บทที่ 3
การจัดการโปรเซส

3.1 ความนำ

ในระบบคอมพิวเตอร์ เมื่อมีการเรียกใช้ระบบปฏิบัติการ และโปรแกรมประยุกต์ขึ้นมาทำงานก็จะเกิดงานต่างๆ ขึ้นมามากมาย งานเหล่านี้จะต้องเข้าไปทำงานในหน่วยประมวลผลกลาง เพื่อให้หน่วยประมวลผลกลางทำการประมวลผลให้ได้ออกมาซึ่งผลลัพธ์ของการทำงาน โดยคอมพิวเตอร์ส่วนใหญ่มักมีหน่วยประมวลผลกลางเพียงตัวเดียว แต่ต้องรองรับงานที่เกิดขึ้นมากมายเหล่านี้ หน้าที่หลักจึงตกอยู่ที่ระบบปฏิบัติการ จะต้องคอยจัดการงานเหล่านี้ให้สามารถเข้าไปทำงานในหน่วยปนระมวลผลกลางได้อย่างมีประสิทธิภาพ

3.2 ความหมายของโปรเซส

ผู้เขียนได้ศึกษาค้นคว้าพบว่า มีผู้รู้ได้กล่าวถึงความหมาย และให้คำนิยาม ตลอดจนอธิบายเกี่ยวกับโปรเซสเอาไว้ดังนี้

(น.ท.ไพศาล โมลิสกุลมงคล และคนอื่น ๆ, 2545) ได้กล่าวเอาไว้ว่าในช่วงแรกระบบคอมพิวเตอร์จะเอ็กซิคิวต์โปรแกรมครั้งละโปรแกรมเดียว ระบบจะความคุมอย่างสมบูรณ์ในโปรแกรมนั้น และโปรแกรมนั้นจะใช้รีซอร์สของระบบนั้นทั้งหมดแต่เพียงผู้เดียว แต่ระบบคอมพิวเตอร์ในปัจจุบันนี้สามารถโหลดโปรแกรมหลาย ๆ โปรแกรมไว้บนหน่วยความจำและเอ็กซิคิวต์ได้พร้อม ๆ กันหลาย ๆ โปรแกรมวิวัฒนาการนี้ทำให้ต้องมีการแบ่งโปรแกรมนี้เป็นช่วงๆ และต้องควบคุมอย่างดี นี่คือที่มาของคำว่า “โปรเซส” (Process) หรือก็คือโปรแกรมที่กำลังเอ็กซิคิวต์อยู่ โปรเซสนี้เป็นส่วนหนึ่งของระบบแบ่งเวลาที่จำเป็นต้องใช้รีซอร์สของระบบเพื่อทำงานในเสร็จ รีซอร์สที่ต้องการ เช่น เวลาซีพียู, หน่วยความจำ, ไฟล์, และอินพุต/เอาต์พุตดีไวซ์ อาจจะกำหนดเมื่อสร้างโปรเซส หรือเมื่อกำลังเอ็กซิคิวต์โปรเซสก็ได้ โปรเซสเป็นหน่วยหนึ่งของระบบ ในระบบหนึ่งๆ จะประกอบด้วยโปรเซสมากมาย โปรเซสเหล่านี้สามารถเอ็กซิคิวต์ทั้งโค้ดของระบบและโค้ดของผู้ใช้ได้พร้อม กันระบบปฏิบัติการมีหน้าที่ในการจัดการโปรเซสในระบบทั้งหมด ไม่ว่าจะเป็นการสร้าง และลบโปรเซสทั้งโปรเซสของระบบและของผู้ใช้ การจัดเวลาสำหรับโปรเซส รวมทั้งการซินโครไนซ์และการติดต่อสื่อสารภายในโปรเซส

(วิเชษฐ์ พลายมาศ, 2552) ได้อธิบายว่าโดยทั่วไปแล้ว กระบวนการก็คือ โปรแกรมที่ถูกกระทำการนั่นเอง แต่กระบวนการมีความหมายมากว่ารหัสโปรแกรมซึ่งบางครั้งเรียกว่าส่วนข้อความ (text section) รวมทั้งกิจกรรมปัจจุบันซึ่งถูกแทนด้วยค่าของตัวนับโปรแกรม (program counter) และค่ารีจิสเตอร์ (register) โดยปกติ กระบวนการจะประกอบด้วยค่าของกองซ้อนกระบวนการ (process stack) ซึ่งบรรจุด้วยข้อมูลชั่วคราว เช่น ค่าพารามิเตอร์ เลยที่อยู่กลับ (return address) และตัวแปลเฉพาะที่ (local variable) กับส่วนข้อมูล (data section) ซึ่งบรรจุด้วยค่าตัวแปรส่วนกลาง (global variable)

(สุจิตรา อดุลย์เกษม, 2552) ได้กล่าวเอาไว้ว่า ในสมัยแรกๆ นั้น ระบบคอมพิวเตอร์ทำงานในลักษณะโมโนโปรแกรมมิ่ง (mono-Programming) กล่าวคือ ณ ขณะใดขณะหนึ่งจะมีเพียงโปรแกรมเดียวเท่านั้นที่ทำงาน ซึ่งพบว่ามีข้อเสียคือทำให้ระบบใช้ทรัพยากรอย่างไม่เต็มประสิทธิภาพ เนื่องจากมีบางช่วงเวลาที่ทรัพยากรบางชนิดของระบบว่าง ไม่ถูกใช้งาน เช่น ขณะที่โปรแกรมอ่านข้อมูลนั้นอุปกรณ์รับ/ส่งข้อมูลถูกใช้งานแต่ซีพียูว่าง เมื่อโปรแกรมอ่านข้อมูลเสร็จเรียบร้อยและต้องการประมวลผล ซีพียูจะถูกใช้งานแต่อุปกรณ์รับส่ง/ส่งว่าง เป็นต้น ดังนั้นจึงได้มีความพยายามพัฒนาระบบคอมพิวเตอร์ที่สามารถทำได้หลายงานพร้อมกัน และสามารถใช้ทรัพยากรของระบบร่วมกัน ระบบดังกล่าวนี้ได้แบ่งงานหรือโปรแกรมออกเป็นส่วนย่อยที่ทำงานแตกต่างกัน เรียกว่า “โปรเซส” (process) เช่น โปรเซสการอ่านข้อมูล, โปรเซสการคำนวณ เป็นต้น จะเห็นว่าแต่โปรเซสต้องการใช้ทรัพยากรที่แตกต่างกัน จึงสามารถทำงานพร้อมกันได้ (กรณีที่แต่ละโปรเซสเป็นอิสระจากกัน) แน่นอนว่าระบบปฏิบัติการที่นำมาใช้กับระบบที่มีการทำงานหลายโปรเซสพร้อมกันนี้จำเป็นต้องมีการทำงานสลับซ้อนมากยิ่งขึ้น

