บทที่ 2
ทฤษฎีและวรรณกรรมที่เกี่ยวข้อง

2.1 แนวคิดและทฤษฎีเรื่องการประเมินต้นทุนซอฟต์แวร์

ตั้งแต่ปี 1965 จนถึงปัจจุบันมีการค้นคว้าและวิจัยทางด้านการประเมินต้นทุนซอฟต์แวร์มากมาย จากโครงการพัฒนาซอฟต์แวร์ต่างๆ ซึ่งการศึกษาเรื่องการประเมินซอฟต์แวร์สามารถแบ่งออกเป็น 2 ส่วน คือ

1. การวัดขนาดของซอฟต์แวร์ (Software Size) ซึ่งสามารถวัดได้ 2 วิธีคือ

1.1 การนับจำนวนบรรทัด (Source Line of Code)

1.2 การวิเคราะห์ฟังก์ชันพอยต์ (Function Point Analysis)

2. การประเมินต้นทุนซอฟต์แวร์ (Software Cost Estimation) ซึ่งสามารถประเมินได้จาก Constructive Cost Model (COCOMO)

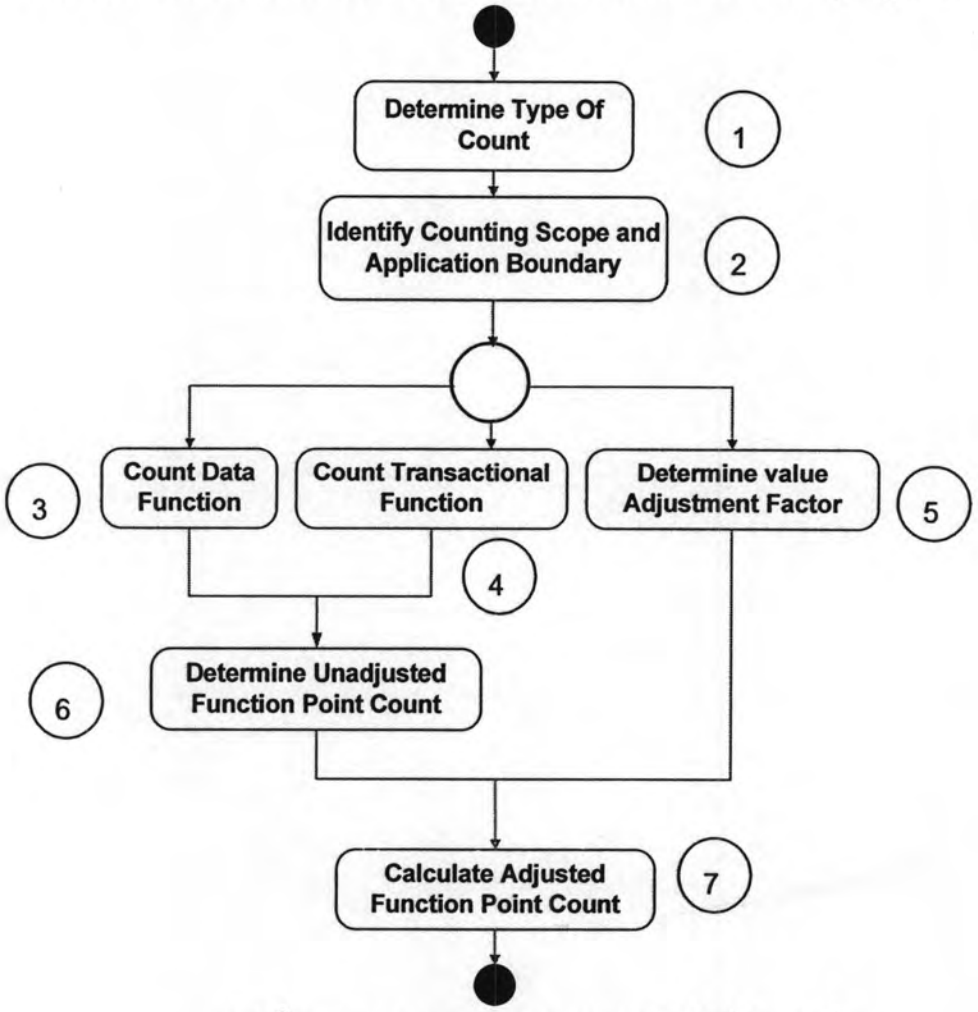
2.1.1 การวิเคราะห์ฟังก์ชันพอยต์ (Function Point Analysis) (Longstreet, 2004)

การวิเคราะห์ฟังก์ชันพอยต์ คือ วิธีการในการประเมินต้นทุนซอฟต์แวร์ โดยวัดขนาดซอฟต์แวร์จากการนับฟังก์ชันการทำงานซอฟต์แวร์ โดยวิธีการนี้ถูกคิดขึ้นในปี ค.ศ.1970 โดย A.J.Albrecht จากบริษัท IBM ได้เสนอการวัดขนาดของซอฟต์แวร์ด้วยการวิเคราะห์ฟังก์ชันพอยต์ (Function Point Analysis : FPA) ซึ่งอาศัยแนวคิดที่ว่ามนุษย์มักจะแก้ปัญหาโดยแบ่งปัญหานั้นออกเป็นส่วนย่อยๆ ซึ่งทำให้ความเข้าใจและแก้ปัญหาเหล่านั้นได้ง่ายขึ้น เนื่องจากเมื่อแบ่งปัญหาออกเป็นส่วนๆ แล้วจะสามารถนำวิธีการ หลักเกณฑ์ หรือนิยามต่างๆ เข้ามาช่วยแก้ปัญหาในส่วนย่อยๆ ได้ อย่างเป็นอิสระต่อกัน ทำให้ง่ายและสะดวกในการทำงาน ซึ่งการวิเคราะห์ฟังก์ชันพอยต์ ก็ใช้หลักการดังกล่าวนี้เช่นกัน โดยแบ่งแยกซอฟต์แวร์เป็นส่วนย่อยๆ แล้วจึงพิจารณานับฟังก์ชันพอยต์จากส่วนย่อยๆ เหล่านั้น ทำให้สามารถวิเคราะห์และทำความเข้าใจได้ง่าย

หลังจากนั้นได้มีการรวมตัวของของนักวิจัยและผู้สนใจก่อตั้งองค์กร IFPUG (International Function Point User Group) และ JEPUG (Japan Function Point Users Group) เพื่อศึกษาและปรับปรุงการวิเคราะห์ฟังก์ชันพอยต์ในการใช้งาน โดยมีการรวบรวม Best Practices ที่ได้ใช้ในการนับฟังก์ชันพอยต์เอาไว้ และผู้ที่สนใจยังสามารถหารายละเอียดเกี่ยวกับฟังก์ชันพอยต์เพิ่มเติม ได้จากเว็บไซต์ http://www.ifpug.org

ขั้นตอนการนับฟังก์ชันพอยต์ (Function Point Counting Procedure)

สำหรับการวิเคราะห์ฟังก์ชันพอยต์นั้นจะแบ่งขั้นตอนออกเป็นส่วนย่อยๆ ดังรูปที่ 2.1



รูปที่ 2.1 แสดงแผนภาพกิจกรรมการนับฟังก์ชันพอยต์

(International Function Point Users Group, 1999)

ขั้นตอนที่1 : Determine Type of Count ฟังก์ชันพอยต์นั้นสามารถนำไปใช้กับโครงการพัฒนาซอฟต์แวร์หรือแอพพลิเคชัน โดยโครงการพัฒนาซอฟต์แวร์แบ่งออกเป็น 3 ประเภท คือ Development, Enhancement และ Maintenance เมื่อปรับให้ตรงกับประเภทของฟังก์ชันพอยต์สามารถแบ่งการนับฟังก์ชันพอยต์ออกเป็น 3 ประเภท คือ Development, Enhancement และ Application

1. ดีเวลล็อฟเมนต์โปรเจ็ค (Development Project Function Point Count) ฟังก์ชันพอยต์สามารถถูกนับได้ในทุกๆ เฟส (Phase) ของการพัฒนาโครงการ ประเภทของการนับฟังก์ชันพอยต์นี้เหมาะกับโครงการที่พัฒนาขึ้นใหม่ ขอบเขตงานที่เพิ่มขึ้นสามารถติดตาม ตรวจสอบและเข้าใจ

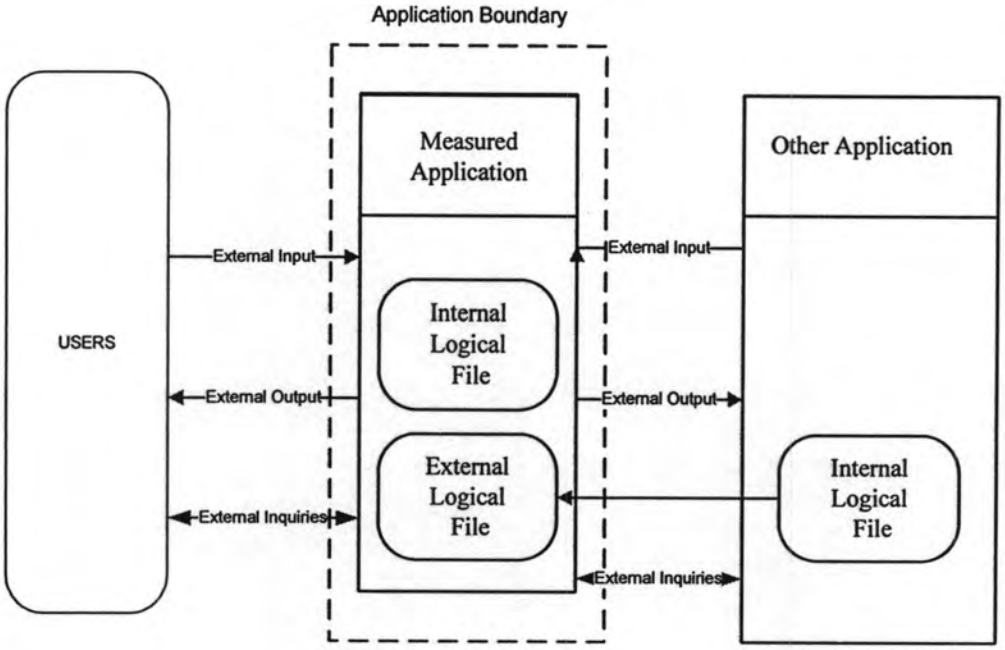
ขนาดของฟังก์ชันการทำงานในทุกๆ เฟสของโครงการ ซึ่งประเภทการนับในรูปแบบนี้มักจะถูกเรียกว่าเป็น " A Baseline Function Point Count "

2. เอนฮานซ์เมนโปรเจ็ค (Enhancement Project Function Point Count) เป็นเรื่องปกติเมื่อซอฟต์แวร์ได้รับการพัฒนาเสร็จสิ้นแล้ว และมีการปรับปรุงคุณภาพหรือปรับปรุงประสิทธิภาพเพิ่มเติม ประเภทของการนับฟังก์ชันพอยต์ประเภทนี้จะนับในส่วนของโครงการที่มีการปรับปรุงเพิ่มเติมจากระบบเดิม

3. แอพพลิเคชัน (Application Function Point Count) การนับแอพพลิเคชันนั้นจะนับเมื่อแอพพลิเคชันนั้นเสร็จสิ้นแล้ว ลักษณะของการนับในรูปแบบนี้อาจใช้การติดตามและตรวจสอบในส่วนของการ Maintenance

ขั้นตอนที่ 2 : Identify Counting Scope and Application Boundary ในขั้นตอนนี้จะเป็นการการกำหนดขอบเขต (Boundary) เพื่อระบุคุณสมบัติ หรือสิ่งที่ต้องการนับด้วยฟังก์ชันพอยต์ดังรูปที่ 2.2

Function Point Model



รูปที่ 2.2 แสดงโมเดลของการวิเคราะห์ฟังก์ชันพอยต์

(Fetcke, Abran และ Nguyen, 1998)

สำหรับในขั้นตอนที่ 3 และขั้นตอนที่ 4 นั้นจะเป็นขั้นตอนการนับฟังก์ชันตามประเภทของฟังก์ชันพอยต์ (Function Point Type) โดยจากรูปที่ 2.2 ทำให้สามารถแบ่งออกเป็น 5 ประเภท โดยในแต่ละประเภทจะมีการกำหนดการให้น้ำหนักที่แตกต่างกันตาม

1. Internal Logical Files (ILF)
2. External Interface Files (EIF)
3. External Input (EI)
4. External Output (EO)
5. External Inquiry (EQ)

ซึ่งก่อนการนับประเภทต่างของฟังก์ชันพอยต์นั้น จะต้องมีการระบุลักษณะรายละเอียดต่างๆ ขององค์ประกอบพื้นฐานที่เกี่ยวข้อง โดยในแต่ละฟังก์ชันพอยต์นั้น มีองค์ประกอบต่างๆ ในแต่ละฟังก์ชันแตกต่างกัน โดยสามารถสรุปในเบื้องต้นได้ดังตารางที่ 2.1

ตารางที่ 2.1 แสดงความสัมพันธ์ของประเภทฟังก์ชันและองค์ประกอบ

Component	DET	RET	FTR
Internal Logical Files (ILF)	√	√	
External Interface Files (EIF)	√	√	
External Input (EI)	√		√
External Output (EO)	√		√
External Inquiry (EQ)	√		√

ขั้นตอนที่ 3 : Count Data Function เป็นขั้นตอนการนับจำนวน Internal Logical Files (ILF) และ External Interface Files (EIF) โดยในส่วนนี้มีการกำหนดลักษณะขององค์ประกอบพื้นฐานที่เกี่ยวข้องกับ ILF และ EIF คือ องค์ประกอบของข้อมูล (Data Element Type: DET) และเรคคอร์ดข้อมูล (Record Element Type: RET) ตามกฎของ IFPUG ดังนี้คือ

การนับ DET ทั้งในส่วนของ ILF และ EIF มีหลักการดังนี้

1. สำหรับแต่ละฟิลด์ข้อมูลนับเป็น 1 DET

2. หากระบบงานตั้งแต่ 2 ระบบขึ้นไปมีการเก็บข้อมูล หรืออ้างอิงไปยัง ILF หรือ EIF เดี่ยวกันแต่แยก DET ให้นับ DET เฉพาะที่มีการเรียกใช้ในแต่ละระบบงาน

3. นับ DET หนึ่ง DET สำหรับแต่ละกลุ่มข้อมูลที่ถูกกำหนดโดยผู้ใช้ เพื่อสร้างความสัมพันธ์กับ ILF หรือ EIF อื่น (อ้างอิงถึงกรณีการนับ Foreign Key)

การนับ RET ทั้งในส่วนของ ILF และ EIF มีหลักการดังนี้

1. นับเป็น 1 RET สำหรับแต่ละกลุ่มย่อยของ ILF หรือ EIF

2. ถ้าไม่มีกลุ่มย่อยให้นับเป็น 1 RET

- **อินเทอร์นอลลอจิคอลไฟล์** (Internal Logical Files : ILF) เป็นกลุ่มของข้อมูลหลักของระบบ หรือข้อมูลควบคุมที่ใช้ภายในแอพพลิเคชัน เช่น แฟ้มข้อมูลหรือฐานข้อมูล ซึ่งมีการสร้าง การเรียกใช้ และปรับปรุงโดยแอพพลิเคชัน โดย ILF จะแสดงถึง Entity ของข้อมูลในลักษณะ 2^{nd} หรือ 3^{rd} นอร์มัลฟอร์ม (Normal Form)

