**เอกสารประกอบการสอน**

**วิชา ระบบปฏิบัติการ 2 (Operating Systems 2) รหัส 4121402**
**บทที่ 8 กรณีศึกษา (Case Study): ระบบปฏิบัติยูนิกซ์**
**(UNIX Operating System)**
**หลักสูตรระดับปริญญาตรี**
**พุทธศักราช 2551 (ปรับปรุง 2554)**

**โดย**

**จุฑาวุฒิ จันทรมาลี**

**สาขาวิทยาการคอมพิวเตอร์**
**คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏสวนดุสิต**

# บทที่ 8 กรณีศึกษา (Case Study): ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System)

ยูนิกซ์ (UNIX) เป็นระบบปฏิบัติการที่ได้รับความนิยมของนักวิจัยและผู้พัฒนาระบบปฏิบัติการทั่วโลก ระบบหนึ่งเพราะมีจุดเด่นด้วยกันหลายประการ เช่น มีความยืดหยุ่น (Flexibility) สามารถที่จะแก้ไขหรือปรับปรุงให้เข้ากับระบบอื่นได้ตามต้องการ รองรับการทำงานทั้งในระบบงานภายในองค์กรขนาดกลางและองค์กรขนาดใหญ่ เป็นระบบที่มี Open Source ทำให้ผู้พัฒนาหรือผู้ที่สนใจสามารถที่จะเข้าไปค้นคว้าและทำการศึกษาได้ด้วยต้นเอง อีกทั้งยังเป็นระบบปฏิบัติการที่ไม่ยึดติดกับอุปกรณ์ (Device) หรือฮาร์ดแวร์ (Hardware) ที่เกี่ยวกับกับระบบอีกด้วย เป็นต้น

## 8.1 ประวัติความเป็นมา (History)

ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) มีต้นกำเนิดจากห้องปฏิบัติการวิจัย Bell ประเทศสหรัฐอเมริกา โดย Ken Thompson และ Dennis Ritchie ปี พ.ศ. 2512 โดย สถาบัน MIT (Massascusetts Institute of Technology) ณ ห้องปฏิบัติการวิจัย AT&T Bell Labs และบริษัท GE (General Electric) ร่วมกันพัฒนาโครงการ Multics ในปี 1960 เพื่อพัฒนาระบบปฏิบัติการสำหรับใช้งานร่วมกับเครื่องคอมพิวเตอร์เมนเฟรม (Mainframe Computer) รุ่น GE 635 ได้พัฒนาให้ระบบปฏิบัติการมีความสามารถทำงานในลักษณะแบบโต้ตอบ (Interactive) และมีระบบที่ช่วยอำนวยความสะดวกต่อการใช้แฟ้ม (File) และข้อมูล (Data) ร่วมกันได้ แต่เกิดปัญหาหลายประการ จนกระทั่ง Bell Labs ได้ถอนตัวออกจากโครงการ แต่โครงการก็ยังดำเนินการต่อโดย Ken Thompson และ Dennis Ritchie ได้ร่วมกันพัฒนาระบบปฏิบัติการใหม่ขึ้นมาเพื่อให้สามารถทำงานบนเครื่องคอมพิวเตอร์รุ่น PDP-7 และใช้ชื่อว่าระบบปฏิบัติการที่ร่วมกันพัฒนาขึ้นมานี้ว่า ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) เพื่อให้ออกเสียงใกล้เคียงกับระบบ Multics ดังนั้นต้นกำหนดของระบบปฏิบัติการยูนิกซ์ก็คือ Multics นั่นเอง ไม่ว่าจะเป็นชื่อระบบแฟ้มข้อมูล (File Name) ที่ใช้ แนวคิดของตัวแปรคำสั่ง (Shell) ระบบปฏิบัติการยูนิกซ์ได้มีการพัฒนาอย่างต่อเนื่องมาเป็น Version 2 เพื่อทำงานบนเครื่องคอมพิวเตอร์รุ่น PDP-11/20 โดยใช้ภาษา Assembly และได้พัฒนาปรับปรุงด้วยภาษาซี (C Language) และเผยแพร่ไปสู่มหาวิทยาลัยต่างๆ ด้วย Version 6 ในปี ค.ศ. 1976

ในปี ค.ศ. 1978 Version 7 ก็ถูกพัฒนาออกมา ซึ่งเป็นต้นแบบของระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) รุ่นใหม่ๆ ออกมา ภายใต้ความรับผิดชอบของสถาบัน AT&T ซึ่งเป็นองค์กรแม่ของ Bell Labs ตลอดจนควบคุมการออกตัวระบบปฏิบัติการยูนิกซ์รุ่นใหม่ๆ ออกมาทำให้ระบบปฏิบัติการยูนิกซ์กลายเป็นผลิตภัณฑ์ที่มีใช้งานกันอย่างแพร่หลายแทนที่จะเป็นเครื่องมือวิจัยภายในสถาบัน ซึ่งทางสถาบัน AT&T ก็ได้พัฒนาระบบปฏิบัติการยูนิกซ์ออกมาใช้งานภายนอก ภายใต้ชื่อรุ่น System III ในปี 1982 และปี 1983 ก็ออกรุ่น System V และพัฒนาต่อมาเรื่อยๆ จนได้เป็นระบบปฏิบัติที่ได้รับความนิยมในปัจจุบัน

