โปรแกรมช่วยวิเคราะห์ข้อมูล Personal Software Process

A program to assist in data analysis for Personal Software Process

นายปิยะพงษ์ ฐานะตระกูล รหัส 50365178

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2553

ใบรับรองปริญญานิพนธ์

หัวข้อโครงงาน โปรแกรมช่วยวิเคราะห์ข้อมูล Personal Software Process

ผู้ดำเนินโครงงาน นายปิยพงษ์ ฐานะตระกูล รหัส 50365178

อาจารย์ที่ปรึกษา ดร.สุรเดช จิตประไพกุลศาล

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2553

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์

..... ประธานกรรมการ

(ดร.สุรเดช จิตประไพกุลศาล)

..... กรรมการ

(อาจารย์ภาณุพงศ์ สอนคม)

..... กรรมการ

(อาจารย์สิรภพ คชรัตน์)

หัวข้อโครงงาน	**โปรแกรมช่วยวิเคราะห์ข้อมูล Personal Software Process**
ผู้ดำเนินโครงงาน	นายปิยะพงษ์ ฐานะตระกูล รหัส 50365178
อาจารย์ที่ปรึกษา	ดร.สุรเดช จิตประไพกุลศาล
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2553

บทคัดย่อ

โครงงานนี้เป็นการพัฒนาโปรแกรมต้นแบบ(Prototype) เพื่อช่วยในการวิเคราะห์ข้อมูลที่ได้จากการบันทึกการทำงานตามแนวคิดของกระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal software process; PSP) โปรแกรมนี้จะแสดงผลการวิเคราะห์สมรรถนะการทำงานส่วนบุคคลของนักพัฒนาโปรแกรมในรูปแบบของตารางและแผนภูมิชนิดต่างๆได้ 4 ด้านดังนี้ การวางแผน กระบวนการคุณภาพ และภาพรวม โดยมีแผนภูมิ 45 แผนภูมิและตาราง 4 ตาราง

โปรแกรมนี้พัฒนาโดยใช้ภาษาจาวา(Java) เจฟรีชาร์ต (JFreeChart) และเอชเอสคิวแอลดีบี (HSQLDB) โปรแกรมนี้จึงสามารถทำงานได้บนทุกระบบปฏิบัติการที่รองรับ Java virtual machine version 1.6 ขึ้นไป

Project Title	A program to assist in data analysis for Personal Software Process		
Name	Mr.Piyapong	Thanatrakul	ID. 50365178
Project Advisor	Dr.Suradet	Jitprapaikulsarn	
Major	Computer Engineering		
Department	Electrical and Computer Engineering		
Academic Year	2010		

.....

ABSTRACT

In this project, we developed a prototypical program to assess the performance of a software developer in four aspects: planning performance, process performance, quality performance, and overall performance. There are 45 diagrams and 4 tables presented in this program: 11 for planning, 7 process, 30 quality, and 1 overall.

This program was developed using Java programming language together with JFreeChart and HSQLDB. This program should be able to run or any operating system supporting Java virtual machine version 1.6.

กิตติกรรมประกาศ

โครงงานวิศวกรรมคอมพิวเตอร์ฉบับนี้สำเร็จลุล่วงมาได้นั้น เนื่องจากความอนุเคราะห์จากท่านอาจารย์ที่ปรึกษาโครงงาน ดร.สุรเดช จิตประไพกุลศาล ที่กรุณาสละเวลาให้คำแนะนำในการทำงานตลอดถึงการตรวจสอบการทำงานพร้อมทั้งเสนอแนวทางการแก้ไขปัญหาตลอดระยะเวลาการทำโครงงาน พร้อมทั้งให้คำแนะนำที่เป็นประโยชน์ทำให้การทำโครงงานเป็นไปอย่างราบรื่น

ทั้งนี้ต้องขอขอบพระคุณกรรมการทั้งสองท่านอันได้แก่อาจารย์ภาณุพงศ์ สอนคม และอาจารย์สิรภพ คชรัตน์ อาจารย์ประจำภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ มหาวิทยาลัยนเรศวร ที่เสียสละเวลาอันมีค่าให้ปรึกษาและแนะแนวทางในการแก้ปัญหาต่างๆ

สุดท้ายนี้ผู้จัดทำต้องขอขอบพระคุณ บิดา มารดา และอาจารย์ทุกท่าน ที่คอยสั่งสอนให้ความรู้จนผู้จัดทำสำเร็จการศึกษา และขอขอบคุณเพื่อนๆที่คอยให้กำลังใจ ช่วยให้คำปรึกษาทั้งในเรื่องเรียนเรื่องส่วนตัวจนสำเร็จลุล่วงมาได้ด้วยดี

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

ตาราง หน้า

สารบัญรูป

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

เนื่องจากโปรแกรม PSP Student Workbook และ PSP Dashboard ที่ใช้เป็นเครื่องมือที่ช่วยในการวิเคราะห์ข้อมูลที่ได้บันทึกผลตามแนวคิดของกระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal Software Process; PSP) พบว่ายังมีปัญหาในการใช้งานและยังขาดคุณสมบัติในการใช้งาน เช่น มีความซับซ้อนในการใช้งานสูง ขาดความยืดหยุ่นในการใช้งาน เป็นต้น ดังนั้นโปรแกรมที่จะพัฒนาใหม่จะแก้ไขข้อบกพร่องเพื่อให้ใช้งานได้ง่ายรวมถึงเพิ่มคุณสมบัติใหม่ๆที่จะช่วยวิเคราะห์และประเมินคุณภาพของผู้พัฒนาซอฟต์แวร์ได้เด่นชัดขึ้น [1]

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อสร้างชุดเครื่องมือที่ช่วยในการวิเคราะห์ข้อมูล Personal Software Process โดยให้ครอบคลุมการวิเคราะห์ข้อมูลที่มีอยู่ในโปรแกรม PSP Student และ PSP Dashboard

1.3 ขอบเขตของโครงงาน

1. สร้างโปรแกรมสำหรับช่วยวิเคราะห์และประเมินประสิทธิภาพของผู้พัฒนาโปรแกรม
2. สร้างเครื่องมือวิเคราะห์ข้อมูล Personal Software Process ซึ่งแบ่ง 4 ด้าน ดังนี้
    1. Plan Performance
        - Actual Development Time
        - Actual Size
        - Percent Compile + Test Time
        - Percent Compile Time
        - Percent Planning + Postmortem Time
        - Percent Planning Time
        - Percent Postmortem Time

- Percent Test Time
- Percent Time in Phase To Date
- Size Estimating Error
- Time Estimating Error

2. Process Performance
   - A/FR vs Yield
   - Productivity
   - Productivity vs A/FR
   - Productivity vs Yield
   - Test Defect vs A/FR
   - Test Defect vs Yield
   - Yield vs A/FR
3. Quality Performance
   - Appraisal Cost of Quality
   - Appraisal to Failure Ratio
   - Compile vs Test Defects
   - CR Review Rate vs Yield
   - Defect Age
   - Defect Age (Chart)
   - Defect Fix Time by Type
   - Defect Injection % by Phase
   - Defect Removal Yield
   - Defects Injected by Phase To Date
   - Defects Injected in Code
   - Defects Injected in Design
   - Defects Removal % by Phase
   - Defects Removal Leverage
   - Defects Removed by Phase To Date

- Defects Removed by Type
- Defects Removed in Code Review
- Defects Removed in Compile
- Defects Removed in Design Review
- Defects Removed in Test
- DLDR Review Rate vs Yield
- Failure Cost of Quality
- PSP Defect Density Report
- PSP Defect Fix Time Report
- PSP Percent Injected and Removed by Type Report
- PSP Percentage Defects Found by Compile Report
- Review Rate
- Review Rate vs Yield
- Total Cost of Quality
- Total Defects

4. Overall
   - Process Quality Index

1.4 ขั้นตอนการดำเนินงาน

1. ศึกษาความรู้เกี่ยวกับ Personal Software Process
2. ศึกษาเครื่องมือการวิเคราะห์ข้อมูลของ PSP Student
3. ศึกษาเครื่องมือการวิเคราะห์ข้อมูลของ PSP Dashboard
4. ศึกษาการสร้างแผนภูมิชนิดต่างๆ
5. ศึกษาโครงสร้างฐานข้อมูลที่ใช้ในการวิเคราะห์ข้อมูล
6. ออกแบบโครงสร้างของโปรแกรม
7. สร้างเครื่องวิเคราะห์ข้อมูลให้ได้ตามจุดประสงค์
8. ทดสอบความถูกต้องของเครื่องมือวิเคราะห์ข้อมูล
9. จัดทำเอกสารและคู่มือการใช้

1.5 ผลที่คาดว่าจะได้รับ

1. ได้เครื่องมือวิเคราะห์ข้อมูลสมรรถนะ 4 ด้าน ดังนี้ การวางแผน กระบวนการ คุณภาพและภาพรวม
2. เพื่อช่วยให้การวิเคราะห์ข้อมูล PSP มีความสะดวกมากขึ้น

1.6 แผนการดำเนินงาน

ตารางที่ 1.1 แผนการดำเนินการ

กิจกรรม	ปี 2553					ปี 2554		
	ส.ค	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.
1. ศึกษาความรู้ทั่วไปเกี่ยวกับ Personal Software Process								
2. ศึกษาเครื่องมือการวิเคราะห์ข้อมูลของ PSP Student								
3. ศึกษาเครื่องมือการวิเคราะห์ข้อมูลของ PSP Dashboard								
4. ศึกษาการสร้างแผนภูมิชนิดต่างๆ								
5. ศึกษาโครงสร้างฐานข้อมูลที่ใช้ในการวิเคราะห์ข้อมูล								
6. ออกแบบโครงสร้างของโปรแกรม								
7. สร้างเครื่องวิเคราะห์ข้อมูลให้ได้ตามจุดประสงค์								
8. ทดสอบความถูกต้องของเครื่องมือวิเคราะห์ข้อมูล								
9. จัดทำเอกสารและคู่มือการใช้								

1.7 รายละเอียดงบประมาณ

1.	ค่าเอกสาร	500	บาท
2.	ค่าจัดทำรูปเล่มรายงาน	500	บาท
	รวมเป็นเงินทั้งสิ้น	1000	บาท (หนึ่งพันบาทถ้วน)

บทที่ 2
ความรู้เบื้องต้นเกี่ยวกับ Personal Software Process

บทนี้จะกล่าวถึงหลักการเบื้องต้นเกี่ยวกับกระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal Software Process; PSP) และเครื่องช่วยวิเคราะห์ข้อมูลที่จะทำการพัฒนาในแต่ละด้าน ดังนี้ การวางแผน กระบวนการ คุณภาพ และภาพรวม

2.1 กระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal Software Process; PSP)

PSP เป็นการบวนการปรับปรุงตนเองสำหรับการพัฒนาซอฟต์แวร์ส่วนบุคคล PSP พัฒนาโดย วัตส์ ฮัมฟรี่ (Watts Humphrey) แห่งสถาบันวิศวกรรมซอฟต์แวร์ (Software Engineering Institute; SEI) มหาวิทยาลัยคาร์เนกี้ เมลรอน เมืองพิตส์เบิร์ก มลรัฐเพนซิลเวเนีย ประเทศสหรัฐอเมริกา [1]

วัตถุประสงค์ของ PSP คือ ช่วยปรับปรุงทักษะด้านวิศวกรรมซอฟต์แวร์ของนักพัฒนาซอฟต์แวร์ PSP จะช่วยในด้านการบริหารงานของตนเอง ประเมินความสามารถของตนเอง สร้างเสริมทักษะของตนเองในด้านต่างๆ เช่น การวางแผน การตรวจวัดคุณภาพการทำงาน และการบริหารกระบวนการทำงาน

PSP มีส่วนคล้ายกับ Capability Maturity Model (CMM) ของ SEI ในเรื่องของหลักการปรับปรุงกระบวนการ โดย CMM มุ่งเน้นการพัฒนาความสามารถในระดับองค์กร ส่วน PSP มุ่งเน้นการพัฒนาคุณภาพของวิศวกรเป็นรายบุคคล

ผู้ใช้กระบวนการ PSP ในการทำงานจะเก็บข้อมูลที่เกี่ยวข้องกับการทำงาน 4 ด้านดังนี้

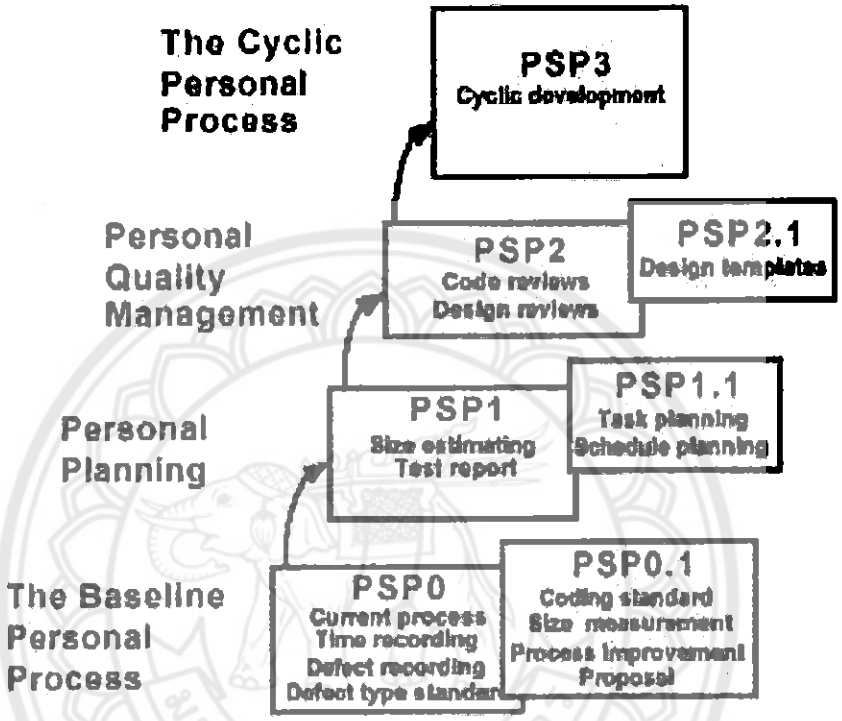
1. แรงงานที่ใช้ในการทำงาน (Effort) ในรูปของเวลาที่ใช้
2. ขนาดของงานที่ทำ (Size) ซึ่งโดยปกติจะนิยมวัดในรูปของจำนวนบรรทัดของโค้ด (Line of Code; LOC)
3. คุณภาพของงานที่ทำ (Quality) ในรูปของจำนวนข้อบกพร่องที่เกิดขึ้น (Defect)
4. กำหนดการ (Schedule)

หลักพื้นฐานของ PSP ของการวางแผนและคุณภาพ [1]

1. **คุณภาพของระบบซอฟต์แวร์จะพิจารณาจากคุณภาพขององค์ประกอบของโปรแกรม (Components) ที่มีคุณภาพต่ำที่สุด**
2. **คุณภาพขององค์ประกอบของโปรแกรม กำหนดโดยผู้พัฒนาซอฟต์แวร์**
3. **คุณภาพขององค์ประกอบของโปรแกรม กำหนดโดยกระบวนการพัฒนาซอฟต์แวร์**
4. **หัวใจของคุณภาพขององค์ประกอบของโปรแกรม คือ ทักษะ การยอมรับ และความรับผิดชอบของผู้พัฒนาซอฟต์แวร์**
5. ความแตกต่างกันระหว่างวิศวกรแต่ละบุคคล จะส่งผลต่อการทำงานอย่างมาก **ดังนั้นวิศวกรจึงต้องวางแผนงานอยู่บนข้อมูลการทำงานส่วนบุคคล**
6. เพื่อให้สมรรถนะในการทำงานเป็นไปอย่างสม่ำเสมอ วิศวกรควรใช้**กระบวนการที่ถูก**กำหนด นิยามไว้อย่างดี และสามารถวัดได้
7. **เพื่อที่จะสร้างผลิตภัณฑ์ที่มีคุณภาพ** วิศวกรต้องมีความรู้สึกรับผิดชอบ**ต่อคุณภาพของ**ผลิตภัณฑ์ที่สร้างขึ้น ผลิตภัณฑ์ที่มีคุณภาพชั้นเยี่ยมจะไม่ถูกสร้างโดยความ**ผิดพลาด วิศวกร**จะต้องมีความพยายามอย่างมากเพื่อการรักษาคุณภาพของงาน
8. การแก้ไขข้อบกพร่องในกระบวนการทำงาน เป็นการลดค่าใช้จ่ายในการทำงาน
9. ประสิทธิภาพการทำงานจะมากขึ้น หากมีการป้องกันข้อบกพร่องที่อาจเกิด**ขึ้นมากกว่าการ**ค้นหาแล้วแก้ไขข้อบกพร่องนั้นๆ
10. ทางที่ดีที่สุดสำหรับการทำงาน คือ การทำงานที่รวดเร็วและประหยัดค่าใช้จ่าย**ที่สุด**

เพื่อให้งานของวิศวกรรมซอฟต์แวร์เป็นไปในทางที่ถูก วิศวกรต้องวางแผน**ก่อนที่จะรับทำสัญญา** หรือเริ่มการทำงาน และจะต้องกำหนดกระบวนการไว้เป็นแผนงาน เพื่อให้เข้าใจถึงสมรรถภาพในการทำงานเป็นรายบุคคล วิศวกรต้องบันทึกเวลาที่ใช้ในแต่ละขั้นตอนของงาน จำนวน**ข้อบกพร่องที่พบ และทำการแก้ไข ขนาดของซอฟต์แวร์ที่สร้าง เพื่อให้คุณภาพของผลิตภัณฑ์เป็นไปอย่างสม่ำเสมอ** วิศวกรต้องวางแผน บันทึก และติดตามคุณภาพของผลิตภัณฑ์ และจะต้องมุ่งเน้นที่คุณภาพตั้งแต่เริ่มโครงการ ในที่สุดแล้ว ต้องมีการวิเคราะห์ผลของงานแต่ละชิ้น และใช้ผลนี้ในการค้นหาวิธีการปรับปรุงกระบวนการของวิศวกรผู้นั้นต่อไป

2.2 ระดับของกระบวนการ PSP (PSP Process Evolution) [2]



รูปที่ 2.1 PSP Process Evolution

ที่มา: [2]

PSP ได้กำหนดกรอบขั้นตอนการปรับปรุงกระบวนการพัฒนาไว้ 4 ระดับ คือ

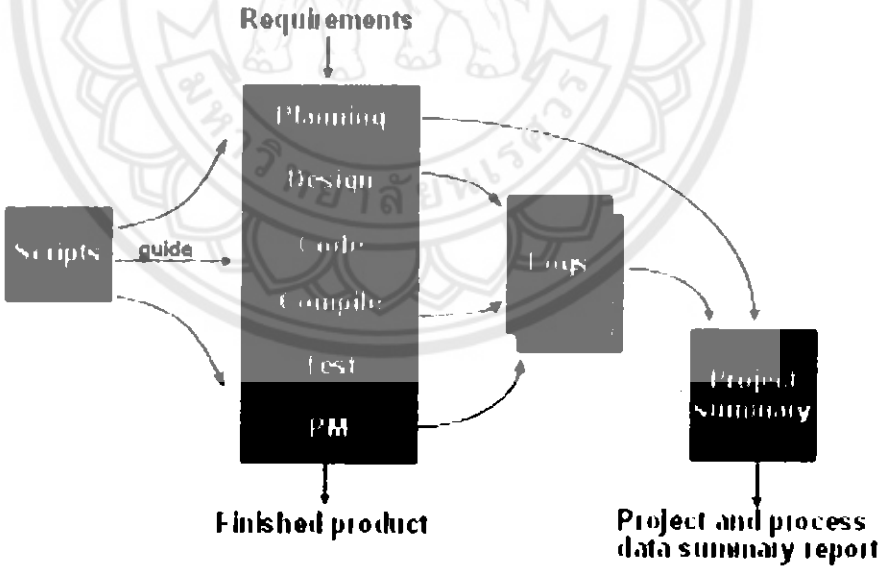
1. PSP0: Baseline Personal Process
2. PSP1: Personal Planning Process
3. PSP2: Personal Quality Management
4. PSP3: Cycle Personal Process

2.2.1 PSP0: Baseline Personal Process

เป็นกระบวนการสร้างพื้นฐานในการวัดข้อมูล ได้แก่ ขนาดของโปรแกรม เวลา และข้อบกพร่องซึ่งเป็นข้อมูลที่ใช้พัฒนาซอฟต์แวร์ของวิศวกรในรูปแบบของแต่ละบุคคล ซึ่งมีวัฏจักรในการพัฒนาซอฟต์แวร์ด้วยกัน 6 ขั้นตอน ประกอบด้วย

1. วางแผน (Planning)
2. ออกแบบ (Design)
3. เขียนโปรแกรม (Code)
4. แปลโปรแกรม (Compile)
5. ทดสอบ (Test)
6. ส่งมอบงาน (Postmortem)

ใน PSP0 นั้นมีกระบวนการย่อยอีกหนึ่งกระบวนการ คือ PSP0.1 เพื่อช่วยในการบันทึกข้อเสนอการปรับปรุงกระบวนการพัฒนา



รูปที่ 2.2 PSP0 Process Flow

ที่มา: [2]

PSP0 เป็นการแนะนำการวัดในกระบวนการขั้นพื้นฐาน และการวางแผน เวลาที่ใช้ในการพัฒนา ข้อบกพร่องที่ปรากฏ และขนาดของโปรแกรมซึ่งถูกวัดและบันทึกบนฟอร์มเอกสารที่กำหนดไว้ให้ แผนงานที่วิศวกรวางไว้ก่อนการพัฒนา และผลสรุปสุดท้ายจะถูกเก็บไว้ในเอกสาร Project Summary และในขั้นตอนของ PSP0.1 ได้แนะนำรูปแบบเอกสารที่ให้บันทึกชื่อ Process Improvement Proposals (PIPS) ไว้สำหรับบันทึกแนวความคิดที่ต้องการใช้เพื่อพัฒนากระบวนการด้วย

ข้อมูลสรุปของโครงงาน (Project Summary Data)

ข้อมูลสรุปของโครงการถูกบันทึกใน Project Plan Summary Form รูปแบบนี้จัดเตรียมเพื่อความสะดวกในการสรุปการวางแผน และค่าที่ได้จากการวัดขนาดของโปรแกรม เวลาที่ใช้ในการพัฒนา และจำนวนบกพร่อง ซึ่งแบ่งออกเป็น 4 ส่วน ได้แก่ Program Size, Time in Phase, Defects Injected และ Defects Remove

ข้อมูลของ PSP Project Plan Summary Form สามารถนำไปใช้ประโยชน์ได้จริงสำหรับวิศวกรซอฟต์แวร์ ข้อมูลเหล่านี้สามารถนำไปใช้ติดตามการทำงานในโครงการที่กำลังทำอยู่ในปัจจุบัน หรือถูกนำมาใช้ในฐานะที่เป็นข้อมูลทางสถิติเพื่อใช้ในการวางแผนโครงการในอนาคต และใช้เป็นข้อมูลพื้นฐาน (baseline) เพื่อการประเมินความก้าวหน้าของงานได้

