1. **DevOps คืออะไร** [1]

DevOps คือการผสมผสานแนวความคิดเชิงวัฒนธรรม แนวทางปฏิบัติ และเครื่องมือต่างๆ ที่ช่วยเพิ่มความสามารถขององค์กรในการส่งมอบแอปพลิเคชันและบริการอย่างรวดเร็ว โดยพัฒนาและปรับปรุงผลิตภัณฑ์ต่างๆ ให้เร็วกว่ากระบวนการการพัฒนาซอฟต์แวร์และการจัดการโครงสร้างพื้นฐานแบบดั้งเดิม ความรวดเร็วนี้ช่วยให้องค์กรสามารถให้บริการแก่ลูกค้าของตนได้ดีขึ้น และสามารถแข่งขันในตลาดได้อย่างมีประสิทธิภาพมากขึ้น



DevOps ทำงานอย่างไร

สำหรับโมเดล DevOps ทีมพัฒนาและทีมปฏิบัติการจะไม่ทำงานแบบ “ต่างคนต่างทำ” อีกต่อไป บางครั้ง ทั้งสองทีมจะจับมือร่วมงานเป็นทีมเดียวกันโดยที่เหล่าวิศวกรจะทำงานตลอดวงจรการทำงานของแอปพลิเคชัน ตั้งแต่การพัฒนาและการทดสอบไปจนถึงการปรับใช้และการปฏิบัติงาน และพัฒนาขอบเขตความสามารถที่ไม่ได้จำกัดอยู่ที่การทำงานเพียงอย่างเดียว

ในบางโมเดลของ DevOps ทีมประกันคุณภาพและทีมรักษาความปลอดภัยอาจทำงานรวมกับทีมพัฒนาและทีมปฏิบัติการอย่างใกล้ชิดยิ่งขึ้นตลอดวงจรการทำงานของแอปพลิเคชัน ในเวลาที่การรักษาความปลอดภัยเป็นจุดสำคัญของทุกคนในทีม DevOps เราจะเรียกว่า DevSecOps

ทีมต่างๆ ใช้ข้อปฏิบัติในการเปลี่ยนกระบวนการต่างๆ ที่เคยทำงานแบบแมนนวลและเชื่องช้าให้ทำงานอัตโนมัติ พวกเขาใช้ชุดเทคโนโลยีและเครื่องมือต่างๆ ที่ช่วยให้พวกเขาดำเนินการและพัฒนาแอปพลิเคชันได้อย่างรวดเร็วและเชื่อถือได้ นอกจากนั้น เครื่องมือเหล่านี้ยังช่วยให้วิศวกรเหล่านี้ทำงานได้อย่างอิสระ (เช่น การปรับใช้โค้ด หรือการจัดเตรียมโครงสร้างพื้นฐาน) ที่โดยปกติแล้วจำเป็นต้องได้รับความช่วยเหลือจากทีมอื่น พร้อมทั้งยังช่วยทีมงานทำงานได้รวดเร็วยิ่งขึ้นอีกด้วย

## 

## ประโยชน์ของ DevOps

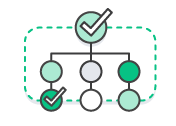
### ความรวดเร็ว

ดำเนินการอย่างรวดเร็วเพื่อให้คุณสามารถสร้างสรรค์นวัตกรรมให้แก่ลูกค้าได้รวดเร็วยิ่งขึ้น ปรับตัวต่อการเปลี่ยนแปลงของตลาดได้ดียิ่งขึ้น และมีประสิทธิภาพยิ่งขึ้นในการกระตุ้นให้เกิดผลลัพธ์ทางธุรกิจ โมเดล DevOps จะทำให้นักพัฒนาและทีมปฏิบัติการของคุณบรรลุผลดังกล่าวได้ ตัวอย่างเช่น [ไมโครเซอร์วิส](https://aws.amazon.com/th/devops/what-is-devops/#microservices)และ[การส่งมอบอย่างต่อเนื่อง](https://aws.amazon.com/devops/continuous-delivery/)จะทำให้ทีมมีความเป็นเจ้าของบริการและออกอัปเดตให้แก่ลูกค้าได้รวดเร็วยิ่งขึ้น



