บทที่ 3

การจัดการกระบวนการและเธรด

(Process Management and Threads)

หน้าที่หลักของซีพียู คือการดำเนินการกระบวนการจำนวนมาก ถึงแม้ว่าซีพียูจะเป็นศูนย์กลางเกี่ยวกับการดำเนินการประมวลผล แต่โปรแกรมผู้ใช้ยังต้องการกิจกรรมอื่น ๆ สำหรับระบบ เพื่อการประมวลผล กิจกรรมเหล่านี้เรียกว่ากระบวนการหรือโปรเซส คือหน่วยของงานในระบบ ที่กำลังดำเนินงานที่ซีพียู และขณะที่โปรเซสกำลังประมวลผลในซีพียูนั้นจะแบ่งตัวเองออกเป็นส่วนย่อย ๆ เรียกว่าเธรด เพื่อแบ่งปันการใช้ทรัพยากรร่วมกันเพื่อให้การประมวลผลมีประสิทธิภาพมากขึ้น

พื้นฐานของกระบวนการ

ระบบการทำงานของคอมพิวเตอร์ในปัจจุบันจะทำการโหลดโปรแกรมไว้ในหน่วยความจำได้หลายโปรแกรมและทำการประมวลผลไปพร้อม ๆ กันได้ ซึ่งในการประมวลผลของแต่ละโปรแกรมมีการแบ่งโปรแกรมออกเป็นส่วนย่อย ๆ จำนวนมาก ซึ่งถูกควบคุมโดยระบบปฏิบัติการ ส่วนย่อยแต่ละหน่วยของโปรแกรมนั้นเรียกว่ากระบวนการ (Process) ซึ่งจะได้อธิบายต่อไป

กระบวนการแบบลำดับ

ในขณะที่โปรแกรมกำลังประมวลผลจะถูกแตกให้เป็นกระบวนการย่อย ๆ หลาย กระบวนการโดยกระบวนการย่อยนั้นจะถูกควบคุมให้ดำเนินการประมวลผลในรูปแบบตามลำดับ (Sequential) ของขั้นตอนคำสั่งนั้น ๆ กระบวนการคือกลุ่มของกิจกรรมของโปรแกรมที่กำลังดำเนินการในซีพียู มีการบันทึกสถานะการทำงานไว้ในที่เก็บข้อมูลชั่วคราว (Temporality Data) เช่น ตัวแปรย่อย ตำแหน่งที่อยู่คำสั่งต่อไป ตัวแปรชั่วคราว และมีการบันทึกข้อมูลตัวแปรสาธารณะ (Global Variables) สำหรับโปรแกรมคือสิ่งที่อยู่นิ่งเฉยไม่ทีการเคลื่อนไหวใด ๆ บางครั้งเรียกว่า เอนทิตีพาสซีฟ (Passive Entity) เช่นไฟล์ของโปรแกรมที่บันทึกเป็นไฟล์อยู่ในดิสก์ และเรียกกระบวนการที่กำลังทำงานอยู่ ว่าเอนทิตีแอคทีฟ (Active Entity) มันอยู่ในหน่วยความจำหลักที่ไม่ได้อยู่นิ่งมีการเปลี่ยนสถานะตลอดเวลา มีตัวชี้คำสั่งถัดไปเพื่อจะถูกประมวลผลต่อ ถึงแม้ว่า 2 กระบวนการหรือหลายกระบวนการอาจจะเกี่ยวข้องกับโปรแกรมเดียวกันแต่จะถูกแยกการประมวลกันตามลำดับ โดยทั่วไปกระบวนการจะมีการสร้างกระบวนการย่อยให้เป็นกระบวนการลูกได้ซึ่งจะได้กล่าวในรายละเอียดต่อไป

สถานะของการบวนการ

ในขณะที่กระบวนการกำลังทำการประมวลผลมันจะเปลี่ยนสถานะตลอดเวลา กระบวนการจะมีการเปลี่ยนแปลงสถานะไปตามกิจกรรมและหน้าที่ของแต่ละกระบวนการจะถูกควบคุมโดยระบบปฏิบัติการ ที่ยอมให้มันเปลี่ยนสถานะที่ถูกจำกัดโดยกิจกรรมปัจจุบันของแต่กระบวนการโดยเป็นไปลำดับ อาจจะอยู่ในสถานะใดสถานะหนึ่งคือ พร้อม (Ready) กำลังประมวลผล (Running) หรือกำลังรอ (Waiting) กระบวการจะวนเวียนอยู่ 3 สถานะนี้จนกว่าจะทำงานเสร็จ อาจจะเรียกว่าวงจรทำงานของกระบวนการ (Process Life Cycle) แต่จะมีอีก 2 สถานะที่เกิดขึ้นครั้งแรกและครั้งสุดท้ายคือการเริ่มต้นกระบวนการใหม่ หรือการสิ้นสุดกระบวนการ ดังรูปที่ 3.1แสดงแผนภาพสถานะของกระบวนการดังนี้

