2 Tier คือ ยุค 1980 คอมได้รับความนิยม ระบบ EIS มักเป็นแบบ 2-tier systems จะประกอบด้วย 2 ส่วนคือ Presentation logic ส่วนกำหนดรูปแบบการติดต่อระหว่างผู้กับแอพพลิเคชั่น และ Business logic จะกำหนดว่าข้อมูลจะถูกจัดการอย่างไรในธุรกิจนั้นๆ โปรแกรม 2 ส่วนนี้จะถูกติดตั้งและทำงานใน Client ที่เป็น PC ต่อกับ DB ทำหน้าที่เก็บและควบคุมข้อมูลขององค์กร

ข้อดี คือ

1. ไม่ต้องยุ่งเกี่ยวกับการจัดเก็บข้อมูลโดยตรง

2. ประหยัดหน่วยความจำ เพราะข้อมูลไม่ได้เก็บไว้ในเครื่อง Client

3. เครื่อง Client สามารถใช้ข้อมูลร่วมกันได้ ทำให้ไม่มีปัญหาข้อมูลไม่เหมือนกัน (ทั้ง 3 ข้อที่กล่าวมาคือการลดภาระการทำงานของคอมพิวเตอร์)

4. เหมาะกับระบบงานขนาดกลางและไม่ซับซ้อน

ข้อเสีย คือ

1.ไม่ปลอดภัย เพราะ Client สามารถเข้าถึงฐานข้อมูลได้โดยตรง

2. กรณีเป็นระบบใหญ่ๆจะทำให้ Client ทำงานหนัก

3. หาก app. มีการเปลี่ยนแปลงจะต้องเสียเวลาในการติดตั้ง app. เพิ่มเติมเนื่องจากต้องติดตั้งให้กับ Client ทุกเครื่อง

3 Tier คอนเซปต์พื้นฐานคือการแบ่งแยกหน้าที่ความรับผิดชอบของแต่ละ Tier ให้เด็ดขาดจากกัน ไม่ว่าจะเป็น Presentation logic, Business logic, Database

ข้อดี คือ

1.หากมีการเปลี่ยนแปลงในบาง layer จะส่งผลกระทบต่อ layer อื่นน้อยมาก

2. สามารถนำกลับมาใช้ใหม่ได้เช่น Business logic

3. คอมพิวเตอร์แต่ละตัวรับภาระน้อยลง รองรับการทำงานปริมาณมากๆ

ข้อเสีย คือ

1. ออกแบบและพัฒนาระบบค่อยข้างยาก

สถาปัตยกรรมแบบ Thin และ Fat Clients

Thin-client model: เป็นระบบที่ให้ client มีการประมวลผลน้อยที่สุด โดยการใช้เพียงแค่ Browser ซึ่งทำหน้าที่แสดงผล (Presentation) เท่านั้น การประมวลผลส่วนใหญ่จะอยู่ที่เครื่องแม่ข่าย (Server) รวมทั้ง Code ของซอฟต์แวร์ด้วย ทั้ง Application processing และ การจัดการข้อมูล (Data management)

ข้อดี คือ

1. ระบบนี้จะมีความสะดวกในการบริหารจัดการ เพราะสามารถทำได้จากส่วนกลาง ตัวอย่างของระบบนี้เช่น การใช้ http://www.ในระบบอินเตอร์เน็ต เป็นต้น

ข้อเสีย คือ

1. ระบบนี้คือเครื่องแม่ข่ายจะทำงานหนักมาก

Fat-client model: เป็นระบบที่ให้เครื่อง Client ทำงานเป็นส่วนใหญ่ ทั้งการแสดงผล (Presentation) และการประมวลผลการทำงาน (Application processing)

Fat-client: จะทำหน้าที่ติดตั้งซอฟแวร์หรือ Code ซึ่งจะเป็นการกระจายการประมวลผล ไปทั้งที่เครื่อง Client และเครื่อง Server แต่จะหนักไปที่เครื่อง Client

ข้อดีของแบบนี้คือระบบนี้คือเครื่องแม่ข่ายไม่จำเป็นต้องมีขนาดใหญ่มาก แต่เครื่อง client จะต้องมีประสิทธิภาพที่ดี เพราะจะต้องทำการประมวลผลที่เครื่อง client

ข้อเสียของแบบนี้คือการบริหารจัดการค่อนข้างยาก เพราะจะต่องติดตั้งซอฟต์แวร์ทั้งที่เครื่องแม่ข่าย และเครื่องลูกข่ายทั้งหมด

Fat-Server: คือทำหน้าที่เป็น data management และจะหนักไปในเครื่อง Server

ข้อดีของแบบนี้คือง่ายการต่อการอัพเดท ถ้าประมวลผลบน Server จะทำงานเสถียรกว่า ยิ่งถ้า Client มีความความแตกต่างกันมากจะทำงานง่ายยิ่งขึ้น

