การออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์ห่อหุ้มโปรแกรมเลียนแบบเครื่องปลายทาง

นายถาวร ลิ้มวัฒนาชัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2553

DESIGN AND DEVELOPMENT OF TERMINAL EMULATER WRAPPER APPLICATION PROGRAMMING INTERFACE

Mr. Thaworn Limwattanachai

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Software Engineering
Department of Computer Engineering
Chulalongkorn University
Academic Year 2010

หัวข้อวิทยานิพนธ์	การออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์ห่อหุ้มโปรแกรมเลียนแบบเครื่องปลายทาง
โดย	นาย ถาวร ลิ้มวัฒนาชัย
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร.อติวงศ์ สุชาโต

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร. บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร. โปรดปราน บุณยพุกกณะ)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(ผู้ช่วยศาสตราจารย์ ดร.อติวงศ์ สุชาโต)

..... กรรมการ

(รองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี)

..... กรรมการภายนอกมหาวิทยาลัย

(ดร.ประกาศิต กายะสิทธิ์)

ถาวร ลิ้มวัฒนาชัย : การออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์ห่อหุ้มโปรแกรมเลียนแบบเครื่องปลายทาง. (DESIGN AND DEVELOPMENT OF TERMINAL EMULATER WRAPPER APPLICATION PROGRAMMING INTERFACE) อ. ที่ปรึกษาวิทยานิพนธ์หลัก: ผศ. ดร. อติวงศ์ สุชาโต, 94 หน้า.

วิทยานิพนธ์ฉบับนี้ได้นำเสนอเอพีไอ (API) ที่ช่วยในการปรับปรุงส่วนต่อประสานผู้ใช้งานของโปรแกรมเลียนแบบเครื่องปลายทาง (Terminal Emulator) ด้วยกลวิธีสกรีนสเครปปิง (Screen Scraping) เพื่อให้ผู้พิการทางการเห็นสามารถทำงานร่วมกับระบบเก่า (Legacy System) ด้วยโปรแกรมอ่านหน้าจอได้ โดยเอพีไอนี้ ได้นำแบบจำลองวัตถุเชิงเอกสาร (Document Object Modeling) มาประยุกต์ใช้ในการกำหนดข้อมูลหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางให้อยู่ในรูปแบบของวัตถุ พร้อมทั้งพัฒนาเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนเครื่องปลายที่ช่วยในการเก็บข้อมูลบนหน้าจอของโปรแกรมเลียนแบบเครื่องปลายทางให้อยู่ในรูปแบบไฟล์เอ็กซ์เอ็มแอล (XML) ด้วยเอพีไอและเครื่องมือนี้จะช่วยให้ผู้พัฒนาโปรแกรมไม่จำเป็นต้องทำความเข้าใจการทำงานของระบบเก่าทั้งหมด ผู้เชี่ยวชาญระบบเก่าจะใช้เครื่องมือในการเก็บข้อมูลและสร้างเอกสารอ้างอิงเพื่อให้ผู้พัฒนาสามารถเรียกดูข้อมูลได้ผ่านส่วนต่อประสานโปรแกรมประยุกต์

ภายในวิทยานิพนธ์ได้ทดสอบเอพีไอที่ถูกสร้างขึ้นโดยการเปรียบเทียบกับเอพีไอดั้งเดิมชื่อ EHLLAPI (Extended High Level Language Application Programming Interface) ได้แบ่งการทดสอบออกเป็น 2 ส่วน คือ การเปรียบเทียบกระบวนการในการเขียนโปรแกรมและการเปรียบเทียบจำนวนบรรทัดของโค้ด โดยกำหนดขอบเขตในการปรับปรุงระบบเก่า ผลการทดสอบพบว่าเอพีไอที่ออกแบบสามารถช่วยลดขั้นตอนในการพัฒนาและจำนวนโค้ดในการเขียนโปรแกรมได้ และสามารถใช้งานได้สะดวกมากขึ้นกว่าเอพีไอประยุกต์ดั้งเดิมอีกด้วย

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต ถาวร ลิ้มวัฒนาชัย

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก [illegible]

ปีการศึกษา 2553

5070284821 : MAJOR SOFTWARE ENGINEERING

KEYWORDS: API / Legacy / Screen Reader / Blind / Wrapper/Assistive Technology

THAWORN LIMWATTANACHAI :DESIGN AND DEVELOPMENT OF TERMINAL EMULATER WRAPPER APPLICATION PROGRAMMING INTERFACE. ADVISOR :ASST. PROF. ATIWONG SUCHATO, Ph.D., 94 pp

This thesis proposes an application programming interface (API) that facilitates programmers to develop any application to connect with terminal emulator interface that will be readable by screen reader program. Using screen scraping technique, the document object modeling method (DOM) is utilized in the API development to categorize on-screen data in terminal emulator into object forms. The thesis also provides the tool to generate terminal emulator screen interaction model to store data in xml file type. By using this API and the tool, application development can be completed without any expertise in the legacy system because the data will be collected by skilled users and the rest will be automatically handled by the tool.

This thesis compares the proposed API with the EHLLAPI (Extended High Level Language Application Programming Interface). The experiments are divided into 2 parts. The first part is to compare the process of coding and the second part is to compare the number of lines of code. The results show that the proposed API is very effective at minimizing the steps of programming deployment while reducing coding tasks.

Department : Computer Engineering
Field of Study : Software Engineering
Academic Year : 2010

Student's Signature ธวร ลิ้มวัฒนชัย
Advisor's Signature อ.ดร.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความเมตตาและความช่วยเหลืออย่างยิ่งจากผู้ช่วยศาสตราจารย์อติวงศ์ สุชาโต อาจารย์ที่ปรึกษา ที่ค่อยสละเวลาให้คำปรึกษา คำแนะนำที่ดี รวมทั้งยังคอยเป็นกำลังใจและผลักดันให้งานวิจัยเดินหน้าไปได้ด้วยดี

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. โปรดปราน บุญญพุกกณะ ประธานกรรมการสอบวิทยานิพนธ์ รองศาสตราจารย์ ดร.พรศิริ หมื่นไชยศรี และ ดร.ประกาศิต กายะสิทธิ์ ที่เมตตาและสละเวลาเป็นคณะกรรมการสอบ และให้คำแนะนำสำหรับโครงร่างวิทยานิพนธ์ให้มีคุณภาพมากยิ่งขึ้น

ขอขอบพระคุณคณาจารย์ทุกท่านในภาควิชาคอมพิวเตอร์ที่ให้ความรู้แก่ข้าพเจ้าทั้งในเวลาและนอกเวลาเรียน และคอยให้ความช่วยเหลือในทุกเรื่องตลอดการเรียนที่ผ่านมา

ขอขอบคุณเพื่อน ๆ ระดับปริญญาโทที่คอยช่วยเหลืองานในด้านต่างๆ จนทุกอย่างสำเร็จลุล่วงได้ด้วยดี

ขอขอบคุณพี่ ๆ น้อง ๆ ที่ห้องแลบปฏิบัติการเทคโนโลยีช่วยเหลือผู้พิการที่ให้ความเมตตาข้าพเจ้า ช่วยดูแลให้ความช่วยเหลือด้านวิทยานิพนธ์และให้ความเป็นกันเองอย่างที่สุด ขอขอบคุณพี่ ๆ เจ้าหน้าที่ภาควิชาวิศวกรรมคอมพิวเตอร์ที่คอยให้ความช่วยเหลือในด้านการจัดการเอกสารต่าง ๆ

สุดท้ายนี้ขอขอบคุณญาติ และครอบครัว โดยเฉพาะ คุณพ่อ คุณแม่ที่ให้กำลังใจและสนับสนุนในทุกด้าน ช่วยให้ผมมีกำลังที่คิดจะสู้ต่อไปจนถึงที่สุดจนผ่านพ้นช่วงเวลายากลำบากไปจนได้ครับ

สารบัญ

หน้า

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

หน้า

หน้า

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของงานวิจัย

ความเข้าถึงได้เริ่มเข้ามามีบทบาทสำคัญกับซอฟต์แวร์คอมพิวเตอร์ในปัจจุบัน โปรแกรมอ่านหน้าจอ (Screen Reader) เป็นซอฟต์แวร์ประเภทหนึ่งที่ช่วยให้ผู้พิการทางการเห็นสามารถเข้าถึงข้อมูลต่าง ๆ และทำงานได้ หลายองค์กรทั้งภาครัฐและเอกชนเริ่มเล็งเห็นถึงศักยภาพและสนใจเปิดรับบุคลากรเข้าทำงานด้านดังกล่าว แต่ปัญหาสำคัญอย่างหนึ่งที่พบคือโปรแกรมประยุกต์ที่ใช้ตามบริษัทหรือองค์กรส่วนใหญ่ในปัจจุบันไม่ได้ถูกออกแบบมาให้ผู้พิการสามารถเข้าถึงได้ โดยเฉพาะอย่างยิ่งบริษัทประกัน กระทรวงต่าง ๆ และธนาคาร ยังคงใช้ระบบเมนเฟรมซึ่งเป็นระบบเก่า (Legacy System) [1] กันอย่างแพร่หลาย ระบบเหล่านี้จะทำการเชื่อมต่อกับโปรแกรมเลียนแบบเครื่องปลายทาง (Terminal Emulator) [2] เพื่อใช้รับคำสั่งและแสดงผลข้อมูลบนเครื่องคอมพิวเตอร์ลูกข่าย ด้วยหน้าจอที่มีลักษณะคล้ายระบบดอส ข้อมูลต่าง ๆ ที่ถูกแสดงผลเป็นเพียงตัวอักษร การสืบค้นผลลัพธ์จำเป็นต้องให้โปรแกรมอ่านหน้าจออ่านไปจนถึงตำแหน่งที่แสดงผล ทำให้ผู้พิการทางการเห็นไม่สามารถทำงานที่เกี่ยวข้องกับระบบเก่าได้อย่างมีประสิทธิภาพ

การแก้ปัญหาในลักษณะนี้ไม่ควรต้องแทนที่ด้วยระบบใหม่หรือแก้ไขโค้ดเดิมที่มีอยู่ อันเป็นเหตุนำมาซึ่งความเสี่ยงต่อการสูญหายของกระบวนการทางธุรกิจของบริษัทที่มีมาตลอดระยะเวลาหลายสิบปี [3] อีกทั้งต้องใช้ต้นทุนสูงและเวลาจำนวนมาก กลวิธีสกรีนสเครปปิง (Screen Scraping Technique) [4] จึงเป็นวิธีหนึ่งที่ช่วยดึงข้อมูลจากหน้าจอของระบบเก่ามาจัดรูปและแสดงผลบนส่วนต่อประสานผู้ใช้งานสมัยใหม่ (Modern User Interface) กลวิธีดังกล่าวเป็นวิธีที่ใช้ปรับปรุงระบบเก่าในลักษณะกล่องดำ (Black Box Modernization) [3] โดยการห่อหุ้มระบบเดิมที่มีอยู่ด้วยส่วนต่อประสานสมัยใหม่ ช่วยให้ผู้พัฒนาสามารถออกแบบส่วนต่อประสานได้ในระยะเวลาอันรวดเร็วและเป็นไปตามแนวทางการพัฒนาซอฟต์แวร์ที่ทุกคนเข้าถึงได้ โดยไม่จำเป็นต้องแก้ไขโค้ดระบบเก่า

ถึงแม้ว่าการสร้างโปรแกรมประยุกต์ด้วยกลวิธีสกรีนสเครปปิงผ่านส่วนต่อประสานโปรแกรมประยุกต์ที่มีอยู่จะแก้ไขปัญหาที่เกิดขึ้นข้างต้นได้ แต่ผู้พัฒนาโปรแกรมประยุกต์ก็จำเป็นต้องทำความเข้าใจขั้นตอนการใช้งานของระบบเก่าเพื่อที่จะสามารถพัฒนาโปรแกรมประยุกต์

ที่นำทางไปยังแต่ละหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางและส่งหรือรับข้อมูลกลับมา ผู้พัฒนายังต้องเข้าใจเงื่อนไขข้อผิดพลาด (Error) ต่างๆ ที่อาจเกิดขึ้นจากตัวโปรแกรมด้วย โดยข้อผิดพลาดเป็นได้ทั้งข้อผิดพลาดจากตัวโปรแกรมประยุกต์เองและข้อผิดพลาดจากการซิงโครไนท์หน้าจอ ปัญหาที่สำคัญอีกอย่างหนึ่งของการพัฒนาด้วยกลวิธีสกรีนสเครปปิงก็คือ หากมีการปรับเปลี่ยนหน้าจอของระบบเก่าขึ้นใหม่ ผู้พัฒนาก็จำเป็นต้องแก้ไขโปรแกรมประยุกต์และติดตั้งที่เครื่องลูกข่ายทั้งหมดให้อีกครั้งหนึ่งซึ่งอาจเป็นปัญหาด้านการบำรุงรักษาต่อไปในอนาคต

วิทยานิพนธ์นี้จึงได้มุ่งเน้นในการออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์ใหม่ที่ช่วยให้ผู้พัฒนาสามารถพัฒนาโปรแกรมได้ทันทีโดยไม่จำเป็นต้องทำความเข้าใจรูปแบบการทำงานของระบบเดิมที่มีอยู่ ส่วนต่อประสานโปรแกรมประยุกต์ใหม่จะทำการแยกส่วนของการเก็บข้อมูลและการเขียนโปรแกรมออกจากกัน โดยทำการเก็บข้อมูลที่ใช้ในการปฏิสัมพันธ์กับหน้าจอให้อยู่ในรูปของไฟล์เอ็กซ์เอ็มแอล (XML) เพื่อให้ง่ายต่อการแก้ไขข้อมูลได้ในภายหลังด้วยเครื่องมือการสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทางที่ถูกสร้างขึ้น ส่วนต่อประสานโปรแกรมประยุกต์ใหม่นี้ได้นำหลักการของแบบจำลองวัตถุเอกสาร (Document Object modeling) มาประยุกต์ใช้ในการกำหนดองค์ประกอบต่าง ๆ ของข้อมูลในแต่ละหน้าจอของระบบเก่าให้อยู่ในรูปของวัตถุ ช่วยให้ง่ายต่อการเรียกใช้งานมากขึ้น

1.2 วัตถุประสงค์ของงานวิจัย

1) นำเสนอวิธีการเลือกกลยุทธ์ที่ใช้ในการปรับปรุงระบบเก่าให้เหมาะสมกับงานวิจัย

2) ออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์เพื่อใช้ในการเชื่อมต่อเพื่อรับและส่งข้อมูลกับระบบเก่า

3) ออกแบบและพัฒนาเครื่องมือเพื่อใช้ในการสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

1.3 ข้อจำกัดและขอบเขตของงานวิจัย

1) ทำการออกแบบส่วนต่อประสานโปรแกรมประยุกต์เพื่อใช้พัฒนาแอปพลิเคชันที่ต้องการเชื่อมต่อกับโปรแกรมเลียนแบบเครื่องปลายทาง โดยมีขอบเขตดังต่อไปนี้

a. สามารถนำไปใช้ในการเชื่อมต่อโปรแกรมเลียนแบบเครื่องปลายทางที่ส่งข้อมูลในรูปแบบกระแสข้อมูล 5250 ได้

b. สามารถนำไปใช้ในการพัฒนาด้วยภาษาเขียนโปรแกรม JAVA

2) พัฒนาเครื่องมือตามรอยและบันทึกขั้นตอนการใช้งานโปรแกรมเลียนแบบเครื่องปลายทางต้นแบบ โดยเครื่องมือนี้มีขอบเขตการทำงานดังต่อไปนี้

a. ใช้เก็บข้อมูลหน้าจอจากโปรแกรมเลียนเครื่องแบบปลายทาง 5250
b. การเก็บข้อมูลหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางจะสามารถเก็บข้อมูลได้ทีละหนึ่งเซสชัน (Session)
c. เครื่องมือต้นแบบสามารถเก็บข้อมูลหน้าจอเพิ่มเติมจากหน้าจอเดิมที่เคยเก็บผ่านทางเครื่องมือได้ แต่ไม่สามารถแก้ไขข้อมูลหน้าจอที่เก็บไว้ผ่านทางเครื่องมือได้
d. ข้อมูลที่ถูกจัดเก็บจะอยู่ในรูปของเอกสารเอ็กซ์เอ็มแอล
e. การนำข้อมูลแบบจำลองหน้าจอที่เก็บได้จากเครื่องมือไปใช้งาน ผู้พัฒนาต้องกำหนดที่อยู่ที่ใช้ในการอ้างอิงไฟล์เอ็กเอ็มแอล

3) การทดสอบและการประเมินผลส่วนต่อประสานโปรแกรมประยุกต์ มีขอบเขตการทดลองดังต่อไปนี้

a. จะใช้โปรแกรมเลียนแบบเครื่องปลายทางรุ่น 5250 เป็นตัวทดสอบ
b. ระบบที่ใช้ในการทดลองนั้นคือ ระบบ ICBS ซึ่งเป็นระบบฐานข้อมูลของลูกค้าจากธนาคารสแตนดาร์ดชาร์ดเตอร์ดไทย จำกัด มหาชน
c. การทดสอบจะเป็นการปรับปรุงระบบเก่าซึ่งมีความต้องการตามที่ธนาคารสแตนดาร์ดชาร์เตอร์ดกำหนด
d. การประเมินผลจะทำการเปรียบเทียบขั้นตอนการพัฒนาด้วยไลบรารี่ Extended High Level Language Application Programming Interface (EHLLAPI) และส่วนต่อประสานโปรแกรมประยุกต์ในงานวิจัย และเปรียบเทียบจำนวนบรรทัดโค้ดที่ใช้ในการเขียนโปรแกรม

1.4 ประโยชน์ของงานวิจัย

สามารถนำส่วนต่อประสานโปรแกรมประยุกต์ไปใช้ในการพัฒนาโปรแกรมประยุกต์ที่มีส่วนต่อประสานผู้ใช้สมัยใหม่ซึ่งผู้พิการทางการเห็นสามารถอ่านได้ด้วยโปรแกรมอ่านหน้าจอ

และนำไปใช้ในการต่อยอดพัฒนาโปรแกรมที่ต้องการการเชื่อมต่อกับระบบเก่าผ่านทางโปรแกรมเลียนแบบเครื่องปลายทางได้

1.5 ขั้นตอนและวิธีการวิจัย

1) ศึกษาโครงสร้างระบบเก่า และวิธีการปรับปรุงระบบเก่า
2) ศึกษาเทคโนโลยีที่เกี่ยวข้องกับระบบเก่า
3) ศึกษา วิเคราะห์และเก็บรวมรวมความต้องการของผู้เกี่ยวข้องระบบเก่าและระบบเก่า
4) กำหนดกลยุทธ์ในการปรับปรุงระบบเก่า
5) ออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์
6) พัฒนาเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์กับหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง
7) ทดสอบการทำงานของส่วนต่อประสานโปรแกรมประยุกต์ที่ถูกสร้างขึ้น
8) สรุปผลและจัดทำวิทยานิพนธ์

1.6 ผลงานตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของงานวิทยานิพนธ์นี้ได้รับการตีพิมพ์เป็นบทความวิชาการ ได้แก่

- หัวเรื่อง "Terminal Emulator Wrapper Application Programming" โดย ถาวร ลิ้มวัฒนาชัย, อติวงศ์ สุชาโต และ โปรดปราน บุณยพุกกณะ ในงานประชุมวิชาการ "7th The National Conference on Computing and Information Technology (NCCIT)" ซึ่งจัดขึ้น ณ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ประเทศไทย ระหว่างวันที่ 11-12 พฤษภาคม 2554

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 การปรับปรุงระบบเก่า (Modernization Legacy System)

ระบบเก่าเป็นระบบขนาดใหญ่ที่ถูกพัฒนาและใช้ภายในองค์กรมาเป็นระยะเวลาหลายสิบปี ระบบเก่าของบางองค์กรยังคงถูกใช้เป็นแกนหลักสำคัญในการขับเคลื่อนธุรกิจของบริษัทด้วยปัจจัยในหลาย ๆ ด้านเช่น การเปลี่ยนแปลงกฎภายในองค์กร การเปลี่ยนแปลงกลไกการตลาด การเพิ่มเติมความต้องการของระบบ ส่งผลให้ต้องมีการปรับปรุงระบบเหล่านี้ตามความต้องการที่เปลี่ยนแปลงไปอยู่ตลอดเวลาผ่านผู้รับผิดชอบรุ่นหนึ่งไปยังอีกรุ่นหนึ่ง และเมื่อเทคโนโลยีล้าสมัยก็เริ่มหาผู้ที่จะมารับหน้าที่บำรุงรักษาระบบได้ยากขึ้น ทำให้ค่าใช้จ่ายในการบำรุงรักษาก็สูงขึ้นตามไปด้วย Mihaela Carmen กับ Gabriel Claudiu [5] ได้พูดถึงความยุ่งยากของเทคโนโลยีที่ล้าสมัยอย่างภาษาโคบอล (COBOL) ซึ่งเมื่อต้องการดึงข้อมูลจากไฟล์ข้อมูลชนิดโคบอล หนทางเดียวในการดึงก็คือจำเป็นต้องเขียนการติดต่อด้วยภาษาโคบอลเท่านั้น จะเห็นได้ว่าการแทนที่ด้วยระบบใหม่เลยเป็นวิธีที่ดีที่สุด แต่ปัญหาหลัก ๆ ที่ทำให้หลายองค์กรยังใช้ระบบเก่าเหล่านี้เป็นเพราะเนื่องจากอุปสรรคดังต่อไปนี้

1) ความซับซ้อนของระบบ (Complexity) ด้วยขนาดและความเข้าใจในตัวระบบที่ยากยิ่งของระบบเก่าโดยส่วนใหญ่ ทำให้การจัดการเพื่อลดความซับซ้อนของระบบไม่สามารถประมาณค่าได้ ซึ่งความซับซ้อนนี้ถือเป็นข้อจำกัดมากที่สุดในการปรับปรุงระบบเก่า ความซับซ้อนต่าง ๆ เหล่านี้มาจาก

- ตัวเลือกในการปรับปรุงที่สามารถทำได้หลายทาง ซึ่งแต่ละวิธีอาจต้องเข้าไปเกี่ยวข้องกับระบบธุรกิจที่สำคัญ
- การขาดเอกสารความต้องการของระบบ ซึ่งจำเป็นต้องการผู้เชี่ยวชาญระบบมาให้ข้อมูลเพิ่มเติมในสิ่งที่ขาดหายไป
- ต้องแก้ปัญหาความไม่แน่นอนของการทำงานของระบบเก่า ซึ่งรวมถึง ฟังก์ชันการทำงาน การรวมระบบ และคุณภาพของระบบ
- ต้องการข้อมูลทั้งเชิงปริมาณและเชิงคุณภาพเพื่อเป็นพื้นฐานในการตัดสินใจ

- ต้องค้นหาผลกระทบที่เกิดขึ้นจากการปรับปรุงระบบเก่า ในมุมของผู้ที่เกี่ยวข้องทั้งหลายของระบบเก่า

- ต้องรองรับข้อจำกัดขององค์กรและโครงการและการตัดสินใจร่วมกันทั่วทั้งองค์กร

2) เทคโนโลยีซอฟต์แวร์และกระบวนการวิศวกรรม (Software Technology and Engineering Process) ในการสร้างระบบขนาดใหญ่ จำเป็นที่จะต้องมีความเข้าใจในทฤษฎีเชิงลึกของกระบวนการวิศวกรรม

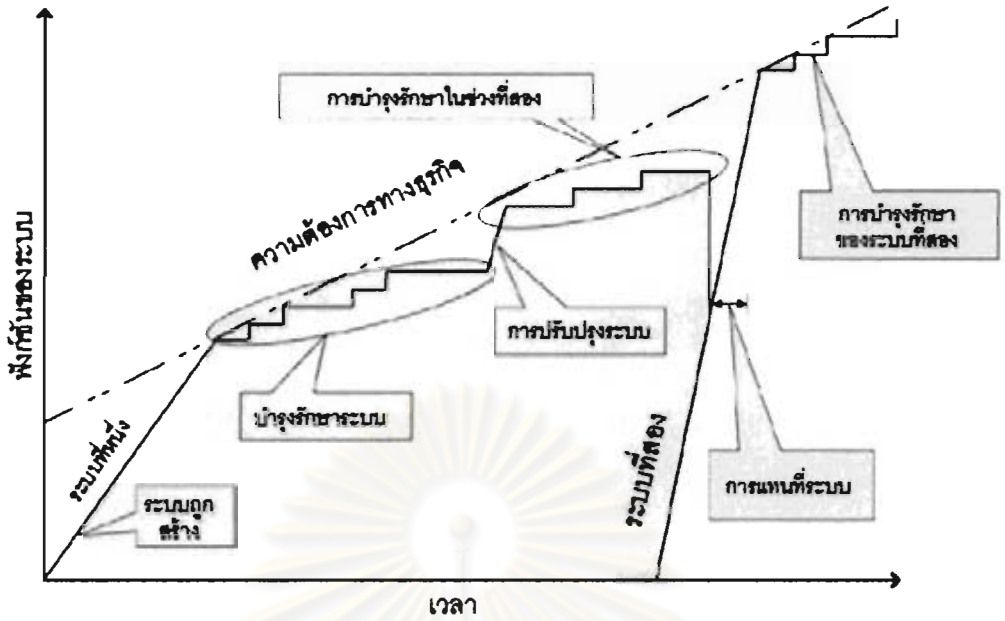
3) ความเสี่ยง (Risk) หลายองค์กรขาดความรู้ความเข้าใจในการบริหารจัดการความเสี่ยงซึ่งอาจเกิดจากความไม่เข้าใจในระบบและวิธีการจัดการความเสี่ยง

4) ส่วนประกอบสำเร็จรูป (Commercial Components) เราอาจต้องใช้ซอฟต์แวร์เสริมจากผู้พัฒนารายอื่นเข้ามาช่วยในการพัฒนาระบบเก่า แต่ซอฟต์แวร์เหล่านี้มักจะมีปัญในตัวของมันเองอยู่แล้ว โดยสังเกตได้จากการอัพเดทเวอร์ชั่นใหม่ๆ อยู่ตลอดเวลา ซึ่งถือเป็นความเสี่ยงที่สำคัญต่อความสเถียรภาพของระบบเก่าที่ถูกปรับปรุง

5) วัตถุประสงค์ทางธุรกิจ (Business Objectives) ด้วยงบประมาณที่มีอยู่ของบริษัท ในการปรับปรุงระบบเก่า โดยทั่วไปองค์กรเหล่านี้มีความต้องการที่จะปรับปรุงระบบเก่าเพื่อให้ได้มาซึ่งฟังก์ชันเดิมที่เคยมีอยู่ด้วยค่าใช้จ่ายที่ลดลง ซึ่งเป็นปัญหาหากว่าเกินงบประมาณทางธุรกิจที่กำหนดก็อาจทำให้การปรับปรุงถูกล้มเลิกความตั้งใจลงไป

2.1.1 วิวัฒนาการระบบเก่า

เมื่อเราพิจารณาจากรูปที่ 2.1 แสดงถึงวิวัฒนาการของระบบเก่า [3] เส้นประตรงแสดงถึงการเติบโตของความต้องการทางธุรกิจ (Business Needs) ในขณะที่เส้นตรงทึบแสดงถึงฟังก์ชันของระบบเก่าที่ถูกพัฒนาเพิ่มขึ้นเรื่อย ๆ เพื่อใช้งาน จะเห็นว่าเมื่อระบบถูกสร้างขึ้นมา ก็จะมีการบำรุงรักษา (Maintenance) ในระยะเวลาต่อมาเพื่อตอบสนองความต้องการทางธุรกิจ เมื่อเวลาผ่านไป ความต้องการมากขึ้นจนไม่สามารถจะบำรุงรักษาต่อไปได้อีก ก็จำต้องเป็นต้องมีการปรับปรุงระบบเก่า (Modernization) และเมื่อไม่สามารถปรับปรุงระบบเก่าได้อีก ก็จะต้องทำการแทนที่ระบบใหม่



รูปที่ 2.1 วิวัฒนาการของระบบเก่า [3]

การเลือกวิธีการนั้นขึ้นอยู่กับความเหมาะสมของช่วงเวลาต่าง ๆ ซึ่งเราสามารถแบ่งความแตกต่างของแต่ละช่วงเวลาดังกล่าวออกได้ดังต่อไปนี้

1) การบำรุงรักษา คือกระบวนทำซ้ำเพื่อเปลี่ยนแปลงแก้ไขระบบต่าง ๆ ซึ่งอาจเป็นการแก้ไขข้อผิดพลาดหรือการเพิ่มเติมฟังก์ชันการทำงานให้แก่ระบบ การบำรุงรักษามีข้อจำกัดดังต่อไปนี้

- มีข้อจำกัดในเรื่องการนำเอาเทคโนโลยีใหม่เข้ามาเพิ่มมาก เพราะไม่มีการปรับปรุงในส่วนของส่วนต่อประสานผู้ใช้งานและสถาปัตยกรรมของระบบ
- ค่าใช้จ่ายในการบำรุงรักษาจะเพิ่มขึ้นไปตามกาลเวลา เพราะจะหาผู้มีความรู้ความเข้าใจในระบบได้ยากขึ้น ซึ่งอาจต้องมีการฝึกฝนให้แก่วิศวกร
- การเปลี่ยนแปลงระบบเก่าเพื่อเพิ่มความต้องการทางธุรกิจลงไปเป็นเรื่องที่ยากขึ้นเรื่อย ๆ และอาจส่งผลกระทบต่อส่วนอื่น ๆ ได้

