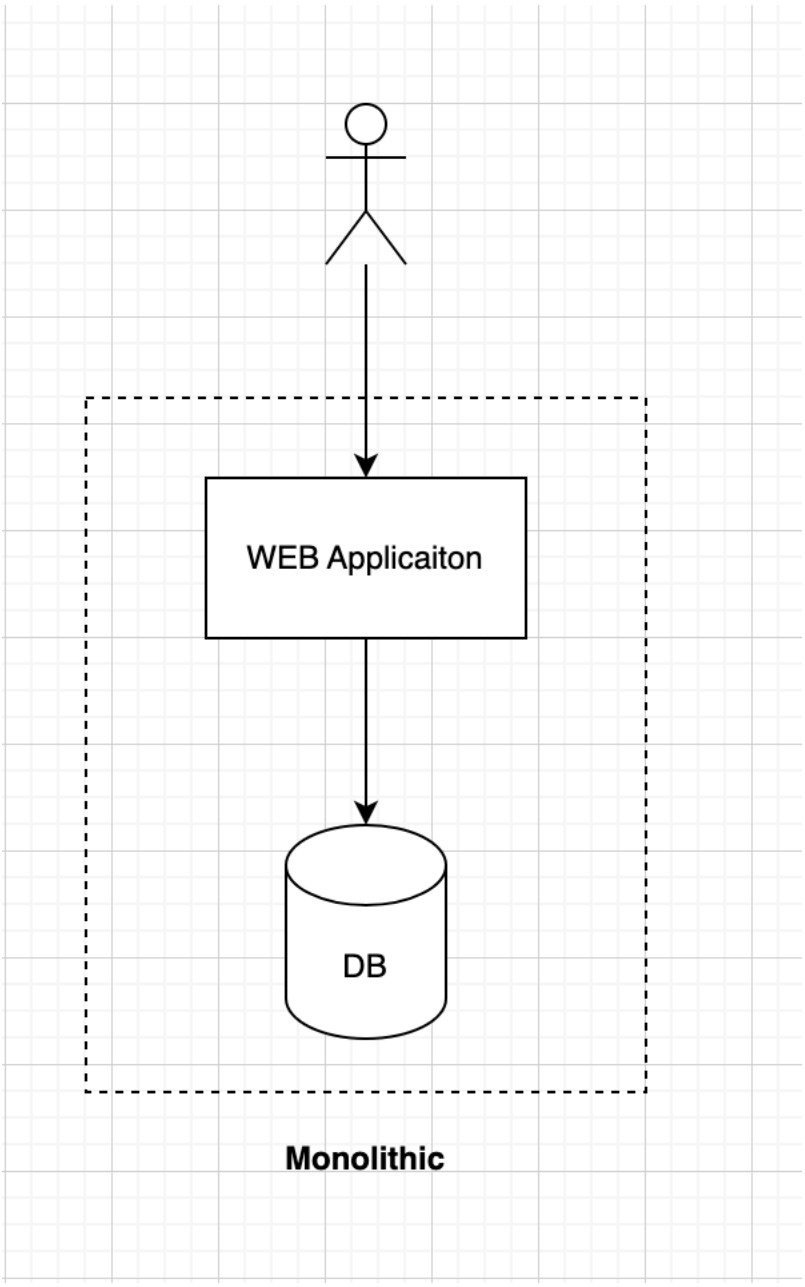
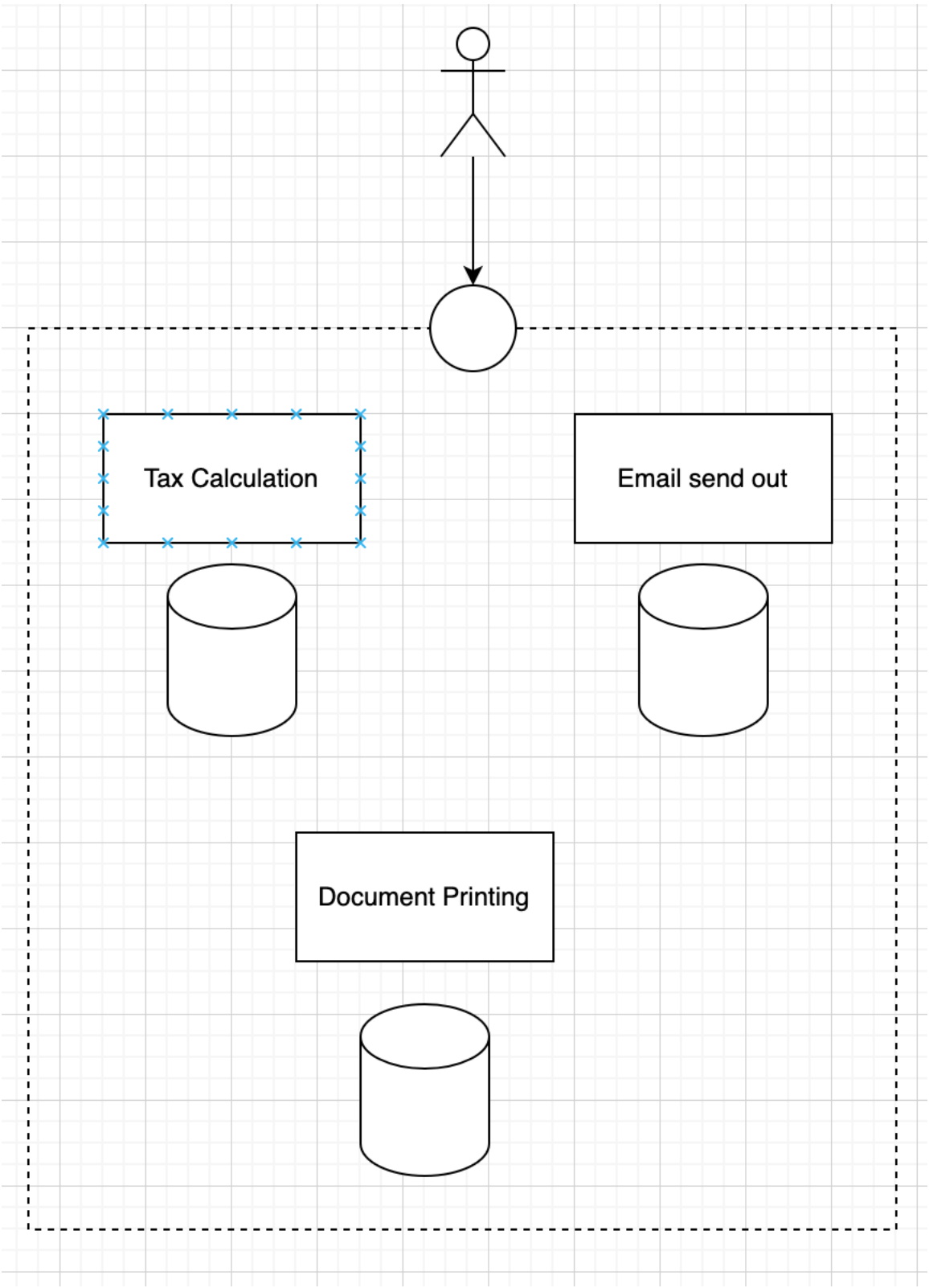