Distributed programming

เป็นการเขียนโปรแกรมเพื่อรองรับการคำนวณแบบกระจายตัวเช่น บิลเทอเรล ระบบการซื้อขายผ่านไปรณีย์ ระบบธนาคารแบบกระจายตัว หรือการบริการขนส่งข้ามคืน

ข้อดีคือใช้เวลาตอยสนองได้เร็วขึ้น, ใช้ต้นทุนน้อยกว่า, ปรังปรุงความถูกต้องของข้อมูล, การใช้ทรัพยากรร่วมกัน, ลดต้นทุนตัวประมวลผลหลัก, เพิ่มความน่าเชื่อถือ

ข้อเสียคือการขาดแคลนผู้เชียวชาญด้าน MIS, มาตรฐานของระบบ, ความถูกต้องของข้อมูล

Peer to peer มีทั้งหมด 2 แบบ

Reader centric: เข้าถึงข้อมูลโดยใครก็ได้

Publisher centric: กำหนดสิทธิในการเข้าถึงข้อมูล คล้ายๆการตีพิมพ์หนังสือ
คือไม่จำเป็นต้องผู้แลหรือจัดการระบบ หรือเรียกอีกชื่อว่า Woke group เหมาะสำหรับการทำงานกับคอม ไม่เกิน 10 เครื่อง มีทรัพยากรที่แชร์กันไม่มากเช่นไฟล์ เครื่องพิมพ์ ไม่จำเป็นต้องรักษาความปลอดภัยของข้อมูล การขยายตัวของเครือข่ายในอนาคตไม่มาก เหมาะกับองค์กรขนาดเล็ก เป็นได้ทั้ง Client และ Server คนใช้ต้องฝึกอบรม แต่อาจจะเป็นการยากเนื่องจากผู้ใช้แต่ละคนอาจมีงานอื่นที่ต้องทำ

TCP และ UDP เป็น protocol สำคัญที่อยู่ใน transport layer protocol ซึ่งถูกออกแบบให้มีคุณสมบัติหน้าที่การทำงานที่เหมาะกับงานที่แตกต่างกัน

คุณสมบัติของ TCP

1. ไว้วางใจได้ว่าข้อมูลส่งไปถึงผู้รับอย่างแน่นอน หากส่งไปไม่ถึง TCP จะมีการส่งซ้ำ
2. มีการเชื่อมต่อช่องทางการรับส่งข้อมูลก่อนที่จะเริ่มส่ง
3. มีการควบคุมปริมาณการรับส่งระหว่างต้นทาง-ปลายทาง
4. มีการควบคุมไม่ให้ส่งเข้าไปในเครือข่ายที่มีความหนาแน่นของข้อมูลสูงมาก

คุณสมบัติของ UDP ตรงข้ามกับ TCP ถ้าดูแบบนี้อ่ะ TCP จะดีกว่า UDP เยอะแต่ในความเป็นจริง UDP ก็มีข้อดีเหนือ TCP หลายข้อ

ข้อดีของ UDP

– เริ่มต้นส่งข้อมูลได้เร็วกว่าเพราะไม่ต้องรอการสร้าง connection

– ส่งข้อมูลได้เร็วกว่าเพราะไม่ต้องรอการตรวจสอบ

– ส่งข้อมูลได้ปริมาณมากว่าเพราะไม่มี flow control และ congestion control

การใช้งานที่เหมาะสม TCP เช่น SMTP, Telnet, HTTP, FTP

การใช้งานที่เหมาะสม UDP เช่น NFS, RIP, DNS, SNMP

การสร้าง concurrent server โดยวิธี fork และ select มีข้อแตกต่างอย่างไร ข้อดีข้อเสียของแต่ละแบบคืออะไร

- concurrent Server คือโปรแกรมฝั่ง server ที่รับ request ได้หลายๆตัว พร้อมกัน นั้นก็คือ Multi-Thread Server fork() ทำให้เกิด child process นั้นคือ สร้าง Thread เพื่อช่วยประมวลผล ในทาง socket programming

- ใช้ fork ในสร้าง process เพื่อรับ request เพิ่มจากที่มีอยู่แล้ว select() ทำให้สับเปลี่ยนระหว่าง client ที่ตอบสนองอยู่เรื่อยๆ เช่น เมื่อ client ตัวหนึ่ง เชื่อมต่อเป็นเวลานาน ทำให้ client ท่านอื่นไม่ได้ติดต่อกับ server select() จะช่วยในการสับเปลี่ยนไปยัง client ที่ทำการติดต่อเข้ามา เพื่อใช้งาน server