2.2.2 PSP1: Personal Planning Process

PSP1 และ PSP1.1 เน้นเทคนิคการจัดการโครงการส่วนบุคคล ได้แนะนำการประมาณขนาดของโปรแกรมและแรงงาน (effort) ที่คาดว่าจะใช้ล่วงหน้า การวางแผนตารางเวลา และวิธีการติดตามผลในผลตารางเวลาที่วางไว้ (schedule tracking) การประมาณขนาดของโปรแกรมและแรงงานโดยใช้วิธี PROBE (Proxy-Based Estimation) เป็นการอาศัยหลักความสัมพันธ์จากการประมาณค่าตัวแทน (proxy) เช่น จำนวนของวัตถุ (object) จำนวนฟังก์ชัน จำนวนกระบวนคำสั่ง (procedures) เพื่อใช้ในการประมาณขนาดของโปรแกรมเบื้องต้น จากนั้นใช้ค่าข้อมูลที่เคยเก็บไว้ (Historical data) ในการแปลงค่าจากขนาดของ proxy ไปเป็นขนาดของ LOC (Line of Code)

PROBE เป็นกระบวนการประมาณที่ใช้ใน PSP เพื่อประมาณขนาดและแรงงาน เป็นวิธีการประมาณค่าที่แนะนำโดย Watts Humphrey ซึ่งอยู่บนแนวคิดที่ว่า "ถ้าวิศวกรสร้างส่วนประกอบที่คล้ายกับวิศวกรคนก่อนแล้วแรงงานที่ใช้ก็จะเท่ากับสิ่งที่วิศวกรเคยทำมาก่อน" วิศวกรแต่ละคนจะใช้ฐานข้อมูลเพื่อติดตามขนาดและแรงงานของทุกงานที่พัฒนา โดยแบ่งส่วนประกอบต่างๆเป็นส่วนเล็กๆ โดยที่ทุกๆส่วนประกอบในฐานข้อมูลจะถูกระบุประเภท ("calculation", "data", "logic" etc.) และขนาด(จากเล็กไปใหญ่) เมื่อมีโครงการใหม่จะต้องมีการประมาณ ซึ่งจะแบ่งออกตามงานที่สอดคล้องกับประเภทและขนาด อ้างอิงโดยการใช้การคำนวณการประมาณการของแต่ละงาน

Line of Code (LOC) คือ หน่วยการวัดขนาดของซอฟต์แวร์โดยใช้จำนวนบรรทัดเป็นวิธีการที่ใช้ในการวัดที่ง่ายที่สุดโดยการนับจำนวนบรรทัดที่พัฒนาซอฟต์แวร์ได้ Line of Code (LOC) หรือ Kilo Line of Code (KLOC) ในบางครั้งอาจจะใช้ Kilo of Delivered Source Instruction (KDSI) ซึ่งได้มีการสรุปลักษณะการวัด LOC ดังนี้

1. นับเฉพาะบรรทัดที่มีการจัดส่งเป็น Source Code ไม่นับรวมส่วนของการทดสอบ
2. นับเฉพาะส่วนที่พัฒนาโดยบุคลากร ไม่รวมส่วนที่ระบบสามารถสร้างได้อัตโนมัติ
3. กำหนดให้หนึ่งคำสั่ง คือ หนึ่ง LOC
4. นับส่วนประกาศค่า (Declaration) เป็นส่วนของคำสั่ง (Instruction)
5. ไม่นับส่วนขยาย (Comment)

2.2.3 PSP2: Personal Quality Management

ในขั้นตอนของ PSP2 ได้เพิ่มการออกแบบส่วนบุคคล (personal design) และการทบทวนชุดคำสั่ง (code review) การทบทวนนี้ช่วยให้วิศวกรซอฟต์แวร์พบข้อบกพร่องได้เร็วขึ้นในกระบวนการและเห็นคุณค่าของการค้นพบข้อบกพร่อง การวิเคราะห์ข้อบกพร่องที่พบและใช้เป็นข้อมูลในการสร้างรายการตรวจสอบที่ใช้เพื่อการทบทวน (review checklist) ข้อดีของการตรวจสอบที่ได้จากประสบการณ์ที่ผ่านมาของแต่ละบุคคล ทำให้ค้นพบข้อบกพร่องที่มักจะเกิดขึ้นโดยไม่แตกต่างกัน

การทบทวนมีประสิทธิผลสำหรับการกำจัดข้อบกพร่องที่มักพบในขั้นตอนการแปลโปรแกรม และข้อบกพร่องที่ส่วนมากพบที่ขั้นตอนการทดสอบ แต่การลดข้อบกพร่องในขั้นตอนการทดสอบที่สำคัญเกิดจากการออกแบบที่มีคุณภาพ PSP2.1 จึงชี้ให้เห็นถึงความจำเป็นของการออกแบบโดยเพิ่มส่วนของ design notation ให้มีแม่แบบ (template) ของการออกแบบสิ่

แบบ และวิธีการทบทวนการออกแบบ (design review methods) โดยความตั้งใจนี้ไม่ใช่เพื่อแนะนำวิธีการออกแบบแบบใหม่แต่เพื่อให้แน่ใจว่านักออกแบบได้ไตร่ตรอง และทำเอกสารการออกแบบในมุมมองที่แตกต่างกันสี่ด้าน ได้แก่

1. ข้อกำหนดทางการปฏิบัติ (operational specification)
2. ข้อกำหนดทางฟังก์ชัน (functional specification)
3. ข้อกำหนดทางสถานะ (state specification)
4. ข้อกำหนดทางตรรกะ (logic specification)

2.2.4 PSP3: Cycle Personal Process

PSP3 ชี้ให้เห็นถึงขนาดของโครงการที่เหมาะสมสำหรับผู้พัฒนาคนหนึ่งที่สามารถทำได้โดยไม่สูญเสียคุณภาพของงาน หรือแม้แต่กำลังการผลิต (productivity) วิศวกรซอฟต์แวร์จะเรียนรู้ถึงประสิทธิภาพของการผลิตที่สามารถทำได้ในช่วงของขนาดโครงการที่เล็กที่สุดจนถึงขนาดของโครงการที่ใหญ่ที่สุด ณ จุดที่ประสิทธิภาพเริ่มลดลงก็คือจุดที่บอกถึงขนาดของโครงการที่สูงสุดที่ผู้พัฒนาจะรับได้ PSP3 ได้แนะนำกลยุทธ์ในการพัฒนาเป็นวงรอบ (a cyclic development strategy) ซึ่งโปรแกรมขนาดใหญ่จะถูกแบ่งเป็นส่วนๆในแต่ละวงรอบในการพัฒนาตามความสามารถของนักพัฒนาเอง และทำส่วนที่เหลือในวงรอบถัดไป และเพื่อเป็นการสนับสนุนแนวคิดนี้ PSP3 ยังได้แนะนำ high-level design, high-level design review, cycle planning และ development cycles บนพื้นฐานของกระบวนการ PSP2.1 รูปแบบเอกสารใหม่สองรูปแบบได้ถูกแนะนำ ได้แก่ เอกสารสรุปในแต่ละวงรอบซึ่งบันทึกผลสรุปของขนาด เวลาที่ใช้ และข้อบกพร่องที่ค้นพบในแต่ละวงรอบ และเอกสารที่ติดตามประเด็นปัญหาสำคัญ (issue tracking log) ที่อาจส่งผลต่อวงรอบในอนาคต หรือเมื่อพัฒนาทุกวงรอบอย่างสมบูรณ์แล้ว

2.3 ตัวชี้วัดใน PSP (PSP Measures)

ตัวชี้วัดพื้นฐาน (Base or Explicit Measures) ใน PSP ได้แก่

1. แรงงานที่ใช้ในการทำงาน (Effort) ในรูปของเวลาที่ใช้
2. ขนาดของงานที่ทำ (Size) ซึ่งโดยปกติจะนิยมวัดในรูปของจำนวนบรรทัดของโค้ด (Line of Code; LOC)
3. คุณภาพของงานที่ทำ (Quality) ในรูปของจำนวนข้อบกพร่องที่เกิดขึ้น (Defect)
4. กำหนดการ (Schedule)

ส่วนตัวชี้วัดอื่นๆ ประเภทตัวชี้วัดอนุพัทธ์ (Derived measure) ที่เกิดจากการนำตัวชี้วัดพื้นฐานจากการวัดพื้นฐานมาคำนวณ

2.3.1 เวลาที่ใช้พัฒนา (Development time)

เวลาที่ใช้พัฒนา จะทำการวัดซึ่งมีหน่วยเป็นนาที โดยจะต้องมีการเก็บบันทึกเวลาทั้งหมดที่ใช้ในแต่ละช่วงการทำงานของ PSP ตั้งแต่เริ่มกระบวนการจนกระทั่งจบการทำงานในช่วงการทำงานนั้น ลบด้วยเวลาที่ถูกขัดจังหวะการทำงาน เก็บบันทึกในทุกๆช่วงการทำงานของ PSP

2.3.2 ข้อบกพร่องที่เกิดขึ้น (Defects)

ข้อบกพร่อง (Defect) ถูกนิยามไว้ว่า "การเปลี่ยนแปลงใดๆจะต้องถูกทำบนการออกแบบ (Design) หรือการเขียนโปรแกรม (Coding) เพื่อให้การแปลโปรแกรม (Compile) หรือการทดสอบ (Testing) เป็นไปอย่างถูกต้อง"

ข้อบกพร่องจะถูกบันทึกไว้ใน Defect Recoding Log เมื่อถูกค้นพบและได้รับการแก้ไข ซึ่งประกอบด้วย

1. วันที่ที่พบ (Data)
2. ลำดับการเกิด (Number)
3. ประเภทข้อบกพร่อง (Type)
4. ช่วงการทำงานที่เกิดข้อบกพร่อง (Inject)
5. ช่วงการทำงานที่ข้อบกพร่องได้รับการแก้ไข (Remove)

6. เวลาที่ได้รับการแก้ไข (Fix Time)
7. ข้อบกพร่องที่เกิดจากข้อบกพร่องเก่า (Fix defect)
8. คำอธิบายปัญหาและการแก้ไข (Description)

ตารางที่ 2.1 Defect Type Standard

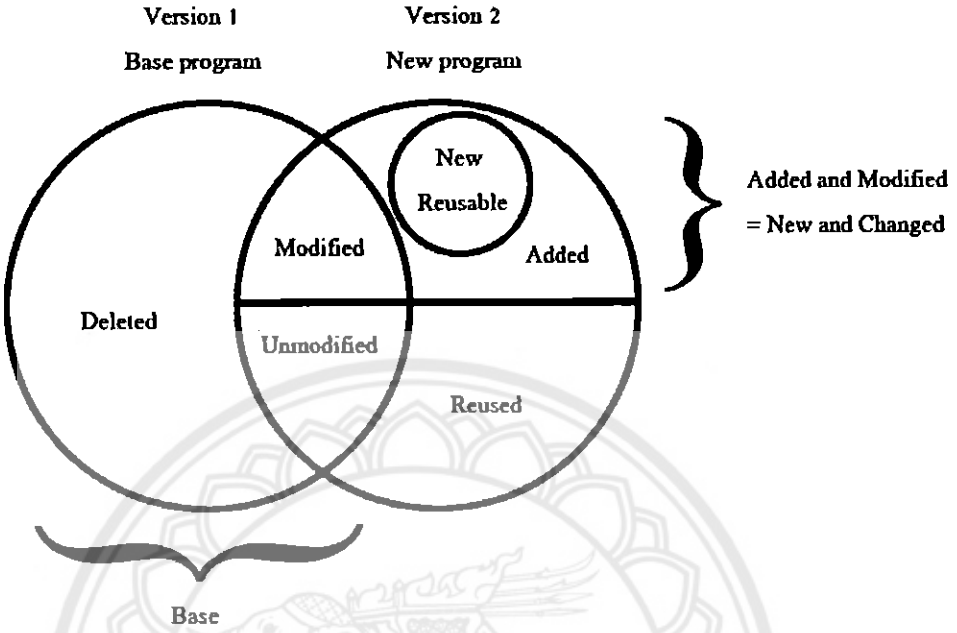
Type Number	Type Name	Description
10	Documentation	Comment, messages
20	Syntax	Spelling, punctuation, typos, instruction formats
30	Build, Package	Change management, library, version control
40	Assignment	Declaration, duplicate name, scope, limits
50	Interface	Procedure calls and references, I/O, user formats
60	Checking	Error messages, inadequate check
70	Data	Structure, content
80	Function	Logic, pointers, loops, recursion, computation, function defects
90	System	Configuration, timing, memory
100	Environment	Design, compile, test, or other support system problems

ที่มา: [2]

2.3.3 ขนาดของโปรแกรม (size)

วัดถุประสงค์เบื้องต้นของการวัดขนาดโปรแกรม เพื่อประมาณเวลาที่ใช้ในการพัฒนาทั้งหมด โดยการนับจำนวนบรรทัดของโปรแกรม (Line of Code) เพราะ

1. สามารถนับได้อย่างอัตโนมัติ
2. สามารถนิยามวิธีวัดได้อย่างเที่ยงตรง
3. มีความสัมพันธ์เป็นอย่างดีกับงานวิจัยทางด้าน PSP ในเรื่องการพัฒนากำลังผลิต
4. ขนาดของโปรแกรมถูกใช้เพื่อให้เป็นบรรทัดฐานของข้อมูลประเภทอื่นๆ



รูปที่ 2.3 Version size LOC

ที่มา: [2]

ตารางที่ 2.2 ประเภทของขนาด LOC

Type of LOC	Definition
Base	LOC from a previous version
Deleted	Deletions from the Base LOC
Modified	Modification to the Base LOC
Added	New objects, functions, procedures, or any other added LOC
Reused	LOC from a previous program that is used without modification
New and Changed	The sum of Added and Modified LOC
Total LOC	The total program LOC
Total New Reused	New or added LOC that were written be reusable

ที่มา: [2]

2.3.4 กำหนดการ (Schedule)

กำหนดการ คือ รายการแสดงเวลาที่เหตุการณ์ต่างๆจะเกิดขึ้น เช่น เวลาเริ่มต้น เวลาที่จะเขียนโปรแกรม เวลาสิ้นสุด เป็นต้น

2.4 เครื่องมือที่ใช้ในการวิเคราะห์ข้อมูล (PSP Analysis Tools)

แบ่งออกเป็นสมรรถนะได้ 4 ด้าน ดังนี้

1. สมรรถนะด้านการวางแผน (Plan Performance)
2. สมรรถนะด้านกระบวนการ (Process Performance)
3. สมรรถนะด้านคุณภาพ (Quality Performance)
4. ภาพรวม (Overall)

2.4.1 สมรรถนะด้านการวางแผน (Plan Performance)

1. Actual Development Time
   - เป็นแผนภูมิเส้นแสดงเวลาที่ใช้จริงในการพัฒนาโปรแกรม
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Time Hour} = \frac{\text{ActMin}}{60}$$

(สมการที่ 2.1)

2. Actual Size
   - เป็นแผนภูมิเส้นแสดงถึงจำนวนขนาดที่เกิดขึ้นจริงของแต่ละโปรแกรม
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Actual Size} = \text{Add Size} + \text{Modify Size}$$

(สมการที่ 2.2)

3. Percent Compile + Test Time
   - แผนภูมิเส้นแสดงเปอร์เซ็นต์เวลาที่ใช้ในการแปลโปรแกรมและการทดสอบโปรแกรม
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Percent Compile and Test Time} = 100 \times \left\{\frac{(\text{Compile Time} + \text{Test Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.3)

4. Percent Compile Time
   - เป็นแผนภูมิเส้นแสดงเปอร์เซ็นต์เวลาที่ใช้ในการแปลโปรแกรม
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Percent Compile Time} = 100 \times \left(\frac{\text{Compile Time}}{\text{ActMin}}\right)$$

(สมการที่ 2.4)

5. Percent Planning + Postmortem Time
   - เป็นแผนภูมิเส้นแสดงเปอร์เซ็นต์เวลาที่ใช้ในการวางแผนงานและจัดส่งงาน
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Percent Planning and Postmortem Time} = 100 \times \left(\frac{\text{Postmortem Time}}{\text{ActMin}}\right)$$

(สมการที่ 2.5)

6. Percent Planning Time
   - เป็นแผนภูมิเส้นแสดงเปอร์เซ็นต์เวลาที่ใช้ในการวางแผน
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Percent Planning Time} = 100 \times \left(\frac{\text{Planning Time}}{\text{ActMin}}\right)$$

(สมการที่ 2.6)

7. Percent Postmortem Time

- เป็นแผนภูมิเส้นแสดงเปอร์เซ็นต์เวลาที่ใช้ในการวจัดส่งงาน
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Percent Postmortem Time} = 100 \times \left(\frac{\text{Postmortem Time}}{\text{ActMin}}\right)$$

(สมการที่ 2.7)

8. Percent Test Time

- แผนภูมิเส้นแสดงเปอร์เซ็นต์เวลาที่ใช้ในการทดสอบโปรแกรม
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Percent Test Time} = 100 \times \left(\frac{\text{Test Time}}{\text{ActMin}}\right)$$

(สมการที่ 2.8)

9. Percent Time in Phase To Date

- แผนภูมิวงกลมแสดงเปอร์เซ็นต์เวลาสะสมที่ใช้ในแต่ละช่วงการทำงาน
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{To Date \%} = \frac{(\text{To Date} \times 100)}{\text{sum of all To Date}}$$

(สมการที่ 2.9)

คิดแต่ละเช่วงการทำงาน โดยที่

To Date % คือ เปอร์เซ็นต์ To Date

To Date คือ ค่ารวมในแต่ละช่วงการทำงาน

10. Size Estimating Error

- เป็นแผนภูมิเส้นแสดงค่าผิดพลาดในการประเมินขนาดของโปรแกรม
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Size Estimating Error} = 100 \times \left\{\frac{(\text{ActLOC} - \text{EstLOC})}{\text{EstLOC}}\right\}$$

(สมการที่ 2.10)

11. Time Estimating Error

- เป็นแผนภูมิเส้นแสดงค่าผิดพลาดในการประเมินเวลาที่ใช้พัฒนาโปรแกรม
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Time Estimating Error} = 100 \times \left\{\frac{(\text{ActMin} - \text{EstMin})}{\text{ActMin}}\right\}$$

(สมการที่ 2.11)

2.4.2 สมรรถนะด้านกระบวนการ (Process Performance)

1. A/FR vs Yield

- แผนภูมิเส้นแสดงอัตราส่วนระหว่างการประเมินค่ากับค่าความผิดพลาดต่อเปอร์เซ็นต์ข้อบกพร่อง
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Productivity} = \left(\frac{\text{ActLOC}}{\text{ActMin}}\right) \times 60$$

(สมการที่ 2.12)

$$\text{Appraisal COQ} = 100 \times \left\{\frac{(\text{Design Review Time} + \text{Code Review Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.13)

$$\text{Failure COQ} = 100 \times \left\{\frac{(\text{Compile Time} + \text{Test Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.14)

$$\text{A/FA} = \frac{\text{Appraisal COQ}}{\text{Failure COQ}}$$

(สมการที่ 2.15)

$$\begin{aligned}\text{Early Inject} = {} & \text{Defect Injected Planning} + \text{Defect Injected Design} \\ & + \text{Defect Injected Design Review} + \text{Defect Injected Code} \\ & + \text{Defect Injected Code Review Time)}\end{aligned}$$

(สมการที่ 2.16)

$$\begin{aligned}\text{Early Remove} = {} & \text{Defect Removed Planning} + \text{Defect Removed Design} + \\ & \text{Defect Removed Design Review} + \text{Defect Removed Code} \\ & + \text{Defect Removed Code Review}\end{aligned}$$

(สมการที่ 2.17)

$$\text{Yield\%} = 100 \times \left(\frac{\text{Early Remove}}{\text{Early Inject}}\right)$$

(สมการที่ 2.18)

2. Productivity
   - แผนภูมิเส้นแสดงความสามารถในการผลิต
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Productivity} = \left(\frac{\text{ActLOC}}{\text{ActMin}}\right) \times 60$$

(สมการที่ 2.19)

3. Productivity vs A/FR
   - แผนภูมิเส้นแสดงความสามารถในการผลิตต่ออัตราส่วนระหว่างการประเมินค่ากับค่าความผิดพลาด
   - สูตรที่ใช้ในการคำนวณ คือ

$$\text{Productivity} = \left(\frac{\text{ActLOC}}{\text{ActMin}}\right) \times 60$$

(สมการที่ 2.20)

$$\text{Appraisal COQ} = 100 * \left\{\frac{(\text{Design Review Time} + \text{Code Review Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.21)

$$\text{Failure COQ} = 100 * \left\{\frac{(\text{Compile Time} + \text{Test Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.22)

$$\text{A/FA} = \frac{\text{Appraisal COQ}}{\text{Failure COQ}}$$

(สมการที่ 2.23)

4. Productivity vs Yield

- แผนภูมิเส้นแสดงความสามารถในการผลิตต่อเปอร์เซ็นต์ข้อบกพร่อง
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Productivity} = \left(\frac{\text{ActLOC}}{\text{ActMin}}\right) \times 60$$

(สมการที่ 2.24)

$$\begin{aligned}\text{Early Inject} = {} & \text{Defect Inject Planning Time} + \text{Defect Inject Design Time} \\ & + \text{Defect Inject Design Review Time} + \text{Defect Inject Code} \\ & + \text{Defect Inject Code Review Time}\end{aligned}$$

(สมการที่ 2.25)

$$\begin{aligned}\text{Early Remove} = {} & \text{Defect Remove Planning} + \text{Defect Remove Design} + \\ & \text{Defect Remove Design Review} + \text{Defect Remove Code} \\ & + \text{Defect Remove Code Review}\end{aligned}$$

(สมการที่ 2.26)

$$\text{Yield\%} = 100 \times \left(\frac{\text{Early Remove}}{\text{Early Inject}}\right)$$

(สมการที่ 2.27)

5. Test Defect vs A/FR

- แผนภูมิเส้นแสดงผลการทดสอบข้อผิดพลาดที่พบต่ออัตราส่วนระหว่างการประเมินค่ากับค่าความผิดพลาด
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Test Defects} = 1000 \times (\text{Defect Injected Test} / \text{ActLOC})$$

(สมการที่ 2.28)

$$\text{Productivity} = (\text{ActLOC} / \text{ActMin}) \times 60$$

(สมการที่ 2.29)

$$\text{Appraisal COQ} = 100 \times \left\{\frac{(\text{Design Review Time} + \text{Code Review Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.30)

$$\text{Failure COQ} = 100 \times \left\{\frac{(\text{Compile Time} + \text{Test Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.31)

$$\text{A/FA} = \frac{\text{Appraisal COQ}}{\text{Failure COQ}}$$

(สมการที่ 2.32)

6. Test Defect vs Yield

- แผนภูมิเส้นแสดงผลการทดสอบข้อผิดพลาดที่พบต่อเปอร์เซ็นต์ข้อบกพร่อง
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Test Defects} = 1000 \times \left(\frac{\text{Defect Injected Test}}{\text{ActLOC}}\right)$$

(สมการที่ 2.33)

$$\begin{aligned}\text{Early Inject} = {} & \text{Defect Injected Planning} + \text{Defect Injected Design} \\ & + \text{Defect Injected Design Review} \\ & + \text{Defect Injected Code} + \text{Defect Injected Code Review}\end{aligned}$$

(สมการที่ 2.34)

$$\begin{aligned}\text{Early Remove} = {} & \text{Defect Removed Planning} + \text{Defect Removed Design} + \\ & \text{Defect Removed Design Review} + \text{Defect Removed Code} \\ & + \text{Defect Removed Code Review}\end{aligned}$$

(สมการที่ 2.35)

$$\text{Yield\%} = 100 \times \left(\frac{\text{Early Remove}}{\text{Early Inject}}\right)$$

(สมการที่ 2.36)

