บทที่ 4

การแก้ไขปัญหาเรื่องเวลาการทำงานของโปรแกรมภาษา ABAP/4

ในบทนี้จะกล่าวถึงการวิเคราะห์หาสาเหตุของปัญหาและเสนอแนวทางการแก้ไขปัญหา รวมทั้งการลงมือการแก้ไขปัญหา จัดทำเอกสารบัญชีสำหรับตรวจสอบประสิทธิภาพการทำงานของโปรแกรม (Performance Checklist) และนำเสนอผลการแก้ไข

4.1 ข้อมูลข้อบกพร่องของโปรแกรม

ข้อบกพร่องของโปรแกรมแบ่งเป็น 2 กลุ่ม ตามระดับความรุนแรง ได้แก่ ความรุนแรงระดับที่ 1 คือ โปรแกรมสามารถประมวลผลได้สำเร็จแต่ใช้เวลาการทำงานมากกว่า 1 วัน และความรุนแรงระดับที่ 2 คือ โปรแกรมไม่สามารถประมวลผลได้สำเร็จ

ผู้ใช้งานแจ้งข้อมูลโปรแกรมที่มีปัญหามีทั้งหมด 12 โปรแกรม ซึ่งโปรแกรมเหล่านี้เคยทำงานได้ตามที่ผู้ใช้งานคาดหวัง โปรแกรมที่อยู่ในกลุ่มข้อบกพร่องที่มีความรุนแรงระดับที่ 1 มีด้วยกัน 9 โปรแกรม และโปรแกรมที่อยู่ในกลุ่มข้อบกพร่องที่มีความรุนแรงระดับที่ 2 ซึ่งเป็นระดับที่รุนแรงที่สุด มีจำนวน 3 โปรแกรม

การเก็บข้อมูลข้อบกพร่องของกรณีศึกษานี้ไม่สามารถเก็บในลักษณะความถี่ในการเกิดข้อบกพร่องนั้นๆ เพื่อนำมาพิจารณาแนวโน้มของการเกิดข้อบกพร่องต่างๆ และจัดทำรูปแบบของการเกิดปัญหาได้ การเกิดปัญหานี้เป็นปัญหาเร่งด่วนที่ต้องได้รับการแก้ไขในทันที เนื่องจากมีผลกระทบค่อนข้างมากต่อการทำงานของผู้ใช้ ซึ่งสอดคล้องกับการบำรุงรักษาหลังการเกิดเหตุหรือเกิดการขัดข้องจึงดำเนินการซ่อมบำรุง (Breakdown Maintenance) จะต้องทำการตรวจสอบและวิเคราะห์สาเหตุอย่างเร่งด่วน เพื่อลดความสูญเสียจากการขัดข้อง อย่างไรก็ตามผู้ให้คำปรึกษาขอเสนอแนะว่าข้อมูลของโปรแกรมที่เกิดปัญหาเหล่านี้ควรมีการบันทึกไว้เป็น เพื่อใช้ในการบำรุงรักษาต่อไป

4.2 การวิเคราะห์สาเหตุของปัญหาด้วยแผนผังสาเหตุและผลและเสนอแนวทางการแก้ไข

4.2.1 การวิเคราะห์สาเหตุของปัญหาด้วยแผนผังสาเหตุและผล

การวิเคราะห์หาสาเหตุที่ทำให้เกิดปัญหาตามหลักการของแผนผังสาเหตุและผล จากปัจจัยที่เกี่ยวข้องในการพัฒนาโปรแกรม 3 ปัจจัยหลัก ได้แก่ บุคคลากร เครื่องมือ และกระบวนการ โดยสาเหตุจากแต่ละปัจจัยได้มาจากการระดมสมองของทุกคนในทีมงานพัฒนาระบบ ส่วนงานการพัฒนาโปรแกรม

ปัจจัยแรก คือ บุคคลากร สาเหตุที่น่าจะเกิดจากบุคคลากร ได้แก่ (1) ผู้ออกแบบการทำงานของโปรแกรม ทำการออกแบบโปรแกรมโดยไม่ทราบถึงผลกระทบต่อประสิทธิภาพการทำงานของโปรแกรม (2) ผู้พัฒนาโปรแกรม พัฒนาโปรแกรมโดยไม่ทราบหลักการของการพัฒนาโปรแกรมให้ทำงานอย่างมีประสิทธิภาพ (3) ผู้พัฒนาโปรแกรมมีประสบการณ์ค่อนข้างน้อยในการพัฒนาโปรแกรม (4) บุคคลากรผู้รับผิดชอบในแต่ละขั้นตอนการทำงาน ละเลยที่จะปฏิบัติตามขั้นตอนของกระบวนการนั้นๆ

ปัจจัยที่ 2 เครื่องมือที่เกี่ยวข้องกับการพัฒนาโปรแกรม ได้แก่ ระบบเครือข่าย เครื่อง Application server และเครื่อง Database server เป็นต้น เครื่องมือที่กล่าวมานี้จัดเป็นเครื่องมือส่วนกลาง หากเกิดปัญหาขึ้น ผู้ใช้ระบบทุกคนจะไม่สามารถทำงานได้เลย ซึ่งจะต้องแจ้งให้ทางทีมพัฒนาระบบส่วนงานเครือข่ายเข้ามาเป็นผู้ดูแล ทีมพัฒนาโปรแกรมจึงไม่มองว่าเครื่องมือในส่วนนี้เป็นสาเหตุให้เกิดข้อบกพร่องของโปรแกรม เพราะในขณะที่โปรแกรมเกิดปัญหา ระบบส่วนกลางนี้ยังคงทำงานได้ตามปกติ สำหรับเครื่องมือที่ทางทีมพัฒนาโปรแกรมพิจารณาว่ามีส่วนทำให้เกิดข้อบกพร่องของโปรแกรม ได้แก่ (1) ในส่วนของ Data dictionary ของระบบ SAP ที่มีดรรชนีของฐานข้อมูลไม่เพียงพอ และเหมาะสมกับการนำมาใช้งาน (2) ขาดเครื่องมือในการควบคุมคุณภาพ คือ การที่ไม่มีข้อกำหนดหรือหลักการในการพัฒนาโปรแกรมให้ทำงานอย่างมีประสิทธิภาพ

ปัจจัยที่ 3 คือ กระบวนการ สำหรับสาเหตุที่น่าจะมาจากกระบวนการในการพัฒนาโปรแกรม ได้แก่ (1) การใช้ชุดคำสั่งที่ไม่ถูกต้องในการพัฒนาโปรแกรม (2) การเลือกใช้ชุดคำสั่งที่ไม่มีประสิทธิภาพในการพัฒนาโปรแกรม (3) การกำหนดเงื่อนไขที่ไม่เหมาะสมในการกรองข้อมูลที่จะเข้ามาประมวลผลในโปรแกรม (4) การกำหนดทีมงานที่ดูแลรับผิดชอบในบางขั้นตอนการทำงานไม่เหมาะสม (5) การที่ไม่มีขั้นตอนในการทดสอบประสิทธิภาพการทำงานของโปรแกรม

หลังจากที่รวบรวมสาเหตุที่มีผลทำให้โปรแกรมเกิดปัญหาประมวลผลไม่สำเร็จ หรือประมวลผลสำเร็จแต่ใช้เวลาการทำงานเกิน 1 วัน สามารถนำมาสร้างแผนผังสาเหตุและผลได้ดังภาพที่ 4.1

4.2.2 แนวทางการแก้ไข

ผู้ให้คำปรึกษาและทีมงานพัฒนาโปรแกรมร่วมกันเสนอแนวทางการแก้ไขปัญหาจากสาเหตุที่ได้ร่วมกันระบุไว้ดังนี้