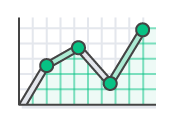
### การส่งมอบอย่างรวดเร็ว

ออกรุ่นใหม่ๆ ถี่ขึ้นและรวดเร็วยิ่งขึ้น เพื่อให้คุณสามารถสร้างสรรค์นวัตกรรมและปรับปรุงผลิตภัณฑ์ของคุณได้รวดเร็วยิ่งขึ้น ยิ่งคุณสามารถออกคุณสมบัติใหม่ๆ และแก้ไขจุดบกพร่องได้เร็วเท่าไร คุณก็ยิ่งตอบสนองความต้องการของลูกค้าและสร้างความได้เปรียบในการแข่งขันได้เร็วเท่านั้น [การบูรณาการอย่างต่อเนื่อง](https://aws.amazon.com/devops/continuous-integration/)และ[การส่งมอบอย่างต่อเนื่อง](https://aws.amazon.com/devops/continuous-delivery/) เป็นข้อปฏิบัติที่ทำให้กระบวนการออกซอฟต์แวร์ทำงานอัตโนมัติตั้งแต่การสร้างไปจนถึงการปรับใช้



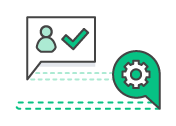
### ความเชื่อถือได้

ทำให้แน่ใจถึงคุณภาพของการอัปเดตแอปพลิเคชันและการเปลี่ยนแปลงโครงสร้างพื้นฐาน เพื่อให้คุณสามารถส่งมอบได้อย่างน่าเชื่อถือในความเร็วที่มากขึ้น ในขณะที่ยังคงรักษาประสบการณ์ใช้งานเชิงบวกของผู้ใช้ปลายทาง ใช้ข้อปฏิบัติอย่างเช่น [การบูรณาการอย่างต่อเนื่อง](https://aws.amazon.com/devops/continuous-integration/) และ[การส่งมอบอย่างต่อเนื่อง](https://aws.amazon.com/devops/continuous-delivery/) เพื่อทดสอบว่าการเปลี่ยนแปลงนั้นทำงานได้และปลอดภัยหรือไม่ ข้อปฏิบัติใน[การตรวจสอบและการบันทึก](https://aws.amazon.com/th/devops/what-is-devops/#monitoring)จะช่วยให้คุณทราบถึงประสิทธิภาพการทำงานในทันที



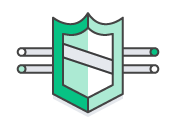
### ขนาด

ดำเนินการและจัดการโครงสร้างพื้นฐานและกระบวนการพัฒนาของคุณในขนาดต่างๆ การทำงานอัตโนมัติและความสอดคล้องจะช่วยให้คุณจัดการระบบที่ซับซ้อนหรือมีการเปลี่ยนแปลงได้อย่างมีประสิทธิภาพ โดยมีความเสี่ยงที่ลดลง ตัวอย่างเช่น [โครงสร้างพื้นฐานเป็นโค้ด](https://aws.amazon.com/th/devops/what-is-devops/#iac)จะช่วยคุณในการจัดการสภาพแวดล้อมในการพัฒนา การทดสอบ และการทำงานจริงในลักษณะที่ทำซ้ำได้และมีประสิทธิภาพยิ่งขึ้น



### การทำงานร่วมกันที่ปรับปรุงดีขึ้น

สร้างทีมงานที่มีประสิทธิผลมากขึ้นภายใต้โมเดลเชิงวัฒนธรรมของ DevOps ซึ่งให้ความสำคัญกับคุณ เช่น ความเป็นเจ้าของและความรับผิดชอบ นักพัฒนาและทีมปฏิบัติการจะ[ทำงานร่วมกัน](https://aws.amazon.com/th/devops/what-is-devops/#communication)อย่างใกล้ชิด รับผิดชอบร่วมกันในหลายกรณี และนำลำดับการทำงานมารวมเข้าด้วยกัน ซึ่งจะช่วยลดความไร้ประสิทธิภาพและประหยัดเวลา (เช่น ลดระยะเวลาการส่งมอบระหว่างนักพัฒนากับทีมปฏิบัติการ เขียนโค้ดที่คำนึงถึงสภาพแวดล้อมที่นำโค้ดไปใช้งาน)