ตัวอย่าง ฟังก์ชันที่มีลักษณะของ ILF

- รายการข้อมูลภายในระบบงาน (Application Transaction Data) เช่น ไฟล์ข้อมูลสินค้าคงคลัง (Inventory Issue Records) ในระบบคลังสินค้า ไฟล์พนักงาน (Employee Record) ในระบบงานบุคลากร (Personal Information System) ซึ่งอาจกล่าวได้ว่าเป็น File ข้อมูลภายในที่ได้รับการจัดเก็บในระบบทั่วๆ ไป
- ข้อมูลควบคุม (Control Information) ที่ได้รับการจัดเก็บภายในซอฟต์แวร์ หรือภายในระบบงาน รวมไปถึงข้อมูลด้านความปลอดภัย เช่น ข้อมูลรหัส (Password Data)

ลักษณะขององค์ประกอบพื้นฐานที่เกี่ยวข้องกับ ILF คือ องค์ประกอบของข้อมูล (Data Element Type: DET) และเรคคอร์ดข้อมูล (Record Element Type: RET)

- องค์ประกอบของข้อมูล (Data Element Type: DET) คือ ฟิลด์ (Field) ข้อมูลที่สนใจในแต่ละฟิลด์
- เรคคอร์ดข้อมูล (Record Element Type: RET) คือ ประเภทของเรคคอร์ดข้อมูล (Data Record) ที่เกี่ยวข้องสัมพันธ์กับฟังก์ชันที่สนใจหรือส่วนที่เป็น Sub Entity ภายใน Entity ที่สนใจ

เนื่องจากการกำหนดระดับความซับซ้อนมีผู้วิจัยได้ศึกษาและนำเสนอออกมาในหลายรูปแบบแต่การกำหนดระดับความซับซ้อนที่ได้รับความนิยมและได้รับการยอมรับคือ การกำหนดระดับความซับซ้อน ILF ขององค์กร IFPUG (International Function Point User Group) โดยองค์ประกอบใดมีความซับซ้อนมากค่าน้ำหนักที่ให้จะมีค่ามาก และในทางกลับกันถ้าองค์ประกอบใดมีความซับซ้อนน้อยค่าน้ำหนักที่ให้ก็จะมีค่าน้อยเช่นกัน

การกำหนดระดับความซับซ้อนที่จะใช้สำหรับ ILF จะขึ้นอยู่กับ Data Element Type (DET) และ Record Element Type (RET) ซึ่งแบ่งออกเป็น 3 อัตราคือ Low, Average และ High ดังตารางที่ 2.2

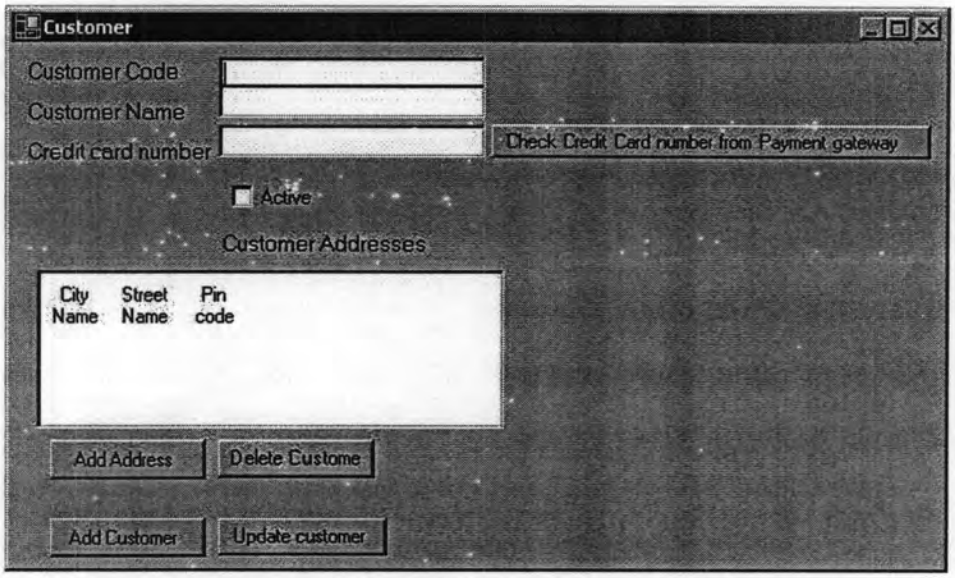
ตารางที่ 2.2 แสดงการกำหนดระดับความซับซ้อนของ ILF (Longstreet, 2004)

Record Element Type (RET)	จำนวน Data Element Type (DET)		
	1-19	20-50	>50
1	ต่ำ(7)	ต่ำ(7)	ปานกลาง(10)
2 – 5	ต่ำ(7)	ปานกลาง(10)	สูง (15)
> 5	ปานกลาง(10)	สูง (15)	สูง (15)

โดยตัวเลขในวงเล็บ () คือ ตัวถ่วงน้ำหนักที่อยู่ในแต่ละระดับความซับซ้อน

- ระดับต่ำ ตัวถ่วงน้ำหนัก คือ 7
- ระดับปานกลาง ตัวถ่วงน้ำหนัก คือ 10
- ระดับสูง ตัวถ่วงน้ำหนัก คือ 15

ตัวอย่างการนับ ILF



รูปที่ 2.3 แสดงแบบฟอร์มของลูกค้า (Customer Form)

จากรูปที่ 2.3 พบว่าจากหน้าจอ (Screen) จะมีทั้งหมด 2 ILFs คือ Customer และ Customer Address ซึ่งมีรายละเอียดดังต่อไปนี้

Customer : ILF ในส่วนนี้จะพิจารณา DET และ RET จาก

1 . ปุ่ม Add เท่ากับ 2 DETs : Add Address และ Add Customer

2. ปุ่ม Update Customer เท่ากับ 1 DET

3. ปุ่ม Delete Customer เท่ากับ 1 DET

4. ปุ่ม The Credit Check เท่ากับ 1 DET

5. Address List Box เท่ากับ 1 DET

6. Check Box Active เท่ากับ1 DET

7. Text Boxes เท่ากับ 3 DETs : Customer Code, Customer Name และ Credit Card Number

8. มี 1 RET คือ Customer Address ซึ่งในที่นี้ไม่ได้นับรวม Credit Check เป็น RET เนื่องจาก Credit Check ไม่ได้เป็น Sub Element ของ Customer

ดังนั้นในส่วน ILF ของ Customer มีค่า DET เท่ากับ 10 และ RET เท่ากับ 1 จากตาราง 2_2 ทำให้ ILF ส่วนนี้มีระดับความซับซ้อนต่ำ จึงมีฟังก์ชันทั้งหมด 1x7 = 7 FP

Customer Address : ILF ในส่วนนี้จะพิจารณา DET และ RET จาก

1. ในส่วนนี้จะพิจารณา Column Name ใน List Box ทั้งหมดเท่ากับ 3 DETs : City Name, Street Name และ Pin Code

2. ในส่วนนี้จะไม่มี RET

ดังนั้นในส่วน ILF ของ Customer Address มีค่า DET เท่ากับ 3 และ RET เท่ากับ 0 แต่กฎของ IFPUG กำหนดว่าถ้าไม่มีกลุ่มย่อยให้นับเป็น 1 RET จากตาราง 2_2 ทำให้ ILF ส่วนนี้มีระดับความซับซ้อนต่ำ จึงมีฟังก์ชันทั้งหมด 1x7 = 7 FP

- **เอ็กเทอร์นอลอินเตอร์เฟสไฟล์** (External Interface Files : EIF) จะมีลักษณะเหมือน ILF แต่จะอยู่ภายนอกระบบ (External) โดยมีระบบงานอื่นเป็นตัวจัดเก็บ เป็นไฟล์ข้อมูลที่มีวัตถุประสงค์เพื่อใช้อ้างอิงในโปรแกรมเท่านั้น ไม่มีการแก้ไขเปลี่ยนแปลงหรือเพิ่มเติมใดๆ โดย EIF จะเป็น ILF ของแอพพลิเคชันอื่น

ตัวอย่าง ฟังก์ชันที่มีลักษณะของ EIF

- รายการข้อมูลภายนอกระบบงาน (Application Transaction Data) ที่ระบบมีการเรียกใช้ แต่ได้รับการจัดเก็บไว้ภายนอกระบบงาน อาจจะมีการเรียกอ่าน (Read) มาจากระบบอื่นหรือมีการกระจาย (Extract) มาจากระบบงานอื่น

- ข้อมูลควบคุม (Control Information) ที่ได้รับการจัดเก็บไว้ภายนอกซอฟต์แวร์หรือภายนอกระบบงาน รวมไปถึงข้อมูลด้านความปลอดภัย เช่น ข้อมูลรหัส (Password Data)

ลักษณะขององค์ประกอบพื้นฐานที่เกี่ยวข้องกับ EIF คือ องค์ประกอบของข้อมูล (Data Element Type: DET) และเรคคอร์ดข้อมูล (Record Element Type: RET) เหมือนกับ ILF

เนื่องจากการกำหนดระดับความซับซ้อนมีผู้วิจัยได้ศึกษาและนำเสนอออกมาในหลายรูปแบบแต่การกำหนดระดับความซับซ้อนที่ได้รับความนิยมและได้รับการยอมรับคือ การกำหนดระดับความซับซ้อน EIF ขององค์กร IFPUG (International Function Point User Group)โดยองค์ประกอบใดมีความซับซ้อนมากค่าน้ำหนักที่ให้จะมีค่ามาก และในทางกลับกันถ้าองค์ประกอบใดมีความซับซ้อนน้อยค่าน้ำหนักที่ให้ก็จะมีค่าน้อยเช่นกัน

การกำหนดระดับความซับซ้อนที่จะใช้สำหรับ EIF จะขึ้นอยู่กับ Data Element Type (DET) และ Record Element Type (RET) ซึ่งแบ่งออกเป็น 3 อัตราคือ Low, Average และ High ดังตารางที่ 2.3

ตารางที่ 2.3 แสดงการกำหนดระดับความซับซ้อนของ EIF (Longstreet, 2004)

Record Element Type (RET)	Data Element Type (DET)		
	1-19	20-50	>50
1	ต่ำ(5)	ต่ำ(5)	ปานกลาง(7)
2 – 5	ต่ำ(5)	ปานกลาง(7)	สูง (10)
> 5	ปานกลาง(7)	สูง (10)	สูง (10)

โดยตัวเลขในวงเล็บ () คือ ตัวถ่วงน้ำหนักที่อยู่ในแต่ละระดับความซับซ้อน

- ระดับต่ำ ตัวถ่วงน้ำหนัก คือ 5
- ระดับปานกลาง ตัวถ่วงน้ำหนัก คือ 7
- ระดับสูง ตัวถ่วงน้ำหนัก คือ 10

ตัวอย่างการนับ EIF

จากรูปที่ 2.3 พบว่าจากหน้าจอ (Screen) จะมีทั้งหมด 1 EIF คือ Credit Card Information ซึ่งมีรายละเอียดดังต่อไปนี้

Credit Card Information : EIF ในส่วนนี้จะพิจารณา DET และ RET จากการอ้างอิงของ Credit Card Information ซึ่ง File นี้มีการอ้างอิงสำหรับ Credit Card Check เพียงอย่างเดียว ดังนั้นจึงมี 1 DET และ มี RET เท่ากับ 0

ดังนั้นในส่วน EIF ของ Credit Card Information มีค่า DET เท่ากับ 1 และ RET เท่ากับ 0 แต่กฎของ IFPUG กำหนดว่าถ้าไม่มีกลุ่มย่อยให้นับเป็น 1 RET จากตาราง 2_3 ทำให้ EIF ส่วนนี้มีระดับความซับซ้อนต่ำ จึงมีฟังก์ชันทั้งหมด 1x5 = 5

ขั้นตอนที่ 4 : Count Transactional Function เป็นขั้นตอนการนับจำนวน External Input (EI), External Output (EO) และ External Inquiry (EQ)

- **เอ็กซ์เทอร์นอลอินพุท** (External Input : EI) คือ เป็นข้อมูลจากภายนอกที่แอพพลิเคชันต้องการใช้เพื่อประมวลผล โดยข้อมูลที่ผ่านเข้ามานั้นจะใช้สำหรับ ปรับปรุงหรือเปลี่ยนแปลง ILF ซึ่งในกระบวนการนี้จะมีการดึงข้อมูล External Inquiry ขึ้นมาก่อนการแก้ไข

ตัวอย่าง ฟังก์ชันที่มีลักษณะของ EI

- ข้อความ (Message) จากระบบงานอื่นที่ส่งมาเพื่อให้ระบบทำงาน
- ไฟล์ข้อมูลรายการ (Transaction Files) จากระบบงานอื่น
- ข้อมูลที่ส่งมาจากภายนอก
- รายการข้อมูลที่ใช้ในการรักษา ILF
- การดูและรักษา (Maintenance) ของข้อมูล ILF ที่นับรวมข้อความช่วยเหลือ (Help) ไฟล์ข้อความ (Message File) รวมไปถึงพารามิเตอร์ต่างๆ

ลักษณะขององค์ประกอบพื้นฐานที่เกี่ยวข้องกับ EI คือ องค์ประกอบของข้อมูล (Data Element Type: DET) ประเภทของไฟล์ข้อมูลอ้างอิง (File Type of Reference: FTR) ตามกฎของ IFPUG ดังนี้คือ

องค์ประกอบของข้อมูล (Data Element Type: DET) คือ ฟิลด์ (Field) ข้อมูลที่สนใจในแต่ละฟิลด์

การนับ DET มีหลักการดังนี้

1. นับเป็น 1 DET สำหรับแต่ละฟิลด์หรือแอททริบิวต์ของข้อมูลที่ได้รับการกำหนดไว้ ในลักษณะ User-Recognizable คือ มีการกำหนดมาจากความต้องการของซอฟต์แวร์หรือความต้องการ (Requirement) และไม่มีซ้ำ รวมทั้งนับในส่วนของ Foreign Key ที่เกิดขึ้นระหว่างระบบงาน เพื่อให้กระบวนการย่อยสุดของระบบงานได้ทำงานอย่างสมบูรณ์

2. ไม่นับรวม DET สำหรับฟิลด์ข้อมูลที่ผู้ใช้ไม่ได้ทำการกรอกข้อมูล เช่น ฟิลด์วันที่ที่ระบบขึ้นให้อัตโนมัติ

3. นับเป็น 1 DET สำหรับแต่ละ Capability หรือความสามารถในการส่งข้อความตอบรับ หรือ System Response Message ไปยังภายนอกของระบบงานเพื่อแจ้งเตือน Error หรือเพื่อทำการยืนยันว่ากระบวนการได้ทำงานเสร็จสมบูรณ์ หรือทำการครวจสอบกระบวนการ