7. Yield vs A/FR

- แผนภูมิเส้นแสดงเปอร์เซ็นต์ข้อบกพร่องต่ออัตราส่วนระหว่างการประเมินค่ากับค่าความผิดพลาด
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Early Inject} = \text{Defect Injected Planning} + \text{Defect Injected Design} + \text{Defect Injected Design Revie} + \text{Defect Injected Code} + \text{Defect Injected Code Review})$$

(สมการที่ 2.37)

$$\text{Early Remove} = \text{Defect Removed Planning} + \text{Defect Removed Design} + \text{Defect Removed Design Review} + \text{Defect Removed Code} + \text{Defect Removed Code Review}$$

(สมการที่ 2.38)

$$\text{Yield\%} = 100 \times \left(\frac{\text{Early Remove}}{\text{Early Inject}}\right)$$

(สมการที่ 2.39)

$$\text{Appraisal COQ} = 100 \times \left\{\frac{(\text{Design Review Time} + \text{Code Review Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.40)

$$\text{Failure COQ} = 100 \times \left\{\frac{(\text{Compile Time} + \text{Test Time})}{\text{ActMin}}\right\}$$

(สมการที่ 2.41)

$$\text{A/FA} = \frac{\text{Appraisal COQ}}{\text{Failure COQ}}$$

(สมการที่ 2.42)

2.4.3 สมรรถนะด้านคุณภาพ (Quality Performance)

1. Appraisal Cost of Quality
   - แผนภูมิเส้นแสดงเปอร์เซ็นต์การประเมินค่า
   - สูตรที่ใช้ในการคำนวณ คือ

$$Appraisal\ COQ = 100 \times \left\{\frac{(Design\ Review\ Time + Code\ Review\ Time)}{ActMin}\right\}$$

(สมการที่ 2.43)

2. Appraisal to Failure Ratio
   - แผนภูมิเส้นแสดงเปอร์เซ็นต์อัตราส่วนระหว่างการประเมินค่าและค่าความผิดพลาด
   - สูตรที่ใช้ในการคำนวณ คือ

$$Appraisal\ COQ = 100 \times \left\{\frac{(Design\ Review\ Time + Code\ Review\ Time)}{ActMin}\right\}$$

(สมการที่ 2.44)

$$Failure\ COQ = 100 \times \left\{\frac{(Compile\ Time + Test\ Time)}{ActMin}\right\}$$

(สมการที่ 2.45)

$$A/FA = \frac{Appraisal\ COQ}{Failure\ COQ}$$

(สมการที่ 2.46)

3. Compile vs Test Defects
   - แผนภูมิเส้นแสดงขนาดของการแปลโปรแกรมต่อการทดสอบข้อผิดพลาด
   - สูตรที่ใช้ในการคำนวณ คือ

$$Compile = 1000 \times \left(\frac{Defect\ Removed\ Compile}{ActLOC}\right)$$

(สมการที่ 2.47)

$$Test\ Defects = 1000 \times \left(\frac{Defect\ Removed\ Test}{ActLOC}\right)$$

(สมการที่ 2.48)

4. CR Review Rate vs Yield

- แผนภูมิเส้นแสดงอัตราการทบทวนการเขียนโปรแกรมต่อเปอร์เซ็นต์ข้อบกพร่อง
- สูตรที่ใช้ในการคำนวณ คือ

$$CRR = 60 \times \left(\frac{ActLOC}{Code\ Review\ Time}\right)$$

(สมการที่ 2.49)

$$CRR\ Yield = 100 \times \left\{\frac{Defect\ Removed\ Code\ Review}{\left(\begin{array}{c} Defect\ Injected\ Planning\ + \\ Defect\ Injected\ Design\ + \\ Defect\ Injected\ Design\ Review\ + \\ Defect\ Injected\ Code\ + \\ Defect\ Injected\ Code\ Review\ + \\ Defect\ Removed\ Planning\ + \\ Defect\ Removed\ Design\ + \\ Defect\ Removed\ Design\ Review\ + \\ Defect\ Removed\ Code \end{array}\right)}\right\}$$

(สมการที่ 2.50)

5. Defect Age

- แผนภูมิแท่งแสดงความถี่ของข้อผิดพลาดที่ค้างอยู่ในแต่ละช่วงเวลา
- สูตรที่ใช้ในการคำนวณ คือ

ให้แต่ละช่วงเวลามีลำดับตัวเลข คือ

ตารางที่ 2.3 กำหนดค่าลำดับตัวเลขให้กับช่วงการทำงาน

ลำดับ	ช่วงเวลา
Planning	1
Design	2
Design Review	3
Code	4
Code Review	5
Compile	6
Test	7
Postmortem	8

ในข้อมูลที่อยู่ในตาราง Defects LOC โดยทำการเทียบช่วงเวลาแต่ละช่วงเวลากับลำดับของช่วงเวลา

$$Defect\ Age = Removed - Injected$$

(สมการที่ 2.51)

6. Defect Fix Time by Type

- แผนภูมิแท่งแสดงเวลาที่ใช้ในการแก้ไขข้อผิดพลาดในแต่ละประเภท
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Fix\ Time\ by\ Type = number\ of\ defect\ fix\ time\ by\ type$$

(สมการที่ 2.52)

7. Defect Injection % by Phase
   - แผนภูมิเส้นแสดงเปอร์เซ็นต์ข้อผิดพลาดที่พบในช่วงการทำงาน
   - แสดง 2 ช่วง คือ
     - ช่วงโค๊ด
     - ช่วงออกแบบ
   - สูตรที่ใช้ในการคำนวณ คือ

$$Code\ Line = 100 \times \left\{ \frac{\left(\begin{array}{c} Defect\ Injected\ Planning + \\ Defect\ Injected\ Design + \\ Defect\ Injected\ Design\ Review + \\ Defect\ Injected\ Code \end{array} \right)}{ActDef} \right\}$$

(สมการที่ 2.53)

$$Design\ Line = 100 \times \left\{ \frac{(Defect\ Injected\ Planning + Defect\ Injected\ Design)}{ActDef} \right\}$$

(สมการที่ 2.54)

8. Defect Removal Yield
   - แผนภูมิแสดงเปอร์เซ็นต์การแก้ไขข้อผิดพลาด
   - สูตรที่ใช้ในการคำนวณ คือ

$$\begin{aligned} Early\ Inject = {} & Defect\ Injected\ Planning + Defect\ Injected\ Design \\ & + Defect\ Injected\ Design\ Review + Defect\ Injected\ Code \\ & + Defect\ Injected\ Code\ Review\ Time \end{aligned}$$

(สมการที่ 2.55)

$$\begin{aligned} Early\ Remove = {} & Defect\ Removed\ Planning + Defect\ Removed\ Design + \\ & Defect\ Removed\ Design\ Review + Defect\ Removed\ Code \\ & + Defect\ Removed\ Code\ Review \end{aligned}$$

(สมการที่ 2.56)

$$Yield\% = 100 \times \left(\frac{Early\ Remove}{Early\ Inject}\right)$$

(สมการที่ 2.57)

9. Defects Injected by Phase To Date

- แผนภูมิวงกลมแสดงจำนวนข้อผิดพลาดสะสมในแต่ละช่วงการทำงาน
- สูตรที่ใช้ในการคำนวณ คือ

$$To\ Date\ \% = \frac{(To\ Date \times 100)}{sum\ of\ all\ To\ Date}$$

(สมการที่ 2.58)

คิดแต่ละช่วงการทำงาน โดยที่

To Date % คือ เปอร์เซ็นต์ To Date ช่วงการทำงานที่พบข้อผิดพลาด

To Date คือ ค่ารวมในแต่ละช่วงการทำงาน

10. Defects Injected in Code

- แผนภูมิเส้นแสดงข้อผิดพลาดที่พบในช่วงเขียนโปรแกรม
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Injected\ Code = 1000 \times \left(\frac{Defect\ Injected\ Code}{ActLOC}\right)$$

(สมการที่ 2.59)

11. Defects Injected in Design

- แผนภูมิเส้นแสดงข้อผิดพลาดที่พบในช่วงออกแบบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Injected\ Design = 1000 * \left(\frac{Defect\ Injected\ Design}{ActLOC}\right)$$

(สมการที่ 2.60)

12. Defects Removal % by Phase

- แผนภูมิเส้นแสดงเปอร์เซ็นต์การแก้ไขข้อผิดพลาดที่พบในช่วงการทำงาน
- แสดง 4 ช่วง คือ
  - ช่วงทบทวนการออกแบบ
  - ช่วงทบทวนการเขียนโปรแกรม
  - ช่วงแปลโปรแกรม
  - ช่วงทดสอบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Design\ Review\ Line = 100 \times \left\{ \frac{\left(\begin{array}{c} Defect\ Removed\ Planning + \\ Defect\ Removed\ Design + \\ Defect\ Removed\ Design\ Review \end{array} \right)}{ActDef} \right\}$$

(สมการที่ 2.61)

$$Code\ Review\ Line = 100 \times \left\{ \frac{\left(\begin{array}{c} Defect\ Removed\ Planning + \\ Defect\ Removed\ Design + \\ Defect\ Removed\ Design\ Review + \\ Defect\ Removed\ Code + \\ Defect\ Removed\ Code\ Review \end{array} \right)}{ActDef} \right\}$$

(สมการที่ 2.62)

$$Compile\ Line\ =\ 100\ \times \left\{ \frac{\left(\begin{array}{c} Defect\ Removed\ Planning\ + \\ Defect\ Removed\ Design\ + \\ Defect\ Removed\ Design\ Review\ + \\ Defect\ Removed\ Code\ + \\ Defect\ Removed\ Code\ Review\ + \\ Defect\ Removed\ Compile \end{array} \right)}{ActDef} \right\}$$

(สมการที่ 2.63)

$$Compile\ Line\ =\ 100\ \times \left\{ \frac{\left(\begin{array}{c} Defect\ Removed\ Planning\ + \\ Defect\ Removed\ Design\ + \\ Defect\ Removed\ Design\ Review\ + \\ Defect\ Removed\ Code\ + \\ Defect\ Removed\ Code\ Review\ + \\ Defect\ Removed\ Compile\ + \\ Defect\ Removed\ Test \end{array} \right)}{ActDef} \right\}$$

(สมการที่ 2.64)

13. Defect Removal Leverage

- แผนภูมิเส้นแสดงประสิทธิภาพในการแก้ไขข้อผิดพลาด
- แสดง 3 ส่วน คือ
  - การทบทวนการออกแบบต่อการทดสอบ
  - การทบทวนการเขียนโปรแกรมต่อการทดสอบ
  - การแปลโปรแกรมต่อการทดสอบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Design\ Review\ = 60\ \times \left(\frac{Defect\ Removed\ Design\ Review}{Design\ Review\ Time}\right)$$

(สมการที่ 2.65)

$$Code\ Review\ = 60 \times \left(\frac{Defect\ Removed\ Code\ Review}{Code\ Review\ Time}\right)$$

(สมการที่ 2.66)

$$Compile\ = 60\ \times \left(\frac{Defect\ Compile}{Compile\ Time}\right)$$

(สมการที่ 2.67)

$$Test\ = 60\ \times \left(\frac{Defect\ Removed\ Test}{Test\ Time}\right)$$

(สมการที่ 2.68)

$$Defect\ Removal\ Leverage = \frac{Defects\ Removed\ per\ hour\ for\ a\ review\ or\ Compile\ phase}{Defects\ Removed\ per\ hour\ for\ unit\ test}$$

(สมการที่ 2.69)

14. Defects Removed by Phase To Date

- แผนภูมิวงกลมแสดงจำนวนการแก้ไขข้อผิดพลาดสะสมในแต่ละช่วงการทำงาน
- สูตรที่ใช้ในการคำนวณ คือ

$$To\ Date\ \% = \frac{(To\ Date\ \times\ 100)}{sum\ of\ all\ To\ Date}$$

(สมการที่ 2.70)

คิดแต่ละช่วงการทำงาน โดยที่

To Date % คือ เปอร์เซ็นต์ To Date ช่วงการทำงานที่แก้ไขข้อผิดพลาด

To Date คือ ค่ารวมในแต่ละช่วงการทำงาน

15. Defects Removed by Type

- แผนภูมิแท่งแสดงจำนวนข้อผิดพลาดที่เกิดขึ้นตามประเภทของข้อผิดพลาด
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Removed\ by\ Tpye\ =\ number\ of\ defect\ removed\ by\ type$$

(สมการที่ 2.71)

16. Defects Removed in Code Review

- แผนภูมิเส้นแสดงการแก้ไขข้อผิดพลาดที่พบในช่วงทบทวนการเขียนโปรแกรม
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Removed\ Code\ Review\ =\ 1000\ \times \left(\frac{Defect\ Injected\ Code\ Review}{ActLOC}\right)$$

(สมการที่ 2.72)

17. Defects Removed in Compile

- แผนภูมิเส้นแสดงการแก้ไขข้อผิดพลาดที่พบในช่วงแปลโปรแกรม
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Removed\ Compile\ =\ 1000\ \times \left(\frac{Defect\ Injected\ Compile}{ActLOC}\right)$$

(สมการที่ 2.73)

18. Defects Removed in Design Review

- แผนภูมิเส้นแสดงการแก้ไขข้อผิดพลาดที่พบในช่วงทบทวนการออกแบบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Removed\ Design\ Review = 1000 \times \left(\frac{Defect\ Injected\ Design\ Review}{ActLOC}\right)$$

(สมการที่ 2.74)

19. Defects Removed in Test

- แผนภูมิเส้นแสดงการแก้ไขข้อผิดพลาดที่พบในช่วงการทดสอบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Removed\ Test = 1000 \times \left(\frac{Defect\ Injected\ Test}{ActLOC}\right)$$

(สมการที่ 2.75)

20. DLDR Review Rate vs Yield

- แผนภูมิเส้นแสดงอัตราการทบทวนการออกแบบต่อเปอร์เซ็นต์ข้อบกพร่อง
- สูตรที่ใช้ในการคำนวณ คือ

$$DLDR = 60 \times \left(\frac{ActLOC}{Design\ Review\ Time}\right)$$

(สมการที่ 2.76)

$$DLDR\ Yield = 100 \times \left\{\frac{Defect\ Removed\ Design\ Review}{\left(\begin{array}{c} Defect\ Injected\ Planning\ + \\ Defect\ Injected\ Design\ + \\ Defect\ Injected\ Design\ Review\ + \\ Defect\ Removed\ Planning\ + \\ Defect\ Removed\ Design \end{array}\right)}\right\}$$

(สมการที่ 2.77)

21. Failure Cost of Quality

- แผนภูมิเส้นแสดงเปอร์เซ็นต์ค่าความผิดพลาด
- สูตรที่ใช้ในการคำนวณ คือ

$$Failure\ COQ\ =\ 100\ \times \left\{\frac{(Compile\ Time\ +\ Test\ Time)}{ActMin}\right\}$$

(สมการที่ 2.78)

22. PSP Defect Density Report

- ตารางสรุปผลข้อผิดพลาด
- ประกอบด้วย
  - ลำดับของโปรแกรม
  - จำนวนข้อผิดพลาดทั้งหมด
  - ขนาดของโปรแกรม
  - ค่าความผิดพลา
  - จำนวนข้อผิดพลาดเฉพาะช่วงแปลโปรแกรม
  - ค่าความผิดพลาดเฉพาะช่วงแปลโปรแกรม
  - จำนวนข้อผิดพลาดเฉพาะการทดสอบ
  - ค่าความผิดพลาดเฉพาะช่วงการทดสอบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Defect\ Density\ =\ 1000\ \times \left(\frac{Total\ Defects}{Actual\ A\&M\ Size}\right)$$

(สมการที่ 2.79)

$$Compile\ Density\ =\ 1000\ \times \left(\frac{Compile\ Defect}{Actual\ A\&M\ Size}\right)$$

(สมการที่ 2.80)

$$Test\ Density\ =\ 1000\ \times \left(\frac{Test\ Defect}{Actual\ A\&M\ Size}\right)$$

(สมการที่ 2.81)

23. PSP Defect Fix Time Report

- ตารางแสดงเวลาที่ใช้ในการแก้ไขตามช่วงต่างๆ
- แสดง 2 ช่วง คือ
  - ช่วงออกแบบ
  - ช่วงเขียนโปรแกรม
- แต่ละช่วงแบ่งย่อยอีก 3 อย่าง คือ
  - เวลาที่ใช้ในการแก้ไข
  - จำนวนข้อผิดพลาด
  - ค่าเฉลี่ยเวลาที่ใช้ในการแก้ไข
- ประกอบด้วย
  - ช่วงที่เกิดข้อผิดพลาด
  - การแก้ไขเฉพาะช่วงแปลโปรแกรม
  - การแก้ไขเฉพาะช่วงทดสอบ
  - การแก้ไขทั้งช่วงแปลโปรแกรมและทดสอบ
- สูตรที่ใช้ในการคำนวณ คือ

$$Avg.Fix\ Time\ =\ \frac{Fix\ Time}{Total}$$

(สมการที่ 2.82)

24. PSP Percent Injected and Removed by Type Report

- ตารางสรุปผลเปอร์เซ็นต์ข้อผิดพลาดที่เกิดขึ้น
- ประกอบด้วย
  - ประเภทของข้อผิดพลาด
  - จำนวนข้อผิดพลาดที่เกิดขึ้นเฉพาะช่วงออกแบบ
  - จำนวนข้อผิดพลาดที่เกิดขึ้นเฉพาะช่วงเขียนโปรแกรม

- เปอร์เซ็นต์ข้อผิดพลาดที่เกิดขึ้นเฉพาะช่วงออกแบบ
- เปอร์เซ็นต์ข้อผิดพลาดที่เกิดขึ้นเฉพาะช่วงเขียนโปรแกรม
- จำนวนข้อผิดพลาดที่ถูกแก้ไขเฉพาะช่วงออกแบบ
- จำนวนข้อผิดพลาดที่ถูกแก้ไขเฉพาะช่วงเขียนโปรแกรม
- เปอร์เซ็นต์ข้อผิดพลาดที่เถูกแก้ไขเฉพาะช่วงออกแบบ
- เปอร์เซ็นต์ข้อผิดพลาดที่เถูกแก้ไขเฉพาะช่วงเขียนโปรแกรม

- สูตรที่ใช้ในการคำนวณ คือ

$$Total\ Number\ Injected\ Design = Summation\ of\ Number\ Injected\ Design$$

(สมการที่ 2.83)

$$Total\ Number\ Injected\ Code = Summation\ of\ Number\ Injected\ Code$$

(สมการที่ 2.84)

$$Percentage\ Injected\ Design = \frac{Number\ Injected\ Design}{Total\ Number\ Injected\ Design}$$

(สมการที่ 2.85)

$$Percentage\ Injected\ Code = \frac{Number\ Injected\ Code}{Total\ Number\ Injected\ Code}$$

(สมการที่ 2.86)

$$Total\ Number\ Removed\ Compile = Summation\ of\ Number\ Removed\ Compile$$

(สมการที่ 2.87)

$$Total\ Number\ Removed\ Test = Summation\ of\ Number\ Removed\ Test$$

(สมการที่ 2.88)

$$Percentage\ Removed\ Compile = \frac{Number\ Removed\ Compile}{Total\ Number\ Removed\ Design}$$

(สมการที่ 2.89)

$$Percentage\ Removed\ Test = \frac{Number\ Removed\ Test}{Total\ Number\ Removed\ Code}$$

(สมการที่ 2.90)

25. PSP Percentage Defects Found by Compile Report

- ตารางสรุปผลเปอร์เซ็นต์ข้อผิดพลาดที่พบในช่วงคอมไพล์
- ประกอบด้วย
  - ประเภทของข้อผิดพลาด
  - จำนวนข้อผิดพลาดที่เกิดในช่วงคอมไพล์
  - จำนวนข้อผิดพลาดที่พบในช่วงคอมไพล์
  - เปอร์เซ็นต์ข้อผิดพลาดที่พบในช่วงคอมไพล์
- สูตรที่ใช้ในการคำนวณ คือ

$$Percentage\ of\ Defects\ Found\ by\ the\ Compile = \frac{Number\ of\ Defects\ at\ Compile\ Entry}{Number\ of\ Defects\ Found\ in\ Compile}$$

(สมการที่ 2.91)

26. Review Rate

- แผนภูมิเส้นแสดงอัตราการทบทวน
- แสดง 3 ส่วน คือ
  - ส่วนอัตราการทบทวนการออกแบบ (DLDR)
  - ส่วนอัตราการทบทวนการเขียนโปรแกรม (CDR)
  - ส่วนอัตราการทบทวนทั้งการออกแบบและเขียนโปรแกรม (Both)
- สูตรที่ใช้ในการคำนวณ คือ

$$DLDR = 60 \times \left(\frac{ActLOC}{Design\ Review\ Time}\right)$$

(สมการที่ 2.92)

$$CDR = 60 \times \left(\frac{ActLOC}{Code\ Review\ Time}\right)$$

(สมการที่ 2.93)

$$Both = 60 \times \left\{\frac{ActLOC}{(Design\ Review\ Time + Code\ Review\ Time)}\right\}$$

(สมการที่ 2.94)

27. Review Rate vs Yield

- แผนภูมิเส้นแสดงอัตราการทบทวนต่อเปอร์เซ็นต์ข้อบกพร่อง
- สูตรที่ใช้ในการคำนวณ คือ

$$Review\ Rate = 60 \times \left\{\frac{ActLOC}{(Design\ Review\ Time + Code\ Review\ Time)}\right\}$$

(สมการที่ 2.95)

$$Early\ Inject = Defect\ Injected\ Planning\ Time + Defect\ Injected\ Design\ Time + Defect\ Injected\ Design\ Review\ Time + Defect\ Injected\ Code + Defect\ Injected\ Code\ Review\ Time$$

(สมการที่ 2.96)

$$Early\ Remove = Defect\ Removed\ Planning + Defect\ Removed\ Design + Defect\ Removed\ Design\ Review + Defect\ Removed\ Code + Code\ Review$$

(สมการที่ 2.97)

$$Yield\% = 100 \times \left(\frac{Early\ Remove}{Early\ Inject}\right)$$

(สมการที่ 2.98)

28. Total Cost of Quality

- แผนภูมิเส้นแสดงเปอร์เซ็นต์ผลรวมของการประเมินค่าและค่าความผิดพลาด
- สูตรที่ใช้ในการคำนวณ คือ

$$Appraisal\ COQ\ =\ 100\ \times \left\{\frac{(Design\ Review\ Time\ +\ Code\ Review\ Time)}{ActMin}\right\}$$

(สมการที่ 2.99)

$$Failure\ COQ\ =\ 100\ \times \left\{\frac{(Compile\ Time\ +\ Test\ Time)}{ActMin}\right\}$$

(สมการที่ 2.100)

$$Total\ COQ\ =\ Appraisal\ COQ\ +\ Failure\ COQ$$

(สมการที่ 2.101)

29. Total Defects

- แผนภูมิเส้นแสดงข้อผิดพลาดทั้งหมดที่พบ
- สูตรที่ใช้ในการคำนวณ คือ

$$\text{Total Defects}\ =\ 1000\ \times \left(\frac{\text{ActDef}}{\text{ActLOC}}\right)$$

(สมการที่ 2.102)