2) การปรับปรุงระบบเก่า จะมีการเปลี่ยนแปลงที่มากขึ้นแต่จะยังคงระบบเก่าไว้อยู่ส่วนใหญ่จะเป็นการปรับปรุงโครงสร้างของระบบ การเพิ่มเติมฟังก์ชั่นการทำงาน เปลี่ยนแปลงคุณสมบัติของซอฟต์แวร์ การปรับปรุงระบบเก่าจะควรนำมาใช้เมื่อมีความต้องการการเปลี่ยนแปลงที่มากเกินกว่าที่จะบำรุงรักษาได้ แต่ยังคงเก็บกระบวนการทางธุรกิจหลัก ๆ เอาไว้อยู่ การปรับปรุง

ระบบเก่า สามารถจำแนกลักษณะได้จากความแตกต่างของระดับความเข้าใจของระบบที่ต้องใช้ในการปรับปรุงได้ 2 ลักษณะ ดังนี้

- การปรับปรุงแบบกล่องขาว (White Box Modernization) การปรับปรุงแบบกล่องขาวเป็นการปรับปรุงที่ต้องอาศัยความรู้โครงสร้างภายในของระบบเก่า ทำความเข้าใจโปรแกรมซึ่งถือเป็นหลักการของการทำวิศวกรรมผันกลับ (Reverse engineering) โดยจะเกี่ยวข้องกับรูปแบบของโดเมน การดึงข้อมูลออกจากโค้ดโดยใช้กลไกที่มีประสิทธิภาพ การวิเคราะห์โค้ดเพื่อนำไปปรับปรุง

- การปรับปรุงแบบกล่องดำ (Black Box Modernization) การปรับปรุงแบบกล่องดำจะเป็นการศึกษารูปแบบข้อมูลนำเข้าและข้อมูลนำออกของระบบเก่าที่สัมพันธ์กับการดำเนินงานเพื่อให้เข้าใจส่วนต่อประสานของระบบมากขึ้น โดยพื้นฐานของการปรับปรุงระบบเก่าแบบกล่องดำมักจะมาจากกลวิธีการแร็ปปิง (Wrapping) ซึ่งเป็นการห่อหุ้มระบบเก่าด้วยชั้นของซอฟต์แวร์ที่ซ่อนความซับซ้อนที่ไม่จำเป็นออกไปและนำเสนออยู่ในรูปส่วนต่อประสานที่ใหม่กว่า กลวิธีแร็บพิงถูกใช้เพื่อลบความไม่เข้ากันระหว่างส่วนต่อประสานที่สร้างขึ้นโดยซอฟต์แวร์กับส่วนต่อประสานที่ต้องการนำมารวมกัน โดยสนใจแต่เพียงการติดต่อภายนอกเท่านั้น

3) การแทนที่ระบบเก่า เป็นการสร้างระบบใหม่ขึ้นมาทั้งหมด วิธีนี้จะถูกนำมาใช้ก็ต่อเมื่อระบบเก่าไม่สามารถเติมเต็มความต้องการทางธุรกิจได้ทั้งหมดอีกแล้ว หรือเมื่อการปรับปรุงระบบเก่าไม่สามารถทำให้มีประสิทธิภาพได้หรือมีค่าใช้จ่ายที่สูงมาก การแทนที่ระบบเก่าจะมีความเสี่ยงตามมาด้วย ซึ่งจำเป็นต้องมีการประเมินความเสี่ยงเสียก่อน

2.1.2 วิศวกรรมย้อนกลับของระบบเก่า (Software Reengineering)

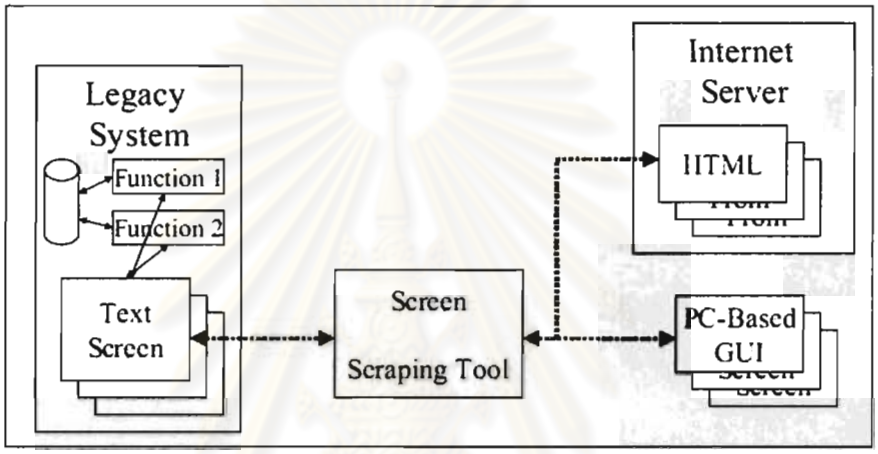
วิศวกรรมย้อนกลับเป็นรูปแบบของการปรับให้ทันสมัยขึ้นโดยการลดข้อจำกัดและการบำรุงรักษาระบบเก่าลงผ่านการใช้เทคโนโลยีสมัยใหม่ กลวิธีการทำวิศวกรรมย้อนกลับแบ่งออกเป็น 6 ประเภท คือ

1) การเปลี่ยนเป้าหมาย (Retargeting) เป็นการปรับปรุงระบบเก่าไปยังระบบฮาร์ดแวร์ใหม่ กลวิธีนี้ทำเพื่อเพิ่มกำลังในการทำงานด้านฮาร์ดแวร์

2) การรื้อจัดหน้าใหม่ (Revamping) เป็นการแทนที่ด้วยส่วนต่อประสานใหม่ ปัจจุบันการปรับส่วนต่อประสานจะอยู่ในรูปของการทำงานผ่านเว็บเบราว์เซอร์แทนที่หน้าจอสีเขียว การรื้อจัดหน้าใหม่ช่วยเพื่อความง่ายต่อการใช้งาน โดยทั่วกลวิธีนี้เป็นการปรับปรุงระบบเก่าใน

ลักษณะกล่องดำ และจะถูกเรียกว่าวิธีสกรีนสเครปปิง (Screen scraping) เทคนิคสกรีนสเครปปิง (รูปที่ 2.2) ซึ่งเทคนิคนี้จะเกี่ยวข้องกับการดึงข้อมูลที่แสดงผลอยู่ในรูปของตัวอักษร และปรับเปลี่ยนแทนที่ด้วยส่วนต่อประสานกราฟิกสมัยใหม่

ปัจจุบันมีเครื่องมือมากมายออกมาจำหน่ายในเชิงธุรกิจ จากมุมมองของระบบเก่า การปรับปรุงในลักษณะนี้ไม่ได้แก้ปัญหาที่เกิดขึ้น เพราะเนื่องจากส่วนประกอบภายในก็ยังคงเหมือนเดิม ไม่ได้มีการลดค่าใช้จ่ายในการบำรุงรักษาแต่อย่างใด เทคนิคสกรีนสเครปปิงยังถูกใช้นำไปสร้างส่วนต่อประสานโปรแกรมประยุกต์ได้อีกด้วย



รูปที่ 2.2 การแร็บพิงโดยกลวิธีสกรีนสเครปปิง (Legacy System Wrapping using Screen Scraping)[6]

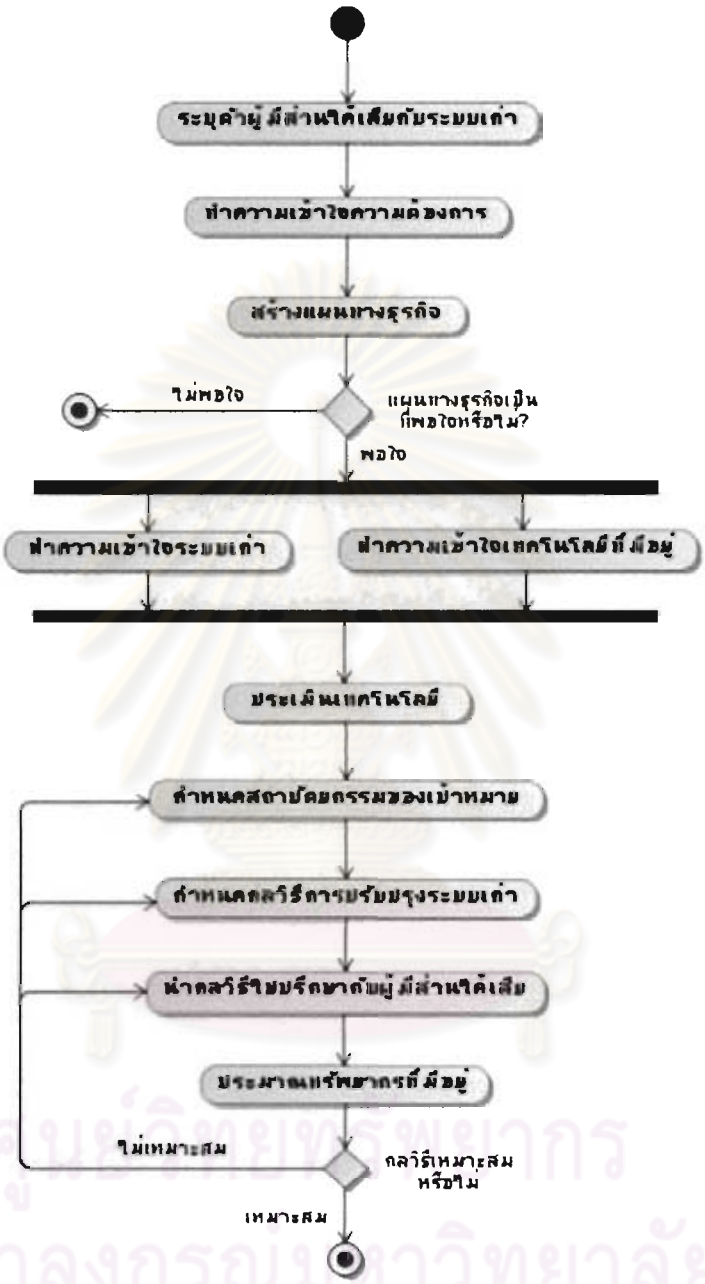
3) ใช้ส่วนประกอบสำเร็จรูป (Commercial Components) เป็นการแทนที่โค้ดด้วยโค้ดสำเร็จรูปที่ขายกันอยู่ในท้องตลาดซอฟต์แวร์ การแทนที่โค้ดอาจจำแนกได้เป็น 2 ประเภท คือ (1) ส่วนประกอบของโครงสร้างพื้นฐาน (Infrastructure) (2) ส่วนประกอบของฟังก์ชั่น

4) การแปลโค้ดของโปรแกรม (Source Code Translation) เป็นการเปลี่ยนภาษาของโปรแกรมในระบบเก่าไปยังภาษาที่สมัยใหม่กว่า เช่น การเปลี่ยนจากภาษาโคบอลไปยังจาวา (Java)

5) ลดโค้ด (Code Reduction) เป็นการนำโค้ดที่ไม่จำเป็นออกไปก่อนที่จะทำการเคลื่อนย้ายโค้ดที่เหลือไปยังแพลตฟอร์มอื่น

6) การเปลี่ยนแปลงฟังก์ชั่น (Functional Transformation) เป็นการปรับปรุงโครงการของโปรแกรม (Program Structure) ความเป็นกลุ่มก้อนของโปรแกรม (Program Modularization) และการทำวิศวกรรมย้อนกลับข้อมูล (Data Reengineering)

2.1.3 **วิธีการปรับปรุงระบบเก่าโดยบริหารความเสี่ยง** (Risk-Managed Modernization Approach)



รูปที่ 2.3 กระบวนการเลือกกลยุทธ์ปรับปรุงระบบเก่า[8]

เพื่อให้การปรับปรุงระบบเก่าเป็นไปอย่างมีประสิทธิภาพ จำเป็นต้องมีการพิจารณากระบวนการต่าง ๆ เพื่อลดความเสี่ยงที่เกิดขึ้น [3] จากรูปที่ 2.3 แสดงขั้นตอนการปรับปรุงระบบเก่าโดยบริหารความเสี่ยง โดยมีรายละเอียดต่าง ๆ ดังนี้

1) ระบุผู้มีส่วนเกี่ยวข้อง (Identify Stakeholder) เป็นการพิจารณาผู้เกี่ยวข้องทั้งหมดในระบบเก่า ได้แก่ ผู้ใช้งานระบบ คนทดสอบระบบ คนพัฒนาระบบ คนดูแลระบบ ลูกค้า เป็นต้น

2) เข้าใจความต้องการ (Understand Requirements) เราสามารถแบ่งความต้องการออกได้เป็น 4 ประเภท ดังนี้

a. ความต้องการของผู้ใช้งานระบบ
b. ความต้องการของระบบ
c. ข้อจำกัดต่าง ๆ
d. ความต้องการที่ไม่ใช่หน้าที่

3) ทำความเข้าใจระบบเก่า (Understand Legacy System) เพื่อให้เข้าใจโครงสร้างทั้งหมดก่อนที่จะทำการปรับปรุงระบบ มีประโยชน์ช่วยให้เราสามารถกำหนดวิธีการปรับปรุงได้อย่างถูกต้องมากยิ่งขึ้น

4) ทำความเข้าใจเทคโนโลยีที่มีอยู่ (Understand Existing Software Technologies) โดยทั่วไปนั้น มีเทคโนโลยีอยู่ 3 กลุ่มที่ถูกนำมาพิจารณาในการปรับปรุงระบบเก่า

a. เทคโนโลยีที่ใช้ในการสร้างระบบ ได้แก่ ภาษาและฐานข้อมูลของระบบ
b. เทคโนโลยีสมัยใหม่ ที่นิยมมาใช้ในการแทนที่ระบบเก่าในช่วงเวลานั้นได้อย่างมีประสิทธิภาพ
c. เทคโนโลยีที่นำเสนอโดยผู้ขายระบบเดิม เทคโนโลยีเหล่านี้มักจะมีความเข้ากันได้และมีเครื่องมือที่ช่วยสนับสนุนให้ผู้พัฒนาสามารถปรับปรุงระบบเก่าได้โดยมีความเสี่ยงน้อยมากขึ้น

5) ประเมินเทคโนโลยี (Evaluate Technology) เมื่อเราเข้าใจเทคโนโลยีที่มีอยู่แล้วนั้น การประเมินผลเทคโนโลยีก็เป็นส่วนแรกของการสร้างส่วนประกอบต่าง ๆ ที่เกี่ยวข้องกับสถาปัตยกรรมที่ใช้ในระบบ

6) กำหนดสถาปัตยกรรมของระบบเป้าหมาย (Define Target Architecture) เพื่อแสดงถึงสถาปัตยกรรมที่จะใช้กับระบบใหม่ที่ได้รับการปรับปรุง

7) กำหนดกลยุทธ์การปรับปรุงระบบ (Define Modernization Strategy) ด้วยกลยุทธ์ที่มีประสิทธิภาพจะเป็นตัวกำหนดการเปลี่ยนแปลงจากระบบเก่าไปสู่ระบบใหม่ ในขณะที่มีการ

ปรับปรุงระบบ เทคโนโลยีอาจมีการเปลี่ยนแปลง ต้องการความรู้เพิ่มเติมเกี่ยวกับระบบเก่าที่มีอยู่ หรือความต้องการของผู้ใช้งานเปลี่ยน กลยุทธ์ในการปรับปรุงจะต้องรองรับเรื่องต่าง ๆ เหล่านี้ได้ด้วย เพื่อช่วยในการลดค่าใช้จ่าย เวลา และความเสี่ยงต่าง ๆ ที่อาจเกิดขึ้นได้

8) ทำข้อตกลงกันในเรื่องการปรับปรุงระบบกับผู้ที่เกี่ยวข้อง (Reconcile Modernization Strategy with Stakeholders)

9) ประมาณทรัพยากรต่าง ๆ ที่ต้องใช้ในการปรับปรุงระบบ (Estimate Resources for Modernization Strategy)

2.2 โปรแกรมเลียนแบบเครื่องปลายทาง (Terminal Emulator)

ในระบบเก่า เครื่องปลายทางรุ่น 3270 (3270 Terminal) เป็นระบบเมนเฟรมของบริษัทไอบีเอ็ม (IBM) ที่ได้รับความนิยมมากที่สุดในช่วงปี 70 และ 80 และยังคงถูกใช้มาจนถึงปัจจุบัน รูปแบบเครื่องปลายทางนี้ถูกออกแบบเพื่อใช้แสดงผลแบบเมทริกโดยมีขนาด 24 แถว และ 80 คอลัมน์ ในการเขียนโปรแกรม ผู้พัฒนาจำเป็นต้องระบุตำแหน่งเป็นแถวและคอลัมน์เพื่อแสดงผลข้อมูลในแต่ละหน้า ข้อมูลเหล่านี้จะถูกส่งออกมาในรูปแบบกระแสข้อมูล (Data stream) ซึ่งใช้เป็นตัวบอกว่าข้อมูลที่ส่งออกมานั้นจะให้แสดงผลออกมาอย่างไรและแสดงตรงตำแหน่งไหนบนหน้าจอ และด้วยความที่เป็นเครื่องปลายทางรุ่น 3270 ดังนั้นกระแสข้อมูลที่ถูกส่งออกมานั้นจึงมีการกำหนดวิธีการส่งออกมาเป็นมาตรฐานของกระแสข้อมูล 3270 และเมื่อไอบีเอ็มผลิตเครื่องปลายทางรุ่น 5250 กระแสข้อมูลเหล่านี้ก็ถูกพัฒนาออกมาเป็นมาตรฐานอีกมาตรฐาน คือ กระแสข้อมูล 5250 และสุดท้ายจึงกลายมาเป็นการส่งข้อมูลในรูปแบบ ASCII และถูกใช้กันโดยแพร่หลายในโปรแกรมประยุกต์เครื่องแม่ข่าย

เมื่อคอมพิวเตอร์สมัยใหม่เป็นที่นิยมมากขึ้น การเข้าถึงระบบเมนเฟรมจึงถูกปรับเปลี่ยนจากการเข้าถึงโดยตรงในสมัยก่อนเป็นการเข้าถึงผ่านซอฟต์แวร์โปรแกรมเลียนแบบเครื่องปลายทาง [2] ซอฟต์แวร์เหล่านี้จะทำการสร้างการเชื่อมต่อระหว่างเครื่องลูกข่ายไปยังเครื่องแม่ข่ายเพอร์ซันนอลคอมมิวนิเคชัน (Personal communication) เป็นตัวอย่างซอฟต์แวร์โปรแกรมเลียนแบบเครื่องปลายซึ่งเป็นผลิตภัณฑ์จาก ไอบีเอ็มเช่นเดียวกัน โดยลักษณะการใช้งานก็ยังคงไม่แตกต่างกัน การแสดงผลทั้งหมดก็ยังเป็นลักษณะข้อความเช่นเดียวกัน หรือเรียกได้ว่าเป็นการจำลองหน้าจอเครื่องปลายทางเพื่อให้ผู้ใช้ ใช้งานผ่านระบบโครงข่าย

โปรแกรมเลียนแบบเครื่องปลายทางมีอยู่สองประเภทหลัก ๆ ที่ใช้ในการติดต่อกับระบบเก่าทั่วไป คือ ประเภทเชิงกลุ่มระเบียน (Block oriented category) เช่น Terminal tn3270 และ tn5250 ถูกใช้เชื่อมต่อกับระบบ IBM Cics และ As/400 และ ประเภทเชิงกระแสข้อมูล (Stream oriented one) เช่น Terminal ของตระกูล Vt: (Vt52, Vt100, Vt220, Vt320, Vt420) ประเภททั้งสองนี้มีแนวคิดเรื่องแบบจำลองการปฏิสัมพันธ์ (Interaction Model) และโปรโตคอลการติดต่อสื่อสาร (Communication Protocol) เพื่อแลกเปลี่ยนข้อมูลที่แตกต่างกัน

ในการแสดงผลของโปรแกรมเลียนแบบเครื่องปลายทางรุ่น 3270 หน้าจอที่แสดงผลถูกเรียกว่าพรีเซนเตชั่นสเปซ (Presentation space) ใช้ในการแสดงข้อมูลต่าง ๆ และรับรองการแสดงได้สี่ขนาด โดยอ้างอิงจากแบบจำลอง 2, 3, 4 และ 5 เราสามารถเลือกใช้แบบจำลองขนาดใหญ่ได้ ถึงแม้ว่าหน้าจอนั้นจะไม่รองรับก็ตาม

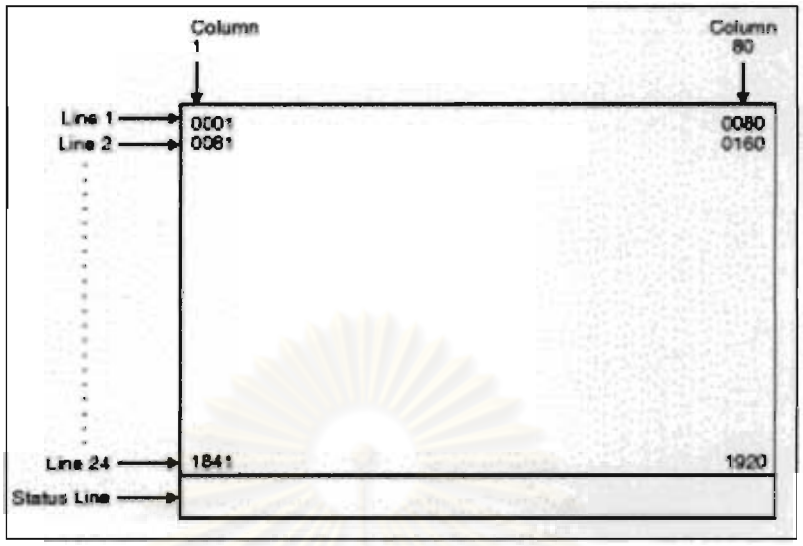
แบบจำลองหน้าจอ (Screen model) ของ 3270 สนับสนุนจำนวนแถวและคอลัมน์ในแต่ละแบบจำลองตามตารางที่ 1 และตัวอย่างหน้าจอแสดงผลแบบจำลองที่ 2 ในรูปที่ 2

ตารางที่ 2.1 แบบจำลองหน้าจอของโปรแกรมเลียนแบบเครื่องปลายทาง tn3270

แบบจำลอง (Model)	แถว (Rows)	คอลัมน์ (Columns)
2	24	80
3	32	80
4	43	80
5	27	132

2.2.1 **ที่พักข้อมูลหน้าจอ** (Screen Buffer)

ข้อมูลที่ถูกแสดงในพรีเซนเตชั่นสเปซจะเก็บไว้ในที่พักข้อมูลหน้าจอ ในการเข้าถึงข้อมูลจำต้องทราบตำแหน่งของพรีเซนเตชั่นสเปซที่มีข้อมูลบรรจุอยู่ แบบแผนตำแหน่งในแต่ละแบบจำลองหน้าจอ พรีเซนเตชั่นสเปซเริ่มต้นที่ตำแหน่งที่ 1 และสิ้นสุดที่จำนวนสูงสุดของตำแหน่งพรีเซนเตชั่นสเปซของแบบจำลองหน้าจอนั้น ๆ เช่น แบบจำลองหน้าจอที่ 2 มีทั้งสิ้น 1920 ตำแหน่ง



รูปที่ 2.4 หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

2.2.2 **ฟิลด์ในพรีเซนเตชันสเปซ** (Fields in the Presentation Space)

ฟิลด์ในพรีเซนเตชั่นสเปซแบ่งออกได้เป็น 2 ชนิด คือ

1) ฟิลด์มีรูปแบบ (Formatted Field)

2) ฟิลด์ไม่มีรูปแบบ (Unformatted Field)

เราสามารถเขียนข้อมูลใด ๆ ก็ได้ลงไปในฟิลด์ที่ไม่มีรูปแบบ แต่ต้องแน่ใจว่าไม่เกินในตำแหน่งของแบบจำลองหน้าจอ และต้องมั่นใจว่าหน้าจอนั้นไม่มีฟิลด์ที่ถูกออกแบบมาให้อ่านข้อมูลในบริเวณที่กำหนด ในขณะที่ฟิลด์ที่มีรูปแบบโดยทั่วไปมักจะมีข้อมูลที่ได้รับการระบุชนิดบรรจุอยู่ การเขียนโปรแกรมเพื่อส่งข้อมูลลงในฟิลด์เหล่านี้ต้องพิจารณาเงื่อนไขต่าง ๆ ด้วย

2.2.3 **ลักษณะประจำฟิลด์** (Field Attributes)

แต่ละฟิลด์ในพรีเซนเตชั่นสเปซจะประกอบไปด้วยคุณลักษณะที่ใช้กำหนดข้อมูลแบบหน่วยไบท์ให้กับตำแหน่งนั้น ดังนี้

1) ขอบเขตเริ่มต้นของฟิลด์

2) ฟิลด์ที่ถูกป้องกันและไม่ถูกป้องกัน

3) ชนิดของข้อมูลที่ยอมให้กำหนดลงในฟิลด์

4) การปรากฏของฟิลด์

5) ตำแหน่งฟิลด์ซึ่งถูกระบุด้วยเคอร์เซอร์

6) การแก้ไขของฟิลด์

2.3 กลวิธีสกรีนสเครปปิงของระบบเก่า

การดึงข้อมูลด้วยวิธีสกรีนสเครปปิงจำเป็นต้องใช้ส่วนต่อประสานโปรแกรมประยุกต์จากไลบารี่เฉพาะที่ถูกสร้างขึ้นโดยผู้พัฒนานั้น ๆ ส่วนต่อประสานโปรแกรมประยุกต์เหล่านี้จะทำหน้าที่เชื่อมต่อกับระบบเก่าและสามารถส่งคำสั่งและดึงข้อมูลต่างๆ ออกมาจัดรูปแบบใหม่ได้ ส่วนต่อประสานโปรแกรมประยุกต์ที่ถูกใช้ในการทำสกรีนสเครปปิงกันอย่างแพร่หลายถูกเรียกว่า HLLAPI (High Level Language Application Programming Interface) ซึ่งกำหนดโดยบริษัทไอบีเอ็ม เพื่อใช้ในการเชื่อมต่อระบบเมนเฟรมผ่าน กระแสข้อมูล 3270 และถูกพัฒนาเพิ่มเติมเพื่อรองรับการส่งข้อมูลผ่าน กระแสข้อมูล 5250 (5250 Data Stream) ในภายหลังและถูกเปลี่ยนชื่อเป็น EHLLAPI หลังจากนั้นไอบีเอ็มได้ร่วมมือกับอินเตอร์เนทเอนจิเนียริงทาส์กฟอซ (Internet Engineering Taskforce) [7] เพื่อพัฒนาวิธีและข้อกำหนดในการเขียนโปรแกรมระดับสูง เพื่อเข้าถึงข้อมูลคอมพิวเตอร์แม่ข่าย ช่วยให้ผู้พัฒนาทั่วไปสามารถนำส่วนต่อประสาน (Interface) ที่มีคลาสและเมธอดที่ได้มาตรฐานไปพัฒนาแอปพลิเคชันที่ใช้ในการเข้าถึงข้อมูลเครื่องคอมพิวเตอร์แม่ข่าย

2.3.1 High Level Language Application Programming Interface (HLLAPI)

ไลบรารี่ HLLAPI [2] นั้น อาจไม่ต้องการการปฏิสัมพันธ์จากผู้ใช้งาน หมายความว่าสามารถสั่งการควบคุมระหว่างโปรแกรมหนึ่งไปยังโปรแกรมเลียนแบบเครื่องปลายทางได้เลย

การใช้งานจะเป็นการเรียกฟังก์ชั่นผ่านไลบรารี่ โดยมีเมธอดที่ใช้ในการเรียกหนึ่งเมธอด ขึ้นอยู่กับว่าจะส่งหมายเลขฟังก์ชั่นไหนไปเพื่อร้องขอการใช้งานคำสั่งนั้น รูปแบบการเรียกโดยทั่วไปมักอยู่ในรูปดังนี้

HLLAPI(functionNo, dataString, length, returnCode,...)