### การรักษาความปลอดภัย

ดำเนินการอย่างรวดเร็วในขณะที่ยังคงรักษาการควบคุมและการปฏิบัติสอดคล้องกับกฎระเบียบ คุณสามารถรับเอาโมเดล DevOps มาใช้โดยไม่ต้องสูญเสียความปลอดภัยโดยใช้นโยบายการปฏิบัติตามกฎระเบียบที่ทำงานอัตโนมัติ การควบคุมโดยละเอียด และเทคนิคการจัดการการกำหนดค่า ตัวอย่างเช่น คุณสามารถใช้โครงสร้างพื้นฐานเป็นโค้ดและ[นโยบายเป็นโค้ด](https://aws.amazon.com/th/devops/what-is-devops/#policyascode) เพื่อกำหนดและติดตามการปฏิบัติตามกฎระเบียบในขนาดต่างๆ

**DevOps มีความสัมพันธ์กับการพัฒนาซอฟต์แวร์อย่างไร**[2]

1. ลดปัญหาการทำงานที่ซ้ำซ้อน   
   ถึงแม้แต่ละทีมจะทำงานแยกกัน แต่คนไอทีเองก็มักจะพบปัญหาเรื่องการทำงานที่ซ้ำซ้อนกันอยู่ในบางส่วน เช่น เมื่อ Developer พัฒนา Software มักจะมีระบบ Install / Uninstall มาด้วย และเมื่อส่งต่องานมาที่ฝั่ง Infrastructure ซึ่งมีหน้าที่พิจารณาปัจจัยอื่นๆ ที่เกี่ยวข้อง เช่น เรื่อง System Environment หากพิจารณาแล้วพบว่ามีบางส่วนของ Software ที่ยังไม่สมบูรณ์และจำเป็นต้องได้รับการแก้ไขให้เหมาะสม งานในส่วนนี้ก็จะไปซ้ำซ้อนกับสิ่งที่ทีม Developer ดูแลอยู่ และเมื่อถูกแก้ไขไปแล้วก็อาจส่งผลให้ระบบไม่ Stable และเกิดข้อผิดพลาดอื่นๆ ตามมา ซึ่งอาจจะต้องโยนงานกลับไปที่ทีม Developer แก้ไขปัญหาอีกที จะเห็นได้ว่า แต่ละทีมก็มีเป้าหมายและมุ่งทำแต่ในส่วนของตัวเอง เมื่อทำงานของตัวเองเสร็จแล้วก็มักจะ “โยนงาน” ไปให้อีกทีมดูแลต่อ พอปรับแก้เสร็จแล้วก็โยนกลับไปอีก สรุปแล้วคือโยนกลับกันไปมา พอจะเห็นภาพไหมว่า สุดท้ายก็ “ตีกัน”   
     
   แต่แนวทาง DevOps จะสามารถช่วยลดปัญหาเหล่านี้ โดยจะเข้ามาปรับเปลี่ยนขั้นตอนการทำงานให้มีประสิทธิภาพมากยิ่งขึ้น ทีมงานจะมีการวางแผนกันอย่างชัดเจนว่าใครทำอะไร ตรงส่วนใด เมื่อใด ทำให้ไม่เกิดปัญหาการทำงานที่ซ้ำซ้อนกันและลดความขัดแย้งระหว่างทีมงานลงได้
2. ช่วยพัฒนาศักยภาพให้คนไอทีมีความรู้ที่ครอบคลุมมากขึ้น   
   จากปัญหาที่เกิดจากการทำงานแยกทีมกัน ทำให้แต่ละทีมมีความรู้เชี่ยวชาญเฉพาะทางในแต่ละด้านไป ทางฝั่ง Developer ไม่มีความรู้ความเข้าใจงาน Infra. แบบลึกซึ้ง ในขณะที่ทีม Infra. เองก็ไม่ถนัดในการเขียนโปรแกรม แต่แนวคิด DevOps ได้เข้ามาช่วยปรับให้การ Utilize คนไอทีในแต่ละทีมให้เกิดประโยชน์และมีประสิทธิภาพมากขึ้น อาจจะจับคนทั้ง 2 ทีมมาเรียนรู้งานแต่ละส่วนร่วมกัน ทำให้เกิดแชร์ความรู้ซึ่งกันและกัน (Knowledge Sharing) หรืออย่างน้อยก็ได้เข้าใจงานของอีกทีมมากขึ้น
3. ทำให้ Flow การทำงานราบรื่น   
   ปกติเวลาทำงาน ผู้ดูแลระบบมักจะมีส่วนที่ต้อง Manual อยู่หลายขั้นตอน ซึ่งก็อาจส่งผลให้เกิดความล่าช้าและความผิดพลาดอยู่บ้าง แต่แนวคิด DevOps เป็นแนวทางที่มุ่งหวังให้งานทุกอย่างมันทำงานได้แบบราบรื่นด้วย Network Automation ทำให้ลดความผิดพลาด ในขณะเดียวกันก็เกิดความรวดเร็วในการทำงานอีกด้วย