จากความหมาย คำนิยาม และคำอธิบายจากผู้รู้ต่างๆ ข้างต้น สามารถสรุปได้ว่า โปรเซส (Process) หรือกระบวนการ คือ ส่วนของโปรแกรมที่กำลังทำงานอยู่ เนื่องจากการทำงานแบบหลายโปรแกรม จะมีการแบ่งโปรแกรมออกเป็นช่วงๆ โปรเซสเป็นส่วนหนึ่งของระบบแบ่งเวลาที่จำเป็นต้องใช้ทรัพยากรของระบบเพื่อให้ทำงานสำเร็จ เช่นเวลาในการครอบครองซีพียู หน่วยความจำ ไฟล์ และอุปกรณ์รับและแสดงผลข้อมูลเป็นต้น ระบบปฏิบัติการจะมีหน้าที่ในการจัดการโปรเซสทั้งหมด เช่นการสร้างโปรเซส การลบโปรเซส การจัดเวลาของโปรเซส การจัดการซิงโครไนซ์ และการติดต่อสื่อสารในโปรเซส

3.3 องค์ประกอบของโปรเซส

โปรเซสที่สมบูรณ์จะประกอบด้วย

3.3.1 หมายเลขโปรเซส (Process ID)

เป็นหมายเลขประจำโปรเซสเพื่อกำหนดลำดับการประมวลผล ซึ่งหมายเลขโปรเซสนี้ในแต่ละโปรเซสจะไม่เหมือนกัน

3.3.2 โค้ดโปรแกรม (Program Code)

เป็นโค้ดคำสั่งภาษาเครื่องที่คอมพิวเตอร์สามารถนำไปประมวลผลได้ทันที

3.3.3 ข้อมูล (Data)

ส่วนนี้จะเป็นตัวแปรโกลบอล (Global Variable) ที่เก็บข้อมูลเพื่อสนับสนุนให้โปรเซสสามารถประมวลผลโปรแกรมได้

3.3.4 บล็อกควบคุมโปรเซส (Process Control Block : PCB)

เป็นเนื้อที่ของหน่วยความจำ ที่ระบบปฏิบัติการกำหนดไว้เพื่อเก็บข้อมูลที่สำคัญของโปรเซสไว้ เมื่อระบบปฏิบัติการมอบเวลาซีพียูให้โปรเซสอื่นครอบครอง หลังจากที่โปรเซสเดิมได้เวลาซีพียู กลับมาครอบครองอีกครั้ง โปรเซสจะนำข้อมูลในส่วนนี้กลับมาใช้งาน

ภาพที่ 3.1 แสดงลักษณะข้อมูลใน บล็อกควบคุมโปรเซส
ที่มา (http://www.cs.jhu.edu/~yairamir/cs418/os2/sld007.htm, 2551)

ลักษณะข้อมูลใน บล็อกควบคุมโปรเซส มีดังภาพที่ 3.1 ซึ่งสามารถอธิบายข้อมูลต่างๆ ของบล็อกควบคุมโปรเซส ได้ดังนี้

1) พอยเตอร์ (Pointer) สำหรับชี้ตำแหน่งของโปรเซสที่อยู่ในหน่วยความจำ และตำแหน่งของทรัพยากรที่โปรเซสครอบครองอยู่

2) สถานะของโปรเซส (Process Status) แสดงสถานะของโปรเซสที่เป็นอยู่ในปัจจุบัน

3) หมายเลขโปรเซส (Process ID) เป็นหมายเลขประจำตัวของโปรเซส

4) ตัวนับจำนวน (Program Counter) เป็นตัวนับที่แสดงที่อยู่ของคำสั่งต่อไปที่จะถูกประมวลผล

5) รีจิสเตอร์ (Register) ทำหน้าที่เก็บข้อมูลสถานะระบบ เมื่อมีการขัดจังหวะ (Interrupt) เกิดขึ้นเพื่อทำให้โปรแกรมสามารถทำงานต่อไปได้เมื่อกลับมาทำงานอีกครั้ง รีจิสเตอร์จะมีค่าและประเภทที่เปลี่ยนแปลงได้ขึ้นอยู่กับสถาปัตยกรรมของคอมพิวเตอร์ ประเภทของรีจิสเตอร์คือ Accumulator, Index, Stack Pointer และรีจิสเตอร์ทั่วไป

6) ข้อมูลการจัดการเวลาซีพียู (CPU Scheduling Information) เป็นข้อมูลที่ประกอบด้วยลำดับความสำคัญของโปรเซส ที่ถูกกำหนดโดยระบบปฏิบัติการเมื่อโปรเซสถูกสร้างขึ้นมา สามารถเปลี่ยนค่าไปได้ ซึ่งโปรเซสใดที่มีความสำคัญมากระบบปฏิบัติการจะให้สิทธิมากกว่าโปรเซสอื่น เช่นให้เวลาซีพียูนานกว่า เป็นต้น

7) ข้อมูลการจัดการหน่วยความจำ (Memory Management Information) เป็นข้อมูลเกี่ยวกับหน่วยความจำที่ระบบปฏิบัติการกำหนดไว้ เช่นขนาดหน่วยความจำ ค่าของรีจิสเตอร์ Page table และ Segment table เป็นต้น

8) ข้อมูลแอ็กเคาต์ (Account Information) เป็นข้อมูลที่อาจประกอบด้วยจำนวน CPU เวลาที่กำหนด หมายเลขแอ็กเคาต์ หมายเลขโปรเซส และอื่นๆ

9) ข้อมูลสถานะอินพุต/เอาต์พุต (I/O Status Information) เป็นข้อมูลแสดงรายการของอุปกรณ์อินพุต/เอาต์พุตที่โปรเซสนี้ใช้ เป็นต้น