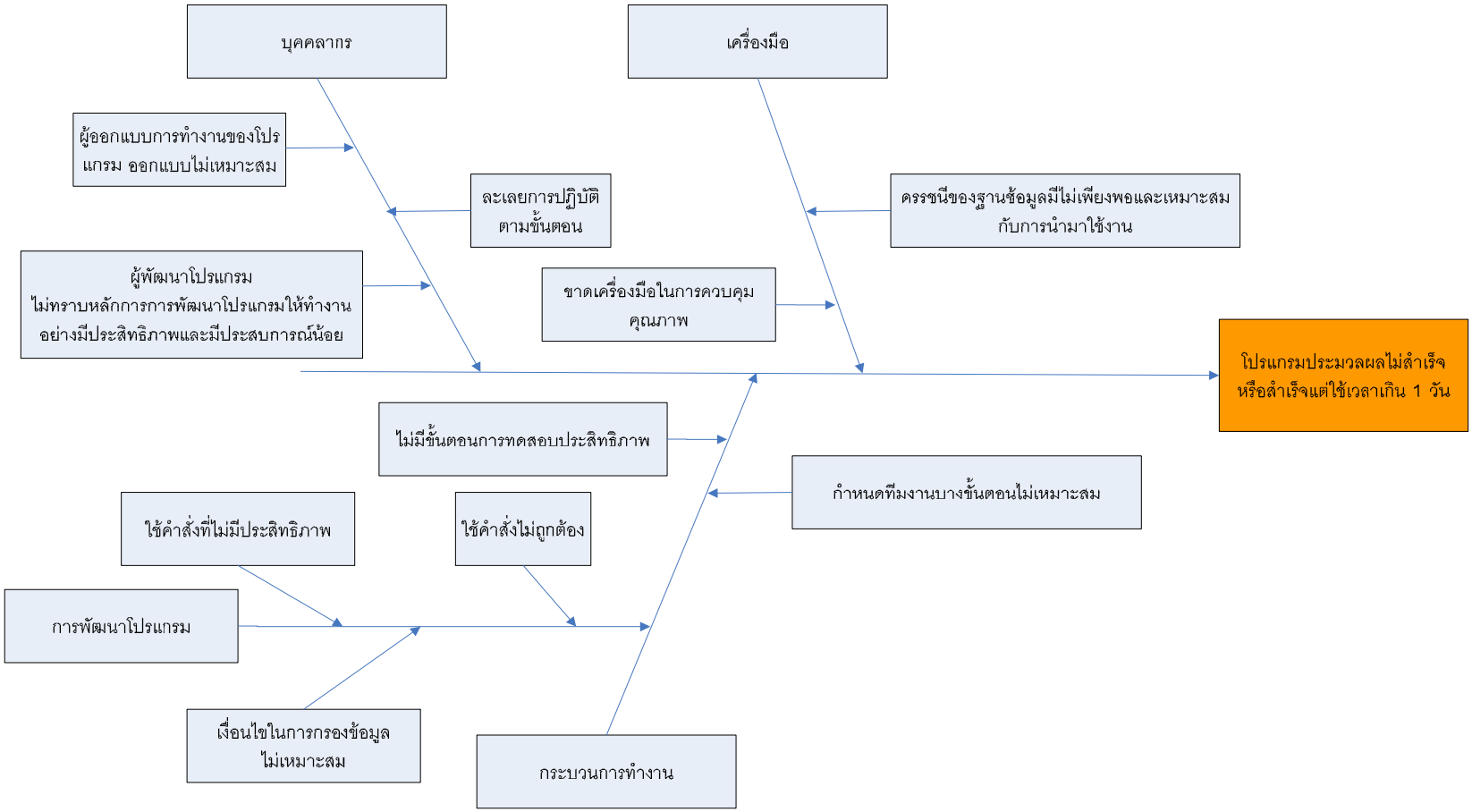
แนวทางการแก้ไขปัญหาจากสาเหตุที่น่าจะเกิดจากบุคคลากร ได้แก่ (1) การที่ผู้ออกแบบโปรแกรมเป็นผู้เก็บรวบรวมความต้องการของผู้ใช้แล้วนำมาจัดทำเอกสารรายละเอียดการทำงานของโปรแกรม โดยที่ไม่ทราบถึงความต้องการในด้านเทคนิคซึ่งมีผลกระทบต่อประสิทธิภาพการทำงานของโปรแกรม ในการออกแบบรายละเอียดการทำงานของโปรแกรมจึงควรขอความคิดเห็นจากผู้เชี่ยวชาญด้านเทคนิคเกี่ยวกับข้อจำกัดในเชิงเทคนิคหรือข้อควรระวังต่างๆ (2) สาเหตุที่เกิดจากตัวผู้พัฒนาโปรแกรมเองขาดประสบการณ์ในการทำงาน จึงควรที่จะปรึกษาผู้พัฒนาโปรแกรมที่มีประสบการณ์มากกว่า เพื่อเรียนรู้เทคนิคในการพัฒนาโปรแกรมให้ทำงานได้อย่างมีประสิทธิภาพ และควรทำการเรียนรู้เป็นทีมเพื่อจะได้พัฒนาโปรแกรมไปในแนวทางเดียวกัน (3) การที่บุคคลากรละเลยการปฏิบัติตามขั้นตอนการทำงาน ควรแก้ไขด้วยการจัดประชุมเพื่อชี้แจงว่าการทำงานในทุกขั้นตอนล้วนมีผลต่อผลิตภัณฑ์ที่ออกมา กล่าวคือ โปรแกรมที่พัฒนาเพิ่มและชี้ให้เห็นถึงความสำคัญของขั้นตอนการทำงานเหล่านั้น เพื่อจะได้ไม่ต้องมาตามแก้ไขในภายหลัง เช่น ขั้นตอนการจัดทำเอกสารรายละเอียดโปรแกรม ควรมีการทบทวนเอกสารเหล่านั้นว่าถูกต้องครบถ้วนตามความต้องการของผู้ใช้และไม่ติดขัดข้อจำกัดในเชิงเทคนิค เป็นต้น

แนวทางการแก้ไขปัญหาจากสาเหตุที่น่าจะเกิดจากเครื่องมือในการพัฒนาโปรแกรม ได้แก่ (1) ให้ทีมพัฒนาโปรแกรมรวบรวมความต้องการในการใช้งานดรรชนีฐานข้อมูลของแต่ละโปรแกรม เพราะในบางโปรแกรมอาจใช้ดรรชนีร่วมกันได้ และพิจารณาสร้างดรรชนีฐานข้อมูลเหล่านั้น สำหรับการสร้างดรรชนีฐานข้อมูลจะต้องทำการพิจารณาร่วมกับทีมดูแลระบบ SAP (Basis Administrator) เพื่อให้เกิดผลกระทบน้อยที่สุดในการบำรุงรักษาระบบ (2) ให้ทีมพัฒนาโปรแกรมทำการศึกษาและรวบรวมข้อมูลที่เป็นหลักการการพัฒนาโปรแกรมให้ทำงานอย่างมีประสิทธิภาพ และจัดทำเป็นเอกสารบัญชีสำหรับตรวจประสิทธิภาพการทำงานของโปรแกรม (Performance Checklist) เพื่อเป็นแนวทางในการพัฒนาที่เป็นมาตรฐานเดียวกัน

แนวทางการแก้ไขปัญหาจากสาเหตุที่น่าจะเกิดจากกระบวนการพัฒนาโปรแกรม ได้แก่ (1) ให้ผู้พัฒนาโปรแกรมอาวุโสเป็นผู้ตรวจทานการใช้ชุดคำสั่งในโปรแกรมทุกครั้งเพื่อเป็นการทบทวนก่อนที่จะนำส่งให้กับขั้นตอนการทดสอบ เพื่อแนะนำการแก้ไขสำหรับคำสั่งที่อาจส่งผลให้เกิดการทำงานที่ไม่รู้จบในโปรแกรม ซึ่งส่งผลให้โปรแกรมประมวลผลไม่สำเร็จ และใช้โปรแกรมวัดประสิทธิภาพการทำงานของโปรแกรม และปรับเปลี่ยนไปใช้คำสั่งที่มีประสิทธิภาพ ในอนาคตหากมีการสร้างเอกสารหลักการพัฒนาโปรแกรมและบัญชีสำหรับตรวจประสิทธิภาพการทำงานของโปรแกรม น่าจะช่วยแก้ไขสาเหตุนี้ให้ลดลงได้ (2) การกำหนดเงื่อนไขเพื่อกรองข้อมูลที่จะเข้ามาประมวลผลในโปรแกรม ควรแนะนำให้ผู้ใช้ระบุเงื่อนไขเพิ่มขึ้นเพื่อเป็นการลดปริมาณข้อมูลที่เข้ามาประมวลผลในโปรแกรม หรือกำหนดเงื่อนไขที่มีอยู่เดิมให้เป็นเงื่อนไขบังคับเพื่อเป็นการบังคับให้ผู้ใช้ไม่ลืมที่จะระบุขอบเขตของข้อมูล หรืออาจจะอาจจะเพิ่มจำนวนเงื่อนไข และปรับเปลี่ยนเงื่อนไขให้ตรงกับการจัดเก็บข้อมูลของฐานข้อมูล ซึ่งการเพิ่มและปรับเปลี่ยนจำนวนเงื่อนไขจะต้องทำการแก้ไขโปรแกรมซึ่งมีผลต่ออัลกอริธึมในโปรแกรม ทำให้ผู้ใช้ต้องทดสอบความถูกต้องของโปรแกรมซ้ำอีก (3) จากสาเหตุในเรื่องบุคคลากร กล่าวคือ ผู้ออกแบบการทำงานของโปรแกรมไม่ทราบถึงข้อจำกัดบางอย่างในด้านเทคนิค จึงขอเสนอแนะให้ในขั้นตอนการออกแบบโปรแกรมและหลังจากที่จัดทำเอกสารรายละเอียดโปรแกรมเรียร้อยแล้ว ควรให้ผู้พัฒนาโปรแกรมที่มีความเชี่ยวชาญช่วยทบทวนเอกสารเหล่านั้นและปรับแก้ตามความเหมาะสมก่อนที่จะส่งมอบให้ผู้พัฒนาโปรแกรมลงมือทำ (4) เสนอให้เพิ่มขั้นตอนการทดสอบประสิทธิภาพการทำงานของโปรแกรม และจัดทำเป็นเอกสารว่าโปรแกรมได้ผ่านการตรวจสอบแล้ว เพื่อช่วยลดการเกิดปัญหาในภายหลัง