4. นับเป็น 1 DET สำหรับแต่ละ Capability .ในการกำหนดการทำงานที่จัดการโดย EI เช่น ในส่วนของ Command Line นั้น หากมีหลายๆ Command Line ทำงานร่วมกันเพื่อทำกิจกรรมหนึ่งอย่าง จะนับเพียงแค่ 1 DET ไม่นับตามจำนวน Command Line

ประเภทของไฟล์ข้อมูลอ้างอิง (File Type of Reference: FTR) คือ ประเภทของไฟล์ที่เกี่ยวข้องในแต่ละ EI โดยไฟล์ ILF มีการอ่านหรือจัดการในส่วนการดูแลรักษาโดย EI หรือไฟล์ EIF ที่อ่านเข้ามาโดย EI ซึ่งการนับ FTR สำหรับ EI นั้นมีข้อกำหนดดังนี้

การนับ FTR มีหลักการดังนี้

1. นับเป็น 1 FTR สำหรับแต่ละประเภทของ ILF ที่ได้รับการรักษาไว้ภายใต้กระบวนการของ EI

2. นับเป็น 1 FRT สำหรับแต่ละประเภทของ ILF หรือ EIF ที่ได้รับการอ้างอิงจากกระบวนการของ EI

3. นับเพียง 1 FTR สำหรับแต่ละ ILF ที่ได้รับการอ้างอิง หรือเรียกอ่าน หรือเก็บรักษาไว้ภายใต้ EI

เนื่องจากการกำหนดระดับความซับซ้อนมีผู้วิจัยได้ศึกษาและนำเสนอออกมาในหลายรูปแบบแต่การกำหนดระดับความซับซ้อนที่ได้รับความนิยมและได้รับการยอมรับคือ การกำหนดระดับความซับซ้อน EI ขององค์กร IFPUG (International Function Point User Group)โดย

องค์ประกอบใดมีความซับซ้อนมากค่าน้ำหนักที่ให้จะมีค่ามาก และในทางกลับกันถ้าองค์ประกอบใดมีความซับซ้อนน้อยค่าน้ำหนักที่ให้ก็จะมีค่าน้อยเช่นกัน

การกำหนดระดับความซับซ้อนที่จะใช้สำหรับ EI จะขึ้นอยู่กับ Data Element Type (DET) และ File Type Reference (FTR) ซึ่งแบ่งออกเป็น 3 อัตราคือ Low, Average และ High ดังตารางที่ 2.4

ตารางที่ 2.4 แสดงการกำหนดระดับความซับซ้อนของ EI (Longstreet, 2004)

File Type Reference (FTR)	Data Element Type (DET)		
	1-4	5 - 15	>15
< 2	ต่ำ(3)	ต่ำ(3)	ปานกลาง(4)
2	ต่ำ(3)	ปานกลาง(4)	สูง (6)
> 2	ปานกลาง(4)	สูง (6)	สูง (6)

โดยตัวเลขในวงเล็บ () คือ ตัวถ่วงน้ำหนักที่อยู่ในแต่ละระดับความซับซ้อน

- ระดับต่ำ ตัวถ่วงน้ำหนัก คือ 3
- ระดับปานกลาง ตัวถ่วงน้ำหนัก คือ 4
- ระดับสูง ตัวถ่วงน้ำหนัก คือ 6

ตัวอย่างการนับ EI

จากรูปที่ 2.3 พบว่าจากหน้าจอ (Screen) จะมีทั้งหมด 1 EI คือ Customer ซึ่งมีรายละเอียดดังต่อไปนี้

Customer : EI ในส่วนนี้จะพิจารณา DET และ FTR จาก

1 .ปุ่ม Add เท่ากับ 2 DETs : Add Address และ Add Customer

2. ปุ่ม Update Customer เท่ากับ 1 DET

3. ปุ่ม Delete Customer เท่ากับ 1 DET

4. ปุ่ม The Credit Check เท่ากับ 1 DET

5. Address List Box เท่ากับ 1 DET

6. Check Box Active เท่ากับ1 DET

7. Text Boxes เท่ากับ 3 DETs : Customer Code, Customer Name และ Credit Card Number

8. มี 2 FTRs คือ Address และ Credit Card Information

ดังนั้นในส่วน EI ของ Customer มีค่า DET เท่ากับ 10 และ FTR เท่ากับ 2 จาก ตาราง 2_4 ทำให้ EI ส่วนนี้มีระดับความซับซ้อนปานกลาง จึงมีฟังก์ชันทั้งหมด 1x4 = 4 FP

- **เอ็กซ์เทอร์นอลเอาท์พุท** (External Output : EO) เป็นกระบวนการพื้นฐานที่มีการส่งข้อมูล หรือคำสั่งควบคุม (Control Information) ไปยังภายนอกระบบงาน เพื่อที่จะแสดงข้อมูลกลับไปยังผู้ใช้ผ่านกระบวนการทำงานของระบบงาน โดยกระบวนการส่งข้อมูลออกไปยังภายนอกนั้นมักจะเกี่ยวข้องกับการคำนวณทางคณิตศาสตร์ หรือ เกี่ยวกับสูตรในการสร้างคำนวณหรือเปลี่ยนแปลงข้อมูล ILF ในระบบงาน

โดยกระบวนการในการทำงานของ EO เช่น การออกรายงานหนึ่งรายงาน ถือว่าเป็นหนึ่ง EO ถึงแม้ว่ารายงานนั้นจะประกอบด้วยหน้ารายงานเพียงหน้าเดียว หรือหลายๆ หน้า หากกระบวนการในการเรียกอ่านของระบบงาน เพื่อออกรายงานเป็นกระบวนการเดียวกัน หรือฟังก์ชันเดียวกันจะถือว่ามีเพียง 1 EOโดยในส่วนของกระบวนการที่เกี่ยวข้องกับ EO นั้นมีลักษณะเช่นเดียวกับ EI แต่จะมองเป็นกระบวนการที่ต้องการเตียมข้อมูลออกสู่ภายนอก

ตัวอย่าง ฟังก์ชันที่มีลักษณะของ EO

- รายงานที่ต้องมีการคำนวณ หรือมีการสร้างอัลกอริทึมเพื่อรองรับการออกรายงาน เช่น รายงานประเภทประจำสัปดาห์ เช่น Weekly Sale Report
- การส่งผ่านข้อมูล ไฟล์ หรือข้อความไปยังระบบงานอื่น โดยข้อมูลเหล่านั้นระบบจะต้องสร้างกระบวนการเพื่อคำนวณ หรือทำให้เกิดการเปลี่ยนแปลงปรับปรุงข้อมูล
- ข้อมูลที่ได้จากการคำนวณ หรือ Derived Data ที่มีการแสดงที่หน้าจอ หรือผ่านค่าไปยังไฟล์
- ภาพกราฟิกของกราฟแท่ง หรือแบบวงกลม หรือกราฟแบบต่างๆ

ลักษณะขององค์ประกอบพื้นฐานที่เกี่ยวข้องกับ EO คือ องค์ประกอบของข้อมูล (Data Element Type: DET) ประเภทของไฟล์ข้อมูลอ้างอิง (File Type of Reference: FTR) ตามกฎของ IFPUG ดังนี้คือ

การนับ DET มีหลักการดังนี้คือ

1. นับเป็น 1 DET สำหรับแต่ละฟิลด์หรือแอททริบิวต์ของข้อมูลที่ได้รับการกำหนดไว้ ในลักษณะ User-Recognizable คือ มีการกำหนดมาจากความต้องการของซอฟต์แวร์หรือความต้องการ (Requirement) และไม่มีซ้ำ รวมทั้งนับในส่วนของ Foreign Key ที่เกิดขึ้นระหว่างระบบงาน เพื่อให้กระบวนการย่อยสุดของระบบงานได้ทำงานอย่างสมบูรณ์

2. นับเป็น 1 DET สำหรับแต่ละฟิลด์ หรือแอททรบิวต์ที่อยู่ภายนอกระบบงาน

3. หาก DET นั้นๆ เกิดขึ้นทั้งภายนอก และเข้าสู่ระบบงานให้นับเพียงครั้งเดียว

4. นับเป็น 1 DET สำหรับแต่ละ Capability หรือความสามารถในการส่งข้อความตอบรับ หรือ System Response Message ไปยังภายนอกของระบบงานเพื่อแจ้งเตือน Error หรือเพื่อทำการยืนยันว่ากระบวนการได้ทำงานเสร็จสมบูรณ์ หรือทำการตรวจสอบกระบวนการ

5. นับเป็น 1 DET สำหรับแต่ละ Capability ในการกำหนดการทำงานที่จัดการโดย EO เช่น นับเป็น 1 DET สำหรับ OK Button, Function Key, Action Key หรือ Mouse Click ที่ให้ผลเท่ากับการตอบตกลง

6. ไม่นับการทำ Paging, Paging Commands หรือ Position Information รวมไปถึงการทำ Page Number

7. นับเพียง 1 DET สำหรับฟิลด์ข้อมูลที่มีการจัดเก็บไว้หลายที่ แต่เป็นเพียงหนึ่งฟิลด์ในเชิงตรรกะ หรือแม้แต่ฟิลด์ที่ให้เพียงความหมายเดียวแต่จัดเก็บแยกฟิลด์ เช่น Address ที่สามารถแยกเป็น Street, City และ Zip Code แต่อยู่ภายใน Address Label

8. นับเพียง 1 DET สำหรับข้อมูลที่ประกอบด้วยคำ ประโยค หรือย่อหน้า

การนับ FTR มีหลักการดังนี้คือ

1. นับเป็น 1 FTR สำหรับแต่ละประเภทของ ILF ที่ได้รับการเก็บรักษาไว้ภายใต้กระบวนการของ EO

2. นับเป็น 1 FTR สำหรับแต่ละประเภทของ ILF หรือ EIF ที่ได้รับการอ้างอิงจากกระบวนการของ EO

3. นับเพียง 1 FRT สำหรับแต่ละ ILF ที่ทั้งได้รับการอ้างอิง หรือเรียกอ่าน หรือเก็บรักษาไว้ภายใต้ EO

เนื่องจากการกำหนดระดับความซับซ้อนมีผู้วิจัยได้ศึกษาและนำเสนอออกมาในหลายรูปแบบแต่การกำหนดระดับความซับซ้อนที่ได้รับความนิยมและได้รับการยอมรับคือ การกำหนดระดับความซับซ้อน EO ขององค์กร IFPUG (International Function Point User Group) โดยองค์ประกอบใดมีความซับซ้อนมากค่าน้ำหนักที่ให้จะมีค่ามาก และในทางกลับกันถ้าองค์ประกอบใดมีความซับซ้อนน้อยค่าน้ำหนักที่ให้ก็จะมีค่าน้อยเช่นกัน

การกำหนดระดับความซับซ้อนที่จะใช้สำหรับ EO จะขึ้นอยู่กับ Data Element Type (DET) และ File Type Reference (FTR) ซึ่งแบ่งออกเป็น 3 อัตราคือ Low, Average และ High ดังตารางที่ 2.5

ตารางที่ 2.5 การกำหนดระดับความซับซ้อนของ EO (Longstreet, 2004)

File Type Reference (FTR)	Data Element Type (DET)		
	1-5	6 - 19	>19
< 2	ต่ำ(4)	ต่ำ(4)	ปานกลาง(5)
2-3	ต่ำ(4)	ปานกลาง(5)	สูง (7)
> 3	ปานกลาง(5)	สูง (7)	สูง (7)

โดยตัวเลขในวงเล็บ () คือ ตัวถ่วงน้ำหนักที่อยู่ในแต่ละระดับความซับซ้อน

- ระดับต่ำตัวถ่วงน้ำหนัก คือ 4
- ระดับปานกลาง ตัวถ่วงน้ำหนัก คือ 5
- ระดับสูง ตัวถ่วงน้ำหนัก คือ 7

ตัวอย่างการนับ EO

จากรูปที่ 2_3 พบว่าจากหน้าจอ (Screen) จะไม่มี EO เนื่องจากระบบลูกค้า (Customer System) ไม่มีการส่งข้อมูลออกไปนอกระบบ (External System) และไม่มีการ เพิ่ม (Add) หรือ ปรับปรุง (Update) นอกระบบ

- **เอ็กซ์เทอร์นอลอินไควรี** (External Inquiry : EQ) มีลักษณะเป็นได้ทั้ง Input และ Output โดย EQ จะเป็นผลมาจากการดึงข้อมูล (Data Retrieval) จาก ILF หรือ EIF โดยมีวัตถุประสงค์เพื่อทำการนำเสนอข้อมูลไปยังผู้ใช้ ในกระบวนการนี้จะไม่มีการจัดเก็บสูตรทางคณิตศาสตร์หรือการคำนวณ และไม่มีการ Derived Data

ตัวอย่าง ฟังก์ชันที่มีลักษณะของ EQ

- ข้อมูลรายการ (Transaction Data) ที่มีการเรียกดึงจาก ILF หรือ EIF เพื่อแสดงฟังก์ชันของผู้ใช้งาน เช่น Print, Browse และ View เป็นต้น
- รายงานที่มีการออกรายงานเป็นประจำ โดยไม่มีสูตรการคำนวณหรือการแปลงค่าข้อมูล
- ไฟล์ท่าส่งผ่านไปยังระบบงานอื่น โดยไม่มีสูตรทางคณิตสาสตร์ หรือการคำนวณ หรือการแปลงค่าข้อมูลใดๆ
- DET เช่น ค่าที่ได้จากการ search การเปลี่ยนสีของ Font หรือ Screen การคลิกเมาส์

ลักษณะขององค์ประกอบพื้นฐานที่เกี่ยวข้องกับ EI คือ องค์ประกอบของข้อมูล (Data Element Type: DET) ประเภทของไฟล์ข้อมูลอ้างอิง (File Type of Reference: FTR) ซึ่งมีลักษณะเช่นเดียวกับ EO

เนื่องจากการกำหนดระดับความซับซ้อนมีผู้วิจัยได้ศึกษาและนำเสนอออกมาในหลายรูปแบบแต่การกำหนดระดับความซับซ้อนที่ได้รับความนิยมและได้รับการยอมรับคือ การกำหนดระดับความซับซ้อน EQ ขององค์กร IFPUG (International Function Point User Group)โดยองค์ประกอบใดมีความซับซ้อนมากค่าน้ำหนักที่ให้จะมีค่ามาก และในทางกลับกันถ้าองค์ประกอบใดมีความซับซ้อนน้อยค่าน้ำหนักที่ให้ก็จะมีค่าน้อยเช่นกัน

การกำหนดระดับความซับซ้อนที่จะใช้สำหรับ EQ จะขึ้นอยู่กับ Data Element Type (DET) และ File Type Reference (FTR) ซึ่งแบ่งออกเป็น 3 อัตราคือ Low, Average และ High ดังตารางที่ 2.6

ตารางที่ 2.6 แสดงการกำหนดระดับความซับซ้อนของ EQ (Longstreet, 2004)