หลังจากนั้นก็มีผู้สนใจพัฒนาระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) เพิ่มขึ้นมาหลายแห่ง เช่น University of California at Berkley ได้พัฒนา BSD UNIX (Berkley Software Distribution) ต่อมาหน่วยงานกระทรวงกลาโหมของสหรัฐฯ (Defense Advanced Research Projects Agency - DARPA) ได้

ให้ทุนกับ Berkley ในการพัฒนาระบบปฏิบัติการยูนิกซ์และเกิด Version 4BSD เพื่อสนับสนุนการทำงานในระบบเครือข่ายของ DARPA ซึ่งมีโปรโตคอลที่ใช้ในการติดต่อสื่อสาร คือ TCP/IP และในปี ค.ศ. 1993 ได้พัฒนารุ่นต่อมาคือ Version 4.4 BSD ได้พัฒนาระบบปฏิบัติการยูนิกซ์ให้สามารถสนับสนุนการทำงานของโปรโตคอล X.25 ด้วย ซึ่งในเวลาต่อมา Berkley ก็หยุดการพัฒนาระบบปฏิบัติการยูนิกซ์ลง

นอกจาก Berkley ที่เป็นผู้พัฒนาระบบปฏิบัติการยูนิกซ์หลักแล้วยังมีผู้พัฒนาระบบปฏิบัติการยูนิกซ์รายอื่นๆ อีกหลายรายด้วยกัน เช่น บริษัทซันไมโครซิสเต็ม (Sun Microsystem Co.,Ltd) ได้พัฒนาระบบปฏิบัติการยูนิกซ์โดยใช้ชื่อว่า SunOS และ Solaris บริษัท DEC ได้พัฒนาระบบปฏิบัติการยูนิกซ์แต่ใช้ชื่อว่า Ultrix และเปลี่ยนชื่อเป็น OSF/1 บริษัทไมโครซอฟต์พัฒนาระบบปฏิบัติการยูนิกซ์โดยใช้ชื่อว่า XENIX บริษัทไอบีเอ็มพัฒนาระบบปฏิบัติการยูนิกซ์โดยใช้ชื่อว่า AIX เป็นต้น แต่ไม่ว่าจะเป็นค่ายใดก็ตาม ต่างก็ยึดแนวทางของ BSD UNIX (Berkley Software Distribution) หรือไม่ก็ใช้ระบบปฏิบัติการยูนิกซ์รุ่น System V มาพัฒนาต่อยอดด้วยกันทั้งนั้น

ปัจจุบัน UNIX เป็นเครื่องหมายการค้าที่จดทะเบียน (Registered Trademark) ของหน่วยงานที่ชื่อ The Open Group ซึ่งจะทำการกำหนด และรับรองมาตรฐานของระบบปฏิบัติการ UNIX มี 2 ลักษณะ คือ

1.ระบบปฏิบัติการที่ได้มาตรฐาน UNIX เป็นระบบปฏิบัติการที่ใช้มาตรฐานของ The Open Group ในการพัฒนาขึ้นมา เช่น Digital UNIX, SCO UNIX, IBM's OpenEdition MVS

2.ระบบปฏิบัติการคล้าย UNIX (UNIX Compatible) เป็นระบบปฏิบัติการที่มีลักษณะคล้ายระบบ UNIX แต่ยังไม่ได้จดทะเบียนรับรองเป็นทางการ เช่น Sun Solaris, IBM AIX, Linux เป็นต้น