2.4.4 ภาพรวม

1. Process Quality Index
   - แผนภูมิใยแมงมุมแสดงคุณภาพในการทำงาน
   - ประกอบด้วย 5 ด้าน คือ
     - คุณภาพในการออกแบบ
     - คุณภาพในการทบทวนการออกแบบ
     - คุณภาพในการเขียนโปรแกรม
     - คุณภาพในการทบทวนการเขียนโปรแกรม
     - คุณภาพของโปรแกรม
     - รวมทั้งมีตัวชี้วัดคุณภาพในการทำงาน (PQI)
   - สูตรที่ใช้ในการคำนวณ คือ

$$Design\ Code\ Time = \frac{Design}{Code} \quad \text{(สมการที่ 2.103)}$$

$$Code\ Review\ Time = 2.0 \times \left(\frac{Code\ Review}{Code}\right) \quad \text{(สมการที่ 2.104)}$$

$$Design\ Review\ Time = 2.0 \times \left(\frac{Design\ Review}{Design}\right) \quad \text{(สมการที่ 2.105)}$$

$$Compile\ Defects\ KLOC = \frac{20}{\left(\frac{10}{\left(1000 \times \left(\frac{Compile}{ActLOC}\right)\right)}\right)} \quad \text{(สมการที่ 2.106)}$$

$$Unit\ Test\ Defects\ KLOC\ =\ \frac{10}{\left(\frac{5}{\left(1000\ \times\ \left(\frac{Test}{ActLOC}\right)\right)}\right)}$$

(สมการที่ 2.107)

$$Design\ Quality\ =\ Minimum\ (1.0, Design\ Code\ Time)$$

(สมการที่ 2.108)

$$Code\ Review\ Quality\ =\ Minimum\ (1.0, Code\ Review\ Time)$$

(สมการที่ 2.109)

$$Code\ Quality\ =\ Minimum\ (1.0, Compile\ Defects\ KLOC)$$

(สมการที่ 2.110)

$$Program\ Quality\ =\ Minimum\ (1.0, Unit\ Test\ Defects\ KLOC)$$

(สมการที่ 2.111)

$$Design\ Review\ Quality\ =\ Minimum\ (1.0, Design\ Review\ Time)$$

(สมการที่ 2.112)

$$PQI\ =\ Design\ Quality\ \times\ Design\ Review\ Quality\ \times\ Code\ Quality\ \times\ Code\ Review\ Quality\ \times\ Program\ Quality$$

(สมการที่ 2.113)

บทที่ 3
วิธีการดำเนินงาน

วิธีการดำเนินงานนั้นเป็นการนำแนวคิดของกระบวนการพัฒนาซอฟต์แวร์ระดับบุคคล (Personal Software Process; PSP) มาออกแบบและพัฒนาเป็นโปรแกรมเพื่อช่วยวิเคราะห์ข้อมูล PSP ในหลายๆรูปแบบ เช่น แผนภูมิเส้น แผนภูมิวงกลม หรือตาราง เป็นต้น

3.1 คำอธิบายของระบบ (System Description)

โปรแกรมช่วยวิเคราะห์ข้อมูล Personal Software Process (PSP) เป็นการวิเคราะห์ข้อมูลเพื่อแสดงประสิทธิภาพในการทำงานของผู้ใช้ตามสมรรถนะด้านต่างๆ ดังนี้ การวางแผน กระบวนการ คุณภาพและภาพรวม

3.2 ความต้องการ (Requirement)

3.2.1 ความต้องการเชิงคุณภาพ (Quality Attribute or Non-functional Requirements)

1. โปรแกรมที่พัฒนาขึ้นจะใช้ฐานข้อมูลที่ผู้ใช้สามารถใช้ได้โดยไม่ต้องเสียค่าลิขสิทธิ์
2. โปรแกรมที่พัฒนาขึ้นจะใช้ไลบรารีที่ผู้ใช้ไม่ต้องเสียค่าลิขสิทธิ์เพิ่มเติม

3.2.2 ความต้องการเชิงหน้าที่ (Functional Requirements)

ตารางที่ 3.1 ความต้องการเชิงหน้าที่ (Functional Requirements)

เครื่องวิเคราะห์ข้อมูล	การแสดงผล	การควบคุมคุณภาพ (Quality Control)
1. ด้านการวางแผน		
- Actual Development Time	แผนภูมิเส้น	มี
- Actual Size	แผนภูมิเส้น	มี
- Percent Compile + Test Time	แผนภูมิเส้น	มี
- Percent Compile Time	แผนภูมิเส้น	มี
- Percent Planning + Postmortem Time	แผนภูมิเส้น	มี
- Percent Planning Time	แผนภูมิเส้น	มี
- Percent Postmortem Time	แผนภูมิเส้น	มี
- Percent Test Time	แผนภูมิเส้น	มี
- Percent Time in Phase To Date	แผนภูมิวงกลม	ไม่มี
- Size Estimating Error	แผนภูมิเส้น	มี
- Time Estimating Error	แผนภูมิเส้น	มี
2. ด้านกระบวนการ		
- A/FR vs Yield	แผนภูมิเส้น	ไม่มี
- Productivity	แผนภูมิเส้น	ไม่มี
- Productivity vs A/FR	แผนภูมิเส้น	ไม่มี
- Productivity vs Yield	แผนภูมิเส้น	ไม่มี
- Test Defect vs A/FR	แผนภูมิเส้น	ไม่มี
- Test Defect vs Yield	แผนภูมิเส้น	ไม่มี
- Yield vs A/FR	แผนภูมิเส้น	ไม่มี
3. ด้านคุณภาพ		
- Appraisal Cost of Quality	แผนภูมิเส้น	มี
- Appraisal to Failure Ratio	แผนภูมิเส้น	มี

ตารางที่ 3.1 (ต่อ) ความต้องการเชิงหน้าที่ (Functional Requirements)

เครื่องวิเคราะห์ข้อมูล	การแสดงผล	การควบคุมคุณภาพ (Quality Control)
- Compile vs Test Defects	แผนภูมิเส้น	มี
- CR Review Rate vs Yield	แผนภูมิเส้น	ไม่มี
- Defect Age	แผนภูมิแท่ง	มี
- Defect Age (Chart)	แผนภูมิวงกลม	มี
- Defect Fix Time by Type	แผนภูมิแท่ง	ไม่มี
- Defect Injection % by Phase	แผนภูมิเส้น	ไม่มี
- Defect Removal Yield	แผนภูมิเส้น	มี
- Defects Injected by Phase To Date	แผนภูมิวงกลม	ไม่มี
- Defects Injected in Code	แผนภูมิเส้น	มี
- Defects Injected in Design	แผนภูมิเส้น	มี
- Defects Removal % by Phase	แผนภูมิเส้น	ไม่มี
- Defects Removal Leverage	แผนภูมิเส้น	ไม่มี
- Defects Removed by Phase To Date	แผนภูมิวงกลม	ไม่มี
- Defects Removed by Type	แผนภูมิแท่ง	ไม่มี
- Defects Removed in Code Review	แผนภูมิเส้น	มี
- Defects Removed in Compile	แผนภูมิเส้น	มี
- Defects Removed in Design Review	แผนภูมิเส้น	มี
- Defects Removed in Test	แผนภูมิเส้น	มี
- DLDR Review Rate vs Yield	แผนภูมิเส้น	ไม่มี
- Failure Cost of Quality	แผนภูมิเส้น	มี
- PSP Defect Density Report	ตาราง	ไม่มี
- PSP Defect Fix Time Report	ตาราง	ไม่มี
- PSP Percent Injected and Removed by Type Report	ตาราง	ไม่มี

ตารางที่ 3.1 (ต่อ) ความต้องการเชิงหน้าที่ (Functional Requirements)

เครื่องวิเคราะห์ข้อมูล	การแสดงผล	การควบคุมคุณภาพ (Quality Control)
- PSP Percentage Defects Found by Compile Report	ตาราง	ไม่มี
- Review Rate	แผนภูมิเส้น	ไม่มี
- Review Rate vs Yield	แผนภูมิเส้น	ไม่มี
- Total Cost of Quality	แผนภูมิเส้น	มี
- Total Defects	แผนภูมิเส้น	มี
4. ภาพรวม		
- Process Quality Index	แผนภูมิใยแมงมุม	มี

3.3 สมมติฐานของการออกแบบ (Design Assumption)

1. โครงสร้างข้อมูลในส่วนของตัวชี้วัดพื้นฐานไม่อยู่ในขอบเขตของโปรแกรมนี้ โดยถือว่าได้รับข้อมูลมาจากโปรแกรมช่วยบันทึกข้อมูล
2. การแสดงผลจะแสดงเฉพาะผลการคำนวณโดยไม่ได้แสดงข้อแนะนำในการปรับปรุง หรือวิเคราะห์เพิ่มเติมในเชิงลึก

3.4 การออกแบบโปรแกรม (Design)

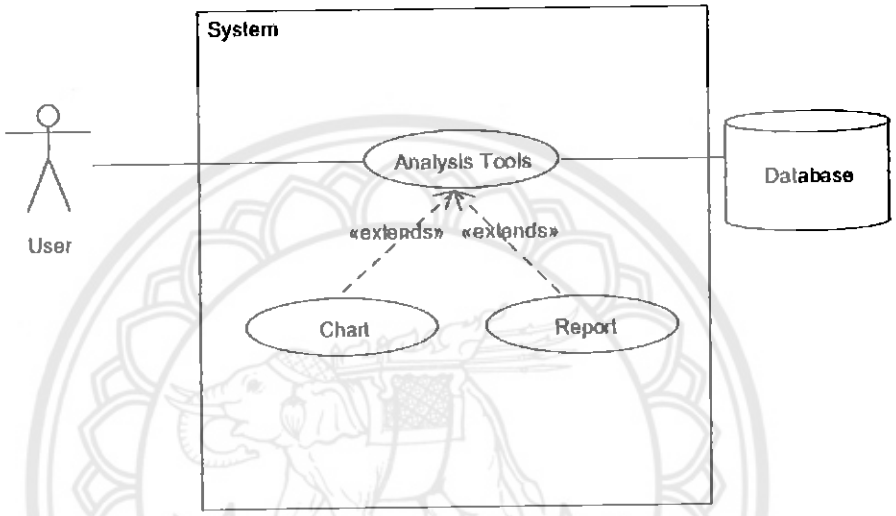
การออกแบบโปรแกรมใช้มาตรฐานยูเอ็มแอล (Unified Modeling Language; UML) ที่เป็นภาษาที่ใช้แสดงแบบการทำงานของระบบ ซึ่งอธิบายพฤติกรรมของซอฟแวร์ด้วยแผนภาพ ตามมุมมองต่างๆ ดังตาราง

ตารางที่ 3.2 Mapping UML and PSP Views

	Dynamic	Static
External	Use cases Diagrams, Sequence Diagrams	Class Diagrams

3.4.1 External และ Dynamic View

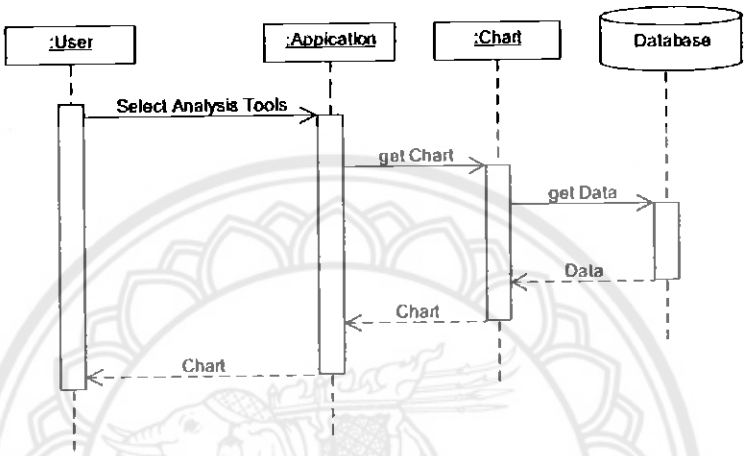
- Use Case Diagrams



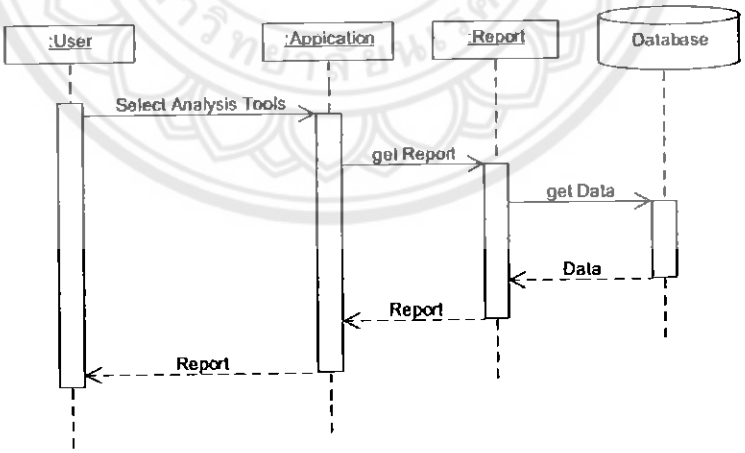
รูปที่ 3.1 Use Case Diagrams ของระบบ

- Sequence Diagrams

  แสดงการทำงานของระบบกับการวิเคราะห์ข้อมูลที่แสดงผลในรูปแบบของแผนภูมิและรูปแบบตารางรายงานผล



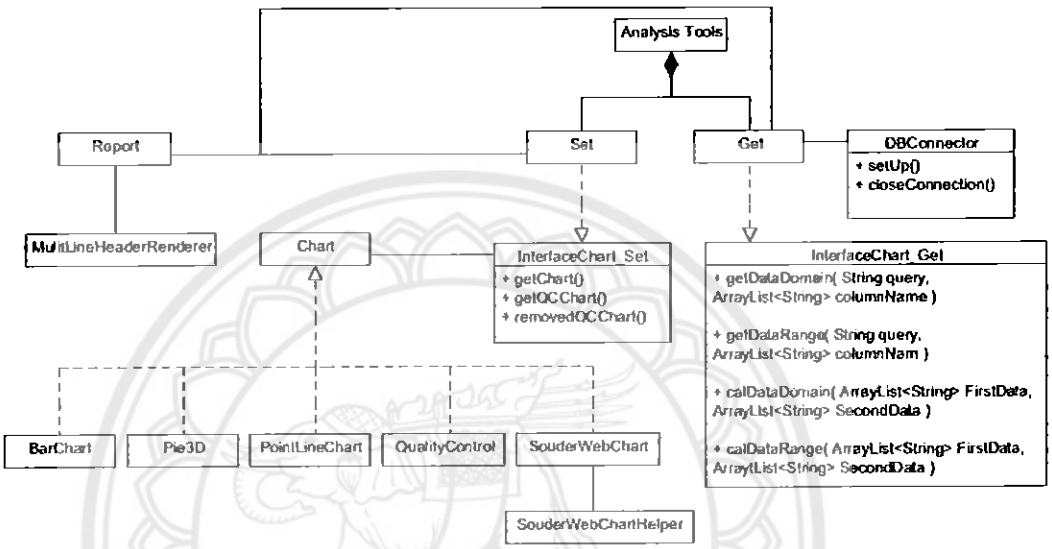
รูปที่ 3.2 Sequence Diagram แสดงการทำงานของระบบกับการวิเคราะห์ที่แสดงผลเป็นแผนภูมิ(chart)



รูปที่ 3.3 Sequence Diagram แสดงการทำงานของระบบกับการวิเคราะห์ที่แสดงผลเป็นตาราง

3.4.2 External และ Static View

- Class Diagrams



รูปที่ 3.4 Class Diagram ของระบบ

3.5 โครงสร้างข้อมูล (Database Schema)

โครงสร้างของข้อมูลที่นำมาวิเคราะห์ใช้อยู่ 2 ตาราง ซึ่งเป็นการรวมข้อมูลที่จำเป็นในการวิเคราะห์ ประกอบด้วย

1. Basic Analysis
   - เป็นตารางข้อมูลที่เก็บรวมรวบขนาด เวลา และจำนวนข้อผิดพลาดในแต่ละช่วง เพื่อนำไปวิเคราะห์และแสดงผลในหลายรูปแบบ เช่น แผนภูมิเส้น, แผนภูมิแท่ง แผนภูมิใยแมงมุม และรายงานสรุปผล เป็นต้น
   - โดยมีโครงสร้างข้อมูล ดังนี้

ตารางที่ 3.3 Database Schema of Basic Analysis

ProgramID	Integer
EstLOC	Integer
ActLOC	Integer
EstMin	Integer
ActMin	Integer
EstDef	Integer
ActDef	Integer
TP_Planning	Double
TP_Design	Double
TP_DesignReview	Double
TP_Code	Double
TP_CodeReview	Double
TP_Compile	Double
TP_Test	Double
TP_PM	Double
DIP_Planning	Double
DIP _Design	Double
DIP _DesignReview	Double

ตารางที่ 3.5 (ต่อ) Database Schema of Basic Analysis

DIP _Code	Double
DIP _CodeReview	Double
DIP _Compile	Double
DIP _Test	Double
DIP_Planning	Double
DRP _Design	Double
DRP _DesignReview	Double
DRP _Code	Double
DRP _CodeReview	Double
DRP _Compile	Double
DRP _Test	Double

หมายเหตุ

- TP คือ Time by Phase
- DIP คือ Defect Injected by Phase
- DRP คือ Defect Removed by Phase

2. Defects LOC

- เป็นตารางข้อมูลที่เก็บรวบรวมข้อผิดพลาดที่เกิดขึ้นทั้งหมด เพื่อนำไปวิเคราะห์และแสดงผลในหลายรูปแบบ เช่น แผนภูมิเส้น แผนภูมิแท่ง และรายงานสรุปผล เป็นต้น
- โดยมีโครงสร้างข้อมูล ดังนี้

ตารางที่ 3.4 Database Schema of Defects LOC

ProgramID	Integer
Date	Date
Number	Integer
Type	Integer
Injected	VARCHAR(50)
Removed	VARCHAR(50)
FixTime	Double
FixRef	Integer
Description	VARCHAR(500)

บทที่ 4
ผลการทดสอบ

บทนี้จะกล่าวถึงแผนการทดสอบ ซึ่งแบงเป็นการทดสอบ ได้ 3 ส่วน ดังนี้

- Unit Test คือ การทดสอบฟังก์ชันการทำงานว่าสามารถทำงานได้อย่างถูกต้อง
- Integration Test คือ การทดสอบการทำงานร่วมกันของคลาสและฟังก์ชันการทำงาน
- System Test คือ การทดสอบระบบโดยรวมทั้งหมด

4.1 แผนการทดสอบ

ตารางที่ 4.1 ตารางแผนการทดสอบ

ลำดับ	รายการ
1	Unit Test ● ทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Plan Performance ● ทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Process Performance ● ทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Quality Performance ● ทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Overall
2	Integration Test ● ทดสอบการทำงานของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Plan Performance ● ทดสอบการทำงานของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Process Performance ● ทดสอบการทำงานของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Quality Performance ● ทดสอบการทำงานของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Overall
3	System Test ● ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Plan Performance ● ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Process Performance ● ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Quality Performance ● ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Overall

4.2 ข้อมูล

Item							Time by Phase							Defects Injected by Phase								Defects Removed ed by Phase						
Program	EstLOC	ActLOC	EstMin	ActMin	EstDef	ActDef	Planning	Design	Design Review	Code	Code Review	Compile	Test	PM	Planning	Design	Design Review	Code	Code Review	Compile	Test	Planning	Design	Design Review	Code	Code Review	Compile	Test
1	0	63	80	83	0	2	0	0	0	57	0	7	16	3	0	0	0	1	0	1	0	0	0	0	0	0	0	2
2	150	47	60	47	0	7	1	4	0	25	0	7	4	7	0	0	0	7	0	0	0	0	0	0	0	0	4	3
3	78	106	22	95	0	3	0	3	0	29	0	30	11	23	0	0	0	3	0	0	0	0	0	0	0	0	3	0
4	108	111	57	141	0	3	3	25	0	56	0	1	23	34	0	2	0	1	0	0	0	0	0	0	0	0	0	3
5	207	149	68	109	0	6	2	11	0	34	0	2	45	16	0	5	0	1	0	0	0	0	0	0	0	0	1	5
6	97	88	12	11	0	1	4	0	0	0	0	2	5	2	0	0	0	1	0	0	0	0	0	0	0	0	0	1
7	325	63	41	25	0	2	3	2	0	9	0	2	4	4	0	0	0	2	0	0	0	0	0	0	0	0	0	0
8	0	252	0	175	0	3	0	2	0	115	0	2	14	42	0	1	0	2	0	0	0	0	0	0	0	0	2	0
9	0	422	0	94	0	3	7	11	0	61	0	0	8	7	0	0	0	3	0	0	0	0	0	0	0	0	0	3
10	287	58	0	72	0	1	1	3	0	41	0	0	6	20	0	0	0	1	0	0	0	0	0	0	0	0	0	1

รูปที่ 4.1 ตาราง Basic Analysis

Project	Date	Num	Type	Injected	Removed	FixTime	Fix Ref.	Description
1	5/15/2005	1	20	Compile	UT	6.6		
1	5/15/2005	2	70	Code	UT	4.9		
2	5/15/2005	3	20	Code	Compile	0.4		
2	5/15/2005	4	50	Code	Compile	2.8		
2	5/15/2005	5	20	Code	Compile	2.3	4	
2	5/15/2005	6	20	Code	Compile	1.1		
2	5/15/2005	7	20	Code	Test	0.7		
2	5/15/2005	9	50	Code	Test	0.1		
2	5/15/2005	10	70	Code	Test	2.5		
3	5/15/2005	11	20	Code	Compile	9.2		
3	5/15/2005	12	20	Code	Compile	0.5		
3	5/15/2005	13	50	Code	Compile	0.6		
4	5/15/2005	14	50	Code	Test	0.5		
4	5/15/2005	15	70	DLD	Test	0.2		
4	5/15/2005	16	50	DLD	Test	1.5		
5	5/15/2005	17	20	Code	Compile	1.8		
5	5/15/2005	18	50	DLD	Test	2.3		
5	5/15/2005	19	50	DLD	Test	4.0	17	
5	5/15/2005	20	50	DLD	Test	4.4		
5	5/15/2005	21	70	DLD	Test	0.3		
5	5/15/2005	22	70	DLD	Test	2.8		
6	5/15/2005	23	80	Code	Test	2.6		
7	5/15/2005	24	20	Code	Compile	1.5		
7	5/15/2005	25	20	Code	Compile	0.7		
8	5/15/2005	26	80	Code	Test	1.2		
8	5/15/2005	27	20	Code	Test	1.7		
8	5/15/2005	28	20	DLD	Test	2.2		
9	5/15/2005	29	20	Code	Test	2.0		
9	5/15/2005	30	20	Code	Test	1.4		
9	5/15/2005	31	20	Code	Test	0.0		
10	5/15/2005	32	20	Code	Test	2.3		

รูปที่ 4.2 ตาราง Defects LOC

4.3 Unit Test

จุดประสงค์ เพื่อตรวจสอบว่าโปรแกรมสามารถคำนวณค่าได้ถูกต้องตรงกับค่าที่คำนวณได้ โดยใช้เครื่องมือที่ใช้ในการทดสอบร่วม คือ JUnit

4.3.1 ทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Plan Performance

ตารางที่ 4.2 ผลการทดสอบหน่วนย่อยในกลุ่ม Plan Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. Actual Development Time	
- getDataRange	1.38 0.78 1.58 2.35 1.82 0.18 0.42 2.92 1.57 1.20
2. Actual Size	
- getDataRange	63 74 106 111 149 88 63 252 422 58
3. Percent Compile + Test Time	
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
4. Percent Compile Time	
- getDataRange	8.43 14.89 31.58 0.71 1.83 18.18 8.00 1.14 0.00 0.00
5. Percent Planning + Postmortem Time	
- getDataRange	3.61 17.02 24.21 26.24 16.51 54.55 28.00 24.00 14.89 29.17
6. Percent Planning Time	
- getDataRange	0.00 2.13 0.00 2.13 1.83 36.36 12.00 0.00 7.45 1.39
7. Percent Postmortem Time	
- getDataRange	3.61 14.89 24.21 24.11 14.68 18.18 16.00 24.00 7.45 27.78
8. Percent Test Time	
- getDataRange	19.28 8.51 11.58 16.31 41.28 45.45 16.00 8.00 8.51 8.33
9. Percent Time in Phase To Date	
- getDataRange	2.45 7.13 0.00 49.88 0.00 6.19 15.89 18.46