File Type Reference (FTR)	Data Element Type (DET)		
	1-5	6 - 19	>19
< 2	ต่ำ(3)	ต่ำ(3)	ปานกลาง(4)
2-3	ต่ำ(3)	ปานกลาง(4)	สูง (6)
> 3	ปานกลาง(4)	สูง (6)	สูง (6)

โดยตัวเลขในวงเล็บ () คือ ตัวถ่วงน้ำหนักที่อยู่ในแต่ละระดับความซับซ้อน

- ระดับต่ำ ตัวถ่วงน้ำหนัก คือ 3
- ระดับปานกลาง ตัวถ่วงน้ำหนัก คือ 4

- ระดับสูง ตัวถ่วงน้ำหนัก คือ 6

ตัวอย่างการนับ EQ

จากรูปที่ 2_3 พบว่าจากหน้าจอ (Screen) จะไม่มี EQ เนื่องในขณะนี้ยังไม่มีส่วนของการออกรายงาน

ขั้นตอนที่ 5 : Determine Unadjusted Function Point Count (UAF)

ในขั้นตอนนี้จะนำค่าที่ได้จากกการประเมินในแต่ละ Internal Logical Files (ILF), External Interface Files (EIF), External Input (EI), External Output (EO) และ External Inquiry (EQ) นำมาใส่ตารางที่ 2.7 เพื่อคำนวณค่าฟังก์ชันพอยต์ที่ยังไม่มีการปรับค่า

ตารางที่ 2.7 แสดงการนับค่าฟังก์ชันพอยต์ (Albrecht and Gaffney,1983)

ชนิด	ความหมาย	ความซับซ้อน			รวม
		ต่ำ	กลาง	สูง	
EI	External Input	_____x 3	_____x 4	_____x 6	_____
EO	External Output	_____x 3	_____x 5	_____x 7	_____
ILF	Internal Logical Files	_____x 7	_____x 10	_____x 15	_____
ELF	External Interface File	_____x 5	_____x 7	_____x 10	_____
EQ	External Inquiry	_____x 3	_____x 4	_____x 6	_____
UAF			รวมค่าฟังก์ชันพอยต์ที่ยังไม่ปรับค่า		_____

ตัวอย่างการนับค่าฟังก์ชันพอยต์

จากรูปที่ 2.3 พบว่าจากหน้าจอ (Screen) สามารถคำนวณฟังก์ชันพอยต์ที่ยังไม่ปรับค่าได้ดังนี้

ชนิด	ความหมาย	ความซับซ้อน			รวม
		ต่ำ	กลาง	สูง	
EI	External Input	____x 3	__1_x 4	____x 6	__4__
EO	External Output	____x 3	____x 5	____x 7	__0__
ILF	Internal Logical Files	__2_x 7	____x 10	____x 15	__14__
ELF	External Interface File	__1_x 5	____x 7	____x 10	__5__
EQ	External Inquiry	____x 3	____x 4	____x 6	__0__
UAF			รวมค่าฟังก์ชันพอยต์ที่ยังไม่ปรับค่า		__23__

ขั้นตอนที่ 6 : Determine Value Adjustment Factor (VAF) เป็นการกำหนดค่า VAF ให้กับซอฟต์แวร์ ซึ่งค่าดังกล่าวถูกแบ่งตามลักษณะและประเภทของซอฟต์แวร์ (General System Characteristics) ซึ่งมีอยู่ด้วยกัน 14 ประการ คือ

1. Data Communication
2. Distributed Data Processing
3. Performance
4. Heavily Used Configuration
5. Transaction Rate
6. On-Line Data Entry
7. End-User Efficiency
8. On-Line Update
9. Complex Processing
10 .Reusability
11. Installation Ease
12. Operation Ease

13. Multiple Sites

14. Facilitate change

โดยแต่ละประเภทจะมีการให้ค่า Degree of Influence ตั้งแต่ 0 ถึง 5 และค่านี้จะใช้ในการคำนวณฟังก์ชันพอยต์สุดท้ายต่อไป

สูตรที่ใช้ในการคำนวณ Value of Adjustment Factor (VAF)

$$VAF = 0.65 + [\{\sum_{i=1}^{14} C_i\} / 100]$$

เมื่อ C_i คือ ค่าระดับของอิทธิพลของแต่ละปัจจัย

i คือ ชนิดของปัจจัยมีทั้งหมด 14 ชนิด

VAF จะมีค่าอยู่ในระหว่าง 0.65 -1.35

ลักษณะของระบบโดยทั่วไป (General System Characteristics) ซึ่งมีอัตรากำหนดไว้โดยที่ค่าแต่ละค่าจะถูกประเมินด้วยระดับการมีอิทธิพลต่อแอพพลิเคชัน โดยมีระดับการดังนี้

0 หมายถึง ไม่มีอิทธิพลต่อแอพพลิเคชัน

1 หมายถึง มีอิทธิพลต่อแอพพลิเคชันเพียงเล็กน้อยหรือไม่ได้ตั้งใจ

2 หมายถึง มีอิทธิพลต่อแอพพลิเคชันปานกลาง

3 หมายถึง มีอิทธิพลต่อแอพพลิเคชันในระดับทั่วๆ ไป

4 หมายถึง มีอิทธิพลต่อแอพพลิเคชันค่อนข้างมากและเห็นได้เด่นชัด

5 หมายถึง มีอิทธิพลต่อแอพพลิเคชันอย่างมาก

1. **Data Communication** จะมองเรื่องของจำนวนของสิ่งที่ช่วยอำนวยความสะดวกในการถ่ายโอนข้อมูล หรือแลกเปลี่ยนข้อมูลระหว่างแอพพลิเคชันหรือระบบ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.8

ตารางที่ 2.8 แสดงการกำหนดระดับปัจจัยในส่วนของ Data Communication (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	แอพพลิเคชันทั้งหมดรันแบบ Batch หรือ standalone PC
1	แอพพลิเคชันเป็นแบบ Batch แต่มีการใช้ลักษณะของ Remote Data Entry หรือ Remote Printing
2	แอพพลิเคชันเป็นแบบ Batch แต่มีการใช้ลักษณะของ Remote Data Entry และ Remote Printing
3	แอพพลิเคชันเป็นแบบ Online Data Collection หรือ Teleprocessing จากส่วน Front-end สู่ Batch
4	แอพพลิเคชันมีลักษณะของ Teleprocessing ซึ่งมากกว่าในส่วนของ Front –end แต่สนับสนุนการติดต่อ Teleprocessing Protocol เพียงชนิดเดียวเท่านั้น
5	แอพพลิเคชันมีลักษณะของ Teleprocessing ซึ่งมากกว่าในส่วนของ Front –end แต่สนับสนุนการติดต่อ Teleprocessing Protocol ที่หลากหลายมากขึ้น

ข้อเสนอแนะ (Comment)

- TCP/IP (Transmission Control Protocol และ internet protocol) นั้นใช้ Local Protocol หลากหลาย เช่น Ethernet, Netware, AppleTalk, DECnet และอื่นๆ ซึ่งเป็นตัวอย่างของ TP (Tele Processing)
- แอพพลิเคชันนั้นอนุญาติให้ทำการ Query ข้อมูลออนไลน์ผ่านทางเว็บไซต์ และการ Query ผ่านส่วน Local นั้นควรจะให้ระดับของปัจจัยที่มีไว้สำหรับปรับค่าเท่ากับ 3

- แอพพลิเคชันนั้นอนุญาติให้มีการ อัพเดท ILF ผ่านทางอินเทอร์เน็ต (Internet) และ การอัพเดทในส่วนของ Local นั้นควรจะให้ระดับปัจจัยที่มีไว้สำหรับปรับค่าเท่ากับ 5

2. **Distributed Data Processing** จะมองในเรื่องการจัดการข้อมูลในเชิง Distributed และขั้นตอนหรือหน้าที่ในการจัดการ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.9

ตารางที่ 2.9 แสดงการกำหนดระดับปัจจัยในส่วนของ Distributed Data Processing (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	แอพพลิเคชันมาสามารถช่วยในการถ่ายข้อมูลหรือช่วยในการดำเนินการระหว่างกันได้
1	แอพพลิเคชันสามารถเตรียมข้อมูลสำหรับ End User เพื่อดำเนินขั้นตอนต่อไปในส่วนอื่นๆ เช่น PC Spreadsheets และ PC DBMS
2	แอพพลิเคชันสามารถเตรียมข้อมูลเพื่อถ่ายโอน และถ่ายโอนไปยังขั้นตอนอื่นๆ ในระบบเพื่อประมวลผลได้ (ไม่เพียงแต่ถ่ายโอนสำหรับ End User)
3	แอพพลิเคชันมีลักษณะของ Distributed processing และการถ่ายโอนข้อมูลเป็นแบบออนไลน์แต่เป็นในทิศทางเดียว
4	แอพพลิเคชันมีลักษณะของ Distributed processing และการถ่ายโอนข้อมูลเป็นแบบออนไลน์แต่เป็นใน 2 ทิศทาง
5	แอพพลิเคชันมีการจัดการการถ่ายโอนข้อมูลแบบไดนามิกขึ้นอยู่กับแต่ละส่วนของระบบ

ข้อเสนอแนะ (Comment)

- การทำสำเนาไฟล์จากเครื่องเมนเฟรมไปสู่ Local PC โดยผ่านทาง Internet หรือ Intranet นั้น ควรจะให้ระดับปัจจัยที่มีไว้สำหรับปรับค่า เท่ากับ 2
- การอ่านข้อมูลจากเครื่อง Client ได้ผ่านทาง Internet หรือ Intranet นั้นควรจะให้ระดับปัจจัยที่มีไว้สำหรับปรับค่า เท่ากับ 3
- การอ่านข้อมูลและสามารถอัพเดทได้ผ่านทาง Internet หรือ Intranet นั้นควรจะให้ระดับปัจจัยที่มีไว้สำหรับปรับค่า เท่ากับ 4
- ความสามารถของแอพพลิเคชันที่ขึ้นอยู่กับแหล่งข้อมูลที่สามารถเข้าถึงโดยผ่าน Local บน Server บน Internet หรือ Intranet นั้นควรจะให้ระดับปัจจัยที่มีไว้สำหรับปรับค่าเป็น 5

3. **Performance** จะมองในเรื่องของ Response Time ประสิทธิภาพของแอพพลิเคชันนั้นจะได้รับการประเมินโดย User ซึ่งเป็นผลสืบเนื่องจาก การออกแบบ พัฒนา การติดตั้งและรองรับแอพพลิเคชันนั้นๆ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.10

ตารางที่ 2.10 แสดงการกำหนดระดับปัจจัยในส่วนของ Performance (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีสภาพของประสิทธิภาพที่ User ต้องการเป็นพิเศษ
1	ประสิทธิภาพและการออกแบบ ตามความต้องการของ User ได้รับการทบทวนแต่ไม่มีความต้องการกระทำอะไรเป็นพิเศษ
2	Response Time และผลลัพธ์ของการทำงานเป็นเรื่องสำคัญและจำเป็นระหว่างช่วงเวลาที่มีการใช้งานสูงๆ การประมวลผลต้องให้เรียบร้อยก่อนวันถัดไป
3	Response Time และผลลัพธ์ของการทำงานเป็นเรื่องที่สำคัญและจำเป็นระหว่างช่วงเวลาที่มีการใช้งานในธุรกิจ ไม่มีการออกแบบสำหรับ CPU ที่ต้องการใช้งานเฉพาะมีการประมวลผลที่มีความต้องการเฉพาะ รวมทั้งการออกแบบอินเตอร์เฟสมีการระบุเฉพาะ
4	เพิ่มในส่วนของความต้องการของ User ในเรื่องของประสิทธิภาพตั้งแต่ขั้นตอนของการวิเคราะห์ระบบงาน
5	เพิ่มในส่วนของการนำเครื่องมือในการวิเคราะห์ประสิทธิภาพเข้ามาใช้ในขั้นตอนการออกแบบ พัฒนา และนำไปใช้เพื่อที่จะให้ตรงกับประสิทธิภาพที่ User มี

4. **Heavily Used Configuration** จะมองในเรื่องของการใช้ที่เกี่ยวกับฮาร์ดแวร์และ Platform ซึ่งเป็นคุณลักษณะหนึ่งของแอพพลิเคชัน เช่น User ต้องการรันแอพพลิเคชันบนอุปกรณ์หรือเครื่องที่มีการใช้งานอย่างหนัก ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.11

ตารางที่ 2.11 แสดงการกำหนดระดับปัจจัยในส่วนของ Heavily Used Configuration (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีความชัดเจนในเรื่องของความเข้มงวดในความสามารถในการจัดการให้สามารถใช้งานได้
1	มีความเข้มงวดในการจัดการให้เพียงแค่ใช้งานได้แต่ความเข้มงวดก็ยังน้อยในแอพพลิเคชันซึ่งเป็นตัวอย่าง
2	มีระบบรักษาความปลอดภัยหรือการคำนึงถึงข้อจำกัดด้านเวลารวมอยู่ด้วย
3	มีการเจาะจงถึง Processor (เช่น Centrino P4) สำหรับแอพพลิเคชัน
4	มีการระบุถึงความเข้มงวดอย่างยิ่งในการประมวลผลแอพพลิเคชันต่อหน่วยประมวลผลกลาง (Central Processor) ซึ่งมีลักษณะที่เฉพาะเจาะจง
5	มีข้อจำกัดพิเศษบนแอพพลิเคชันในส่วนประกอบแบบ Distributed ของระบบ

5. **Transaction Rate** จะมองในเรื่องความถี่ในการประมวลผล Transaction ว่าเป็นแบบรายวัน รายสัปดาห์ รายเดือน ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.12

ตารางที่ 2.12 แสดงการกำหนดระดับปัจจัยในส่วนของ Transaction Rate (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีการคาดถึงช่วงที่มีระดับสูงสุดของ Transaction
1	ช่วงที่ระดับของจำนวน Transaction เพิ่มสูงสุดนั้นได้ถูกคาดคะเน (รายเดือน รายปักษ์ รายปี)
2	มีการคาดคะเนว่าจำนวน Transaction สูงสุดเป็นรายสัปดาห์
3	มีการคาดคะเนว่าจำนวน Transaction สูงสุดเป็นรายวัน
4	มีอัตราของ Transaction สูงมากซึ่งได้รับการบอกกล่าวจาก User ในแอพพลิเคชันนั้นๆ Requirement ต้องการมากเพียงพอเพื่อที่จะมีการวิเคราะห์ประสิทธิภาพของงานในขั้นตอนการออกแบบ
5	มีอัตราของ Transaction สูงมาซึ่งได้รับการบอกกล่าวจาก User ในแอพพลิเคชันนั้นๆ Requirement ต้องมากเพียงพอเพื่อที่จะมีการวิเคราะห์ประสิทธิภาพของงานในขั้นตอนการออกแบบ รวมไปถึงขั้นตอนของการพัมนา และการติดตั้ง

6. **On-Line Data Entry** จะมองในเรื่องของเปอร์เซ็นต์ของข้อมูลที่ผ่านมาทางออนไลน์ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.13