รูปที่ 3.1 แผนภาพสถานะของกระบวนการ

จากรูปที่ 3.1สถานะการทำงานของกระบวนการต่าง ๆ เริ่มตั้งแต่กระบวนการใหม่ ถูกคัดเลือกให้เข้ามาอยู่ในสถานะพร้อม และได้รับการคัดเลือก 1 กระบวนการให้เข้าสู่สถานะกำลังประมวลผล ขณะกำลังประมวลผลอาจจะถูกสั่งให้หยุดชั่วคราวหรือหยุดรออยู่ในไอ/โอคิว และกลับเข้ามาอยู่ในสถานะพร้อมเพื่อวนรอบการทำงานจนเข้าสู่สถานะหยุดทำงานคือทำงานเสร็จแล้ว แต่ละสถานะมีการทำงานดังต่อไปนี้

กระบวนการใหม่ (New Process) มีกระบวนการมากมายที่ถูกสร้างขึ้นมาใหม่และต้องการเข้าประมวลผลต้องรออยู่ในสถานะกระบวนการใหม่

พร้อม (Ready) เป็นสถานะที่กระบวนการรออยู่ในแถวคอยของหน่วยความจำ กำลังรอคิวเพื่อถูกนำเข้าประมวลผลที่ซีพียู ซึ่งมีได้หลายกระบวนการ

กำลังประมวลผล (Running) กระบวนที่อยู่ในสถานะพร้อม จะถูกเลือกมาเพียงกระบวนการเดียวเพื่อส่งเข้าทำการประมวลผลที่ซีพียู จะอยู่ในสถานะกำลังประมวลผล ขณะที่กระบวนการกำลังประมวลผลเมื่อได้รับสัญญาณขัดจังหวะจากซีพียู ให้กระบวนการหยุดทำงานชั่วคราว และย้ายเข้าไปสู่สถานะพร้อมต่อไป

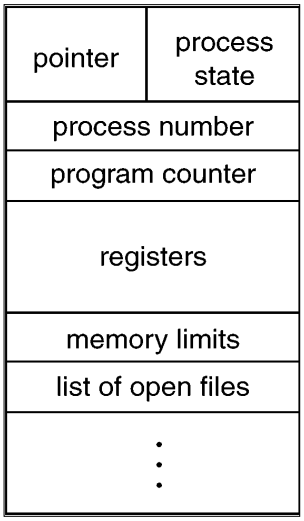
หยุดชั่วคราว (Block) ขณะที่กระบวนการกำลังประมวลผลอาจจะได้รับคำสั่งการขัดจังหวะจากซีพียู ให้กระบวนการหยุดทำงานชั่วคราว และย้ายเข้าไปสู่สถานะพร้อมต่อไป

กำลังรอ (Waiting) ขณะที่กระบวนการกำลังประมวลผล อาจจะมีความจำเป็นต้อง ทำไอ/โอ จะรออยู่ในคิวของ ไอ/โอหรือกระบวนการอาจจะกำลังรอสำหรับบางเหตุการณ์ได้

หยุดทำงาน (Terminated) ทุก ๆ กระบวนการเมื่อดำเนินงานเสร็จสิ้นจะหยุดการเข้าใช้งานที่ซีพียู ออกจากวงจรการทำงนของกระบวนการ เพื่อให้กระบวนการอื่น ๆ ได้เข้ามาทำการประมวลต่อไป

บล็อกควบคุมกระบวนการ (Process Control block)

แต่ละกระบวนการมีการถูกเก็บสถานะ การทำงานต่าง ๆ ไว้ในบล็อกควบคุมกระบวนการ (Process Control Block : PCB) บางครั้งเรียกสั้น ๆ ว่า พีซีบีโดยระบบปฏิบัติการจะทำการบันทึกรายละเอียดในการทำงานต่าง ๆ ของกระบวนการไว้ใน บล็อกควบคุมกระบวนการของแต่ละกระบวนการ ตามรูปที่ 3.2 ดังรายการต่อไปนี้



รูปที่ 3.2 บล็อกควบคุมกระบวนการ

เลขกระบวนการ (Process Number) คือเลขประจำตัวของกระบวนการ แต่ละกระบวนการต้องมีเลขประจำตัวที่ไม่ซ้ำกัน เพื่อให้ระบบปฏิบัติการได้รู้จักกระบวนการเหล่านี้ คล้ายกับเลขที่บัตรประชาชนที่ทุกคนต้องมีและต้องไม่ซ้ำกัน

สถานะกระบวนการ (Process State) เก็บสถานะปัจจุบันของกระบวนการอาจจะเป็นสถานะใหม่ พร้อม กำลังประมวลผล กำลังรอ หรือหยุดทำงานชั่วคราว

ตัวชี้คำสั่งโปรแกรม (Program Counter) เก็บตำแหน่งของคำสั่งหรือขั้นตอนถัดไปของกระบวนการที่จะดำเนินการต่อไป