ข้อมูลที่ส่งผ่านฟังก์ชั่นหลัก ๆ มีดังต่อไปนี้

1. โปรแกรมที่จะทำการเรียกใช้งาน อาจระบุได้ด้วย Process ID
2. หมายเลขฟังก์ชั่น (function no)
3. ข้อมูลสตริงที่ต้องการส่งผ่านไปยังโปรแกรมเลียนแบบเครื่องปลายทาง (Data string)

4. ความยาวของข้อมูลสตริง (Data length)

5. ตำแหน่งของข้อมูล (Position)

6. ข้อมูลบอกสถานะที่ส่งกลับมา (Return code)

ในการดำเนินการ ผู้ใช้จำเป็นต้องทราบตำแหน่งของข้อมูลที่จะส่ง วิธีการเชื่อมต่อผ่านหมายเลขฟังก์ชั่นต่าง ๆ โดยฟังก์ชั่นแต่ละฟังก์ชั่นอาจมีความต้องการการใช้งานฟังก์ชั่นอื่น ๆ ก่อนหน้า เช่น ก่อนการส่งข้อมูลไปยังโปรแกรมเลียนแบบเครื่องปลายทาง

สำหรับ EHLLAPI จะมีลักษณะการใช้งานเช่นเดียวกับ HLLAPI แต่ได้เพิ่มเติมส่วนของเมธอดต่าง ๆ ที่รองรับการทำงานร่วมกับเทอร์มินอล 5250 ซึ่งมีคุณสมบัติเพิ่มเติมเข้ามา

2.3.2 Host Access Class Library (HACL)

HACL [8] ได้นำเสนอกลุ่มของคลาสและเมธอดที่ช่วยในการเข้าถึงข้อมูลบนหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง และสามารถพัฒนาโปรแกรมแบบเชิงวัตถุได้ โดยไลบรารี่ HACL จะมีวัตถุ Session ที่ห้อหุ้มวิธีการและลักษณะเฉพาะของการเชื่อมต่อกับระบบเก่าเอาไว้และเปิดให้ใช้งานได้ โดยทั่วไป เมื่อเกิดข้อผิดพลาด HACL ก็จะทำการโยนข้อผิดพลาดออกมาในรูปแบบของวัตถุ ECLErr การจับข้อผิดพลาดเหล่าจะใช้ try/catch ดักจับข้อผิดพลาดเหล่านั้น

2.4 แบบจำลองวัตถุเอกสาร (Document Object Modeling)

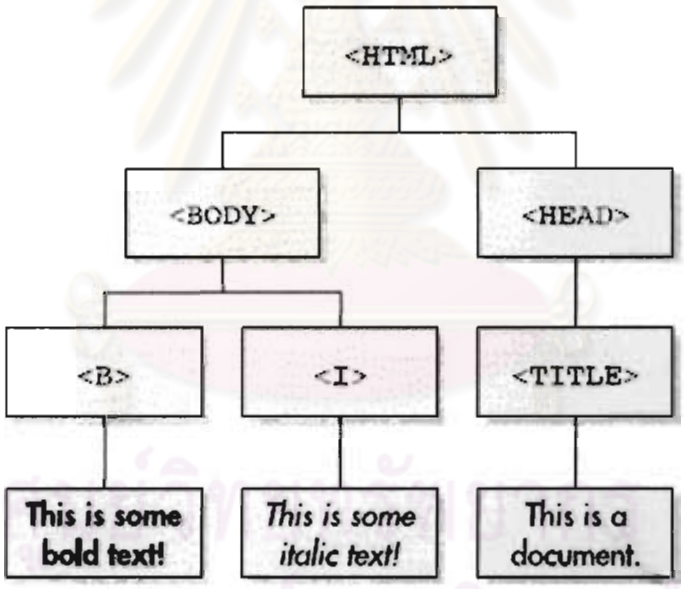
ข้อกำหนดแบบจำลองวัตถุเอกสาร [9] ได้เสนอมาตรฐานกลุ่มของส่วนต่อประสานการเขียนโปรแกรม (set of programming interface) ที่ใช้ในการจัดการกับข้อมูลที่ถูกเก็บอยู่ในไฟล์เอ็กซ์เอ็มแอล , เอ็ชทีเอ็มแอล (HTML) และชนิดของเอกสารในรูปแบบอื่น ๆ ที่ถูกเก็บเอาไว้เป็นโครงสร้าง การใช้ส่วนต่อประสานโปรแกรมประยุกต์ของ DOM จะช่วยให้ผู้พัฒนาโปรแกรมสามารถเขียนโค้ดโดยใช้เพียงกลุ่มของเมธอดเพียงกลุ่มเดียวแต่ช่วยจัดการการทำงานได้ครอบคลุมวัตถุทั้งหมด

DOM เป็นกลุ่มของส่วนต่อประสานโปรแกรมประยุกต์ที่ใช้ในการเขียนโปรแกรมที่ใช้อธิบายถึงวิธีการจัดการข้อมูลที่ถูกเก็บเอาไว้ในโครงสร้างของเอกสารเอ็กซ์เอ็มแอล และ เอชทีเอ็มแอล

```
<HTML>
<HEAD>
<TITLE>This is a document.</TITLE>
</HEAD>
<BODY>
<B>This is some bold text!</B>
<I>This is some italic text!</I>
</BODY>
</HTML>
```

รูปที่ 2.5 โค้ดตัวอย่าง HTML[9]

เอกสาร DOM ได้ถูกนำเสนอออกมาในรูปแบบโครงสร้างต้นไม้ ซึ่งแต่ละจุดของต้นไม้เรียกว่า โหนด (Node) ดังแสดงในตัวอย่างโค้ดรูปที่ 2.5 จะเห็นได้ว่าแต่ละกล่องในภาพที่ จะถูกเรียกว่า โหนด แต่ละโหนดแทนให้กับวัตถุหนึ่งวัตถุ (Object) ที่อยู่ในเนื้อหาของเอกสาร โหนดทั้งหลายจะมีชื่อเฉพาะของมันเอง ขึ้นอยู่กับว่ามันอยู่ส่วนไหนของแผนภาพต้นไม้เอกสารและความสัมพันธ์ของโหนดนั้น ๆ ว่าเกี่ยวข้องกับโหนดในตำแหน่งอื่น ๆ อย่างไร



รูปที่ 2.6 แผนภาพต้นไม้เอกสาร [9]

เอกสารแต่ละอันจะมี โหนดราก (Root Node) หนึ่งอันเสมอ ซึ่งใช้แทนจุดเริ่มต้นของโครงสร้างต้นไม้ทั้งหมด จากภาพที่ จะเห็นว่า โหนดรากของโครงสร้างต้นไม้นี้ คือ ป้ายระบุ <HTML> และ ถ้ามีมากกว่าหนึ่งโหนด โหนดอื่น ๆ ที่อยู่ในระดับชั้นต่ำลงมาจะถือว่าเป็น โหนดลูก (Child Node) ของโหนดที่ระบุ และโหนดลูกแต่ละโหนดนี้จะถือว่าเป็น ลูกของโหนดครอบครัวนั้น ๆ พิจารณาจากภาพ ป้ายระบุ <HEAD> คือ ลูกของโหนด <HTML> และในทำนองเดียวกัน ป้ายระบุ

<TITLE> ก็จะเป็นโหนดลูกของแท็ก <HEAD> ด้วย ความสัมพันธ์ในลักษณะนี้จะถูกนำไปใช้อ้างอิงผ่านทางส่วนต่อประสานโปรแกรมประยุกต์ของ DOM โหนดที่ไม่มีลูกเลยจะเรียกว่า โหนดกิ่ง ซึ่งโหนดเหล่านี้จะเป็นโหนดที่อยู่ปลายสุดของโครงสร้างต้นไม้ ในที่นี้ก็คือ ข้อความที่เป็นเนื้ออยู่ภายใต้ป้ายระบุ <B>, <I>, <TITLE>

เมื่อมาพิจารณาจากความสัมพันธ์ต่อ โหนดที่อยู่ในระดับเดียวกัน ซึ่งแปลงปันพ่อและลูกเดียวกัน จะถูกเรียกว่าโหนดพี่น้อง (Sibling Node) ซึ่งในที่นี้คือ ป้ายระบุ <BODY> และ <HEAD> โหนดไม่มีการจำกัดประเภทของวัตถุว่าต้องเป็นอะไร เช่น <HEAD> และ <TITLE> แต่ทุก ๆ วัตถุที่อยู่ในเอกสารหนึ่ง ๆ เราสามารถแทนเป็นโหนดหนึ่งโหนดได้

2.4.1 ชนิดของโหนดในแบบจำลองเอกสารเชิงวัตถุ

ในข้อกำหนดของ DOM ได้มีการกำหนดประเภทของโหนดออกเป็น 12 ประเภทที่แตกต่างกัน ที่สามารถบรรจุอยู่ในเอกสารได้ โหนดแต่ละประเภทมีกลุ่มของแอททริบิวและการใช้งานในตัวของมันเอง ตารางต่อไปนี้แสดงประเภทของโหนดที่มีอยู่ใน เอกสาร DOM Level 2

ตารางที่ 2. 2 ประเภทของโหนดในแบบจำลองเอกสารเชิงวัตถุ

ประเภทของโหนด	คำอธิบาย
Element	แสดงส่วนย่อยในเอกสาร HTML หรือ XML ซึ่งเขียนโดยแท็ค ดังรูป โหนดเหล่านี้เป็นประเภทของส่วนย่อย
Attribute	แสดงคุณสมบัติของส่วนย่อย
Text	แสดงเนื้อหาภายในส่วนย่อยหนึ่ง ๆ
CDATASection	แสดงส่วนของข้อมูลตัวอักษรภายในเอกสารเอ็กซ์เอ็มแอล
Document	แสดงโหนดรากของเอกสารหนึ่ง ๆ
DocumentType	แต่ละโหนดเอกสารจะมีโหนด documentType ที่มีไว้แสดงรายการของคุณลักษณะเฉพาะที่ถูกกำหนดเอาไว้ในเอกสาร
DocumentFragment	ถูกนำมาใช้เพื่อแยกส่วนของเอกสารจากการประมวลผล
ProcessingInstruction	คือขั้นตอนเฉพาะที่ถูกใช้ในตัวประมวลผลเอกสาร
Entity	คือเอนทิตีในเอกสารเอ็กซ์เอ็มแอล
EntityReference	คือเอนทิตีอ้างอิงในเอกสารเอ็กซ์เอ็มแอล
Notation	คือสัญกรณ์ในเอกสารเอ็กซ์เอ็มแอล สัญกรณ์ไม่มีโหนดแม่

2.5 งานวิจัยที่เกี่ยวข้อง

มีงานวิจัยที่เกี่ยวข้องหลายงานที่ได้พูดถึงการปรับปรุงระบบเก่าด้วยวิธีการต่าง ๆ แต่งานวิจัยเหล่านี้ได้แก้ไขด้วยวิธีการปรับปรุงแบบกล่องขาว ซึ่งจำเป็นต้องเข้าไปเกี่ยวข้องกับโค้ดของระบบเก่า ในวิทยานิพนธ์นี้มุ่งเน้นไปที่การปรับปรุงส่วนต่อประสานใหม่ด้วยวิธีกล่องดำ เพราะการให้ผู้พิการทำงานร่วมกับบริษัทที่ใช้ระบบเก่าได้นั้น ไม่ควรจะต้องมีการแก้ไขโค้ดภายในระบบ และต้องใช้ระยะเวลาอันรวดเร็ว ด้วยเหตุนี้การออกแบบส่วนต่อประสานโปรแกรมประยุกต์จึงใช้เทคนิคกล่องดำเพื่อช่วยในการติดต่อกับระบบเก่า งานวิจัยต่าง ๆ มีดังต่อไปนี้

2.5.1 Legacy Application Modeling With Attachmate Verastream Host Integrator : Programmatic Integration vs. Traditional Screen Scraping

บริษัท Attachmate [10] ได้นำเสนอประวัติความเป็นมาของการทำสกรีนสเครปปิงด้วยส่วนต่อประสานโปรแกรมประยุกต์ HLLAPI และแสดงให้เห็นถึงข้อจำกัดของเทคนิคดังกล่าว ไม่ว่าจะเป็นเรื่องประสิทธิภาพของการทำงานซึ่งต้องส่งดาต้าสตรีมไปยังเครื่องลูกข่ายก่อนที่จะวิเคราะห์หน้าได้และกำหนดกระบวนการทำงานทำให้เสียเวลาในการทำงานในส่วนนี้ หรือการซิงโครไนท์ซึ่งอาจเกิดความผิดพลาดได้หากระบบเมนเฟรมเกิดปัญหาและตัดการเชื่อมต่อ ทำให้การส่งคีย์คำสั่งและการรับข้อมูลไม่เป็นไปตามที่กำหนดไว้ หรือปัญหาความยุ่งยากในการแก้ไขแอปพลิเคชันหากมีการเพิ่มหน้าจอเทอร์มินอลที่ต้องใช้ในการทำงาน และได้นำเสนอเครื่องมือที่ช่วยในการพัฒนาแอปพลิเคชันเพื่อแก้ไขปัญหาข้างต้นด้วยการสร้างแบบจำลองการนำล่องส่วนต่าง ๆ ของแอปพลิเคชันเมนเฟรมและการเข้าถึงข้อมูลที่จำเป็น ซึ่งการใช้แบบจำลองนี้สามารถดำเนินการคำสั่งหลายคำสั่งได้ด้วยการเรียกใช้งานเพียงครั้งเดียวช่วยเพิ่มประสิทธิภาพในการทำงาน มีการกำหนดรองรับข้อผิดพลาดในรูปแบบได้ตั้งแต่ขั้นตอนการสร้างแบบจำลองและสามารถเปลี่ยนแปลงแก้ไขแบบจำลองได้โดยไม่กระทบการเขียนโค้ด

2.5.2 Accelerating Integration with Verastream Host Integrator : High-level abstraction is the key

งานวิจัยฉบับนี้ [11] เป็นของบริษัท Attachmate เช่นเดียวกัน โดยได้นำเสนอการใช้เครื่องมือในการสร้างแบบจำลองและเก็บข้อมูลหน้าจอเมนเฟรม ที่ข้อมูลมีลักษณะเป็นตารางและเรียกดูข้อมูลโดยใช้ส่วนต่อประสานโปรแกรมประยุกต์ระดับสูงที่สร้างขึ้นมาด้วยคำสั่งมาตรฐาน

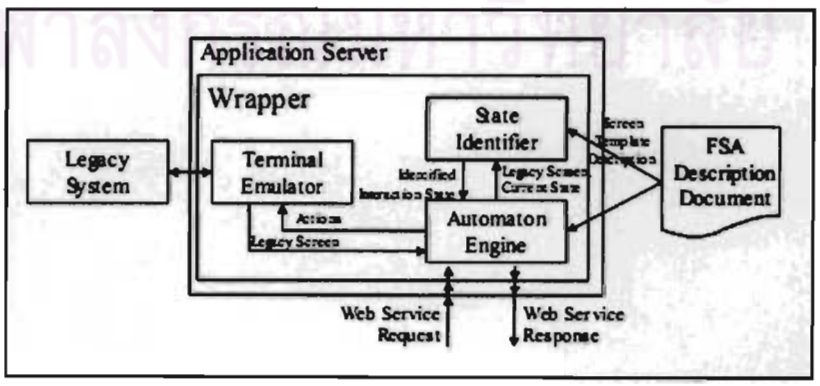
เอสคิวแอล (SQL) และเปรียบเทียบโค้ดกับการเขียนโปรแกรมด้วยส่วนต่อประสานโปรแกรมประยุกต์แบบเก่าที่ยังคงใช้ HLLAPI ซึ่งแสดงให้เห็นว่าช่วยลดขั้นตอนการเขียนโปรแกรมได้มาก

2.5.3 Migrating Interactive Legacy Systems To Web Services

G. Canfora, et al. [6] ได้นำระเบียบวิธีสกรีนสเครปปิง ซึ่งในงานวิจัยนี้ถูกเรียกว่าระเบียบวิธีแร็ปปิง มาเสนอในการปรับปรุงฟังก์ชันเชิงโต้ตอบของระบบเก่าให้อยู่ในรูปแบบการเข้าถึงด้วยเว็บเซอร์วิส (Web Service) ได้ โดยระเบียบวิธีแร็บพปิงถือเป็นระเบียบวิธีหนึ่งที่ใช้ในการสื่อสารผ่านส่วนต่อประสานผู้ใช้งานของระบบเก่า ซึ่งระบบเก่าเหล่านี้จะใช้โปรแกรมเลียนแบบเครื่องปลายทางเชื่อมต่อกับแม่ข่ายเพื่อช่วยดึงข้อมูลออกมาแสดงผลทางคอมพิวเตอร์

ตัวแร็ปเปอร์ (The Wrapper) จะรับผิดชอบหน้าที่การติดต่อกับระบบเก่าในการดำเนินการเหตุการณ์ที่เป็นไปได้ซึ่งเกี่ยวข้องกับยูสเคส (Use Case) ที่ได้กำหนดไว้ตั้งแต่ตอนเริ่มต้น ซึ่งแบบจำลองการปฏิสัมพันธ์ระหว่างระบบเก่าและผู้ใช้งานช่วงการดำเนินการตามยูสเคสที่ระบุไว้ถูกแสดงออกเป็นไฟไนท์สเตทออโตมาตอน (Finite state Automaton)

สุดท้ายระบบมีออโตเมตอนเอนจิน เป็นศูนย์กลางในการดึงข้อมูล และมีตัวระบุสถานะ (State Identifier) คอยรับผิดชอบการทำความเข้าใจว่าสกรีนเทมเพลต (Screen template) ซึ่งถูกเก็บข้อมูลต่างๆ เอาไว้เพื่อใช้ระบุหน้าจออันไหนจากสถานะที่เป็นไปนี้ แล้วตรงกันกับหน้าจอที่ส่งคืนกลับมาโดยระบบเก่า โดยที่โปรแกรมเลียนแบบเครื่องปลายทางทำหน้าที่ควบคุมการติดต่อสื่อสารระหว่างตัวแร็บพเปอร์และระบบเก่า เอกสารอธิบายเอฟเอสเอ (FSA Description Document) เป็นคลังข้อมูลเก็บไฟล์ข้อมูลต่าง ๆ ของคำอธิบายสถานะการปฏิสัมพันธ์ การเปลี่ยนแปลง สกรีนเทมเพลตและการกระทำที่จะส่งให้ออโตเมตอนเอนจินให้อยู่ในรูปภาษาเอ็กซ์เอ็มแอล (XML) และสามารถเรียกใช้งานได้ด้วยเว็บเซอร์วิส (Web service)



รูปที่ 2.7 โครงสร้างของตัวแร็บพปิง (The architecture of the wrapper)

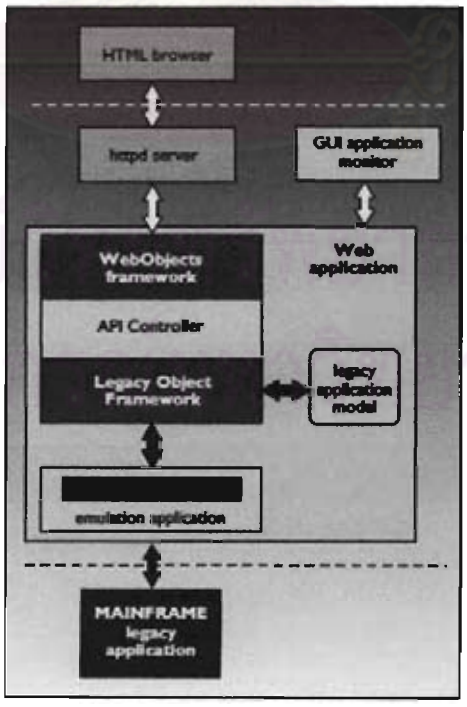
2.5.4 Internationalizing Mainframe Applications

C. Durand [12] ได้นำเสนองานวิจัยนี้เป็นกรณีศึกษาการใช้กลวิธีสกรีนสเครปปิง ในการปรับปรุงระบบเก่า ที่มีหน้าจอที่ต้องใช้งานกว่า 400 หน้าจอ และ รูปแบบรายงานมากกว่า 100 แบบ ต้องทำภายในเวลาและงบที่จำกัด โดยการปรับปรุงนี้จะเปลี่ยนการทำงานบนหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง ให้เป็นการทำงานผ่านหน้าเว็บเบราเซอร์ โดยแบ่งการแปลงหน้าจอออกเป็นสามหมวด คือ ข้อความแบบคงที่ (Static text) ข้อความผันแปร (Dynamic text) และข้อมูลแอปพลิเคชัน

2.5.5 Legacy Object Modeling

W. B. Noffsinger, et al. [13] พยายามชี้ให้เห็นถึงข้อจำกัดของการพัฒนาโปรแกรมด้วย HLLAPI เช่นกัน พร้อมทั้งได้นำเสนอแบบจำลองวัตถุระบบเก่า (Legacy Object Modeling) เพื่อแยกส่วนชั้นการเขียนโค้ดด้วย HLLAPI ออกจากกัน โดยแบบจำลองดังกล่าวจะหุ้มข้อมูลระบบเก่า และประพฤติตัวเหมือนรูปแบบของกลุ่มวัตถุ ดังแสดงในรูปที่ 2.8

ด้วยกรอบงานนี้ จะช่วยให้องค์กรสามารถเปิดการติดต่อข้อมูลระบบเก่าให้กับระบบอื่นแบบเปิดกว้างได้ โดยแบบจำลองวัตถุระบบเก่าจะยอมให้แอปพลิเคชั่นใหม่เข้าถึงข้อมูลระบบเก่าผ่านส่วนต่อประสารเทอร์มินัลโดยไม่ต้องเปลี่ยนสิ่งใดทางฝั่งเซอร์เวอร์ของระบบเก่า



รูปที่ 2.8 แผนภาพการทำงานของกรอบงานเชิงวัตถุของระบบเก่า

จากงานวิจัยข้างต้นที่ได้กล่าวมา จะเห็นได้ว่ามีงานวิจัยหลายงานที่นำเสนอวิธีแก้ไขปัญหาด้วยการนำกลวิธีสกรีนสเครปปิงมาช่วยในการปรับปรุงระบบเก่า บริษัทแอทแทชเมทได้นำเสนอเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์กับหน้าจอต่าง ๆ ของระบบเก่าและให้ผู้พัฒนาสามารถเข้าถึงข้อมูลได้ด้วยคำสั่งเอสคิวแอล หรือจะเป็น W. B. Noffsinger ที่ออกแบบแบบจำลองวัตถุระบบเก่าและเครื่องมือเพื่อเก็บข้อมูลการปฏิสัมพันธ์และเปลี่ยนแปลงได้ในภายหลัง ทั้งสองงานวิจัยนี้พยายามแก้ไขปัญหาข้อจำกัดจากการพัฒนาด้วยเทคนิค Screen Scraping และถูกนำไปใช้ในทางธุรกิจที่ยังคงใช้ระบบเก่าอยู่ในปัจจุบัน

ในขณะที่ G. Canfora ได้นำเทคนิคสกรีนสเครปปิงมาเสนอในการปรับปรุงฟังก์ชันเชิงโต้ตอบของระบบเก่าให้อยู่ในรูปแบบการเข้าถึงด้วยเว็บเซอร์วิสได้ ซึ่งภายในงานวิจัยนี้ได้นำเสนอภาพของแบบจำลองการปฏิสัมพันธ์ของผู้ใช้งานเอาไว้ โดยผู้ใช้งานระบบจะเป็นผู้ส่งข้อมูลนำเข้า รับข้อมูลส่งออก และส่งคำสั่งเพื่อให้หน้าจอเปลี่ยนสถานะ และมีตัวตีความช่วยในการตีความหน้าจอนั้น ๆ เขาได้แบ่งการเก็บข้อมูลออกเป็น 3 ประเภท คือ 1.ฟิลด์ป้ายชื่อ (Label Field) 2.ฟิลด์ข้อมูลส่งออก (Output Field) และ 3.ฟิลด์ข้อมูลนำเข้า (Input Field) งานวิจัยของ C. Durand ได้นำเสนอกรณีศึกษาในการพัฒนาส่วนต่อประสานผู้ใช้งานใหม่ในรูปแบบเว็บเบราเซอร์ด้วยเครื่องมือการเก็บข้อมูลการปฏิสัมพันธ์กับระบบเก่าด้วยเทคนิคสกรีนสเครปปิง งานวิจัยทั้งสองข้างต้นนี้ไม่ได้พูดถึงการเก็บข้อมูลบางประเภท เช่น การเก็บข้อมูลประเภทตาราง การเก็บข้อมูลเมื่อมีหน้าจอพิเศษเกิดขึ้นตามเงื่อนไขของการทำงานของโปรแกรมเลียนแบบเครื่องปลายทาง เป็นต้น ซึ่งการออกแบบส่วนต่อประสานโปรแกรมประยุกต์ใหม่ต้องการประเภทของข้อมูลที่ละเอียดขึ้น ข้อมูลส่งออกอาจแยกเป็นข้อมูลส่งออกแบบเดี่ยวหรือแบบตารางได้ ซึ่งทำให้การคืนค่าแก่ผู้พัฒนามีประเภทที่แตกต่างกัน ส่วนต่อประสานโปรแกรมประยุกต์ใหม่ยังต้องคอยจัดการเซสชันของระบบที่อาจทำงานได้มากกว่าหนึ่งเซสชันพร้อม ๆ กันอย่างอัตโนมัติ และรองรับการแสดงผลของข้อผิดพลาดต่าง ๆ ที่เกิดขึ้นจากระบบด้วย เช่น ข้อผิดพลาดที่เกิดจากการกรอกรหัสผ่านผิดของผู้ใช้งานจะแสดงผลออกเป็นข้อความที่ตำแหน่งใด ๆ ของหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง ซึ่งจำเป็นต้องนำมาพิจารณาเพื่อแก้ปัญหาการปฏิสัมพันธ์กับข้อมูลบนระบบเก่าเพื่อนำไปออกแบบเอพีไอในงานวิจัยฉบับนี้

บทที่ 3

ขั้นตอนและวิธีการดำเนินงานวิจัย

ในบทนี้จะกล่าวถึงขั้นตอนในการดำเนินงานวิจัย โดยได้อ้างอิงขั้นตอนการดำเนินการวิเคราห์ระบบตามวิธีการปรับปรุงระบบเก่าโดยบริหารความเสี่ยงตามทฤษฎีที่เกี่ยวข้องที่ได้กล่าวในบทที่ 2 ซึ่งขั้นตอนการดำเนินงานวิจัยต่างๆ นี้ จะเป็นข้อมูลที่นำไปใช้ในการตัดสินใจเลือกกลยุทธ์การปรับปรุงระบบเก่าที่เหมาะสมที่สุด และเป็นที่มาของการออกแบบส่วนต่อประสานโปรแกรมยุกต์และเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทางในงานวิจัยฉบับนี้

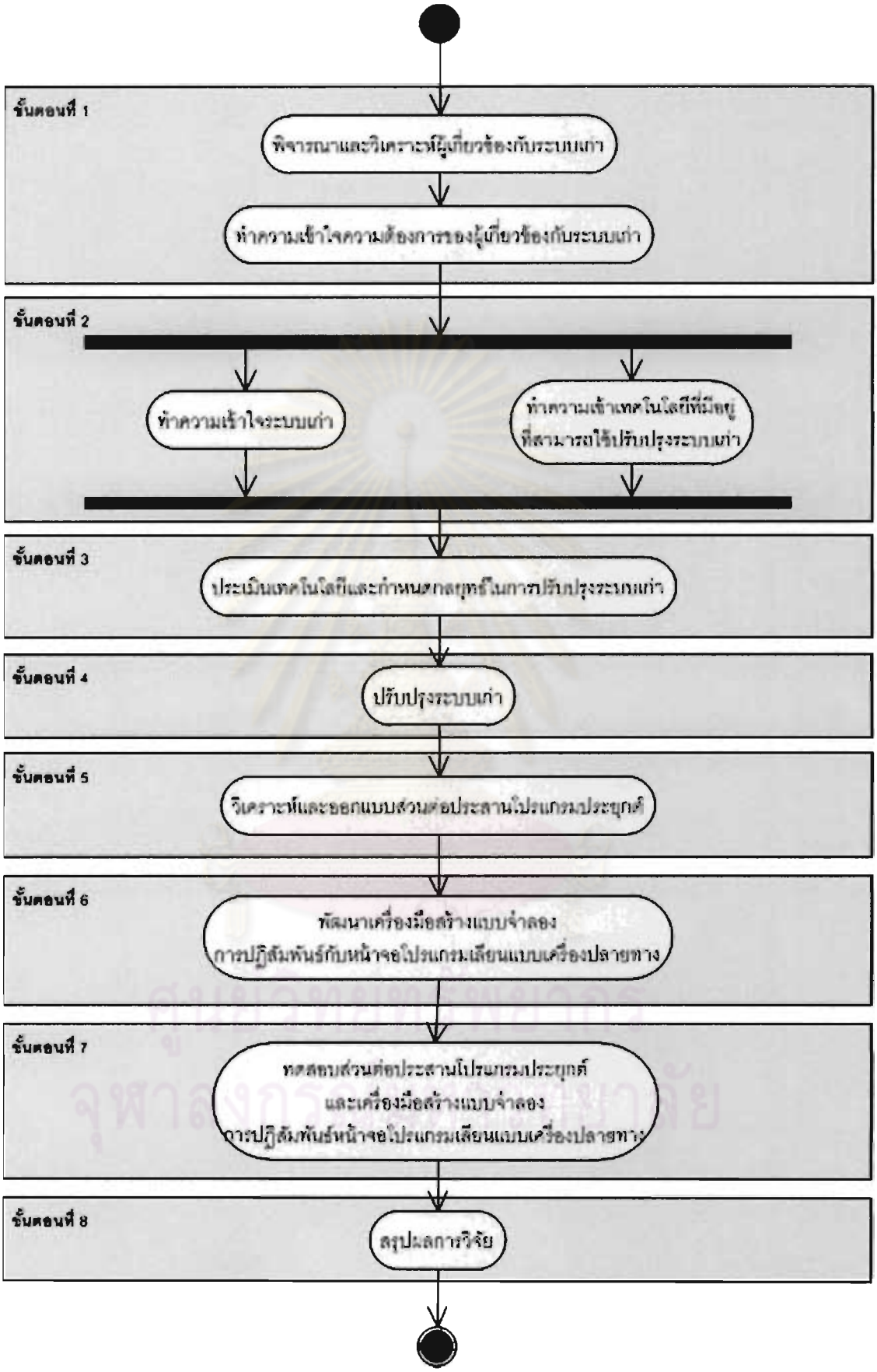
สำหรับขั้นตอนของการดำเนินงานวิจัยจะเป็นไปตามรูปที่ 3.1 และมีรายละเอียดต่าง ๆ ดังต่อไปนี้

3.1 การพิจารณาและวิเคราะห์ผู้เกี่ยวข้องกับระบบเก่า

ในขั้นตอนนี้ เราได้ทำการแบ่งประเภทของผู้เกี่ยวข้องและวิเคราะห์ความต้องการต่าง ๆ ของผู้เกี่ยวข้อง เพื่อให้ทราบถึงความต้องการทั้งหมดของผู้เกี่ยวข้อง เนื่องจากผู้เกี่ยวข้องแต่ละคนจะมีมุมมองต่อระบบเก่าที่แตกต่างกัน เราต้องศึกษาปัญหาโดยคำนึงถึงมุมมองของผู้ใช้งานทั้งสองประเภทดังกล่าวเพื่อให้การปรับปรุงระบบไม่ส่งผลกระทบต่อบุคคลในกลุ่มใดกลุ่มหนึ่ง