3.3.5 PSW (Program Status Word)

เป็นตัวควบคุมลำดับการประมวลผลคำสั่งของโปรเซส และเก็บข้อมูลเกี่ยวกับสถานะของโปรเซส ที่อยู่ของคำสั่งที่จะประมวลผลต่อไป โดย PSW จะทำหน้าที่คล้ายกับตัวนับจำนวน

3.3.6 คุณสมบัติของโปรเซส

คุณสมบัติของโปรเซสประกอบด้วย

1) ลำดับความสำคัญของโปรเซส (Priority) โดยเมื่อโปรเซสถูกสร้างขึ้นมา ลำดับความสำคัญของโปรเซสจะถูกกำหนดโดยระบบปฏิบัติทันที สามารถเปลี่ยนค่าไปได้ ซึ่งโปรเซสใดที่มีความสำคัญมากระบบปฏิบัติการจะให้สิทธิมากกว่าโปรเซสอื่น

2) อำนาจหน้าที่ของโปรเซส (Authority) เป็นการบอกอำนาจหน้าที่ของโปรเซสนั้นว่าสามารถทำอะไรได้บ้าง ใช้อุปกรณ์อะไรได้บ้าง เป็นต้น

3) คุณสมบัติอื่น เป็นคุณสมบัติอื่นใดที่ระบบปฏิบัติการกำหนดให้มี

3.4 สถานะของโปรเซส

ในขณะที่โปรเซสกำลังถูกประมวลผลอยู่นั้น โปรเซสอาจมีการเปลี่ยนสถานะก็ได้ การกำหนดสถานะของโปรเซสจะเป็นไปตามกิจกรรมที่กำลังดำเนินการอยู่ในขณะนั้น โดยแต่ละโปรเซสอาจมีสถานะต่อไปนี้

3.4.1 สถานะเริ่มต้น (New)

คือ สถานะที่กระบวนการเริ่มสร้างโปรเซสขึ้นมา

3.4.2 สถานะพร้อม (Ready)

คือ สถานะที่โปรเซสคอยที่จะเข้าครองซีพียู หรือพร้อมที่จะใช้ซีพียูทันทีที่ระบบปฏิบัติการมอบหมายให้ ในสถานะนี้ไม่มีการทำงานของโปรเซส แต่การทำงานจะเกิดขึ้นทันทีที่ได้ครองซีพียู โดยจะทำงานต่อจากงานเดิมที่ทำค้างไว้

3.4.3 สถานะทำงาน (Running)

คือ สถานะที่โปรเซสกำลังครอบครองซีพียู อยู่และใช้ซีพียูในการทำงาน โดยใช้ซีพียูทำงานตามคำสั่งในโปรแกรมของโปรเซส นั้น

3.4.4 สถานะรอคอย (Wait)

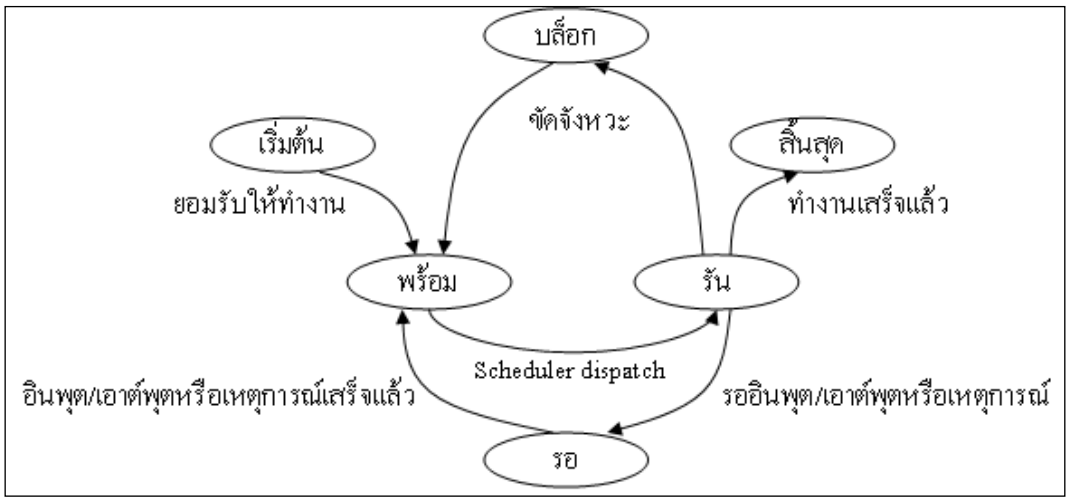
คือ สถานะที่โปรเซส กำลังรอคอยการทำงานอยู่ เช่น รอคอยการ อินพุต/เอาต์พุต หรือ รอคอยทรัพยากรที่จะนำมาทำโปรเซส

3.4.5 สถานะติดขัด (Blocked) หรือ สถานะพัก (Suspend)

คือ สถานะที่โปรเซส ติดต่อกับอุปกรณ์ หรือ คอยเหตุการณ์ใดเหตุการณ์หนึ่งให้เกิดขึ้น โปรเซส ในสถานะนี้ไม่จำเป็นต้องใช้ซีพียูและยังไม่พร้อมที่จะครอบครองซีพียู และเมื่อติดต่อกับอุปกรณ์เสร็จแล้ว หรือได้เหตุการณ์ที่คอยแล้วก็จะกลับเข้ามาในสถานะพร้อม เพื่อคอยการเข้าครองซีพียูเพื่อใช้ในการทำงานต่อไป

3.4.6 สถานะสิ้นสุด (Terminate)

คือ สถานะที่โปรเซสสิ้นสุดกระบวนการทำงาน อาจจะสิ้นสุดโดยการที่โปรเซสจบสิ้นกระบวนการแบบสมบูรณ์ หรือ ไม่สมบูรณ์ก็ได้