**8.2 เป้าหมายของการออกแบบ (The Gold of Design)**

ยูนิกซ์ (UNIX) เป็นระบบปฏิบัติการขนาดใหญ่ ดังนั้นเป้าหมายหลักของการออกแบบระบบปฏิบัติการนี้ขึ้นมาเพื่อให้สามารถใช้งานในลักษณะการทำงานหลายๆ งานพร้อมกัน (Multitasking) และเป็นแบบผู้ใช้งานหลายคน (Multi-User) คือ มีผู้ใช้หลาย ๆ คนพร้อมกัน เป็นระบบที่พัฒนามาใช้กับเครื่องคอมพิวเตอร์ขนาดใหญ่ เช่น เครื่องเมนเฟรม มินิคอมพิวเตอร์และเวิร์กสเตชั่น (Workstation) เครื่องไมโครคอมพิวเตอร์ธรรมดา ๆ ที่ติดตั้งระบบปฏิบัติการยูนิกซ์ (UNIX) สามารถทำงานรองรับเครื่องคอมพิวเตอร์ที่มี User ต่อเชื่อมเข้ามาได้มากถึง 120 ตัว ไปพร้อม ๆ กันและเหมาะสมสำหรับระบบเครือข่าย (Network) นอกจากนั้นยังสามารถเคลื่อนย้ายงานและแอพพลิเคชั่นไปมาระหว่างสถาปัตยกรรม (Platform) ได้ อีกทั้งยังสามารถย้ายงานที่ประมวลผลอยู่บนดอส (DOS) หรือระบบปฏิบัติวินโดว์ (Windows Opearating System) มาใช้ประมวลผลบนระบบปฏิบัติการยูนิกซ์และยังมีอรรถประโยชน์ (Utility) ที่ช่วยเพิ่มประสิทธิภาพให้กับ UNIX อีกด้วย

**8.3 การจัดการหน่วยความจำ (Memory Management)**

เป็นระบบปฏิบัติการยูนิกซ์ (UNIX) จะใช้วิธีการจัดการหน่วยความจำโดยใช้เทคนิควิธีการสลับเข้า/ออก (Swapping) หรือเทคนิควิธีจัดการเพจตามความต้องการ (Demand Paging) โดยพิจารณาจากงาน (Task) หรือกระบวนการ (Process) ที่เข้ามาใช้งานในระบบ ดังนี้

เทคนิควิธีการสลับเข้า/ออก (Swapping) จะเหมาะกับงานหรือกระบวนการที่มีขนาดเล็กโดยจะเก็บงานหรือกระบวนการที่ต้องการประมวลผล (Execute) ลงในหน่วยความจำหลัก (Main Memory) แต่วิธีก็มี

ข้อจำกัดเกิดขึ้นตรงที่ว่าการกำหนดขนาดของงานหรือกระบวนการที่เข้ามาประมวลผล เช่น ถ้าระบบมีขนาดหน่วยความจำ 256 MB ต้องใช้พื้นที่สำหรับจัดเก็บระบบปฏิบัติการขนาด 128 MB เหลือพื้นที่สำหรับกระบวนการอยู่ 128 MB ซึ่ง ณ เวลาหนึ่งถ้ามีงานหรือกระบวนการขนาดเท่ากับหน่วยความจำที่เหลือแสดงว่าจะมีเพียงกระบวนการเดียวเท่านั้นที่ประมวลผลอยู่ หากกระบวนการอื่นที่มีขนาดเล็กหรือไกล้เคียงกับกระบวนการแรกต่อหยุดรอ (Wait) ให้กระบวนการแรกทำงานเสร็จก่อนจึงจะสลับเอากระบวนการอื่นเข้ามาประมวลผลแทนกระบวนการเดิมทำให้ต้องเสียเวลาและใช้ประสิทธิภาพของระบบไม่ดีเท่าที่ควร

เทคนิควิธีจัดการเพจตามความต้องการ (Demand Paging) เป็นวิธีการทำงานควบคู่กับเทคนิคการจัดการหน่วยความจำเสมือน (Virtual Memory) ทำให้งานหรือกระบวนการที่มีขนาดใหญ่สามารถเข้ามาประมวลผลในหน่วยความจำที่เล็กกว่าได้ โดยนำงานหรือกระบวนการที่ต้องการประมวลผลไว้ในหน่วยความจำหลักส่วนงานหรือกระบวนการที่รอการประมวลผลถูกโหลดเข้าไปเก็บไว้ในหน่วยความจำสำรอง แสดงได้ดังรูปที่ 8.1

รูปที่ 8.1 แสดงส่วนประกอบต่างๆ ของหน่วยความจำในระบบปฏิบัติการยูนิกซ์

จากรูปที่ 8.1 แสดงส่วนประกอบต่างๆ ของหน่วยความจำในระบบปฏิบัติการยูนิกซ์ โดยกำหนดส่วนสแต็ก (Stack) และส่วนข้อมูล (Data) เป็นส่วนเฉพาะที่งานและกระบวนการนั้นๆ ทำงานเท่านั้นเมื่อทำงานเสร็จก็จะคืนพื้นที่หน่วยความจำที่ใช้แล้วคืนระบบ ไล่ขึ้นไปจนถึงส่วนบนสุดของหน่วนความจำจะเป็นสแต็ก (Stack) ซึ่งใช้เก็บข้อมูลคำสั่งในการเรียก (Call) หรือคำสั่งในการขัดจังหวะ (Interrupt) ส่วนการทำงานของโค้ดโปรแกรม (Program Code) จะถูกแบ่งปัน (Share) เพื่อให้ทุกกระบวนการ (Process) สามารถที่จะเรียกใช้งานได้แต่จะกำหนดสิทธิเฉพาะอ่านอย่างเดียว (Read) โดยโค้ดโปรแกรมจะถูกกันไว้ให้แต่ละกระบวนการทำงานจนเสร็จสิ้น โดยระบบปฏิบัติการยูนิกซ์จะใช้ตารางตัวอักษร (Text Table) สำหรับเก็บรอบ (Track) ของกระบวนการที่ใช้งานโค้ดโปรแกรม