เราได้แบ่งประเภทของผู้เกี่ยวข้องกับระบบเก่าของธนาคารได้ดังนี้

1) กลุ่มผู้ใช้งาน ในที่นี้เราหมายถึงกลุ่มผู้ใช้งานที่เกี่ยวข้องกับการให้บริการตอบรับทางโทรศัพท์ (Call Center) เราสามารถแบ่งกลุ่มผู้ใช้งานออกได้เป็น 2 ประเภท คือ กลุ่มผู้ใช้งานทั่วไปและกลุ่มผู้ใช้งานพิการทางการเห็น
2) ทีมพัฒนา ได้แก่ นักเขียนโปรแกรม ผู้ออกแบบระบบ ผู้ทดสอบระบบ ซึ่งบุคคลเหล่านี้อาจจะมีความคุ้นเคยหรือไม่คุ้นเคยกับระบบเก่าก็ได้
3) ผู้บำรุงรักษาระบบเก่า ได้แก่นักเขียนโปรแกรม หรือ วิศวกร ที่ต้องคอยบำรุงรักษาระบบที่ปรับปรุงใหม่ต่อไป
4) กลุ่มผู้เกี่ยวข้องต่าง ๆ ใน องค์กร ผู้ซึ่งมีอำนาจในการตัดสินใจ ในการดำเนินงานต่างๆ ที่เกี่ยวข้องกับระบบเก่า และยังอยู่ในเงื่อนไขทางธุรกิจของธนาคาร



รูปที่ 3.1 ขั้นตอนการดำเนินงานของวิทยานิพนธ์

ซึ่งกลุ่มผู้เกี่ยวข้องเหล่านี้จะเป็นปัจจัยสำคัญในการกำหนดกลยุทธ์การปรับปรุงระบบเก่า ซึ่งไม่ได้เพียงแต่ปรับปรุงเพื่อผู้พิการทางการเห็น แต่ยังรวมไปถึงในเรื่องการบำรุงรักษา การปรับปรุงเปลี่ยนแปลงในอนาคตอีกด้วย

3.2 ทำความเข้าใจความต้องการของผู้เกี่ยวข้องกับระบบเก่า

เราพิจารณาถึงความต้องการของผู้เกี่ยวข้องต่าง ๆ โดยมีรายละเอียดดังต่อไปนี้

3.2.1 ความต้องการของธนาคาร

ธนาคารมีความต้องการเปิดรับผู้พิการทางการเห็นเข้าทำงานเป็นพนักงานของธนาคาร โดยเริ่มแรกมีเป้าหมายในการว่าจ้างงานผู้พิการทางการเห็นจำนวนสองคนเพื่อทำงานร่วมกับระบบเก่า และจะมีการเปิดรับเพิ่มขึ้นในภายหลัง โดยมีข้อสรุปที่จะให้ผู้พิการทางการเห็นทำงานในส่วนของพนักงานตอบรับทางโทรศัพท์ โดยรับหน้าที่หลัก ๆ สองหน้าที่ในปัจจุบันดังนี้

1) รับหน้าที่เปิดบัตรเครดิต (Card Activation) สำหรับลูกค้าที่ทำบัตรใหม่และต้องการเปิดการใช้งาน
2) รับหน้าที่โทรแจ้งและสอบถามลูกค้าในกรณีพัสดุต่าง ๆ ถูกส่งไปไม่ถึงลูกค้าและถูกตีกลับมายังธนาคาร (Mail Return)

ซึ่งสองหน้าที่จะเกี่ยวข้องกับระบบหลักสองระบบคือ ระบบไอซีบีเอส (ICBS) และ ระบบซีซีเอ็มเอส (CCMS) ระบบดังกล่าวนี้ ธนาคารมีความต้องการให้ผู้พิการทางการเห็นเริ่มต้นทำงานโดยมีความต้องการดังนี้คือ

3.2.1.1 ความต้องการในการทำงานร่วมกับระบบไอซีบีเอส

1) เรียกดูข้อมูลพื้นฐานของบัญชีกระแสรายวันโดยใช้หมายเลขบัญชีในการสืบค้นข้อมูล
    a. ข้อมูลของลูกค้า แบ่งได้เป็น 3 ประเภท ดังนี้
        i. ข้อมูลส่วนบุคคลทั่วไป เช่น ชื่อ นามสกุล วันเกิด
        ii. ข้อมูลที่ใช้ในการติดต่อ เช่น ที่อยู่ เบอร์โทรศัพท์
        iii. ข้อมูลเกี่ยวกับที่ทำงานปัจจุบันของลูกค้าบัญชีนั้น ๆ
    b. Basic Note Data แบ่งได้เป็น 3 ประเภท ดังนี้
        i. ข้อมูลบัญชีของลูกค้า เช่น หมายเลขบัญชี

ii. ข้อมูลดุลบัญชี เช่น วงเงินปัจจุบัน บันทึกแนบของวงเงินปัจจุบัน

iii. บันทึกกิจกรรมการเงินต่าง ๆ เช่น การใช้วงเงินล่าสุด การฝากครั้งล่าสุด การถอนครั้งล่าสุด

c. ข้อมูลการชี้แจงสินทรัพย์

i. Statement Control Data ประกอบไปด้วยรหัสต่าง ๆ

ii. ข้อมูลการชี้แจงสินทรัพย์

d. การตัดยอดบัตรเครดิต/ การประกันภัยต่างๆ

e. ข้อมูลเกี่ยวกับ Prior Statement

2) เรียกดูข้อมูลพื้นฐานของบัญชีเงินกู้โดยใช้หมายเลขบัญชีในการสืบค้นข้อมูล

a. ข้อมูลของลูกค้าเช่นเดียวกับบัญชีกระแสรายวัน

b. Basic Note Data เช่นเดียวกับบัญชีกระแสรายวัน

c. ตารางการจ่ายเงิน

i. เป็นการดึงข้อมูลออกมาแสดงผลในรูปแบบตาราง ซึ่งแสดงประวัติการจ่ายเงินของลูกค้า

d. ข้อมูลใบเสร็จและยอดค้างจ่าย

e. ข้อมูลบันทึกต่าง ๆ เกี่ยวกับบัญชี

f. ข้อมูลประวัติการทำกิจกรรมบัญชีของลูกค้า

3) จัดการข้อมูลของลูกค้า

a. ฟังก์ชันแสดงผลข้อมูลต่าง ๆ ของลูกค้า ผู้ใช้งานสามารถใช้ฟังก์ชันนี้เพื่อเช็คว่าข้อมูลประเภทไหนที่สามารถเปลี่ยนแปลงได้ และข้อมูลไหนเปลี่ยนแปลงไม่ได้ โดยมีรายละเอียดข้อมูลต่าง ๆ ดังนี้

i. Account Number

ii. CIF Number

iii. Last Change CIF info

iv. Last Change Account Address

v. Telephone number

vi. Email

vii. Account Open date

viii. Last Transaction

ix. Last Re-Issue

x. PLC Card

b. **ฟังก์ชันแก้ไขข้อมูลลูกค้า**

i. **แก้ไขที่อยู่**

ii. **แก้ไขเบอร์โทรศัพท์**

iii. **เพิ่มเติมข้อมูล เช่น การโทรเข้า หรือ โทรออก**

โดยหลังจากที่มีการแก้ไข ข้อมูลทั้งหมดจะยังไม่ถูกอัพเดททันที จำเป็นต้องมีคนตรวจสอบข้อมูลเสียก่อน

3.2.1.2 **ความต้องการของระบบซีซีเอ็มเอส**

1) **เรียกดูข้อมูลส่วนตัวของลูกค้า เช่น ชื่อ นามสกุล ที่อยู่ ข้อมูลเหล่านี้อาจไม่ใช่ข้อมูลเดียวกับข้อมูลจากระบบไอซีบีเอส**
2) **เปิดบัตรเครดิตใหม่ให้กับลูกค้า**
3) **บันทึกข้อมูลในการเปิดบัตรให้กับลูกค้า**
4) **เรียกดูข้อมูลเกี่ยวกับบัตรเครดิตเพื่อตอบคำถามให้กับลูกค้า**

ในการทำงานทั้งหมดนี้ ทางธนาคารจะมีเป้าหมายและข้อกำหนดที่เป็นมาตรฐานที่ต้องการให้พนักงานปฏิบัติ ด้วยเหตุนี้จึงมีการวัดผลการทำงานของพนักงานออกเป็นตัวเลขทั้งหมด โดยมีข้อกำหนดต่าง ๆ ดังต่อไปนี้

- **ในการรับโทรศัพท์ทุก ๆ 100 สาย ต้องสามารถรับได้ 98 สาย**
- **เจ้าหน้าที่ต้องรับสายให้ทันภายในเวลา 15 วินาที แต่ไม่ต่ำกว่า 85 ครั้ง ต่อ 100 สาย**
- **ผลของแบบสอบถามวัดความพึงพอใจของลูกค้า ต้องได้เกณฑ์ตอบรับมากกว่า 85 เปอร์เซนต์**
- **พนักงานใหม่ที่เข้ามา ต้องได้รับการฝึกอบรมเป็นเวลา หนึ่งเดือนครึ่ง โดยมีการแบ่งงานในส่วนต่าง ๆ ในระบบของธนาคาร**
- **มีการเก็บข้อมูลการทำงานและประเมินผล**

3.2.2 **ความต้องการของผู้ใช้งาน**

ความต้องการของผู้ใช้งานถูกแบ่งเป็น 2 กลุ่ม คือ ความต้องการของผู้พิการทางการเห็น ซึ่งมีความต้องการที่จะสามารถทำงานร่วมกับระบบเก่าผ่านทางโปรแกรมเลียนแบบเครื่องปลายทางได้ด้วยโปรแกรมอ่านหน้าจอ โดยขณะที่ผู้ใช้งานทั่วไปไม่จำเป็นต้องศึกษาระบบที่จะถูกสร้างขึ้นใหม่อีก และสามารถให้ความรู้แก่ผู้พิการทางการเห็นที่เข้าใหม่ได้

3.2.2.1 ความต้องการของผู้พิการทางการเห็น

3.2.2.1.1 ความต้องการเชิงหน้าที่ (Functional Requirement)

1) ผู้พิการทางการเห็นต้องสามารถใช้งานโปรแกรมเลียนแบบเครื่องปลายทางด้วยแป้นพิมพ์ได้อย่างครบถ้วน
2) การปรับปรุงโปรแกรมเลียนแบบเครื่องปลายทางต้องสามารถใช้งานร่วมกับโปรแกรมอ่านหน้าจอได้
3) การปรับปรุงส่วนต่อประสานต้องสามารถทำงานร่วมกับระบบปฏิบัติการวินโดว์ 2000 ได้
4) ผู้พิการทางการเห็นต้องสามารถเข้าถึงข้อมูลในระบบเก่าได้อย่างครบถ้วน

3.2.2.1.2 ความต้องการไม่ใช่หน้าที่ (Non-Functional Requirement)

1) ส่วนต่อประสานที่ได้รับการปรับปรุงต้องมีความเสถียรในการทำงาน
2) ส่วนต่อประสานที่ได้รับการปรับปรุงต้องใช้งานง่ายไม่ซับซ้อน
3) ส่วนต่อประสานที่ได้รับการปรับปรุงต้องแจ้งเตือนการทำงานเพิ่มเติมเฉพาะให้แก่ผู้พิการทางการเห็น

3.2.2.2 ความต้องการของพนักงานทั่วไป

1) ระบบเก่าที่ปรับปรุงแล้วต้องใช้ระยะเวลาในการเรียนรู้ไม่มากนัก
2) ระบบเก่าที่ปรับปรุงแล้วสามารถดูแลจัดการในส่วนของการใช้งานของผู้พิการทางการเห็นได้ง่าย

3) ระบบเก่าที่ปรับปรุงนั้นพนักงานทั่วไปต้องสามารถเข้าถึงร่วมกับผู้พิการทางการเห็นได้

3.2.3 ความต้องการของผู้ปรับปรุงระบบเก่า

มีความต้องการพัฒนาระบบที่สอดคล้องกับแนวทางธุรกิจของทางธนาคาร โดยไม่เพิ่มต้นทุนในด้านฮาร์ดแวร์หรือซอฟต์แวร์จำนวนมหาศาล และสามารถพัฒนาได้ในระยะเวลาอันรวดเร็ว

3.2.4 ความต้องการของผู้ดูแลระบบที่ปรับปรุงใหม่

การดูแลระบบจะต้องไม่ควรที่จะเข้าไปแก้ไขระบบที่ปรับปรุงใหม่โดยไม่จำเป็น ซึ่งอาจเป็นผลให้การทำงานหยุดชะงักได้ ผู้บำรุงรักษาจะต้องสามารถเรียนรู้ขั้นตอน และวิธีการแก้ไขข้อมูล และสามารถแก้ไขข้อมูลได้ในระยะเวลาอันรวดเร็ว เพื่อรองรับการปรับเปลี่ยนของข้อมูลที่อยู่บนหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางอยู่ตลอดเวลา

3.3 ทำความเข้าใจระบบเก่าและประเมินเทคโนโลยีที่มีอยู่

3.3.1 ทำความเข้าใจระบบเก่า

ธนาคารสแตนดาร์ดชาร์เตอร์ดไทย จำกัด มหาชน มีระบบหลักในการทำงานอยู่บนเมนเฟรมโอเอส 400 (OS/400) และพนักงานทำงานอยู่บนระบบโปรแกรม 2 ระบบหลัก คือ 1) ไอซีบีเอสและ 2) ซีซีเอ็มเอส ซึ่งระบบเหล่านี้ได้มีการเชื่อมต่อข้อมูลระบบผ่านเครือข่ายอินเตอร์เน็ตร่วมกับธนาคารสาขาอื่น ๆ ทั่วโลก

ในการใช้งานทั้ง 2 ระบบดังกล่าว ผู้ใช้งานจะทำงานผ่านทางโปรแกรมเลียนแบบเครื่องปลายทางเพื่อรับและส่งข้อมูลให้กับระบบเก่า ซึ่งโปรแกรมนี้มีผู้ใช้งานภายในองค์กรมากกว่า 100 คน

3.3.2 เทคโนโลยีที่มีอยู่ในองค์กร

จากการศึกษาพบว่า เทคโนโลยีที่จำเป็นต้องใช้ในการพิจารณาเพื่อเลือกกลยุทธ์การปรับปรุงระบบเก่ามีอยู่ 2 ประเภทคือ

1) เทคโนโลยีช่วยเหลือผู้พิการทางการเห็น

เพื่อให้ผู้พิการทางสามารถทำงานร่วมกับคอมพิวเตอร์ได้ จำเป็นต้องมีเครื่องมือช่วยเหลือ ซึ่งจากศึกษาพบว่า เครื่องมือช่วยเหลือที่สำคัญที่ให้คนพิการสามารถทำงานร่วมกับ

คอมพิวเตอร์ได้นั้น คือ โปรแกรมอ่านหน้าจอ โดยมีคุณสมบัติที่ช่วยดึงข้อความจากหน้าจอเพื่อส่งไปยังตัวประมวลผลเสียงสังเคราะห์ เพื่อสังเคราะห์เสียงออกมาให้ผู้พิการทางการเห็นทราบถึงสถานะต่างๆ บนหน้าจอคอมพิวเตอร์ โปรแกรมอ่านหน้าจอที่ได้รับความนิยมของผู้พิการทางการเห็นในประเทศไทยมีอยู่ 2 ประเภทคือ 1) โปรแกรมอ่านหน้าจอ JAWS และ 2) โปรแกรมอ่านหน้าจอ เอ็นวีดีเอ (NVDA) โปรแกรมอ่านหน้าจอจอวส์ (JAWS) เป็นโปรแกรมที่ผู้ใช้ต้องซื้อลิขสิทธิ์เพื่อการใช้งาน โดยมีค่าลิขสิทธิ์ในปัจจุบันอยู่ที่ประมาณ 40,000 บาทต่อเครื่องคอมพิวเตอร์ 3 เครื่อง ในขณะที่โปรแกรมอ่านหน้าจอ เอ็นวีดีเอ เป็นซอฟต์แวร์โอเพ่นซอส (Open Source) ซึ่งเปิดให้ผู้พัฒนาอิสระสามารถช่วยกันพัฒนาโปรแกรมได้ และเปิดให้ผู้พิการทางการเห็นสามารถใช้งานได้โดยไม่เสียค่าลิขสิทธิ์ใด ๆ

นอกเหนือจากโปรแกรมอ่านหน้าจอแล้วนั้น ยังมีเทคโนโลยีช่วยเหลืออื่น ๆ ที่ช่วยอำนวยความสะดวกให้แก่ผู้พิการทางการเห็น เช่น คีย์บอร์ดอักษรเบลล์ เป็นต้น

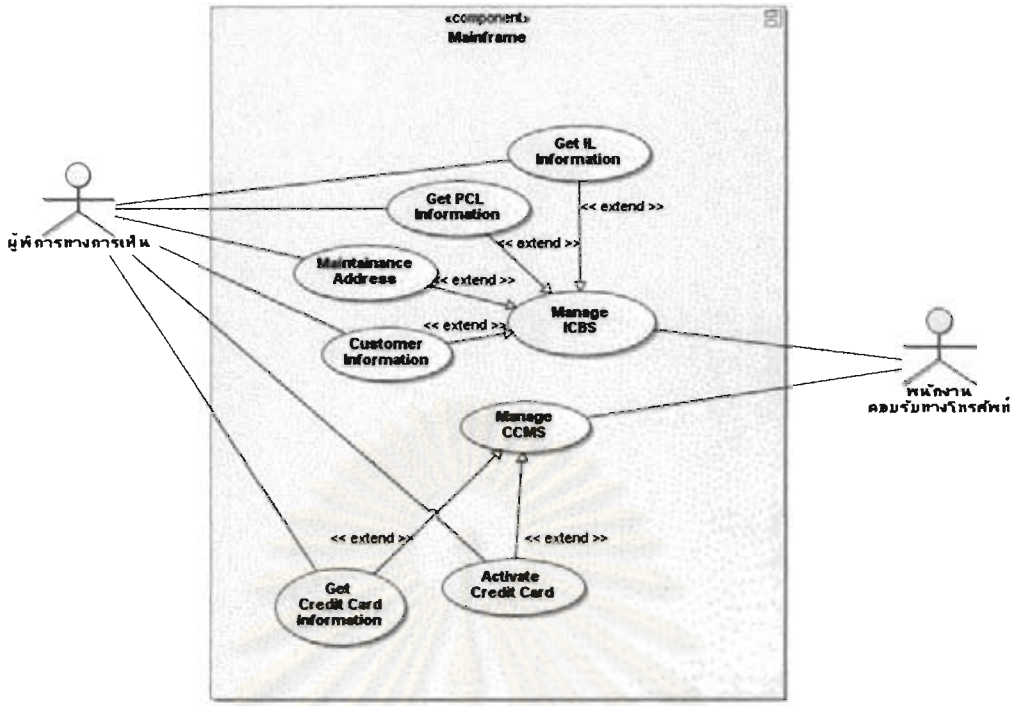
2) เทคโนโลยีในการพัฒนาโปรแกรม

ในปัจจุบันทางธนาคารจะแผนกที่ช่วยในการบำรุงรักษาซึ่งจะคอยปรับเปลี่ยนแก้ไขโปรแกรมในส่วนแอปพลิเคชันสมัยใหม่ ในส่วนที่ต้องดำเนินการปรับปรุงระบบเก่า จะถูกกระทำผ่านทางธนาคารสาขาต่างประเทศ โดยสาขาในประเทศไทยสามารถปรับเปลี่ยนแก้ไขข้อมูลของระบบเก่าได้ภายใต้เงื่อนไขที่กำหนด เทคโนโลยีที่เกี่ยวข้องต่างๆ มีดังต่อไปนี้

a. เทคโนโลยีประเภทฐานข้อมูล ที่ต้องเขียนโปรแกรมติดต่อด้วยภาษาโคบอล
b. เทคโนโลยีสมัยใหม่ คือ ภาษาจาวา ซีชาร์ป และวิชวล เบสิก
c. เทคโนโลยีที่เจ้าของโปรแกรมเดิมสนับสนุนได้ คือ เครื่องมือช่วยเหลือในการปรับปรุงแก้ไขหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง และไลบรารี่ประเภทต่าง ๆ ที่ช่วยในการทำเทคนิคสกรีนสเครปปิง

3.4 การกำหนดกลยุทธ์เพื่อใช้ในการปรับปรุงระบบเก่า

จากข้อมูลในขั้นตอนต่าง ๆ ข้างต้น ทำให้เห็นถึงผู้เกี่ยวข้องและหน้าที่ต่าง ๆ ของผู้เกี่ยวข้องซึ่งแสดงออกเป็นแผนภาพยูสเคสได้ดังรูปที่ 3.2



รูปที่ 3.2 แผนภาพยูสเคสการทำงานของผู้พิการทางการเห็นและพนักงานตอบรับทางโทรศัพท์

จากการวิเคราะห์ได้เห็นว่าการปรับปรุงระบบเก่ามีความเกี่ยวข้องกับคนหลายประเภทโดยมีเป้าหมายหลักในการปรับปรุงคือ เพื่อให้ผู้พิการทางการเห็นสามารถทำงานร่วมกับธนาคารได้ตามความต้องการที่ได้กำหนดเอาไว้ในหัวข้อที่ 3.2.1

โดยการปรับปรุงต้องคำนึงถึงทิศทางของธนาคารเป็นสำคัญ โดยสามารถสรุปได้ดังนี้

1) การปรับปรุงระบบเก่าต้องใช้ระยะเวลาอันสั้น
2) การปรับปรุงระบบเก่าต้องไม่ส่งผลให้เกิดความเสี่ยงต่อการสูญหายของกระบวนการทางธุรกิจของบริษัท
    a. ต้องไม่มีการแก้ไขโค้ดฐานข้อมูลภายในระบบ
    b. ต้องพัฒนาและทดสอบภายในกฎของทางธนาคาร
3) การปรับปรุงระบบเก่า ไม่ควรจะต้องใช้เทคโนโลยีช่วยเหลือผู้พิการทางการเห็นเพิ่มเติมที่มีค่าใช้จ่ายที่สูงมาก
4) ธนาคารมีความต้องการให้ผู้พิการทางการเห็นสามารถทำงานด้านคอมพิวเตอร์ได้ ด้วยโปรแกรมอ่านหน้าจอ เอ็นวีดีเอ ซึ่งไม่มีค่าลิขสิทธิ์เพิ่มเติม

5) ธนาคารมีความต้องการที่จะเป็นองค์กรต้นแบบในการเผยแพร่โปรแกรมอ่านหน้าจอ เอ็นวีดีเอ

ด้วยเหตุนี้ผู้วิจัยจึงไม่สามารถกำหนดกลยุทธ์การปรับปรุงระบบเก่าด้วยวิธีการปรับปรุงระบบเก่าแบบกล่องขาวได้ เนื่องจากการปรับปรุงในลักษณะนี้จำเป็นต้องมีการเข้าไปแก้ไขโค้ดภายในระบบเก่า ผู้วิจัยจึงมุ่งเน้นไปที่การปรับปรุงระบบเก่าแบบกล่องดำ โดยทั่วไปพื้นฐานของการปรับปรุงแบบกล่องดำนั้นคือการห่อหุ้ม การห่อหุ้มคือการครอบระบบเก่าด้วยชั้นของซอฟต์แวร์ที่ซ่อนความซับซ้อนที่ไม่ต้องการของระบบเก่าออกไป และนำเสนอข้อมูลออกมาให้อยู่ในรูปของส่วนต่อประสานสมัยใหม่ การห่อหุ้มแบบกล่องดำอาจทำได้ทั้งที่ฟังก์ชันการทำงาน ข้อมูล หรือส่วนต่อประสานผู้ใช้งาน ซึ่งจากทฤษฎีที่เกี่ยวข้องหัวข้อ 2.1.2 ได้ถูกจำแนกวิธีการปรับปรุงระบบเก่าออกได้ดังต่อไปนี้ [14]

1) การห่อหุ้มส่วนต่อประสานผู้ใช้

ด้วยวิธีการปรับปรุง โดยทั่วไปกลวิธีที่ถูกใช้คือ กลวิธีสกรีนสเครปปิง ซึ่งจะเป็นการห่อหุ้มส่วนต่อประสานที่เป็นลักษณะข้อความให้อยู่ในรูปของส่วนต่อประสานสมัยใหม่ โดยทั่วไปผู้ค้าโปรแกรมเลียนแบบเครื่องปลายทางของธนาคารจะแนบไลบรารี่ที่ช่วยให้ผู้พัฒนานำไปปรับปรุงเพิ่มเติมได้

2) การห่อหุ้มข้อมูล

การห่อหุ้มข้อมูลจะช่วยให้สามารถเข้าถึงข้อมูลระบบเก่าได้ผ่านทางโปรโตคอลหรือส่วนต่อประสานที่แตกต่างไปจากที่ข้อมูลเดิมให้มา การเปิดช่องทางติดต่อข้อมูลที่ถูกห้อหุ้มสามารถทำได้ 3 วิธี คือ

a. การทำเกตเวย์ฐานข้อมูล ปัจจุบันมีผู้จำหน่ายมากมายได้พัฒนาโปรโตคอลที่ใช้ในการเข้าถึงฐานข้อมูล แต่โดยหลัก ๆ มีผู้จำหน่ายไม่กี่รายที่นำเสนอโปรโตคอลที่ได้มาตรฐานทางอุตสาหกรรม คือ

   i. Open Database Connectivity (ODBC)

   ii. Java Database Connectivity (JDBC)

   iii. Object Data Management Group (ODMG)

   การทำเกตเวย์ฐานข้อมูลโดยทั่วไปจะทำการแปลความการเข้าถึงโปรโตคอลของผู้จัดจำหน่ายเฉพาะให้อยู่ในโปรโตคอลมาตรฐานอันใด

อันหนึ่งเหล่านี้ การแปลความหมายนี้เป็นประโยชน์ต่อแอปพลิเคชันสมัยใหม่ที่จะเข้าถึงข้อมูลจากระบบเก่า

b. การนำมาตรฐานเอ็กเอ็มแอลเข้ามาใช้ร่วมกัน

ภาษาเอ็กซ์เอ็มแอล เป็นภาษาที่ใช้จัดรูปแบบให้อยู่ในรูปของโครงสร้างเอกสาร และเปิดให้ผู้จัดเก็บสามารถออกแบบวิธีการเข้าถึงข้อมูลของตัวเองได้ ด้วยวิธีนี้จะช่วยให้สามารถเรียกใช้ข้อมูลได้ผ่านทางไฟล์เอ็กซ์เอ็มแอล

c. การจำลองฐานข้อมูล

การจำลองฐานข้อมูลเป็นกระบวนการคัดลอก และบำรุงรักษาวัตถุของฐานข้อมูลในหลาย ๆ ฐานข้อมูลที่ถูกสร้างในรูปแบบฐานข้อมูลแบบกระจาย (Distributed Database System) หากมีการเปลี่ยนที่ไซท์หนึ่งก็จะถูกดักจับและจัดเก็บในพื้นที่ก่อนที่จะถูกส่งต่อไปยังแหล่งข้อมูลส่วนกลาง ด้วยวิธีนี้จะช่วยให้ผู้ใช้งานสามารถเข้าถึงข้อมูลได้รวดเร็ว

d. การห่อหุ้มฟังก์ชันของข้อมูล

การห่อหุ้มฟังก์ชันจะช่วยครอบทั้งฟังก์ชันการทำงานและฐานข้อมูลของระบบเก่า การหุ้มฟังก์ชันการทำงานจะช่วยให้สามารถใช้ประโยชน์จากโค้ดภาษาโคบอลที่มีอยู่ได้ ซึ่งสามารถแบ่งกลวิธีการห่อหุ้มฟังก์ชันได้ดังนี้

e. ซีจีไออินทิเกรชัน (CGI Integration)

Common Gatway Interface (CGI) เป็นมาตรฐานของการทำส่วนต่อประสานกับโปรแกรมภายนอกด้วยแม่ข่ายข้อมูล (Data Server) ซึ่งวิธีนี้มีความยืดหยุ่นกว่ากลวิธีสกรีนสเครปปิงเพราะไม่จำเป็นต้องคำนึงถึงส่วนต่อประสานผู้ใช้งานโปรแกรมเลียนแบบเครื่องปลายทาง

f. การห่อหุ้มเชิงวัตถุ (Object-Oriented Wrapping)

กลวิธีนี้มักถูกนำไปใช้กับซอฟต์แวร์ที่มีความซับซ้อน แนวคิดของการห่อหุ้มด้วยวิธีนี้จะมองแอปพลิเคชันเดี่ยว ๆ ออกเป็นวัตถุ การบริการทั่วไปเป็นวัตถุ และกระบวนการทางธุรกิจเป็นวัตถุ ซึ่งจำเป็นต้องมีความเข้าใจโค้ดของระบบเก่าด้วย

g. การห่อหุ้มองค์ประกอบ

กลวิธีนี้มีความคล้ายคลึงกับการห่อหุ้มเชิงวัตถุ แต่องค์ประกอบจะต้องสอดคล้องกับแบบจำลองขององค์ประกอบ ด้วยข้อกำหนดนี้ช่วยให้กรอบงานองค์ประกอบจัดให้องค์ประกอบที่ให้บริการทางทำงานที่มีคุณภาพ

จากวิธีการข้างต้นสามารถแจกแจงข้อดีข้อเสียต่าง ๆ ของแต่ละวิธีออกมาได้ดังตารางที่ 3.1 ซึ่งเมื่อนำมาวิเคราะห์กับความต้องการของทางธนาคารที่ต้องการการปรับปรุง จะเห็นได้ว่าการปรับปรุงครั้งนี้ไม่มีความจำเป็นที่จะต้องปรับปรุงขนาดใหญ่เนื่องจากธนาคารต้องการเน้นในเรื่องของการปรับปรุงงานให้ผู้พิการทางการเห็นสามารถทำงานได้ตามกรอบงานที่กำหนดเอาไว้ และไม่ต้องการให้มีค่าใช้จ่ายสูง และความเสี่ยงน้อยที่สุด