ตารางที่ 2.13 แสดงการกำหนดระดับปัจจัยในส่วนของ On-Line Data Entry (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ทุกๆ Transaction จะเป็นรูปแบบของ Batch
1	1 % ถึง 7% ของ Transaction เป็น interactive Data Entry
2	8 % ถึง 15% ของ Transaction เป็น interactive Data Entry
3	16 % ถึง 23% ของ Transaction เป็น interactive Data Entry
4	24 % ถึง 30% ของ Transaction เป็น interactive Data Entry
5	> 30% ของ Transaction เป็น interactive Data Entry

7. **End-User Efficiency** จะมองในเรื่องของผู้ใช้งานระบบว่าสามารถใช้งานระบบได้อย่างมีประสิทธิภาพหรือไม่ ซึ่งลักษณะของปัจจัยที่ส่งผลต่อ User Friendly การให้ความสำคัญกับประสิทธิภาพประกอบด้วย ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2_14

- Navigation Aids
- Menu
- Online Help and Document
- Automated Cursor Movement
- Scrolling
- Remote Printing (Via Online Transaction)
- Reassigned Function Keys
- Batched Job Submitted From Online Transactions
- Cursor Selection Of Screen Data
- Heavy Use Of Reverse Video Highlighting Color Underling

And Other Indicators

- Hard Copy User Documentation Of Online Transactions
- Mouse Interface

- Pop Up Windows
- As Few Screen As Possible To Accomplish A Business Function
- Bilingual Support
- Multilingual Support

ตารางที่ 2.14 แสดงการกำหนดระดับปัจจัยในส่วนของ End-User Efficiency (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีการให้ความสำคัญกับประสิทธิภาพตามข้อกำหนดข้างต้น
1	มีการให้ความสำคัญกับประสิทธิภาพ 1-3 ข้อจากข้อกำหนดข้างต้น
2	มีการให้ความสำคัญกับประสิทธิภาพ 4-5 ข้อจากข้อกำหนดข้างต้น
3	มากกว่า 6 ข้อจากข้อกำหนดข้างต้นโดย User ไม่ได้มีความต้องการที่เฉพาะเจาะจงในเรื่องของ Efficiency
4	มากกว่า 6 ข้อจากข้อกำหนดข้างต้นโดย User มีการให้ความสนใจกับเรื่องของ Efficiency โดยต้องมีการออกแบบแอพพลิเคชันอย่างเฉพาะเพื่อสนับสนุน Efficiency
5	มากกว่า 6 ข้อจากข้อกำหนดข้างต้นโดย User มีการให้ความสนใจกับเรื่องของ Efficiency โดยต้องมีการใช้งานอุปกรณ์และขั้นตอนที่พิเศษซึ่งจะทำให้การทำงานนั้นบรรลุตามวัตถุประสงค์

8. **On-Line Update** จะมองในเรื่องของ Internal Logical File ว่ามีจำนวนมากน้อยเพียงใด ที่ได้รับการอัพเดทผ่าน Online Transaction ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.15

ตารางที่ 2.15 แสดงการกำหนดระดับปัจจัยในส่วนของ On-Line Update (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีการอัพเดทออนไลน์
1	การอัพเดทเป็นการอัพเดท 1-3 ไฟล์ โดยการอัพเดทนั้นเป็นไปได้ช้า
2	การอัพเดทเป็นการอัพเดท 4 ไฟล์ขึ้นไป โดยการอัพเดทนั้นเป็นไปได้ช้า
3	สามารถออนไลน์อัพเดทได้ในส่วนหลักๆ ของInternal Logical Files
4	เพิ่มเติมการป้องกันข้อมูลสูญหายในขณะทำ Online Update ซึ่งได้มีการออกแบบและสร้างอย่างพิเศษในซอฟต์แวร์
5	เพิ่มเติมการป้องกันข้อมูลสูญหายของข้อมูลรวมทั้งมีการจัดการระบบกู้ข้อมูลอัตโนมัติ

9. **Complex Processing** เป็นส่วนของคุณลักษณะของแอพพลิเคชัน โดยส่วนประกอบมีดังนี้ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2_16

- มีส่วนขยายของ logical Processing
- มีส่วนขยายของ Mathematical Processing
- มีการประมวลผลที่ซับซ้อนเพื่อจัดการกับ Input และ Output เช่น การมี Multimedia
- ข้อผิดพลาดที่เกิดจากการประมวลผลก่อให้เกิด Transaction ที่ไม่สมบูรณ์นั้นต้องทำการประมวลผลอีกครั้ง
- Sensitive Control เช่นความปลอดภัยในการประมวลผลข้อมูล

ตารางที่ 2.16 แสดงการกำหนดระดับปัจจัยในส่วนของ Complex Processing (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีคุณลักษณะของแอพพลิเคชันตามลักษณะที่กล่าวไว้ด้านบน
1	มีคุณลักษณะของแอพพลิเคชันตามลักษณะที่กล่าวไว้ด้านบน 1 ข้อ
2	มีคุณลักษณะของแอพพลิเคชันตามลักษณะที่กล่าวไว้ด้านบน 2 ข้อ
3	มีคุณลักษณะของแอพพลิเคชันตามลักษณะที่กล่าวไว้ด้านบน 3 ข้อ
4	มีคุณลักษณะของแอพพลิเคชันตามลักษณะที่กล่าวไว้ด้านบน 4 ข้อ
5	มีคุณลักษณะของแอพพลิเคชันตามลักษณะที่กล่าวไว้ด้านบนครบทั้ง 5 ข้อ

10. Reusability จะมองในเรื่องของแอพพลิเคชันที่ได้ทำการพัฒนานั้นสามารถนำไปปรับใช้หรือสามารถนำไปพัฒนาต่อในโปรเจคอื่นๆได้หรือไม่ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.17

ตารางที่ 2.17 แสดงการกำหนดระดับปัจจัยในส่วนของ Reusability (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีการ Reuse โค้ด
1	การ Reuse มีเพียงภายในแอพพลิเคชัน
2	มีการนำโค้ดไป Reuse น้อยกว่า 10% ในหลายๆ งาน
3	มีการนำโค้ดไป Reuse มากกว่า 10% ในหลายๆ งาน
4	ทั้งแอพพลิเคชันและเอกสารประกอบการพัฒนาค่อนข้างเอื้อต่อการนำไป Reuse
5	ทั้งแอพพลิเคชันและเอกสารประกอบการพัฒนาเอื้อต่อการนำไป Reuse

11. Installation Ease จะมองในเรื่องของการติดตั้งซอฟต์แวร์ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.18

ตารางที่ 2.18 แสดงการกำหนดระดับปัจจัยในส่วนของ Installation Ease (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีลักษณะพิเศษในการติดตั้ง
1	มีลักษณะพิเศษในการติดตั้ง แต่ไม่มีความต้องการเพิ่มเติมจาก User
2	การติดตั้งมีการกำหนดจาก User มีคู่มือในการติดตั้งและปรับเปลี่ยนโดยคู่มือที่จัดทำนั้นต้องผ่านการทดสอบ และจะไม่คำนึงถึงผลกระทบจากการติดตั้ง
3	การติดตั้งมีการกำหนดจาก User มีคู่มือในการติดตั้งและปรับเปลี่ยนโดยคู่มือที่จัดทำนั้นต้องผ่านการทดสอบ และคำนึงถึงผลกระทบจากการติดตั้ง
4	เพิ่มจากระดับของปัจจัยที่ 2 โดยมีระบบติดตั้งและเปลี่ยนแปลงอัตโนมัติ
5	เพิ่มจากระดับของปัจจัยที่ 3 โดยมีระบบติดตั้งและเปลี่ยนแปลงอัตโนมัติ

12. Operation Ease จะมองในเรื่องของประสิทธิภาพหรือระบบอัติโนมัติในการ Start Up Back up และขั้นตอนการ Recovery และมีการทดสอบอย่างมีประสิทธิภาพ โดยต้องมีคู่มือประกอบแอพพลิเคชันที่ได้จัดทำขึ้น (Manual) ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.19

ตารางที่ 2.19 แสดงการกำหนดระดับปัจจัยในส่วนของ Operation Base (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีลักษณะพิเศษในการจัดทำ Start Up, Back Up และ Recovery
1-4	มีลักษณะพิเศษที่มีประสิทธิภาพในการจัดทำ Start Up, Back Up และ Recovery
5	ไม่ต้องมีผู้จัดการแอพพลิเคชันโดยแอพพลิเคชันนั้นสามารถจัดการเองได้ รวมทั้งสามารถย้อนเพื่อแก้ไขข้อผิดพลาด

13. Multiple Sites จะมองในเรื่องของแอพพลิเคชันว่าสามารถพัฒนา และสนับสนุน การติดตั้ง ใช้งานได้หลายพื้นที่ หรือหลายองค์กร ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.20

ตารางที่ 2.20 แสดงการกำหนดระดับปัจจัยในส่วนของ Multiple Sites (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	User ไม่มีความต้องการพิเศษในการทำ Multiple Site
1	ความจำเป็นที่จะต้องมี Multiple Sites หรือไม่นั้นต้องมีการพิจารณา ซึ่งถ้าเป็นแบบ Multiple Sites แล้วนั้น แอพพลิเคชันนั้นจะถูกพัฒนาภายใต้ลักษณะของ ฮาร์ดแวร์และซอฟต์แวร์เดียวกัน
2	ความจำเป็นที่จะต้องมี Multiple Sites หรือไม่นั้นต้องมีการพิจารณา ซึ่งถ้าเป็นแบบ Multiple Sites แล้วนั้น แอพพลิเคชันนั้นจะถูกพัฒนาภายใต้ลักษณะของ ฮาร์ดแวร์และซอฟต์แวร์ที่เหมือนกัน
3	ความจำเป็นที่จะต้องมี Multiple Site หรือไม่นั้นต้องมีการพิจารณา ซึ่งถ้าเป็นแบบ Multiple Site แล้วนั้น แอพพลิเคชันนั้นจะถูกพัฒนาภายใต้ลักษณะของ ฮาร์ดแวร์และซอฟต์แวร์ที่แตกต่างกัน
4	เอกสารประกอบได้มีการจัดการทดสอบและสนันสนุนแอพพลิเคชันแบบ Multiple Sites ภายใต้ระดับของปัจจัยที่ 1 หรือ 2
5	เอกสารประกอบได้มีการจัดการทดสอบและสนันสนุนแอพพลิเคชันแบบ Multiple Sites ภายใต้ระดับของปัจจัยที่ 3

14. Facilitate change จะมองในเรื่องของแอพพลิเคชันว่าสามารถออกแบบและพัฒนา เพื่อสามารถเปลี่ยนแปลงซอฟต์แวร์ได้ตามความสะดวกหรือไม่ ซึ่งมีการกำหนดระดับของปัจจัยดังตารางที่ 2.21

คุณลักษณะของ Facilitate

- การ Query ข้อมูลที่ยืดหยุ่นและสิ่งที่ช่วยออกแบบรายงานให้ง่ายขึ้น โดยสามารถดึงข้อมูลอย่างง่ายๆ ที่ต้องการได้ เช่น ตรรกแบบ and/or ซึ่งใช้งานกับ 1 Internal Logical File

- การ Query ข้อมูลที่ยืดหยุ่นและสิ่งที่ช่วยออกแบบรายงานให้ง่ายขึ้น โดยสามารถดึงข้อมูลที่ค่อนข้างซับซ้อยได้ เช่น ที่ต้องการได้ เช่น ตรรกแบบ and/or ซึ่งใช้งานมากกว่า 1 Internal Logical File

- การ Query ข้อมูลที่ยืดหยุ่นและสิ่งที่ช่วยออกแบบรายงานให้ง่ายขึ้น โดยสามารถดึงข้อมูลที่ซับซ้อนได้ เช่น ตรรกแบบ and/or ซึ่งใช้งานมากกว่า 1 Internal Logical File

- ข้อมูลที่ใช้ในการควบคุมทางธุรกิจนั้นจะถูกเก็บไว้ในตารางโดย User จะทำการ Maintain โดยการ Online Interactive Processes การเปลี่ยนแปลงใดๆ ที่เกิดขึ้นนั้นจะกระทบกับวันที่มาทำธุรกิจในวันถัดไปได้

- ข้อมูลที่ใช้ในการควบคุมทางธุรกิจนั้นจะถูกเก็บไว้ในตารางโดย User จะทำการ Maintain โดยการใช้ Online Interactive Processes การเปลี่ยนแปลงใดๆ ที่เกิดขึ้นจะส่งผลทันทีทันใด

ตารางที่ 2.21 แสดงการกำหนดระดับปัจจัยในส่วนของ Multiple Sites (Longstreet, 2004)

ระดับของปัจจัย	คำอธิบาย
0	ไม่มีคุณลักษณะของ Facilitate ตามที่กล่าวมาด้านบน
1	มีคุณลักษณะของ Facilitate ตามที่กล่าวมาด้านบน 1 ข้อ
2	มีคุณลักษณะของ Facilitate ตามที่กล่าวมาด้านบน 2 ข้อ
3	มีคุณลักษณะของ Facilitate ตามที่กล่าวมาด้านบน 3 ข้อ
4	มีคุณลักษณะของ Facilitate ตามที่กล่าวมาด้านบน 4 ข้อ
5	มีคุณลักษณะของ Facilitate ตามที่กล่าวมาด้านบน 5 ข้อ

ขั้นตอนที่ 7 : Calculate Adjusted Function Point Count เป็นขั้นตอนสุดท้ายในการคำนวณหาจำนวนของฟังก์ชันพอยต์โดยใช้สูตรคำนวณเฉพาะ ซึ่งข้อมูลต่างๆ ที่ใช้ประกอบการคำนวณได้จากขั้นตอนการทำงานข้างต้น และสูตรที่ใช้จะเจาะจงสำหรับประเภทของโครงการ ได้แก่ Development Project หรือ Application

สูตรคำนวณค่าของ Function Point ที่มีการปรับค่าแล้ว

$$FP = UAF \times VAF$$

โดย

FP คือ ค่าฟังก์ชันพอยต์ของซอฟต์แวร์ มีหน่วยเป็น Function Point

UAF คือ ผลรวมของจำนวนฟังก์ชันพอยต์ที่ได้จากการนับจากส่วนปนะกอบของซอฟต์แวร์ในขั้นตอนที่ 5

VAF คือ ที่ได้จากการคำนวณในขั้นตอนที่ 6 ซึ่งเป็นค่าที่ประเมินปัจจัย 14 ปัจจัย

และจากความสัมพันธ์ระหว่างจำนวนบรรทัดของโค้ด (SLOC) และฟังก์ชันพอยต์ A.J. Albrecht ได้ทำการวิจัยหาค่าเฉลี่ยโดยประมาณของจำนวนบรรทัดของโค้ดต่อการสร้าง 1 ฟังก์ชันพอยต์สำหรับภาษาคอมพิวเตอร์ต่างๆ ดังนั้นผู้พัฒนาสามารถประมาณจำนวนบรรทัดของโค้ดได้ถ้าทราบจำนวนฟังก์ชันพอยต์ทั้งหมดของซอฟต์แวร์ ดังตารางที่ 2_22