ภาพที่ 3.2 แสดงสถานะของโปรเซส

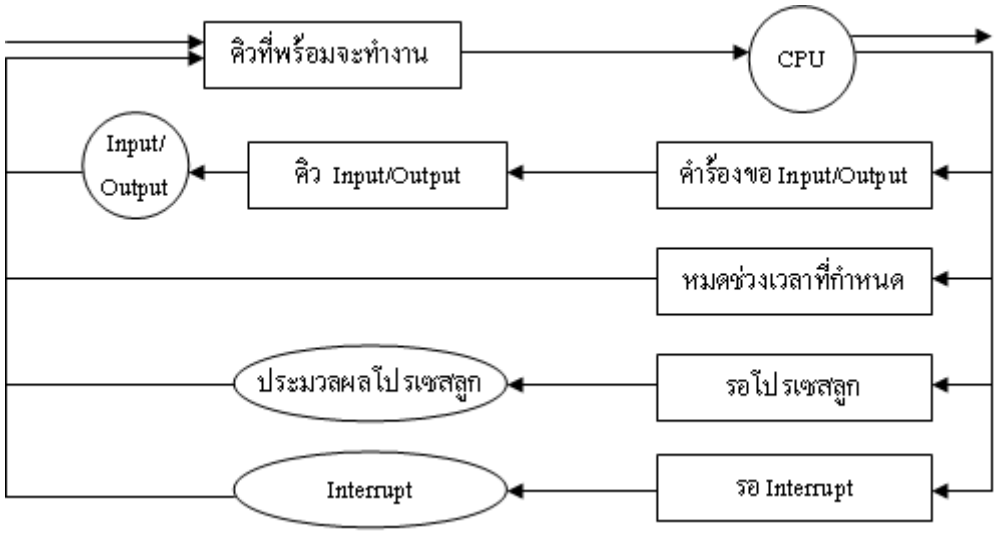
ที่มา (http://www.oknation.net/blog/print.php?id=303806, 2551)

3.5 การจัดเวลาโปรเซส

ในการที่จะประมวลผลโปรเซสหลายๆ โปรเซสพร้อมกันนั้น จำเป็นต้องจัดเวลาในการประมวลผลแต่ละโปรเซสให้ดี เพราะหากเป็นโปรเซสที่มีขนาดใหญ่จะใช้เวลามากซึ่งทำให้โปรเซสอื่นรอนาน ดังนั้นจึงมีการจัดเวลาให้แต่ละโปรเซส โดยจะแบ่งช่วงเวลาที่จะประมวลผลแต่ละโปรเซสให้เท่ากัน ระยะเวลาที่กำหนดนี้เรียกว่า เวลาควอนตัม (Quantum Time) ซึ่งเมื่อมีโปรเซสใดที่ครองเวลาในการประมวลผลนานกว่าเวลาควอนตัม ระบบปฏิบัติการจะเปลี่ยนให้โปรเซสอื่นที่ต่อคิวอยู่เข้ามาทำงานแทน ส่วนโปรเซสที่ยังไม่เสร็จก็จะถูกนำไปต่อคิวใหม่ โดยโปรเซสใหม่ที่เข้าไปแทนจะเปลี่ยนจากสถานะพร้อมเป็นสถานะทำงาน ส่วนโปรเซสเดิมจะเปลี่ยนจากสถานะทำงานเป็นสถานะพร้อม ยกเว้นว่าจะทำงานเสร็จจึงจะเปลี่ยนเป็นสถานะสิ้นสุด

โปรเซสที่อยู่ในสถานะบล็อกเช่นต้องการติดต่อกับอุปกรณ์อินพุต/เอาต์พุต หรือเกิดการขัดจังหวะระหว่างทำการโปรเซส ซึ่งโปรเซสจะถูกบังคับให้หยุดการทำงานและเปลี่ยนสถานะไปเป็นสถานะบล็อก ดังนั้นแทนที่จะให้ ซีพียูว่าง ก็จะนำโปรเซสที่อยู่ในสถานะบล็อกไปพักไว้ก่อนและให้อยู่ในสถานะรอคอย จนกว่าจะรับส่งข้อมูลหรือแก้การขัดจังหวะได้แล้ว จากนั้นจึงส่งโปรเซสไปไว้หัวคิวและกำหนดให้เป็นสถานะพร้อม

สำหรับคิวของโปรเซสที่รอการตอบสนองจากการใช้อุปกรณ์อินพุต/เอาต์พุต เรียกว่า ดีไวซ์คิว (Device Queue) ซึ่งอุปกรณ์แต่ละชิ้นจะมีคิวเป็นของตนเอง ที่มีพอยเตอร์เก็บส่วนหัวและส่วนท้ายของบล็อกควบคุมโปรเซสที่เรียกใช้อุปกรณ์นี้ และในส่วนของบล็อกควบคุมโปรเซส ก็จะมีพอยเตอร์ที่บอกว่าโปรเซสใดเป็นโปรเซสต่อไปที่จะทำการประมวลผล



ภาพที่ 3.3 แสดงดีไวซ์คิวของโปรเซส

ที่มา (http://www.cs.jhu.edu/~yairamir/cs418/os2/sld006.htm, 2551)

เมื่อซีพียูทำการประมวลผลโปรเซสจนเสร็จเรียบร้อยแล้ว จะต้องมีการสลับไปยังโปรเซสในคิวต่อไป การสลับคิวโปรเซสนี้เรียกว่าคอนเท็กซ์สวิตช์ (Context Switch) ในช่วงที่ทำคอนเท็กซ์สวิตช์จะไม่มีการทำงานใดๆ ส่วนเวลาที่ใช้ในการสวิตช์จะแตกต่างไปตามเครื่องคอมพิวเตอร์ เช่นความเร็วของหน่วยเก็บความจำ จำนวนรีจิสเตอร์ที่ถูกคัดลอกและคำสั่งพิเศษที่มีอยู่ในระบบ โดยทั่วไปความเร็วจะอยู่ระหว่าง 1-1,000 ไมโครวินาที

3.6 การติดต่อระหว่างโปรเซส

โปรเซสที่ทำการประมวลผลจะมีทั้งโปรเซสอิสระ (Independent Process) และโปรเซสสื่อประสาน (Cooperating Process) โดยโปรเซสอิสระจะมีการทำงานและการใช้ทรัพยากรที่ไม่เกี่ยวข้องกับโปรเซสอื่น หากโปรเซสนี้หายไปก็ไม่มีผลต่อโปรเซสอื่น แต่โปรเซสสื่อประสานจะเป็นโปรเซสที่เกี่ยวข้องกับโปรเซสอื่นๆ เช่นการส่งข้อมูล การติดต่อสื่อสาร การแบ่งปันทรัพยากร เป็นต้น และเมื่อมีโปรเซสใดโปรเซสหนึ่งหายไปก็อาจจะทำให้โปรเซสอื่นได้รับผลกระทบไปด้วย เหตุผลที่จำเป็นต้องมีโปรเซสสื่อประสานด้วยสาเหตุคือ