เมื่อพิจารณาในส่วนของพนักงานทั่วไปนั้น พนักงานของบริษัทไม่ต้องการใช้เวลานานในการเรียนรู้ระบบใหม่ที่พัฒนาขึ้น การปรับปรุงจึงควรจะเน้นไปที่การปรับปรุงส่วนต่อประสานผู้ใช้งานให้ผู้พิการทางการเห็นสามารถเข้าถึงได้ด้วยโปรแกรมอ่านหน้าจอ ด้วยเหตุนี้กลวิธีสกรีนสเครปปิงจึงเป็นวิธีที่หน้าสนใจที่สุด เพราะมีค่าใช้จ่ายไม่มาก ไม่ต้องการเทคโนโลยีภายนอกเข้ามาประยุกต์ใช้ อันนำมาซึ่งความเสี่ยงต่อกระบวนการของทางบริษัท

ตารางที่ 3.1 เปรียบกลวิธีของการปรับปรุง

	สิ่งที่ทำการปรับปรุง	เป้าหมาย	จุดแข็ง	จุดด้อย
Screen Scrapping	ส่วนต่อประสานผู้ใช้งานแบบข้อความ	ส่วนต่อประสานผู้ใช้งานสมัยใหม่	• ค่าใช้จ่าย • ระยะเวลา • สนับสนุนอินเตอร์เน็ต	• ความยืดหยุ่น • มีข้อจำกัดการบำรุงรักษา
Database Gateway	โปรโตคอลการเข้าถึงข้อมูลเดิม	โปรโตคอลการเข้าถึงข้อมูลที่ได้มาตรฐาน	• ค่าใช้จ่าย • เครื่องมือสนับสนุน	• มีข้อจำกัดการบำรุงรักษา
XML Integration	โปรโตคอลการเข้าถึงข้อมูลเดิม	เซอร์เวอร์ XML	• ความยืดหยุ่น • เครื่องมือสนับสนุน • B2B	• เครื่องมือสนับสนุน • เทคโนโลยีที่เกี่ยวข้อง
Database	ฐานข้อมูลส่วนกลาง	ฐานข้อมูลแบบ	• ประสิทธิภาพ	• ความข้อง

Replication		กระจาย	• ความมั่นคง	เกี่ยวกับของข้อมูล • การทำงานกับปัญหาที่มีความเฉพาะมากๆ
CGI Integration	ข้อมูลเมนเฟรมหรือ TM Service	หน้าเว็บ HTML	• ค่าใช้จ่าย • สนับสนุนอินเตอร์เน็ต	• ความยืดหยุ่น • การบังคับใช้
OO Wrapping	ทรัพยากรใด ๆ ขององค์กร	แบบจำลองเชิงวัตถุ	• ความยืดหยุ่น	• ค่าใช้จ่าย
Component Wrapping	ทรัพยากรใด ๆ ขององค์กร	แบบจำลององค์ประกอบ	• ความยืดหยุ่น • รวมการให้บริการ	• ค่าใช้จ่าย

3.5 ปรับปรุงระบบเก่า

ขั้นตอนนี้ผู้วิจัยได้ทำงานปรับปรุงระบบเก่าด้วยกลวิธีสกรีนสเครปปิงตามขอบเขตของธนาคารได้ที่กำหนดเอาไว้ เพื่อวิเคราะห์ประโยชน์และข้อผิดพลาดที่เกิดขึ้น โดยรายละเอียดการใช้งานของโปรแกรมที่ถูกพัฒนาสามารถดูได้ใน ภาคผนวก ก และ ข

3.6 วิเคราะห์และออกแบบส่วนต่อประสานโปรแกรมประยุกต์

จากการวิเคราะห์ความต้องการและปัญหาข้างต้นจึงเป็นที่มาของงานวิจัยฉบับนี้ คือการออกแบบส่วนต่อประสานโปรแกรมประยุกต์ใหม่เพื่อที่จะลดข้อกำจัดและปัญหาที่เกิดขึ้นจากการพัฒนาด้วยกลวิธีสกรีนสเครปปิง ซึ่งจะทำให้ผู้พัฒนาโปรแกรมสามารถพัฒนาส่วนต่อประสานโปรแกรมประยุกต์เพื่อผู้พิการทางการเห็นได้โดยไม่ต้องเข้าใจระบบเก่าทั้งหมด และสามารถเรียกดูข้อมูลได้ง่าย ซึ่งวิธีการออกแบบและใช้งานส่วนต่อประสานโปรแกรมประยุกต์นี้จะกล่าวถึงในบทที่ 4

3.7 สร้างเครื่องมือเก็บแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

ในขั้นตอนนี้ จะเป็นการสร้างเครื่องมือเพื่อเก็บข้อมูลหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางให้สามารถเรียกใช้งานได้จากส่วนต่อประสานโปรแกรมประยุกต์ที่ถูกสร้างขึ้น ซึ่งจะช่วยให้

ผู้พัฒนาโปรแกรมไม่จำเป็นต้องเก็บข้อมูลด้วยตัวเอง เพราะสามารถให้ผู้เชี่ยวชาญโปรแกรมเลียนแบบเครื่องปลายทางสามารถเก็บข้อมูลตามขอบเขตงานที่ได้กำหนดเอาไว้ให้แก่ผู้พิการ

3.8 การทดสอบส่วนต่อประสานโปรแกรมประยุกต์และเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

ในขั้นตอนนี้จะเป็นการทำส่วนต่อประสานโปรแกรมประยุกต์ที่พัฒนาขึ้นไปทำการทดสอบโดยเปรียบเทียบส่วนต่อประสานโปรแกรมประยุกต์เดิมที่ถูกพัฒนาขึ้นโดยบริษัทไอบีเอ็ม เพื่อประเมินประสิทธิภาพการทำงานที่เกิดขึ้น โดยแบ่งการทดสอบออกเป็น 2 ประเภทคือ 1) การทดสอบกระบวนการในการเขียนโปรแกรม และ 2)การวัดจำนวนโค้ดที่ใช้ในการพัฒนา

3.9 สรุปผลงานวิจัย

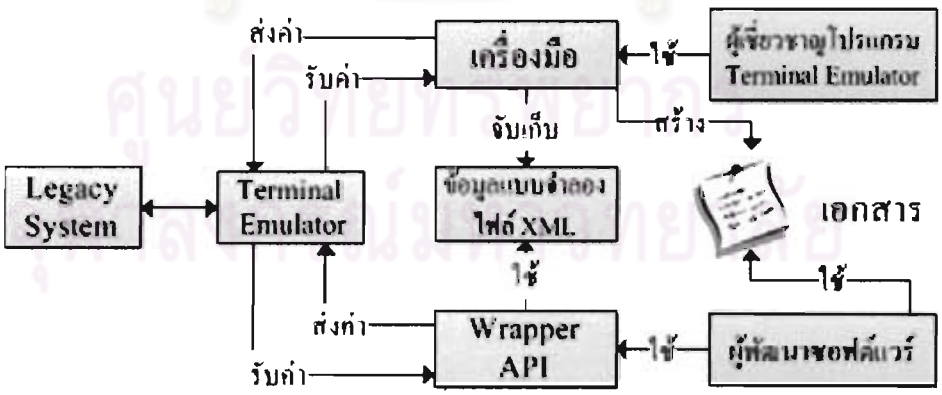
ขั้นตอนนนี้จะเป็นการสรุปผลงานวิจัยที่ได้ทำไป พร้อมทั้งแสดงข้อเสนอแนะและงานที่คาดว่าจะทำต่อไปในอนาคต

บทที่ 4

การออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์

วิทยานิพนธ์ฉบับนี้ได้นำเสนอวิธีการปรุงส่วนต่อประสานผู้ใช้งานสมัยใหม่ เพื่อให้ผู้พิการทางการเห็นสามารถทำงานร่วมกับระบบเก่าได้ด้วยโปรแกรมอ่านหน้าจอ โดยมุ่งเน้นไปที่สองส่วนหลัก ๆ คือ 1) การออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์ที่ใช้ติดต่อกับโปรแกรมเลียนแบบเครื่องปลายทาง และ 2) พัฒนาเครื่องมือสร้างแบบจำลองหน้าจอเพื่อใช้ในการปฏิสัมพันธ์กับผู้ใช้งาน ในส่วนของการออกแบบและพัฒนาส่วนต่อประสานโปรแกรมประยุกต์ที่ใช้ติดต่อกับโปรแกรมเลียนแบบเครื่องปลายทาง เราได้นำหลักการการจัดการข้อมูลแบบจำลองวัตถุเชิงเอกสารเข้ามาประยุกต์ใช้ในการออกแบบ โดยการมองข้อมูลที่แสดงผลบนหน้าจอของระบบเก่าให้อยู่ในรูปแบบของวัตถุชนิดหนึ่ง และยอมให้ผู้พัฒนาสามารถจัดการกับวัตถุเหล่านั้นได้ตามคุณสมบัติที่มีอยู่ในตัววัตถุนั้น ๆ ผ่านทางคลาสและเมธอดที่กำหนดให้หนึ่งคลาส คลาสนั้นจะรับผิดชอบหน้าที่ในการปฏิสัมพันธ์กับระบบเก่าอย่างอัตโนมัติตามข้อมูลที่เก็บเอาไว้จากเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์

ในงานวิจัย เรายังพัฒนาเครื่องมือสร้างแบบจำลองหน้าจอเพื่อใช้ในการปฏิสัมพันธ์กับผู้ใช้งาน โดยเครื่องมือจะสร้างรายงานข้อมูลและช่วยให้ผู้พัฒนาใช้ในการอ้างอิงเพื่อเขียนโปรแกรม โดยในงานวิจัยนี้ดำเนินการตามรูปที่ 4.1



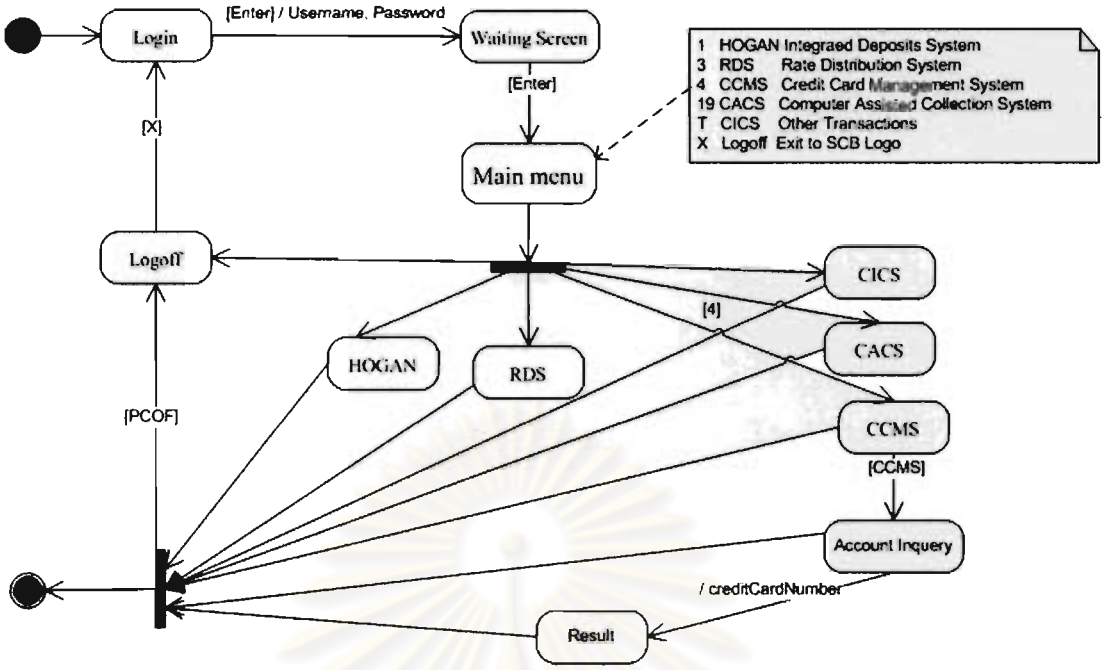
รูปที่ 4.1 ภาพรวมในการทำงานของงานวิจัย

4.1 การวิเคราะห์ข้อมูลหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

การเก็บข้อมูลมีความสำคัญเพื่อให้สามารถรู้ว่าผู้ใช้งานกำลังทำงานอยู่กับระบบไหน และแต่ละระบบมีหน้าจออะไรบ้างที่เกี่ยวข้องในการทำงานและมีการปฏิสัมพันธ์อย่างไร รวมทั้งแสดงข้อผิดพลาดออกมาได้เมื่อเกิดปัญหาขึ้นผ่านการเรียกเมธอดของคลาส ซึ่งทั้งหมดนั้น เราจำเป็นต้องศึกษาขั้นตอนการทำงานของระบบเก่า โดยพื้นฐานการทำงานของระบบเก่าจะเป็นแบบดำเนินขั้นตอนตามลำดับ จากรูปที่ 4.2 แสดงตัวอย่างขั้นตอนการสืบค้นข้อมูลด้านการเงินจากระบบซีซีเอ็มเอสซึ่งเป็นระบบดูแลจัดการข้อมูลเกี่ยวกับบัตรเครดิตของลูกค้าของธนาคารสแตนดาร์ดชาร์เตอร์ดและแสดงผลในโปรแกรมเลียนแบบเครื่องปลายทาง โหนดแต่ละโหนดแทนหน้าจอหนึ่งหน้าจอ ในแต่ละหน้าจอประกอบไปด้วยข้อมูลเข้าและข้อมูลออก

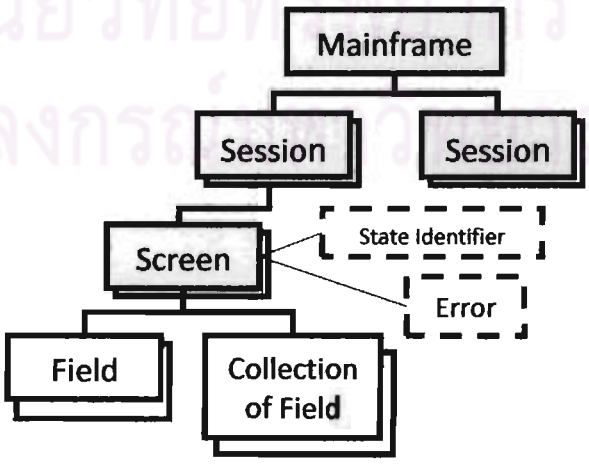
- ณ โหนดเริ่มต้น ระบบแสดงหน้ารอลงชื่อผู้ใช้งานและรหัสผ่าน (Login)
- เมื่อผู้ใช้กรอกเรียบร้อยและกดปุ่ม "Enter" ระบบจะเข้าไปสู่โหนดหน้าพักข้อมูล (Waiting Screen) เพื่อรอให้ผู้ใช้กดปุ่ม Enter อีกครั้งเพื่อเข้าไปยังหน้าเมนูหลัก (Main Menu)
- หลังจากกด "Enter" โหนดเมนูหลักก็จะปรากฏให้เลือก ซึ่งการเข้าใช้งานขึ้นอยู่ข้อมูลเข้าที่ใส่ลงไป โดยมีหมายเลขที่ใส่ได้เป็น 1, 3, 4, 19, T และ X ตามลำดับ การพิมพ์ X มีผลทำให้โปรแกรมกลับไปสู่สถานะหน้าจอลงชื่อเข้าใช้งาน
- ผู้ใช้ พิมพ์เลข "4" เพื่อเลือกหมวดการใช้ระบบการจัดการบัตรเครดิต (Credit Card Management System) ก็จะเข้าสู่โหนดถัดไป จากนั้นจะมีช่องรอคำสั่งว่าจะร้องขอดูข้อมูลในส่วนใดของลูกค้า ซึ่งการร้องขออาจจำเป็นต้องทราบข้อมูลส่วนตัวของลูกค้าเพื่อเรียกดู ในกรณีนี้ พิมพ์คำสั่ง "pciq" เป็นคำสั่งเรียกดูการใช้จ่ายของบัตรเครดิต
- เมื่อเข้าไปแล้วจำเป็นต้องกรอกเลขที่บัตรเครดิต กดปุ่ม Enter อีกครั้ง ข้อมูลการใช้จ่ายเหล่านั้นก็จะแสดงผลออกมา

จะเห็นว่าทุก ๆ โหนดสามารถใช้คำสั่ง "pcof" เพื่อกลับไปอยู่หน้าเมนูหลัก



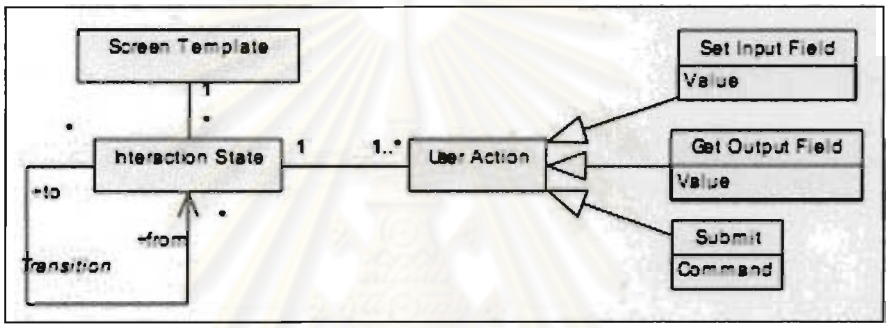
รูปที่ 4.2 การทำงานของระบบซีซีเอ็มเอส

การทำงานในลักษณะนี้เป็นการทำงานของหนึ่งระบบ แต่ในความเป็นจริงแล้ว ผู้ใช้ต้องใช้งานหลายระบบในเวลาเดียวกัน ในหนึ่งเมนเฟรมอาจจะประกอบไปด้วยระบบการทำงานมากกว่าหนึ่งระบบ โดยหนึ่งระบบจะเท่ากับหนึ่งเซสชัน (Session) แต่ละเซสชัน จะประกอบไปด้วยกลุ่มของสกรีน (Screen) และในแต่ละหน้าจอมีข้อมูลต่าง ๆ ที่ใช้ในการทำงาน ถูกแสดงผลออกเป็นข้อความ ข้อความเหล่านี้ถูกเรียกว่าฟิลด์ (Field) ฟิลด์อาจแสดงผลได้เป็นกลุ่มของฟิลด์ (Collection of Fields) หรือเรียกว่า ตาราง (Table) เราสามารถแสดงความสัมพันธ์แบบโครงสร้างต้นไม้ได้ดังรูปที่ 4.3

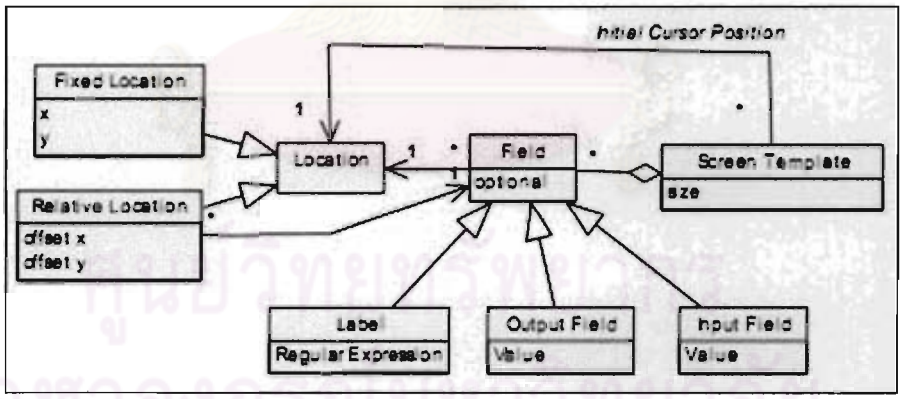


รูปที่ 4.3 แผนภาพโครงสร้างต้นไม้แสดงความสัมพันธ์ของระบบเก่า

เพื่อให้เข้าถึงฟิลด์เหล่านี้ได้ผ่านทางส่วนต่อประสานโปรแกรมประยุกต์ เราต้องเก็บข้อมูลแบบจำลองการปฏิสัมพันธ์ของผู้ใช้งาน G. Canfora [6] ได้นำเสนอแบบจำลองการปฏิสัมพันธ์ผู้ใช้งาน (User Interaction Model) ไว้ในงานวิจัยดังรูปที่ 4.4 ผู้ใช้งานระบบจะเป็นผู้ส่งข้อมูลนำเข้า รับข้อมูลส่งออก และสั่งคำสั่งเพื่อให้หน้าจอเปลี่ยนสถานะ และการที่จะรับรู้สถานะของหน้าจอนั้นได้ เราต้องเก็บข้อมูลที่ใช้ในการระบุตัวตนของแต่ละหน้าจอเหล่านั้น โดยการเก็บข้อมูลได้ถูกแสดงออกมาในรูปแบบคลาสไดอะแกรมของสกรีนเทมเพลต (Screen Template) ซึ่งประกอบไปด้วย Label, Output Field, Input Field ดังรูปที่ 4.5



รูปที่ 4.4 คลาสไดอะแกรมการปฏิสัมพันธ์ของผู้ใช้งาน[6]



รูปที่ 4.5 คลาสไดอะแกรมแผ่นแบบของหน้าจอ[6]

แต่ในการออกแบบส่วนต่อประสานโปรแกรมยุกต์นั้น เราต้องการประเภทของข้อมูลที่ละเอียดกว่า ข้อมูลส่งออกอาจเป็นฟิลด์เดี่ยวหรือตารางก็ได้ ซึ่งทำให้การคืนค่าแก่ผู้พัฒนามีประเภทที่แตกต่างกัน และการทำงานของระบบเก่า อาจมีได้มากกว่าหนึ่งเซสชันพร้อม ๆ กัน ดังนั้นเราต้องเก็บข้อมูลของแต่ละเซสชัน เพื่อที่จะจัดการเซสชันให้ผู้พัฒนาแบบอัตโนมัติ และเราไม่สามารถที่จะระบุหน้าที่ (Task) ได้แบบจำเพาะเจาะจง การเก็บเส้นทางเป้าหมายไปยังแต่ละ

หน้าจอจะต้องการอ้างอิงให้ผ่านไปได้ทุก ๆ หน้าจอที่ครอบคลุมการทำงานของผู้พิการทางการเห็น ด้วยเหตุนี้ทำให้เราได้แบ่งการเก็บข้อมูลออกเป็น 3 ส่วนดังต่อไปนี้

4.1.1 **ข้อมูลประเภทเซสชัน** (Session Data)

ข้อมูลประเภทนี้จะถูกเก็บเพื่อใช้ในการติดต่อกับระบบต่าง ๆ ที่ต้องการใช้งาน โดยข้อมูลที่เก็บจะเป็น ชื่อของเซสชัน ไอพีและพอทที่ใช้ในการเชื่อมต่อ และค่าปรับแต่งต่าง ๆ ซึ่งเป็นคุณสมบัติพื้นฐานของการแสดงผลหน้าจอของระบบนั้น ๆ เช่น ขนาดของหน้าจอ เป็นต้น

4.1.2 **ข้อมูลประเภทสกรีน** (Screen Data)

ข้อมูลประเภทสกรีนเป็นข้อมูลเฉพาะที่ใช้ในการระบุตัวตนของหน้าจอนั้น ๆ การเก็บเพียงแค่ป้ายชื่อซึ่งเป็นชื่อของฟิลด์ อาจทำให้ข้อมูลเกิดการซ้ำกันได้ ดังนั้นการเก็บข้อมูลเฉพาะที่ใช้ในการระบุตัวตนอาจมีได้มากกว่าหนึ่งตัวแปร ข้อมูลเหล่านี้เป็นได้ทั้ง ข้อความเฉพาะจากฟิลด์ในหนึ่งหน้าจอ ตำแหน่งของเคอร์เซอร์เริ่มต้นเมื่อผู้ใช้เข้าสู่หน้าจอดังกล่าว หรือจำนวนฟิลด์ทั้งหมดหรือคุณลักษณะเฉพาะของฟิลด์ตำแหน่งใด ๆ ที่ปรากฏในหน้าจอ เป็นต้น ซึ่งสามารถสรุปออกมาได้ดังตารางที่

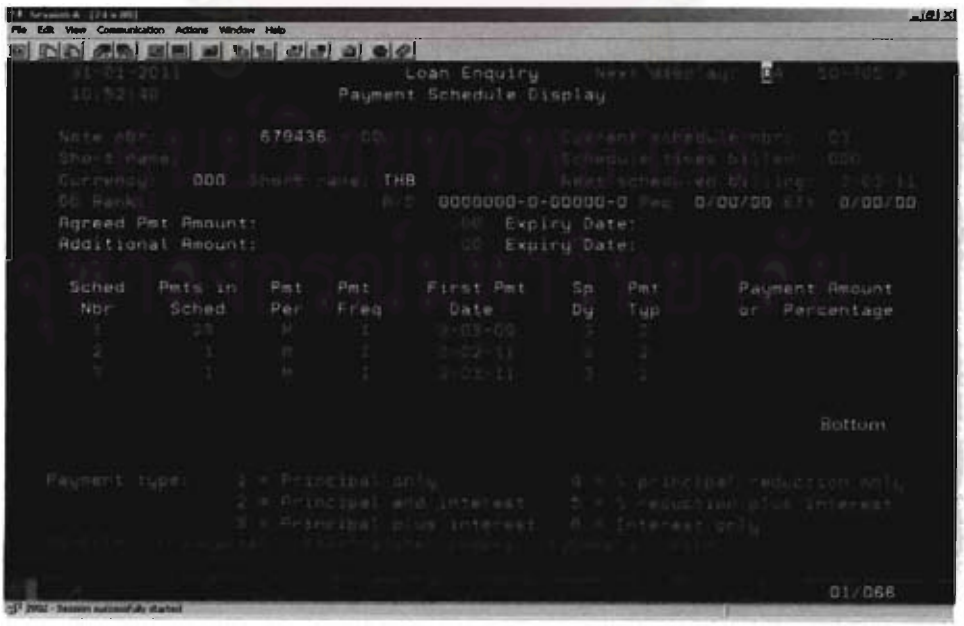
ในการเข้าถึงหน้าจอในแต่ละหน้าจอนั้น ยังต้องคำนึงถึงข้อจำกัดและกฎของการยินยอมให้เข้าถึงอีกด้วย ผู้ใช้งานระบบเก่าจะได้รับสิทธิในการยินยอมให้เข้าใช้งานหน้าจอต่าง ๆ ที่แตกต่างกันเมื่อผู้ใช้เข้าถึงหน้าจอที่ถูกห้ามไว้หรือมีทางเลือก (Option) เสริมก่อนเข้าไปทำงานในหน้านั้นๆ การกำหนดสกรีนเทมเพลตเราจะมองหน้าเหล่านี้เป็นหนึ่งสกรีนเทมเพลตแยกต่างหากและดำเนินการให้โดยอัตโนมัติ

นอกเหนือจากนั้น ในการทำงานของระบบเก่าจะมีส่วนของการตรวจสอบข้อมูลเข้าของผู้ใช้งาน ข้อมูลเหล่านี้จะต้องได้รับการตรวจสอบก่อนที่จะเข้าใช้งานในหน้าจอถัดไปหรือสืบค้นข้อมูลออกมาให้ผู้ใช้งาน เมื่อเกิดข้อผิดพลาด เช่น กรอกรหัสผ่านไม่ถูกต้อง หรือข้อมูลที่ต้องการค้นหาไม่มีอยู่ในระบบ ระบบจะต้องแจ้งเตือนให้ผู้ใช้งานทราบว่าจะต้องทำขั้นตอนใดต่อไป การแสดงผลข้อผิดพลาดเหล่านี้ในระบบเก่าจะถูกแสดงเป็นข้อความทางด้านล่างของหน้าจอภายใต้เงื่อนไขที่กำหนดไว้ ข้อมูลเหล่านี้จะถูกจัดเก็บอยู่ในรูปของรหัสข้อผิดพลาดและข้อความแสดงผลที่เกิดขึ้นบนหน้าจอ

4.1.3 ข้อมูลประเภทฟิลด์ (Field Data)

ข้อมูลบนหน้าจอระบบเก่าออกได้เป็น 2 ประเภทใหญ่ ๆ คือ 1) ข้อมูลนำเข้าและ 2) ข้อมูลส่งออก ข้อมูลนำเข้าเป็นข้อมูลที่ส่งเข้ามาให้กับระบบเก่าเพื่อใช้ในการทำงาน เรายังได้แบ่งข้อมูลนำเข้าออกเป็น 2 ประเภท คือ 1) ข้อมูลเข้าจากผู้ใช้งาน (User Input) และ 2) ข้อมูลเข้าที่เป็นคำสั่งระบบ (System Input) ข้อมูลเข้าจากผู้ใช้งานจะเปลี่ยนแปลงไปตามการใช้งานของผู้ใช้ ส่วนข้อมูลเข้าที่เป็นคำสั่งระบบจะไม่มีการเปลี่ยนแปลง หรือมีการเปลี่ยนแปลงน้อยครั้งมาก เนื่องจากมันถูกกำหนดเอาไว้ภายในระบบเก่า มีไว้เพื่อใช้ในการเข้าถึงข้อมูลหรือเมนูต่าง ๆ

ส่วนข้อมูลส่งออกเป็นข้อมูลที่ใช้ในการแสดงผลลัพธ์ให้กับผู้ใช้งาน ข้อมูลส่งออกสามารถแบ่งออกได้เป็น 2 ประเภท คือ 1) ข้อมูลส่งออกแบบเดี่ยว และ 2) ข้อมูลส่งออกแบบกลุ่ม ข้อมูลส่งออกแบบเดี่ยวเป็นข้อมูลที่มีค่าเฉพาะตัว เช่น ฟิลด์ชื่อ ที่อยู่ ข้อความแสดงชื่อหน้าจอ ข้อความแสดงข้อผิดพลาดของระบบ ข้อมูลส่งออกแบบกลุ่มคือข้อมูลที่มีลักษณะเป็นตาราง ข้อมูลแบบตารางอาจมีได้มากกว่าหนึ่งหน้า โดยจะแสดงป้ายชื่อด้านท้ายตารางเพื่อบอกถึงการสิ้นสุดของตาราง ในแต่ละตารางอาจมีช่องข้อมูลนำเข้าให้ผู้ใช้งานใส่เพื่อดูข้อมูลเฉพาะของแต่ละแถว จากรูปที่ 4.6 แสดงหน้าจอระบบเก่าที่มีข้อมูลส่งออกแบบตาราง จะเห็นว่าหน้าจอนี้อาจเกิดขึ้นซ้ำกันได้ โดยมีการเปลี่ยนแปลงเพียงแต่ส่วนของตาราง ซึ่งจุดสิ้นสุดของตารางถูกกำกับด้วยป้ายชื่อด้านท้ายตารางเป็นคำว่า bottom