รีจีสเตอร์ของซีพียู (CPU Registers) ระบบปฏิบัติการจะบันทึกเก็บค่ารีจีสเตอร์ที่ซีพียูใช้ในการทำงานของแต่ละกระบวนการ มีหลายชนิด คือ Accumulators, Index Registers, Stack Pointers และ General Purpose Registers บางรีจีสเตอร์จะมีความสัมพันธ์กับตัวชี้คำสั่งในโปรแกรมและสถานะของข้อมูลสารสนเทศของกระบวนการจะถูกเก็บทุกครั้งที่มีการขัดจังหวะการทำงานของกระบวนการ เพื่อให้กระบวนการนั้น ๆ กลับมาดำเนินงานต่อไปได้อย่างถูกต้องหลังจากที่ระบบได้จัดการกับการขัดจังหวะเรียบร้อยแล้ว

ข้อมูลการเข้าใช้ซีพียู (CPU Scheduling Information) ประกอบด้วยค่าของข้อมูลสำหรับกระบวนการที่จะได้เข้าทำงานที่ซีพียู เช่น ค่าของลำดับความสำคัญของกระบวนการเพื่อเป็นตัวชี้ในตารางเวลาของคิวการเข้าใช้ซีพียู และตารางเวลาของตัวแปรอื่น ๆ ที่เกี่ยวข้องกับการเข้าประมวลผลของกระบวนการ

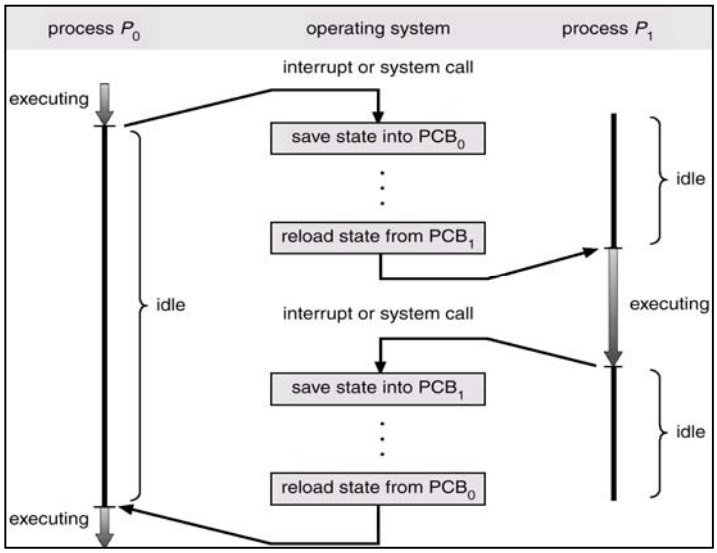
ข้อมูลการจัดการหน่วยความจำ (Memory Management Information) ประกอบด้วย รีจีสเตอร์ขอบเขต (Limit Register) หรือตารางการใช้เพ็จ (Page Tables) ในหน่วยความจำ

ข้อมูลบัญชีการใช้งาน (Accounting Information) ประกอบด้วย จำนวนของซีพียูที่ใช้ในการประมวลผล และเวลาที่ใช้จริง การจำกัดงาน จำนวนผู้ใช้หรือจำนวนกระบวนการที่ทำการประมวลผล

ข้อมูลสถานะของไอ/โอ (I/O Status Information) เก็บสถานะการทำงานต่าง ๆ ของอุปกรณ์ไอ/โอ การใช้ไฟล์ข้อมูลต่าง ๆ ที่เกี่ยวข้องในการประมวลผลของกระบวนการ

บล็อกควบคุมกระบวนการ เป็นที่เก็บข้อมูลการทำงานของแต่ละกระบวนการมีประโยชน์อย่างยิ่งในการที่ระบบทำการเปลี่ยนแปลงการดำเนินการจากกระบวนการหนึ่งไปยังอีกกระบวนการหนึ่ง ดังแสดงในรูปที่ 3.3 ในขณะที่ระบบกำลังทำงาน 2 กระบวนการคือ P_0 กับ P_1 เนื่องจาก ขณะใดขณะหนึ่งซีพียูทำการประมวลผลได้ครั้งละ 1 กระบวนการเท่านั้น ดังนั้น P_0 กับ P_1 จะสลับกัน

ประมวลที่ซีพียู ดังนี้ ขณะที่ P_0 กำลังประมวลผล P_1 รออยู่ เมื่อ P_0 หมดช่วงเวลา หรือขอทำไอ/โอจะถูกขัดจังหวะจากระบบปฏิบัติการและจะบันทึกสถานะ การทำงานของ P_0 ลงบล็อกควบคุมกระบวนการของ P_0 และอ่านสถานะการทำงานของ P_1 จากบล็อกควบคุมกระบวนการของ P_1 แล้วทำการประมวลผล P_1 โดยที่ P_0 หยุดรอ ทำสลับกัน ระหว่าง P_0 กับ P_1 ซ้ำ ๆ จนเสร็จงาน



รูปที่ 3.3 การสับเปลี่ยนซีพียูระหว่างกระบวนการ

กระบวนการที่ทำงานพร้อมกัน

กระบวนการในระบบสามารถทำการประมวลผลพร้อมกันได้หลายกระบวนการ หรือระบบมีการทำงานได้หลายงานบนซีพียูเดียวกัน เพื่อให้ได้ประโยชน์สูงสุดในการทำงานของระบบมีหลายเหตุผลสำหรับการยอมให้มีการประมวลผลร่วมกันดังนี้

