บทที่ 8

การทดสอบระบบ

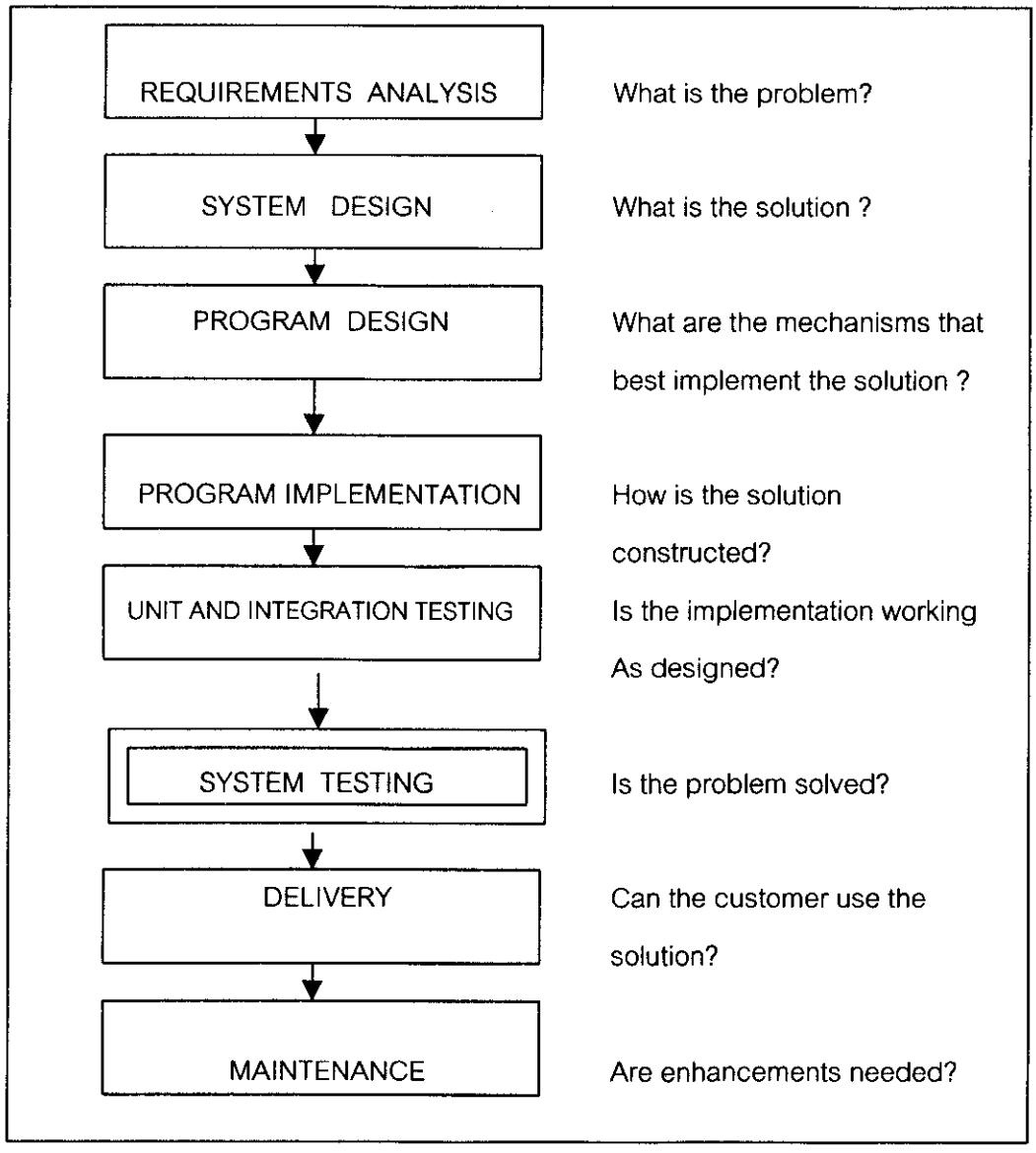
ในบทที่ 3 , 4 , 5 และ 6 ได้กล่าวถึงการตรวจระบบ(review) เพื่อรับประกันว่าซอฟต์แวร์ที่พัฒนานั้นมีคุณภาพ โดยย้ำเตือนถึงความจำเป็นของเอกสารซึ่งสามารถสร้างความเข้าใจถึงความต้องการ ,การออกแบบ และคำสั่งโปรแกรมแก่ผู้พัฒนาระบบได้เป็นอย่างดี สำหรับบทที่ 7 ได้กล่าวถึงการทดสอบเพื่อหาความผิดพลาดในโปรแกรมโมดูลและการปฏิสัมพันธ์ระหว่างโมดูล จากรูปภาพที่ 8.1 แสดงถึงการทดสอบซึ่งสามารถแบ่งได้เป็น 2 ขั้นตอนใหญ่ๆ คือ ขั้นตอนแรกเป็นการทดสอบว่าโปรแกรมที่ถูกสร้างขึ้นโดยโปรแกรมเมอร์เมื่อรวมกันเป็นระบบการทำงานแล้วสามารถกระทำงานได้ตามที่ออกแบบไว้หรือไม่ เป็นการทดสอบในระดับโปรแกรม ซึ่งต้องกำหนดวัตถุประสงค์ของการทดสอบและสร้างแผนการในการรวมโมดูลต่างๆ ส่วนขั้นตอนที่สองเป็นระดับระบบ เป็นการตรวจสอบระบบที่พัฒนาขึ้นนั้นสามารถแก้ปัญหาได้ตรงตามที่ได้กำหนดในเอกสารกำหนดความต้องการได้หรือไม่ แบ่งเป็นขั้นตอนย่อยๆ คือ Function testing , Performance testing , Acceptance testing และ Installation testing

8.1

กฎของการทดสอบระบบ

การทดสอบระบบนั้นแตกต่างจากการทดสอบโปรแกรม ความผิดพลาดสามารถเกิดขึ้นได้ทุกๆ จุดในขั้นตอนการพัมนา ดังรูปภาพที่ 8.2 แสดงถึงสาเหตุของความผิดพลาดที่สามารถเกิดขึ้นในทุกระยะของขั้นตอนการพัฒนาระบบ

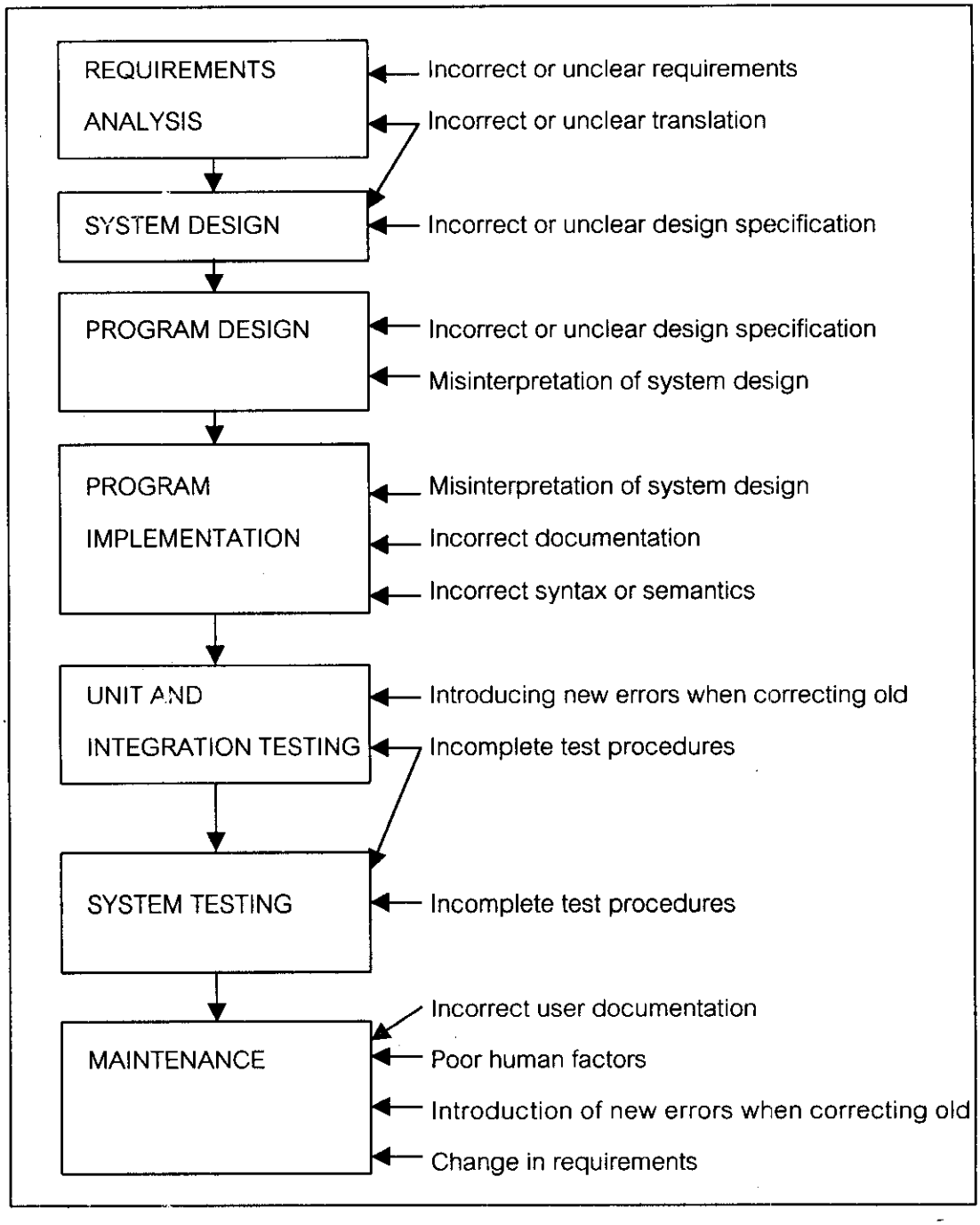
รูปภาพที่ 8.1 The System Development Process



ความผิดพลาดอาจเกิดในระยะแรกของการพัฒนาจากการกำหนดความต้องการซึ่งลูกค้าไม่แน่ใจถึงความต้องการ การแปลความหมายที่ผิดพลาด ซึ่งส่งผลให้ระบบไม่สามารถทำงานตามที่ลูกค้าหวังไว้ การสื่อสารระหว่างทีมงานจากการแปลความต้องการที่ผิดพลาดมีผลให้การออกแบบระบบไม่ถูกต้อง รวมทั้งการออกแบบโปรแกรมที่ผิดพลาดการ การพัฒนาโปรแกรมที่ผิดพลาด เอกสารผิดพลาด โดยความผิดพลาดที่เกิดขึ้นอาจเกิดจากลูกค้า นักออกแบบระบบ