ภาพที่ 4.1

แผนผังสาเหตุและผลของปัญหาเรื่องเวลาการทำงานของโปรแกรม



4.3 การแก้ไขปัญหา

การแก้ไขปัญหาในครั้งนี้เป็นการแก้ไขปัญหาในลักษณะที่เกิดข้อบกพร่องแล้วจึงแก้ไข ก่อนที่จะลงมือแก้ปัญหาจึงได้มีการประชุมและชี้แจงแนวทางการแก้ไขปัญหาและผลกระทบต่างๆที่จะเกิดขึ้นให้กับผู้ใช้ และทีมงานพัฒนาระบบทราบ เพื่อขอความร่วมมือจากทุกๆฝ่าย โดยวิธีการแก้ไขที่เลือกใช้จะต้องได้รับการอนุมัติจากผู้เกี่ยวข้องเสียก่อน สำหรับการแก้ไขอาจจะไม่ใช้เพียงวิธีใดวิธีหนึ่งเพียงอย่างเดียว อาจจะใช้ร่วมกันหลายๆ วิธี แต่วิธีการหลักที่ใช้ คือ การปรับแต่งคำสั่งตามบัญชีสำหรับตรวจสอบประสิทธิภาพ และอีกวิธีหนึ่งที่ใช้ร่วมกัน คือ ขอความร่มมือจากผู้ใช้ช่วยระบุเงื่อนไขในการกรองข้อมูลเพื่อช่วยเพิ่มประสิทธิภาพการทำงาน การแก้ไขปัญหามีดังนี้

1. วัดเวลาในการทำงานของโปรแกรม โดยใช้กรณีทดสอบที่มีปริมาณน้อยลง เพื่อพิจารณาการทำงานของโปรแกรมได้ทุกขั้นตอน เพื่อพิจารณาการทำงานที่ใช้เวลามาก ซึ่งจะเลือกปรับแต่งคำสั่งตามลำดับเวลาในการทำงานเรียงจากมากไปน้อย

2. ผู้พัฒนาโปรแกรมอาวุโสเป็นผู้ตรวจทานการใช้ชุดคำสั่งตามบัญชีสำหรับตรวจประสิทธิภาพ (Performance Checklist) ซึ่งผู้ให้คำปรึกษาและทีมพัฒนาโปรแกรมรวบรวมขึ้นมา และถ้ามีคำสั่งที่ต้องปรับแก้ให้จัดทำเอกสารส่งให้กับผู้พัฒนาโปรแกรมแก้ไขต่อไป

3. ขอความร่วมมือให้ผู้ใช้ระบุเงื่อนไขเพิ่มขึ้น หรือกำหนดเงื่อนไขที่มีอยู่เดิมให้เป็นเงื่อนไขบังคับ เพื่อเป็นการลดปริมาณข้อมูลที่เข้ามาประมวลผลในโปรแกรม เช่น รายงานที่แสดงสรุปยอดรวมจำนวนตามโรงงาน ซึ่งรายงานจะแสดงจำนวนวัตถุดิบทุกรายการออกมา และในการออกรายงานสามารถเรียกดูเป็นรายเลขที่วัตถุดิบ หรือจะเรียกเป็นช่วงของเลขที่วัตถุดิบได้ รายงานแบบนี้สามารถระบุเงื่อนไขเพิ่มเติมเป็นช่วงของเลขที่วัตถุดิบได้ หรือจะกำหนดให้เลขที่วัตถุดิบเป็นเงื่อนไขบังคับในการออกรายงาน เป็นต้น

4. เพิ่มจำนวนเงื่อนไข หรือปรับเปลี่ยนเงื่อนไขในการประมวลผลโปรแกรม เพื่อให้เงื่อนไขเป็นไปตามการจัดเก็บข้อมูลในฐานข้อมูล กล่าวคือเป็นไปตามคีย์หรือดรรชนีของตารางในฐานข้อมูล

5. ปรับแก้โปรแกรมตามบัญชีสำหรับตรวจประสิทธิภาพ ภาพที่ 4.2 แสดงขั้นตอนการปรับแก้โปรแกรม

6. ทำการวัดเวลาในการทำงานของโปรแกรมด้วยเงื่อนไขเดียวกันกับก่อนที่จะทำการแก้ไขโปรแกรม เพื่อตรวจดูว่ายังคงมีการทำงานในส่วนใดที่ยังใช้เวลามากอยู่ จนกระทั่งไม่สามารถปรับเปลี่ยนคำสั่งโปรแกรมได้อีก

7. ส่งมอบให้ผู้ใช้ทดสอบโปรแกรม เพื่อตรวจสอบความถูกต้องของโปรแกรม

4.4 **บัญชีสำหรับตรวจประสิทธิภาพ** (Performance Checklist)

การปรับปรุงประสิทธิภาพโปรแกรมประยุกต์ คือการปรับแต่งโค้ดโปรแกรม ABAP ด้วยการปรับเปลี่ยนการใช้คำสั่งเพื่อให้โปรแกรมมีประสิทธิภาพการทำงานที่ดีขึ้น การเลือกใช้ชุดคำสั่งจากการศึกษารวบรวมจากหนังสือและสิ่งตีพิมพ์ โดยขอสรุปตามข้อมูลของ SAP online help เป็นหลัก เนื่องจากเป็นข้อมูลและข้อแนะนำจากผู้พัฒนาระบบ SAP R/3 ซึ่งพัฒนาขึ้นด้วยภาษา ABAP/4 เช่นกัน ย่อมเสนอแนวทางการเขียนโปรแกรมอย่างมีประสิทธิภาพที่ควรใช้อย่างถูกต้อง ผู้ให้คำปรึกษาจึงจัดทำบัญชีสำหรับตรวจประสิทธิภาพการทำงานของโปรแกรม ซึ่งจะแบ่งเป็น 2 ส่วน คือ ส่วนที่เป็นคำสั่งที่ใช้กับฐานข้อมูล หรือที่เรียกว่า คำสั่ง SELECT และส่วนที่เป็นคำสั่งที่ใช้ในการประมวลผลหรือที่เรียกว่าคำสั่ง ABAP

การปรับเปลี่ยนคำสั่ง SELECT (SELECT statement) เพื่ออ่านข้อมูลจากฐานข้อมูลอย่างมีประสิทธิภาพมีดังนี้

1. ปรับการเขียนคำสั่ง SELECT * ให้มาใช้คำสั่งแบบระบุ Field เพราะจะช่วยลดการส่งผ่านข้อมูลจาก Database server มาที่ Application server

2. ปรับการระบุเงื่อนไขใน WHERE clause ของคำสั่ง SELECT ให้ใช้ตรงกับ Key หรือ Index ของตารางในฐานข้อมูล โดยให้ความสำคัญกับลำดับของ WHERE clause ตามลำดับของ Field ใน Key หรือ Index ด้วย

3. ปรับการระบุเงื่อนไขใน WHERE clause ของคำสั่ง SELECT ให้ใช้ AND และการเปรียบเทียบแบบเท่ากับ เพราะการเปรียบเทียบด้วย NOT, OR และ IN จะไม่เข้าเงื่อนไขการค้นหา

4. ปรับเปลี่ยนการประมวลผลที่ทำไปพร้อมกับคำสั่ง SELECT…ENDSELECT ให้มาใช้คำสั่ง SELECT…INTO TABLE เพื่ออ่านข้อมูลมาพักไว้ในตัวแปร Internal table แล้วค่อยประมวลผลข้อมูล

5. ปรับเปลี่ยนการใช้คำสั่ง SELECT ที่ซ้อน (Nested SELECT) หรือคำสั่ง SELECT ในคำสั่ง LOOP ด้วยการใช้คำสั่ง SELECT…FOR ALL ENTRIES และการที่จะใช้ประโยชน์ของคำสั่งนี้ได้อย่างเต็มประสิทธิภาพ ต้องตรวจสอบด้วยว่ามีข้อมูลแล้วในตัวแปร Internal table ต้นทาง และควรทำการเรียงข้อมูลในตัวแปร Internal table ต้นทางด้วย Column ที่จะนำมาใช้เป็น

Key และทำให้ Key นั้นไม่มีค่าซ้ำกันโดยใช้คำสั่ง DELETE ADJACENT DUPLICATES ก่อนทำการอ่านข้อมูลจากฐานข้อมูล

6. ปรับเปลี่ยนการใช้คำสั่ง SELECT ที่ซ้อน (Nested SELECT) ด้วยการใช้คำสั่ง JOIN โดยเงื่อนไขของการ JOIN ควรเป็นไปตาม Key ของตาราง หากไม่เป็นเช่นนั้นควรใช้คำสั่ง FOR ALL ENTRIES แทน แต่ไม่ควรใช้คำสั่ง JOIN และคำสั่ง FOR ALL ENTRIES ร่วมกัน รวมทั้งไม่ควรใช้คำสั่ง JOIN มากกว่า 2 ตาราง

7. ควรใช้คำสั่ง SELECT SINGLE สำหรับการอ่านข้อมูลจากฐานข้อมูลเพียงรายการเดียว แต่ต้องระบุเงื่อนไขให้ครบตาม Key ของตาราง หากทราบเงื่อนไขไม่ครบตาม Key ให้ใช้คำสั่ง SELECT…UP TO 1 ROWS

8. ควรหลีกเลี่ยงการใช้คำสั่ง SELECT กับ Aggregated function เนื่องจากคำสั่งเหล่านี้เป็นคำสั่งที่ไม่สามารถเก็บค่าพักไว้ใน Buffer ได้ จึงกลายเป็นการเพิ่มภาระงานให้กับการทำงานในฝั่งฐานข้อมูล เช่น SELECT…DISTINCT, SUM, MAX, GROUP BY, ORDER BY และ HAVING