1. ใช้ทรัพยากรร่วมกันทางกายภาพ (Physical Resource Sharing) ทรัพยากรทางกายภาพในระบบคอมพิวเตอร์มีจำกัด เช่น ซีพียู หน่วยความจำ ฮาร์ดดิสก์ และอุปกรณ์อื่น ๆ ในระบบที่มีผู้ใช้หลายคน จึงต้องมีการแบ่งปันการใช้ทรัพยากร หรือมีการใช้ทรัพยากรทางกายภาพร่วมกัน

2. ใช้ทรัพยากรร่วมกันทางตรรกะ (Logical Resources Sharing) ผู้ใช้หลาย ๆ ผู้ใช้อาจจะใช้ทรัพยากรบางอย่างร่วมกันทางตรรกะได้ เช่น ซอฟต์แวร์ คำสั่ง ข้อมูล เป็นต้น ซึ่งระบบจะต้องเตรียมสภาพแวดล้อมเพื่อยอมให้เข้าถึงชนิดของทรัพยากรนั้น ๆ ได้ในเชิงตรรกะ

3. เพิ่มความเร็วในการทำงาน (computation speedup) ถ้าต้องการให้งานทำเสร็จเร็วขึ้น เราต้องแบ่งงานออกเป็นส่วยย่อย (subtask) และให้แต่ละส่วนย่อยของงานนั้นประมวลผลคู่ขนานกันไป กับส่วนย่อยของงานอื่น ๆ ด้วย การเพิ่มความเร็วจะได้ผลดีมากถ้าระบบเป็นแบบหลายตัวประมวลผล

4. การแบ่งส่วน (Modulating) เราอาจจะต้องการที่จะสร้างระบบในแบบใหม่แบบการแบ่งส่วนโดยแบ่งการทำงานออกเป็นส่วน ๆ เพื่อให้การทำงานที่ง่ายขึ้น

5. ผู้ใช้สะดวก (Convenience User) ผู้ใช้งานส่วนใหญ่อาจจะมีหลายงานที่ทำบนคอมพิวเตอร์ ณ เวลาหนึ่งเช่น กำลังแก้ไขรายงาน กำลังพิมพ์ และกำลังการตรวจสอบโปรแกรม ผู้ใช้ต้องการให้มีการดำเนินการคู่ขนานพร้อมกันได้ทุก ๆ งาน ดังนั้นระบบต้องมีกลไกสำหรับการประสานกระบวนการ (Process Synchronization) ให้ทุกกระบวนการจะต้องประสานการทำงานระหว่างกระบวนการและการติดต่อสื่อสารระหว่างกระบวนการได้

การสร้างกระบวนการและยกเลิกกระบวนการ

เพื่อการประมวลผลกระบวนการให้ทำงานได้เร็วขึ้นและมีการทำงานร่วมกันด้วย ต้องมีกลไกสำหรับการสร้างกระบวนการ และการยกเลิกกระบวนการ

การสร้างกระบวนการ (Process Creation) กระบวนการที่ทำการประมวลผลอาจจะสร้างเป็นกระบวนการ ขึ้นมาใหม่ได้หลายกระบวนการโดยผ่านการเรียกระบบ กระบวนการที่กำลังทำการประมวลผลแล้วสร้างกระบวนการใหม่ขึ้นมาเรียกว่ากระบวนการหลักหรือกระบวนการแม่ (Parent Process) และกระบวนการใหม่ที่ถูกสร้างขึ้นมาเรียกว่ากระบวนการลูก (Child Process) แต่ละกระบวนการทีถูกสร้างขึ้นมาใหม่เหล่านี้อาจจะสร้างกระบวนการขึ้นมาใหม่ได้อีกต่อไปกระบวนการลูกจะกลายเป็นกระบวนการแม่ในระดับต่อ ๆ ไป

โดยทั่วไปการประมวลผลของกระบวนการจะต้องการใช้ทรัพยากรแน่นอนใจเพื่อการดำเนินงาน ให้เสร็จภาระของมัน เมื่อกระบวนการแม่สร้างกระบวนการลูกใหม่ขึ้นมา กระบวนการลูกอาจจะสามารถใช้ทรัพยากรโดยตรงจากระบบปฏิบัติการ หรือมันอาจจะถูกบังคับให้ใช้ทรัพยากรของกระบวนการแม่ ซึ่งกระบวนการแม่อาจจะต้องใช้ทรัพยากรของมันร่วมกันกับกระบวนการลูกของมันหลาย ๆ กระบวนการ หรืออาจจะแชร์บางทรัพยากรระหว่างกระบวนการลูกหลาย ๆ กระบวนการ ดังนั้นจึงต้องมีการจำกัดการสร้างกระบวนการลูก ระบบปฏิบัติการจะควบคุมไม่ให้กระบวนการสร้างกระบวนการใหม่ให้มีจำนวนมากเกินไปจนเป็นสาเหตุให้เกิดการแย่งกันใช้ทรัพยากรจนประสิทธิภาพการทำงานของระบบด้อยลง