ขณะที่ระบบปฏิบัติการยูนิกซ์ทำงาน หน่วยความจำจะเก็บส่วนที่เป็นคอร์เนล (Kernel) ไว้ตลอดเวลา เพราะคอร์เนล (Kernel) เป็นกลุ่มของโปรแกรมที่ทำงานพื้นฐานและเก็บเฉพาะเพจ (Page) ที่จำเป็นต้องใช้ไว้หน่วยความจำเท่านั้น โดยคอร์เนล (Kernel) จะมีคำสั่งเรียก (System Call) เพื่อค่อยให้บริการกระบวนการ (Process) ที่ร้องขอการทำงานบางอย่าง เช่น การรับส่งข้อมูลระหว่างเครือข่าย การเรียกใช้อุปกรณ์ในการรับและแสดงผล (I/O Device) ข้อมูล เป็นต้น

## 8.4 การจัดการกระบวนการ (Process Management)

ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) ภายในคอร์เนล (Kernel) จะมีฟังก์ชัน (Function) ในการจัดการกระบวนการ (Process Management) ที่เกี่ยวข้องกับการจัดสรรเวลาในการใช้งานหน่วยประมวลผลกลาง (Central Processing Unit: CPU) การจัดตารางการทำงาน (Process scheduling) ซึ่งจะเกี่ยวข้องกับการคัดเลือกกระบวนการ (Process) ที่เปลี่ยนสถานะจาก สถานะพร้อม (Ready) เข้ามาประมวลผล (Execute) ตามรูปแบบของอัลกอริทึมและเวลาที่จัดสรรไว้ ซึ่งส่วนใหญ่จะใช้อัลกอริทึมแบบแถวลำดับตามความสำคัญ (Priority Queue) ในการเลือกกระบวนการ (Process) เข้ามาประมวลผล (Execute) ส่วนกระบวนการ (Process) ที่ใช้เวลาของ CPU มากจะถูกโหลดเข้าไปเก็บไว้ในหน่วยความจำสำรองแทน เช่น ฮาร์ดดิสก์ (Hard disk) เป็นต้น ระบบปฏิบัติการยูนิกซ์จะมีคำสั่งสำหรับจัดการกระบวนการ ดังนี้

**คำสั่ง Fork** เป็นคำสั่งที่ยอมให้โปรแกรม a สามารถดำเนินการต่างกับโปรแกรม b ได้ เช่น เปิดไฟล์บันทึกข้อมูลของโปรแกรม b โดยแบ่งสำเนา (Copy) ออกเป็น 2 ชุด เมื่อมีการประมวลผล (Execute) จะสร้าง Process ID (PID) ขึ้นมาใหม่และจะมี PID ที่ไม่ซ้ำกัน

**คำสั่ง Wait** เป็นคำสั่งที่บอกให้กระบวนการ (Process) หยุดรอ

**คำสั่ง Exec** ใช้สำหรับสั่งให้โปรแกรมเริ่มทำการประมวลผลใหม่

## 8.5 การจัดการอุปกรณ์ในการรับและแสดงผล (I/O Device Management)

ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) สามารถควบคุมการทำงานของอุปกรณ์ในการรับและแสดงผล (I/O Device Management) ที่ต่างชนิดกันได้โดยใช้ไฟล์ชนิดพิเศษที่ชื่อว่า “i-node” ที่อยู่ภายในไดเร็กทอรีซึ่งเก็บข้อมูลและรายละเอียดของอุปกรณ์ (Device Directory) ในขณะที่มีการรับส่งข้อมูลระหว่างหน่วยความจำหลัก (Main Memory) กับอุปกรณ์ต่อพ่วง (Peripheral Device) และมีโปรแกรมที่ใช้ควบคุมการทำงานของอุปกรณ์ที่ติดตั้งอยู่ในระบบปฏิบัติการที่เรียกว่า “อุปกรณ์ขับ (Device Driver)” ซึ่งโดยทั่วไปจะมีไฟล์ที่เกี่ยวข้องเพื่อใช้เก็บข้อมูลของอุปกรณ์อยู่ในไดเร็กทอรี (Directory) ที่ชื่อว่า /dev นอกจากนี้ระบบปฏิบัติการยูนิกซ์ยังมีไฟล์ conf.c ที่ใช้เก็บค่าพารามิเตอร์ (Parameter) สำหรับใช้ควบคุมทรัพยากรต่างๆ เช่น ขนาดของที่พักข้อมูล (Buffer) ของคอร์เนล (Kernel) หรือขนาดพื้นที่สำหรับการสลับ (Swap) กระบวนการ (Process) เข้าหรือออกเมื่อมีการเรียกใช้งานอุปกรณ์ในการรับและแสดงผล (I/O Device) เป็นต้น โดยระบบปฏิบัติการยูนิกซ์แบ่งระบบ I/O สำหรับจัดการอุปกรณ์ต่อพ่วงต่างๆ เป็น 2 ประเภท ได้แก่