1) การแชร์ข้อมูลข่าวสาร เช่น กรณีที่ต้องการอ่านข้อมูลจากไฟล์ข้อมูลเดียวกันพร้อมกัน ระบบปฏิบัติการจะต้องจัดการทำงานของโปรเซสให้สัมพันธ์กัน เพื่อให้สามารถอ่านข้อมูลได้พร้อมกัน

2) การเพิ่มความเร็วในการคำนวณ ในกรณีที่คอมพิวเตอร์เป็นแบบระบบมัลติโปรเซสซิ่ง (Multiprocessing) หรือมีซีพียูมากกว่า 1 ตัว การแบ่งโปรเซสออกเป็นหลายๆ ส่วน แล้วให้ซีพียูแต่ละตัวประมวลผลโปรเซสไปพร้อมๆ กันหลายๆ โปรเซสจะทำให้การประมวลผลเสร็จเร็วขึ้น

3) ความสะดวก เกิดขึ้นจากการที่ผู้ใช้แต่ละคนที่ต้องการทำงานหลายอย่างในเวลาเดียวกัน เช่นแก้ไขข้อมูล พิมพ์ข้อมูล และประมวลผลข้อมูลพร้อมๆ กัน ดังนั้นจึงจำเป็นต้องมีการสื่อประสานกับโปรเซสอื่น

3.6.1 วิธีการติดต่อระหว่างโปรเซส

การติดต่อระหว่างโปรเซส สามารถกระทำได้ 2 วิธี คือ

1) การติดต่อทางตรง (Direct Communication)

การติดต่อแบบนี้จะต้องกำหนดชื่อเฉพาะในการติดต่อทั้งผู้รับและผู้ส่ง ต้องมีการสร้างลิงค์ (Link) ที่มีคุณสมบัติดังนี้

- การสร้างลิงค์จะเป็นแบบอัตโนมัติระหว่างคู่โปรเซสที่จะติดต่อ
- ลิงค์หนึ่งๆ จะมีความสัมพันธ์เฉพาะ 2 โปรเซสเท่านั้น
- ระหว่างโปรเซสทั้ง 2 จะมีเพียงลิงค์เดียวเท่านั้น
- ลิงค์นี้อาจเป็นได้ทั้งทิศทางเดียว หรือสองทางก็ได้ แต่โดยมากจะเป็นแบบสองทาง

2) การติดต่อทางอ้อม (Indirect Communication)

เป็นการติดต่อกันโดยผ่านเมล์บ็อกซ์ (Mail Box) หรือผ่านทางพอร์ต (Port) โดยข้อมูลที่ส่งไปมาระหว่างโปรเซส จะถูกนำมาเก็บไว้ในเมล์บ็อกซ์ก่อนที่จะส่งไปให้โปรเซสเป้าหมาย ซึ่งแต่ละเมล์บ็อกซ์จะมีหมายเลขไม่ซ้ำกัน ลิงค์แบบนี้มีคุณสมบัติคือ

- จะมีการสร้างลิงค์ระหว่างโปรเซสที่มีการแชร์เมล์บ็อกซ์เท่านั้น
- ลิงค์หนึ่งๆ อาจมีความสัมพันธ์กับโปรเซส มากกว่า 1 โปรเซส

- ระหว่างแต่ละโปรเซสอาจมีหลายลิงค์ที่แตกต่างกันได้ แต่ลิงค์จะมีเพียงเมล์บ็อกซ์เดียว

- การลิงค์อาจเป็นทิศทางเดียว หรือสองทิศทางก็ได้

3.6.2 รูปแบบของเมล์บ็อกซ์

การติดต่อระหว่างโปรเซสมีรูปแบบเมล์บ็อกซ์ที่ใช้ติดต่อ 3 รูปแบบ คือ

1) เมล์บ็อกซ์แบบคิว (Queue Mailbox)

เป็นการดึงข้อมูลออกตามลำดับก่อนหลังของการนำเข้า โดยข้อมูลใดส่งมาถึงก่อนก็จะถูกส่งออกไปก่อน เรียกการทำงานนี้ว่า FIFO (First in First out)

ภาพที่ 3.4 การทำงานแบบ FIFO (First in First out)

ที่มา (http://e-learning.mfu.ac.th/mflu/1302251/queue.htm, 2551)

2) เมล์บ็อกซ์แบบไปป์ (Pipe Mailbox)

จะมีลักษณะคล้ายแบบคิว คือ งานใดมาก่อนก็จะถูกส่งออกไปก่อน แต่ถ้าคิวเต็มจะไม่สามารถรับข้อมูลเข้ามาในคิวได้อีก ส่วนแบบไปป์นั้น เมื่อคิวเต็มจะมีการสามารถต่อคิวออกไปได้เรื่อยๆ ไม่จำกัด

ภาพที่ 3.5 การทำงานแบบไปป์

ที่มา (http://e-learning.mfu.ac.th/mflu/1302251/queue.htm, 2551)

3) เมล์บ็อกซ์แบบสแตก (Stack Mailbox)

ทำงานตรงข้ามกับแบบคิว คือ งานใดที่มาทีหลังจะถูกส่งออกไปก่อน เรียกรูปแบบนี้ว่า FILO (First in Last out)

ภาพที่ 3.6 การทำงานแบบสแตก

ที่มา (http://e-learning.mfu.ac.th/mflu/1302251/queue.htm, 2551)

3.6.3 การจัดบัฟเฟอร์

ในการสร้างลิงค์ นอกจากจะมีเส้นทางแล้วยังต้องมีเนื้อที่ส่วนหนึ่งที่จะเก็บข้อมูลในคิว โดยการเก็บข้อมูลนี้จะเป็นการเก็บชั่วคราว โดยพื้นฐานแล้วมีความจุ 3 ประเภท คือ

1) ความจุศูนย์ (Zero capacity)

เป็นคิวแบบมีความจุเป็น 0 คือจะไม่มีการเก็บข้อมูลไว้ในคิวเลย เมื่อคิวได้รับข้อมูลแล้วจะส่งไปยังเป้าหมายทันที ในกรณีนี้ผู้ส่งจะต้องรอจนกว่าผู้รับจะได้รับข้อมูล