เมื่อเริ่มต้นกระบวนการระบบปฏิบัติการจะจัดสรรทรัพยากรที่จำเป็นให้แก่กระบวนการต่าง ๆ ทั้งทางกายภาพและทางตรรกะ โดยทรัพยากรบางอย่างจากกระบวนการแม่จะส่งไปให้กระบวนการลูกได้โดยตรง การประมวลของกระบวนการแม่และกระบวนการลูกสามรถดำเนินการได้ดังนี้

- กระบวนการแม่และกระบวนการลูกทำการประมวลผลพร้อมกัน
- กระบวนการแม่รอจนกระทั่งกระบวนการลูกของมันทั้งหมดทำงานเสร็จแล้วจึงจะดำเนินการต่อ

การยกเลิกกระบวนการ (Process Termination) ในการทำงานปกติกระบวนการจะหยุดทำงานเมื่อทำงานเสร็จเรียบร้อยแล้ว และให้ระบบปฏิบัติการลบกระบวนการนั้นออกจากหน่วยความจำ แต่ถ้าเป็นกระบวนการลูกอาจจะต้องส่งคืนข้อมูลผลลัพธ์ ไปยังกระบวนการแม่ก่อน หรือบางครั้งการประมูลผลของกระบวนการแม่อาจจะไม่จำเป็นต้องใช้กระบวนการลูกอีกต่อไปจึงต้องยกเลิกกระบวนการลูก หรือบางครั้งกระบวนการต่าง ๆ สร้างกระบวนการลูกมากเกินไปทำให้ประสิทธิภาพการทำงานของระบบด้อยลง ระบบปฏิบัติการจะทำการยกเลิกบางกระบวนการ สำหรับอีกเหตุผลหนึ่งที่กระบวนการแม่ยกเลิกกระบวนการลูกคือกระบวนการลูกขอใช้ทรัพยากรมากเกินไปแต่กระบวนการแม่ไม่มีทรัพยากรเพียงพอที่จะสามารถแบ่งให้ใช้ได้

ความสัมพันธ์ระหว่างกระบวนการ (Relation Between Processes)

กระบวนการที่กำลังประมวลผล ในระบบอาจจะมีทั้งกระบวนการอิสระ (Independent Process) หรือกระบวนการร่วม (Co-operating Process) ประมวลผลของกระบวนการอิสระจะไม่มีผลต่อกระบวนการอื่น ๆ ที่กำลังประมวลผลในนระบบ คุณสมบัติของกระบวนการอิสระมีลักษณะดังต่อไปนี้

- ไม่มีการใช้สถานะของกระบวนการร่วมกับกระบวนการอื่น
- ผลลัพธ์ที่ได้จากการประมวลผลจะมีเพียงค่าเดียวของแต่ละข้อมูลของมัน
- ทำการประมวลผลกระบวนการซ้ำ ๆ กี่ครั้งจะยังคงให้ผลลัพธ์เดิม สำหรับข้อมูลเดิม
- การประมวลผลของกระบวนการ จะหยุดและเริ่มต้นประมวลผลเมื่อไร จะไม่มีผลกระทบต่อการทำงานของกระบวนการอื่น ๆ

สำหรับกระบวนการที่มีการแชร์ข้อมูลใด ๆ ร่วมกับ กระบวนการอื่น ๆ เรียกกระบวนการนั้นว่ากระบวนการร่วม การประมวลผลแต่ละกระบวนการจะมีผลกระทบต่อการประมวลผลของกระบวนการอื่น ๆ ในระบบโดยกระบวนการร่วมมีลักษณะดังต่อไปนี้

- สถานะของกระบวนการ ถูกใช้ร่วมกับกระบวนการ อื่น ๆ

- ผลลัพธ์ของการประมวลผลกระบวนการไม่สามารถคาดการณ์ได้เพราะผลลัพธ์ขึ้นอยู่กับลำดับของการประมวลผลจากกระบวนการอื่นตามลำดับ

- ผลลัพธ์ของของการประมวลผลกระบวนการไม่สามารถกำหนดได้ ถึงแม้จะเป็นข้อมูลเดียวกันก็ตาม