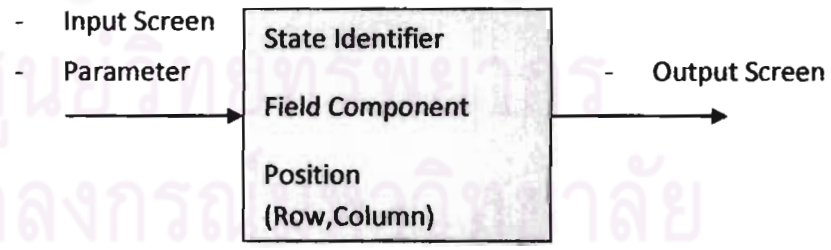


รูปที่ 4.6 หน้าจอโปรแกรมเลียนแบบเครื่องปลายทางที่แสดงผลตาราง

การเก็บข้อมูลส่งออกแบบกลุ่มจะแตกต่างจากการเก็บข้อมูลนำเข้าและข้อมูลส่งออกแบบเดี่ยว เนื่องจากข้อมูลประเภทตาราง เนื่องจากการแสดงผลหนึ่งตารางอาจมีได้หลายหน้าในหนึ่งสกรีนเทมเพลต และข้อมูลหนึ่งชนิดถูกแบ่งออกตามคอลัมน์ทำให้มีหลายค่า เราต้องเก็บข้อมูลที่สามารถให้ผู้พัฒนาเรียกดูข้อมูลได้แบบทั้งตาราง หรือเรียกเพียงบางคอลัมน์หรือแถว หรืออาจเรียกเพียงค่าเดียวจากตารางนั้นได้ ดังนั้นเราจึงเก็บค่าตำแหน่งแถวและคอลัมน์ในแต่ละคอลัมน์ของตาราง เก็บจำนวนแถวที่มากที่สุดที่สามารถแสดงผลได้ในหนึ่งหน้าจอ ชื่อฟิลด์ที่แสดงการสิ้นสุดของตาราง แต่ปัญหาอย่างหนึ่งที่เราพบในการเก็บข้อมูลของตารางก็คือ บางครั้งตารางอาจจะแสดงผลข้อมูลจำนวนมหาศาลมากเกินกว่าจะดึงข้อมูลออกมาได้หมด ด้วยเหตุนี้เราจึงต้องให้ผู้เก็บข้อมูลกำหนดจำนวนหน้ามากสุดที่จะเก็บข้อมูลมาแสดงผลได้ในครั้งหนึ่ง ๆ ด้วย ตามความเหมาะสมของประเภทของข้อมูล เช่น การค้นหาด้วยชื่อลูกค้า หรือเลขบัตรประจำตัวประชน เป็นข้อมูลที่มีความเฉพาะตัวสามารถหาได้ในหน้าแรก ๆ ของตาราง จำนวนหน้ามากสุดก็ไม่จำเป็นต้องมีมากกว่าสิบหน้า เป็นต้น

4.1.4 **ข้อมูลนำร่องไปยังแต่ละหน้าจอ** (Routing Path)

การตามรอยไปยังหน้าจอต่าง ๆ จะเกิดขึ้นได้ต้องประกอบไปด้วยข้อมูลประเภทต่าง ๆ ใน 3 ประเภทข้างต้นที่กล่าวมา การเก็บข้อมูลทั้งสามส่วนนี้ จะมีการเก็บตำแหน่ง (Position) ในลักษณะแถวและคอลัมน์ที่ต้องส่งข้อมูลหรือแสดงข้อมูลในหน้าจอนั้นด้วย (รูปที่ 4.7)



รูปที่ 4.7 ส่วนประกอบของการระบุหน้าจอ

เมื่อแต่ละโหนดทราบถึงสถานะของตัวเอง ทราบว่าต้องมีข้อมูลเข้าอะไร ส่วนประกอบของข้อมูลออกที่ใช้ และตำแหน่งของข้อมูลนั้น ๆ การตามรอยย้อนกลับก็จะสามารถทำได้ หากโหนดเหล่านั้นเคยผ่านการใช้งานและกำหนดคุณลักษณะทั้งสามส่วนนี้มาแล้ว เช่น โหนด

แสดงข้อมูลบัตรเครดิต (Result page) จะทราบว่าหน้าจอก่อนหน้าที่สามารถมาถึงได้มีหน้าจออะไรบ้าง และต้องการพารามิเตอร์อะไรในการเข้าถึง จากรูปที่ 7 แต่ละโหนดจะมีการเก็บข้อมูลตามตารางที่ 2 โหนดลงชื่อเข้าใช้งาน ซึ่งเป็นสถานะเริ่มต้น เริ่มแรกจะมีข้อมูลเข้าอยู่สองอันคือ ช่องกรอกชื่อผู้ใช้งานและช่องกรอกรหัสผ่าน หลังจากทำการกรอกรหัสผ่านเข้าไปยังโหนดอื่น ๆ ซึ่งได้รับการกำหนดสถานะแล้ว โหนดลงชื่อเข้าใช้งานก็จะมีส่วนประกอบเพิ่มเข้ามานอกเหนือไปจากข้อมูลเข้าที่มีอยู่ ซึ่งในที่นี้คือหน้าเมนูหลัก และเมื่อมีการเข้าใช้งานไปจนถึงโหนดแสดงข้อมูลบัตรเครดิต สถานะการเข้าถึงในหน้าจอถัดไปของแต่ละโหนดก็จะเป็นหน้าจอที่ได้ใช้งานและเก็บข้อมูลแล้ว

4.2 ส่วนต่อประสานโปรแกรมประยุกต์ (Application Programming Interface)

จากการเก็บข้อมูลข้างต้นและพิจารณาจากแผนภาพโครงสร้างต้นไม้ ภาพที่ 4.4 เราสามารถแตกประเภทของวัตถุออกได้เป็น 3 ประเภท คือ เซสชัน สกรีน ฟิลด์ แต่ความจริงแล้วเราสามารถมองฟิลด์แบบกลุ่มหรือที่เรียกว่า ตาราง แยกออกมาได้อีกหนึ่งประเภท เนื่องจากมีคุณสมบัติในการเก็บข้อมูลและเรียกใช้งานที่แตกต่างกัน

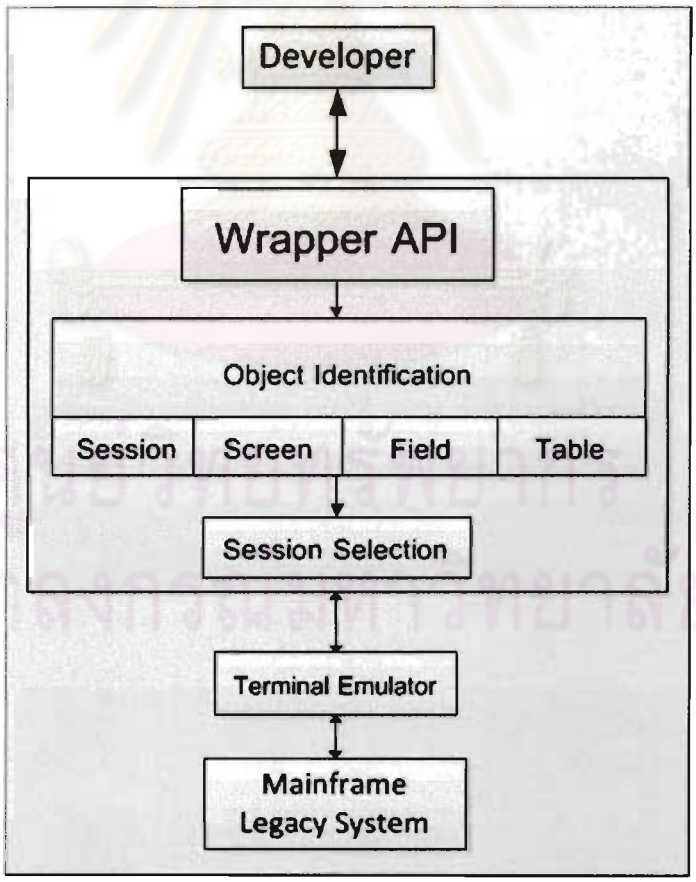
เราออกแบบคลาสที่ชื่อว่าแร็บพเปอร์ให้ผู้พัฒนาสามารถจัดการกับข้อมูลของระบบเก่าได้ภายในคลาสเพียงคลาสเดียว โดยมีเมธอดต่าง ๆ ดังตารางที่ 1

จากตารางที่ 4.1 เราจะเห็นว่า มีเมธอดที่ให้ผู้พัฒนาสามารถกำหนดค่าให้แก่ระบบเก่าได้อยู่หนึ่งเมธอด และเมธอดที่เหลือให้ผู้พัฒนารับค่าได้ ประเภทของค่าที่รับขึ้นอยู่กับเมธอดและวัถตุต่าง ๆ ที่เรียกใช้ มีพารามิเตอร์สองชนิดที่ส่งผ่านเมธอดแต่ละเมธอดเข้าไปคือซีเล็คเตอร์ (Selector) และวาลู (Value) พารามิเตอร์ทั้งสองอันนี้ถูกส่งไปเป็นประเภทของสตริง (String) โดยซีเล็คเตอร์จะเป็นวัตถุที่เราต้องการกำหนดหรือร้องขอค่ามาซึ่งมีอยู่สี่ประเภท ในขณะที่วาลู จะเป็นชุดสตริงที่ผู้พัฒนาส่งเข้าไปเพื่อร้องขอดูข้อมูลที่ต้องการ โดยค่าที่ถูกส่งคืนกลับจะอยู่ในรูปของสตริงหรืออาเรย์ของออฟเจคสตริง เราสามารถแสดงภาพรวมของการทำงานได้ดังรูปที่ 4.8

ตารางที่ 4.1 เมธอดและคุณสมบัติของคลาสแร็บพเปอร์

ชื่อเมธอด	คุณสมบัติ
initial(String selector) :Boolean	เริ่มต้นการทำงานให้ระบบที่กำหนด
initialAll():Boolean	เริ่มต้นการทำงานให้กับระบบทั้งหมด

destroy(String selector) :Boolean	ปิดการทำงานให้ระบบที่กำหนด
destroyAll():Boolean	ปิดการทำงานระบบทั้งหมด
setValue(String selector,String value) :Boolean	เซ็ตค่าให้กับวัตถุที่เลือก
getValue(String selector, String value):String	ขอค่าจากวัตถุที่เลือก
getValueList(String selector, String value):HashMap	ขอกลุ่มของค่าจากวัตถุที่เลือก
getAllValue():HashMap	ขอค่าทั้งหมดจากวัตถุที่เลือก
getArrtibute(String selector, String value):String	ขอแอททริบิวส์จากวัตถุที่เลือก
getArrtibuteList(String selector, String value):HashMap	ขอกลุ่มของแอททริบิวส์จากวัตถุที่เลือก
getAllArrtibute():HashMap	ขอแอททริบิวส์ทั้งหมดจากวัตถุที่เลือก



รูปที่ 4.8 ภาพรวมการทำงานของส่วนต่อประสานโปรแกรมประยุกต์

เมื่อมีการเรียกการใช้งานเมธอดเกิดขึ้น ส่วนต่อประสานโปรแกรมประยุกต์จะทำหน้าที่ตรวจสอบประเภทของวัตถุ และทำการตัดคำออกจากสตริงข้อความที่มากับวัตถุนั้น รูปแบบของการตัดคำขึ้นอยู่กับวัตถุและเมธอดที่เรียกใช้ เราใช้การแบ่งลำดับชั้นของวัตถุด้วยสัญลักษณ์หน้าคำต่าง ๆ ดังนี้

ตารางที่ 4.2 สัญลักษณ์ในการตัดข้อความ

สัญลักษณ์	ความหมาย
# หน้าตัวแปร	แทนตัวแปรชื่อฟิลด์ที่แสดงผลอยู่บนหน้าจอระบบเก่า
[ ]ครอบข้อความ	แทนข้อมูลของวัตถุที่อยู่ในลำดับชั้นที่ต่ำกว่า
ไม่กำหนดสัญลักษณ์หน้าตัวแปร	แทนตัวแปรอะทิบิวที่ใช้เก็บคุณสมบัติของระบบ

เพราะฉะนั้นเราสามารถเรียกใช้งานเมธอดโดยส่งค่าผ่านทางเมธอดได้ดังตัวอย่างรูปที่ 4.9

```
try{
    Wrapper.initailAll();
String name = Wrapper.getValue("field",
"#ccms[#Name]");
}catch(IdentificationException ex){
System.out.println(ex.getDescription();)
}
```

รูปที่ 4.9 โค้ดการเริ่มต้น Session และ รับค่าของ Field ชื่อว่า Name

เมธอดชื่อ initialAll ใช้เพื่อเริ่มต้นระบบเก่าทั้งหมด จากนั้นเราเรียกเมธอดชื่อ getValue จากคลาสแร็ปเปอร์ ที่เป็นสแตติค (Static) พร้อมส่งพารามิเตอร์รูปแบบสตริงเข้าไปสองค่า คือ "field" และ "#Name" พารามิเตอร์ตัวแรกใช้ในการระบุวัตถุที่ต้องการเรียกใช้งาน ส่วนพารามิเตอร์ตัวที่สองคือ ค่าที่เราร้องขอไปขึ้นอยู่กับเมธอดที่เรียกใช้ ในตัวอย่างข้างต้นนี้ เราร้องขอดูข้อมูลจากฟิลด์ที่มีชื่อว่า Name จากหน้าจอปัจจุบันที่ทำงานอยู่ เครื่องหมาย # ตรงหน้า Name แทน

ตัวแปรที่เป็นชื่อฟิลด์ที่เรากำหนดไว้ในไฟล์เอ็กซ์เอ็มแอล เราไม่สามารถเรียกดูค่า Name จากหน้าอื่นได้เนื่องจากฟิลด์มีระดับโครงสร้างต้นไม้ที่ต่ำกว่าวัตถุ Screen และเช่นเดียวกับที่วัตถุ Screen ไม่สามารถร้องขอดูข้อมูลจากระบบอื่นซึ่งเป็นวัตถุประเภท Session ได้ ดังนั้นการไปยังหน้าจอต่าง ๆ จะทำผ่านโดยซีเล็คเตอร์ของวัตถุ Session ที่อยู่สูงกว่าและใช้เมธอดชื่อ setValue เพื่อไปยังหน้าจอที่ต้องการก่อนเรียกดูข้อมูล ดังตัวอย่างรูปที่ 4.10

```
Wrapper.setValue("Session","#ccms[targetScreen=#creditCard,
                                    inputScreen=#login[#username =
                                    punoii,#password = mes123]]");
```

รูปที่ 4.10 โค้ดแสดงการส่งคำสั่งเพื่อไปหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางหน้าถัดไป

เรากำหนดให้วัตถุ Session ที่มีระบบชื่อ ccms ไปยังหน้าจอที่ชื่อ credit card โดยมีหน้าจอที่เราต้องระบุข้อมูลที่จำเป็นเพื่อผ่านไปยังหน้าจอดังกล่าว ซึ่งก็คือ username และ password ที่เป็นฟิลด์ของหน้าจอ login ส่วน targetScreen และ inputScreen เป็นแอททริบิวส์ของวัตถุ Session ไม่ใช่ชื่อของฟิลด์ จึงไม่ต้องใส่ # เอาไว้ด้านหน้า และเราใส่วงเล็บเพื่อเป็นการระบุค่าของแอททริบิวส์หรือโหนดลูกที่อยู่ระดับล่างกว่า ในที่นี้ก็คือ targetScreen, inputScreen สำหรับแอททริบิวส์ของวัตถุ Session และ username, password สำหรับ ฟิลด์ของหน้าจอ login โดย targetScreen คือหน้าจอเป้าหมายที่ผู้พัฒนาต้องการที่จะไปถึงเพื่อดึงข้อมูล และ inputScreen คือหน้าจอที่ต้องกำหนดข้อมูลเข้าที่สำคัญก่อนจะผ่านไปถึงหน้าจอเป้าหมาย

inputScreen อาจมีมากกว่าหนึ่งค่าหน้าจอก็ได้ ซึ่งชื่อของหน้าจอเหล่านี้จะถูกระบุเอาไว้อยู่ในรายการที่สร้างขึ้นจากเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์

บทที่ 5

การพัฒนาเครื่องมือสร้างแบบจำลอง

การปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

เครื่องสร้างแบบจำลองการปฏิสัมพันธ์กับหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง ถูกสร้างขึ้นเพื่อช่วยให้ผู้พัฒนาไม่จำเป็นต้องเสียเวลาในการศึกษาระบบเก่าด้วยตัวเอง โดยให้การเก็บข้อมูลเป็นหน้าที่ของผู้เชี่ยวชาญ ซึ่งเป็นพนักงานขององค์กรเหล่านั้นโดยตรง

ในบทนี้จะกล่าวถึงโครงสร้างของเครื่องมือ และขั้นตอนการทำงานของเครื่องมือดังรายละเอียดต่อไปนี้

5.1 สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือ

5.1.1 สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือด้านฮาร์ดแวร์

เครื่องคอมพิวเตอร์พกพา (Lab Top) 1 เครื่อง

- หน่วยประมวลผล Intel Core 2 Duo ความเร็ว 1.7 กิกะเฮิรตซ์ (GHZ)
- หน่วยความจำหลัก DDR2 ขนาด 4 กิกะไบต์ (GB)
- ฮาร์ดดิสค์ความเร็ว 5,400 รอบ/วินาที ขนาด 160 กิกะไบต์ (GB)

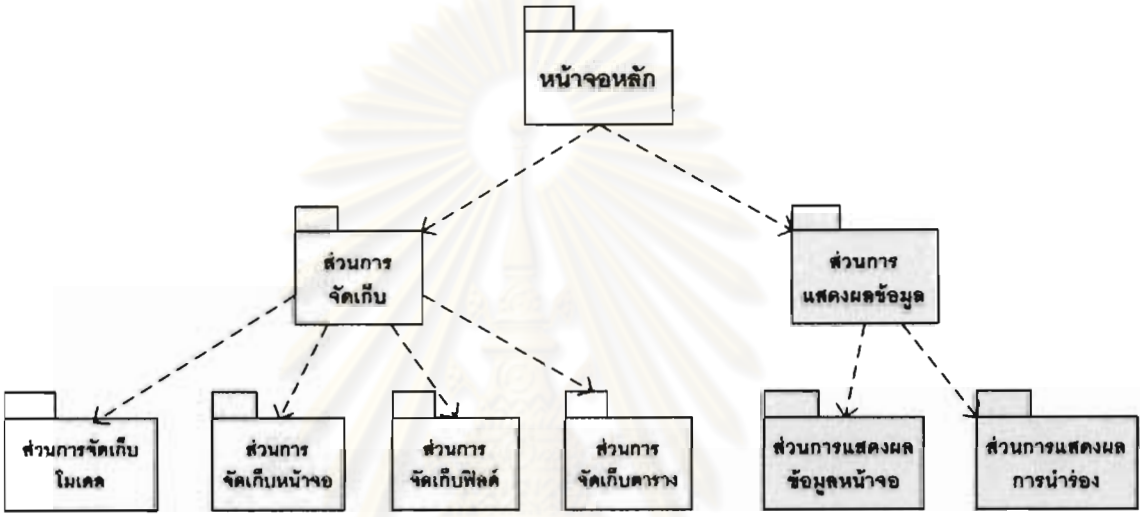
5.1.2 สภาพแวดล้อมที่ใช้ในการพัฒนาเครื่องมือด้านซอฟต์แวร์

- ระบบปฏิบัติการวินโดว์ 7 โปรเฟสชันนัล (Microsoft Window 7)
- ไมโครซอฟท์วิสชวลสตูดิโอ ซีชาร์ปดอทเนต 2008 (Microsoft Visual Studio C#.NET 2008) สำหรับการพัฒนาส่วนต่อประสานผู้ใช้งานสำหรับผู้พิการทางการเห็น
- ไมโครซอฟท์ดอทเน็ตเฟรมเวิร์ก (Microsoft .NET Framework) รุ่น 3.5 ขึ้นไป เพื่อใช้สำหรับการทำงานของวิสชวลสตูดิโอ และการทำงาน (Run) ของเครื่องมือต้นแบบ
- เน็ตบีน (Net Bean IDE) สำหรับพัฒนาเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

- ไมโครซอฟท์แอกเซส 2007 (Microsoft Access 2007) สำหรับโปรแกรมและจัดการข้อมูลในฐานข้อมูล

5.2 โครงสร้างของเครื่องมือ

โครงสร้างของเครื่องสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอ อธิบายได้ด้วยแผนภาพแพคเกจซึ่งแสดงความสัมพันธ์ระหว่างส่วนประกอบต่างๆ แสดงดังรูปที่



รูปที่ 5.1 แผนภาพแพกเกจส่วนประกอบของเครื่องมือ

5.3 ขั้นตอนการทำงานของเครื่องมือ

ให้พิจารณาจากรูปที่ 4 แสดงแผนภาพการทำงานของเครื่องมือสร้างแบบจำลอง เมื่อเริ่มต้นการเก็บข้อมูล ผู้ใช้งานจะต้องกำหนดเซสชันที่ต้องทำการเชื่อมต่อ พร้อมกำหนดค่าพื้นฐานที่ต้องการให้ระบบเก่าทำงานหลังมีการเรียกผ่านทางส่วนต่อประสานโปรแกรมประยุกต์ที่ถูกสร้างขึ้น ในการเก็บข้อมูลฟิลด์แต่ละฟิลด์ ผู้ใช้งานจะต้องเลือกประเภทของข้อมูลที่ต้องการจะเก็บว่าเป็น ข้อมูลนำเข้า ข้อมูลส่งออก ข้อผิดพลาด หรือข้อมูลที่ใช้ในการระบุหน้าจอ โดยระบบจะช่วยดึงข้อมูลพื้นฐานที่จำเป็นต้องใช้ในการเก็บข้อมูล เช่น ชื่อฟิลด์ ตำแหน่งของฟิลด์ที่เคอร์เซอร์อยู่ให้ เมื่อเสร็จสิ้นการเก็บข้อมูล ระบบจะจัดเก็บไฟล์ให้อยู่ในรูปของไฟล์เอ็กซ์เอ็มแอลและเอ็กซ์พอร์ทรายงานของพารามิเตอร์ที่จำเป็นในแต่ละหน้าจอสำหรับผู้พัฒนาในการปฏิสัมพันธ์กับระบบเก่า เราสามารถสรุปการเก็บข้อมูลออกมาเป็นไฟล์เอ็กซ์เอ็มแอลได้ 3 ไฟล์ คือ Session.xml Screen.xml และ Routing.xml ซึ่งสามารถดูได้รูปที่ 5.2,5.3 และ 5.4

Screen.xml เป็นไฟล์ที่เก็บข้อมูลของสกรีนเทมเพลตเอาไว้ทั้งหมด เช่น ฟิลด์ข้อมูลนำเข้า ฟิลด์ข้อมูลส่งออก ข้อผิดพลาด ข้อมูลที่ใช้ระบุตัวตนของหน้าจอ ในขณะที่ Routing.xml จะทำการเก็บข้อมูลในการรู้จำเส้นทางที่จะไปยังหน้าจอต่าง ๆ ของเซสชันหนึ่ง โดย Session.xml จะทำการเก็บข้อมูลที่เกี่ยวข้องกับการเชื่อมต่อระบบเก่าในแต่ละเซสชัน และจะเชื่อมโยงไปยังไฟล์ Screen.xml และ Routing.xml ของเซสชันนั้น ๆ เมื่อเมธอดถูกเรียกใช้ เอพีไอ (API) จะเชื่อมโยงและรู้จำหน้าจอและเส้นทางที่จะไปถึงหน้าจอต่าง ๆ ให้อัตโนมัติ

Model.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
 <xs:element name="models">
  <xs:complexType>
   <xs:sequence>
    <xs:element maxOccurs="unbounded" ref="model"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="model">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="name"/>
    <xs:element ref="host"/>
    <xs:element ref="type"/>
    <xs:element ref="screen"/>
    <xs:element ref="routing"/>
    .....
   </xs:sequence>
   <xs:attribute name="id" use="required" type="xs:integer"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="name" type="xs:NCName"/>
 <xs:element name="host" type="xs:NMTOKEN"/>
 <xs:element name="type" type="xs:integer"/>
 <xs:element name="screen" type="xs:NCName"/>
 <xs:element name="routing" type="xs:NCName"/>
 .....
</xs:schema>
```

รูปที่ 5.2 ไฟล์เอ็กซ์เอ็มแอลสคีมาของการเก็บข้อมูลประเภทเซสชัน

Screen.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
 <xs:element name="screens">
  <xs:complexType>
   <xs:sequence>
    <xs:element maxOccurs="unbounded" ref="screen"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="screen">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="name"/>
    <xs:element ref="identification"/>
    <xs:element ref="properties"/>
    <xs:element ref="errors"/>
   </xs:sequence>
   <xs:attribute name="id" use="required" type="xs:integer"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="identification">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="label"/>
    <xs:element ref="total_field"/>
    <xs:element ref="OIA"/>
    <xs:element ref="cursor_position"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="label">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="description"/>
    <xs:element ref="position"/>
    <xs:element ref="position_x"/>
    <xs:element ref="position_y"/>
    <xs:element ref="length"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="total_field" type="xs:string"/>
 <xs:element name="OIA" type="xs:string"/>
 <xs:element name="cursor_position" type="xs:string"/>
 <xs:element name="properties">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="fields"/>
    <xs:element ref="tables"/>
```

รูปที่ 5.3 ไฟล์เอ็กซ์เอ็มแอลสคีมมาของการเก็บข้อมูลประเภทสกรีน

```
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="fields">
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="field"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="field">
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="name"/>
     <xs:element ref="description"/>
     <xs:element ref="position"/>
     <xs:element ref="position_x"/>
     <xs:element ref="position_y"/>
     <xs:element ref="length"/>
     <xs:element ref="value"/>
     <xs:element ref="type"/>
     <xs:element ref="attributes"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="value">
   <xs:complexType mixed="true">
    <xs:attribute name="type" use="required" type="xs:NCName"/>
   </xs:complexType>
  </xs:element>
  <xs:element name="type" type="xs:NCName"/>
  <xs:element name="attributes">
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="protected"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="protected" type="xs:string"/>
  <xs:element name="tables">
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="table"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="table">
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="name"/>
     <xs:element ref="maxrow"/>
```

รูปที่ 5.4 ไฟล์เอ็กซ์เอ็มแอลสคีมมาของการเก็บข้อมูลประเภทสกรีน

```
      <xs:element minOccurs="0" ref="maxpage"/>
      <xs:element ref="previous_command"/>
      <xs:element ref="next_command"/>
      <xs:element ref="bottom_message"/>
      <xs:element ref="columns"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="maxrow" type="xs:string"/>
<xs:element name="maxpage" type="xs:integer"/>
<xs:element name="previous_command" type="xs:string"/>
<xs:element name="next_command" type="xs:string"/>
<xs:element name="bottom_message">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="message"/>
   <xs:element ref="position_x"/>
   <xs:element ref="position_y"/>
   <xs:element ref="length"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="message" type="xs:string"/>
<xs:element name="columns">
 <xs:complexType>
  <xs:sequence>
   <xs:element maxOccurs="unbounded" ref="column"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="column">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="name"/>
   <xs:element ref="position_x"/>
   <xs:element ref="position_y"/>
   <xs:element ref="length"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="errors">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="error"/>
  </xs:sequence>
  <xs:attribute name="length" type="xs:integer"/>
  <xs:attribute name="x" type="xs:integer"/>
  <xs:attribute name="y" type="xs:integer"/>
 </xs:complexType>
</xs:element>
<xs:element name="error">
```

รูปที่ 5.5 ไฟล์เอ็กซ์เอ็มแอลสคีมมาของการเก็บข้อมูลประเภทสกรีน

```
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="description"/>
    <xs:sequence minOccurs="0">
     <xs:element ref="position_x"/>
     <xs:element ref="position_y"/>
     <xs:element ref="length"/>
    </xs:sequence>
   </xs:sequence>
   <xs:attribute name="id" use="required" type="xs:integer"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="name" type="xs:string"/>
 <xs:element name="description" type="xs:string"/>
 <xs:element name="position" type="xs:string"/>
 <xs:element name="position_x" type="xs:string"/>
 <xs:element name="position_y" type="xs:string"/>
 <xs:element name="length" type="xs:string"/>
</xs:schema>
```

