้เป็นส่วนที่จะใช้ตรวจสอบว่าสิ่งที่ทีมนักพัฒนาสร้างขึ้นนั้นยังทำงานถูกต้องซึ่งจะใช้เวลาไม่มากนัก ในส่วนใหญ่ของการพัฒนาจะใช้การ Build Server มาช่วยเพื่อให้บรรลุเป้าหมายตามที่ตั้งไว้ ซึ่งเริ่มจากการ Integration ตั้งแต่เมื่อมีการเปลี่ยนแปลง Source Code ที่ Repository ส่วนกลาง ระบบจะทำการตรวจสอบ Source Code หลังจากมีการเปลี่ยนแปลงว่าสามารถทำงานร่วมกันได้หรือไม่ตั้งแต่การ Compile และ Testing

Continuous Delivery (CD) คือการทำงานแบบอัตโนมัติในทุก ๆ ขั้นตอนไปจนถึงการ Deployment ขึ้น production และ Continuous Deployment (CD) คือ การทำงานใน Deployment Pipeline จะเริ่มต้นทำงานตั้งแต่การ Compile และ Build ไปจนถึงการทดสอบต่าง ๆ แต่จะยังไม่มีการ Deploy ขึ้น Production ทันที เช่น Acceptance Test เป็นแบบอัตโนมัติทั้งหมด ส่วนขั้นตอนในการ Deployment ขึ้น Production นั้น จะต้องได้รับการอนุมัติหรือการตัดสินใจกันก่อนจากทางฝ่ายธุรกิจ ซึ่งเป็นการทำงานแบบ Manual หรืออาจเป็น One Click Deploy

(Softmelt n.d.) CI/CD มีความสัมพันธ์กับ DevOps ในการช่วยแก้ปัญหาการ Deploy Code ที่ล่าช้า โดย CI/CD จะทำงานตั้งแต่การ Plan, Code, Build, Test, Release, Deploy, Operate และ Monitor หรือที่เรียกว่า Pipeline ซึ่งขั้นตอนการพัฒนาซอฟต์แวร์ตามแนวทางของ CI/CD มี 6 ขั้นตอนดังต่อไปนี้

2.1 Developer เมื่อทำการพัฒนา Feature เสร็จแล้ว จะทำการ Build, Test และ Run บนเครื่องของจนเอง (Local) เพื่อทำให้แน่ใจว่าระบบทำงานได้ถูกต้องและให้แน่ใจว่าสิ่งที่เปลี่ยนแปลงนั้นไม่กระทบกับส่วนอื่น ๆ

2.2 ทำการดึง Source Code ล่าสุดจาก Repository ของซอฟต์แวร์ เพื่อตรวจสอบว่ามีการเปลี่ยนแปลงหรือไม่ ถ้ามีการเปลี่ยนแปลงก็ให้ทำการรวม (merge) ที่เครื่องของ Developer ก่อน จากนั้นจึงทำการ Build, Test และ Run อีกรอบ เมื่อทุกอย่างผ่านทั้งหมด ให้ทำการส่งการเปลี่ยนแปลงไปยัง Repository กลาง

2.3 เมื่อ Repository กลางมีการเปลี่ยนแปลง จะต้องมีระบบ CI ทำการ Build หลังจากนั้นจะส่งต่อไป Run Unit Testing ก่อน ถ้าผ่านหมดถึงจะส่งต่อไปยังระบบ Continuous Delivery เพื่อ Deploy to sit environment

2.4 เมื่อ Source Code ถูก Deploy to sit environment แล้วจะ Trigger ไปสั่งให้ Run job Automated Testing ในระดับของ Test case ซึ่งเป็นชุด Test case ย่อย ๆ ไม่เยอะมากเฉพาะในส่วนของ Feature code ที่ถูก Deploy มาเท่านั้น

2.5 หลังจาก Run test เสร็จแล้วถ้าเกิดว่า Run มีบางส่วนไม่ผ่านทั้งหมดจะไม่ส่งต่อไปยังระบบ Continuous Delivery เพื่อ Deploy to staging environment QA จะทำการ Investigate ว่าเกิดจากอะไร เป็นที่ซอฟต์แวร์มี Bug เกิดขึ้นจริงหรือไม่ ถ้ามี Bug ก็ให้ Dev แก้ไข และ Deploy มาใหม่ วน Loop ใหม่

2.6 กรณีหลังจาก Run test ผ่านทั้งหมดจะส่งต่อไปยังระบบ Continuous Delivery เพื่อ Deploy to uat(staging) environment เมื่อ Source Code ถูก Deploy to uat(staging) แล้ว จะ Trigger ไปสั่งให้ Run job automated testing ใน Level ของ Test case: Regression test และ QA ก็ทำการทดสอบ Acceptance testing ไปด้วยพร้อม ๆ กันที่ uat(staging) environment นี้ เมื่อมีการ Deploy ซ้ำ ๆ เพื่อ Fixed bug จากที่ QA เจอ หรือที่พบเจอจากการ Run regression test แล้ว Fail ก็จะเป็นการวน Loop ตั้งแต่ต้นจนจบ จนกระทั้ง ทุกอย่างผ่านหมดและฝ่ายธุรกิจฟันธงมาว่าเอาขึ้น Production ได้ เป็นการยืนยันว่าสามารถนำเอา Source code version สุดท้ายนี้ขึ้นไปที่ Production environment ได้

3. หากองค์กรต้องการนำเอา DevOps และ DI/CD เข้ามาเป็นส่วนเสริมในกระบวนการพัฒนาซอฟต์แวร์จะต้องทำอย่างไร