1. ระบบ I/O ที่เกี่ยวข้องกับการจัดการอุปกรณ์ที่เก็บข้อมูลเป็นบล็อก (Block) เช่น ดิกส์ (Disk) เทปแม่เหล็ก (Magnetic Tap) ซึ่งอาจจะเรียกอุปกรณกลุ่มนี้ว่า อุปกรณ์ที่ส่งข้อมูลเป็นบล็อก (Block Device) เป็นต้น
2. ระบบ I/O ที่เกี่ยวข้องกับการจัดการอุปกรณ์ที่เก็บข้อมูลเป็นตัวอักษร (Character) หรืออุปกรณ์ที่เก็บข้อมูลแบบไม่เป็นโครงสร้าง (Unstructured I/O) เช่น เครื่องพิมพ์ (Printer) ซึ่งอาจจะเรียกอุปกรณ์กลุ่มนี้ว่าอุปกรณ์ที่ต้องติดต่อสื่อสารเพื่อส่งข้อมูล (Communication Device) หรืออุปกรณ์ที่รับส่งข้อมูลเป็นตัวอักขระ (Character Device) เป็นต้น

**8.6 การจัดการไฟล์ (File Management)**

การจัดการไฟล์ในระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) จะใช้วิธีการจัดการไฟล์ในรูปแบบไดเร็กทอรี่เดียว (Single Directory) แต่มีหลายระดับชั้น (Multi-Level) เริ่มตั้งแต่รูท (root) ไล่ลงมาเรื่อย ๆ จนถึงผู้ใช้ไฟล์ต่าง ๆ ไม่ว่าจะเก็บลงในดิสก์ (Disk) ตัวไหน เทป (Tape) ม้วนไหน ก็จะรวมอยู่ในไดเร็กทอรี่เดียวกันทั้งหมด

**8.6.1 ประเภทของไฟล์ (File Type)** โดยยูนิกซ์ (UNIX) แบ่งไฟล์ออกได้เป็น 3 ชนิด ได้แก่ ไฟล์ธรรมดา (Ordinary File) ไฟล์พิเศษ (Special File) และ ไดเร็กทอรี (Directory) ดังนี้

**1. ไฟล์ธรรมดา (Ordinary File)** คือกลุ่มของข้อมูลหลายๆ ไบต์วางเรียงต่อกันไม่มีโครงสร้างของข้อมูลใด ๆ เช่น ไฟล์ข้อมูล ไฟล์โปรแกรม เป็นต้น การที่ยูนิกซ์มองไฟล์เป็นกลุ่มของไบต์ที่ไม่มีโครงสร้างข้อมูลนี้ สร้างความยืดหยุ่นในการทำงานของโปรแกรมต่าง ๆ โปรแกรมที่ต้องการเก็บข้อมูลไว้ในไฟล์ต้องจัดการเรื่องโครงสร้างข้อมูลเอาเองทั้งหมด โปรแกรมจึงมีอิสระในการจัดโครงสร้างข้อมูลไฟล์ได้อย่างเต็มที่