# **CI / CD คือ**[3]

**Continuous Integration(CI)**คือ กระบวนการรวม source code ของคนในทีมพัฒนาเข้าด้วยกัน และมีการ test ด้วย test script เพื่อให้แน่ใจว่าไม่มี error ในส่วนใดๆ ของโปรแกรม แล้วถึงทำการ commit ไปที่ branch master อีกต่อนึง

**Continuous Delivery และ Continuous Deployment (CD)**อ่าว แล้วทำไม CD ถึงมี 2 ตัวละ แล้วต่างกันอย่างไร เด๋วจะมาอธิบายให้เข้าใจกัน

**Continuous Deployment**เห็นลงท้ายว่า Deployment ทุกคนก็น่าจะเดาได้ว่าต้องมีการ Deploy ขึ้น production แน่ๆ ซึ่งก็เป็นอย่างที่ทุกคนเข้าใจ คือ Continuous Deployment จะทำทุกขั้นตอน ตั้งแต่ complie build ไปจนถึง deploy ขึ้น production แบบอัตโนมัติทั้งหมด

ที่นี้มาถึงในส่วน **Continuous Delivery**มีการทำทุกขั้นตอนคล้ายกันกับ**Continuous Deployment**ต่างกันตรงที่จะไม่มีการ deploy ขึ้น production ขึ้นในทันที แต่จะเป็นการทำ manual deploy หรือจะเป็นแบบ one click deploy ก็ได้ หลังจาก QA หรือ ฝ่าย Business พอใจในตัว product ที่ทีมทำออกมา

**CI/CD มีความสัมพันธ์กับ DevOps อย่างไร**

หลักการของ CI/CD จะช่วยให้การทำงานแบบ DevOps มีประสิทธิภาพมากขึ้นเนื่องจากจะลดระยะเวลาในการพัฒนาลง ลดต้นทุน เมื่อเกิดการแก้ไขหรือข้อผิดพลาดก็จะสามารถแก้ไขได้อย่างทันที ซอฟต์แวร์นั้นจะมีความยืดหยุ่นที่สูงอีกด้วย

1. **หากองค์กรต้องการนำเอา DevOps และ CI/CD เข้ามาเป็นส่วนเสริมในกระบวนการพัฒนาซอฟต์แวร์จะต้องทำอย่างไรบ้าง**

ทุกคนจะทำการศึกษาและค้นคว้าวิธีการ DevOps และ CI/CD แล้วค่อยลงมือปฏิบัติ ให้ทำไปเรื่อยๆจนในทีมเกิดความเคยชิน

อ้างอิง

[1] “DevOps คืออะไร.” https://aws.amazon.com/th/devops/what-is-devops/.

[2] “รู้จัก ‘DevOps’ ให้มากขึ้น ‘ตำแหน่ง’ ที่องค์กรต่างมองหา.” https://www.techstarthailand.com/blog/detail/why-DevOps-skills-continue-to-be-in-high-demand/127.

[3] “เมื่อหัดทำ CI/CD ครั้งแรก.” medium.com/@thraithep.a/เมื่อหัดทำ-ci-cd-ครั้งแรก-f4f79151311c.