9. ปรับเปลี่ยนการใช้ Database view และตารางในฐานข้อมูลอย่างเหมาะสม ในบางครั้งควรอ่านข้อมูลจาก Database view แทนการอ่านจากตาราง เพราะ Database view เกิดจากการ JOIN กันของ 2 ตารางขึ้นไปในฐานข้อมูล หรือในกรณีที่ข้อมูลที่ต้องการใช้ มีอัตราการเพิ่มของข้อมูลในปริมาณเท่าๆกันต่อหน่วยเวลา เช่น ในแต่ละเดือนจะมีปริมาณข้อมูลเพิ่มขึ้นเท่าๆกัน และข้อมูลที่ต้องการใช้เป็นข้อมูลตลอดปี ควรออกแบบการอ่านข้อมูลให้เป็นแบบรายเดือน เพื่อเป็นการลดปริมาณข้อมูลที่เข้ามาผ่านการทำงานในโปรแกรม หรืออ่านข้อมูลจากตารางที่เก็บค่าเป็นยอดรวมต่อเดือนซึ่งระบบ SAP มีไว้ให้อยู่แล้ว ก็จะช่วยลดเวลาในการทำงานของโปรแกรมลงได้

10. ปรับเปลี่ยนคำสั่ง INSERT, UPDATE และ DELETE กับตารางในฐานข้อมูล (Database table) ให้ใช้การทำงานแบบทีละชุดข้อมูล แทนที่จะทำงานทีละรายการ เพราะวิธีนี้ช่วยให้มั่นใจได้ว่า รายการที่เพิ่ม แก้ไข หรือลบ เป็นรายการที่ซ้ำกัน และยังช่วยลดภาระงานในฝั่งฐานข้อมูลด้วย

การปรับเปลี่ยนคำสั่ง ABAP ที่ใช้กับ Internal table เพื่อประมวลผลข้อมูล มีดังนี้

1. การกำหนดค่าเริ่มต้นให้กับตัวแปร Internal table ด้วยคำสั่ง OCCURS 0

2. ปรับเปลี่ยนวิธีการเพิ่มข้อมูลในตัวแปร Internal table จากการใช้คำสั่ง INSERT มาเป็นคำสั่ง APPEND

3. ปรับเปลี่ยนการแก้ไขข้อมูลในฐานข้อมูลจาก Internal table ถ้าหากต้องการแก้ไขเพียงบาง Column ควรใช้คำสั่ง MODIFY…TRANSPORTING f1…fn. และถ้าระบุเงื่อนไขด้วยจะทำให้เร็วขึ้น MODIFY…TRANSPORTING f1…fn WHERE…

4. ควรใช้คำสั่ง REFRESH และ FREE ทั้งก่อนและหลังการใช้งานตัวแปร

5. ปรับเปลี่ยนการอ่านข้อมูลจากตัวแรก Internal table มาใช้วิธีการอ่านแบบ READ TABLE…WITH KEY…BINARY SEARCH โดยก่อนอ่านให้ทำการเรียงลำดับข้อมูลในตัวแปร Internal table ด้วย Field ที่ใช้ระบุเงื่อนไขในการอ่าน

6. ปรับเปลี่ยนคำสั่ง LOOP ซ้อน LOOP มาใช้คำสั่ง LOOP แบบระบุ Index

7. ลดการใช้ตัวแปร Global แล้วปรับมาใช้ตัวแปร Local เพราะ ตัวแปร Local จะจอง Memory เมื่อประกาศใช้ใน Subroutine แต่เมื่อจบการทำงานใน Subroutine ตัวแปร Local จะคืน Memory โดยอัตโนมัติ และควรประกาศตัวแปรเท่าที่จำเป็นต้องใช้เพื่อเป็นการประหยัดการใช้งาน Memory

8. ในกรณีที่มีตัวแปร Internal table 2 ตัวแปร การ Copy ตัวแปร Internal table ควรทำครั้งเดียวทั้งตาราง โดย ITAB2[] = ITAB1[]

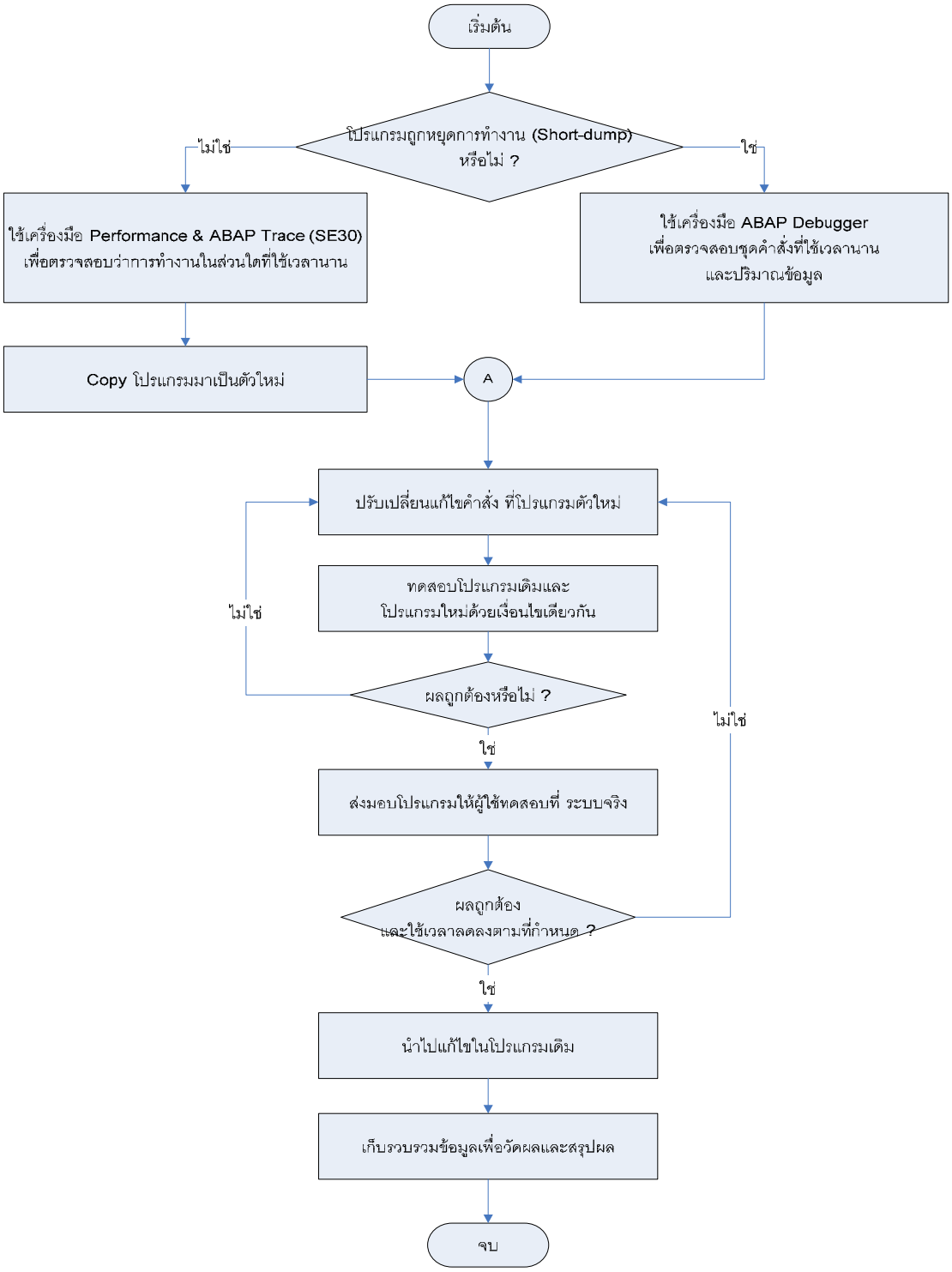
9. ปรับเปลี่ยนการใช้คำสั่ง DELETE กับตัวแปร Internal table จากการ LOOP …CHECK…DELETE…ENDLOOP มาเป็นการลบแบบระบุเงื่อนไขโดยใช้คำสั่ง DELETE ITAB…WHERE

10. ปรับเปลี่ยนการใช้ Header line หรือ Work area ของตัวแปร Internal table มาใช้ตัวแปร Field symbol เพราะเป็นการทำงานที่เนื้อข้อมูลโดยตรง โดยไม่มีการ Copy ข้อมูลจากเนื้อข้อมูลไปที่ตัวแปร Header line หรือ Work area

11. ปรับเปลี่ยนการใช้คำสั่ง SORT โดยต้องการเรียงลำดับข้อมูลแบบใดให้ระบุลงไปด้วยว่าเป็นแบบ ASCENDING หรือ DESCENDING และไม่ควรใช้คำสั่ง SORT ในคำสั่ง LOOP

12. สำหรับการเปรียบเทียบเงื่อนไขกับค่าที่เป็นไปได้มากกว่า 2 ค่า ให้ปรับเปลี่ยนการใช้คำสั่ง IF…ELSEIF…ELSEIF…ENDIF มาใช้คำสั่ง CASE…WHEN…WHEN…ENDCASE