2) ความจุแบบมีขอบเขต (Bounded capacity)

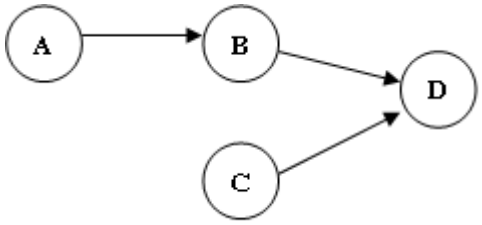
เป็นบัฟเฟอร์ที่มีขนาดความจุคงที่ เมื่อใดที่ข้อมูลยังไม่เต็ม บัฟเฟอร์จะรับข้อมูลมาอยู่ในคิวจนกว่าจะเต็ม เมื่อเต็มแล้วก็จะให้ข้อมูลรอจนกว่าจะมีที่ว่างจึงจะให้เข้ามาในคิว

3) ความจุแบบไม่จำกัด (Unbounded capacity)

เป็นคิวที่มีความจุไม่คงที่ สามารถรับข้อมูลได้ตลอดเวลา ทำให้ผู้ส่งไม่ต้องรอคอย

3.7 การซิงโครไนซ์โปรเซส

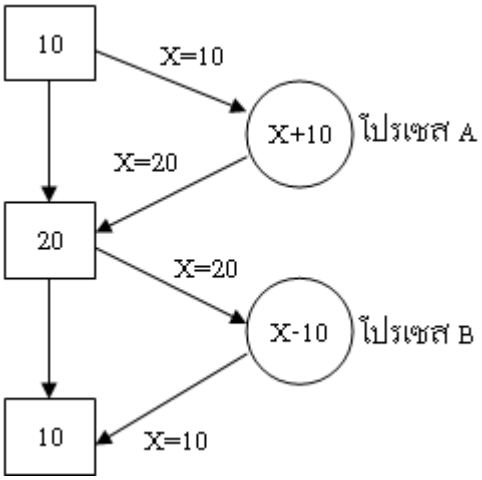
โดยทั่วไปแล้วโปรเซสจะไม่เกี่ยวข้องกัน โดยต่างก็ทำงานของตนและมีความเป็นอิสระกัน เรียกการทำงานแบบนี้ว่า อะซิงโครนัส (Asynchronous) แต่ก็มีบางโปรเซสที่มีการติดต่อกัน และยังมีการเข้าจังหวะของโปรเซสอีกด้วย การเข้าจังหวะของโปรเซสนั้นเป็นการทำงานที่เรียกว่า ซิงโครไนซ์ (Synchronize) การซิงโครไนซ์โปรเซส (Process Synchronization) หมายถึง การทำงานของโปรเซส 2 โปรเซสที่เกี่ยวข้องกัน เช่น ใช้ทรัพยากรร่วมกัน หรือใช้ข้อมูลต่อเนื่องกัน ทำให้ต้องมีการรอจังหวะให้สามารถทำงานได้ถูกต้อง ดังตัวอย่างต่อไปนี้



ภาพที่ 3.7 การซิงโครไนซ์โปรเซส
ที่มา (http://csnon04.blogspot.com/2008/03/2-thread.html, 2551)

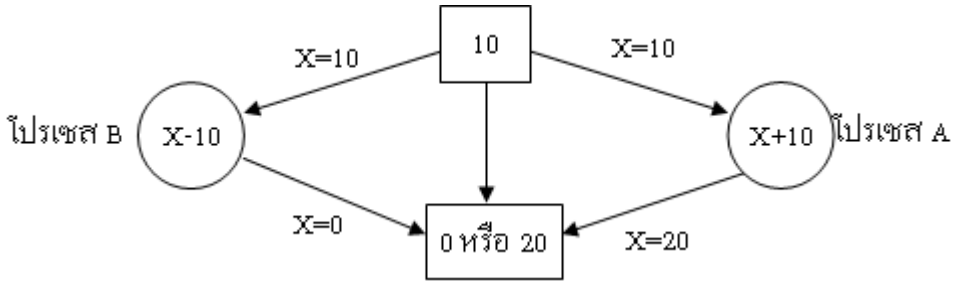
จากภาพที่ 3.7 จะเห็นว่าโปรเซส D จะประมวลผลได้เมื่อได้รับข้อมูลจากโปรเซส B และ C แต่โปรเซส B จะประมวลผลได้เมื่อได้รับผลจากโปรเซส A ดังนั้นเพื่อที่จะให้ได้โปรเซส D ระบบปฏิบัติการต้องจัดการประมวลผลโปรเซสทั้งหมดให้สอดคล้องกัน ซึ่งการจัดการนี้เรียกว่า การซิงโครไนซ์

ถ้ากำหนดค่า X = 10 โปรเซส A ทำการเพิ่มค่าขึ้น 10 (X -> X+10) และโปรเซส B ลดค่าลง 10 (X -> X-10) การทำงานของโปรเซสจะได้ผลดังภาพที่ 3.8



ภาพที่ 3.8 การทำงานของโปรเซส A ร่วมกับโปรเซส B

แต่ถ้าโปรเซส A และโปรเซส B ทำการประมวลผลโปรเซสทั้งหมดพร้อมกันก็อาจทำให้การทำงานของโปรเซสไม่สอดคล้องกัน เกิดความผิดพลาดในการประมวลผลได้ดังภาพที่ 3.9

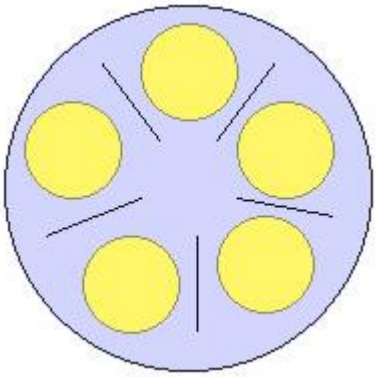


ภาพที่ 3.9 โปรเซส A ทำงานพร้อมกันกับโปรเซส B

การทำงานของโปรเซสบางครั้งอาจเกิดปัญหาขึ้นมาได้ ปัญหาการทำงานของโปรเซสที่อาจเกิดขึ้นมีดังนี้