รูปที่ 5.6 ไฟล์เอ็กซ์เอ็มแอลสคีมมาของการเก็บข้อมูลประเภทสกรีน

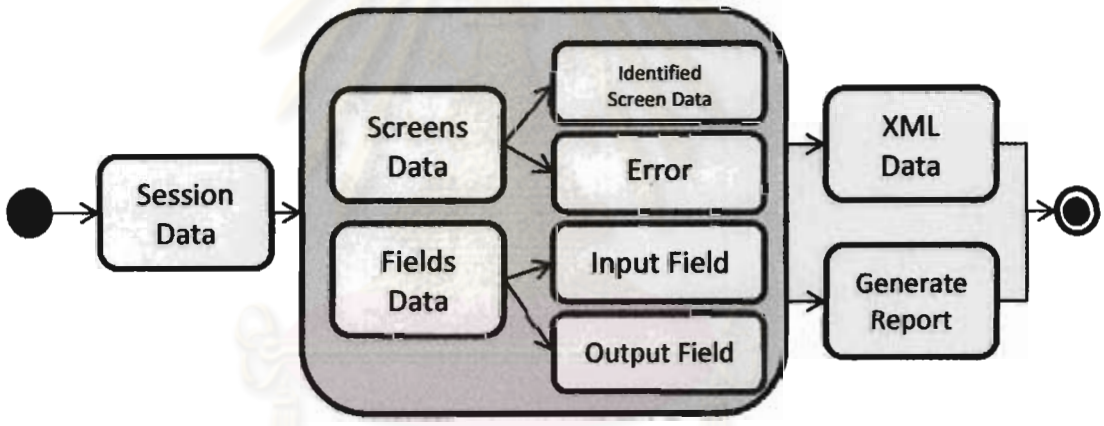
Routing.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
 <xs:element name="routing">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="screen"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="screen">
  <xs:complexType>
   <xs:sequence>
    <xs:element maxOccurs="unbounded" ref="destination"/>
   </xs:sequence>
   <xs:attribute name="id" use="required" type="xs:integer"/>
  </xs:complexType>
 </xs:element>
 <xs:element name="destination">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="screen_gateway"/>
    <xs:element maxOccurs="unbounded" ref="input_field"/>
    <xs:element ref="command"/>
```

รูปที่ 5.7 ไฟล์เอ็กซ์เอ็มแอลสคีมมาของการเก็บข้อมูลประเภทเราติง