นักออกแบบโปรแกรม โปรแกรมเมอร์ ทีมงานทดสอบ รวมทั้งทีมงานในการบำรุงรักษาระบบ ดังแสดงในรูปภาพที่ 8.2 ซึ่งแสดงถึงความผิดพลาดที่อาจเกิดขึ้นในการพัฒนาระบบ

รูปภาพที่ 8.2 แสดงถึงข้อผิดพลาดที่เกิดขึ้นในการพัฒนาระบบ

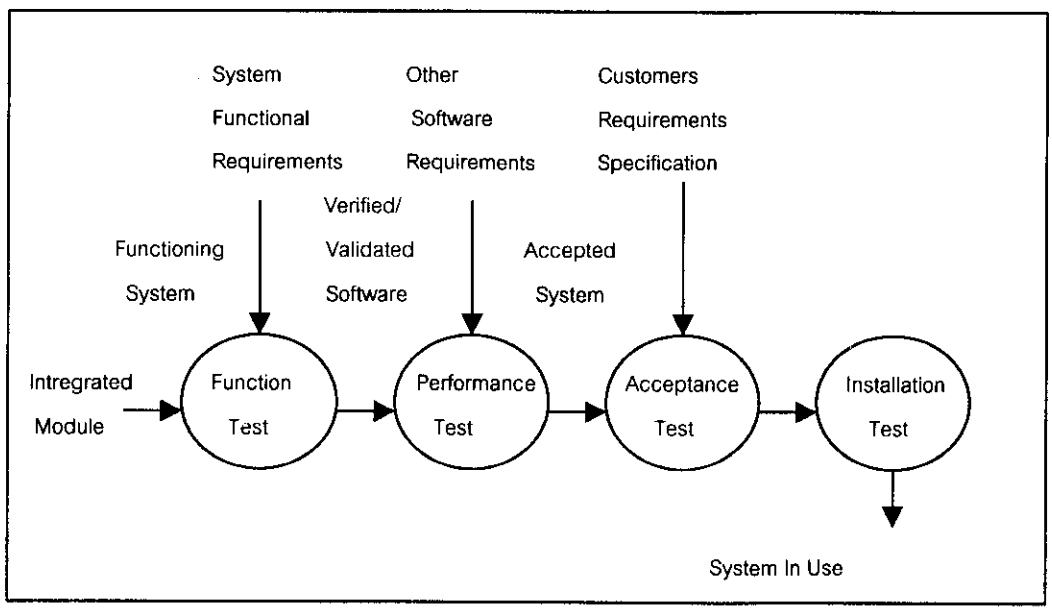


ขั้นตอนในการทดสอบระบบ

ขั้นตอนในการทดสอบระบบแบ่งออกเป็นหลายขั้นตอนคือ

1. Function testing
2. Performance testing
3. Acceptance testing
4. Installation testing

รูปภาพที่ 8.3 แสดงขั้นตอนในการทดสอบระบบ



ขั้นตอนต่างๆแสดงดังรูปภาพที่ 8.3 ซึ่งแต่ละขั้นตอนมีจุดประสงค์ในการทดสอบแตกต่างกัน โดย Function test จะตรวจสอบระบบถึงหน้าที่ต่างๆที่สามารถกระทำได้ว่าตรงกับความต้องการหรือไม่ เช่น ในระบบฝากถอนเงิน การตรวจสอบจะกระทำการตรวจสอบถึงการทำงานที่ระบบสามารถรับข้อมูลเพื่อทำการฝากเงิน ถอนเงิน คำนวณดอกเบี้ย ประมวลผล และปรับยอดโดยพิมพ์ยอดเงินคงเหลือได้อย่างถูกต้อง โดยการตรวจสอบต่างๆเหล่านี้จะอ้างอิงกับเอกสารระบุความต้องการ เพื่อตรวจสอบความต้องการที่ต้องการให้ระบบกระทำได้ หลังจากตรวจสอบความต้องการต่างๆแล้ว ขั้นตอนต่อไปเป็นการตรวจสอบประสิทธิภาพของระบบคือ

Performance test อันได้แก่ความปลอดภัย ความถูกต้อง ความเร็วในการประมวลผล เวลาในการตอบสนองกับผู้ใช้ ความน่าเชื่อถือ หรือข้อกำหนดหรือกฎเกณฑ์ต่างๆ โดยการทดสอบสามารถกระทำในสภาพแวดล้อมและผู้ใช้งานจริง เรียกว่า Validated system หรือทดสอบในสภาพแวดล้อมจำลอง เรียกว่า Verified system ต่อจากนั้นเราจะนำผลลัพธ์ของการทดสอบประสิทธิภาพของระบบไปเสนอให้แก่ลูกค้าเพื่อให้ลูกค้าตรวจสอบว่าตรงกับสิ่งที่ลูกค้าได้กำหนดไว้ในเอกสารกำหนดความต้องการ ซึ่งขั้นตอนนี้เรียกว่า Acceptance test เพื่อตรวจสอบคุณลักษณะของระบบเพื่อให้แน่ใจว่าสามารถกระทำงานได้ตามที่ต้องการ และถ้าระบบยังไม่ติดตั้งในสภาพแวดล้อมของผู้ใช้งาน ขั้นตอนสุดท้ายเป็น Installation test เป็นการติดตั้งระบบเพื่อให้ผู้ใช้งานได้ทดลองใช้งานและตรวจสอบถึงความผิดพลาดของระบบ ตัวอย่าง ระบบหนึ่งออกแบบสำหรับกองทัพเรือ การทดสอบจะกระทำเป็นลำดับจนแน่ใจว่าถูกต้องตามความต้องการทุกประการ ในขั้นตอนของ Installation test เราจะนำระบบนี้ไปติดตั้งกับเรือที่ใช้ระบบนี้และทดสอบการทำงานในสภาพแวดล้อมจริงคือกลางทะเล เป็นต้น

การทดสอบระบบใหญ่ๆนั้น ควรมีการแบ่งการทดสอบระบบเป็นระยะ (phased system testing) เป็นลำดับของระบบย่อย โดยมีการกำหนดขอบเขตหรือหน้าที่ที่ชัดเจนเป็น functionality อาทิเช่น ระบบการเชื่อมต่อโทรศัพท์ การติดต่อสื่อสารนี้เราสามารถแบ่งระบบออกเป็น 4 ระบบย่อยๆ ประกอบด้วย

System A : การติดต่อภายในสาย

System B : การติดต่อภายในเมือง

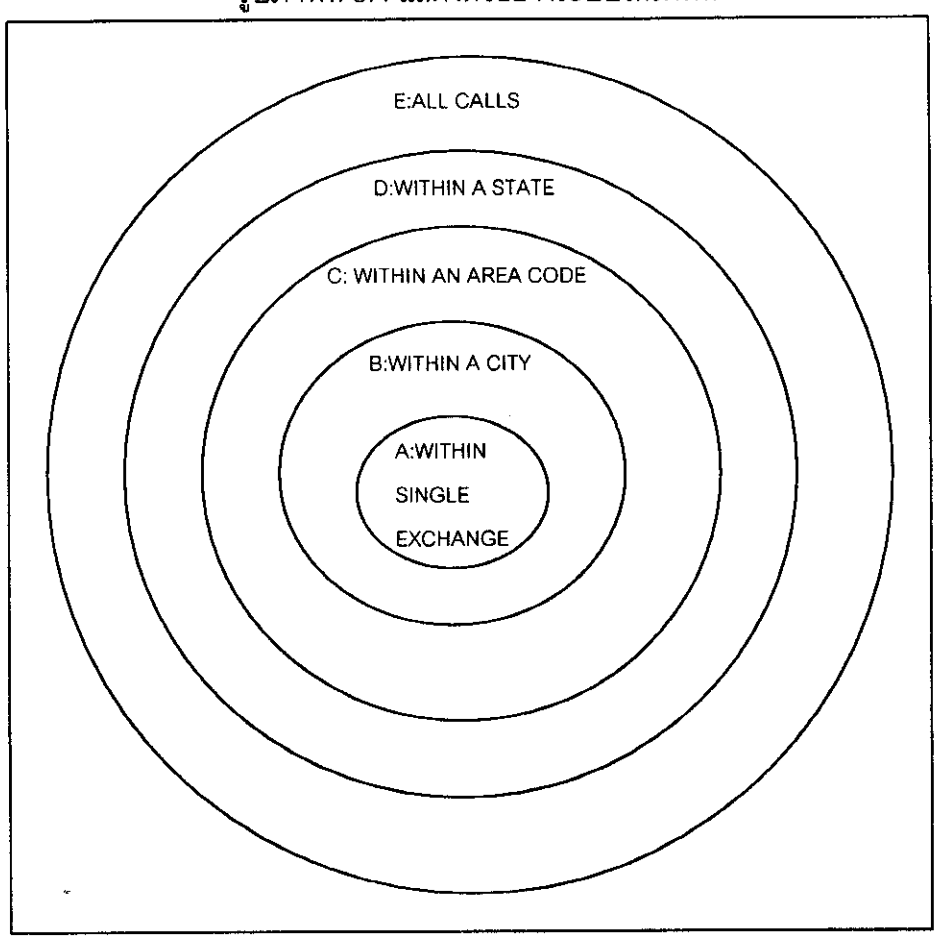
System C : การติดต่อภายในพื้นที่

System D : การติดต่อภายในประเทศ

System E : การติดต่อทุกๆที่

การทดสอบจะเริ่มจากการทดสอบ System A เมื่อระบบสามารถกระทำได้ตามวัตถุประสงค์ จะทดสอบ System B , C , D , E ไปตามลำดับ โดยการทดสอบทีมงานต้องสร้าง built plan หรือ integration plan เพื่อกำหนดการทดสอบในระบบย่อยๆ ว่าจะกระทำอย่างไร เมื่อใด ที่ไหน และโดยใคร ซึ่งระบบย่อยๆที่เราจะนำไปสร้างแผนการทดสอบบางครั้งเรียกว่า spin หรือ driver โดย spin เป็นจะถูกอ้างเป็นตัวเลขในระดับที่ต่ำที่สุดเรียกว่า spin 0 จากตัวอย่าง ระบบในรูปภาพที่ 8.4 สามารถสร้างแผนการทดสอบ (build plan) ดังตารางที่ 8.1

รูปภาพที่ 8.4 แสดงตัวอย่างระบบโทรศัพท์



ตารางที่ 8.1 Build Plan for Phone System

Spin	Function	Test Start	Test End
0	Calls within exchange	1 Sep 04	15 Sep 04
1	Calls within city	30 Sep 04	15 Oct 04
2	Calls within area code	25 Oct 04	5 Nov 04
3	Calls within State	10 Nov 04	20 Nov 04
4	All possible calls	1 Dec 04	15 Dec 04