ตารางที่ 2.22 แสดงจำนวนบรรทัดเฉลี่ยของโค้ดต่อ 1 ฟังก์ชันพอยต์สำหรับภาษาต่างๆ (Pressman, 1997)

Programming Language	LOC/FP (Average)
Assembly	320
C	128
COBOL	105
Fortran	105
Pascal	90
C++	64
Ada 95	53
Visual Basic	32
SmallTalk	22
PowerBuilder	16
SQA	12
Object –Oriented Languages	30
Fourth Generation Languages (4GLs)	20
Graphical Languages Icon	4

2.1.2 การประเมินต้นทุนด้วย COCOMO (Constructive Cost Model) (Boehm, 1981)

ในปี 1981 Barry Boehm ได้คิดโมเดลการคำนวณต้นทุนซอฟท์แวร์ (Software Costing Model) เรียกว่า Constructive Cost Model (COCOMO 81) ซึ่งได้รับการยอมรับและนำมาใช้แพร่หลายในองค์กรต่างๆ ของรัฐบาล ในประเทศสหรัฐอเมริกา และได้พัฒนาเป็น Construction Cost Model II (COCOMO II) ในปี 2000 โดยแนวคิดการนำโมเดลมาใช้ในการประเมินต้นทุนซอฟต์แวร์ โดยโมเดลดังกล่าวเกิดจากแนวคิดที่ว่า ในการจัดทำโครงการแต่ละโครงการนั้น มีลักษณะเฉพาะที่แตกต่างกันทั้งในส่วนขององค์กร (Organization) บุคลากร (Person) ลักษณะเฉพาะของโครงการ (Project Characteristics) รวมไปจนกระทั่งลูกค้า หรือบุคคลอื่นๆที่เกี่ยวข้อง (Stakeholders) ดังนั้น COCOMO II ได้ทำการประเมินต้นทุนซอฟต์แวร์ โดยนำลักษณะเฉพาะของโครงการมาทำการ Calibration หรือคำนวณค่า เพื่อประเมินต้นทุน บุคลากร และเวลาที่จะเกิดขึ้น โดยเน้นให้เห็นถึงความสำคัญของ Economic Scale ที่มีผลต่อต้นทุนซอฟต์แวร์ นอกจากนั้นในปัจจุบัน COCOMO II ยังได้นำแนวคิดของกรอบมาตรฐาน CMM (Capability Maturity Model : CMM) มาประยุกต์ใช้ด้วยเช่นกัน (Boehm, 2002)

2.1.2.1 COCOMO 81

การคำนวณจะใช้ข้อมูลนำเข้าเป็นจำนวนบรรทัดของภาษาที่ใช้ในการพัฒนา (Source Line of Code: SLOC) หรือ Function Point และใช้แปลงค่าให้เป็น SLOC จากโมเดลโคโคโมนี้ จะได้ผลการประเมินออกมาเป็น Effort ของโครงการ โดยจะแบ่งรูปแบบของโมเดลเป็นโครงการ 3 แบบได้แก่

1. Organic ลักษณะเป็นโครงการขนาดเล็ก ผู้พัฒนามีความคุ้นเคยอยู่แล้ว และค่อนข้างไม่มีสภาพแวดล้อมที่เป็นข้อจำกัด
2. Semidetached เป็นลักษณะของโครงการขนาดกลาง มีลักษณะที่ไม่เสถียรนัก ผู้พัฒนาไม่มีความคุ้นเคยมาก มีการเปลี่ยนแปลงได้ ค่อนข้างมีสภาพแวดล้อมที่จำกัด
3. Embedded เป็นลักษณะของโครงการขนาดใหญ่ ผู้พัฒนาไม่คุ้นเคยหรือไม่เคยทำมาก่อน มีข้อจำกัดสูงในการพัฒนา

สูตรคำนวณหา Effort ตามประเภทโครงการ

$$Effort = C \times Size^{K}$$

โดย

Size คือ จำนวนบรรทัดของโปรแกรมที่ใช้ในการพัฒนา โดย KLOC = 10^3 (SLOC)

C และ K คือ ค่าคงที่ ที่ได้จากตารางที่ 2.23

ตารางที่ 2.23 แสดงค่าคงที่ C และ K ตามประเภทขององค์กร (Hughes และ Cotterell, 1995)

System Type	C	K
Organic	2.4	1.05
Semi-Detached	3.0	1.12
Embedded	3.6	1.20

2.1.2.2 COCOMO II

เมื่อนำโมเดลออกเผยแพร่ทำการคำนวณค่า Cost Driver ต่างๆ ที่มี ผลต่อโครงการ Barry Boehm ได้ทำการเก็บข้อมูลจากโครงการจำนวน 161 โครงการ และนำค่าที่ได้มาปรับโมเดลแล้ว จึงได้พัฒนาจาก COCOMO 81 เป็น COCOMO II โดยพิจารณาถึงลักษณะเฉพาะภายในโครงการที่แบ่งออกเป็นแต่ละด้านที่มีตัวแปรที่ส่งผลต่อต้นทุนที่เรียกว่า Cost Driver โดยใน COCOMO II ในรูปแบบที่มีความยืดหยุ่นสูงขึ้นและนำไปใช้ในการประเมินได้อย่างมีประสิทธิภาพสูงขึ้น

โดยจุดที่น่าสนใจใน COCOMO II คือ การนำแนวคิดของการคำนวณค่าทางคณิตศาสตร์ (Arithmetic Model) โดยนำค่าทางสถิติเดิมมาใช้ร่วมกับแนวคิดที่ แบ่งส่วนที่สนใจในการพัฒนาซอฟต์แวร์ ตามหลักในการบริหารจัดการนั่นคือ ผลิตภัณฑ์ (Product) กระบวนการ (Process) โครงการ (Project) และบุคลากร (Personal) รวมไปถึงรูปแบบ (Platform) หรืออีกนัยหนึ่งคือเทคโนโลยี (Technology) ที่ใช้ในการดำเนินการ โดย COCOMO II ได้มองว่าในการประเมินต้นทุนของซอฟต์แวร์แต่ละตัวนั้น นอกจากขนาด (Size) ของซอฟต์แวร์แล้วยังมีปัจจัยอื่นๆ ที่ส่งผลกระทบในการประเมินต้นทุนโดยแบ่งออกเป็นด้านต่างๆ ตามที่ได้กล่าวมาแล้ว และทำการคำนวณค่า เพื่อประเมินหาค่าที่เหมาะสม โดยเรียกการทำการคำนวณค่าเพื่อประเมินต้นทุน (Calibration)

สูตรคำนวณหาค่าความพยายาม (Effort)

$$PM = A \times (Size)E \times (EM) + PMauto$$

โดย

PM คือ ค่าความพยายาม (Effort) มีหน่วยเป็น Man-Month

A คือ ค่าคงที่ที่ได้จากการรวบรวมข้อมูลใน 161 โครงการ โดย A = 2.94 (Boehm, 1981)

E คือ Economics of Scale ซึ่งเป็นผลที่ขนาดซอฟต์แวร์ที่มีความสัมพันธ์กับขนาดของโครงการ

Size คือ จำนวนบรรทัดของโปรแกรมที่ใช้ในการพัฒนามีหน่วยเป็น 1,000 บรรทัด

$$E = B + 0.01 \times \sum \text{Scale Factor}$$

โดย

B คือ Scaling Base-Exponent สำหรับหารคำนวณ Effort เท่ากับ 0.91 (Boehm, 1981)

EM คือ ค่า Effort Multipliers เป็นค่าที่ได้จากการคำนวณค่า Cost Driver ที่เกี่ยวข้องในโครงการ โดยค่าดังกล่าวจะส่งผลต่อค่าความพยายาม (Effort) ในการพัฒนาซอฟต์แวร์

PMauto คือ ค่าความพยายาม (Effort) ที่ได้จากการแปลงโค้ดอัติโนมัติ (Automatically Transtated Code) ซึ่งจะเกิดขึ้นเมื่อมีการ Reuse Code โดยค่าดังกล่าวนั้นจะไม่ถือเป็นส่วนของการพัฒนา แต่เนื่องจากมีผลกระทบต่อค่าใช้จ่ายจึงนำมาคิดด้วย ดังนั้นหากเป็นการพัฒนาซอฟต์แวร์ใหม่ทั้งหมดสามารถให้ค่า PMauto เป็นศูนย์

สูตรการคำนวณระยะเวลาที่ใช้ในการพัฒนา Time To Develop : TDEV

$$TDEV = [\ C \times (PM)F\ ] \times \frac{SCED\%}{100}$$

โดย

C คือ Schedule Coefficient ที่นำมาทำการคำนวณ โดย C = 3.67

F คือ Scaling Exponent สำหรับระยะเวลา โดย F = [D + 0.2(E-B)]

D คือ Scaling Base-Exponent สำหรับระยะเวลา โดย D = 0.28

SCED คือ ความรีบเร่งของเวลาเมื่อเทียบกับการพัฒนาปกติ หรือ เปอร์เซ็นต์ของการลดและเพิ่มเวลาที่ใช้พัฒนาซอฟต์แวร์

ดังนั้นจะเห็นได้ว่า Effort ที่ใช้ในการพัฒนานั้นขึ้นอยู่กับขนาดของซอฟต์แวร์ที่จะทำการพัฒนาหรือ Size ของซอฟต์แวร์หลัก แต่ขนาดของซอฟต์แวร์เท่านั้นยังไม่เพียงพอในการคำนวณ เนื่องจากขนาดซอฟต์แวร์ที่เปลี่ยนไปนั้นอาจมีผลต่อการใช้ Effort มากขึ้น หรือน้อยลงแตกต่างกัน ตามผลกระทบจากค่า E

2.2 แนวคิดและทฤษฎีเรื่องเว็บไซต์

2.2.1 ส่วนประกอบของเว็บเพจ มีผู้กำหนดส่วนประกอบไว้ดังนี้

กิตติ ภักดีวัฒนกุล (กิตติ ภักดีวัฒนกุล, 2540) ได้กล่าวถึงส่วนประกอบของโฮมเพจว่ามีส่วนประกอบต่างๆ ที่จำเป็นดังนี้

- **Text** เป็นข้อความปกติ โดยสามารถตกแต่งให้สวยงามและมีลูกเล่นต่างๆ
- **Graphic** ประกอบด้วยรูปภาพ ลายเส้น ลายพื้น ต่างๆ มากมาย
- **Multimedia** ประกอบด้วยรูปภาพ ภาพเคลื่อนไหว และแฟ้มเสียง
- **Counter** ใช้นับจำนวนผู้ที่เข้ามาเยี่ยมชมเว็บเพจ
- **Cool Links** ใช้เชื่อมโยงไปยังเว็บเพจหรือเว็บเพจอื่นๆ
- **Forms** เป็นแบบฟอร์มที่ให้ผู้เข้าเยี่ยมชมกรอกรายละเอียดแล้วส่งกลับมา
- **Frames** เป็นการแบ่งจอภาพเป็นส่วนๆ แต่ละส่วนก็จะแสดงข้อมูลที่แตกต่างกันและเป็นอิสระจากกัน
- **Image Maps** เป็นรูปภาพขนาดใหญ่ที่กำหนดส่วนต่างๆ บนรูป เพื่อเชื่อมโยงไปยังเว็บเพจอื่นๆ
- **Java Applets** เป็นโปรแกรมสำเร็จรูปเล็กๆ ที่ใส่ลงในเว็บเพจ เพื่อให้การใช้งาน เว็บเพจมีประสิทธิภาพมากยิ่งขึ้น
- **สมุดเยี่ยม** (Guestbook) สมุดเยี่ยม ทำหน้าที่คล้ายๆ กับสมุดบันทึกเมื่อมีผู้เข้ามาเยี่ยมและเมื่อผู้ชมได้เขียนคำติ-ชมหรือความคิดเห็นต่างๆ ลงในแบบฟอร์มที่ได้จัดทำได้ โปรแกรมก็จะทำการประมวลผลโดย CGIและแสดงผลที่ผู้เขียนได้บันทึกไว้ออกมาทางเว็บเพจที่เรากำหนดไว้
- **เว็บบอร์ด** (Webboard) เว็บบอร์ด เป็นส่วนประกอบหนึ่งที่ทำให้เว็บกลายเป็นที่นิยม โดยเว็บบอร์ดทำหน้าที่คล้ายๆ กับการให้ผู้เข้าเยี่ยมชมร่วมแสดงความคิดเห็นต่างๆ ตามที่มีการตั้งหัวข้อหรือกระทู้เอาไว้

กิดานันท์ มลิทอง (กิดานันท์ มลิทอง, 2542) ได้จัดแบ่งส่วนประกอบของหน้าเว็บออกเป็น 2 ประเภท คือ ข้อความและภาพ โดยจะมีการจัดโครงสร้างในส่วนย่อยให้มีความแตกต่างกันไป เช่น การจัดพื้นหลัง การให้สี การใช้เสียงประกอบ เป็นต้น

- **ข้อความ ประกอบด้วย**
  - พื้นหลังของข้อความในหน้าเว็บ
  - การเชื่อมโยง ประกอบด้วย

ลักษณะการเชื่อมโยง ซึ่งสามารถแบ่งได้ดังนี้

- การเชื่อมโยงภายใน (Internal Link Local หรือ Page Link) เป็นการเชื่อมโยงกับหน้าเว็บอื่นๆ ภายในเว็บไซต์เดียวกัน การเชื่อมโยงแบบนี้จะใช้ชื่อโดเมน (Domain name) เดียวกัน
- การเชื่อมโยงภายนอก (External link หรือ Remote link) เป็นการเชื่อมโยงกับหน้าเว็บไซต์ในเว็บไซต์อื่นบนเวิลด์ไวด์เว็บ (www) การเชื่อมโยงเหล่านี้จะใช้โดเมนที่แตกต่างออกไป
- การเชื่อมโยงอีเมล์ (E-mail Link) การเชื่อมโยงนี้จะไม่เป็นการนำไปสู่หน้าเว็บอื่นๆ แต่เป็นการเปิดแบบฟอร์มของจดหมายอิเล็กทรอนิกส์แทนเพื่อให้พิพม์ข้อความส่งไป
- การเกี่ยวโยง (Anchors) เป็นการโยงไปสู่ส่วนอื่นๆ ในหน้าเดียวกันโดยไม่เชื่อมโยงไปยังหน้าเว็บอื่นๆ การใช้ลักษณะนี้เป็นประโยชน์มากสำหรับหน้าเว็บที่มีเนื้อหายาวมากๆ