ภาพที่ 4.2
ขั้นตอนการปรับแก้โปรแกรม



4.5 ผลการแก้ไขปัญหา

4.5.1 ผลด้านเทคนิคโปรแกรม

1. ผลเวลาการทำงานของโปรแกรมก่อนการแก้ไข ก่อนที่จะทำการแก้ไขได้เก็บผลเวลาการทำงานของโปรแกรมก่อนการแก้ไขพบว่า ทุกโปรแกรมมีอัตราส่วนเวลาในการทำงานในส่วนของการเข้าถึงฐานข้อมูลสูงกว่า 50% แสดงได้ดังตารางที่ 4.1 ซึ่งจะต้องพิจารณาถึงรายละเอียดของการอ่านข้อมูลเพื่อทำการปรับแต่งคำสั่งโปรแกรม

ตารางที่ 4.1

ข้อมูลอัตราส่วนเวลาในการทำงานของโปรแกรมก่อนการแก้ไข

โปรแกรม	อัตราส่วนเวลาในการทำงานของโปรแกรมก่อนการแก้ไข (%)		
	ABAP	Database	System
Program No. 1	28.4	63.8	7.8
Program No. 2	0	100	0
Program No. 3	27.2	69.1	3.7
Program No. 4	22.1	77.9	0
Program No. 5	25.9	74	0.1
Program No. 6	25.3	68.9	5.8
Program No. 7	0.1	99.9	0
Program No. 8	16.1	75	8.9
Program No. 9	32.4	67.3	0.3
Program No. 10	47.8	51.1	1.1
Program No. 11	17	82.9	0.1
Program No. 12	1.1	98.9	0

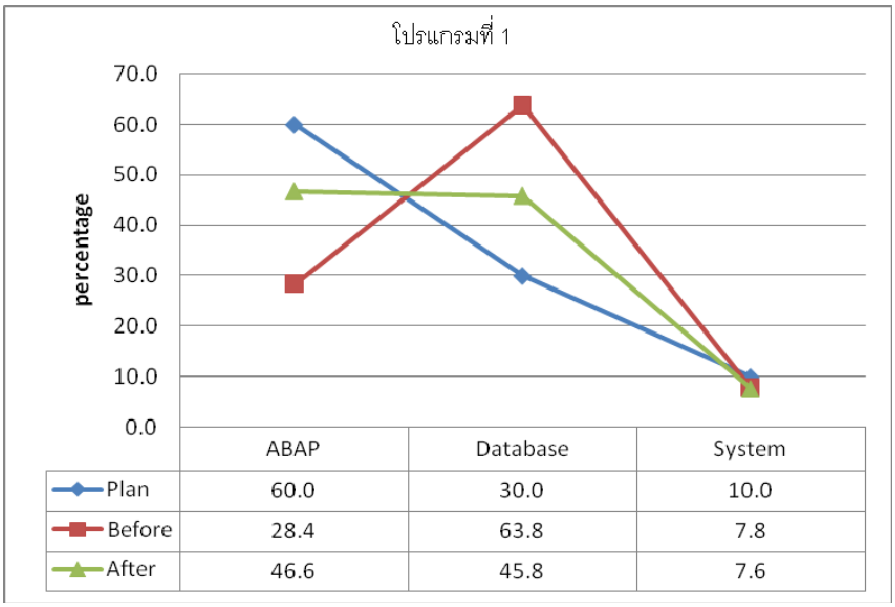
2. ผลเวลาการทำงานของโปรแกรมหลังการแก้ไข หลังจากการแก้ไขโปรแกรมตามบัญชีสำหรับตรวจประสิทธิภาพการทำงานของโปรแกรม จึงเก็บข้อมูลเวลาการทำงานของโปรแกรม 3 ครั้งนำมาหาค่าฉลี่ยและแสดงผลเวลาก่อนการแก้ไข หลังการแก้ไข และค่าจากสมมติฐาน เพื่อเปรียบเทียบกับสมมติฐาน โดยแสดงเป็นกราฟเส้นดังภาพที่ 4.3 ถึงภาพี่ 4.14 จากผลการแก้ไขพบว่ามีเพียง 3 โปรแกรมที่มีแนวโน้มของเวลาในการทำงานเป็นไปตามสมมติฐาน อัตราส่วนเวลาในการทำงานของโปรแกรม ได้แก่ โปรแกรมที่ 5, 6 และ10 คิดเป็น ร้อยละ 25 ของโปรแกรมที่แก้ไขทั้งหมด

อย่างไรก็ตามเมื่อนำผลหลังการแก้ไขมาเปรียบเทียบกับผลก่อนการแก้ไขในส่วนของเวลาที่ใช้ในการอ่านข้อมูลจากฐานข้อมูลดังตารางที่ 4.2 พบว่าใช้เวลาลดลง แบ่งเป็น 5 กลุ่ม ได้แก่ กลุ่มที่ใช้เวลาลดลง 1-19% มี 2 โปรแกรม คือ โปรแกรมที่ 3 และ 11 กลุ่มที่ใช้เวลาลดลง 20-29% มี 7 โปรแกรม คือ โปรแกรมที่ 1, 4, 7, 8, 9, 10 และ 12 กลุ่มที่ใช้เวลาลดลง 30-39% มี 1 โปรแกรม คือ โปรแกรมที่ 2 กลุ่มที่ใช้เวลาลดลง 40-49% มี 1 โปรแกรม คือ โปรแกรมที่ 5 และกลุ่มที่ใช้เวลาลดลงมากที่สุด 50% ขึ้นไป มี 1 โปรแกรม คือ โปรแกรมที่ 6

โปรแกรมที่ใช้เวลาลดลงในกลุ่มที่ลดลงสูงที่สุดสอดคล้องกับโปรแกรมที่มีแนวโน้มเวลาการทำงานเป็นไปตามสมมติฐาน ได้แก่ โปรแกรมที่ 5 และ 6 แสดงให้เห็นว่าโปรแกรมที่ยังคงต้องแก้ไขเพิ่มเติมอีก 9 โปรแกรม ควรมุ่งเน้นที่จะลดเวลาในการทำงานส่วนการอ่านข้อมูลจากฐานข้อมูลต่อไปอีก

ภาพที่ 4.3

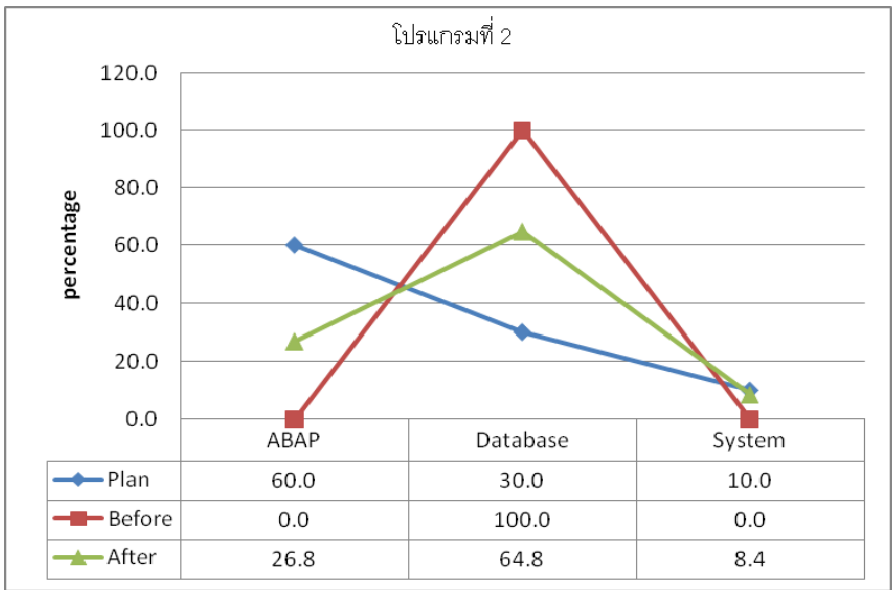
ผลเวลาการทำงานของโปรแกรมที่ 1



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	28.4	63.8	7.8
After	46.6	45.8	7.6

ภาพที่ 4.4

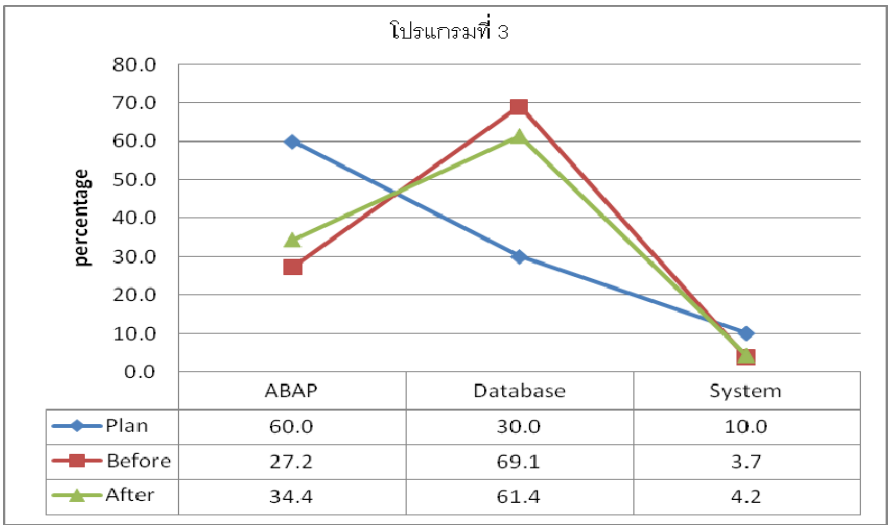
ผลเวลาการทำงานของโปรแกรมที่ 2



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	0.0	100.0	0.0
After	26.8	64.8	8.4