Configuration Management Team เป็นทีมงานที่ทำงานรวมกับทีมงานทดสอบโดยทำหน้าที่รับผิดชอบถึงการเปลี่ยนแปลงต่างๆที่เกิดขึ้นในระบบเมื่อมีข้อผิดพลาดเกิดขึ้น โดยเป็นทีมงานที่ทราบถึงตำแหน่ง ผลกระทบต่างๆที่เกิดขึ้นของการดำเนินงานในระบบทั้งหมด เป็นผู้รับผิดชอบต่อการเปลี่ยนแปลงระบบเพื่อให้แน่ใจว่าการแก้ไขต่างๆนั้นถูกต้อง ไม่ก่อให้เกิดข้อผิดพลาดใหม่ๆในระบบตามมา ซึ่งทีมงานนี้จะบันทึกถึงการดำเนินการทดสอบ ผลของการทดสอบ การเปลี่ยนแปลงต่างๆในรูปของเอกสาร (documentation)

8.2

FUNCTION TESTING

Function Testing เป็นขั้นตอนแรกในการทดสอบระบบ หลังจากทีมงานได้ทดสอบโปรแกรมแล้วว่าสามารถทำงานได้ตามที่ได้ออกแบบไว้ ในขั้นตอนนี้เราจะไม่สนใจโครงสร้างของระบบแต่สนใจเฉพาะกิจกรรมที่ระบบสามารถกระทำได้เท่านั้น ดังนั้นในการทดสอบจึงจำเป็นอย่างยิ่งที่ต้องทราบถึงหน้าที่และกิจกรรมต่างๆที่ต้องการให้ระบบสามารถกระทำได้

กิจกรรมหรือหน้าที่ต่างๆที่ระบบกระทำนั้น ประกอบด้วยกลุ่มโมดูลต่างๆที่ทำงานร่วมกัน เรียกว่า thread ซึ่งการทดสอบในขั้นตอนนี้บางครั้งเรียกว่า thread testing สำหรับกลุ่มโมดูลที่มีขนาดเล็ก จะทำให้ผู้ทดสอบสามารถค้นหาความผิดพลาดพบได้ง่ายกว่ากลุ่มของโมดูลที่มีขนาดใหญ่ นอกจากนี้ควรใช้ทีมงานทดสอบเป็นทีมงานใหม่ที่ไม่ใช่ผู้ออกแบบและโปรแกรมเมอร์ที่พัฒนาระบบ มีการเลือกข้อมูลทดสอบทั้งค่าที่เป็นไปได้และค่าที่เป็นไปไม่ได้ และควรทราบด้วยว่าผลลัพธ์ที่ได้มีค่าเป็นอย่างไร ซึ่งการแก้ไขระบบต้องไม่อยู่ในระหว่างการทดสอบ และต้องทราบด้วยว่าจะหยุดการทดสอบเมื่อใด

Cause and Effect graphs

การทดสอบที่ดีควรมีการสร้างกรณีทดสอบ(test case) จากความต้องการ กระบวนการทดสอบนั้น จะเป็นการหาความสัมพันธ์ระหว่างข้อมูลเข้า(input) และข้อมูลออก(output) หรือระหว่างข้อมูลเข้าและการเปลี่ยนสถานะ(transformation) โดยข้อมูลเข้าเรียกว่า Cause ส่วนข้อมูลออกและการเปลี่ยนสถานะเรียกว่า Effect ผลลัพธ์ของการหาความสัมพันธ์นี้ได้เป็น กราฟบูลีน (Boolean graph) เรียกว่า **cause and effect graph**

การสร้างบูลีนกราฟ เริ่มจากการกำหนดข้อมูลต่างๆตามข้อกำหนดและสภาพแวดล้อมของระบบ นำไปแสดงในรูปแบบและกฎเกณฑ์เป็นกราฟ ต่อจากนั้นเปลี่ยนกราฟเป็นตารางการตัดสินใจ ซึ่งแต่ละคอลัมภ์ของตารางการตัดสินใจ จะมีส่วนเกี่ยวข้องกับกรณีทดสอบของการทดสอบหน้าที่ต่างๆ

ขั้นตอนการสร้าง Cause and Effect Graph มีหลายขั้นตอน ดังนี้

- พิจารณาถึงความต้องการของระบบ และแยกความต้องการออกเป็นฟังก์ชันเดี่ยวๆ
- พิจารณาถึง cause และ effect ต่างๆที่เป็นไปได้ทั้งหมด
- กำหนดหมายเลขให้แก่ cause และ effect ต่างๆ และสร้างเป็นโหนดของกราฟ
- ให้โหนดของ cause อยู่ทางด้านซ้ายของกราฟ ส่วน effect เป็นโหนดที่อยู่ทางขวาของกราฟ
- วาดความสัมพันธ์ทางตรรกระหว่าง cause และ effect โดยใช้การแทนด้วยภาพดังแสดงในรูปภาพที่ 8.6

ตัวอย่าง การทดสอบระบบตรวจสอบระดับของน้ำในทะเลสาบ โดยระบบจะส่งข้อมูลให้แก่โอเปอร์เรเตอร์ เกี่ยวกับความปลอดภัยของระดับน้ำในทะเลสาบ โดยข้อมูลเข้าเป็นฟังก์ชันที่อยู่ในรูปแบบ ดังนี้ INPUT : LEVEL(A,B) ซึ่ง A หมายถึงความสูงของน้ำหลังเขื่อนมีหน่วยเป็นฟุต B หมายถึงจำนวนฝนตกใน24ชั่วโมงที่ผ่านมาหน่วยเป็นนิ้ว
การประมวลผล เป็นฟังก์ชันในการคำนวณถึงระดับของน้ำในเขื่อน ว่าอยู่ในช่วงที่ปลอดภัย หรือระดับน้ำสูงเกินไป หรือระดับน้ำอยู่ในระดับต่ำ ผลลัพธ์ จะแสดงเป็นสารสนเทศปรากฏบนหน้าจอ ดังนี้

1. "LEVEL = SAFE" ถ้าระดับน้ำอยู่ในระดับปลอดภัยหรือต่ำ

2. "LEVEL = HIGH" ถ้าผลลัพธ์น้ำอยู่ในระดับที่สูง
3. "INVALID SYNTAX" ในกรณีอื่นๆ

เราสามารถแยกความต้องการออกเป็น 5 causes ดังนี้

cause 1 : ตัวอักษรห้าตัวแรกของคำสั่งต้องเป็นคำว่า 'LEVEL'

cause 2 : เป็นพารามิเตอร์สองตัวที่แยกด้วยเครื่องหมาย ',' (comma) และอยู่ภายในวงเล็บเล็ก(parentheses)

cause 3 : ค่าของพารามิเตอร์เป็นเลขจำนวนจริง เมื่อนำไปคำนวณถึงระดับน้ำจะอยู่ในระดับต่ำ

cause 4 : ค่าของพารามิเตอร์เป็นเลขจำนวนจริง เมื่อนำไปคำนวณถึงระดับน้ำจะอยู่ในระดับปลอดภัย

cause 5 : ค่าของพารามิเตอร์เป็นเลขจำนวนจริง เมื่อนำไปคำนวณถึงระดับน้ำจะอยู่ในระดับสูง

เราสามารถกำหนดถึงผลกระทบต่างๆที่เกิดขึ้นได้ดังนี้

Effect 1 : แสดง "LEVEL = SAFE " บนจอภาพ

Effect 2 : แสดง "LEVEL = HIGH " บนจอภาพ

Effect 3 : แสดง "INVALID SYNTAX " บนจอภาพ

เนื่องจากระบบนี้มีการตรวจสอบพารามิเตอร์ต่างๆ จึงมีการสร้างฟังก์ชันเพื่อตรวจสอบความถูกต้องของพารามิเตอร์เหล่านี้ โดยกำหนดโหนดเพิ่มขึ้นดังนี้

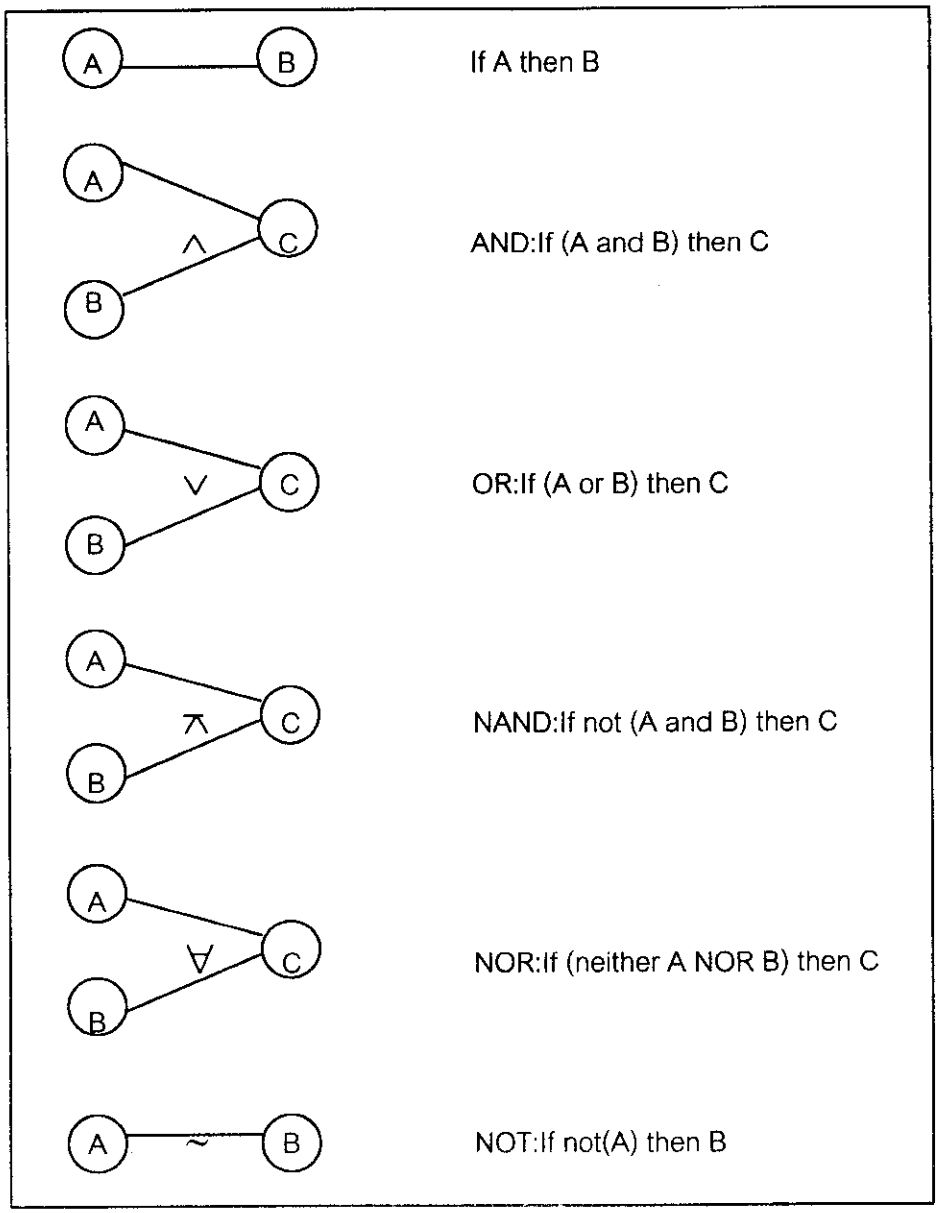
Node 10 : ตรวจสอบการใช้คำสั่งว่าถูกต้องใช่หรือไม่