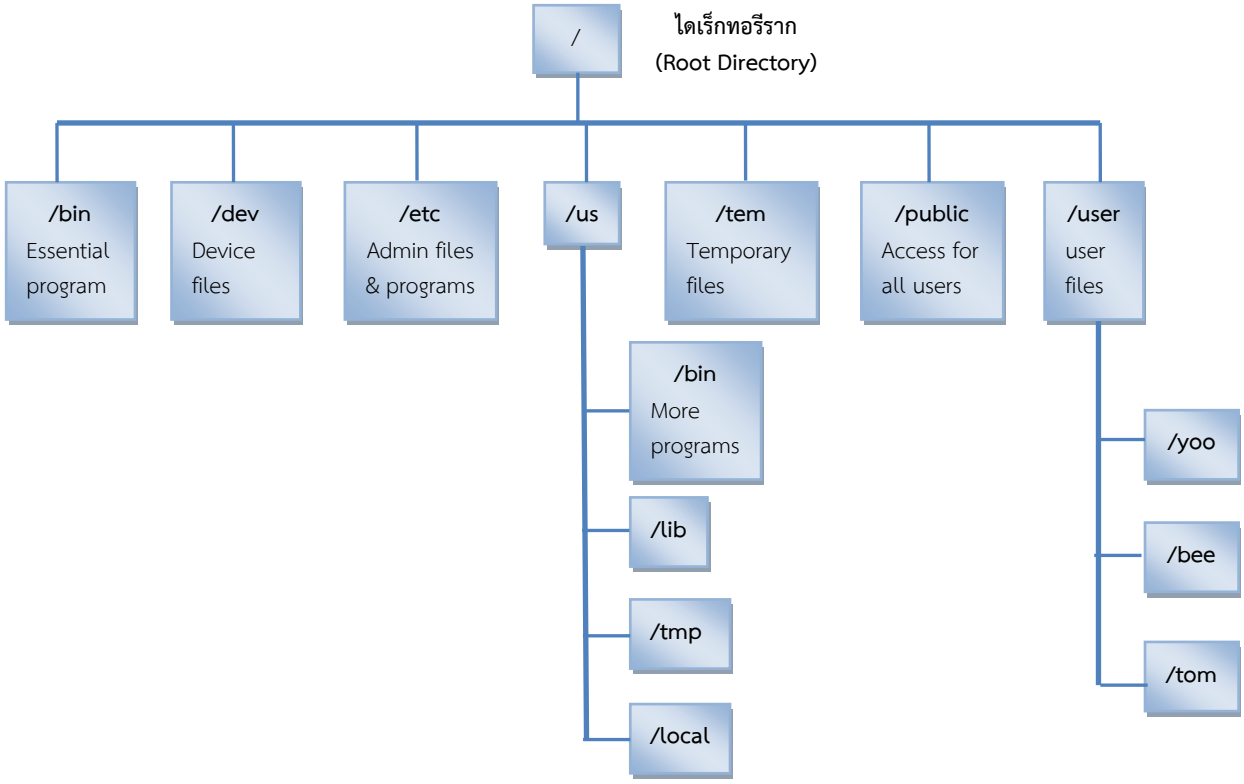
**2. ไฟล์พิเศษ (Special File)** คือ ไฟล์ที่สามารถอ้างอิงไปถึงอุปกรณ์ต่าง ๆ ในระบบ ยูนิกซ์มองอุปกรณ์ต่าง ๆ ในระบบเป็นไฟล์ เช่น เครื่องพิมพ์ 1 เครื่องเป็น 1 ไฟล์ จอภาพ 1 จอภาพเป็น 1 ไฟล์ ไฟล์เหล่านี้จะติดต่อกับตัวขับอุปกรณ์ของอุปกรณ์นั้น ๆ ดังนั้นการที่จะส่งข้อมูลหรือรับข้อมูลจากอุปกรณ์ต่าง ๆ ต้องส่งหรือรับข้อมูลผ่านทางไฟล์ของอุปกรณ์นั้นๆ ซึ่งวิธีนี้เกิดผลดีในแง่ของการถ่ายเทข้อมูลของโปรเซสต่าง ๆ

**3. ไฟล์ไดเร็กทอรี (Directory)** คือ ไฟล์ที่ยูนิกซ์กำหนดโครงสร้างข้อมูลไว้ให้ โดยมีโครงสร้างเป็นไดเร็กทอรีของระบบคือ ข้อมูลหนึ่งตัวจะประกอบไปด้วยชื่อไฟล์ และหมายเลขไอโหนดของไฟล์นั้น นอกจากนี้ไดเร็กทอรี่ยังสามารถเชื่อมโยงถึงกันได้ด้วย ไดเร็กทอรีและไฟล์ที่เก็บรวมอยู่ในดิสก์หรือเทป หรืออุปกรณ์อื่น ๆ ที่ไม่ได้เป็นไดเร็กทอรีเดียวกันกับไดเร็กทอรีของยูนิกซ์ สามารถนำมาต่อเชื่อมโยงเข้ากับไดเร็กทอรีของระบบเป็น ไดเร็กทอรี่ย่อยหนึ่งได้ โดยยูนิกซ์มีคำสั่งพิเศษคือคำสั่ง mount เพื่อใช้เชื่อมไดเร็กทอรีอื่นเข้ากับไดเร็กทอรีของระบบ และในทำนองเดียวกันยูนิกซ์ก็มีคำสั่ง unmount เพื่อตัดไดเร็กทอรีย่อยออกจากระบบทั้งคำสั่ง mount และ unmount นี้มีประโยชน์สำหรับการใช้ดิสก์ หรือเทปที่ถอดเข้าออกจากตัวขับได้