ตารางที่ 4.2 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Plan Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
10. Size Estimating Error	
- getDataRange	0.00 -50.67 35.90 5.71 -28.02 -9.28 -80.62 0.00 0.00 -79.79
11. Time Estimating Error	
- getDataRange	3.75 -21.67 331.82 147.37 60.29 -8.33 -39.02 0.00 0.00 0.00

4.3.2 การทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Process Performance

ตารางที่ 4.3 ผลการทดสอบหน่วนย่อยในกลุ่ม Process Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. A/FR vs Yield	
- getDataDomain	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
2. Productivity	
- getDataRange	45.54 91.47 66.95 47.23 82.02 480.00 151.20 86.40 269.36 48.33
3. Productivity vs A/FR	
- getDataDomain	27.71 23.40 43.16 17.02 43.12 63.64 21.00 9.14 8.51 8.33
4. Productivity vs Yield	
- getDataRange	45.54 94.47 66.95 47.23 82.02 480.00 151.20 86.40 269.36 48.33
5. Test Defects vs A/FR	
- getDataDomain	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
6. Test Defect vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
7. Yield vs A/FR	
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33

4.3.3 การทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Quality Performance

ตารางที่ 4.4 ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. Appraisal Cost of Quality	
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
2. Appraisal to Failure Ratio	
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
3. Compile vs Test Defects	
- getDataRange	31.75 40.54 0.00 27.03 33.56 11.36 0.00 11.90 7.11 17.24
4. CR Review Rate vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
5. Defect Age	
- getDataDomain	1 2 3 4 5 6 7 8
6. Defect Fix Time by Type	
- getDataDomain	20 50 70 80
7. Defect Injection % by Phase	
- getDataRange	50.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00
8. Defect Removal Yield	
- getDataRange	1.00 7.00 3.00 3.00 6.00 1.00 2.00 3.00 3.00 1.00
9. Defects Injected by Phase To Date	
- getDataRange	0.00 25.81 0.00 70.97 0.00 3.23 0.00
10. Defects Injected in Code	
- getDataRange	15.87 94.59 28.30 9.01 6.71 11.36 31.75 7.94 7.11 17.24
11. Defects Injected in Design	
- getDataRange	0.00 0.00 0.00 18.02 33.56 0.00 0.00 3.94 0.00 0.00

ตารางที่ 4.4 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

12. Defects Removal % by Phase	
- getDataRange	100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00
13. Defects Removal Leverage	
- getDataRange	0.00 0.76 0.00 0.00 4.50 0.00 0.00 0.00 0.00 0.00
14. Defects Removed by Phase To Date	
- getDataRange	0.00 0.00 0.00 0.00 32.26 67.74
15. Defects Removed by Type	
- getDataRange	16 5 8 2
16. Defects Removed in Code Review	
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
17. Defects Remvoed in Compile	
- getDataRange	0.00 54.05 28.30 0.00 6.71 0.00 31.75 0.00 0.00 0.00
18. Defects Removed in Design Review	
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
19. Defects Removed in Test	
- getDataRange	31.75 40.54 0.00 27.03 33.56 11.36 0.00 11.90 7.11 17.24
20. DLDR Review Rate vs Yield	
- getDataRange	1.00 7.00 3.00 3.00 6.00 1.00 2.00 3.00 3.00 1.00
21. Failure Cost of Quality	
- getDataRange	27.71 23.40 43.16 17.02 43..12 63.64 24.00 9.14 8.51 8.33
22. PSP Defect Density Report	
- getData	[1, 2, 63, 31.75, 0, 0.0, 2, 31.75] [2, 7, 74, 94.59, 4, 54.05, 3, 40.54] [3, 3, 106, 28.3, 3, 28.3, 0, 0.0] [4, 3, 111, 27.03, 0, 0.0, 3, 27.03] [5, 6, 149, 40.27, 1, 6.71, 5, 33.56] [6, 1, 88, 11.36, 0, 0.0, 1, 11.36] [7, 2, 63, 31.75, 2, 31.75, 0, 0.0] [8, 3, 252, 11.9, 0, 0.0, 3, 11.9] [9, 3, 422, 7.11, 0, 0.0, 3, 7.11] [10, 1, 58, 17.24, 0, 0.0, 1, 17.24]

ตารางที่ 4.4 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

23. PSP Defect Fix Time Report	
- getData	[DLD, Fix Time, 0.00, 17.70, 7.70] [, Total, 0.00, 8.00, 8.00] [, Avg. Fix Time, 0.00, 2.21, 2.21] [, , , ,] [CODE, Fix Time, 20.90, 19.90, 40.80] [, Total, 10.00, 12.00, 22.00] [, Avg. Fix Time, 2.09, 1.66, 1.85] [, , , ,] [TOTALS, Fix Time, 20.90, 37.60, 58.50] [, Total, 10.00, 20.00, 30.00] [, Avg. Fix Time, 2.09, 1.88, 1.95]
24. PSP Percent Injected and Removed by Type Report	
- getData	[20: Syntax, 1, 14, 12.50 %, 63.64 %, 8, 8, 80.00 %, 38.10 %] [50: Interface, 4, 4, 50.00 %, 18.18 %, 2, 6, 20.00 %, 28.57 %] [70: Data, 3, 2, 37.50 %, 9.09 %, 0, 5, 0.00 %, 23.81 %] [80: Function, 0, 2, 0.00 %, 9.09 %, 0, 2, 0.00 %, 9.52 %] [Total, 8, 22, , , 10, 21, ,]
25. PSP Percentage Defects Found by Compile Report	
- getData	[20 : Syntax, 16, 8, 50.00 %] [50 : Interface, 8, 2, 25.00 %] [70 : Data, 5, 0, 0.00 %] [80 : Function, 2, 0, 0.00 %] [Total, 31, 10, 32.26 %]
26. Review Rate	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
27. Review Rate vs Yield	
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
28. Total Cost of Quality	
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
29. Total Defects	
- getDataRange	31.75 94.59 28.30 27.03 40.27 11.36 31.75 11.90 7.11 17.24

4.3.4 การทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Overall

ตารางที่ 4.5 ผลการทดสอบหน่วนย่อยในกลุ่ม Overall

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. Process Quality Index	
- getDataRange	0.49 0.00 1.00 1.00 0.00 21.00 0.00

4.4 Integration Test

จุดประสงค์ เพื่อตรวจสอบความถูกต้องของการทำงานต่างๆ เมื่อมีการ Integrate unit / module เข้าร่วมกัน

4.4.1 ทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Plan Performance

ตารางที่ 4.6 ผลการทดสอบหน่วนย่อยในกลุ่ม Plan Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. Actual Development Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	1.38 0.78 1.58 2.35 1.82 0.18 0.42 2.92 1.57 1.20
2. Actual Size	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	63 74 106 111 149 88 63 252 422 58
3. Percent Compile + Test Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
4. Percent Compile Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	8.43 14.89 31.58 0.71 1.83 18.18 8.00 1.14 0.00 0.00
5. Percent Planning + Postmortem Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	3.61 17.02 24.21 26.24 16.51 54.55 28.00 24.00 14.89 29.17
6. Percent Planning Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 2.13 0.00 2.13 1.83 36.36 12.00 0.00 7.45 1.39

ตารางที่ 4.6 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Plan Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
7. Percent Postmortem Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	3.61 14.89 24.21 24.11 14.68 18.18 16.00 24.00 7.45 27.78
8. Percent Test Time	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	19.28 8.51 11.58 16.31 41.28 45.45 16.00 8.00 8.51 8.33
9. Percent Time in Phase To Date	
- getDataDomain	Planning Design Design Review Code Code Review Compile Test Postmortem
- getDataRange	2.45 7.13 0.00 49.88 0.00 6.19 15.89 18.46
10. Size Estimating Error	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 -50.67 35.90 5.71 -28.02 -9.28 -80.62 0.00 0.00 -79.79
11. Time Estimating Error	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	3.75 -21.67 331.82 147.37 60.29 -8.33 -39.02 0.00 0.00 0.00

4.4.2 การทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Process Performance

ตารางที่ 4.7 ผลการทดสอบหน่วนย่อยในกลุ่ม Process Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. A/FR vs Yield	
- getDataDomain	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
- getDataRange	1.00 7.00 3.00 3.00 6.00 1.00 2.00 3.00 3.00 1.00
- calDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- calDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
2. Productivity	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	45.54 91.47 66.95 47.23 82.02 480.00 151.20 86.40 269.36 48.33
3. Productivity vs A/FR	
- getDataDomain	27.71 23.40 43.16 17.02 43.12 63.64 21.00 9.14 8.51 8.33
- getDataRange	45.54 94.47 66.95 47.23 82.02 480.00 151.20 86.40 269.36 48.33
- calDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
4. Productivity vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- getDataRange	45.54 94.47 66.95 47.23 82.02 480.00 151.20 86.40 269.36 48.33
- calDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
5. Test Defects vs A/FR	
- getDataDomain	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- calDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
6. Test Defect vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- calDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

ตารางที่ 4.7 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Process Performance

7. Yield vs A/FR	
- getDataDomain	1.00 7.00 3.00 3.00 6.00 1.00 2.00 3.00 3.00 1.00
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
- calDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- calDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00

4.4.3 การทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Quality Performance

ตารางที่ 4.7 ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
1. Appraisal Cost of Quality	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
2. Appraisal to Failure Ratio	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
- calDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
3. Compile vs Test Defects	
- getDataDomain	0.00 54.05 28.30 0.00 6.71 0.00 31.75 0.00 0.00 0.00
- getDataRange	31.75 40.54 0.00 27.03 33.56 11.36 0.00 11.90 7.11 17.24
4. CR Review Rate vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
5. Defect Age	
- getDataDomain	1 2 3 4 5 6 7 8
- getDataRange	COMPILE UT CODE UT CODE COMPILE CODE COMPILE CODE COMPILE CODE COMPILE CODE UT CODE UT CODE UT CODE COMPILE CODE COMPILE CODE COMPILE CODE UT DLD UT DLD UT CODE COMPILE DLD UT DLD UT DLD UT DLD UT DLD UT CODE UT CODE COMPILE CODE COMPILE CODE UT CODE UT DLD UT CODE UT CODE UT CODE UT CODE UT
- calDataDomain	1 3 2 2 2 2 3 3 3 2 2 2 3 5 5 2 5 5 5 5 5 3 2 2 3 3 5 3 3 3 3
- calDataRange	1 10 12 0 8 0 0 0

ตารางที่ 4.8 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

6. Defect Fix Time by Type	
- getDataDomain	20 50 70 80
- getDataRange	34.50 10.70 16.20 3.80
7. Defect Injection % by Phase	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	50.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00
8. Defect Removal Yield	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	1.00 7.00 3.00 3.00 6.00 1.00 2.00 3.00 3.00 1.00
- calDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
9. Defects Injected by Phase To Date	
- getDataDomain	Planning Design Design Review Code Code Review Compile Test
- getDataRange	0.00 25.81 0.00 70.97 0.00 3.23 0.00
10. Defects Injected in Code	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	15.87 94.59 28.30 9.01 6.71 11.36 31.75 7.94 7.11 17.24
11. Defects Injected in Design	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 0.00 0.00 18.02 33.56 0.00 0.00 3.94 0.00 0.00
12. Defects Removal % by Phase	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00 100.00
13. Defects Removal Leverage	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 0.76 0.00 0.00 4.50 0.00 0.00 0.00 0.00 0.00

ตารางที่ 4.8 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

14. Defects Removed by Phase To Date	
- getDataDomain	Planning Design Design Review Code Code Review Compile Test
- getDataRange	0.00 0.00 0.00 0.00 32.26 67.74
15. Defects Removed by Type	
- getDataDomain	"20 50 70 80
- getDataRange	16 5 8 2
16. Defects Removed in Code Review	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
17. Defects Remvoed in Compile	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 54.05 28.30 0.00 6.71 0.00 31.75 0.00 0.00 0.00
18. Defects Removed in Design Review	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
19. Defects Removed in Test	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	31.75 40.54 0.00 27.03 33.56 11.36 0.00 11.90 7.11 17.24
20. DLDR Review Rate vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- getDataRange	1.00 7.00 3.00 3.00 6.00 1.00 2.00 3.00 3.00 1.00
- calDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
21. Failure Cost of Quality	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	27.71 23.40 43.16 17.02 43..12 63.64 24.00 9.14 8.51 8.33
22. PSP Defect Density Report	

ตารางที่ 4.8 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

- getData	[1, 2, 63, 31.75, 0, 0.0, 2, 31.75] [2, 7, 74, 94.59, 4, 54.05, 3, 40.54] [3, 3, 106, 28.3, 3, 28.3, 0, 0.0] [4, 3, 111, 27.03, 0, 0.0, 3, 27.03] [5, 6, 149, 40.27, 1, 6.71, 5, 33.56] [6, 1, 88, 11.36, 0, 0.0, 1, 11.36] [7, 2, 63, 31.75, 2, 31.75, 0, 0.0] [8, 3, 252, 11.9, 0, 0.0, 3, 11.9] [9, 3, 422, 7.11, 0, 0.0, 3, 7.11] [10, 1, 58, 17.24, 0, 0.0, 1, 17.24]
23. PSP Defect Fix Time Report	
- getData	[DLD, Fix Time, 0.00, 17.70, 7.70] [, Total, 0.00, 8.00, 8.00] [, Avg. Fix Time, 0.00, 2.21, 2.21] [, , , ,] [CODE, Fix Time, 20.90, 19.90, 40.80] [, Total, 10.00, 12.00, 22.00] [, Avg. Fix Time, 2.09, 1.66, 1.85] [, , , ,] [TOTALS, Fix Time, 20.90, 37.60, 58.50] [, Total, 10.00, 20.00, 30.00] [, Avg. Fix Time, 2.09, 1.88, 1.95]
24. PSP Percent Injected and Removed by Type Report	
- getData	[20: Syntax, 1, 14, 12.50 %, 63.64 %, 8, 8, 80.00 %, 38.10 %] [50: Interface, 4, 4, 50.00 %, 18.18 %, 2, 6, 20.00 %, 28.57 %] [70: Data, 3, 2, 37.50 %, 9.09 %, 0, 5, 0.00 %, 23.81 %] [80: Function, 0, 2, 0.00 %, 9.09 %, 0, 2, 0.00 %, 9.52 %] [Total, 8, 22, , , 10, 21, ,]

ตารางที่ 4.8 (ต่อ) ผลการทดสอบหน่วนย่อยในกลุ่ม Quality Performance

25. PSP Percentage Defects Found by Compile Report	
- getData	[20 : Syntax, 16, 8, 50.00 %] [50 : Interface, 8, 2, 25.00 %] [70 : Data, 5, 0, 0.00 %] [80 : Function, 2, 0, 0.00 %] [Total, 31, 10, 32.26 %]
26. Review Rate	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- calBothRate	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
27. Review Rate vs Yield	
- getDataDomain	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
- getDataRange	0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
28. Total Cost of Quality	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	27.71 23.40 43.16 17.02 43.12 63.64 24.00 9.14 8.51 8.33
29. Total Defects	
- getDataDomain	1 2 3 4 5 6 7 8 9 10
- getDataRange	31.75 94.59 28.30 27.03 40.27 11.36 31.75 11.90 7.11 17.24

4.4.4 การทดสอบหน่วยย่อยของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Overall

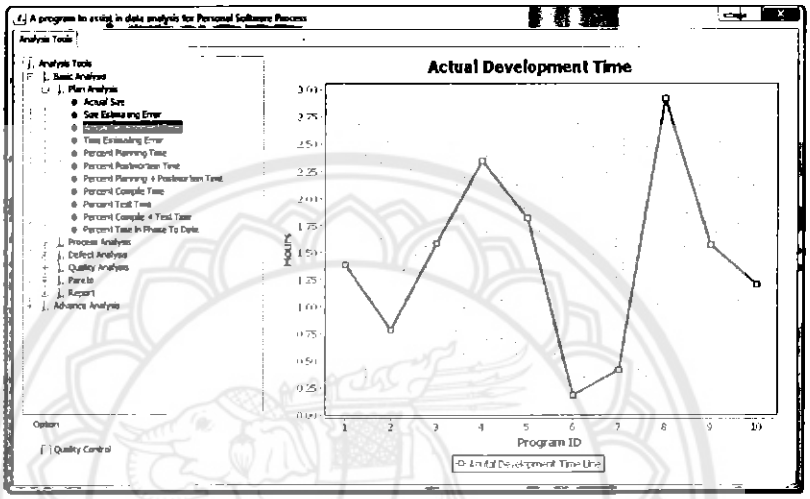
ตารางที่ 4.8 ผลการทดสอบหน่วนย่อยในกลุ่ม Overall

เครื่องมือวิเคราะห์และฟังก์ชัน	ผลของข้อมูลที่คาดว่าจะได้
2. Process Quality Index	
- getDataRange	"0.49 0.00 1.00 1.00 0.00 21.00 0.00"

4.5 System Test

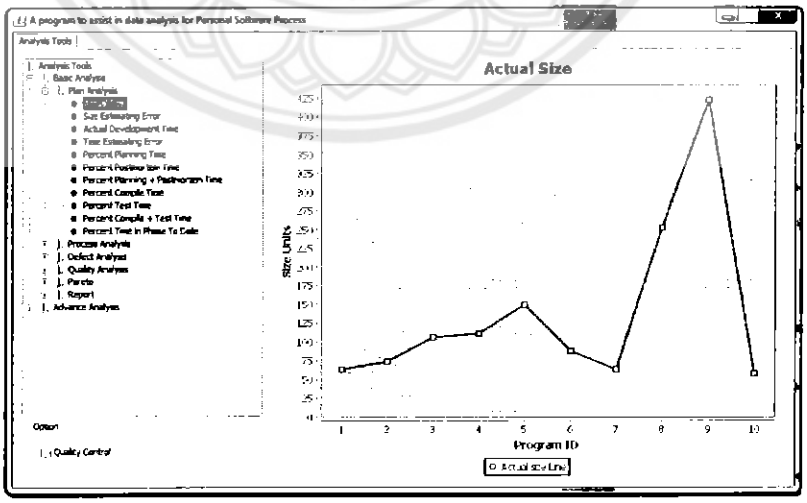
4.5.1 ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Plan Performance

1. Actual Development Time



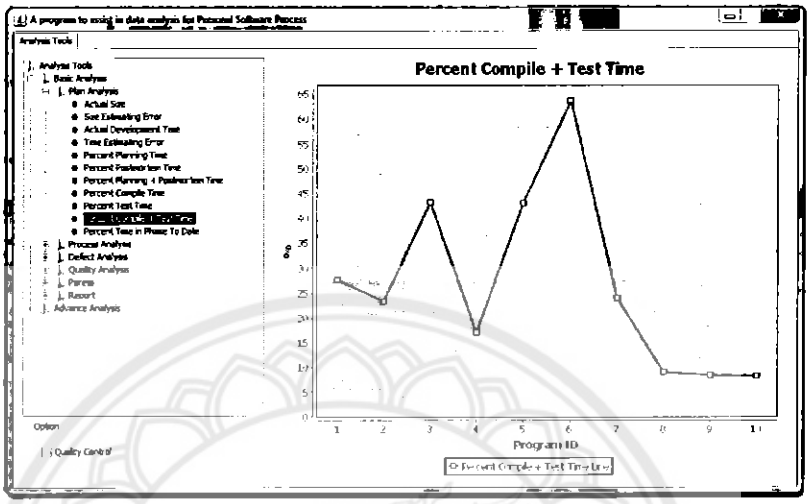
รูปที่ 4.3 Actual Development Time

2. Actual Size



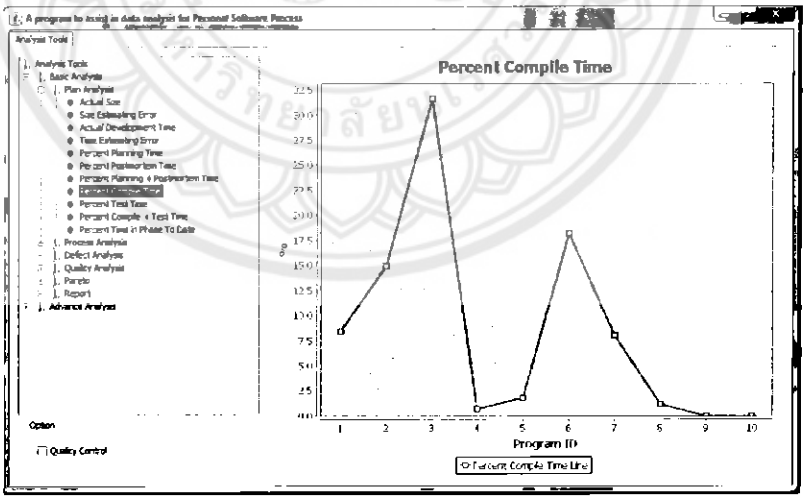
รูปที่ 4.4 Actual Size

3. Percent Compile + Test Time



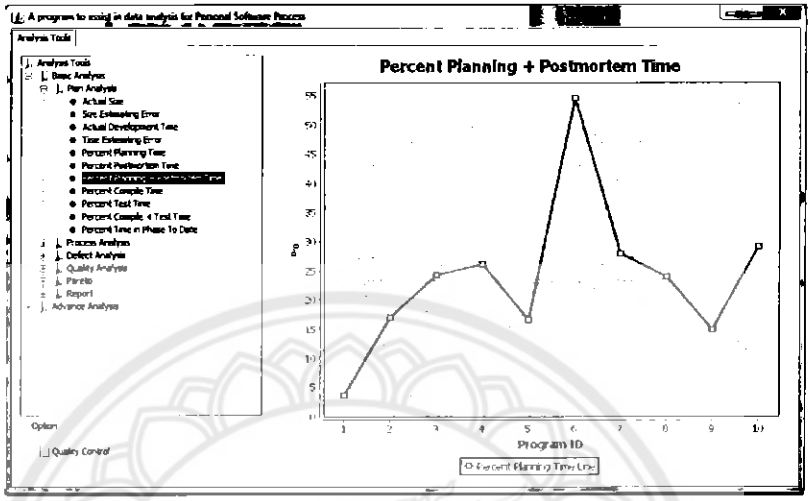
รูปที่ 4.5 Percent Compile + Test Time

4. Percent Compile Time



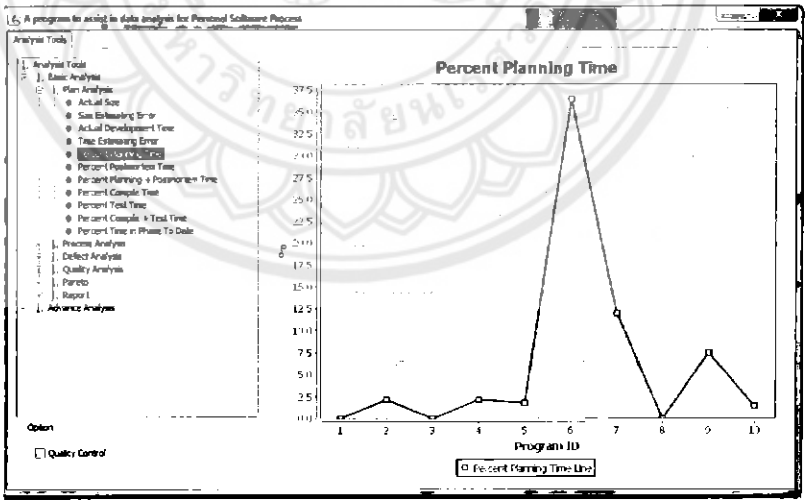
รูปที่ 4.6 Percent Compile Time

5. Percent Planning + Postmortem Time



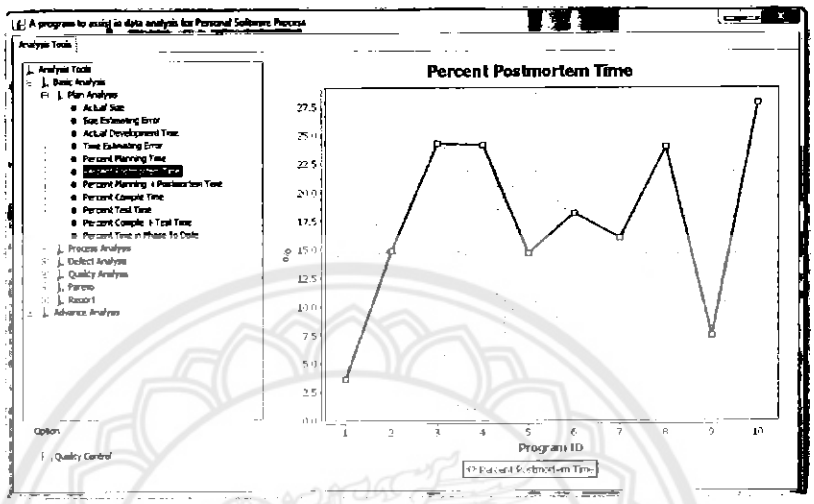
รูปที่ 4.7 Percent Planning + Postmortem Time