Node 11 : ตรวจสอบโอเปอร์แรนด์ว่าถูกต้องใช่หรือไม่

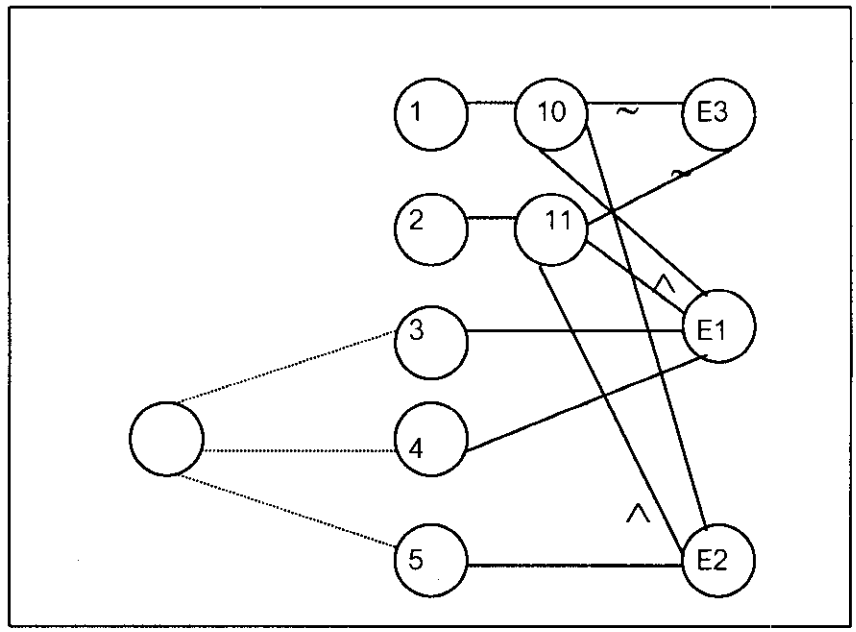
เราสามารถวาดรูปความสัมพันธ์ระหว่าง Cause และ Effect ได้ตามรูปภาพที่ 8.7 สำหรับโหนดทางซ้ายที่เชื่อมโยงด้วยเส้นปะ แสดงถึงเหตุการณ์ที่เกิดขึ้นได้นั้นเพียงเหตุการณ์ใดเหตุการณ์หนึ่งเท่านั้น

จากความสัมพันธ์ที่ได้นี้ เราสามารถนำไปสร้างเป็นตารางการตัดสินใจ (decision table) โดยแทนแถวของตารางด้วย Cause และ Effect ต่างๆ จากตัวอย่างนี้ประกอบด้วย 5 แถวสำหรับ Cause และ 3 แถวสำหรับ Effect สำหรับ Column แทนด้วยกรณีทดสอบต่างๆ ซึ่งมีผลให้ Cause มีค่าเป็นจริง(true)หรือเท็จ(false)

รูปภาพที่ 8.6 Cause and Effect Relationships



รูปภาพที่ 8.7 Cause and Effect Graph ของ ฟังก์ชัน LEVEL



ตารางที่ 8.2 ตารางการตัดสินใจของ Cause และ Effect Graph ของ ฟังก์ชัน LEVEL

		Tests				
		1	2	3	4	5
	1	I	I	I	I	S
	2	I	I	I	X	I
Causes	3	I	S	S	X	X
	4	S	I	S	X	X
	5	S	S	I	X	X
	E1	P	P	A	A	A
Effects	E2	A	A	P	A	A
	E3	A	A	A	P	P

จากตารางตัดสินใจที่ 8.2 แต่ละคอลัมภ์ของตารางการตัดสินใจเป็นกรณีทดสอบที่ส่งผลให้ Cause และ Effect มีสถานะที่แตกต่างกัน สำหรับสถานะของ Cause ที่มีค่าเป็น I หมายถึงค่า จริง (true) สำหรับ S แทนสถานะที่เป็นเท็จ (false) แต่ถ้าไม่สนใจว่า สถานะของ Cause มีค่าเป็นจริงหรือเป็นเท็จ (don't care) สถานะมีค่าเป็น X

ค่า P ของสถานะ Effect แสดงว่าเป็นเหตุการณ์ที่เกิดขึ้น สำหรับ A แสดงว่าเป็นเหตุการณ์ที่ไม่เกิดขึ้น จะเห็นได้ว่า ทุกกรณีทดสอบจะเกิดผลกระทบ (Effect) เพียงเหตุการณ์เดียวเท่านั้น

ประโยชน์ของ Cause and Effect Graph นี้ทำให้เราทราบถึงผลกระทบต่างๆที่เกิดขึ้นจากการนำ Cause ต่างๆมาพิจารณาร่วมกัน แต่วิธีการนี้ไม่เหมาะสำหรับระบบที่มีการทำงานวนรอบหรือซ้ำๆกัน

8.3

PERFORMANCE TESTING

เมื่อระบบที่พัฒนาขึ้นมานั้นผ่านกระบวนการทดสอบหน้าที่ตรงตามเอกสารกำหนดความต้องการแล้ว ขั้นตอนต่อไปเป็นการทดสอบประสิทธิภาพของระบบที่สามารถกระทำได้อันประกอบไปด้วย

1. ***Stress tests*** เป็นการทดสอบความสามารถของระบบภายในช่วงเวลาที่กำหนด อทิ เช่น ทดสอบว่าระบบสามารถกระทำงานได้ภายใต้อุปกรณ์ทั้งหมดที่เชื่อมต่อ หรือ ผู้ใช้ที่ปฏิบัติงานทั้งหมด เป็นจำนวนสูงสุดเท่าใด ถ้าทดสอบแล้วได้ผลว่าสามารถทำงานพร้อมกันได้สูงสุด 30 คน แสดงว่าถ้ามีการใช้งานมากกว่านี้ระบบจะไม่มีประสิทธิภาพนั่นเอง

2. **Volume tests** เป็นการทดสอบจำนวนของข้อมูลที่มากที่สุดที่ระบบสามารถกระทำได้ โดยตรวจสอบจากความต้องการเพื่อคำนวณถึงโครงสร้างข้อมูลต่างๆที่ระบบต้องปฏิบัติ รวมทั้งการปฏิบัติการต่างๆกับโครงสร้างข้อมูลต่างๆเหล่านั้น
3. **Configuration tests** เป็นการวิเคราะห์ถึง Hardware และ Software ที่สามารถปฏิบัติงานกับระบบ โดยตรวจสอบจากเอกสารระบุความต้องการอีกเช่นเดียวกัน โดยทดสอบการปฏิบัติงานกับระบบคอมพิวเตอร์ต่างๆเหล่านั้นเพื่อให้แน่ใจว่าระบบสามารถกระทำได้จริงตามที่ได้ระบุไว้
4. **Compatibility tests** เป็นการทดสอบที่จำเป็นสำหรับระบบที่มีการปฏิสัมพันธ์กับระบบอื่นๆ โดยเฉพาะอย่างยิ่งการเข้าถึงข้อมูล การดึงข้อมูล หรือการอ่านข้อมูล การทดสอบนั้นจะทดสอบในแง่ของความเร็ว ความถูกต้องในการดึงข้อมูลจากระบบฐานข้อมูลนั่นเอง นอกจากนี้การทดสอบนี้มีผลต่อการค้าเพราะทดสอบว่าซอฟต์แวร์นี้ใช้ร่วมกับซอฟต์แวร์อื่นในท้องตลาดได้หรือไม่
5. **Regression tests** การทดสอบนี้เป็นสิ่งจำเป็นอย่างยิ่ง สำหรับการนำระบบงานใหม่แทนที่ระบบงานที่กำลังปฏิบัติงานอยู่ เพื่อรับประกันว่าระบบใหม่มีประสิทธิภาพดีกว่า ซึ่งใช้ทดสอบสำหรับการพัฒนาที่มีหลายระยะ (phased development) หรือทดสอบโปรแกรมส่วนที่ได้รับการแก้ไขเฉพาะส่วน
6. **Security tests** เป็นการทดสอบความปลอดภัยของระบบ ซึ่งได้กำหนดไว้ในเอกสารความต้องการ โดยตรวจสอบการเข้าถึงในแง่ของฟังก์ชันการทำงาน การเข้าถึงข้อมูลต่างๆ ของผู้ใช้ระดับต่างๆ
7. **Timing tests** คำนวณถึงเวลาตอบสนองกับผู้ใช้ที่กระทำหน้าที่ต่างๆของระบบ
8. **Environmental tests** เป็นการพิจารณาความสามารถของระบบว่าสามารถทำงาน ณ สถานที่ตั้งได้หรือไม่ ซึ่งการติดตั้งอาจมีปัญหามาจากความร้อน ความชื้น สารเคมี สัญญาณไฟฟ้า หรือสภาพแวดล้อมที่ส่งผลกระทบต่อการทำงานของระบบ
9. **Quality tests** เป็นการคำนวณคุณภาพของซอฟต์แวร์ ในเรื่องของความน่าเชื่อถือ การบำรุงรักษาระบบ และการได้มาของระบบ