การออกแบบการเชื่อมโยง ซึ่งสามารถแบ่งได้ดังนี้

- การเชื่อมโยงหรือลิงก์ (Link) ที่ใช้อยู่บนเทคโนโลยีเวิลด์ไวด์เว็บ เป็นหัวใจหลักในการทำงานของระบบโดยรวมทั้งหมด การออกแบบลิงก์ต่างๆ จะมีส่วนสัมพันธ์โดยตรงกับเรื่องของเนวิเกชันบาร์ด้วย สำหรับรูปแบบของลิงก์นั้น สามารถแบ่งออกได้เป็น 2 แบบ คือ
- ลิงก์ที่เป็นตัวอักษร (Text Link) หมายถึง ลักษณะของลิงก์ที่ใช้ตัวอักษรในการสร้างไฮเปอร์ลิงก์ขึ้น เพื่อการเชื่อมโยงไปยังที่ต่างๆ โดยมากจะอยู่ในข้อความที่เป็นหัวข้อหรือไม่ก็เป็นข้อความสำคัญต่างๆ การใช้ตัวอักษรเป็นลิงก์สามารถทำได้ง่ายเนื่องจากไม่จำเป็นจะต้องมีการตกแต่งข้อความให้เป็นรูปต่างๆ และสามารถทำให้เอกสารมีขนาดเล็กลงเป็นผลทำให้ระยะเวลาในการดาวน์โหลดเอกสารมาแสดงบนเว็บเบราเซอร์เป็นไปได้อย่างรวดเร็ว
- ลิงก์กราฟิก (Graphic Link) หมายถึง การนำรูปภาพต่างๆ

มาทำเป็นไฮเปอร์ลิงก์ในการเชื่อมโยงไปยังเอกสารและเว็บไซต์ต่างๆซึ่งคุณสมบัติของลิงก์จะเหมือนกับ ลิงก์ตัวอักษรทุกประการ ข้อดีของลิงก์กราฟิก คือเรื่องของความสวยงามที่สามารถออกแบบให้สวยงามมากกว่าแบบอักษร เนื่องจากสามารถนำภาพต่างๆที่สร้างขึ้นมาเป็นลิงก์ได้ทันที แต่ข้อด้อยสำหรับลิงก์แบบนี้ก็คือ เรื่องขนาดของไฟล์กราฟิก และเอกสารจะมีขนาดใหญ่กว่าแบบอักษรหลายเท่าตัว ซึ่งส่งผลให้ระยะเวลาการดาวน์โหลดเอกสารเพื่อให้แสดงบนเว็บเบราเซอร์ต้องใช้ระยะเวลานาน รูปแบบของลิงก์แบบกราฟิกยังแสดงได้ หลายรูปแบบเช่น

- ลิงก์แบบภาพเดี่ยว หมายถึง ลักษณะของการสร้างลิงก์จากภาพต่างๆ โดยที่ภาพหนึ่งจะเชื่อมไปยังเอกสาร หรือเว็บไซต์ เพียงแห่งเดียวเท่านั้น ลักษณะแบบนี้อาจพบได้ทั่วไป เช่น แบรนเนอร์สำหรับโฆษณา เนวิเกชันบาร์ เป็นต้น
- ลิงก์แบบภาพรวม หมายถึง ลักษณะของการสร้างลิงก์จากภาพต่างๆ โดยที่ภาพหนึ่งภาพจะเชื่อมไปยังเอกสาร หรือเว็บไซต์มากกว่าหนึ่งแห่ง ซึ่งโดยมากมักจะใช้ภาพขนาดใหญ่ๆ มาแบ่งเป็นส่วนต่างๆที่เรียกว่า อิมเมจแม็พ (Image Map) หรือฮอตสปอตร์ (Hot Spot) ซึ่งจะทำให้สามารถกำหนดขอบเขตของแต่ละ Link ได้ว่าเราต้องการจะให้มีขนาดและขอบเขตอย่างไร และจะให้เชื่อมไปยังเอกสารหรือเว็บไซต์ใดบ้าง ปัจจุบันการใช้วิธีการแบบอิมเมจแม็พนั้น มีจำนวนของเว็บไซต์ที่ใช้อยู่ไม่มากนัก เนื่องจากส่วนใหญ่ จะหันมาใช้ Macromedia Flash แทน เนื่องจากการทำงานของอิมเมจแม็พไม่มีลักษณะการทำงานแบบ Dynamic ที่สามารถตอบสนองผู้ชมได้ดีดังเช่นที่ Macromedia Flash มีอยู่

➢ ตาราง (Table) โดยทั่วไปข้อความในเว็บอาจจะมีการจัดอยู่ในลักษณะของคอลัมน์เดียว ปกติแล้วผู้ใช้จะไม่ทราบว่าข้อความนั้นจัดอยู่ในตารางทั้งนี้เนื่องจากนักออกแบบได้ซ่อนเส้นตารางไว้เนื่องจากเส้นตารางจะทำให้บนหน้าเว็บรกและไม่สวยงาม

➢ กรอบ (Frame) กรอบจะมีลักษณะแตกต่างจากตารางแต่มองจะมีลักษณะที่คล้ายกันเนื่องจากมีลักษณะเป็นมีคอลัมน์เหมือนกัน การจะดูว่าส่วนใดเป็นกรอบให้สังเกตจากแถบเลื่อน (Scroll Bar) ที่อยู่ด้านข้างหรือด้านล่าง แต่บางครั้งอาจไม่มีแถบเลื่อนก็ได้ ในหน้าเว็บหนึ่งหน้าอาจจะมีตั้งแต่ 1-4 กรอบหรือมากกว่านั้นแล้วแต่การออกแบบ

➢ แบบฟอร์ม (form) ลักษณะพิเศษอย่างหนึ่งของสื่อในระบบเชื่อมตรง คือการให้ผู้ใช้งานสามารถส่งข้อมูลป้อนกลับไปยังเว็บไซต์นั้นได้ทันที ซึ่งนอกจากจะเป็นในลักษณะอีเมล์แล้วยังมีลักษณะของการกรอกแบบฟอร์มในช่องข้อความ การใส่รหัสผ่าน รวมถึงการคลิกปุ่มเลือกตอบ ปุ่มส่ง หรือปุ่มจัดใหม่ และการเลือกตัวเลือกในเมนูที่มีทั้งแบบดึงลงและเลื่อนหาข้อความได้ด้วย

- **ภาพกราฟิก**

ภาพกราฟิกที่ใช้จะอยู่ในรูปแบบของ GIF หรือ JPEG ซึ่งความแตกต่างกันจะอยู่ในเรื่องการบีบอัดภาพ สี การแสดงภาพนิ่งหรือภาพเคลื่อนไหว

2.3 วรรณกรรมที่เกี่ยวข้อง

งานวิจัยของ David Longstreet (Longstreet, 2004) แสดงให้ว่าแต่ละลักษณะของ GUI นั้นเป็น Data Element Type ที่อยู่บน Internal Logical File โดยกำหนดการนับจำนวนของ Data Element Type ได้ดังนี้

- Display of Graphical Image or Icon การที่ใช้ไอคอนรูปต่างๆนั้นก็เพื่อที่จะทำให้ผู้ใช้งานได้เข้าใจง่ายขึ้น ตัวอย่างการนับ Data Element Type เช่น ในแอพลิเคชั่น ในเรื่องการค้นหาความหมายของคำศัพท์ เมนูจะประกอบไปด้วย 7 ไอคอน คือ การสั่งพิมพ์ พจนานุกรมส่วนบุคคลจากนายกราชบัณฑิตยสถาน แนะนำบัณฑิตยสถาน สิ่งพิมพ์ของราชบัณฑิตยสถาน วิธีใช้ และเกี่ยวกับโปรแกรม โดยจะนับเป็น 1 Data Element Type ดังรูปที่ 2.4

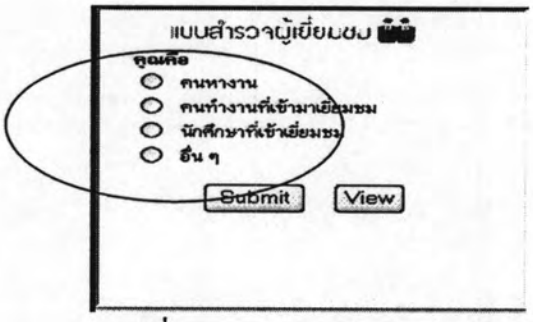
ตัวอย่าง

รูปที่ 2.4 แสดงภาพเมนูในรูปแบบไอคอน

จากรูปที่ 2.4 จะนับแต่ละไอคอนเป็น 1 DET ดังนั้นจากรูปมี 7 DET

- Radio Button นั้นเป็นส่วนหนึ่งของ Data Element Type ซึ่ง Radio Button สามารถกดเพื่อเลือกได้เพียงตัวเลือกเดียวเท่านั้น ดังนั้นก็จะนับเป็นแค่ 1 Data Element Type ภายใต้เงื่อนไขหรือกรอบหนึ่งๆ

ตัวอย่าง



รูปที่ 2.5 แสดงภาพ Radio Button

จากรูปที่ 2.5 จะมี Radio Button ทั้งหมด 4 Radio Button แต่ทั้งหมดจะนับเป็น 1 DET เท่านั้น เพราะเป็นทางเลือกแบบเลือกอย่างใดอย่างหนึ่งเพียงรายการเดียว

- Check boxes จะต่างจาก Radio Button ตรงที่ว่า Check boxes นั้นสามารถเลือกได้หลายตัวเลือกในครั้งเดียว ภายใต้เงื่อนไขหรือกรอบหนึ่งๆ ซึ่งจะนับเป็น 1 Data Element Type ทุกๆ Check boxes

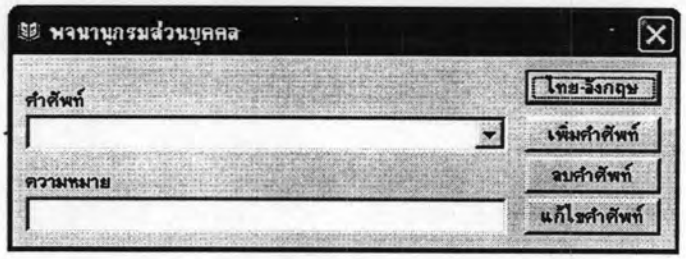
ตัวอย่าง

รูปที่ 2.6 แสดงกล่องเลือก

จากรูปที่ 2.6 Check Boxes จะนับเฉพาะ DET เฉพาะ Check Boxes ที่ถูกเลือกเท่านั้น จากรูปมี Check Boxes ที่ถูกเลือก 2 รายการ ดังนั้น DET เท่ากับ 2

- Command button นั้นใช้สำหรับระบุถึงการกระทำบางอย่างเช่น เพิ่ม เปลี่ยนแปลง ลบ หรือดึงข้อมูลขึ้นมาแสดง เช่นปุ่ม OK นั้นสามารถสร้าง Transaction ได้หลายรูปแบบ

ตัวอย่าง



รูปที่ 2.7 แสดง Command button

จากรูปที่ 2.7 เมื่อมีการเพิ่มข้อมูลโดยปุ่ม "เพิ่มคำศัพท์" จะแสดงถึง 1 External Input เมื่อกดปุ่ม "แก้ไขคำศัพท์" จะแสดงถึง 1 External Input เมื่อกดปุ่ม "ลบคำศัพท์"จะแสดงถึง 1 External Input

- Sound bytes โดยทั่วไปแล้วนั้นแอพลิเคชั่นแบบ GUI นั้นจะมีเสียงต่างๆแนบมาด้วย โดยเสียงที่แนบมาเหล่านี้จะนับเป็น 1 Data Element Type โดยทั่วไปแล้วนั้นเสียงที่แนบมานั้นจะมีลักษณะของการวนใช้ข้อมูลหรือ (Recursive Information) โดยถ้าเป็นลักษณะของข้อมูลแบบวนใช้ แม้ว่าเสียงที่ใช้นั้นจะเพิ่มในรูปแบบใดๆก็ตามก็ยังนับเพียงแค่ 1 Data Element Type โดยคำนึงว่ายิ่งเล่นนานเท่าไหร่ลักษณะการวนซ้ำยิ่งมากขึ้นเท่านั้น
- Photographic Images นั้นจะนับเป็น 1 Data Element Type ตัวอย่างเช่น โปรแกรมของแผนกบุคลากรนั้นจะแสดงข้อมูลพนักงาน รวมทั้งจะมีรูปภาพของพนักงานด้วย ตำแหน่งที่แสดงรูปของพนักงานนั้น จะนับเป็น 1 Data Element Type
- Message ข้อความที่เราจะพิจารณาในแอพลิเคชั่น GUI นั้นมี 3 แบบ
    1. Error Massage
    2. Confirmation Massage
    3. Notification Massage

โดย Error Massage และ Confirmation Massage นั้นจะระบุถึงความผิดพลาดที่เกิดขึ้นหรือเตือนถึงขั้นตอนทำงานที่เสร็จเรียบร้อยแล้ว

ตัวอย่าง

รูปที่ 2.8 แสดง Error Massage

จากรูปที่ 2.8 Error Massage จะนับเป็น 1 Data Element Type

2.4 สถิติที่ใช้ในงานวิจัย

2.4.1 การคำนวณสถิติเบื้องต้นในเชิงสถิติพรรณา (Descriptive Statistics)

การแสดงค่าสถิติพรรณาแยกตามชนิดของตัวแปร

2.4.1.1 ข้อมูลเชิงกลุ่ม (Categorical Data) โดยข้อมูลอาจจะเป็นข้อมูลสเกลนามกำหนด หรือสเกลอันดับ ซึ่งสามารถแสดงตารางการแจกแจงความถี่ของแต่ละค่าข้อมูลได้ เช่น ความถี่แยกตามเพศ

ตารางที่ 2.24 แสดงข้อมูลเพศชายและเพศหญิง

เพศ	จำนวน (ความถี่)	ร้อยละ
ชาย	80	40
หญิง	120	60
รวม	200	100

2.4.1.2 ข้อมูลเชิงปริมาณ (Quantitative Data) เป็นข้อมูลชนิดตัวเลขที่มีค่าจริง นั่นคือ เป็นสเกลแบบช่วงและสเกลอัตราส่วน เช่น รายได้ น้ำหนัก ความสูง เป็นต้น โดยสามารถแสดงค่าที่เป็นค่ากลางของข้อมูลได้ เช่น ค่าเฉลี่ย ค่ามัธยฐาน ค่าฐานนิยม เป็นต้น

2.4.2 สถิติที่ใช้วัดค่าเฉลี่ยของข้อมูลเชิงปริมาณ (Mean)

ค่าเฉลี่ยตัวอย่าง : $\overline{X} = \frac{\sum_{i=1}^{n} x_i}{n}$; n = จำนวนข้อมูลตัวอย่าง

2.4.3 การตรวจสอบการแจกแจงข้อมูล

ในการทำวิจัยข้อมูลที่ได้มาส่วนใหญ่เป็นข้อมูลตัวอย่าง จะต้องคำนวณค่าสถิติ หรือใช้เทคนิคการวิเคราะห์ทางสถิติ เพื่อสรุปลักษณะของประชากร จึงจำเป็นต้องมีการตรวจสอบการแจกแจง หรือลักษณะของข้อมูลตัวอย่างเพื่อที่จะได้อ้างอิงถึงลักษณะของประชากรต่อไป