ภาพที่ 4.5

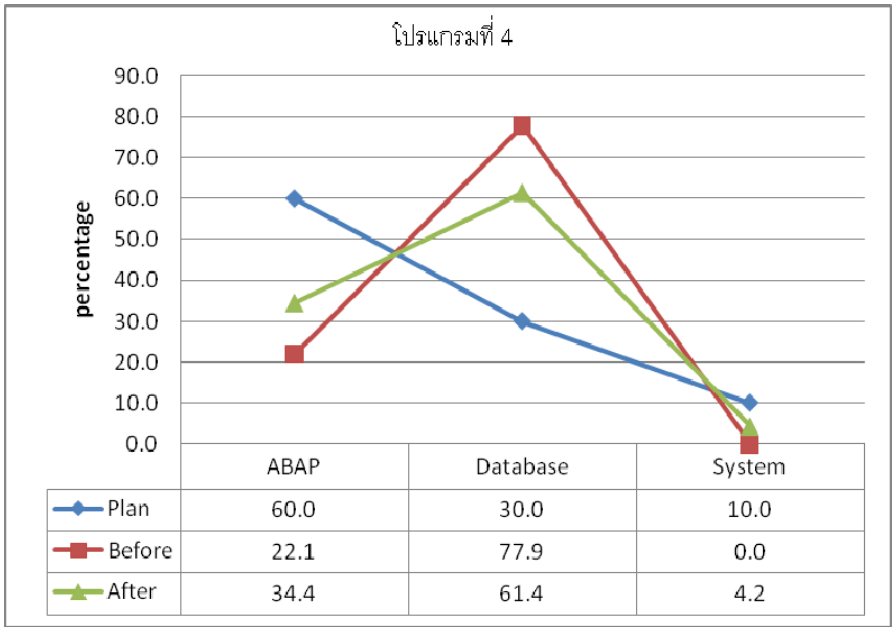
ผลเวลาการทำงานของโปรแกรมที่ 3



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	27.2	69.1	3.7
After	34.4	61.4	4.2

ภาพที่ 4.6

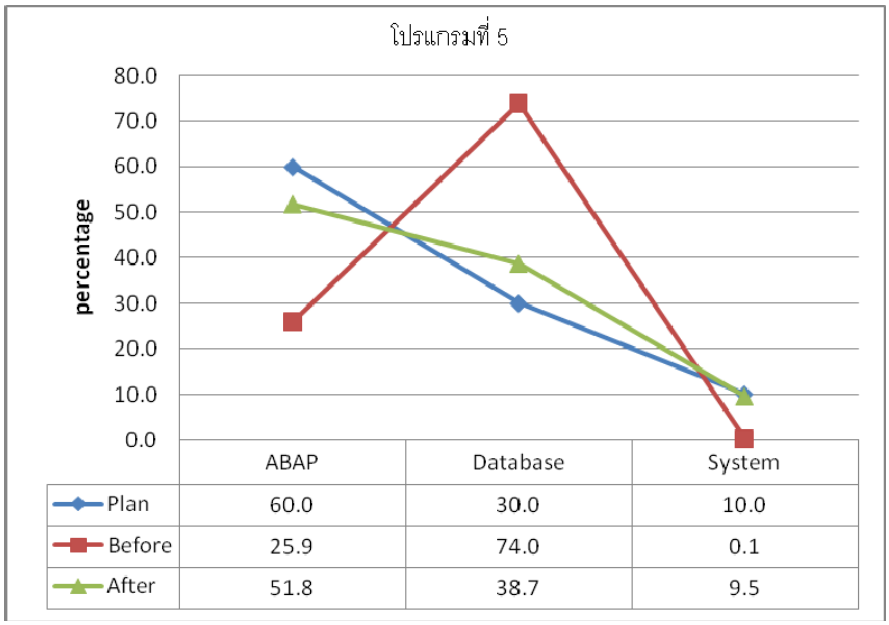
ผลเวลาการทำงานของโปรแกรมที่ 4



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	22.1	77.9	0.0
After	34.4	61.4	4.2

ภาพที่ 4.7

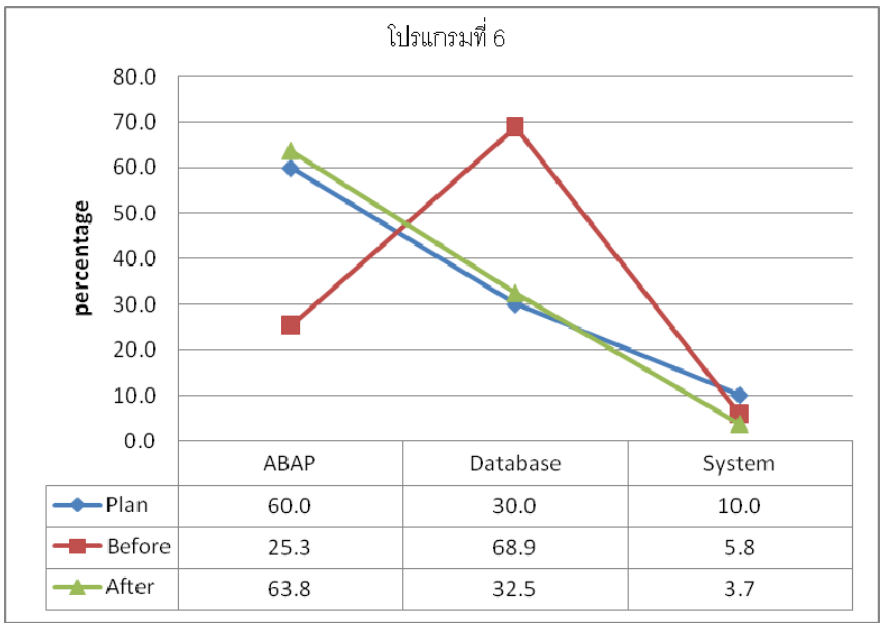
ผลเวลาการทำงานของโปรแกรมที่ 5



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	25.9	74.0	0.1
After	51.8	38.7	9.5

ภาพที่ 4.8

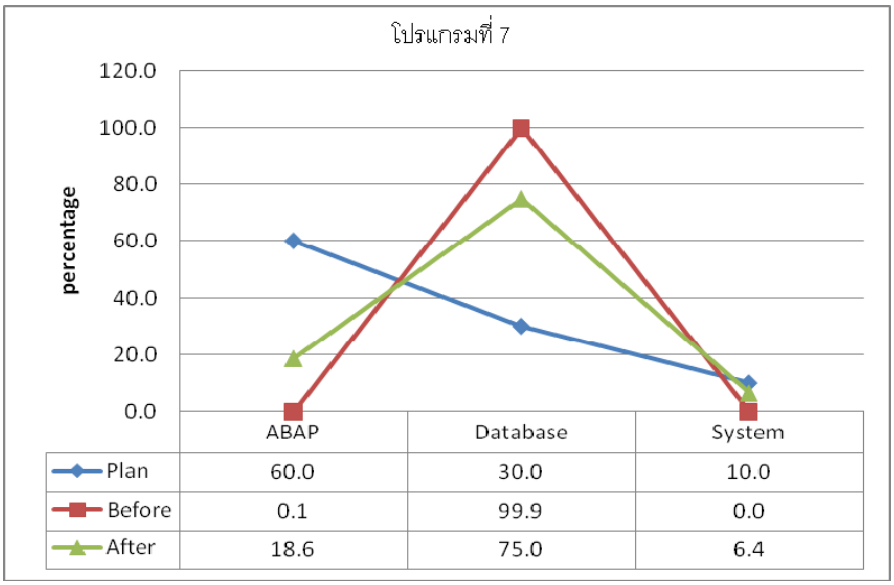
ผลเวลาการทำงานของโปรแกรมที่ 6



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	25.3	68.9	5.8
After	63.8	32.5	3.7

ภาพที่ 4.9

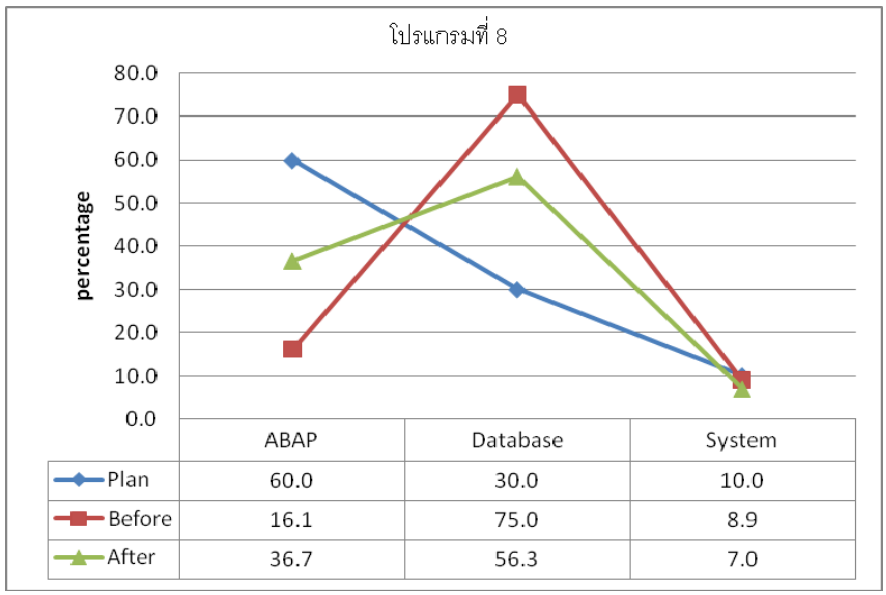
ผลเวลาการทำงานของโปรแกรมที่ 7



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	0.1	99.9	0.0
After	18.6	75.0	6.4