10. **Recovery tests** เป็นการทดสอบการค้นคืนข้อมูล หรือทดสอบการตอบสนองของระบบในกรณีเกิดความผิดพลาดของข้อมูลหรือ อุปกรณ์หรือ กำลังไฟ เป็นต้น
11. **Maintenance tests** เป็นการทดสอบการบำรุงรักษาระบบ ในกรณีต้องการเครื่องมือหรือกระบวนการที่ช่วยสำหรับวิเคราะห์ความผิดพลาดของระบบ โดยตรวจสอบถึงเครื่องมือต่างๆที่ได้ระบุไว้ เช่นโปรแกรมวิเคราะห์ความผิดพลาด แผนที่หน่วยความจำ การติดตามรายการปฏิบัติงาน ไดอะแกรมของวงจร หรือเครื่องมืออื่นๆ เพื่อตรวจสอบว่าเครื่องมือต่างๆเหล่านี้มีอยู่จริง และสามารถนำมาช่วยในการบำรุงรักษาระบบได้
12. **Documentation tests** ตรวจสอบเอกสารต่างๆที่จำเป็นและได้ระบุไว้ในเอกสารระบุความต้องการ พิจารณาและตรวจสอบว่าระบุได้อย่างถูกต้อง และคงที่ ง่ายต่อการอ่านหรือไม่
13. **Human factor tests** เป็นการทดสอบการแสดงผลทางจอภาพ ข่าวสารต่างๆที่แสดง รูปแบบของรายงาน ว่าชัดเจนหรือสวยงามหรือไม่ ง่ายต่อผู้ใช้ หรือไม่

8.4

ACCEPTANCE TESTING

เมื่อผ่านขั้นตอนการทดสอบฟังก์ชันและทดสอบประสิทธิภาพมาแล้ว แสดงว่าระบบที่พัฒนาขึ้นนั้นสามารถกระทำงานได้ตามความต้องการที่ระบุไว้ทั้งหมด สำหรับขั้นตอนต่อไปเป็นการนำระบบไปให้ลูกค้าหรือผู้ใช้ทดสอบ โดยให้ลูกค้ากำหนดถึงกรณีทดสอบต่างๆได้ตามความต้องการ

Type of Acceptance Testing

สำหรับการทดสอบโดยลูกค้านั้นมีด้วยกัน 3 วิธี คือ

1. **Benchmark test** การทดสอบด้วยวิธีนี้ลูกค้าจะเตรียมกลุ่มของกรณีทดสอบซึ่งแทนการปฏิบัติงานของระบบจริงๆ ลูกค้าจะทดสอบประสิทธิภาพของระบบในแต่ละกรณีทดสอบ โดยใช้ผู้ใช้ที่กระทำงานจริงๆในระบบหรืออาจเป็นทีมงานทดสอบที่ปฏิบัติงานเฉพาะกรณีก็ได้ วิธีการนี้ลูกค้าสามารถใช้สำหรับความต้องการพิเศษก็ได้ เช่น ลูกค้าต้องการติดตั้งข่ายงานเสียงและข้อมูล ความต้องการคือต้องการความเร็วมากๆและใช้งานง่ายด้วย ในกรณีที่มีบริษัทหลายบริษัทได้มาเสนอขาย ลูกค้าสามารถทดสอบโดยใช้วิธี benchmark เพื่อตัดสินใจถึงระบบที่ตรงกับความต้องการมากที่สุด
2. **Pilot test** เป็นการติดตั้งระบบเพื่อทำการทดสอบ ผู้ใช้ระบบจะทำการทดสอบระบบในสิ่งที่ต้องกระทำซ้ำๆกันทุกๆวัน(everyday working) ทดสอบทุกๆหน้าที่ ซึ่งแตกต่างจากวิธีแรกที่ทดสอบเฉพาะความต้องการพิเศษ บางครั้งการทดสอบระบบนี้อาจจะทดสอบกับผู้ใช้ระบบจริงที่อยู่ในองค์กรซึ่งเรียกว่า alpha test ก่อนจะนำไปให้ลูกค้าทดสอบที่เรียกว่า beta test การทดสอบวิธีนี้เหมาะสำหรับระบบที่มีปรับปรุงหรือแก้ไขการทำงานของระบบเดิม เช่น ระบบสำนักงานอัตโนมัติ หรือระบบปฏิบัติงานเวอร์ชันใหม่
3. **Parallel test** เป็นการทดสอบโดยระบบเก่าและระบบใหม่ทำงานขนานกันไป ผู้ใช้ระบบสามารถเปรียบเทียบ และทดสอบการทำงานของระบบใหม่กับระบบเก่า เพื่อให้แน่ใจว่าระบบเก่าสามารถทำงานมีประสิทธิภาพและแทนที่ระบบเก่าได้

8.5

INSTALLATION TESTING

ในขั้นตอนที่แล้วลูกค้าเป็นผู้กำหนดกรณีทดสอบ ซึ่งผลของการทดสอบเป็นที่น่าพอใจ ลูกค้ายอมรับ ขั้นตอนต่อไปเป็นการทดสอบการติดตั้งระบบ แต่ถ้าในขั้นตอนที่แล้วทดสอบในสถานที่ตั้งอยู่แล้วขั้นตอนนี้ไม่จำเป็น การทดสอบในขั้นตอนนี้เป็นการทดสอบถึงอุปกรณ์ที่นำมาใช้ร่วมกับระบบ นั้นสามารถติดต่อหรือทำงานร่วมกับระบบได้หรือไม่ หน้าที่ต่างๆที่ระบบกระทำได้รวมทั้งการปฏิบัติการกับข้อมูลการเข้าถึงข้อมูลต่างๆถูกต้องหรือไม่ ผลของการทดสอบคือ ระบบสามารถกระทำได้โดยสมบูรณ์ หรือ ต้องแก้ไข ซึ่งอาจมีผลมาจากเงื่อนไขของสถานที่ติดตั้ง

8.6

TEST TOOLS

เครื่องมือที่สามารถนำมาใช้ช่วยในการทดสอบนั้นมีมากมาย ส่วนมากเป็นเครื่องมืออัตโนมัติเพื่อใช้สำหรับจับ(capture) ข้อมูล เพื่อใช้สำหรับคำนวณประสิทธิภาพของระบบ เช่น

1. Simulator เป็นเครื่องมือที่ช่วยแสดงคุณลักษณะทั้งหมดของอุปกรณ์และระบบ แต่เป็นการจำลองไม่มีเครื่องมือหรืออุปกรณ์จริงๆ เช่น การจำลองการบินที่ไม่มีเครื่องบินจริงๆ แต่มีแต่อุปกรณ์จำลองที่สามารถให้ผู้ใช้ได้เรียนรู้การบินจากเครื่องบินจำลองนี้
2. Monitor เป็นอุปกรณ์หรือซอฟต์แวร์ในการดักจับข้อมูลที่ผ่านจากโปรเซสหนึ่งไปยังอีกโปรเซสหนึ่ง เช่น โปรแกรมมอนิเตอร์ข้อมูลเข้าและข้อมูลออก

ระหว่างโปรเซสเซอร์ 2 ตัว เราใช้โปรแกรมนี้เก็บเหตุการณ์ก่อนและเหตุการณ์หลังจากมีข้อผิดพลาดเกิดขึ้นในการส่งผ่านข้อมูลใน snapshot ซึ่งช่วยให้เราค้นหาหรือติดตามแหล่งของความผิดพลาดต่างๆที่เกิดขึ้นเพื่อแก้ไขได้ง่ายขึ้น

3. Analyzer เป็นเครื่องมือในการวิเคราะห์ข้อมูลต่างๆตามระเบียบที่กำหนด เช่นวิเคราะห์จำนวนคำสั่งต่างๆที่ปฏิบัติงานในระหว่างการทดสอบ รวมทั้งรูทีนและคำสั่งที่ไม่ได้ปฏิบัติงาน นอกจากนี้มีการวิเคราะห์ถึงเวลาในการทำงานของระบบ อทิเช่น ในหน่วยความจำ การวิเคราะห์เวลาที่ใช้ในแต่ละพื้นที่ของหน้าที่ต่างๆ ซึ่งการบันทึกต่างๆเหล่านี้มีประโยชน์ในการทดสอบประสิทธิภาพ

8.7

TEST TEAM

ทีมงานในการทดสอบจะรับผิดชอบในขั้นตอนของ function และ performance testing ทีมงานในการทดสอบเป็นทีมงานที่ทำหน้าที่ในการทดสอบระบบตามที่ระบุในความต้องการ โดยใช้วิธีการหรือเครื่องมือช่วยต่างๆ ประกอบไปด้วย

1. Professional Testers เป็นผู้ที่มีความสำคัญในการทดสอบซึ่งเกี่ยวข้องกับการทดสอบตั้งแต่เริ่มต้น การออกแบบแผนการทดสอบ กรณีทดสอบ โดยทำงานร่วมกับ configuration management team เพื่อเตรียมเอกสารหรือเครื่องมือต่างๆเพื่อช่วยในการทดสอบ เป็นผู้บริหารจัดการถึงวิธีการ กระบวนการ ในการทดสอบ

2. Analysts เป็นนักวิเคราะห์ระบบที่คุ้นเคยระบบเป็นอย่างดี ซึ่งการทดสอบต้องมีการนำระบบใหม่และระบบเก่ามาเปรียบเทียบเพื่อทดสอบ และวิเคราะห์ว่ามีความจำเป็นและตรงกับความต้องการของลูกค้าหรือไม่ ดังนั้นทีมงานต้องทำงานร่วมกับนักวิเคราะห์ระบบเพื่อกำหนดถึงวิธีการหรือหนทางในการแก้ปัญหาต่างๆ
3. System Designers เป็นนักออกแบบระบบที่เข้าใจถึงวิธีการแก้ปัญหา รู้ถึงข้อกำหนดต่างๆ ทราบถึงการแบ่งระบบออกเป็นระบบย่อยๆ และข้าใจถึงการทำงานของระบบ ดังนั้นการออกแบบกรณีทดสอบเพื่อให้ครอบคลุมทุกๆกรณี ทีมงานในการทดสอบต้องพึ่งพิงนักออกแบบระบบเพื่อช่วยในการกำหนดทุกทางเลือกที่เป็นไปได้ทั้งหมดเพื่อใช้ในการทดสอบ
4. Users เป็นสมาชิกหนึ่งในทีมงานทดสอบ เป็นผู้ที่จะชี้ชัดว่าระบบที่พัฒนาขึ้นง่ายต่อการใช้งานหรือไม่ เป็นมิตรต่อผู้ใช้หรือไม่
5. Configuration Management Specialists เป็นตัวแทนของทีมงานทดสอบในการเปลี่ยนแปลงสิ่งต่างๆที่เกี่ยวข้องในการทดสอบ อทิเช่นเมื่อพบความผิดพลาด(error) หรือ ต้องการเปลี่ยนแปลงความต้องการ Configuration Management Specialists จะจัดการถึงการเปลี่ยนแปลงในเรื่องต่างๆไม่ว่าจะเป็นเอกสารความต้องการ , การออกแบบ หรือทุกๆสิ่งที่มีผลต่อการเปลี่ยนแปลงที่เกิดขึ้น รวมทั้งการปรับปรุงกระบวนการทดสอบ