6. Percent Planning Time



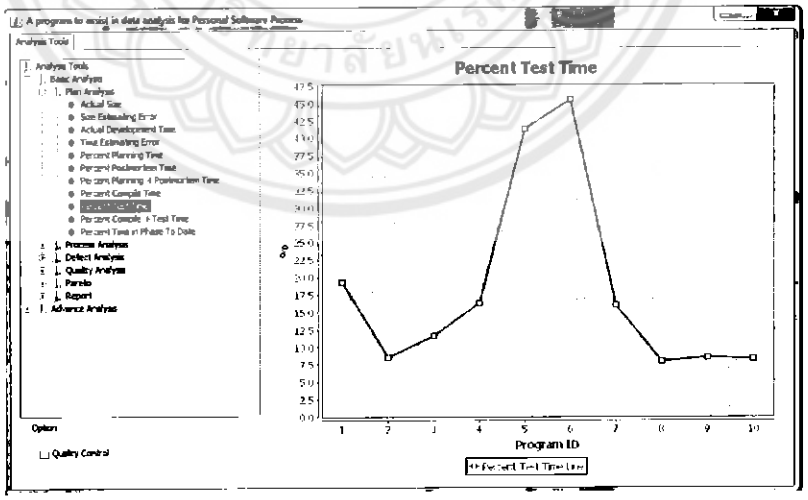
รูปที่ 4.8 Percent Planning Time

7. Percent Postmortem Time



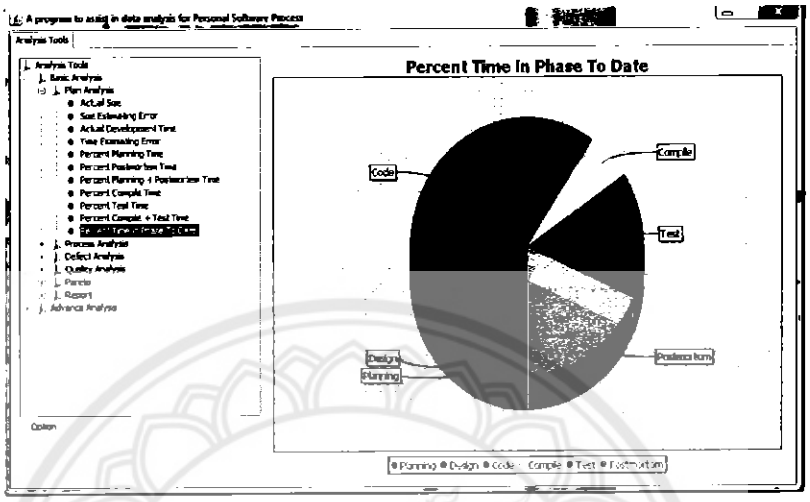
รูปที่ 4.9 Percent Postmortem Time

8. Percent Test Time



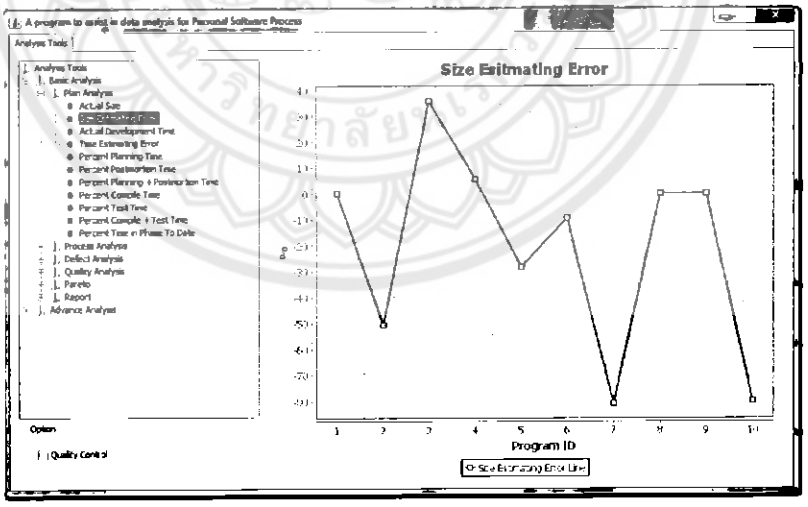
รูปที่ 4.10 Percent Test Time

9. Percent Time in Phase To Date



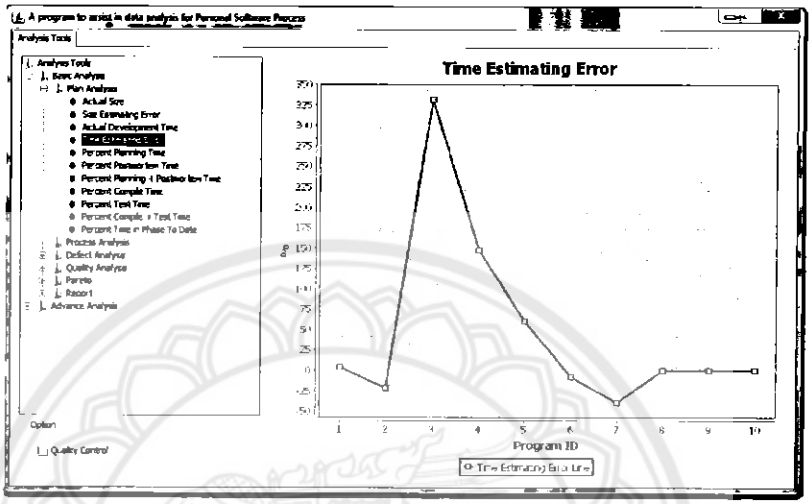
รูปที่ 4.11 Percent Time in Phase To Date

10. Size Estimating Error



รูปที่ 4.12 Percent Time in Phase To Date

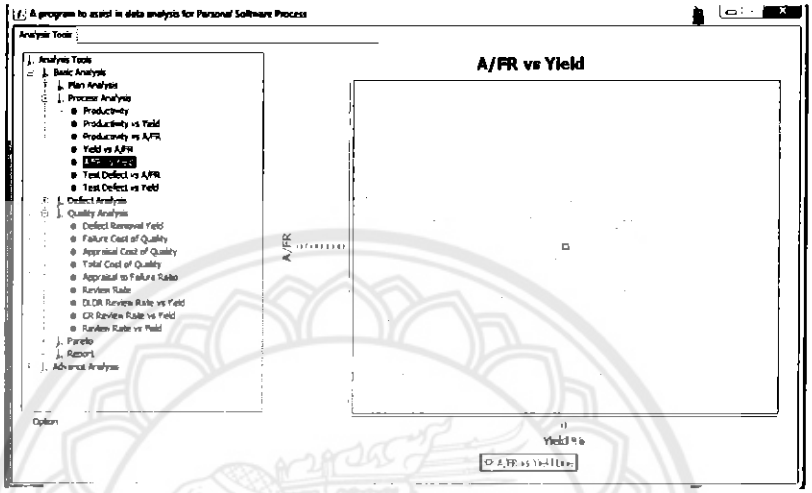
11. Time Estimating Error



รูปที่ 4.13 Time Estimating Error

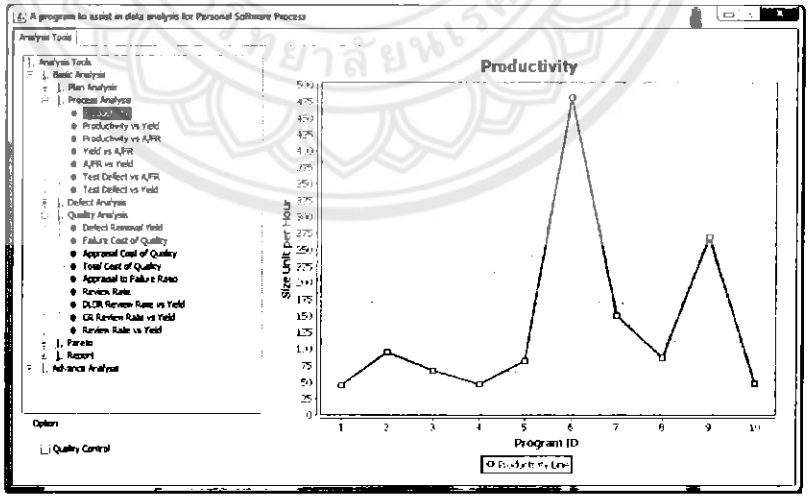
4.5.2 ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Process Performance

1. A/FR vs Yield



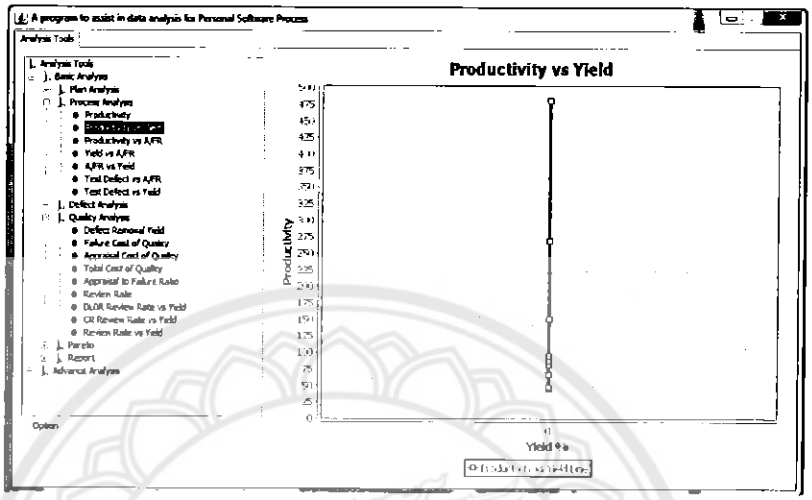
รูปที่ 4.14 A/FR vs Yield

2. Productivity



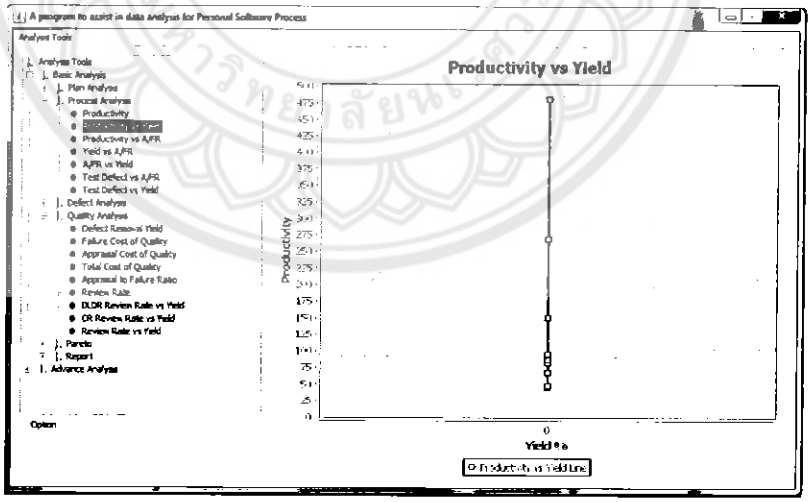
รูปที่ 4.15 Productivity

3. Productivity vs A/FR



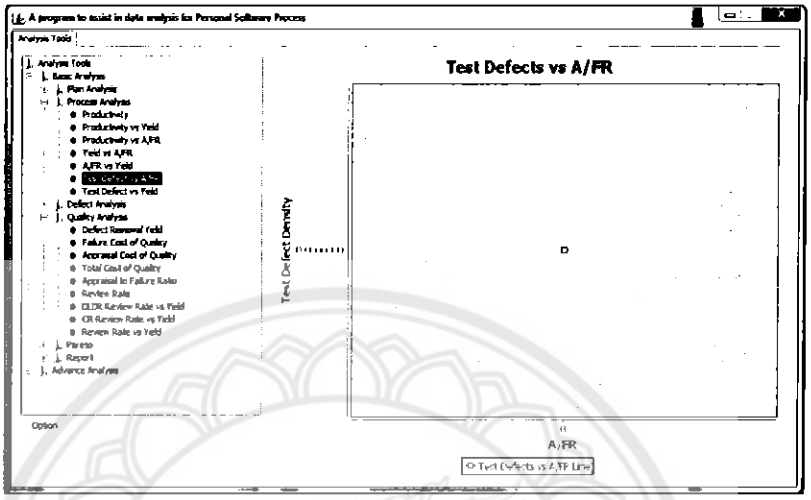
รูปที่ 4.16 Productivity vs A/FR

4. Productivity vs Yield



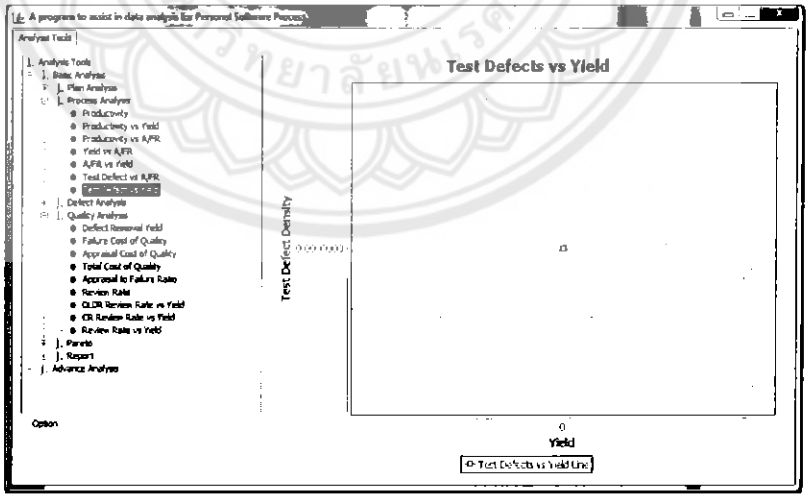
รูปที่ 4.17 Productivity vs Yield

5. Test Defect vs A/FR



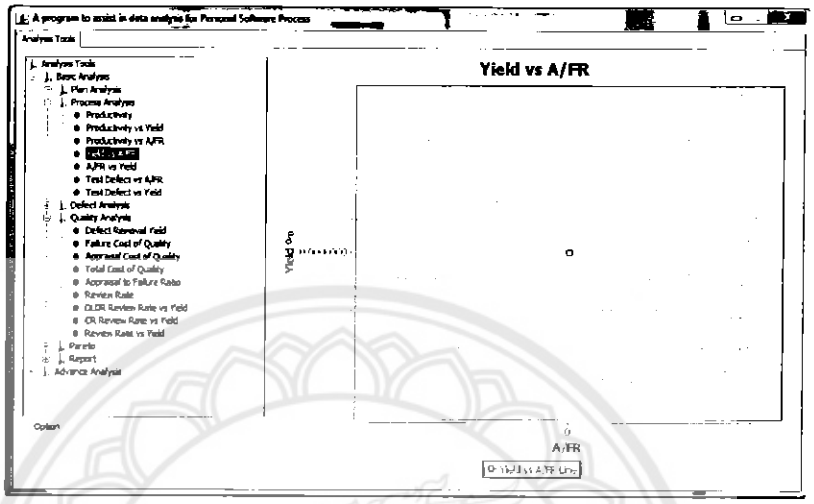
รูปที่ 4.18 Test Defects vs A/FR

6. Test Defect vs Yield



รูปที่ 4.19 Test Defects vs Yield

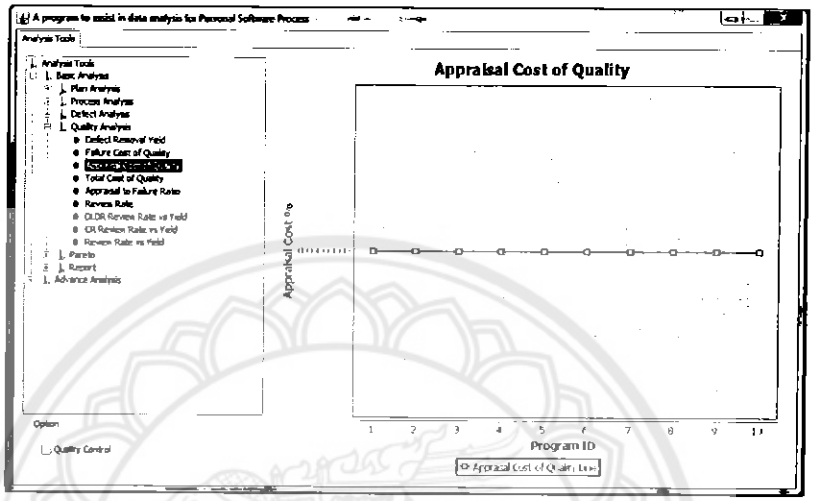
7. Yield vs A/FR



รูปที่ 4.20 Yield vs A/FR

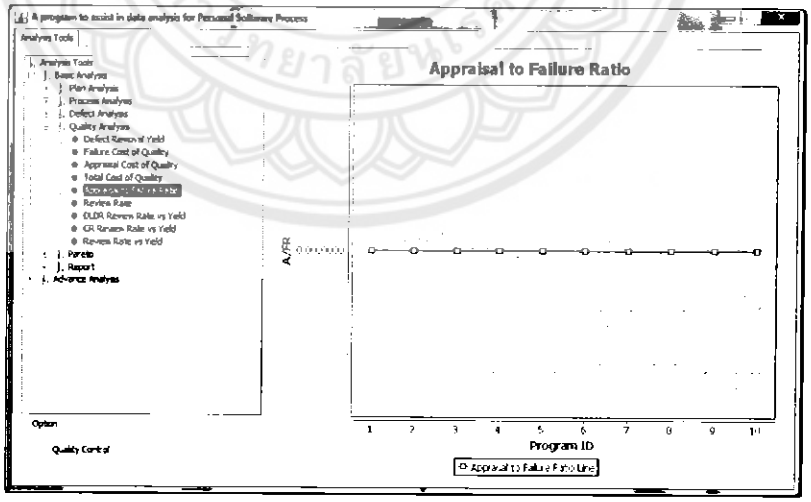
4.5.3 ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Quality Performance