2.4.3.1 วิธีการตรวจสอบการแจกแจงข้อมูลเชิงปริมาณโดยใช้สถิติทดสอบ

- **Kolmogorov-Smirnov Test (K-S Test)**

เป็นสถิติทดสอบที่ใช้ทดสอบการแจกแจงของประชากรว่าเป็นแบบปกติหรือไม่ หลักการของการทดสอบนี้คือ การเปรียบเทียบค่าฟังก์ชันการแจกแจงสะสมของข้อมูลตัวอย่างกับค่าฟังก์ชันการแจกแจงสะสมของข้อมูลภายใต้สมมติฐานว่าประชากร/ข้อมูลการแจกแจงปกติ ถ้าความแตกต่างต่ำแสดงว่าการแจกแจงเป็นปกติ

สมมติฐานของการทดสอบคือ

H_0 : สุ่มตัวอย่างจากประชากรที่มีการแจกแจงแบบปกติ

H_1 : สุ่มตัวอย่างจากประชากรที่ไม่ได้มีการแจกแจงแบบปกติ

สถิติทดสอบ : Kolmogorov- Smirnov

เขตปฏิเสธ H_0 : จะปฏิเสธ H_0 ถ้าค่า p-value หรือ Sig.(Significance) น้อยกว่าระดับนับสำคัญที่กำหนด

- **Shapiro-Wilk Test**

เป็นสถิติที่ใช้ทดสอบการแจกแจงของตัวแปรเชิงปริมาณว่ามีการแจกแจงแบบปกติหรือไม่

สมมติฐานของการทดสอบ คือ

H_0 : สุ่มตัวอย่างจากประชากรที่มีการแจกแจงปกติ

H_1 : สุ่มตัวอย่างจากประชากรที่ไม่มีการแจกแจงปกติ

สถิติทดสอบ : Shapiro-Wilk Test

เขตปฏิเสธ H_0 : จะปฏิเสธ H_0 ถ้าค่า p-value หรือ Sig.(Significance) น้อยกว่าระดับนับสำคัญที่กำหนด

2.4.4 การประมาณค่า การทดสอบสมมติฐาน และการหาความสัมพันธ์

การประมาณค่า คือ การประมาณค่าพารามิเตอร์ซึ่งเป็นลักษณะของประชากร โดยใช้ข้อมูลตัวอย่าง หรือ ทำการประมาณค่าพารามิเตอร์ด้วยค่าสถิติ เช่น

- **การประมาณค่าผลต่างระหว่างค่าเฉลี่ย 2 ประชากร (μ_1, μ_2)**

เมื่อต้องการเปรียบเทียบความแตกต่างระหว่างค่าเฉลี่ย 2 ประชากร เช่น สนใจเปรียบเทียบยอดขายค่าเฉลี่ยรายปีของรถยนต์ยี่ห้อ A และ B หรือเปรียบเทียบอายุการใช้งานเฉลี่ยของตู้เย็น 2 ยี่ห้อ

ค่าประมาณแบบจุดของ $\mu_1 - \mu_2$ คือ $\overline{x}_1 - \overline{x}_2$

โดยที่ให้ x_{ij} = ข้อมูลตัวอย่างที่ j ที่สุ่มจากประชากรที่ i ; i = 1,2,...,n_i

$\overline{x}_i$ = ค่าเฉลี่ยตัวอย่างของประชากรที่ i

n_i = ขนาดตัวอย่างที่สุ่มจากประชากรที่ i

S.E. ($\overline{x}_1 - \overline{x}_2$) = ค่าคลาดเคลื่อนมาตรฐานของ $\overline{x}_1 - \overline{x}_2$

s_i^2 = ค่าแปรปรวนตัวอย่างของตัวอย่างชุดที่ i

2.4.4.1 การประมาณผลต่างระหว่างค่าเฉลี่ย 2 ประชากรเมื่อสุ่มตัวอย่างจากประชากรทั้ง 2 แบบเป็นอิสระกัน

ค่าประมาณแบบช่วงของ $\mu_1 - \mu_2$ ที่ระดับความเชื่อมั่น $(1-\alpha)100\%$ คือ

- **เมื่อขนาดตัวอย่างใหญ่**

$$(\bar{x}_1 - \bar{x}_2) - z_{1-\frac{\alpha}{2}} \text{ S.E.}(\bar{x}_1 - \bar{x}_2) < \mu_1 - \mu_2 < (\bar{x}_1 - \bar{x}_2) + z_{1-\frac{\alpha}{2}} \text{ S.E.}(\bar{x}_1 - \bar{x}_2)$$

- **เมื่อขนาดตัวอย่างขนาดเล็ก**

$$(\bar{x}_1 - \bar{x}_2) - t_{1-\frac{\alpha}{2},df} \text{ S.E.}(\bar{x}_1 - \bar{x}_2) < \mu_1 - \mu_2 < (\bar{x}_1 - \bar{x}_2) + t_{1-\frac{\alpha}{2},df} \text{ S.E.}(\bar{x}_1 - \bar{x}_2)$$

โดยที่ df = องศาอิสระ

- **เมื่อขนาดตัวอย่างใหญ่**
  - **เมื่อทราบค่าแปรปรวนประชากร** (σ_1^2, σ_2^2)

$$\text{S.E.}(\bar{x}_1 - \bar{x}_2) = \sqrt{\frac{\sigma_1^2}{n_1}} + \sqrt{\frac{\sigma_2^2}{n_2}}$$

  - **เมื่อไม่ทราบค่าแปรปรวนประชากร** (σ_1^2, σ_2^2)

$$\text{S.E.}(\bar{x}_1 - \bar{x}_2) = \sqrt{\frac{s_1^2}{n_1}} + \sqrt{\frac{s_2^2}{n_2}}$$

- **เมื่อขนาดตัวอย่างเล็กและไม่ทราบค่าแปรปรวนประชากร**
  - **ไม่ทราบค่า** σ_1^2 **และ** σ_2^2 **แต่ทราบว่า** $\sigma_1^2 \neq \sigma_2^2$

ค่าประมาณแบบช่วงของ $\mu_1 - \mu_2$ ที่ระดับความเชื่อมั่น $(1-\alpha)100\%$ คือ

$$(\bar{x}_1 - \bar{x}_2) - t_{1-\frac{\alpha}{2},df} \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}} < \mu_1 - \mu_2 < (\bar{x}_1 - \bar{x}_2) + t_{1-\frac{\alpha}{2},df} \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

โดยที่ $\text{df} = \dfrac{(s_1^2/n_1 + s_2^2/n_2)^2}{\dfrac{(s_1^2+n_1)^2}{n_1-1} + \dfrac{(s_2^2+n_2)^2}{n_2-1}}$

2.4.4.2 การทดสอบสมมติฐานทางสถิติ

- **ขั้นตอนการทดสอบสมมติฐานทางสถิติ**

การทดสอบค่าเฉลี่ยประชากร (μ) จะใช้สถิติทดสอบ $\overline{x}$

- ประชากรมีการแจกแจงแบบปกติ หรือขนาดตัวอย่างใหญ่ ($n \geq 30$) จะปรับ $\overline{x}$ เป็น Z หรือในโปรแกรม SPSS for Window จะใช้สถิติทดสอบ t
- ประชากรมีการแจกแจงแบบปกติ ตัวอย่างมีขนาดเล็ก ($n < 30$) เนื่องจากเมื่อขนาดตัวอย่างใหญ่ Z = t และไม่ทราบค่าแปรปรวน จะปรับ $\overline{x}$ เป็นสถิติทดสอบ t

- **หลักเกณฑ์การปฏิเสธหรือยอมรับสมมติฐาน H_0**

การทดสอบแบบ 2 ด้าน

$H_0 : \mu_1 = \mu_2$

$H_1 : \mu_1 \neq \mu_2$

โดยที่ μ_0 เป็นค่าคงที่

เป็นการทดสอบแบบ 2 ด้านโดยมีค่าวิกฤติ 2 ค่า คือ

1. จะปฏิเสธ H_0 ถ้า $z \leq -z_{1-\frac{\alpha}{2}}$ หรือ $z \geq z_{1-\frac{\alpha}{2}}$
2. หรือจะปฏิเสธ H_0 ถ้า $t \leq -t_{1-\frac{\alpha}{2}}$, หรือ $t \geq t_{1-\frac{\alpha}{2}}$,

การปฏิเสธ H_0 : $\mu_1 = \mu_2$ โดยใช้ผลลัพธ์จาก SPSS พิจารณาจากค่า Sig. (2-tailed) จากผลลัพธ์ของ SPSS โดยจะปฏิเสธ H_0 ถ้าค่า Sig. (2-tailed) น้อยกว่าระดับนัยสำคัญที่กำหนด

2.4.5 การวิเคราะห์ความสัมพันธ์ระหว่างตัวแปรเชิงกลุ่มชนิดสเกลแบ่งกลุ่ม 2 ตัว (Measures of Association for Nominal Data)

ในกรณีที่สนใจทดสอบความสัมพันธ์ของตัวแปรสเกลแบ่งกลุ่ม 2 ตัว เช่น อาชีพกับความคิดเห็นในด้านการเมือง เป็นต้น

- สถิติ Chi-Square ใช้ในการทดสอบความเป็นอิสระกันของตัวแปร 2 ตัวโดย SPSS จะคำนวณค่าสถิติทดสอบ Chi-Square คือ Pearson Chi-Square
  - Pearson Chi-Square เป็นสถิติทดสอบที่ใช้กับ
    1. ตัวแปรเชิงกลุ่ม
    2. ความสัมพันธ์ของตัวแปรทั้ง 2 จะอยู่ในรูปใดก็ได้ ดังนั้นในสมมติฐานแย้งจึงเป็น H_1 : ตัวแปรทั้งสองมีความสัมพันธ์กัน

$$X^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(O_{ij}-E_{ij})^2}{E_{ij}}$$

E_{ij} = ความถี่ที่คาดว่าจะอยู่ใน cell (i,j) ถ้าตัวแปรทั้งสองเป็นอิสระกัน

$$= \frac{r_i c_j}{n}$$

X^2 = **Pearson Chi Square**

เมื่อปฏิเสธสมมติฐาน H_0 จะสรุปได้เพียงว่า ตัวแปรทั้ง 2 มีความสัมพันธ์กัน แต่ไม่ทราบว่าตัวแปรอยู่ในความสัมพันธ์อยู่ในรูปแบบใด และไม่ทราบทิศทางและระดับของความสัมพันธ์

- **การวัดระดับความสัมพันธ์ของตัวแปรสเกลแบ่งกลุ่ม (Nominal) 2 ตัว ว่ามีความสัมพันธ์มากน้อยเพียงใด โดยใช้ Contingency Coefficient**
  - Contingency Coefficient เป็นสถิติที่ใช้วัดความสัมพันธ์ของตัวแปรสเกลแบ่งกลุ่ม 2 ตัว และสามารถวัดระดับความสัมพันธ์ได้แต่ไม่สามารถระบุทิศทางความสัมพันธ์ได้ โดยมีสถิติทดสอบเป็น

$$C = \sqrt{\frac{X^2}{X^2+n}}$$

โดยที่ X^2 = **Pearson Chi Square**

ถ้า C = 0 แสดงว่าตัวแปร 2 ตัว นั้นเป็นอิสระกัน หรือไม่มีความสัมพันธ์กัน

ถ้า C เข้าใกล้ 1 แสดงว่าตัวแปร 2 ตัวดังกล่าวมีความสัมพันธ์กันมาก

2.4.6 การวิเคราะห์ความสัมพันธ์ของตัวแปรสเกลอันดับ 2 ตัว (Measures of Association for Ordinal Data)

2.4.6.1 ขั้นตอนที่ 1 การตรวจสอบความสัมพันธ์

H_0 : ตัวแปรเกลอันดับ 2 ตัวเป็นอิสระกัน

H_1 : ตัวแปรเกลอันดับ 2 ตัวๆไม่เป็นอิสระกัน

สถิติทดสอบ Pearson Chi-Square เมื่อตัวแปรมีการแจกแจงปกติหรือมีหน่วยตัวอย่างขนาดใหญ่

สถิติทดสอบ สัมประสิทธิ์สหสัมพันธ์ของเสปียร์แมน (Spearman' s Coefficient of Rank Correlation : ρs) เมื่อตัวแปรมีการแจกแจงไม่ปกติ หรือมีขนาดจำนวนหน่วยตัวอย่างน้อย จะใช้การทดสอบสมมติฐานที่ไม่ใช้พารามิเตอร์ (Nonparametric Tests)

การสรุปผลการทดสอบ จะสรุปว่า ถ้ายอมรับ H_0 ไม่ต้องทำต่อในขั้นตอนที่ 2 เนื่องจากตัวแปรทั้ง 2 มีความเป็นอิสระกัน แต่ถ้ายอมรับ H_1 จะทำต่อในขั้นตอนที่ 2

2.4.6.2 ขั้นตอนที่ 2 การหาระดับและทิศทางของความสัมพันธ์

การทดสอบสหสัมพันธ์โดยใช้ลำดับที่ของ Spearman เนื่องจากสัมประสิทธิ์สหสัมพันธ์ของเสปียร์แมน (Spearman' s Coefficient of Rank Correlation : ρs) เป็นค่าที่ใช้วัดสหสัมพันธ์ระหว่างตัวแปร 2 ตัว ว่ามีความสัมพันธ์มากน้อยเพียงใด เมื่อตัวแปรที่นำมาวิเคราะห์เป็นข้อมูลลำดับที่ (ranked data) หรือเป็นข้อมูลที่อยู่ในมาตราแสดงลำดับ (nominal scale)

โดยค่า ρs จะมีค่าระหว่าง -1 ถึง 1 และตัวประมาณของ ρs คือ r_s ซึ่งมีสูตรการคำนวณ คือ

$$r_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)}$$

โดยที่ d = ค่าความแตกต่างระหว่างแต่ละคู่

n = จำนวนคู่

การตั้งสมมติฐาน

- $H_0 : \rho s = 0$

  $H_1 : \rho s \neq 0$

- $H_0 : \rho s \leq 0$

  $H_1 : \rho s > 0$

ตัวสถิติสำหรับการทดสอบ คือ

$$r_s = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2-1)}$$

เขตปฏิเสธ H_0 : จะปฏิเสธ H_0 ถ้า $r_s < -r_{s,\alpha/2}$ **หรือ** $r_s > r_{s,\alpha/2}$

ความหมายของของค่าสถิติ Spearman Correlation

1. เครื่องหมายแสดงทิศทางความสัมพันธ์

- ถ้าเป็นเครื่องหมายลบ จะหมายถึง ตัวแปรสเกลอันดับ 2 ตัวมีสัมพันธ์ในทิศทางตรงกันข้าม
- ถ้าเป็นเครื่องหมายบวก หมายถึงตัวแปรสเกลอันดับ 2 ตัวนั้นมีสัมพันธ์ในทิศทางเดียวกัน

2. ค่าสถิติเข้าใกล้ 1 หรือ 0

- ถ้าค่าที่ได้ใกล้ 1 หมายถึง มีความสัมพันธ์กันมาก
- ถ้าค่าที่ได้อยู่ใกล้ศูนย์ หมายถึง ไม่มีความสัมพันธ์กัน หรือมีความสัมพันธ์กันน้อย