เธรด

เธรด (Threads) คือหน่วยกระบวนการพื้นฐานที่เข้าทำงานที่ซีพียูให้มีประสิทธิภาพ หรือเป็นส่วนย่อยของกระบวนการบางครั้งเรียกกระบวนการ ว่ากระบวนการหนัก (Heavyweight Process) และเรียกเธรดว่ากระบวนการเบา (Lightweight Process) เธรดประกอบด้วย หมายเลขเธรดของกระบวนการตัวชี้ตำแหน่งคำสั่งถัดไปที่จะประมวลผลรีจีสเตอร์เก็บค่าการทำงานต่าง ๆ สแตกเก็บประวัติการประมวลผล การประมวลผลกระบวนการในระบบปฏิบัติการยุคแรก ๆ มีการดำเนินการแบบ 1 กระบวนการมี 1 เธรด (Single Thread) ระบบปฏิบัติการยุคใหม่สามารถให้แต่ละกระบวนการมีได้หลายเธรด (Multithread) หรือเรียกว่ามัลติเธรด ในการทำงาของกระบวนการที่เป็นมัลติเธรด จะมีการใช้ทรัพยากรร่วมกันเช่นคำสั่ง ข้อมูล ไฟล์ อุปกรณ์ต่าง ๆ รวมทั้งทรัพยากรของระบบการทำงานปัจจุบันจะเป็นมัลติเธรดโดยที่กระบวนการจะมีการใช้หลาย ๆ เธรด แต่ละเธรดทำงานแตกต่างกันทำให้กระบวนการนั้นทำงานได้หลายอย่างพร้อม ๆ กันตัวอย่างเช่นกระบวนการที่มี 3 เธรด จะทำงาน 3 อย่างได้พร้อมกันเช่นกำลังโหลดข้อมูลจากอินเตอร์เน็ต กำลังแก้ไขรายงานที่ดำเนินงานบนไมโครซอฟต์เวิร์ด และกำลังแสดงรูปภาพ ในการทำงานแบ่งประเภทของเธรดได้ 2 ประเภทคือ

- เธรดเคอร์เนล (Kernel Thread) คือเธรดที่ได้รับการควบคุมจากระบบปฏิบัติการโดยตรง เคอร์เนลจะจัดการเกี่ยวกับเธรดภายในกลุ่มเคอร์เนลที่ทำงานร่วมกัน ในการจัดตารางเวลา การจัดพื้นที่ให้แก่เธรดต่าง ๆ รวมทั้งการจัดการเธรดในการประมวลผลถ้ามีเธรดใดถูกบลอกก์ เคอร์เนลสามารถนำเอาเธรดอื่น ๆ ของกระบวนการเข้ามาประมวลผลแทน และถ้าเป็นระบบหลายตัวประมวลผลจะจัดการให้เธรดต่าง ๆ ไปประมวลผลที่ซีพียู อื่นได้

- เธรดผู้ใช้ (User Thread) ในการทำงานจะได้รับการสนับสนุนจากเธรดเคอร์เนล และอยู่ในพื้นที่ของกลุ่มเธรดผู้ใช้ มันสามารถสร้างเธรดและจัดการเธรดได้อย่างรวดเร็วด้วยตัวเอง รวมทั้งจัดการเวลาของทรดด้วยตัวมันเอง โดยไม่ต้องพึ่งเคอร์เนล และเธรดเคอร์เนลจะไม่ยุ่งเกี่ยวกับเธรดผู้ใช้ด้วย

ข้อดีของเธรด

ระบบปฏิบัติการรุ่นใหม่ ๆ มีการจัดการกระบวนการให้มีการทำงานแบบหลายเธรดเพื่อความสะดวกและรวดเร็วในการทำงาน การทำงานแบบหลายเธรดมีข้อดีคือ

- การตอบสนอง (Responsiveness) การทำงานของกระบวนการที่มีหลายเธรดบางเธรดอาจจะถูกบล็อค บางเธรดกำลังทำงาน กระบวนยังดำเนินการต่อไปเพื่อตอบสนองผู้ใช้ได้หลายงานในเวลาเดียวกัน

- การใช้ทรัพยากรร่วมกัน (Resource Sharing) การทำงานเธรดจะมีการใช้ทรัพยากรหลาย ๆ อย่าง ของกระบวนการร่วมกันทำให้หลาย ๆ เธรดสามารถทำงานพร้อมกันได้

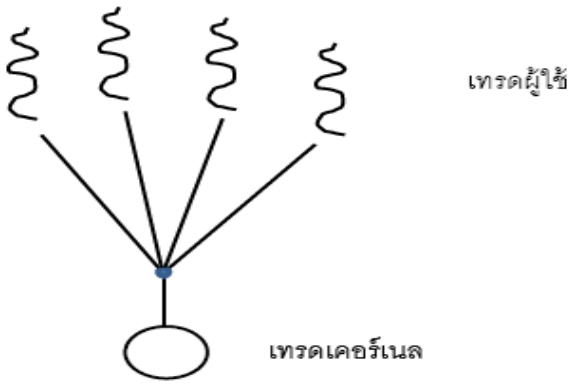
- ประหยัด (Economy) การสร้างเธรดประหยัดเวลาและทรัพยากรมากกว่าการสร้างกระบวนการใหม่ และการทำงานของเธรดมีการใช้ทรัพยากรในกระบวนการร่วมกันจึงทำให้ประหยัดทรัพยากรและระบบสามารถทำงานได้เร็ว

- การใช้ประโยชน์ของสถาปัตยกรรมแบบมัลติโปรเซสเซอร์ (Utilization of Multiprocessor Architectures) กระบวนการที่มีหลายเธรดระบบปฏิบัติการสามารถจัดการให้แต่ละเธรดประมวลผลที่ต่างซีพียูกัน และสามารถประมวลผลควบคู่กันไปได้เป็นการเพิ่มประสิทธิภาพในการทำงาน