8.8

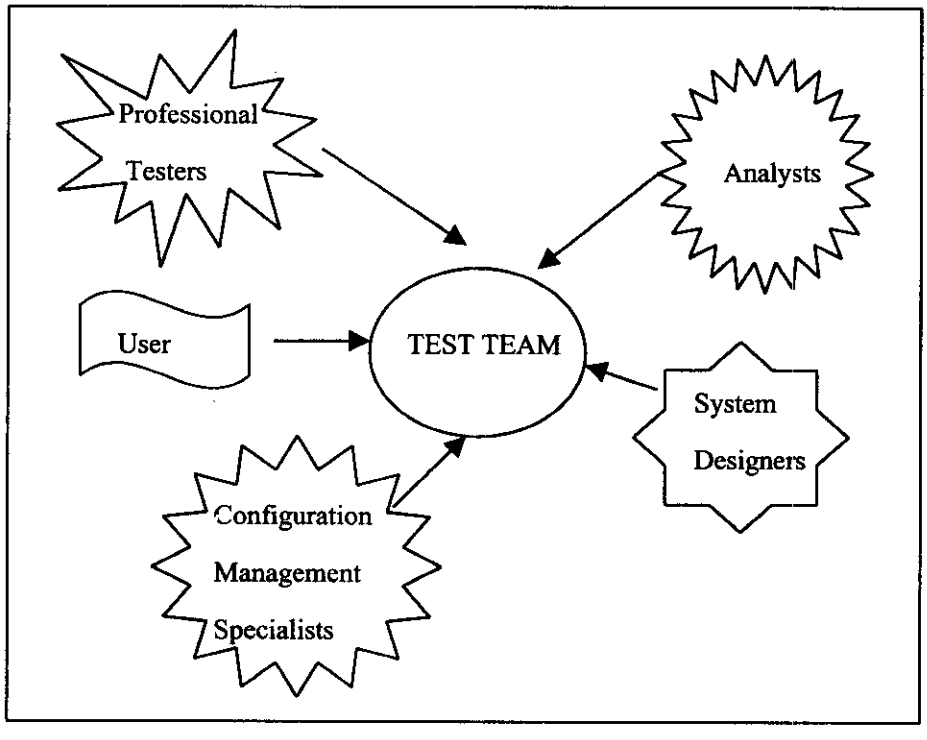
TEST DOCUMENTATION

สำหรับระบบที่มีความซับซ้อนมากๆ มีการประมวลผลแบบกระจายหรือเป็นระบบ real time เป็นระบบใหญ่ ที่มีผู้ใช้ระบบจำนวนมาก การทดสอบจะมีความยากตามไปด้วย ดังนั้นวิธีการในการ

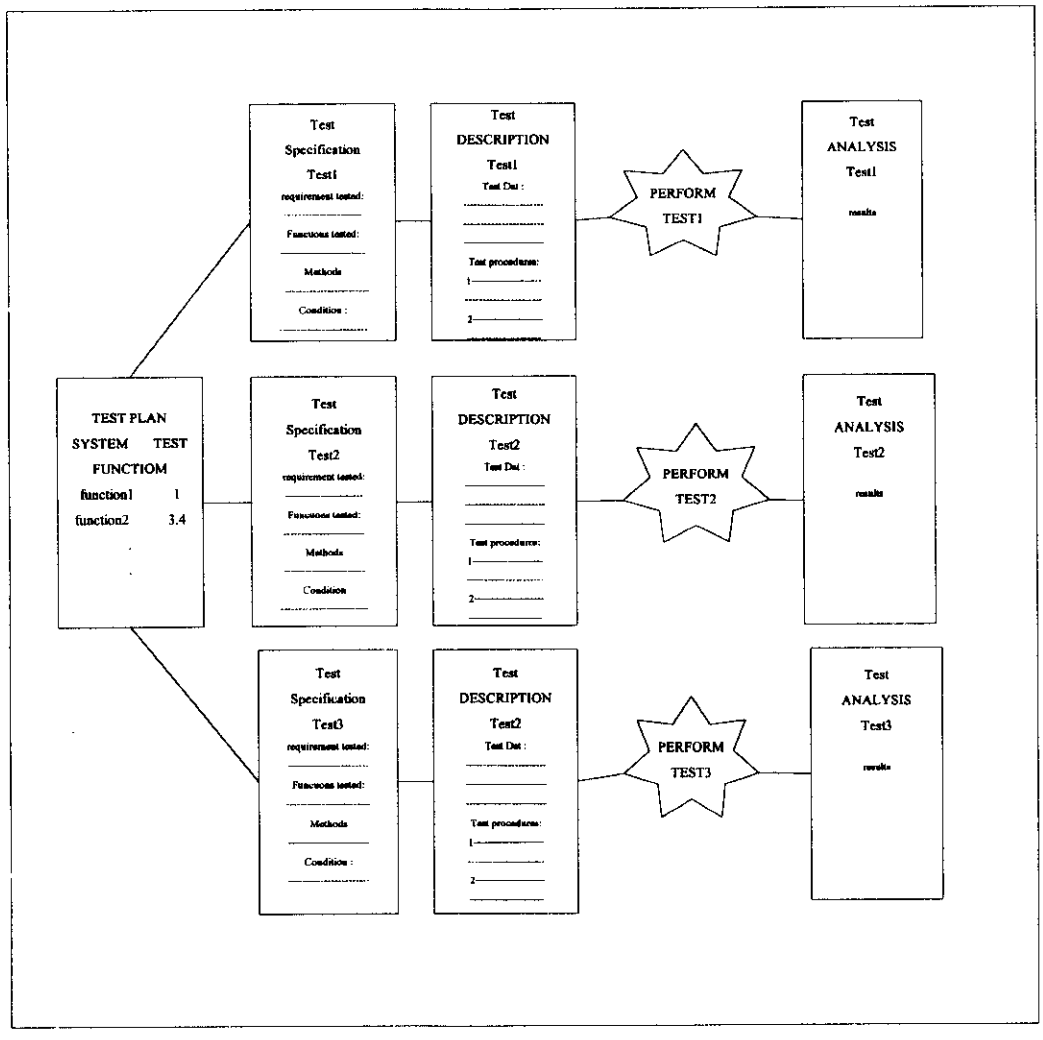
ควบคุมความซับซ้อนและความยากในการทดสอบนั้น เราต้องมีการออกแบบเอกสารทดสอบที่มีความสมบูรณ์และครอบคลุม เอกสารที่จำเป็นมีอยู่ด้วยกันหลายชนิด ได้แก่

1. Test Plan เป็นเอกสารที่บรรยายถึงการทำงานของระบบ ซึ่งวางแผนถึงการทดสอบคุณลักษณะและฟังก์ชั่นทั้งหมดที่สามารถปฏิบัติงานได้
2. Test Specification and Evaluation เป็นเอกสารที่บรรยายถึงรายละเอียดในการทดสอบในแต่ละฟังก์ชัน โดยกำหนดถึงหน้าที่ต่างๆที่ฟังก์ชันนั้นสามารถกระทำได้
3. Test Description เป็นการทดสอบการทำงานของฟังก์ชันต่างๆ ซึ่งมีการป้อนข้อมูลหรือกระทำตามขั้นตอนต่างๆในฟังก์ชันนั้นๆ
4. Test Analysis Report เป็นเอกสารในการสรุปถึงผลของการทดสอบ

รูปภาพที่ 8.9 สมาชิกในทีมงานทดสอบ



รูปภาพที่ 8.10 แสดงถึงความสัมพันธ์ของเอกสารต่างๆในกระบวนการทดสอบ



Test Plan

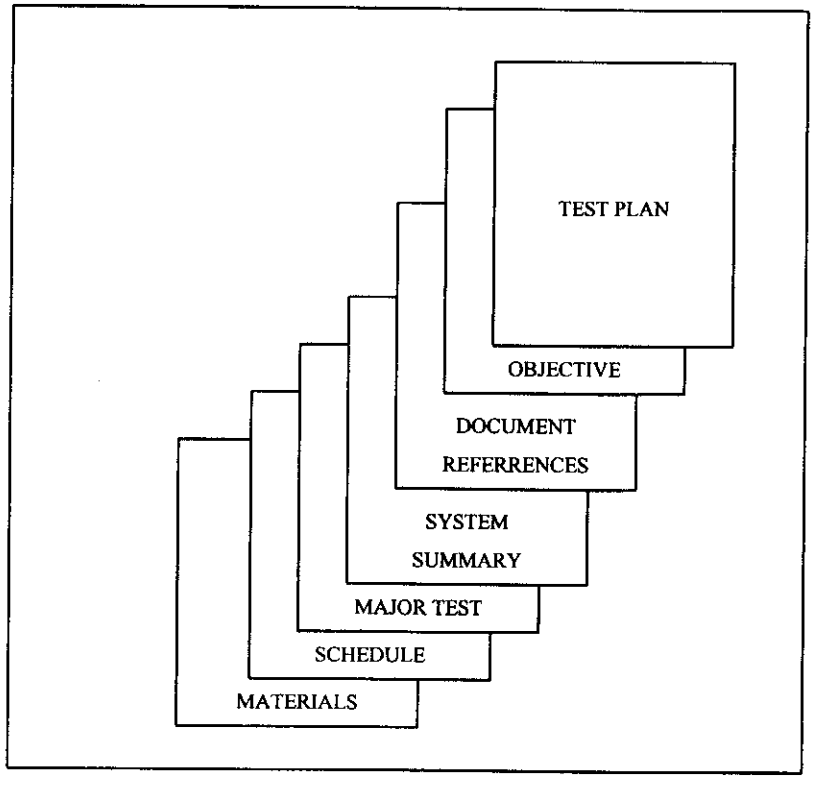
ในบทที่ 7 ได้กล่าวถึงบทบาทของการวางแผนการทดสอบ ซึ่งการวางแผนนี้เป็นประโยชน์สำหรับการทดสอบระบบโดยตรง จากรูปภาพที่ 8.11 แสดงส่วนประกอบของแผนการทดสอบ เริ่มจาก

1. กำหนดวัตถุประสงค์ (Objective) โดยรายละเอียดประกอบไปด้วย

- แนะนำวิธีการจัดการทดสอบ
- แนะนำเทคนิคต่างๆที่ใช้ในระหว่างการทดสอบ
- สร้างแผนงานและกำหนดระยะเวลาในการทดสอบ รวมทั้งรายละเอียดของอุปกรณ์ที่จำเป็น วิธีการทดสอบ ผลลัพธ์ที่ต้องการ
- อธิบายลักษณะและขอบเขตของแต่ละการทดสอบ
- อธิบายถึงหนทางของการทดสอบที่สามารถกระทำการได้สำเร็จและสามารถประเมินค่าของหน้าที่และประสิทธิภาพของระบบได้
- ข้อมูลในการนำเข้า ของแต่ละฟังก์ชันทดสอบ ผลที่คาดว่าจะได้รับ