1. Appraisal Cost of Quality



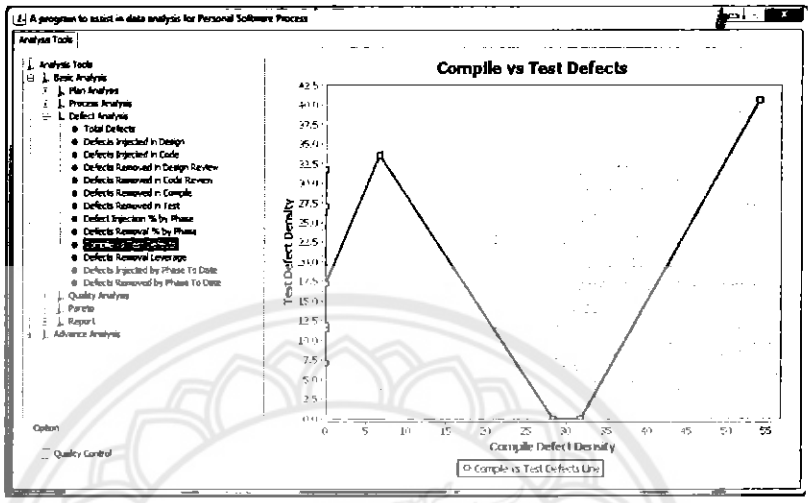
รูปที่ 4.21 Appraisal Cost of Quality

2. Appraisal to Failure Ratio



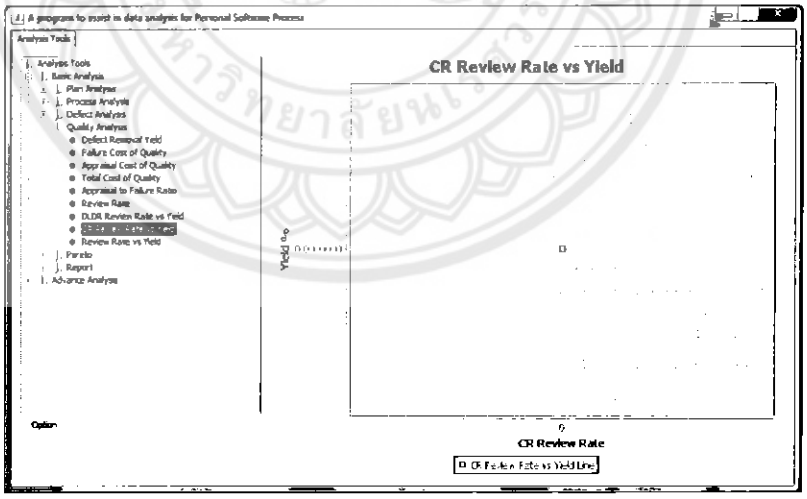
รูปที่ 4.22 Appraisal to Failure Ratio

3. Compile vs Test Defects



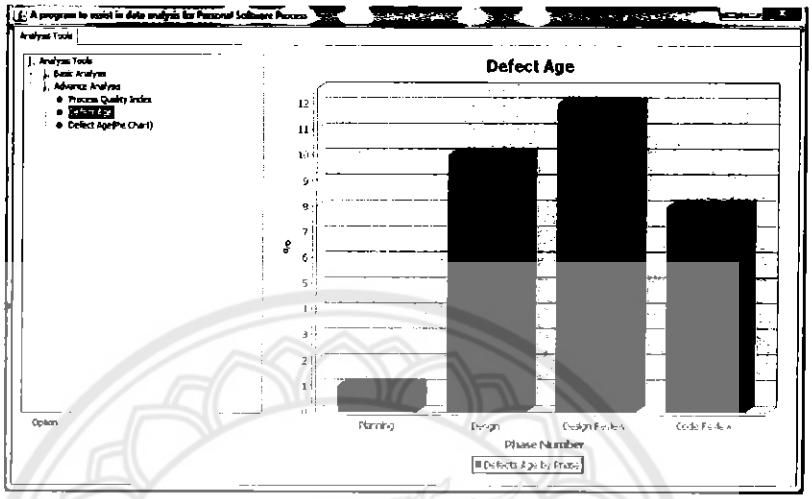
รูปที่ 4.23 Compile vs Test Defects

4. CR Review Rate vs Yield



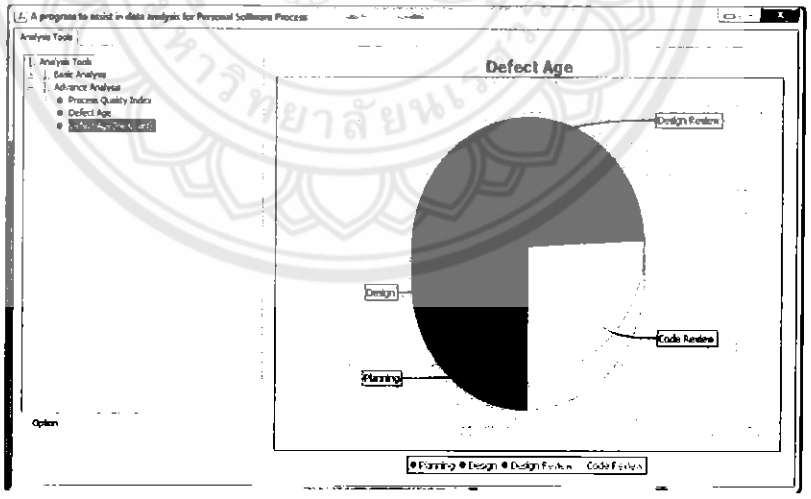
รูปที่ 4.24 CR Review Rate vs Yield

5. Defect Age



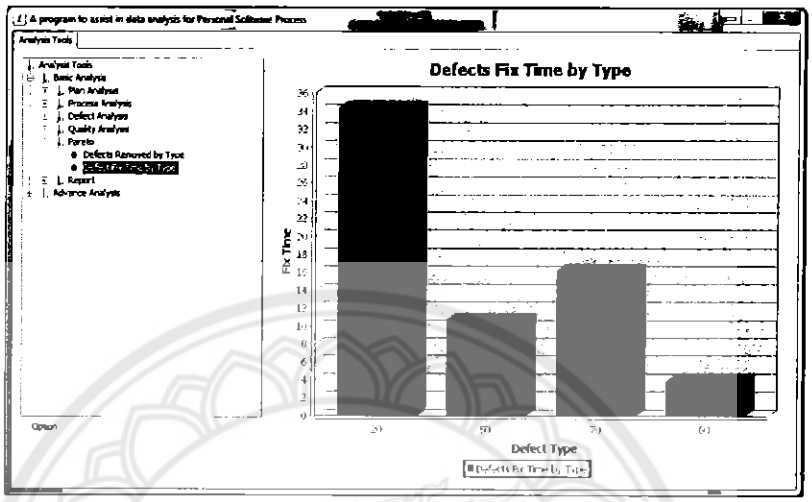
รูปที่ 4.25 Defect Age

6. Defect Age(Pie Chart)



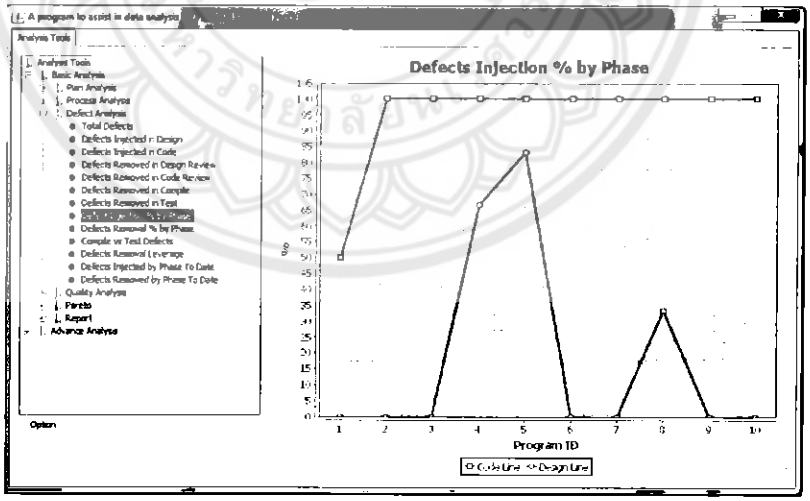
รูปที่ 4.26 Defect Age (Pie Chart)

7. Defect Fix Time by Type



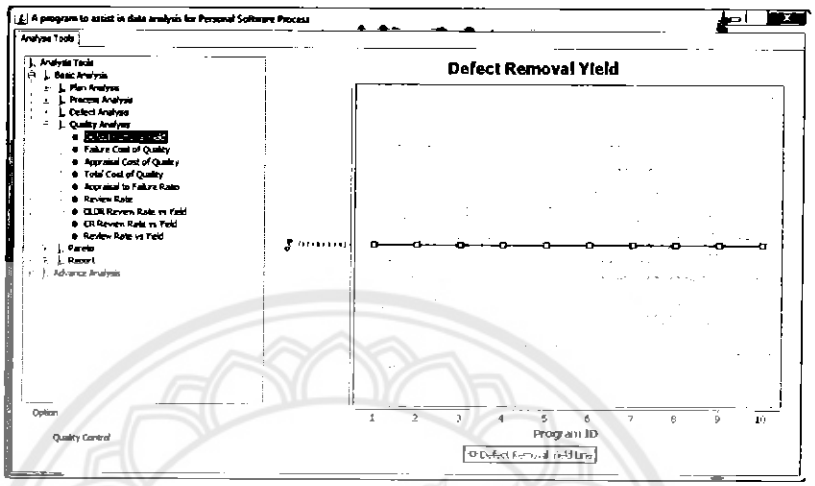
รูปที่ 4.27 Defects Fix Time by Type

8. Defect Injection % by Phase



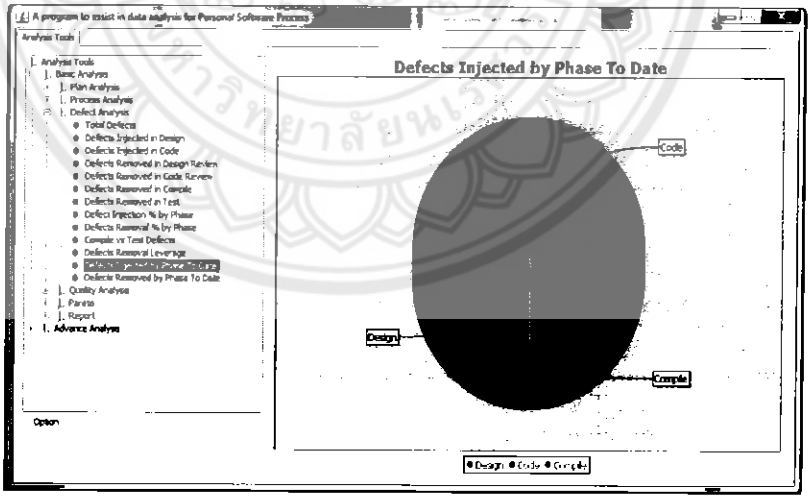
รูปที่ 4.28 Defects Injected % by Phase

9. Defect Removal Yield



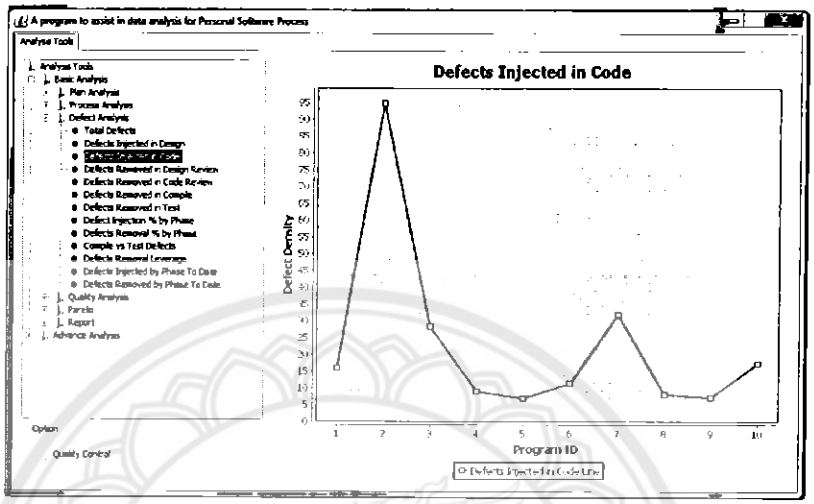
รูปที่ 4.29 Defects Removal Yield

10. Defects Injected by Phase To Date



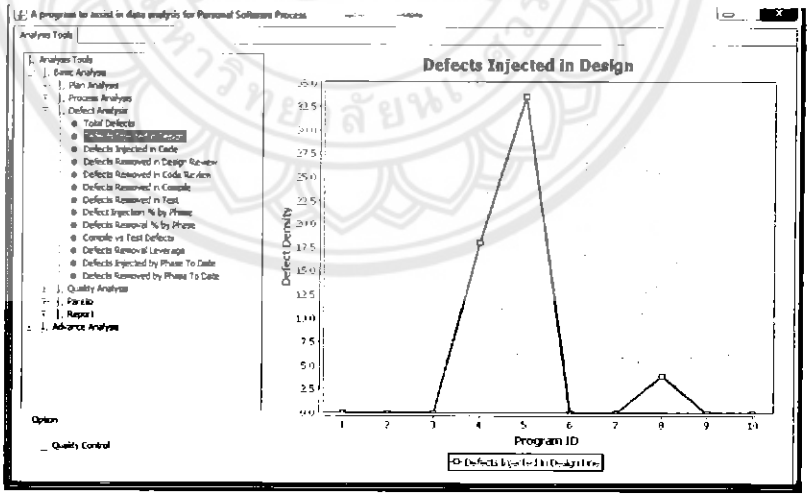
รูปที่ 4.30 Defects Injected by Phase To Date

11. Defects Injected in Code



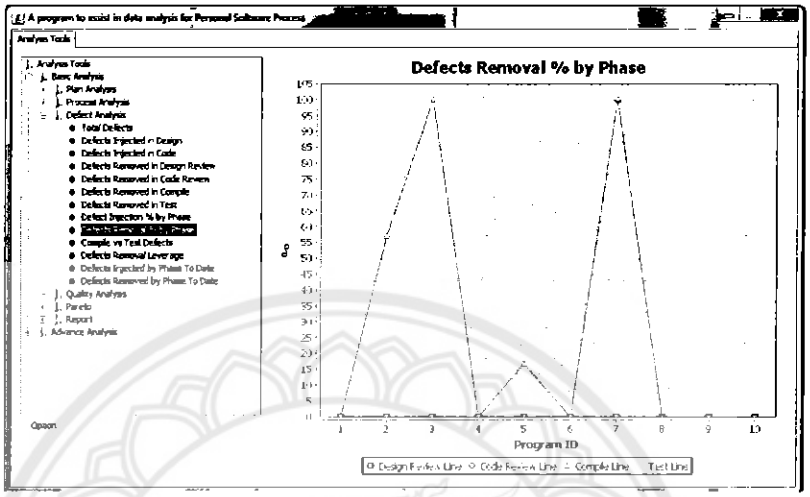
รูปที่ 4.31 Defects Injected in Code

12. Defects Injected in Design



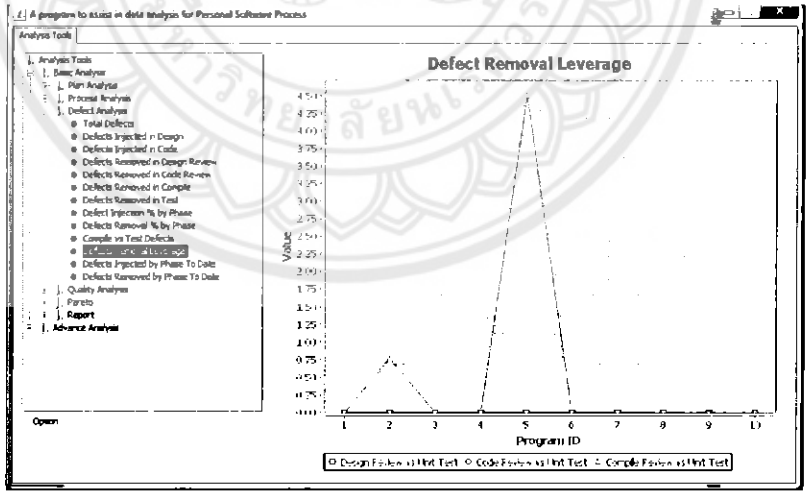
รูปที่ 4.32 Defects Injected InDesign

13. Defects Removal % by Phase



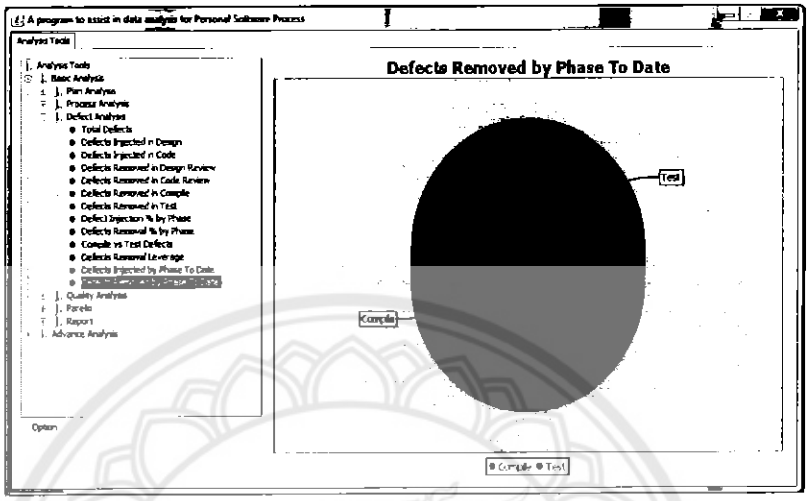
รูปที่ 4.33 Defects Removal % by Phase

14. Defect Removal Leverage



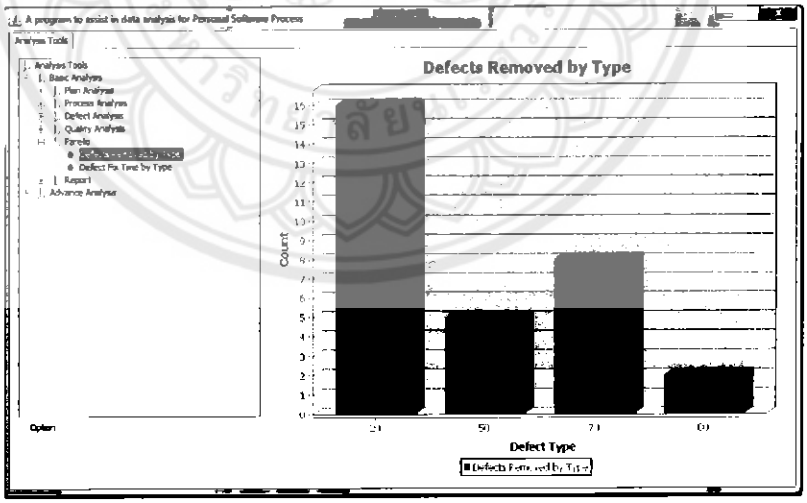
รูปที่ 4.34 Defects Removal Leverage

15. Defects Removed by Phase To Date



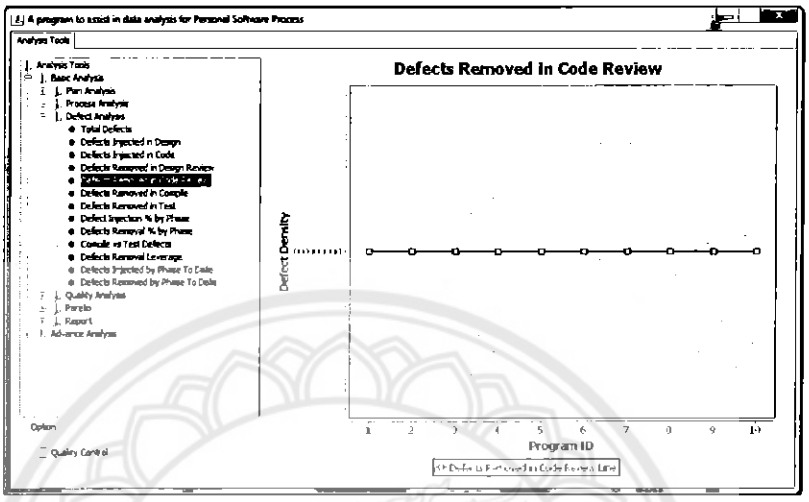
รูปที่ 4.35 Defects Removed by Phase To Date

16. Defects Removed by Type



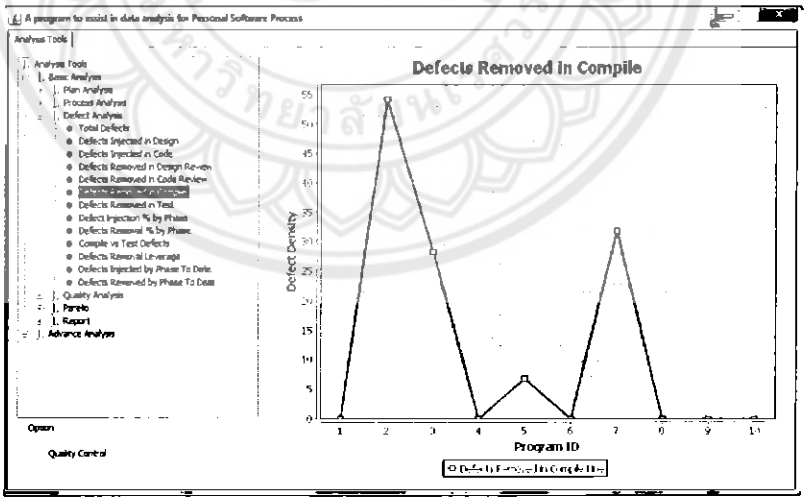
รูปที่ 4.36 Defects Removed by Type

17. Defects Removed in Code Review



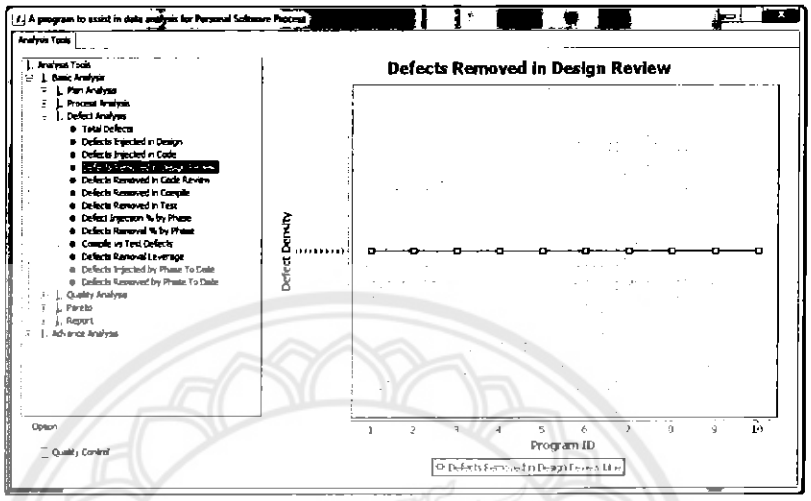
รูปที่ 4.37 Defects Removed in Code Review

18. Defects Removed in Compile



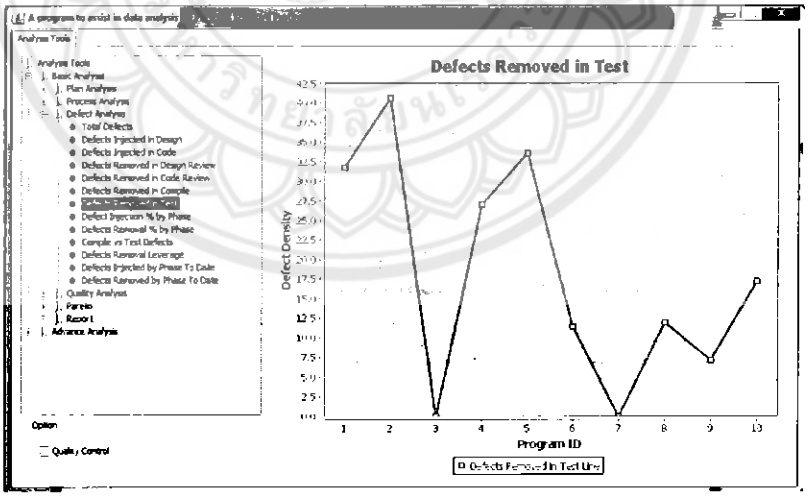
รูปที่ 4.38 Defects Removed in Compile

19. Defects Removed in Design Review



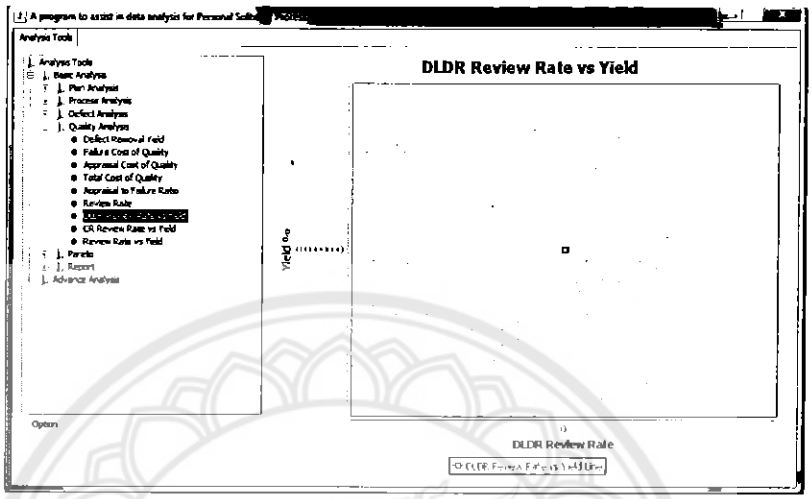
รูปที่ 4.39 Defects Removed in Design Review