1) The Dining Philosophers Problem

เป็นปัญหาในการติดตาย (Deadlock) อันเนื่องมาจากการใช้ทรัพยากรที่มีจำกัดร่วมกัน โดยปัญหานี้จำลองเหตุการณ์ได้จากการที่นักปราชญ์ได้ล้อมวงทานอาหาร โดยที่ข้างชามจะมีตะเกียบอยู่ข้างละ 1 อัน ดังภาพที่ 3.10



ภาพที่ 3.10 The Dining Philosophers Problem
ที่มา (http://msdn.microsoft.com/en-us/magazine/dd882512.aspx, 2551)

แต่การที่แต่ละคนจะทานได้ต้องใช้ตะเกียบ 2 อันพร้อมกัน ซึ่งจากภาพจะทานได้มากที่สุด 2 คนพร้อมกัน โดยคนที่เหลือจะต้องรอจนกว่าคนที่ทานอยู่จะทานเสร็จและวางตะเกียบลง แต่ในกรณีที่ต่างคนต่างหยิบพร้อมๆ กัน เช่นหยิบตะเกียบด้านซ้ายพร้อมกัน ทุกคนจะไม่สามารถทานอาหารได้เพราะต่างคนจะมีตะเกียบอันเดียว ทำให้เกิดการติดตายขึ้น

2) The Readers-Writers Problem

ในการอ่านและเขียนข้อมูลลงในอุปกรณ์บันทึกข้อมูล ปัญหาจะเกิดขึ้นเมื่อมีความต้องการอ่านและเขียนข้อมูลพร้อมๆ กัน เพราะหากมีการเขียนแสดงว่ามีการเปลี่ยนแปลงข้อมูล ซึ่งหากมีการอ่านก็จะทำให้ได้ข้อมูลที่ผิดพลาด ดังนั้นจึงมีการแก้ปัญหาโดยการกำหนดการเขียนและอ่าน

ไม่ให้เกิดขึ้นพร้อมกัน โดยจะจัดลำดับงานตามเวลาที่งานมาถึง โดยหากเป็นช่วงที่อ่านข้อมูล ระบบปฏิบัติการจะกันไม่ให้มีการเขียน แต่ระหว่างนี้จะอนุญาตให้งานอื่นอ่านข้อมูลได้ด้วย

3) The Sleeping Barber Problem

เป็นการจำลองการประมวลผลโดยเปรียบเทียบกับร้านตัดผม โดยจำลองให้ช่างตัดผมเป็นซีพียู ส่วนลูกค้าจะเป็นโปรเซส ในช่วงที่ไม่มีลูกค้า ช่างตัดผมจะนอนรอบนเก้าอี้ตัดผม เมื่อมีลูกค้าเข้ามาลูกค้าจะปลุกช่างตัดผม จากนั้นช่างตัดผมจะลุกให้ลูกค้านั่งแล้วจึงเริ่มตัดผม (ทำการประมวลผล) แต่ถ้าในช่วงที่ยังตัดอยู่มีลูกค้าคนใหม่เข้ามาอีก ลูกค้าคนนั้นต้องนั่งคอยที่เก้าอี้ที่จัดให้ (โปรเซสรอในคิว) เมื่อตัดผมลูกค้าคนแรกแล้วจึงเรียกลูกค้าในคิวมาตัดต่อไป แต่ถ้ามีคนรอในร้านเต็มหมดแล้วจะไม่รับลูกค้าเข้ามาอีก โดยที่เหลือต้องรอนอกร้าน แต่ถ้าไม่มีลูกค้าเหลืออยู่เลย ช่างตัดผมก็จะมานอนที่เก้าอี้ตัดผมเพื่อรอลูกค้า การทำงานแบบนี้ ลูกค้า (โปรเซส) จะเข้ามาเพียงครั้งเดียวจนตัดผมเสร็จ แต่ช่างตัดผม (ซีพียู) จะวนทำงานเรื่อยๆ เพื่อตัดผมลูกค้าทุกคน

3.8 ความหมายของเทรด

เทรด (Thread) คือ ส่วนประกอบย่อยของโปรเซส หรืออาจเรียกว่า LWP (Light Weight Process) เทรดประกอบด้วยส่วนต่าง ๆ ดังนี้

1) หมายเลขเทรด (Thread ID) เป็นหมายเลขเทรดในโปรเซส

2) ตัวนับ เพื่อติดตามให้ทราบคำสั่งต่อไปที่จะประมวลผล

3) ชุดของรีจิสเตอร์ เพื่อเก็บค่าตัวแปรที่ทำงานอยู่

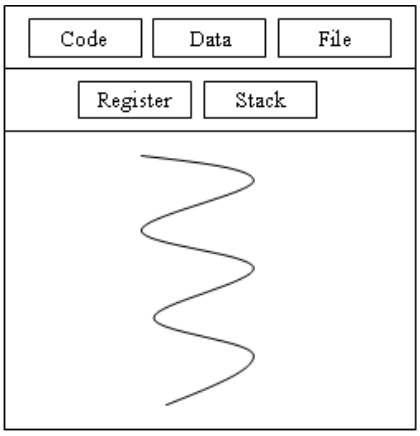
4) สแตก เพื่อเก็บประวัติการประมวลผล

3.9 ประเภทของเทรด

เทรด มี 2 ประเภท คือ

3.9.1 Single-threaded

เป็นเทรดของโปรเซส ที่ประกอบไปด้วย 1 เทรด

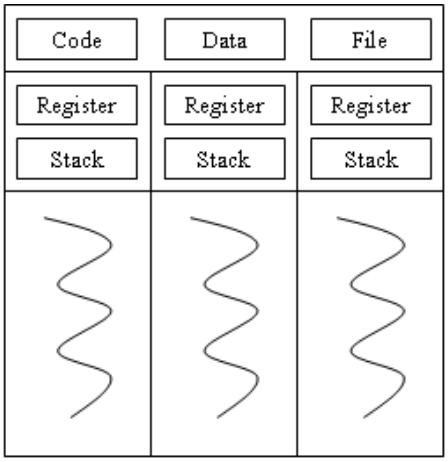


ภาพที่ 3.11 แสดงภาพจำลอง Single-threaded

ที่มี (http://csnon04.blogspot.com/2008_03_01_archive.html, 2551)