2. เอกสารอ้างอิง (References) เป็นเอกสารที่บรรยายความสัมพันธ์ระหว่างเอกสารต่างๆ ประกอบด้วย เอกสารระบุความต้องการ เอกสารออกแบบ เอกสารในการสร้างระบบ และเอกสารในการทดสอบระบบ อทิเช่น การอ้างตัวเลข 4.9 ในเอกสารระบุความต้องการ ซึ่งอาจส่งผลกระทบในโมดูล 5.6 ของเอกสารการออกแบบ และนำมาทดสอบในเอกสารในการทดสอบที่ 8.1 เป็นต้น
3. การสรุประบบ (System summary) เป็นการสรุปถึงแผนงานและเหตุการณ์ต่างๆในการทดสอบ โดยไม่ได้อธิบายในรายละเอียด โดยบรรยายถึงข้อมูลเข้าและผลลัพธ์ที่ได้จากระบบหลักๆ
4. อธิบายถึงการทดสอบหลักๆ(Major tests) เป็นโครงร่างของการทดสอบในภาพรวมของระบบ โดยแบ่งเป็นส่วนประกอบหลักๆ รวมทั้งวิธีการที่ใช้
5. ระยะเวลา (Schedule) เป็นการวางแผนในส่วนของเวลาในการทดสอบในแต่ละส่วนประกอบว่าเริ่มเมื่อใด เป็นระยะเวลาเท่าใด ในส่วนนี้สามารถใช้ Activity Graph หรือ Milestone Chart เพื่อช่วยในการควบคุมและวัดความก้าวหน้าของการทดสอบได้
6. ทดสอบการทำงาน (test materials) ในเทอมของการส่งมอบ เช่น การทดสอบระบบการจัดการฐานข้อมูล เราจะสร้างฐานข้อมูลสมมุติขึ้นมาเพื่อให้ผู้ใช้ที่ทำงานจริงทดลองใช้ก่อนที่ทีมงานทดสอบจริงจะมาทดสอบ หรือ การทดสอบความปลอดภัยหรือสิทธิต่างๆในการทำงาน ผู้ใช้อาจสร้าง password หรือ สิทธิต่างๆขึ้นมาสำหรับทีมงานทดสอบ ก่อนที่การทดสอบจะเริ่มขึ้น เป็นต้น

รูปภาพที่ 8.11 Test Plan Components



TEST SPECIFICATION AND EVALUATION

แผนการทดสอบ(Test Plan) จะบรรยายถึงการแบ่งการทดสอบระบบโดยรวมและแบ่งเป็นส่วนประกอบย่อยๆ อทิเช่น ต้องการทดสอบระบบการประมวลผลแบบกระจายของคอมพิวเตอร์ห การทดสอบระบบนี้จะมีการแบ่งระบบออกเป็นระบบย่อยๆ และในแต่ละระบบย่อย ต้องมีการสร้างรายละเอียดของการทดสอบและการคำนวณการทำงาน (test specification and evaluation) เริ่มจากเขียนความต้องการที่ต้องการจากระบบย่อยนี้ ต่อจากนั้นพิจารณาถึงความต้องการต่างๆที่ได้กำหนดขึ้นนำมาสร้างวัตถุประสงค์ของการทดสอบ

วิธีการนี้ทำให้เรามองเห็นความเกี่ยวพันธ์ระหว่างความต้องการกับการทดสอบ โดยสร้างเป็นตารางหรือผังเพื่อแสดงความสัมพันธ์ ตามตารางที่ 8.3 ซึ่งการเขียนความต้องการต่างๆจะแสดงในส่วนหัวตาราง ส่วนหน้าที่ต่างๆที่ทำการทดสอบจะเขียนทางซ้ายของแต่ละแถว

รูปภาพที่ 8.3 Test-Requirement Correspondence Chart

Test	Requirement		
	Generate And Maintain Data Base	Selectively Retrieve Data	Produce Specialized Reports
1.Add new record	X		
2.Add field	X		
3.Change field	X		
4.Delete record	X		
5.Delete field	X		
6.Create index		X	
Retrieve record with a request			
7.Cell number		X	
8.Water height		X	
9.Canopy height		X	
10.Ground cover		X	
11.Percolation rate		X	
12.Print full data base			X
13.Print Directory			X
14.Print Keywords			X
15.Print simulation summary			X

สำหรับ X แทนถึงกิจกรรมต่างๆที่ระบบสามารถกระทำได้เพื่อให้ได้ตามวัตถุประสงค์ที่กำหนด โดยต้องระบุถึงเงื่อนไขของการทดสอบให้ชัดเจน

- การรับเข้าข้อมูลเข้าสู่ระบบว่ามาจากผู้ใช้หรืออุปกรณ์ใดหรือจากสร้างจากโปรแกรมหรืออุปกรณ์ใด

- การทดสอบนั้นต้องให้ครบทุกๆหน้าที่ ทุกๆทางเลือก ทุกๆส่วน
- ข้อมูลมีการบันทึกอย่างไร
- ถ้าการทดสอบมีชุดหรือลำดับของการทดสอบย่อยๆ ต้องทราบถึงลำดับในการทดสอบ

ในกระบวนการนี้ทีมงานทดสอบสามารถใช้เครื่องมืออัตโนมัติเพื่อช่วยในการประเมินและสรุปรายงาน หรือทำการเปรียบเทียบกับผลลัพธ์ที่คาดหวัง ได้

Test Description

เป็นเอกสารที่แสดงถึงการทดสอบข้อมูลในส่วนรายละเอียด หลังจากระบุถึงการทดสอบในฟังก์ชันย่อยต่างๆแล้ว เอกสารนี้เป็นข้อเสนอแนะในการทดสอบ ซึ่งเริ่มจากการเปิดเครื่องการโต้ตอบกับระบบ ในขั้นตอนต่างๆตามลำดับ ซึ่งต้องมีรายละเอียดที่ละเอียดและชัดเจน อันประกอบด้วย

1. Mean of Control
2. Data
3. Procedure

รายละเอียดโดยทั่วไป เริ่มจากการอธิบายการควบคุมจากผู้ใช้หรือระบบอัตโนมัติที่กระทำ เช่นมีการรับข้อมูลจากผู้ใช้โดยการป้อนทางแป้นพิมพ์ มีการทำปฏิบัติการต่างๆโดยอัตโนมัติเป็นลำดับ ซึ่งในการทดสอบจะมีการนำข้อมูลนำเข้าต่างๆ หรือป้อนคำสั่งที่สั่งให้ระบบกระทำตามลำดับ

ตัวอย่าง รายละเอียดของข้อมูลทดสอบสำหรับรูทีนการเรียงลำดับ (sort)

INPUT DATA :

ข้อมูลนำเข้าถูกเตรียมจากโปรแกรมชื่อ LIST โดยทำการสุ่มข้อมูลเป็นกลุ่มของตัวอักษรเป็นคำใดๆที่แตกต่างกันจำนวน N คำ แต่ละคำมีความยาวไม่เกิน M ตัวอักษร โปรแกรมนี้ถูกเรียกใช้โดยมีรูปภาพแบบดังนี้ RUN LIST(N,M) โปรแกรมนี้เราเรียกว่าตัวขับการทดสอบ (test driver) ผลลัพธ์ของการทำงานจะนำข้อมูลต่างๆไปเก็บในตัวแปรที่เป็น global ชื่อ LISTBUF การ

ทดสอบจะมีการกำหนดกลุ่มของข้อมูลทดสอบเพื่อใช้ในการทดสอบดังนี้ มีการเรียกใช้ โมดูล LIST แต่กำหนดค่า N และ M มีค่าแตกต่างกัน

CASE 1: N = 5 , M = 5

CASE 2: N = 10 , M = 5

CASE 3: N = 15 , M = 5

CASE 4 : N = 50 , M = 5

CASE 5 : M = 100 , M = 10

CASE 6 : M = 150 , M = 10

INPUT COMMANDS :

โดยรูทีนการเรียงลำดับ (sort) สามารถเรียกใช้โดยใช้คำสั่ง

RUN SORT (INBUF , OUTBUF) หรือ RUN SORT (INBUF)

OUTPUT DATA :

ถ้าการเรียกใช้มีพารามิเตอร์ 2 ตัว ผลของการทำงานของรูทีนเรียงลำดับ จะนำข้อมูลที่ผ่านการเรียงลำดับเก็บใน OUTBUF ในกรณีอื่นๆเก็บใน INBUF

SYSTEM MESSAGE :

ในระหว่างที่ดำเนินการเรียงลำดับข้อมูลมี ข่าวสารแสดงทางจอภาพว่า

"Sorting ....Please wait..."

จนกระทั่งการเรียงลำดับเสร็จเรียบร้อย จะปรากฏข่าวสารทางจอภาพว่า

"SORT completed "

และในกรณีต้องการหยุดหรือออกจากกระบวนการทดสอบ ให้กด CTRL-C บนแป้นพิมพ์ โดยทั่วไปเราจะเรียกวิธีการทดสอบในรายละเอียดว่า test script เนื่องจากมีการทดสอบทีละขั้นทีละขั้น (step-by-step)

ตัวอย่าง ส่วนของการ test script ในการทดสอบ "Change Field" ซึ่งเป็นฟังก์ชันจากตารางที่ 8.3 ซึ่งสามารถแสดงขั้นตอนได้ดังตารางที่ 8.4

ตารางที่ 8.4 Test Script for Change Field Function

Step N :	Press function key 4 : Access data file
Step N+1 :	Screen will ask for name of data file. Type 'sys.test.txt"
Step N+2 :	Menu will appear, reading: - Delete file - Modify file - Rename file Place cursor next to 'Modify file ' and press RETURN key.
Step N+3 :	Screen will ask for record number. Type '4016'
Step N+4 :	Screen will fill with data fields for record 4017 : RECORD NUMBER : 4017 CELL X : 0042 CELL Y : 0036 SOIL TYPE : CLAY PERCOLATION RATE : 4 FT/HR VEGETATION : KUDZU CANOPY HEIGHT : 25 FT WATER TABLE : 12 FT CONSTRUCT : OUTHOUSE MAINTENANCE CODE : 3T/4F/9R
Step N+5 :	Press function key 9 : Modify.
Step N+6 :	Entries on screen will be highlighted. Move cursor to VEGETATION field. TYPE 'GRASS' over 'KUDZU' and press RETURN key.
Step N+7 :	Entries on screen will no longer be highlighted . VEGETATION Field should now read 'GRESS'
Step N+8 :	Press function key 16: Return to previous screen.