**8.6.2 การตั้งชื่อไฟล์ในระบบปฏิบัติการยูนิกซ์** ชื่อไฟล์ในระบบปฏิบัติการยูนิกซ์จะเป็นแบบ Case Sensitive คือ การใช้อักษรตัวใหญ่และอักษรตัวเล็กจะมีความหมายที่ไม่เหมือนกัน เช่น ROOT, root, Root และ Root จะถือว่าเป็นชื่อไฟล์ที่ไม่เหมือนกัน ซึ่งชื่อไฟล์ที่ใช้ในการตั้งชื่อสามารถกำหนดความยาวได้ถึง 255 ตัวอักษร โดยมีการใช้โครงสร้างต้นไม้ในการแสดงการอ้างถึงไดเร็กทอรี (Directory) และไฟล์ต่างๆ จะเครื่องหมาย Slash (/) จากจุดบนสุด (Root) ของโครงสร้างต้นไม้ ผ่านลำดับชั้น (Level) ต่างๆ ลงมา แต่หากต้องการอ้างอิงขึ้นไปหนึ่งระดับจากไดเร็กทรอรีปัจจุบัน (Current Directory) จะใช้เครื่องหมาย 2 จุดติดกัน (..) แสดงดังรูปที่ 8.2

นอกจากนี้ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) ยังได้จัดเตรียมคำสั่งสำหรับเรียกดูรายละเอียดในไดร์กทอรี โดยเรียกดูผ่านคำสั่ง ls (ย่อมาจากคำสั่ง List) ซึ่งมีรายละเอียดที่เกี่ยวข้องดังนี้

- l เป็นการกำหนดให้แสดงผลลัพธ์แบบยาว (Long Format)
- a แสดงไฟล์ข้อมูลทั้งหมด
- F แสดงเครื่องหมาย / หลังชื่อไดเร็กทอรีและแสดงเครื่องหมาย * หลังไฟล์ที่กำลังประมวลผล (Execute) ได้




รูปที่ 8.2 แสดงตัวอย่างโครงสร้างลำดับชั้นของการจัดการไฟล์ในระบบปฏิบัติการยูนิกซ์

ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) สามารถกำหนดสิทธิ์ในการเข้าถึงไฟล์ในแต่ละชุดได้ดังตัวอย่างต่อไปนี้

r-- หมายถึง อ่านอย่างเดียว
rw- หมายถึง อ่านและเขียนได้
r-x หมายถึง อ่านและประมวลผล (Execute) ได้
rwx หมายถึง อ่าน เขียนและประมวลผล (Execute) ได้

**8.6.3 คำสั่งต่างๆ ในระบบในระบบปฏิบัติการยูนิกซ์** โดยจัดการผ่านระบบโหมดคำสั่ง (Command Mode) ซึ่งคำสั่งต่างๆ จะอยู่ในรูปตัวอักษรย่อที่ผู้ใช้งานสามารถจดจำคำสั่งได้ง่าย โดยรูปแบบคำสั่ง มีดังนี้

```
command  argument  filenname
```

โดยที่ command หมายถึง คำสั่งของยูนิกซ์ (UNIX) ที่ต้องการใช้
argument หมายถึง ทางเลือกของคำสั่ง
filename หมายถึง ไดเร็กทอรีหรือเส้นทาง

คำสั่งที่สำคัญและมักใช้งานอยู่เป็นประจำในระบบปฏิบัติการยูนิกซ์ แสดงได้ดังตารางที่ 8.2

ตารางที่ 8.2 แสดงคำสั่งต่างๆ ที่เกี่ยวข้องกับการจัดไฟล์ในระบบปฏิบัติการยูนิกซ์

| คำสั่ง | ความหมาย | คำอธิบาย |
|---|---|---|
| ls | List Directory | แสดงรายการชื่อไฟล์ที่อยู่ในไดเร็กทอรี |
| ls -l | Long List | แสดงรายการข้อมูลทั้งหมดในไดเร็กทอรี |
| cd | Change Directory | เปลี่ยนไดเร็กทอรี |
| cp | copy | ทำการคัดลอกไฟล์หรือไดเร็กทอรี |
| rm | Remove | ลบไฟล์หรือไดเร็กทอรี |
| mv | Move | ย้ายไฟล์หรือไดเร็กทอรี |
| lpr | Print | พิมพ์ไฟล์ |
| mkdir | Make Directory | สร้างไดเร็กทอรีใหม่ |
| format | Format | ทำการฟอร์แมท |
| pwd | Print Working Directory | แสดงชื่อของไดเร็กทอรีที่กำลังทำงานอยู่ |

### 8.6.4 Shell และ Script File

**Script File (Command File/Shell File)** ใช้เพื่อทำงานเดิมซ้ำๆ แต่ละคำสั่งภายในไฟล์ซึ่งมีรูปแบบการทำงานดังนี้

**1. การประมวลผลแบบต่อเนื่อง (Synchronous Execution)** จะทำงานตามลำดับ โดยทำที่ละคำสั่งต่อเนื่องกันไปจนเสร็จสิ้นจึงจะสามารถรับคำสั่งจากผู้ใช้เข้ามาประมวลผลใหม่ได้ ซึ่งบางครั้งอาจจะเรียกการทำงานในลักษณะนี้ว่า “โหมดส่วนหน้า (Foreground Mode)”