(VersionOne 2015) VersionOne ได้ทำการสรุปวิธีการในการนำเอาแนวคิด DevOps เข้ามาประยุกต์ใช้ในองค์กร มี 7 ขั้นตอน (7 Ways to Get Started with DevOps Today) ดังต่อไปนี้

3.1 Invite Your Operations Team Into Your Development Process

ทำการเชิญชวนฝ่ายการดำเนินงาน (Operation Team) เข้ามาร่วมในกระบวนการพัฒนาซอฟต์แวร์ด้วย และทำการเปลี่ยนที่ละน้อย ๆ เช่นการประชุมทุก ๆ เช้า (Standup Meeting) กับฝ่ายการดำเนินงาน หรือการให้พนักงานจากฝ่ายการพัฒนา (Development Team) ไปทำงานร่วมกันกับฝ่ายการดำเนินงาน เมื่อมีการพูดคุยหรือการทำ Deploy ซอฟต์แวร์

3.2 Visualize the Work Together

เพิ่มขั้นตอนการทำงานของฝ่ายการพัฒนาและฝ่ายการดำเนินงาน ให้มีการทำงานร่วมกัน เช่นการทำบอร์ดขั้นตอนการทำงานของทั้งสองฝ่าย

3.3 Automate Your Test/Build Process

เปลี่ยนวิธีการทำงานจากแบบ Manual ให้เป็นแบบอัตโนมัติ ตั้งแต่การ Pull source code จาก Repository, การ Compile code, การ Run unit test, การทำการ Package เพื่อให้พร้อม Deploy และการทำการ Deploy

3.4 Create a Deployment Plan

ทำการวางแผนการ Deploy โดยการตั้งคำถามเพื่อให้เห็นจุดบอดเพื่อจะได้นำไปปรับปรุงในครั้งหน้า เช่น การทำการ Deploy ซอฟต์แวร์ในช่วงดึก ๆ ดีหรือไม่ และเมื่อทำการ Deploy ซอฟต์แวร์จะมีปัญหาที่ไม่คาดคิดเกิดขึ้นเสมอ แล้วจะทำการแก้ไขให้ซอฟต์แวร์ทำงานได้ไปก่อนหรือไม่ ซึ่งคำถามเหล่านี้มีส่วนเกี่ยวข้องกับหลาย ๆ ฝ่าย ดังนั้นจ้องทำการวางแผนและลงมือทำ เมื่อเกิดปัญหาต้องพร้อมที่จะแก้ไขปรับปรุงต่อไป

3.5 Identify Fragile Systems

หากลองพิจารณาการวางแผนสำหรับการ Deploy ซอฟต์แวร์ในข้อที่ 3.4 อาจจะมีซอฟต์แวร์อย่างน้อยหนึ่งตัวที่ทำให้เกิดปัญหาหรืออาจเกิดจากตัว Server ที่ทำการ Configuration ไม่ถูกต้อง ซึ่งสามารถนำเอาข้อปฏิบัติของ DevOps เข้ามาช่วยในการทำงานทั้งในเรื่องของการทำงานแบบอัตโนมัติ และการทำงานที่ทำให้พนักงานทุกคนเห็นถึงข้อบกพร่องหรือปัญหาได้อย่างชัดเจน ซึ่งจะทำให้ขั้นตอนการ Deploy ซอฟต์แวร์นั้นมีความรวดเร็วมากยิ่งขึ้น

3.6 Smooth Out Wait States

การจัดการกับขั้นตอนหรือซอฟต์แวร์ที่ต้องรอเวลานาน ๆ ในการทำงาน เช่นการ Approve จากฝ่ายอื่น ๆ ถึงจะสามารถดำเนินงานตามขั้นตอนต่อไปได้ ให้นำเอาขั้นตอนหรือซอฟต์แวร์ในส่วนนั้นมาปรึกษากันระหว่างฝ่ายการพัฒนาและฝ่ายการดำเนินงาน เพื่อหาทางออกที่จะช่วยลดระยะเวลาการรอให้น้อยลง

3.7 Link Your Work to Your Value

จาก 6 ข้อที่กล่าวมาเป็นขั้นตอนในส่วนของ Technical ทั้งหมด แต่สิ่งที่เป็นหัวใจหลักของ DevOps นั้นมันคือการส่งมอบสิ่งที่มีคุณค่าไปยังผู้ใช้งานอย่างต่อเนื่อง รวมทั้งสิ่งต่าง ๆ ที่จำเป็นต่อขั้นตอนการ Deploy ซอฟต์แวร์ด้วย

อ้างอิง

Anon. n.d. “DevOps คืออะไร - Amazon Web Services (AWS).” Retrieved February 1, 2021 (https://aws.amazon.com/th/devops/what-is-devops/).

Gene, Kim, Humble Jez, Debois Patrick, and Willis John. 2016. *DevOps Handbook : How to Create World-Class Agility, Reliability, and Security in Technology Organizations*.

Gene, Kim, Behr Kevin, and Spafford George. 2013. *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win*.

Puisungsoen, Somkiat. 2014. “ความแตกต่างระหว่าง Continuous Delivery กับ Continuous Deployment.” Retrieved February 1, 2021 (https://www.somkiat.cc/continuous-delivery-vs-continuous-deployment/).

Softmelt. n.d. “การพัฒนาระบบตามแนวทาง CI/CD และ DevOps คืออะไร?” Retrieved February 1, 2021 (https://www.softmelt.com/article.php?id=664).

VersionOne. 2015. “7 Ways to Get Started with DevOps Today | Digital.Ai.” Retrieved February 1, 2021 (https://digital.ai/catalyst-blog/7-ways-to-get-started-with-devops-today).