ภาพที่ 4.10

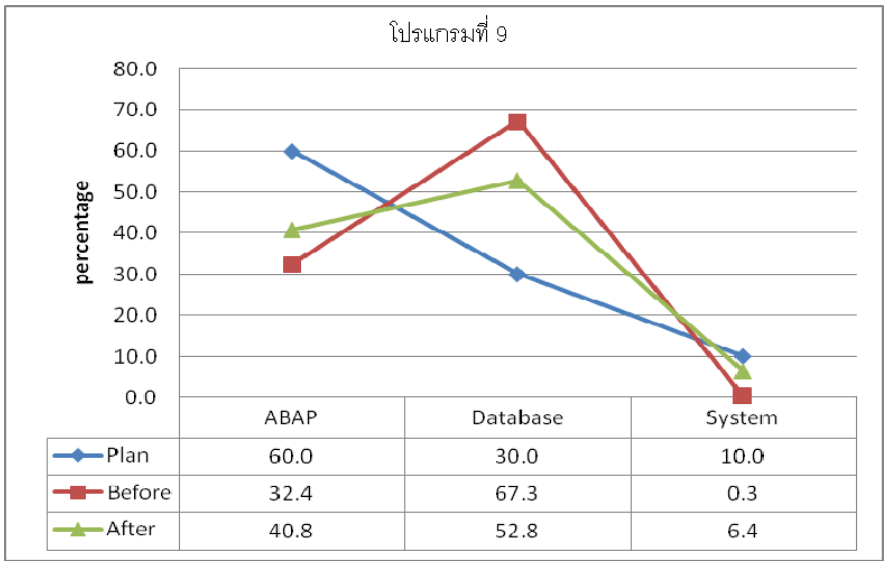
ผลเวลาการทำงานของโปรแกรมที่ 8



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	16.1	75.0	8.9
After	36.7	56.3	7.0

ภาพที่ 4.11

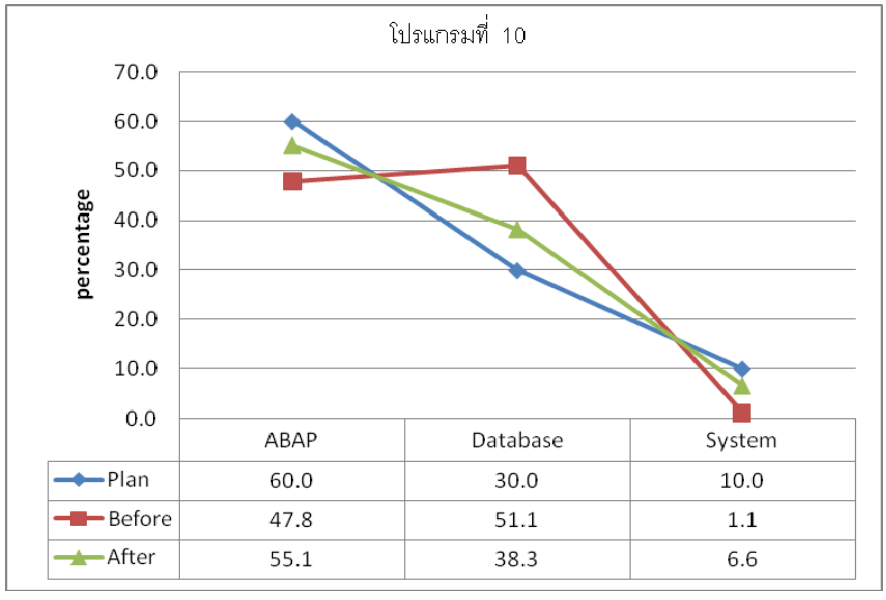
ผลเวลาการทำงานของโปรแกรมที่ 9



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	32.4	67.3	0.3
After	40.8	52.8	6.4

ภาพที่ 4.12

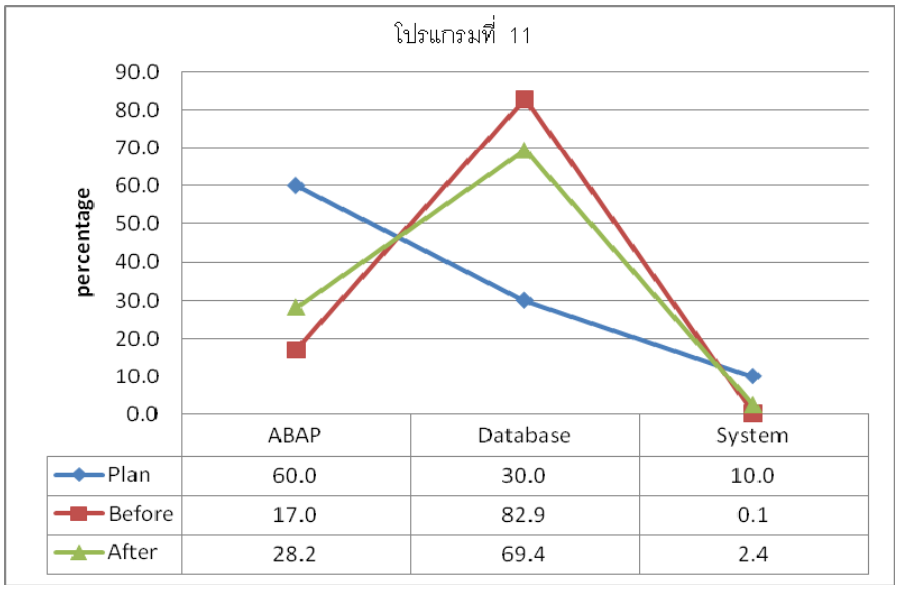
ผลเวลาการทำงานของโปรแกรมที่ 10



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	47.8	51.1	1.1
After	55.1	38.3	6.6

ภาพที่ 4.13

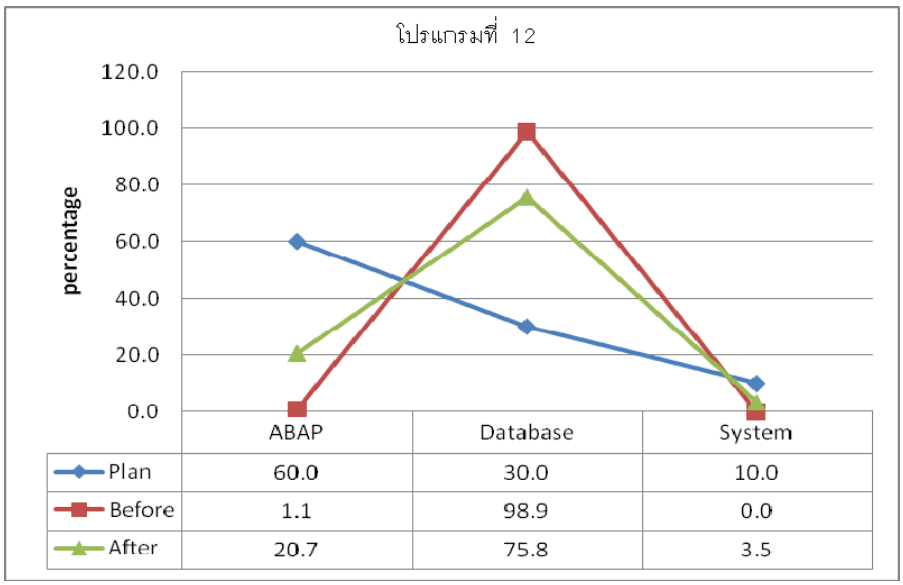
ผลเวลาการทำงานของโปรแกรมที่ 11



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	17.0	82.9	0.1
After	28.2	69.4	2.4

ภาพที่ 4.14

ผลเวลาการทำงานของโปรแกรมที่ 12



	ABAP	Database	System
Plan	60.0	30.0	10.0
Before	1.1	98.9	0.0
After	20.7	75.8	3.5

ตารางที่ 4.2

ข้อมูลเวลาในการทำงานส่วนฐานข้อมูลก่อนและหลังการแก้ไขและเวลาที่ลดลง

โปรแกรม	ก่อนการแก้ไข	หลังการแก้ไข	ใช้เวลาลดลงคิดเป็น %
	Database	Database	
Program No. 1	63.8	45.8	28.27
Program No. 2	100	64.8	35.20
Program No. 3	69.1	61.4	18.09
Program No. 4	77.9	60.0	22.94
Program No. 5	74	38.7	47.70
Program No. 6	68.9	32.5	52.83
Program No. 7	99.9	75.0	24.92
Program No. 8	75	56.3	24.93
Program No. 9	67.3	52.8	21.55
Program No. 10	51.1	38.3	25.05
Program No. 11	82.9	69.4	16.32
Program No. 12	98.9	75.8	23.32