ตารางที่ 8.4 Test Script for Change Field Function (ต่อ)

Step N+9 :	Menu will appear, reading : - Delete file - Modify file - Rename file To verify that modification has been recorded. Place cursor next to 'Modify file ' and press RETURN key
Step N+10:	Screen will ask for record number. TYPE ' 4017'
Step N+11:	Screen will fill with data fields for record 4017 RECORD NUMBER : 4017 CELL X : 0042 CELL Y : 0036 SOIL TYPE : CLAY PERCOLATION RATE : 4 FT/HR VEGETATION : GRASS CANOPY HEIGHT : 25 FT WATER TABLE : 12 FT CONSTRUCT : OUTHOUSE MAINTENANCE CODE : 3T/4F/9R

จากตัวอย่าง ลำดับของการทดสอบรายละเอียดนั้นถูกกำหนดตามลำดับของตัวเลขที่ระบุ โดยอธิบายถึงเหตุการณ์ต่างๆที่เกิดขึ้นจริง อทิเช่นการกดคีย์ต่างๆ การแสดงผลทางจอภาพ ผลของการทำงานที่ได้ อุปกรณ์ต่างๆที่เกี่ยวข้อง รายงานที่ได้รับ เป็นต้น

Test Analysis Report

เป็นเอกสารที่บรรยายผลลัพธ์ของการทดสอบ ซึ่งเป็นเอกสารใช้สำหรับวิเคราะห์การทดสอบ ผลของการทดสอบอาจถูกต้อง หรือเกิดข้อผิดพลาด เอกสารจะทำให้ทราบว่าตำแหน่งใดในระบบเกิดข้อผิดพลาด ทำให้สามารถแก้ไขได้ง่าย โดยสร้างรูปแบบเอกสารที่เป็นมาตรฐานทำให้ทราบตำแหน่งข้อผิดพลาด ทราบสถานะของระบบก่อนเกิดข้อผิดพลาด ทราบถึงความผิดพลาดที่เกิดขึ้น ทราบถึงการกระทำหรือกระบวนการที่เกิดขึ้นเมื่อเกิดเหตุการณ์ผิดพลาด ทราบถึงรายละเอียดของการทำงานที่ปราศจากความผิดพลาด ทราบถึงการอ้างถึงความต้องการที่เกี่ยวข้อง

ผลกระทบของเหตุการณ์ที่เกิดขึ้นในระบบ เป็นต้น รายงานที่แสดงถึงความผิดพลาดที่เกิดขึ้นเรียกว่า discrepancy report form (DRF) ซึ่งประกอบไปด้วย

1. state กล่าวถึงระบบก่อนที่เกิดข้อผิดพลาดขึ้นในระบบ
2. evidence เหตุการณ์ที่เกิดข้อผิดพลาดขึ้น
3. action กิจกรรมหรือการกระทำที่เกิดขึ้นปรากฏขึ้นเมื่อมีความผิดพลาดเกิดขึ้นในระบบ
4. should อธิบายถึงรายละเอียดที่ระบบควรจะกระทำโดยปราศจากความผิดพลาด
5. requirements มีการอ้างอิงที่เกี่ยวพันธ์กับความต้องการที่ระบุหรือกำหนดไว้
6. impact ผลกระทบของการเกิดความผิดพลาดนี้ที่เกิดขึ้นในระบบ
7. severity ระดับของการแก้ปัญหาหรือความรุนแรงที่เกิดขึ้น ถ้าสามารถกระทำได้

โดย DRF มีประโยชน์ต่อทีมงานทดสอบในการตัดสินใจถึงกิจกรรมที่ต้องกระทำ ลูกค้าและผู้ทดสอบมีการปรึกษากันถึงการแก้ไขปัญหา รวมทั้งสามารถใช้กำหนดสมาชิกในทีมงานทดสอบให้แก้ไขข้อผิดพลาดต่างๆ อีกทั้งเป็นเอกสารที่ใช้ในการบำรุงรักษาระบบในภายหลังได้อีกด้วย

8.9

ตัวอย่าง

สมมุติว่าการทดสอบดังตารางที่ 8.4 ต้องการปรับปรุงข้อมูลในระเบียนโดยพิมพ์ 'GRASS' แทนที่ของ 'KUDZU' ซึ่งทดสอบตามขั้นตอนดังนี้

Step N+5 : Press function key 9 :Modify

Step N+6 : Entries on screen will be highlighted. Move cursor to VEGETATION field. Type 'GRASS' over 'KUDZU' and press RETURN key

Step N+7 : Entries on screen will no longer be highlighted.

VEGETATION field should now read 'GRASS'

แต่เมื่อกระทำการทดสอบพบความผิดพลาดที่เกิดขึ้น นั่นคือใน Step N+7 ฟิลด์ VEGETATION ไม่มีการเปลี่ยนแปลงใดๆเกิดขึ้นยังคงเก็บ 'KUDZU' ซึ่งเป็นข้อมูลเก่า ทีมงานต้องมีการบันทึกถึงเหตุการณ์และความผิดพลาดที่เกิดขึ้น ดังรูปภาพที่ 8.12

รูปภาพที่ 8.12 แสดงถึง Discrepancy Report From

DRF Number :..... Tester Name :.....

Date : ..... Time : .....

Text Number :.....

Script step executed when error occurred : .....

Description

.....

.....

Activities before occurrence of error

.....

.....

Expected results :

.....

Requirements affected :

.....

Impact of error on test :

.....

Impact of error on system :

.....

Severity Level : Low 1 2 3 4 5

CT 484 283

ในการแก้ไขความผิดพลาดที่เกิดขึ้นนั้นต้องกระทำเป็นอิสระต่อการทดสอบ โดยการทดสอบจะต้องกระทำต่อไปจนกระทั่งจบสมบูรณ์ เพราะความผิดพลาดในการทดสอบอาจแตกต่างจากข้อผิดพลาดที่เกิดขึ้นในระบบได้เช่น การแก้ไขข้อมูลในเรคอร์ดนั้นมีการเปลี่ยนแปลงข้อมูลในดิสก์เรียบร้อยแล้ว แต่ผิดพลาดในการทำงานของโมดูลในการดึงข้อมูล(retrieve) เพื่อแสดงบนจอภาพ ความผิดพลาดแต่ละชนิดมีผลกระทบต่อระบบมากน้อยแตกต่างกันออกไป เรามีการให้น้ำหนักของความรุนแรงในอัตรา 1 ถึง 5 โดยอัตรา 1 มีผลกระทบน้อย ถ้ามีค่าเท่ากับ 5 คือรุนแรงมากผู้ทดสอบอาจจะตัดสินใจเพื่อหยุดการทดสอบ หรือดำเนินการทดสอบต่อไป หรือ หยุดการทดสอบบางส่วน เพื่อแก้ไขปรับปรุงก่อนก็ได้

8.10

บทสรุป

ในบทนี้เป็นการทดสอบความผิดพลาดที่อาจเกิดขึ้นในระบบ ทั้งที่เป็นหน้าที่ที่ระบบสามารถกระทำได้(functional testing) และไม่ใช่หน้าที่แต่เป็นข้อกำหนดกฎเกณฑ์ที่ต้องการ (Nonfunctional testing) การทดสอบระบบที่สมบูรณ์จบด้วยการทดสอบโดยลูกค้า ณ สถานที่ทำงานจริงโดยมีการติดตั้งระบบเพื่อให้ระบบสามารถกระทำงานได้ตามความต้องการ ผลของการทดสอบที่ประสบความสำเร็จคือความพึงพอใจของลูกค้านั่นเอง

Configuration Management เป็นส่วนสำคัญของกระบวนการทดสอบเพราะเป็นส่วนสำคัญในการเชื่อมโยงถึงการสืบค้นความผิดพลาดโดยอ้างอิงกับความความต้องการที่ลูกค้าต้องการ ถือว่าเป็นเทคนิคที่ช่วยในการควบคุมกระบวนการทดสอบที่ดี

อย่างไรก็ตามมีเครื่องมืออัตโนมัติในการทดสอบเพื่อช่วยทีมงานทดสอบ มีการวางแผนงานการทดสอบอย่างระมัดระวัง มีการสร้างพิมพ์เขียวในการทดสอบระบบโดยภาพรวม และวางแผนการทดสอบในแต่ละฟังก์ชั่น โดยแบ่งการทดสอบออกเป็นรายละเอียดในปลีกย่อยเป็นการทดสอบในส่วนย่อยๆ และมีการบรรยายการทดสอบทีละขั้น(step-by-step) เมื่อเกิดปัญหา

หรือความผิดพลาดเกิดขึ้นมีการบันทึกเป็นรายงานการวิเคราะห์ความผิดพลาด เพื่อใช้ในการแก้ไขในภายหลัง ซึ่งหลังจากขจัดปัญหาต่างๆที่เกิดขึ้นหมดสิ้นแล้ว เราจะนำระบบที่ผ่านการทดสอบนี้ไปส่งมอบต่อลูกค้า เพื่อใช้งานจริงต่อไป

8.11

แบบฝึกหัด

1. การทดสอบระบบ ประกอบด้วยขั้นตอนอะไรบ้าง จงอธิบายพอเข้าใจ
2. Configuration Management คืออะไร มีความสำคัญอย่างไรในกระบวนการทดสอบ
3. Cause and Effect Graph คืออะไร มีความสำคัญอย่างไรในกระบวนการทดสอบ
4. จงเขียน Cause and Effect Graph ของการหาพื้นที่ของสามเหลี่ยมหน้าจั่วใดๆ
5. จงอธิบายถึงการทดสอบต่อไปนี้พอเข้าใจ
   - 5.1 Stress Test
   - 5.2 Volumn Test
   - 5.3 Recovery Test
   - 5.4 Quality Test
   - 5.5 Maintenance Test
6. ลูกค้ามีส่วนสำคัญอย่างไรในการทดสอบระบบ
7. ผู้ใช้งาน ณ สถานที่ติดตั้งระบบ มีส่วนสำคัญอย่างในการทดสอบระบบ
8. ทีมงานในการทดสอบระบบประกอบด้วยใครบ้างมีหน้าที่อย่างไร
9. เอกสารในการทดสอบประกอบด้วยอะไรบ้าง
10. จงสร้าง Test Description ในการหาพื้นที่ของสามเหลี่ยมหน้าจั่วใดๆ
11. จงสร้าง Test Documentation ของโครงงานที่นักศึกษาได้กระทำ