3.9.2 Multithreaded

เป็นเทรดของโปรเซส ที่ประกอบไปด้วยเทรดมากกว่า 1 เทรด



ภาพที่ 3.12 แสดงภาพจำลอง Multithreaded

ที่มี (http://csnon04.blogspot.com/2008_03_01_archive.html, 2551)

ภายในโปรเซสที่ประกอบด้วยเทรด จะมีการแบ่งปันโค้ด ข้อมูล และทรัพยากร เช่นไฟล์ อุปกรณ์ต่างๆ ถ้าเป็นโปรเซสที่มีเทรดเดียว (Heavyweight หรือ Single- Threaded) จะทำงานได้

ทีละงาน แต่โปรเซสที่มีหลายเทรด (Multithreaded) จะทำงานหลายๆ งานได้พร้อมกัน ดังนั้น โปรเซสที่มีหลายเทรดจึงมีข้อได้เปรียบโปรเซสที่มีเทรดเดียว ดังนี้

1) การตอบสนอง ในการทำงานหลายๆ งานพร้อมกันและมีงานบางอย่างถูกบล็อก หรือมีการทำงานยาวนาน ในระบบหลายเทรดนั้นจะยังทำให้งานอื่นๆ ทำงานต่อไปได้ เช่นการเปิดเว็บเพจ (Web Page) หลายๆ หน้าพร้อมกัน เว็บเพจจะโหลดข้อมูลเข้ามาได้เรื่อยๆ แม้จะมีบางหน้าที่กำลังเปิดภาพอยู่ก็ตาม

2) การแชร์ทรัพยากร โดยปกติเทรดจะมีการแชร์ทรัพยากรของโปรเซสอยู่แล้ว ทำให้โปรแกรมสามารถมีกิจกรรมหลายๆ อย่างของเทรดภายในแอดเดรส (Address) เดียวกัน

3) ความประหยัด เนื่องจากระบบหลายเทรด มีการใช้งานหน่วยความจำและทรัพยากรต่างๆ ร่วมกัน ทำให้เกิดความประหยัดค่าใช้จ่าย ในการทำงาน

4) การเอื้อประโยชน์ของสถาปัตยกรรมมัลติโปรเซสเซอร์ (Multiprocessor Architecture) การที่มีหลายเทรด ในโปรเซสเดียวทำให้สามารถแบ่งเทรดให้ ซีพียูหลายตัวช่วยในการประมวลผล ทำให้ประมวลผลโปรเซสได้รวดเร็วขึ้น ซึ่งหากเป็นโปรเซสที่มีเทรดเดียวจะใช้งาน ซีพียูเพียงเทรดเดียวเท่านั้น

บทสรุป

โปรเซส หรือกระบวนการ คือ ส่วนของโปรแกรมที่กำลังทำงานอยู่ ระบบปฏิบัติการมีหน้าที่ในการจัดการโปรเซสทั้งหมด เช่นการสร้างโปรเซส การลบโปรเซส การจัดเวลาของโปรเซส การจัดการซิงโครไนซ์ และการติดต่อสื่อสารในโปรเซส สถานะของโปรเซสมี 6 สถานะ คือ สถานะเริ่มต้น สถานะพร้อม สถานะทำงาน สถานะรอคอย สถานะติดขัด สถานะสิ้นสุด ส่วนประกอบย่อยของโปรเซสเรียกว่า เทรด มี 2 ประเภทคือ Single-threaded และ Multithreaded

คำถามท้ายบทที่ 3

1. จงอธิบายความหมายของคำต่อไปนี้
   1.1 โปรเซส
   1.2 ดีไวซ์คิว
   1.3 คอนเท็กซ์สวิตช์
   1.4 โปรเซสสื่อประสาน
   1.5 เทรด
2. โปรเซสสามารถมีสถานะใดได้บ้าง จงอธิบาย
3. วิธีการติดต่อของโปรเซสสื่อประสานมีกี่ประเภท แต่ละประเภทมีวิธีการอย่างไรบ้าง จงอธิบาย
4. การซิงโครไนซ์โปรเซสคืออะไร และวัตถุประสงค์ของการซิงโครไนซ์คืออะไร
5. ปัญหาการทำงานของโปรเซสมีอะไรบ้าง แต่ละปัญหามีลักษณะอย่างไร
6. Multithreaded คืออะไร และมีข้อดีอย่างไร

ปฏิบัติการ

ให้นักศึกษาทำปฏิบัติการที่ 2

เรื่อง ศึกษาการสั่งแสดงผลอักขระ A-Z ออกทางจอภาพ

โดยดาวน์โหลดแบบฟอร์มปฏิบัติการได้ที่ http://roongrote.crru.ac.th/CT2403.html

เอกสารอ้างอิง

น.ท.ไพศาล โมลิสกุลมงคล, ผศ.น.ท.ดร.ประสงค์ ปราณีตพลกรัง, น.ต.เมธา สุนทรศารทูล, น.ต.สุชาติ วีรกุลวัฒนา, ร.ท.อนุโชติ วุฒิพรพงษ์. (2545). **ระบบปฏิบัติการ (Operating Systems)**. กรุงเทพฯ : ไทยเจริญการพิมพ์.

วิเชษฐ์ พลาย. (2552). **ระบบปฏิบัติการ (Operating Systems)**. กรุงเทพฯ : ซีเอ็ดยูเคชั่น.

สุจิตรา อดุลย์เกษม. (2552). **ระบบปฏิบัติการ (Operating Systems)**. กรุงเทพฯ : โปรวิชั่น.

(2551). [Online]. Available HTTP: http://www.cs.jhu.edu/~yairamir/cs418/os2/sld007.htm.

(2551). [Online]. Available HTTP: http://www.oknation.net/blog/print.php?id=303806.

(2551). [Online]. Available HTTP: http://www.cs.jhu.edu/~yairamir/cs418/os2/sld006.htm.

(2551). [Online]. Available HTTP: http://e-learning.mfu.ac.th/mflu/1302251/queue.htm.

(2551). [Online]. Available HTTP: http://csnon04.blogspot.com/2008/03/2-thread.html.

(2551). [Online]. Available HTTP: http://msdn.microsoft.com/en-us/magazine/dd882512.aspx.

(2551). [Online]. Available HTTP: http://csnon04.blogspot.com/2008_03_01_archive.html.