20. Defects Removed in Test



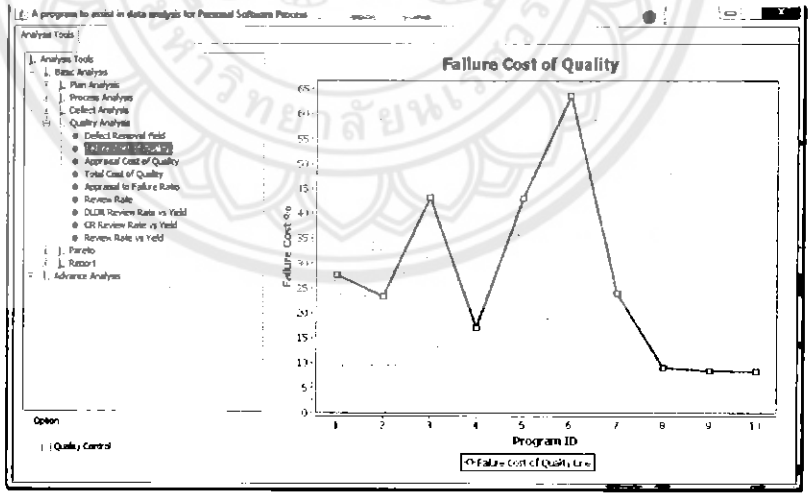
รูปที่ 4.40 Defects Removed in Test

21. DLDR Review Rate vs Yield



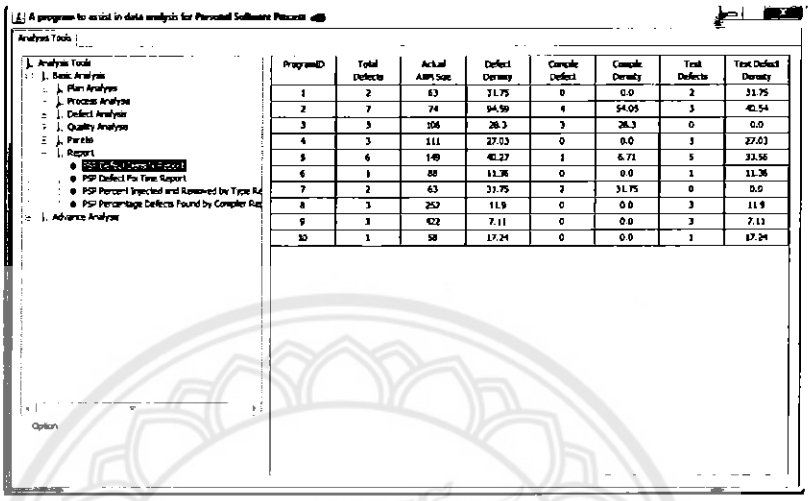
รูปที่ 4.41 DLDR Review Rate vs Yield

22. Failure Cost of Quality



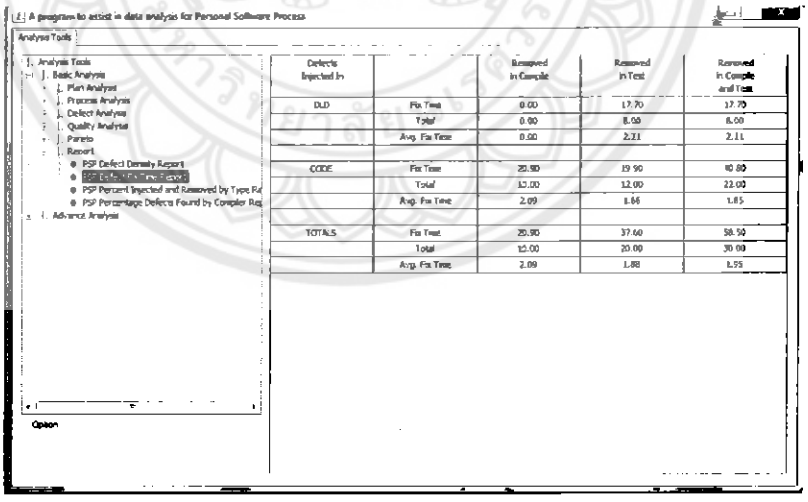
รูปที่ 4.42 Failure Cost of Quality

23. PSP Defect Density Report



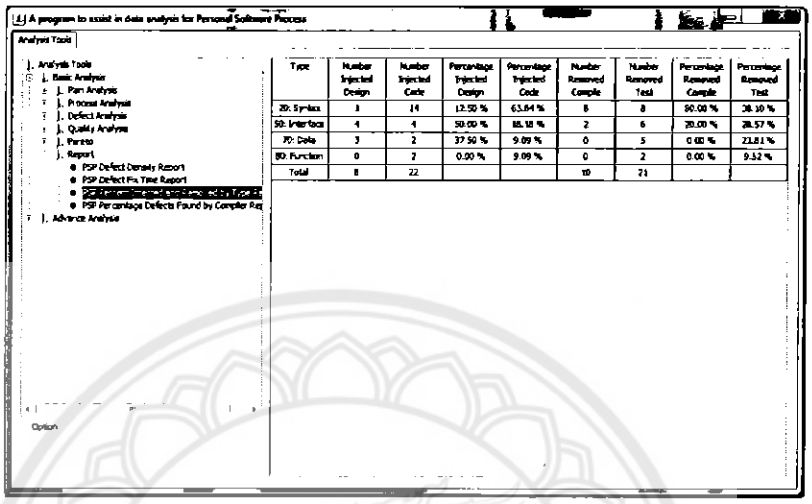
รูปที่ 4.43 PSP Defect Density Report

24. PSP Defect Fix Time Report



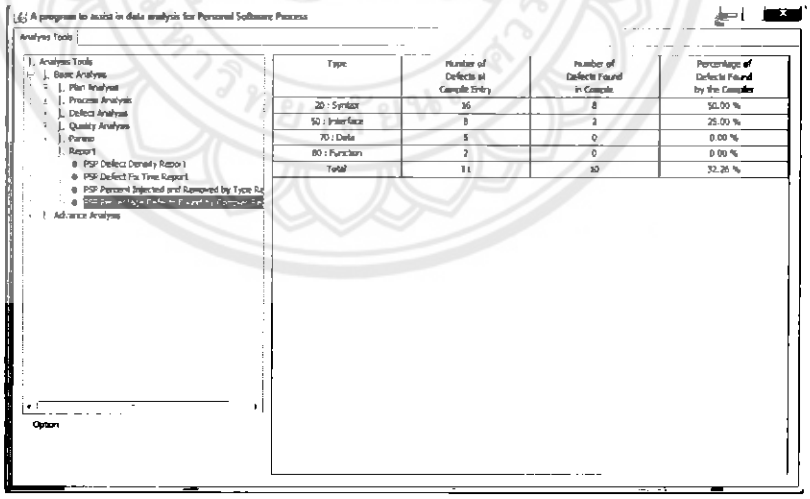
รูปที่ 4.44 PSP Defect Fix Time Report

25. PSP Percent Injected and Removed by Type Report



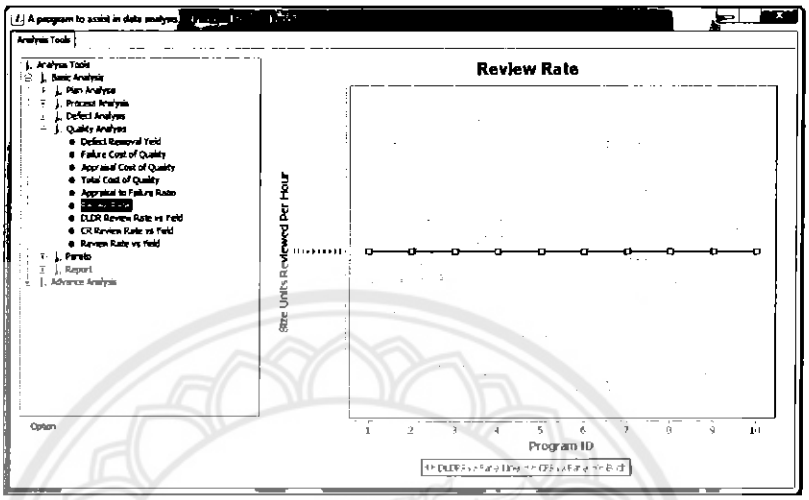
รูปที่ 4.45 PSP Percent Injected and Removed by Type Report

26. PSP Percentage Defects Found by Compile Report



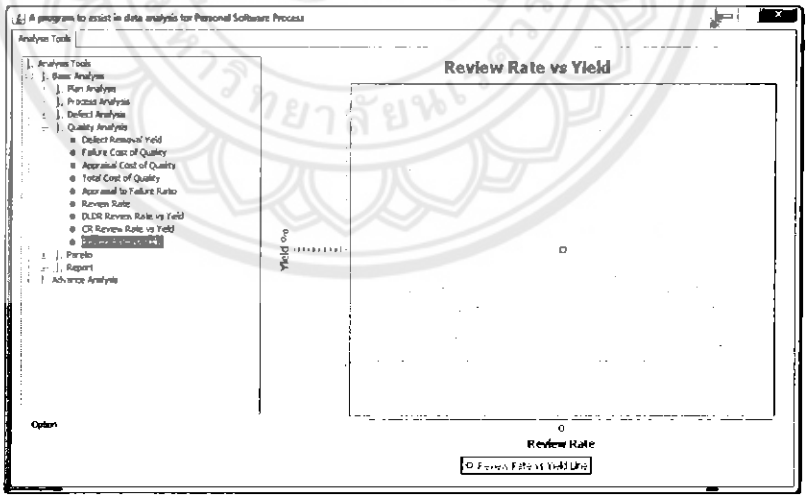
รูปที่ 4.46 PSP Percentage Defects Found by Compile Report

27. Review Rate



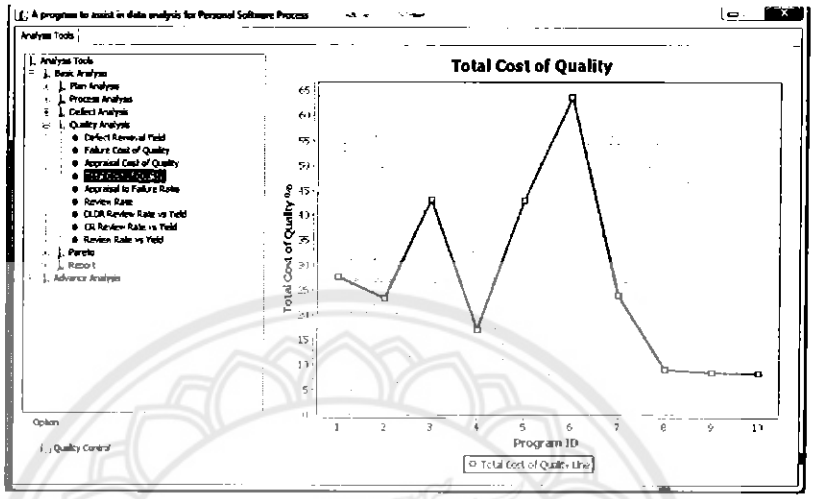
รูปที่ 4.47 Review Rate

28. Review Rate vs Yield



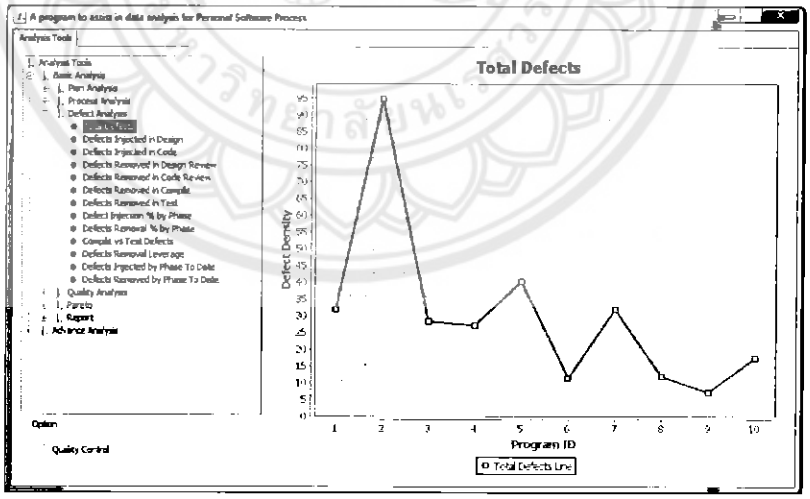
รูปที่ 4.48 Review Rate vs Yield

29. Total Cost of Quality



รูปที่ 4.49 Total Cost of Quality

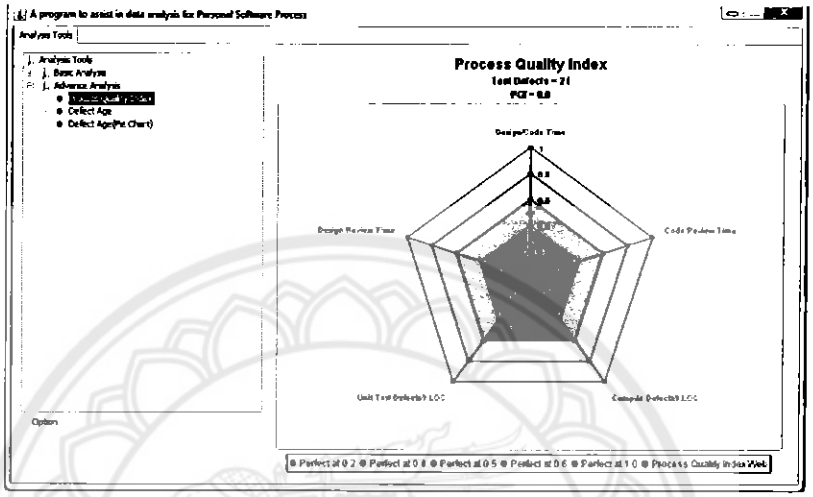
30. Total Defects



รูปที่ 4.50 Total Defects

4.5.4 ทดสอบการแสดงผลของเครื่องมือช่วยวิเคราะห์ข้อมูลในกลุ่ม Overall

1. Process Quality Index



รูปที่ 4.51 Process Quality Index

4.6 แจกแจงเครื่องมือวิเคราะห์ข้อมูล

ตารางที่ 4.9 แจกแจงเครื่องมือวิเคราะห์ข้อมูล

เครื่องมือช่วยวิเคราะห์ข้อมูล PSP	โปรแกรมที่พัฒนา	PSP Student Workbook	PSP Dashboard	หนังสือ PSP
1. เครื่องมือวิเคราะห์ในกลุ่ม Plan Performance				
- Actual Development Time	✓	✓	✓	✓
- Actual Size	✓	✓	✓	✓
- Percent Planning + Postmortem Time	✓	✓	✓	✓
- Percent Planning Time	✓	✓	✓	✓
- Percent Postmortem Time	✓	✓	✓	✓
- Size Estimating Error	✓	✓	✓	✓
- Time Estimating Error	✓	✓	✓	✓
- Percent Compile + Test Time	✓	✓	✓	✓
- Percent Compile Time	✓	✓	✓	✓
- Percent Test Time	✓	✓	✓	✓
- Percent Time in Phase To Date	✓	✗	✓	✓
2. เครื่องมือวิเคราะห์ในกลุ่ม Process Performance				
- Productivity	✓	✓	✓	✓
- Productivity vs Yield	✓	✓	✓	✓
- Productivity vs A/FR	✓	✓	✓	✓
- Yield vs A/FR	✓	✓	✓	✓
- A/FR vs Yield	✓	✓	✓	✓
- Test Defect vs A/FR	✓	✓	✓	✓
- Test Defect vs Yield	✓	✓	✓	✓

ตารางที่ 4.10 (ต่อ) แจกแจงเครื่องมือวิเคราะห์ข้อมูล

3. เครื่องมือวิเคราะห์ในกลุ่ม Quality Performance				
- Total Defects	✓	✓	✓	✓
- Defects Injected in Design	✓	✓	✓	✓
- Defects Injected in Code	✓	✓	✓	✓
- Defects Removed in Design Review	✓	✓	✓	✓
- Defects Removed in Code Review	✓	✓	✓	✓
- Defects Removed in Compile	✓	✓	✓	✓
- Defects Removed in Test	✓	✓	✓	✓
- Defect Injection % by Phase	✓	✓	✓	✓
- Defects Removal % by Phase	✓	✓	✓	✓
- Compile vs Test Defects	✓	✓	✓	✓
- Defects Removal Leverage	✓	×	✓	✓
- Defects Injected by Phase To Date	✓	×	✓	✓
- Defects Removed by Phase To Date	✓	×	✓	✓
- Defect Removal Yield	✓	✓	✓	✓
- Failure Cost of Quality	✓	✓	✓	✓
- Appraisal Cost of Quality	✓	✓	✓	✓
- Total Cost of Quality	✓	✓	✓	✓

ตารางที่ 4.10 (ต่อ) แจกแจงเครื่องมือวิเคราะห์ข้อมูล

- Appraisal to Failure Ratio	✓	✓	✓	✓
- Defect Removal Yield	✓	✓	✓	✓
- Failure Cost of Quality	✓	✓	✓	✓
- Appraisal Cost of Quality	✓	✓	✓	✓
- Total Cost of Quality	✓	✓	✓	✓
- Appraisal to Failure Ratio	✓	✓	✓	✓
- Review Rate	✓	✓	✓	✓
- DLDR Review Rate vs Yield	✓	✓	✓	✓
- CR Review Rate vs Yield	✓	✓	✓	✓
- Review Rate vs Yield	✓	✓	✓	✓
- Defects Removed by Type	✓	✓	✓	✓
- Defect Fix Time by Type	✓	✓	✓	✓
- PSP Defect Density Report	✓	✓	✓	✓
- PSP Defect Fix Time Report	✓	✓	✓	✓
- PSP Percent Injected and Removed by Type Report	✓	✓	✓	✓
- PSP Percentage Defects Found by Compile Report	✓	✓	✓	✓
- Defect Age (Pie Chart)	✓	✗	✗	✓
4. เครื่องมือวิเคราะห์ในกลุ่ม Overall				
- Process Quality Index	✓	✗	✗	✓

บทที่ 5
สรุปผล

โครงงานนี้ได้ศึกษาแนวคิดของกระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal Software Process; PSP) และสร้างเครื่องมือช่วยวิเคราะห์ข้อมูล PSP ซึ่งแบ่งสมรรถนะได้ 4 ด้าน คือ การวางแผน กระบวนการ คุณภาพ และภาพรวม โดยแต่ละด้านจะแสดงผลในรูปแบบต่างๆ เช่น แผนภูมิเส้น แผนภูมิวงกลม หรือตาราง เป็นต้น เพื่อแสดงให้เห็นถึงสมรรถนะในด้านต่างๆได้ชัดเจนยิ่งขึ้น

โครงงานนี้ใช้ภาษาจาวา (Java) ในการพัฒนาโปรแกรม เนื่องจากเป็นภาษาที่ง่ายต่อการพัฒนาโปรแกรมและการออกแบบหน้าตาของโปรแกรมแบบกราฟิก (Graphic User Interface; GUI) รวมทั้งยังสามารถพัฒนาร่วมกับไลบรารี JFreeChart ซึ่งเป็นชุดเครื่องมือที่ช่วยในการสร้างแผนภูมิชนิดต่างๆได้เป็นอย่างดี

5.1 ผลการทดสอบ

ตารางที่ 5.1 ผลการทดสอบ

เครื่องมือช่วยวิเคราะห์ข้อมูล PSP	จำนวนเครื่องมือวิเคราะห์ข้อมูล PSP		
	โปรแกรมที่พัฒนา	PSP Student Workbook	PSP Dashboard
1. ด้านการวางแผน	11	10	11
2. ด้านกระบวนการ	7	7	7
3. ด้านคุณภาพ	30	25	28
4. ภาพรวม	1	-	-

5.2 ปัญหาและอุปสรรค

ตารางที่ 5.2 ปัญหาและอุปสรรค

ปัญหาและอุปสรรค	แนวทางแก้ไข
1. ขาดความชำนาญในการเขียนโปรแกรมด้านภาษาจาวา (Java)	1. ศึกษาภาษาจาวา (Java) โดยเน้นศึกษาในเรื่องการเชื่อมต่อฐานข้อมูล เอสคิวแอล (SQL) ความสัมพันธ์กันของคลาสในรูปแบบของอินเตอร์เฟส
2. เนื่องจากการสร้างแผนภูมิในรูปแบบต่างๆ มีการใช้ไลบรารีเสริม คือ JFreeChart ซึ่งเป็นชุดเครื่องที่มีความซับซ้อนในการทำความเข้าใจ	2. ศึกษาเอกสารของไลบรารี JFreeChart หรือสืบค้นจากแหล่งข้อมูลอื่นๆ เช่น http://www.jfree.org/jfreechart/ เป็นต้น
3. การศึกษาแนวคิดของกระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal Software Proces; PSP) นั้นเป็นเรื่องยาก เนื่องจากเอกสารส่วนใหญ่เป็นภาษาต่างประเทศและมีรายละเอียดค่อนข้างเยอะ	3. ศึกษาโครงสร้างภาษาต่างประเทศนั้นเพื่อสร้างความเข้าใจในหลักภาษา และพยายามแปล

5.3 ข้อเสนอแนะ

จากปัญหาและอุปสรรคที่พบจากหัวข้อ 5.2 ผู้จัดทำจึงมีข้อเสนอแนะสำหรับผู้ที่ต้องการนำไปพัฒนาต่อ ควรจะมีความรู้ในสิ่งต่อไปนี้

1. ควรมีความรู้ในภาษาจาวา (Java) ในเรื่องการเชื่อมต่อฐานข้อมูล เอสคิวแอล (SQL)
2. ควรศึกษา Strategy Pattern เพื่อลดความซับซ้อนของเขียนฟังก์ชันการทำงาน
3. ควรมีความรู้ในไลบรารี JFreeChart เนื่องจากตัวไลบรารีสร้างสามารถแสดงผลได้ในหลายรูปแบบ เช่น ในรูปแบบไฟล์ รูปภาพ เป็นต้น
4. ควรศึกษาแนวคิดของกระบวนการพัฒนาซอฟต์แวร์ส่วนบุคคล (Personal Software Process; PSP) ให้ดี

5.4 แนวทางในการพัฒนาต่อยอด

1. เพิ่มเครื่องมือช่วยวิเคราะห์ข้อมูล
2. สามารถเลือกการวิเคราะห์ในด้านต่างๆได้เอง
3. สามารถแสดงผลได้หลายรูปแบบ เช่น ในรูปแบบรูปภาพ เป็นต้น
4. สามารถทำงานผ่าน Web
5. ทำงานบนมือถือ

5.5 สรุป

จากผลการทดสอบโปรแกรมช่วยวิเคราะห์ข้อมูล PSP โดยใช้ภาษา Java สามารถแสดงผลการวิเคราะห์ข้อมูลได้อย่างถูกต้องและแสดงผลได้ในหลายรูปแบบ เช่น แผนภูมิเส้น แผนภูมิแท่ง แผนภูมิใยแมงมุม และตาราง เป็นต้น โดยทำงานร่วมกับไลบรารี JFreeChart ซึ่งช่วยในการสร้างแผนภูมิชนิดต่างๆ โปรแกรมช่วยวิเคราะห์ข้อมูล PSP ที่พัฒนาขึ้นครอบคลุมเครื่องมือวิเคราะห์ข้อมูลที่มีอยู่ในโปรแกรม PSP Student Workbook และ PSP Dashboard รวมทั้งยังมีส่วนเพิ่มเติม ดังนี้

1. Defect Age แสดงความถี่ของข้อผิดพลาดที่ค้างอยู่ในแต่ละช่วงเวลา
2. Process Quality Index คุณภาพในการทำงาน
3. Quality Control แสดงขอบเขตการควบคุมที่อยู่ในช่วงที่ยอมรับได้

เอกสารอ้างอิง

[1] พรภัทร์ ศิริธรรมกุล. (2008, October) [Online]. www.squared.chula.ac.th

[2] Watts S. Humphrey, *The Personal Software Process*, 1st ed. Massachusetts, United States of America: Addison-Wesley, 2005.

ประวัติผู้เขียนโครงงาน

ชื่อ ปิยะพงษ์ ฐานะตระกูล

ภูมิลำเนา 100/6 ถ.เจษฎาบดินทร์ ต.ท่าอิฐ อ.เมืองอุตรดิตถ์ จ.อุตรดิตถ์ 53000

ประวัติการศึกษา

- จบมัธยมศึกษาจากโรงเรียนอุตรดิตถ์ จังหวัดอุตรดิตถ์
- ปัจจุบันกำลังศึกษาอยู่ระดับปริญญาตรีชั้นปีที่ 4 สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

E-mail: cara_thanatrakul@hotmail.com

thanatrakul@gmail.com