การออกแบบระบบในสมัยเก่านั้น จะนิยมใส่ทุกสิ่งอย่างลงไปรวมกันเป็น 1 ชิ้น เช่น DB, Source code รวมไปถึงการแบ่ง services domain จะไม่ได้มีความชัดเจน ซึ่งจะมีข้อดีคือการตั้งต้นพัฒนาจากโปรเจคเล็กๆ จะทำได้ดีและเร็ว แต่ข้อเสียคือ เมื่อถึงจุดหนึ่งมีผู้ใช้งานเยอะๆ จะทำให้ระบบรับ load ที่หนักมาๆ และอาจมีความเสี่ยงกับการล่มได้


Tax Calculation
Email send out
Document Printing

ดังนั้นจึงได้มีแนวคิดเรื่องการแบ่ง services แต่ละตัวออกไปให้เฉพาะกับ domain เพื่อสร้างความ independability ให้แก่การพัฒนา การเรียกใช้งาน และการทำงานของตัว services เอง ซึ่งจะทำให้การ scale นั้นเกิดขึ้นได้ง่าย และรวดเร็วกว่าการทำ monolithic

หากเป็นสมัยก่อนการตั้ง server ซักหนึ่งตัว จำเป็นต้องมี engineer ที่รู้ network, operating system, source code development รวมถึงการที่พัฒนา source code บนเครื่องใดเครื่องหนึ่ง อาจจะไม่สามารถทำกับเครื่องอื่นๆ ได้

การทำ containerization นั้นเกิดขึ้นเพื่อ abstract ส่วนที่ไม่จำเป็นสำหรับการพัฒนาและการเรียกใช้งานของตัว services ต่างๆ และโฟกัสกับการพัฒนาเท่านั้น ซึ่งจะลดปัญหาในส่วนของ

- การ run application ได้ทุกเครื่องโดยไม่ต้องสนใจ system dependency เช่น environment variable (เพราะเรา abstract ลงไปใน container แล้ว)
- การ scale ที่ทำได้ง่ายและรวดเร็วขึ้น

ดังนั้นเมื่อ services ต่างๆ ถูกแยกย่อยออกมาอย่างชัดเจนผ่าน microservices architecture การพัฒนาและใช้งาน application จะรวดเร็วขึ้นเมื่อแต่ละ microservices ใช้ container เพราะทั้งสองอย่างสร้างขึ้นมาเพื่อตอบโจทย์การ scale out