โดยทั่วไปเธรดจะต้องรันอยู่ภายใต้กระบวนการหากเธรดที่เป็นส่วนประกอบย่อยของกระบวนการมีหลาย ๆ เธรด เรียกว่า Lightweight Process (LWP) โดยปกติกระบวนการ ที่มี 1 เธรดจะเรียกว่า single thread แต่ถ้า 1 กระบวนการ มีหลายเธรด จะเรียกว่า Multithread เพราะในกระบวนการหนึ่งอาจมีหลาย ๆ เธรด

รูปแบบของมัลติเธรด (Model of Multithread) มีระบบปฏิบัติการหลายระบบที่สนับสนุนการทำงานของเธรดทั้งเธรดเคอร์เนลและเธรดผู้ใช้ ซึ่งมีหลายรูปแบบ ดังนี้

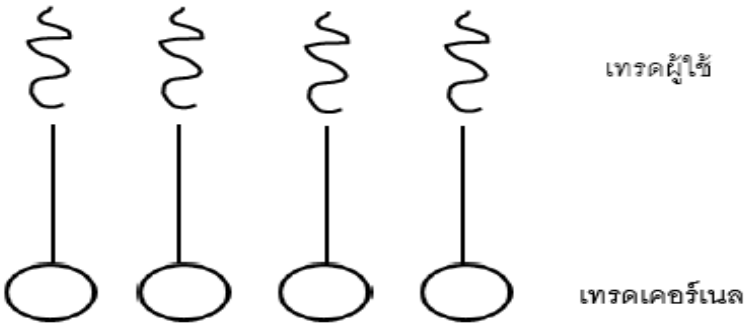
- รูปแบบหลายเธรดต่อ 1 เคอร์เนล (Many-to-one) เป็นการทำงานที่เธรดผู้ใช้หลายหน่วยต่อเธรดเคอร์เนลหน่วยเดียว (ดังรูป 3.4) การดำเนินงานเธรดจะจัดการในส่วนของเธรดผู้ใช้ ขณะใดขณะหนึ่งจะมีเพียงเธรดเดียวเท่านั้นที่ดำเนินการที่เคอร์เนล เธรดผู้ใช้หลาย ๆ เธรดไม่สามารถทำงานพร้อม ๆ กันได้ ถ้ามีเธรดที่บล็อกก์โดยคำสั่งเรียกระบบ กระบวนการจะถูกบลอกก์ด้วย ระบบที่ใช้รูปแบบนี้จะเป็นระปฏิบัติการที่ไม่สนับสนุนเธรดเคอร์เนล



รูปที่ 3.4 รูปแบบหลายเธรดต่อ 1 เธรดเคอร์เนล

(อ้างอิงรูปจาก p 148 , operating system concept with java, Silberschatz, Gavil, Gange, John,Wiley and sons,2004)

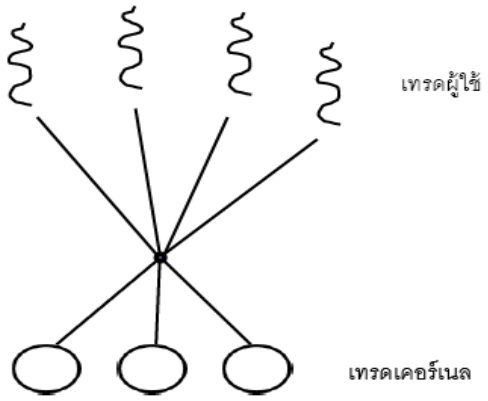
- รูปแบบหนึ่งเธรดต่อหนึ่งเคอร์เนล (One-to-one) เป็นการทำงานที่เธรดผู้ใช้หนึ่งหน่วย จับคู่ทำงานกับเธรดเคอร์เนลหนึ่งหน่วย (ดังรูป 3.5) รูปแบบนี้มีข้อดีคือ ถ้ามีเธรดใดบลอกก์โดยคำสั่งเรียกระบบเธรดอื่น ๆ ยังสามารถทำงานต่อไปได้ และสำหรับระบบที่เป็นมัลติโปรเซสเซอร์ยอมให้หลาย ๆ เธรดทำงานขนานไปพร้อม ๆ กันได้ แต่ระบบนี้ในการสร้างเธรดต้องให้เธรดผู้ใช้สัมพัธ์กับเธรดเคอร์เนลด้วยเนื่องจากการสร้างเธรดเคอร์เนลมีผลต่อประสิทธิภาพของการทำงาน สำหรับข้อจำกัดของรูปแบบนี้คือจำนวนเธรด ในระบบ ระบบปฏิบัติการที่ใช้ รูปแบบนี้คือพวกตระกูล Linux และ Windows



รูปที่ 3.5 รูปแบบหนึ่งเธรดต่อหนึ่งเคอร์เนลเธรด

(อ้างอิงรูปจาก p 148 , operating system concept with java, Silberschatz, Gavil, Gange, John,Wiley and sons,2004)