4.5.2 **ผลด้านความพึงพอใจ**

1. ผลด้านความพึงพอใจต่อโปรแกรมหลังการแก้ไข

ผู้ใช้งานโปรแกรมที่มีปัญหาทั้งหมด 12 ท่านให้ข้อมูลว่า โปรแกรมสามารถทำงานได้ตามปกติและใช้เวลาในการประมวลผลไม่เกิน 1 วันตามที่ต้องการ รวมทั้งได้ตรวจสอบความถูกต้องของโปรแกรมด้วยการทดสอบตามเงื่อนไขต่างๆแล้ว ได้ผลว่าเรียบร้อยดี และเมื่อสอบถามถึงความพึงพอใจต่อโปรแกรมหลังการแก้ไข ทุกท่านให้ข้อมูลว่าพึงพอใจต่อการทำงานของโปรแกรม กล่าวคือ ผู้ใช้งานมีความพึงพอใจต่อผลการแก้ไขคิดเป็นร้อยละ 100 นอกจากนี้ผู้ใช้งานได้ให้ข้อเสนอแนะเพิ่มเติมเกี่ยวกับขั้นตอนในการแก้ไขปัญหาและปัญหาที่พบระหว่างการแก้ไข

สรุปเป็นประเด็นสำคัญได้ดังนี้ (1) การแก้ไขโปรแกรมทำให้ภาระงานของผู้ใช้เพิ่มขึ้นเพราะต้องตรวจสอบความถูกต้องของโปรแกรมใหม่ และมีผลกระทบค่อนข้างมาก ถ้าเป็นไปได้จึงไม่อยากให้มีการปรับแก้โปรแกรม (2) ในระหว่างการแก้ไขไม่อยากให้มีการทำงานซ้ำหลายรอบถ้าเป็นไปได้ต้องการให้ทางทีมออกแบบโปรแกรมและทีมพัฒนาโปรแกรมตรวจงานให้ถี่ถ้วนกว่านี้ก่อนที่จะส่งมาให้ผู้ใช้ทดสอบ (3) การปรับเปลี่ยนเกี่ยวกับเงื่อนไขในการประมวลผลโปรแกรมที่จะช่วยกรองข้อมูลให้ลดลง เงื่อนไขบางอย่างไม่สามารถระบุเพิ่มเติมได้ เนื่องจากในการทำงานจริง ทางผู้ใช้ไม่สามารถทราบเงื่อนไขเหล่านั้น

2. ผลด้านความคิดเห็นต่อแนวทางการแก้ปัญหา และการลงมือแก้ไขปัญหาจริง

ผลการสัมภาษณ์ตัวแทนของแผนกสารสนเทศทั้ง 3 ท่าน ถึงความคิดเห็นที่มีต่อแนวทางการแก้ไขปัญหาที่ทางผู้ให้คำปรึกษาและทีมพัฒนาโปรแกรมเสนอไว้ สามารถสรุปได้ 2 เรื่อง คือ เรื่องที่เห็นด้วยและไม่เห็นด้วย เรื่องที่เห็นด้วยสำหรับแนวทางการแก้ไข ได้แก่ (1) การชี้ให้เห็นถึงความสำคัญของทุกขั้นตอนในการพัฒนาโปรแกรม เพราะสามารถส่งผลต่อโปรแกรม และจากปัญหาที่เกิดขึ้นนี้ทำให้ทุกคนยิ่งต้องตระหนักถึงความรับชอบในงานส่วนของตนเองก่อนที่จะส่งมอบต่อไปให้กับขั้นตอนต่อไป (2) เห็นด้วยกับการจัดทำเอกสารบัญชีสำหรับตรวจประสิทธิภาพการทำงานของโปรแกรมเพราะเป็นการสร้างมาตรฐานในการพัฒนาโปรแกรม (3) เห็นด้วยกับการให้ผู้เชี่ยวชาญด้านเทคนิคเข้าไปมีส่วนร่วมในการออกแบบการทำงานของโปรแกรม ถึงแม้ว่าไม่สามารถนำแนวทางการแก้ปัญหาในข้อนี้มาใช้ได้กับสถานการณ์ปัจจุบันก็ตาม (4) เห็นด้วยกับการเพิ่มขั้นตอนการทดสอบประสิทธิภาพการทำงานของโปรแกรม เพราะช่วยในการกรองปัญหาระดับหนึ่งก่อนส่งถึงมือผู้ใช้ (5) สำหรับเรื่องที่ไม่เห็นด้วย คือ เรื่องของการสร้างดรรชนีฐานข้อมูลเพราะกลัวผลกระทบที่อาจจะเกิดขึ้นต่อการบำรุงรักษาระบบฐานข้อมูลในอนาคต

ผลการสัมภาษณ์ในส่วนของความคิดเห็นต่อการลงมือแก้ไขปัญหาจริง ทุกท่านเห็นด้วยกับขั้นตอนการแก้ปัญหา และให้ข้อเสนอแนะเพิ่มเติมว่าให้ระมัดระวังในเรื่องเวอร์ชันของโปรแกรมให้มาก และให้ย้อนกลับไปแก้ไขเอกสารที่เกี่ยวข้องกับโปรแกรมให้เป็นไปตามการทำงานปัจจุบันของโปรแกรม

4.6 วิเคราะห์ผลการแก้ไขปัญหา

จากผลการแก้ไขโปรแกรม มีโปรแกรมเพียง 3 โปรแกรมจาก 12 โปรแกรม หรือคิดเป็นร้อยละ 25 ที่ใช้เวลาในการทำงานเป็นไปตามสมมติฐานที่ตั้งไว้ ผู้ให้คำปรึกษาและทีมพัฒนาโปรแกรมจึงร่วมกันวิเคราะห์ปัจจัยที่ทำให้เวลาในการทำงานของโปรแกรมไม่เป็นไปตามสมติฐาน ซึ่งถือเป็นโมเดลที่ดี ได้แก่ (1) ในการทำงานของโปรแกรมจำเป็นต้องอ่านข้อมูลจากหลายตาราง เพื่อนำมาประมวลผล และไม่สามารถปรับเงื่อนไขให้ตรงกับคีย์หรือดรรชนีของตารางได้ และ (2) วิธีการแก้ไขปัญหาที่ไม่ได้นำมาใช้ คือ การสร้างดรรชนีเพิ่มเติมเพื่อให้รองรับกับการใช้งาน ซึ่งจะช่วยลดเวลาในการอ่านข้อมูลจากฐานข้อมูลให้น้อยลงได้ แต่เนื่องจากทางแผนกสารสนเทศของบริษัท ไม่เห็นด้วยในการใช้วิธีนี้ จึงทำให้ผู้ให้คำปรึกษาและทีมพัฒนาโปรแกรมต้องทำการศึกษาข้อดีและข้อเสียในการสร้างดรรชนี ซึ่งการศึกษาและทดลองจะต้องทำงานร่วมกับทีมดูแลระบบฐานข้อมูล เพื่อเสนอแนวทางการแก้ไขด้วยวิธีนี้ต่อไป

อย่างไรก็ตาม โปรแกรมที่ได้รับการแก้ไขสามารถทำงานได้ตามเวลาที่ผู้ใช้กำหนด และยังคงความถูกต้องของผลลัพธ์โปรแกรม ซึ่งเป็นผลให้ผู้ใช้ให้ความพึงพอใจกับผลการแก้ไขคิดเป็นร้อยละ 100 และสำหรับข้อเสนอแนะของผู้ใช้ และตัวแทนของแผนกสารสนเทศ ทางทีมพัฒนาโปรแกรมจะนำไปใช้ในการปรับปรุงกระบวนการทำงาน ซึ่งจะนำไปสู่การป้องกันการเกิดปัญหา

4.7 ปัญหาที่พบระหว่างการแก้ไขปัญหา

ปัญหาที่พบในระหว่างการแก้ไขปัญหา ได้แก่ ผู้พัฒนาโปรแกรมไม่สามารถทดสอบในเรื่องเวลาการทำงานของโปรแกรมได้ที่เครื่องสำหรับการพัฒนาโปรแกรม (Development Server) เนื่องจากปริมาณข้อมูลสำหรับทดสอบมีไม่มากพอ ต้องส่งโปรแกรมขึ้นไปทดสอบที่เครื่องรับประกันคุณภาพ (Quality Assurance Server) หรือเครื่องที่ผู้ใช้ทำงานอยู่จริง (Production Server) ทำให้มีจำเป็นต้องมีการทำงานซ้ำหลายรอบ ถ้าหากโปรแกรมยังคงใช้เวลาในการทำงานเกินที่กำหนดไว้ และในส่วนของการวัดเวลาในการทำงานด้วยเครื่องมือ ABAP Trace เป็นการวัดแบบออนไลน์ (Online process) ถ้าในขณะทำการวัดมีกระบวนการออนไลน์อื่นๆ ทำงานอยู่ด้วยอาจส่งผลต่อการวัดของเรา จึงแก้ปัญหาด้วยการวัด 3 ครั้งแล้วนำมาหาค่าเฉลี่ย นอกจากนี้ยังมีปัญหาในเรื่องการติดต่อประสานงานกับผู้ที่เกี่ยวข้องทำให้การวัดผลเวลาการทำงานของโปรแกรมที่ได้ทำการแก้ไขล่าช้ากว่าที่กำหนด