**2. การประมวลผลแบบไม่ต่อเนื่อง (Asynchronous Execution)** จะทำงานแบบไม่เป็นลำดับ โดยสามารถรับคำสั่งจากผู้ใช้เข้ามาประมวลผลใหม่ได้โดยไม่ต้องคำนึงถึงว่าคำสั่งที่ประมวลผลก่อนหน้าจะแล้วเสร็จ ซึ่งบางครั้งอาจจะเรียกการทำงานในลักษณะนี้ว่า “โหมดส่วนหลัง (Background Mode)”

**Shell** จะทำหน้าที่รับคำสั่งจากผู้ใช้ เพื่อส่งให้กับระบบปฏิบัติการทำงานตามคำสั่งที่ผู้ใช้กำหนดไว้ เช่น

**1. Bourne Shell (sh)** เป็น Shell มาตรฐานที่มีในระบบปฏิบัติการยูนิกซ์ทุกเวอร์ชัน (Version)

**2. C Shell (csh)** พัฒนามาจากภาษาซี (C Language) มีประสิทธิภาพการทำงานสูงกว่า Bourne Shell

**3. Korn Shell (ksh)** เป็น Shell ที่นำ Bourne Shell และ C Shell มาปรับปรุงให้มีประสิทธิภาพการทำงานสูงกว่า Shell ทั้งสองและเพิ่มคุณสมบัติให้สามารถเขียน Shell Script ได้ง่ายแต่ตัวโปรแกรมที่ใช้เขียนมาขนาดใหญ่

4. **Bourne Again Shell (bash)** เป็น Shell ที่นำ Bourne Shell มาพัฒนาต่อจนกลายเป็น Shell มาตรฐานของระบบปฏิบัติการลีนุกซ์ (Linux Operating System) สามารถประมวลผล (Run) ได้ทั้ง Script File ของ Bourne Shell และ C Shell

**สรุป**

ยูนิกซ์ (UNIX) เป็นระบบปฏิบัติการที่มีความยืดหยุ่น (Flexibility) สามารถที่จะแก้ไขหรือปรับปรุงให้เข้ากับระบบอื่นได้ตามต้องการ รองรับการทำงานทั้งในระบบงานภายในองค์กรขนาดกลางและองค์กรขนาดใหญ่ เป็นระบบที่มี Open Source ทำให้ผู้พัฒนาหรือผู้ที่สนใจสามารถที่จะเข้าไปค้นคว้าและทำการศึกษาได้ด้วยต้นเอง อีกทั้งยังเป็นระบบปฏิบัติการที่ไม่ยึดติดกับอุปกรณ์ (Device) หรือฮาร์ดแวร์ (Hardware) ที่เกี่ยวกับกับระบบอีกด้วย

การจัดการหน่วยความจำ (Memory Management) ระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) (UNIX) จะใช้วิธีการจัดการหน่วยความจำโดยใช้เทคนิควิธีการสลับเข้า/ออก (Swapping) หรือเทคนิควิธีจัดการเพจตามความต้องการ (Demand Paging) โดยพิจารณาจากงาน (Task) หรือกระบวนการ (Process) ที่เข้ามาใช้งานในระบบโดยใช้เทคนิควิธีการสลับเข้า/ออก (Swapping) จะเหมาะกับงานหรือกระบวนการที่มีขนาดเล็กโดยจะเก็บงานหรือกระบวนการที่ต้องการประมวลผล (Execute) ลงในหน่วยความจำหลัก (Main Memory) แต่วิธีก็มีข้อจำกัดเกิดขึ้นตรงที่ว่าการกำหนดขนาดของงานหรือกระบวนการที่เข้ามาประมวลผล

การจัดการไฟล์ในระบบปฏิบัติการยูนิกซ์ (UNIX Operating System) จะใช้วิธีการจัดการไฟล์ในรูปแบบไดเร็กทอรี่เดียว (Single Directory) แต่มีหลายระดับชั้น (Multi-Level) เริ่มตั้งแต่รูท (root) ไล่ลงมาเรื่อย ๆ จนถึงผู้ใช้ไฟล์ต่าง ๆ ไม่ว่าจะเก็บลงในดิสก์ (Disk) ตัวไหน เทป (Tape) ม้วนไหน ก็จะรวมอยู่ในไดเร็กทอรี่เดียวกันทั้งหมด โดยยูนิกซ์ (UNIX) แบ่งไฟล์ออกได้เป็น 3 ชนิด ได้แก่ ไฟล์ธรรมดา (Ordinary File) ไฟล์พิเศษ (Special File) และ ไดเร็กทอรี (Directory)