- รูปแบบหลายเธรดต่อหลายเคอร์เนล (Many-to-Many) เป็นการทำงานที่มีเธรดผู้ใช้หลายหน่วย จับคู่ทำงานกับเธรดเคอร์เนลหลายหน่วยได้เช่นกัน ซึ่งจำนวนเธรดผู้ใช้อาจจะมีมากกว่าหรือเท่ากับจำนวนเธรดเคอร์เนล (ดังรูป 3.6) จำนวนเธรดเคอร์เนลอาจจะเป็นตัวกำหนดเฉพาะเครื่องหรือเฉพาะโปรแกรมสำหรับรูปแบบนี้จะลดข้อจำกัดของสองรูปแบบข้างต้นเป็นวิธีที่ยืดหยุ่นกว่าผู้ใช้ระบบสามารถสร้างเธรดผู้ใช้ที่สัมพันธ์กับเธรดเคอร์เนล ตามความจำเป็น และสามารถดำเนินการแบบขนานกันไปในระบบที่เป็นมัลติโปรเซสเซอร์ได้ และถ้าเธรดมีการบลอกก์โดยคำสั่งเรียกระบบเคอร์เนลจะจัดการให้นำเธรดอื่นขึ้นมาดำเนินการก่อนได้ ระบบปฏิบัติการที่ใช้รูบแบบนี้คือ ตระกูลยูนิกซ์



รูปที่ 3.6 รูปแบบหลายเธรดผู้ใช้ต่อหลายเธรดเคอร์เนล

(อ้างอิงรูปจาก p 148 , operating system concept with java, Silberschatz, Gavil, Gange, John, Wiley and sons,2004)

องค์ประกอบองเธรด เมื่อกระบวนการสร้างเธรดขึ้นมาประกอบด้วยส่วนต่าง ๆ ดังนี้

- Thrads ID หมายเลขเธรด ในกระบวนการ
- Program counter ใช้นับคำสั่งที่ประมวลผลอย่างเป็นลำดับ
- Register set ใช้เก็บค่าที่ทำงานของเธรด
- Stack ใช้เก็บประวัติการประมวลผล

วงจรชีวิตของเธรด

ในขณะที่เธรดดำเนินงานอยู่ในกระบวนการใด ๆ ซึ่งอาจจะมีเพียง 1 เธรดหรือหลายเทรด แต่ละเธรดมีวงจรชีวิตในการทำงานแบ่งออกได้เป็น 7 สถานะ ดังนี้

1) Born คือสถานะเริ่มต้นของทุก ๆ เธรด

2) Ready เมื่อโปรแกรมสั่งให้เธรดเริ่มทำงาน (Start Thrad) เธรดจะเปลี่ยนสถานะจาก Born ไปเป็นสถานะ Ready คืออยู่ในสถานะพร้อมที่จะการประมวลผล และเข้าทำการประมวลผลที่ซีพียู
3) Running คือสถานะที่เธรดกำลังทำการการประมวลผลที่ซีพียู
4) Dead คือสถานะที่เธรดได้ทำการประมวลผลเสร็จแล้ว
5) Blocked คือสถานะที่เธรดรอคอยการทำงานกับอุปกรณ์ Input/Output
6) Waiting คือสถานะที่เธรดรอคอยเหตุการณ์อะไรบางอย่าง เช่น รอคอยสัญญาณจากเธรดอื่น
7) Sleeping คือสถานะที่ระบบปฏิบัติการสั่งให้เธรดหยุดพักการทำงานชั่วคราว โดยจะมีการกำหนดช่วงเวลาในการรอช่วงเวลาหนึ่ง เรียกว่า Sleep Interval และเมื่อหมดเวลาของ Sleep Interval แล้วระบบปฏิบัติการจะให้เธรดเปลี่ยนสถานะจาก Sleeping เป็น Ready

ปฏิบัติการบทที่ 3

1. ให้นักศึกษาแต่ละกลุ่มร่วมกันสร้าง Mind Map จากความรู้เรื่องของกระบวนการและเทรด
2. แต่ละกลุ่มนำ Mind Map มานำเสนอให้เพื่อนแสดงความคิดเห็นเพื่อแลกเปลี่ยนเรียนรู้กัน

คำถามท้ายบท

1. กระบวนการคืออะไร มีสถานะการทำงานอย่างไรบ้าง
2. จงแสดงภาพวงจรชีวิตการทำงานของกระบวนการ (Process Life Cycle) พร้อมทั้งอธิบาย
3. ส่วนประกอบของบล็อกควบคุมกระบวนการ มีอะไรบ้างแต่ละส่วนมีหน้าที่ทำอะไรบ้าง
4. จงอธิบายวิธีการการสร้างกระบวนการและยกเลิกกระบวนการ และมีการใช้ทรัพยากรร่วมกันอย่างไรบ้าง
5. กระบวนการที่ทำงานพร้อมกันหลาย ๆ กระบวนการ มีความสัมพันธ์กันแบบ กระบวนการอิสระและกระบวนการร่วม แต่ละแบบมีลักษณะเป็นอย่างไรอธิบายให้เข้าใจ
6. จงอธิบายขั้นตอนของการสับเปลี่ยนงาน (Context Switch)
7. เธรดคืออะไร ประกอบด้วยอะไรบ้าง มีข้อดีอย่างไร
8. จงอธิบายรูปแบบของเธรดในรูปแบบต่าง ๆ