```
    </xs:sequence>
    <xs:attribute name="id" use="required" type="xs:integer"/>
   </xs:complexType>
  </xs:element>
  <xs:element name="screen_gateway" type="xs:integer"/>
  <xs:element name="input_field">
   <xs:complexType>
    <xs:simpleContent>
     <xs:extension base="xs:NCName">
      <xs:attribute name="id" type="xs:integer"/>
     </xs:extension>
    </xs:simpleContent>
   </xs:complexType>
  </xs:element>
  <xs:element name="command" type="xs:string"/>
 </xs:schema>
```

รูปที่ 5.8 ไฟล์เอ็กซ์เอ็มแอลสครีมมาของการเก็บข้อมูลประเภทเราติง



รูปที่ 5.9 ขั้นตอนการทำงานของเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอ

โปรแกรมเลียนแบบเครื่องปลายทาง

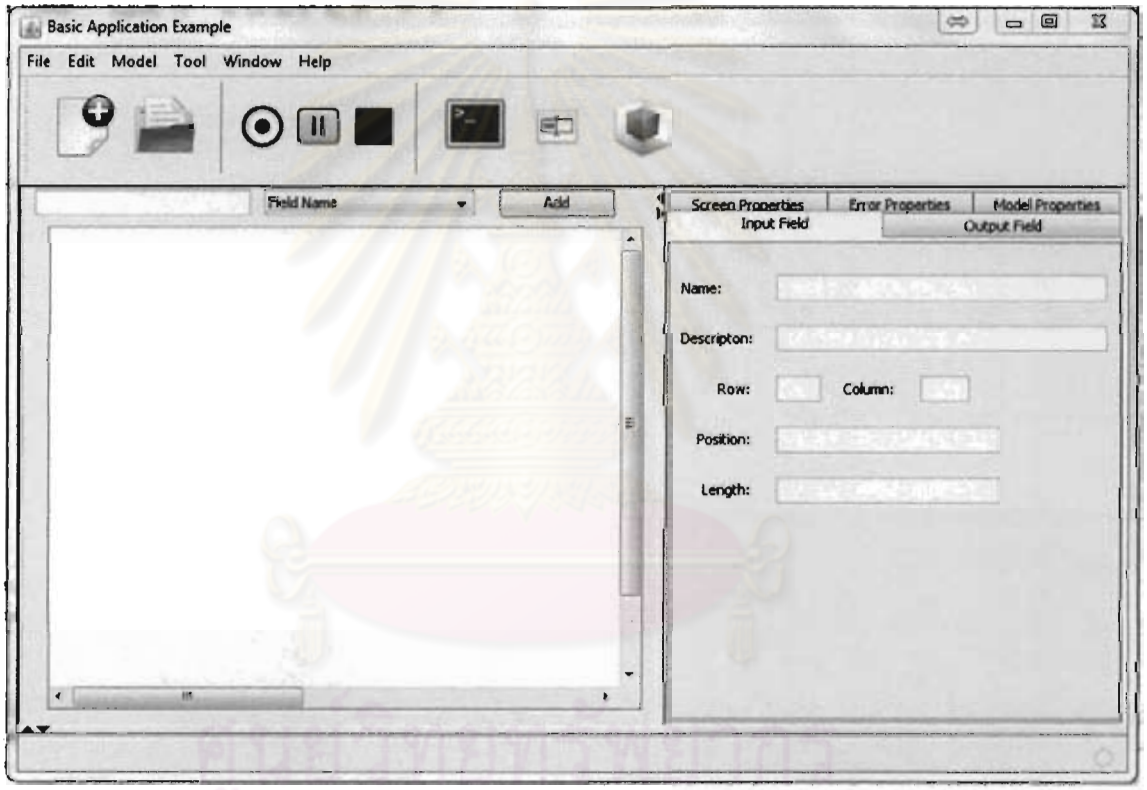
สำหรับรายงานที่สร้างขึ้นมาให้กับผู้พัฒนา จะประกอบไปด้วย 3 ส่วนหลัก ๆ คือ ตารางบอกฟิลด์ข้อมูลเข้าของหน้าจอนั้น ๆ ทั้งหมด ตารางบอกฟิลด์ข้อมูลออกของหน้าจอนั้น ๆ ทั้งหมด ซึ่งแบ่งเป็น ฟิลด์ข้อมูลออกแบบเดี่ยวและแบบตาราง และสุดท้ายคือ ตารางบอก Input Screen ที่จำเป็นต้องกำหนดให้กับเมธอด setValue สำหรับ Session เพื่อไปยัง Target Screen เป้าหมายที่กำหนด ในขณะที่อยู่ ณ Screen ปัจจุบันนั้น ๆ ตัวอย่างดังตารางที่ 3 แสดงให้เห็นว่ามี Input Screen สองหน้าจอที่จำเป็นต้องส่งค่าไปเป็นพารามิเตอร์ให้กับเมธอด เพื่อไปยัง Credit Card Info Screen ในขณะที่หน้าจอปัจจุบันอยู่ที่หน้า Login System เป็นต้น

ตารางที่ 5.1 ข้อมูลระบุ Input Screen ที่ต้องผ่านทางไปยังหน้าจอ Credit Card Info

Screen Name : Login System	
Target Screen : Credit Card Info	
Input Screen	**Input Field**
Login System	Username Password
Account Info	Credit card number

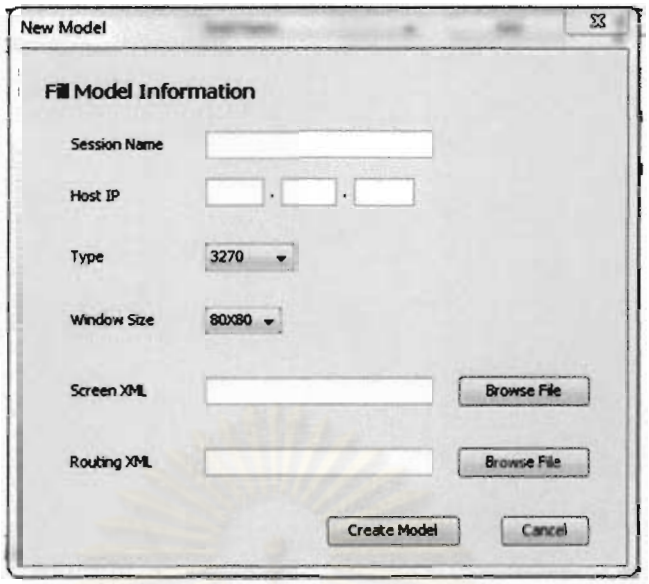
5.4 ส่วนต่อประสานผู้ใช้งานของเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอ

ส่วนต่อประสานผู้ใช้งานหลักของเครื่องมือสร้างแบบส่วนต่อประสานสามารถแสดงได้ดังรูปที่ 5.10 โดยจะแบ่งเครื่องมือในการทำงานออกเป็น 3 ส่วน ดังนี้



รูปที่ 5.10 ส่วนต่อประสานผู้ใช้งานเครื่องมือสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง

1) ส่วนการสร้างแบบจำลอง เป็นส่วนที่ให้ผู้ใช้งานสร้างแบบจำลองเพื่อเก็บข้อมูลหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางใหม่ หรือให้เปิดแบบจำลองเดิมที่เคยเก็บข้อมูลไว้เพื่อเก็บหน้าจอเพิ่มเติมได้อีก โดยเมื่อมีการสร้างแบบจำลองใหม่ ก็จะต้องทำการเก็บข้อมูลต่าง ๆ ผ่านส่วนต่อประสานผู้ใช้งานดังรูปที่ 5.10



รูปที่ 5.11 ส่วนต่อประสานในการสร้างแบบจำลอง

2) ส่วนการบันทึกการปฏิสัมพันธ์ เป็นส่วนของเครื่องมือเพื่อให้ผู้ใช้เลือกประเภทของการเก็บข้อมูลต่างๆ บนหน้าจอ

3) ส่วนของการแสดงผลข้อมูล เป็นการแสดงผลของข้อมูลที่ได้ทำการเก็บข้อมูลไว้แล้วจากแบบจำลองเดิมที่มีอยู่

บทที่ 6

การทดสอบส่วนต่อประสานโปรแกรมประยุกต์

และการประเมินผล

ในบทนี้จะกล่าวถึงการทดสอบส่วนต่อประสานโปรแกรมประยุกต์ที่พัฒนาขึ้น โดยผู้วิจัยมีเป้าหมายเพื่อต้องการที่จะทราบถึงประสิทธิภาพในการทำงานเมื่อเทียบกับการใช้ส่วนต่อประสานโปรแกรมประยุกต์แบบเดิม

6.1 การทดลองเพื่อเปรียบผลจากการปรับปรุงระบบเก่าด้วยส่วนต่อประสานโปรแกรมประยุกต์

6.1.1 วัตถุประสงค์ของการทดลอง

ในการทดลองนี้มีวัตถุประสงค์เพื่อทำการเปรียบเทียบผลการใช้ส่วนต่อประสานโปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นในงานวิจัยนี้ โดยเปรียบเทียบกับไลบรารี่ EHLLAPI โดยจะทำการทดสอบเรื่อง 1) ประสิทธิภาพการทำงานโดยวัดการขั้นตอนในการทำงานของผู้พัฒนาโปรแกรมที่ต้องใช้ไปในการเขียนโปรแกรมโดยใช้ส่วนต่อประสานโปรแกรมประยุกต์แต่ละประเภท 2) ประสิทธิภาพการทำงานของส่วนต่อประสานโปรแกรมประยุกต์โดยวัดจากจำนวนโค้ดที่ผู้พัฒนาต้องใช้ในการเขียนโปรแกรม

6.1.2 วิธีการทดลอง

จากวัตถุประสงค์การทดลองเราสามารถนำมาสร้างเป็นโครงสร้างของการทดลองโดยมีรายละเอียดดังนี้

3) ผู้พัฒนาโปรแกรมปรับปรุงส่วนต่อประสานผู้ใช้งานของระบบไอซีบีเอส ตามขอบเขตที่ธนาคารกำหนด เพื่อหาข้อบกพร่องนำมาปรับปรุงแก้ไขส่วนต่อประสานโปรแกรมประยุกต์ที่พัฒนาขึ้นในงานวิจัย
4) ผู้พัฒนาโปรแกรมเลือกขั้นตอนการทำงานของระบบไอซีบีเอสหนึ่งขั้นตอนทำมาพัฒนาด้วยส่วนต่อประสานโปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นในงานวิจัย
5) เปรียบเทียบจำนวนขั้นตอนการเขียนโปรแกรมจากส่วนต่อประสานโปรแกรมประยุกต์ทั้งสองประเภท
6) เปรียบเทียบจำนวนโค้ดที่ใช้ในการพัฒนาหนึ่งขั้นตอนข้างต้น

6.1.3 โจทย์ปัญหาที่ใช้ในการทดลอง

ในการทดลอง ผู้วิจัยได้ทำการปรับปรุงส่วนต่อประสานใหม่ของระบบไอซีบีเอส ด้วยไลบรารี่ EHLLAPI โดยหยิบขอบเขตของการทำงานแสดงได้ดังรูปที่ 6.1 โดยมีรายละเอียดขั้นตอนการทำงานดังนี้

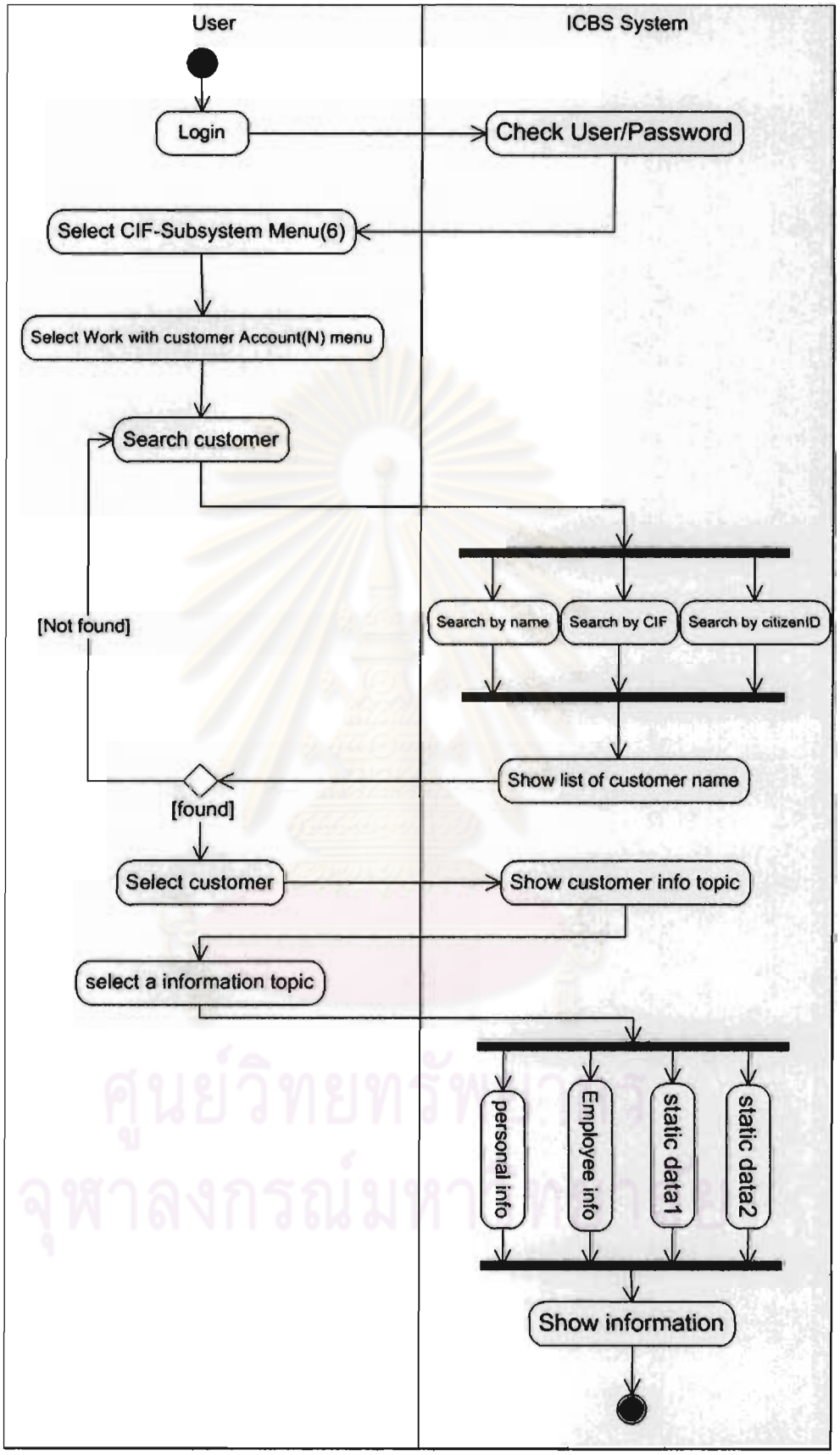
1) ผู้ใช้งานทำการล็อคอินเข้าสู่ระบบไอซีบีเอส
2) ระบบตรวจสอบชื่อและรหัสผ่านผู้ใช้งาน
3) ระบบแสดงหน้าจอเมนูหลักอัตโนมัติ
4) ระบบเข้าสู่หน้าจอเมนูย่อยอัตโนมัติ
5) ผู้ใช้งานเลือกเมนูย่อยค้นหาข้อมูลลูกค้า
6) ระบบเข้าสู่หน้าค้นหาข้อมูลของลูกค้า
7) ผู้ใช้งานเลือกประเภทการค้นหาข้อมูล
    a. ค้นหาด้วยชื่อ
    b. ค้นหาด้วยหมายเลขลูกค้าซีไอเอฟ (CIF)
    c. ค้นหาด้วยหมายเลขบัตรประจำตัวประชาชน
8) ระบบแสดงรายชื่อลูกค้าที่ตรงกับข้อมูลที่ค้นหา
9) ผู้ใช้งานเลือกชื่อลูกค้าที่ต้องการดูข้อมูล
10) ระบบแสดงผลข้อมูลตามหัวข้อต่าง ๆ

สำหรับรายละเอียดการทำงานของโปรแกรมทั้งหมดอยู่ในภาคผนวก ก ซึ่งหลังจากนั้นผู้วิจัยทำการหยิบขั้นตอนการทำงานมาหนึ่งขั้นตอน โดยในงานวิจัยนี้ ผู้วิจัยได้เลือกขั้นตอนการทำงานที่ 1 ถึงขั้นตอนที่ 3 ในการปรับปรุงระบบด้วยส่วนต่อประสานโปรแกรมประยุกต์ในงานวิจัย

6.1.4 สภาพแวดล้อมที่ใช้ในการทดลอง

เครื่องคอมพิวเตอร์ตั้งโต๊ะ (Desktop Computer) 1 เครื่อง

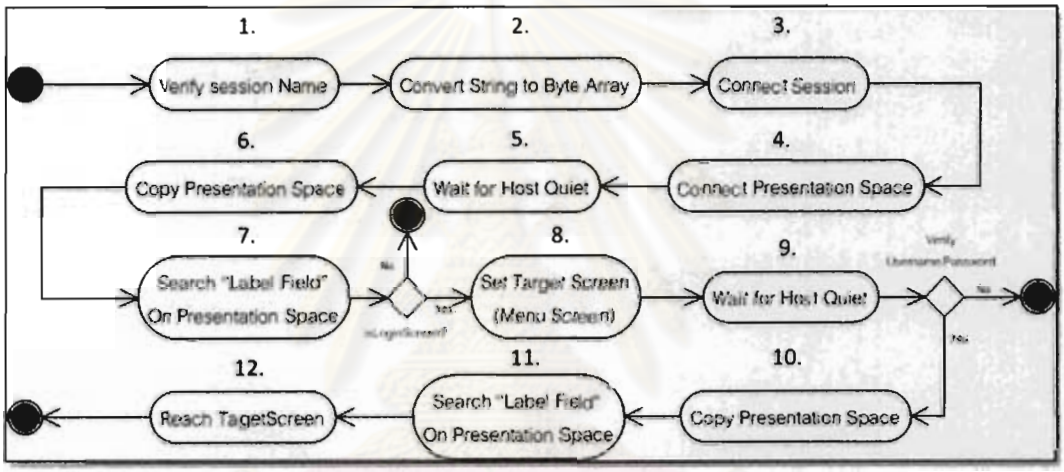
- หน่วยประมวล Pentium[R] 4 ความเร็ว 2.80 กิกะเฮิร์ต
- หน่วยความจำหลักขนาด 504 เมกกะไบต์
- ฮาร์ดดิสก์ความเร็ว 7,200 รอบ/วินาที ขนาด 40 กิกะไบต์



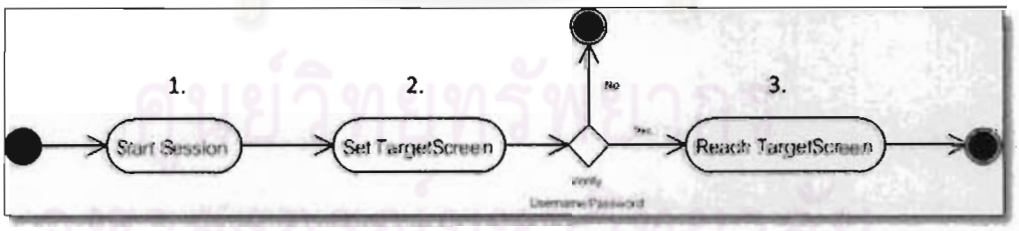
รูปที่ 6.1 แผนภาพขั้นตอนการทำงานของการเรียกดูข้อมูลลูกค้าจากระบบไอซีบีเอส

6.1.5 ผลการทดลอง

จากการปรับปรุงระบบเก่าด้วยส่วนต่อประสานโปรแกรมประยุกต์ EHLLAPI การเขียนโปรแกรมด้วยไลบรารี่นี้ ผู้พัฒนาจำเป็นต้องทำความเข้าใจหมายเลขฟังชันก์ที่จะใช้ในการส่งข้อมูลเพื่อติดต่อกับระบบเก่า แปลงค่าสตริงต่าง ๆ ให้อยู่ในชนิดข้อมูลไบท์ และเขียนโปรแกรมเพื่อรู้จำหน้าจอต่างๆ เราสามารถแสดงกระบวนการพัฒนาจากขั้นตอนที่ 1 ถึง 3 ได้ดังรูปที่ 6.2 ในขณะเดียวกัน เมื่อใช้ส่วนต่อประสานโปรแกรมประยุกต์ในงานวิจัยการพัฒนา จะมีขั้นตอนที่ใช้ในการเขียนโปรแกรมดังรูปที่ 6.3 ผู้พัฒนาสามารถเขียนโปรแกรมติดต่อกับระบบเก่าได้โดยไม่จำเป็นต้องทำความเข้าใจระบบเก่าหรือคอยจัดการเซสชันต่าง ๆ ของระบบเก่า โดยส่วนต่อประสานโปรแกรมประยุกต์จะเป็นตัวจัดการให้แบบอัตโนมัติ เราสามารถสรุปผลการเปรียบได้ดังตารางที่ 6.1



รูปที่ 6.2 แผนภาพขั้นตอนการเขียนโปรแกรมด้วยไลบรารี่ EHLLAPI



รูปที่ 6.3 แผนภาพขั้นตอนการเขียนโปรแกรมด้วยส่วนต่อประสานโปรแกรมประยุกต์ในงานวิจัย

ตารางที่ 6.1 จำนวนขั้นตอนกระบวนการในการเขียนโปรแกรมติดต่อกับโปรแกรมเลียนแบบเครื่องปลายทาง

เอพีไอที่ใช้ในการพัฒนาโปรแกรม	จำนวนขั้นตอนในการเขียนโปรแกรม
EHLLAPI	12
TEWAPI	3

จากกระบวนการเขียนโปรแกรมต่าง ๆ ข้างต้นเราสามารถสรุปจำนวนบรรทัดในการเขียนโค้ดของโปรแกรมดังกล่าวได้ดังตารางที่ 6.2 โค้ดที่เกิดขึ้นเกิดจากการเขียนโปรแกรมเพื่อทำการรู้จำหน้าจอ การแปลงข้อมูลประเภทต่างๆ เพื่อใช้งานร่วมกับฟังก์ชันของไลบรารี่ EHLLAPI และการเขียนโปรแกรมจัดการเซสชันต่าง ๆ เป็นต้น ซึ่งส่วนต่อประสานโปรแกรมประยุกต์ใหม่ในงานวิจัยสามารถจัดการเรื่องดังกล่าให้ และสามารถปรับปรุงเปลี่ยนแปลงแก้ไขข้อมูลได้ในอนาคตอีกด้วย

ตารางที่ 6.2 เปรียบเทียบจำนวนบรรทัดของโค้ดที่ใช้ในการเขียนโปรแกรมของไลบรารี่ EHLLAPI และส่วนต่อประสานโปรแกรมประยุกต์

เอพีไอที่ใช้ในการพัฒนา	จำนวนบรรทัดของโค้ด
EHLLAPI	1011
TEWAPI	3

บทที่ 7

สรุปผลการวิจัยและข้อเสนอแนะ

7.1 สรุปการวิจัย

ในวิทยานิพนธ์ฉบับนี้ได้นำเสนอวิธีการวิเคราะห์การเลือกวิธีการปรับปรุงระบบเก่า โดยเริ่มจากการกำหนดตัวผู้เกี่ยวข้องกับระบบเก่าเพื่อหาความต้องการทั้งหมด จากนั้นถึงวิเคราะห์ระบบเก่าและเทคโนโลยีที่มีอยู่ เพื่อกำหนดวิธีการปรับปรุงระบบเก่าซึ่งช่วยในการลดความเสี่ยงต่าง ๆ ที่อาจเกิดขึ้นในการปรับปรุงระบบเก่า ความต้องการที่สำคัญที่ทำให้ต้องมีการปรับปรุงระบบเก่านั้นเกิดจากธนาคารสแตนดาร์ดชาร์ดเตอร์ดต้องการรับผู้พิการทางการเห็นเข้าทำงานด้านคอมพิวเตอร์ร่วมกับระบบเก่า และพบปัญหาที่ทำให้ผู้พิการทางการเห็นไม่สามารถทำงานได้อย่างมีประสิทธิภาพมาเกิดจากโปรแกรมอ่านหน้าจอไม่สามารถทำงานร่วมกับโปรแกรมเลียนแบบเครื่องปลายทางซึ่งใช้ในการติดต่อกับระบบเก่าได้อย่างมีประสิทธิภาพ การปรับปรุงครั้งนี้จึงไม่ควรจะต้องแก้ไขโค้ดระบบเก่า อันเป็นเหตุที่ทำให้มีความเสี่ยงต่อการสูญหายของกระบวนการทางธุรกิจ และต้องการการพัฒนาที่รวดเร็ว กลวิธีสกรีนสเครปปิงจึงเป็นวิธีที่สำคัญที่จะส่วนปรับปรุงส่วนต่อประสานผู้ใช้งานให้โปรแกรมอ่านหน้าจอสามารถอ่านได้ อย่างไรก็ตามเทคนิคดังกล่าวยังคงมีข้อจำกัดในหลาย ๆ ด้าน เช่น ผู้พัฒนาจำเป็นต้องทำความเข้าใจระบบเก่า หรือการพัฒนาที่มีขั้นตอนที่ซ้ำซากและพบปัญหาในเรื่องการใช้งานไลบรารี่เก่าที่ช่วยในการดึงข้อมูลจากระบบเก่า

ด้วยเหตุนี้จึงได้มีการพัฒนาส่วนต่อประสานโปรแกรมประยุกต์เพื่อครอบการทำงานของระบบเก่าและเปิดเมธอดให้ผู้พัฒนาระบบสามารถเรียกใช้เพื่อดึงข้อมูลจากระบบเก่ามาสร้างส่วนต่อประสานผู้ใช้งานใหม่โดยไม่จำเป็นต้องทำความเข้าใจระบบเก่า อีกทั้งยังได้พัฒนาเครื่องมือที่ช่วยในการสร้างแบบจำลองการปฏิสัมพันธ์เพื่อให้ผู้เชี่ยวชาญโปรแกรมเลียนแบบเครื่องปลายทางสามารถเก็บข้อมูลหน้าจอ โดยเครื่องมือจะช่วยสร้างเอกสารอ้างอิงเพื่อสะดวกต่อผู้พัฒนาโปรแกรมใช้ได้ในภายหลังอีกด้วย

7.2 ข้อเสนอแนะ

ในการวิจัยนี้ได้นำเสนอเครื่องมือต้นแบบสำหรับสร้างแบบจำลองการปฏิสัมพันธ์หน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง ซึ่งสามารถเก็บข้อมูลประเภทเซสชัน สกรีน และฟิลด์ต่าง ๆ ได้ ในขั้นตอนถัดไปอาจทำการพัฒนาฟังก์ชันของเครื่องมือต้นแบบเพิ่ม เพื่อให้สามารถใช้งานได้ง่ายยิ่งขึ้น โดยมีฟังก์ชันที่น่าสนใจดังต่อไปนี้คือ

1.) ฟังก์ชันแผนภาพโครงสร้างต้นไม้แสดงขั้นตอนการติดต่อหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางแต่ละหน้าจอ

2.) ฟังก์ชันแก้ไขข้อมูลประเภทต่างๆ หลังจากเก็บข้อมูลเสร็จไปแล้ว

3.) ฟังก์ชันการตรวจสอบข้อผิดพลาดจากการเก็บข้อมูลในหน้าจอโปรแกรมเลียนเครื่องปลายทางต่าง ๆ

4.) ฟังก์ชันการรู้จำหน้าจอที่ยังไม่ได้การเก็บข้อมูลหรือมีการเปลี่ยนแปลงข้อมูลในหน้าจอต่าง ๆ เพื่อให้ผู้ใช้ทราบและสามารถแก้ไขหรือเก็บข้อมูลเพิ่มเติมได้

รายการอ้างอิง

[1] Zoufaly, F. ,2002, *Issues and Challenges Facing Legacy Systems*. [online] Available from: http://www.developer.com/mgmt/article.php/1492531/Issues-and-Challenges-Facing-Legacy-Systems.htm[2008, january 25]

[2] I. B. M. Corporation. ,July 2006, *Emulator Programming*. [online] Available from: http://publib.boulder.ibm.com/infocenter/pcomhelp/v5r9/index.jsp?topic=/com.ibm.pcomm.doc/books/html/emulator_programming07.htm

[3] Seacord, R.C., et al., *Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices*: Addison Wesley Prodessional, February 14, 2003.

[4] Comella-Dorda, S., et al., "*A Survey of Black-Box Modernization Approaches for Information Systems*," presented at the Proceedings of the International Conference on Software Maintenance (ICSM'00), 2000.

[5] G. C. Mihaela Carmen TRUFASU, "*Reasons for migrating legacy systems,*" *presented at the Economy Informatics*, 2005.

[6] Canfora, G., et al., "*Migrating Interactive Legacy Systems To Web Services,*" *presented at the Proceedings of the Conference on Software Maintenance and Reengineering*, 2006.

[7] Internet Engineering Task Force. [Online] Available form: http://www.ietf.org/ [2010, December 14]

[8] Bouknight, J., *Programming with the Host Access APIs*, First Edition ed.: IBM, 1999.

[9] Marini, J., "*Document Object Model : Processing Structured Documents*," 1st edition ed: McGraw-Hill/OsborneMedia, 2002, :400.

[10] Attachmate. *Legacy Application Modeling With Attachmate Verastream Host Integrator : Programmatic Integration vs. Traditional Screen Scraping*. [online] Available from: http://www.attachmate.com/WhitePapers/Literature_0799.htm

[11] attachmate, "*Accelerating Integration with Verastream Host Integrator High-level*

abstraction is the key."

[12] Durand, C., "*Internationalizing Mainframe Applications through Screen Scraping,*" *presented at the Proceedings of the 3rd International Conference on Internationalization, Design and Global Development*: Held as Part of HCI International 2009, San Diego, CA, 2009.

[13] Noffsinger, W.B., et al., "*Legacy object modeling speeds software integration,*" Commun. ACM, 41, 1998 :80-89

[14] Comella-Dorda, S.W., Kurt ; Seacord, Robert C. ; Robert, John, "*A Survey of Legacy System Modernization Approaches,*" 2000.

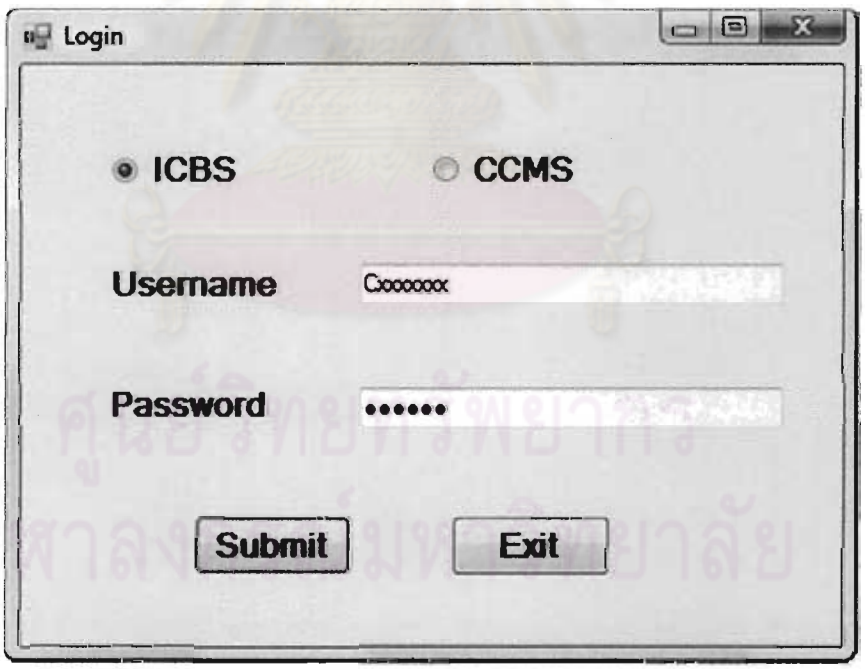
ภาคผนวก

ภาคผนวก ก

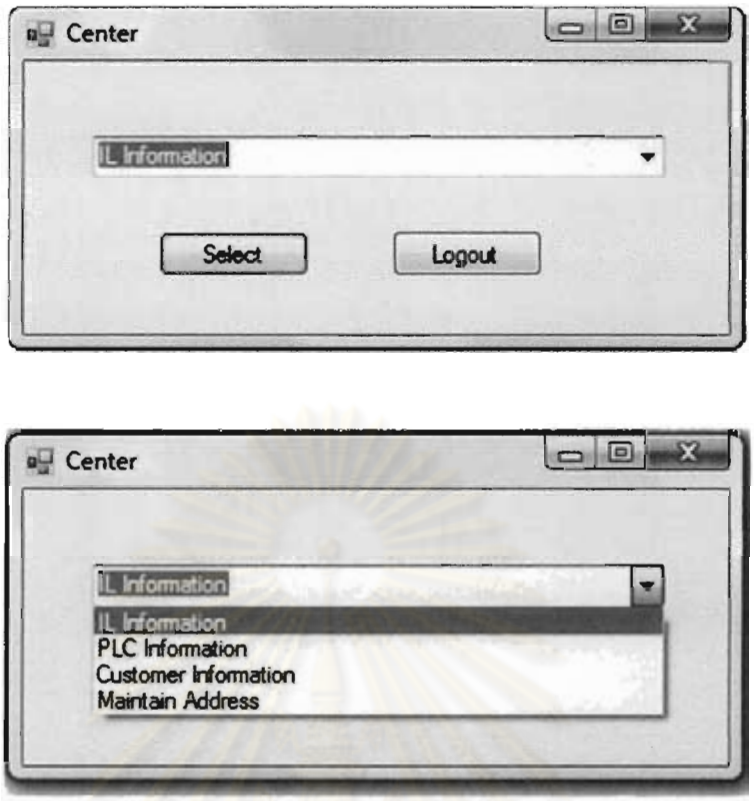
การใช้งานโปรแกรม ICBS

ก.1 การเข้าสู่หน้าเมนูหลักของโปรแกรม ICBS

1. ผู้ใช้งานระบบจะต้องทำการล็อคอินเพื่อเข้าสู่ระบบ ICBS ด้วยสวนต่อประสาน รูปที่ ก.1
2. ผู้ใช้งานระบบเลือกเมนูที่ต้องการใช้งานเพื่อเข้าสู่ระบบ โดยแบ่งเมนูออกได้เป็น 3 ประเภท ดังนี้
    a. IL Information
    b. PCL Information
    c. Customer Information
    d. Maintenance



รูปที่ ก.1 หน้าล็อคอินเข้าระบบ ICBS



รูปที่ ก.2 รายการเมนูหลักของโปรแกรม ICBS สำหรับผู้พิการทางการเห็น

ก.2 การใช้งานเมนู IL Information

1. ใส่เลขที่บัตรเพื่อเรียกดูข้อมูล ดังรูปที่ ก.3

รูปที่ ก.3 ส่วนต่อประสานสำหรับกรอกเลขที่บัตร IL

2. หากเลขที่บัตรถูกต้อง ผู้ใช้งานจะเข้าสู่หน้าเมนูหลักของ IL Information ผู้ใช้งานสามารถเลือกดูข้อมูลได้ตามแถบต่าง ๆ ดังต่อไปนี้ ตามรูปที่ ก. 4

   a. Basic Note Data

b. Payment Schedule Display

c. Billed/ Unpaid Payment

d. Memorands Display

e. History Select

3. เมื่อผู้ใช้ต้องการดูข้อมูลในแท็บต่าง ๆ จะต้องกดปุ่ม Fetch Data เพื่อเรียกข้อมูลจากหน้าจอโปรแกรมเลียนแบบเครื่องปลายทางมาใช้ในการแสดงผลลัพธ์ที่ส่วนต่อประสานใหม่ จากรูปที่ ก.4 แสดงตัวอย่างการดึงข้อมูลจากหน้า Basic Note Data โดยมีขั้นตอนการทำงานดังต่อไปนี้

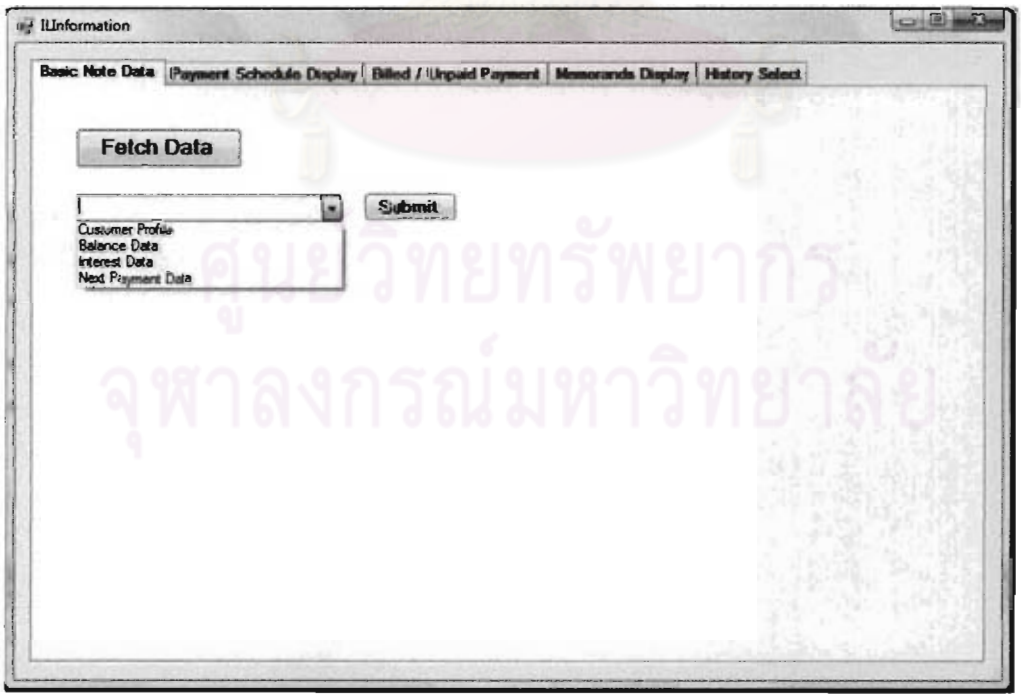
   a. กดปุ่ม Fetch Data เพื่อดึงข้อมูลจากโปรแกรมเลียนแบบเครื่องปลายทาง

   b. เลือกหัวข้อในดรอปดาวน์ลิสต์เพื่อเรียกดูข้อมูลในแต่ละประเภท

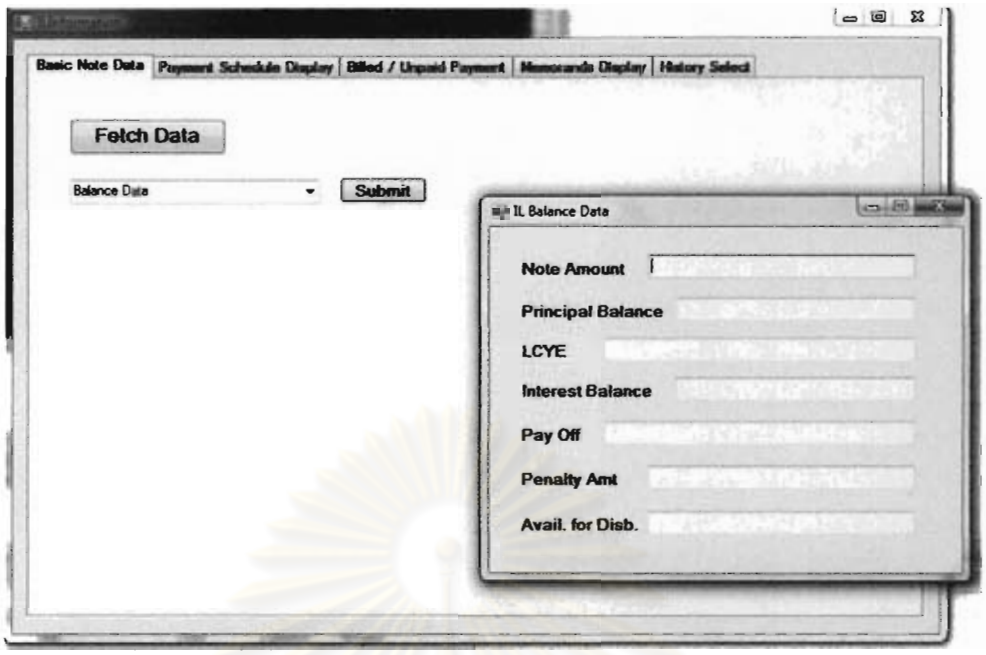
   c. กดปุ่ม Submit เพื่อแสดงผลข้อมูลออกมาดังส่วนต่อประสานรูปที่ ก.5

4. จากรูป ก. 6 และ ก. 7 แสดงตัวอย่างส่วนต่อประสานในหน้า Bill/Unpaid Payment และ History Select

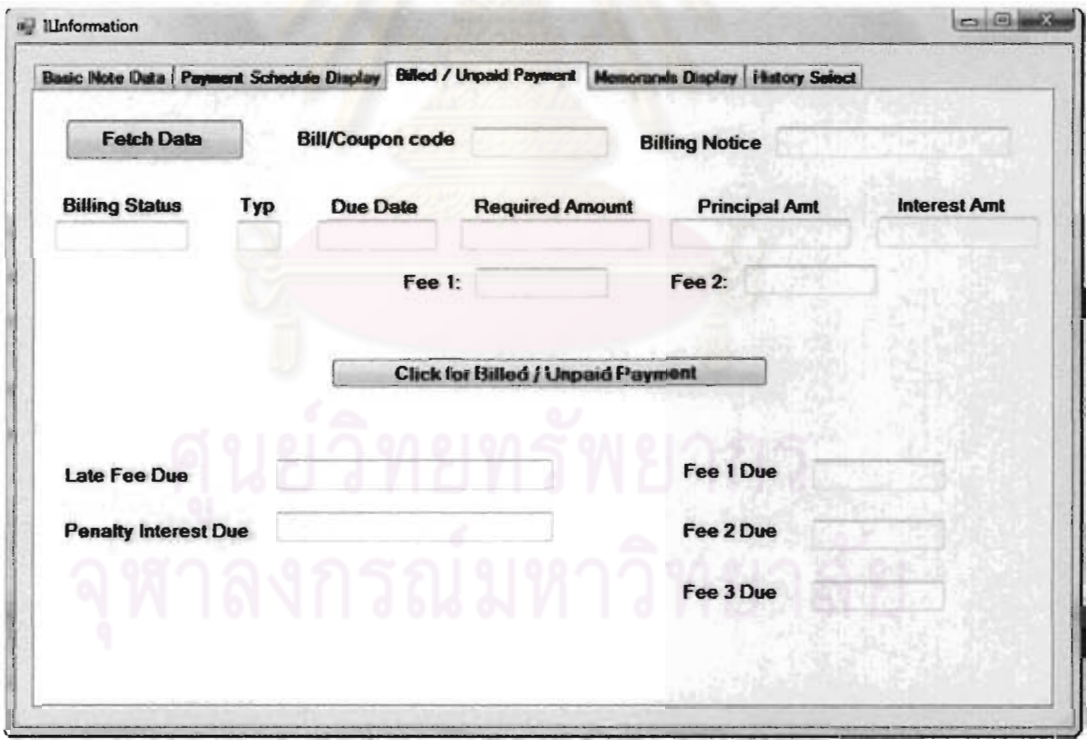
5. สำหรับการใช้งานให้เมนู PCL Information มีลักษณะเช่นเดียวกับ IL Informations



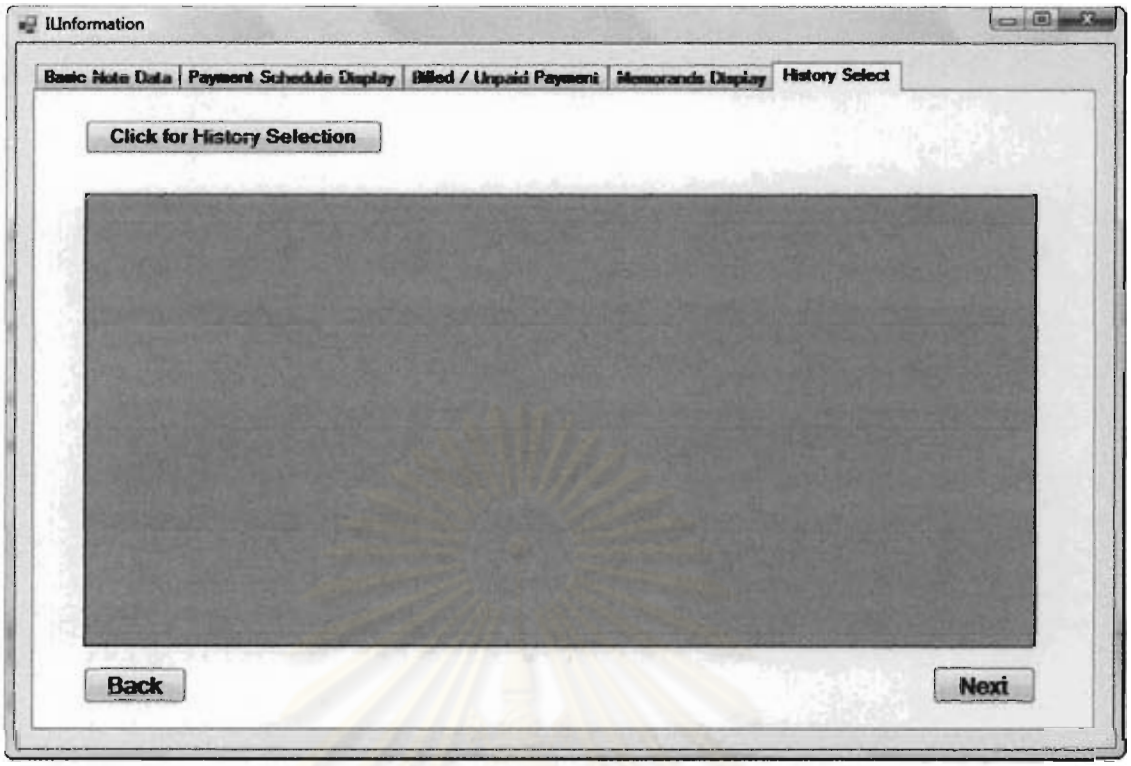
รูปที่ ก.4 ส่วนต่อประสานหน้า Basic Note Data ของ IL Information



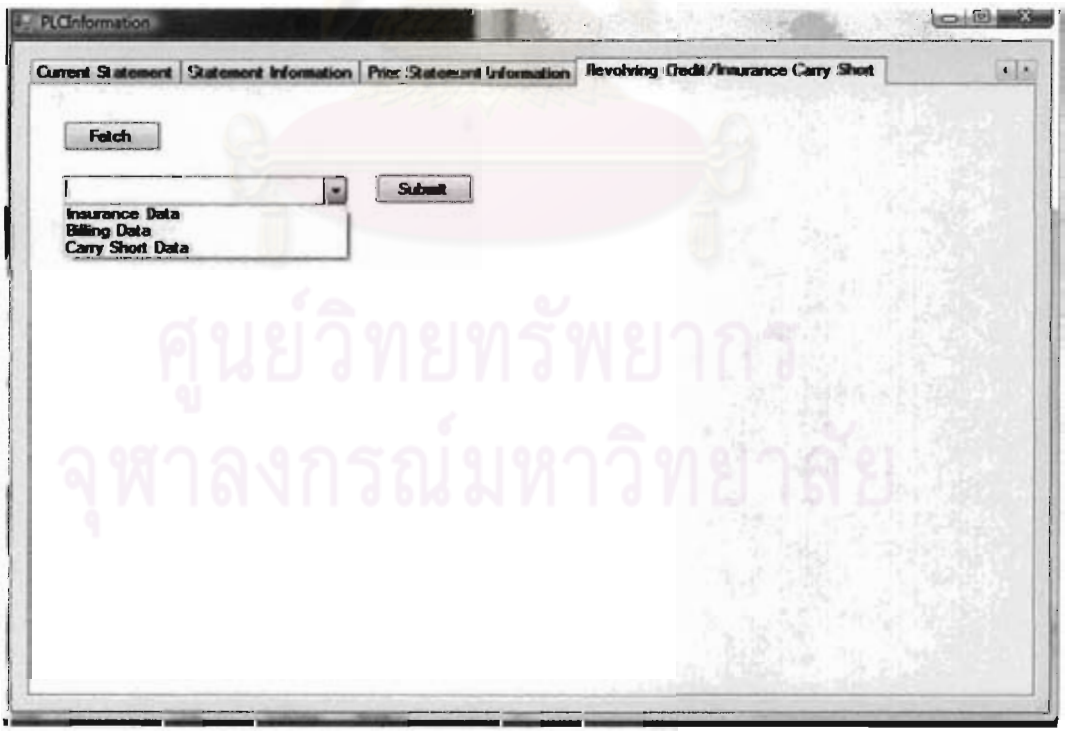
รูปที่ ก.5 ส่วนต่อประสานแสดงผลลัพธ์ของ Balance Data



รูปที่ ก.6 ส่วนต่อประสานหน้าจอ Billed/ Unpaid Payment



รูปที่ ก.7 ส่วนต่อประสานหน้าจอ History Select



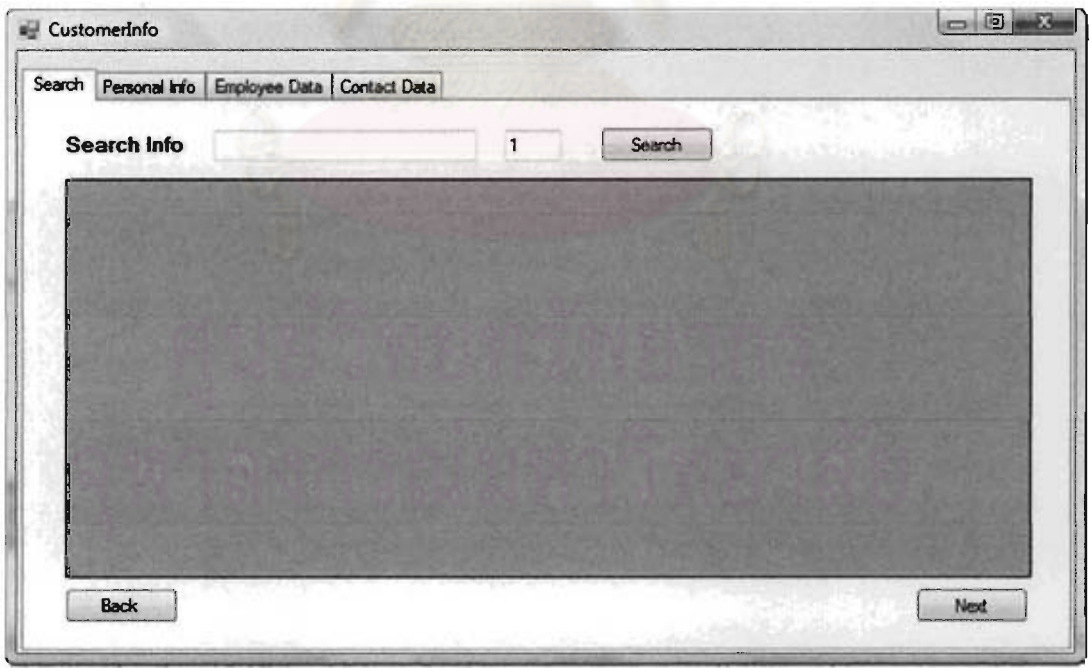
รูปที่ ก.8 ส่วนต่อประสานหน้าจอ Revolving Credit/Insurance Carry Short

ของเมนู PLCInformation

ก.3 การใช้งานเมนู Customer Information

ในเมนูนี้มีไว้ใช้สำหรับการค้นหาข้อมูลทั่วไปเกี่ยวกับลูกค้า เช่น ชื่อ ที่อยู่ เบอร์โทรศัพท์ โดยมีขั้นตอนการทำงานดังต่อไปนี้

1. ในแท็บ Search เมื่อเราต้องการค้นหา สามารถระบุค่าที่จะใช้การค้นหา โดยประเภทของค่าขึ้นอยู่กับรหัสด้านหลังกล่อง Search info โดยกำหนดเป็นหมายเลขต่าง ๆ ซึ่งมีความหมายดังนี้
   a. หมายเลข 1 = ชื่อบุคคล
   b. หมายเลข 2 = Customer Key
   c. หมายเลข 4 = รหัสประจำตัวประชาชน
2. เมื่อกำหนดข้อมูลเรียบร้อยให้ทำการกดปุ่ม Search เพื่อเรียกข้อมูลจากโปรแกรมเลียนแบบเครื่องปลายทาง
3. เราสามารถดูข้อมูลต่าง ๆ ได้ในแท็บ Personal Info, Employee Data และ Contact Data



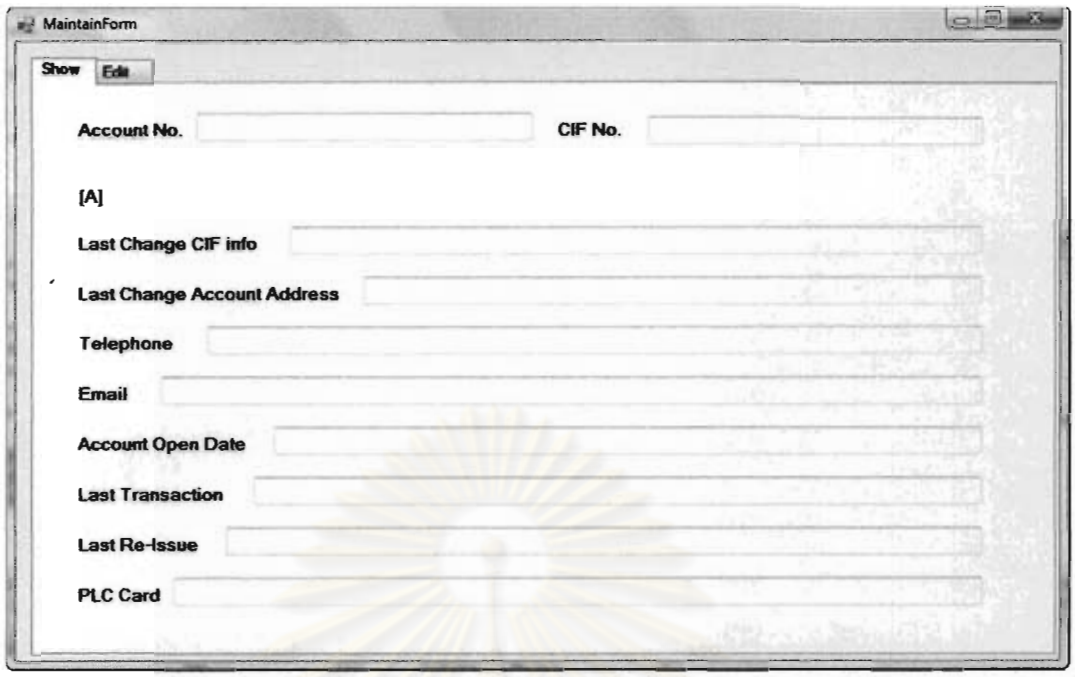
รูปที่ ก.9 ส่วนต่อประสานหน้า Search ของเมนู Customer Information

รูปที่ ก.10 ส่วนต่อประสานหน้า Employee Data ของเมนู Customer Information

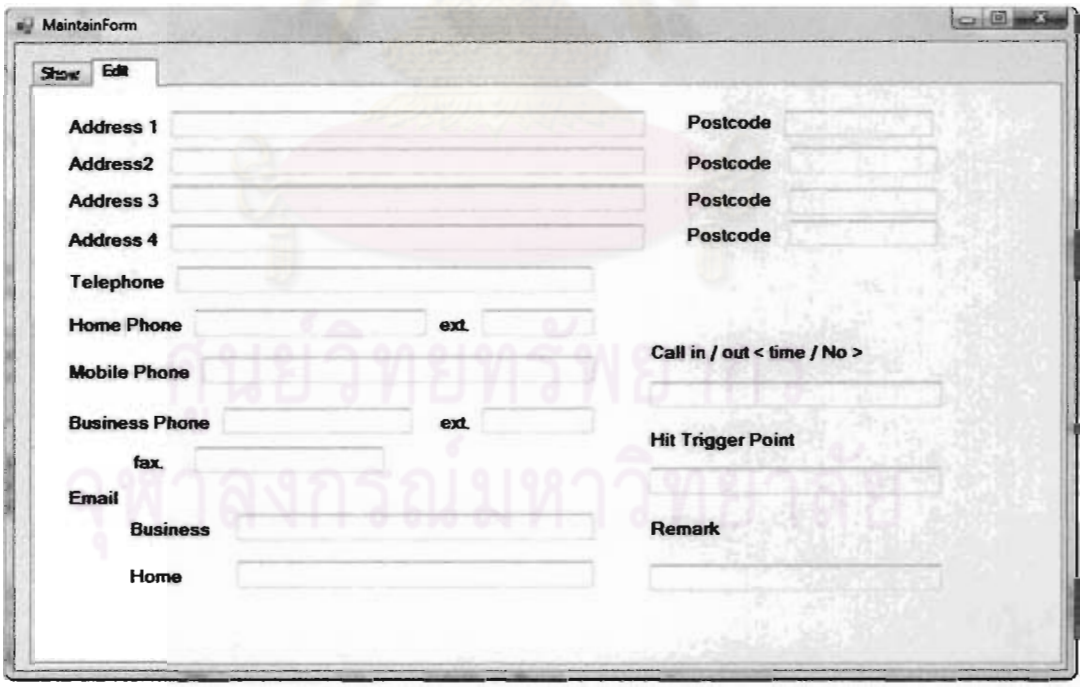
ก.4 การใช้งานเมนู Maintenance

1. ใส่เลขที่บัตร และระบุประเภทของบัตรดังรูปที่
2. เมื่อเข้าสู่หน้าจอหลักจะปรากฏแท็บให้สามารถใช้งานได้ 2 แท็บ ดังนี้
   a. แท็บ Show มีไว้แสดงผลข้อมูลของลูกค้า
   b. แท็บ Edit มีไว้แก้ไขข้อมูลของลูกค้า
3. การแก้ไขจะได้รับการตรวจสอบจากฝ่ายตรวจสอบในภายหลัง

รูปที่ ก.11 ส่วนต่อประสานหน้ากรอกเลขที่บัตร เมนู Maintenace



รูปที่ ก.12 ส่วนต่อประสานหน้า Show ของเมนู Maintenance

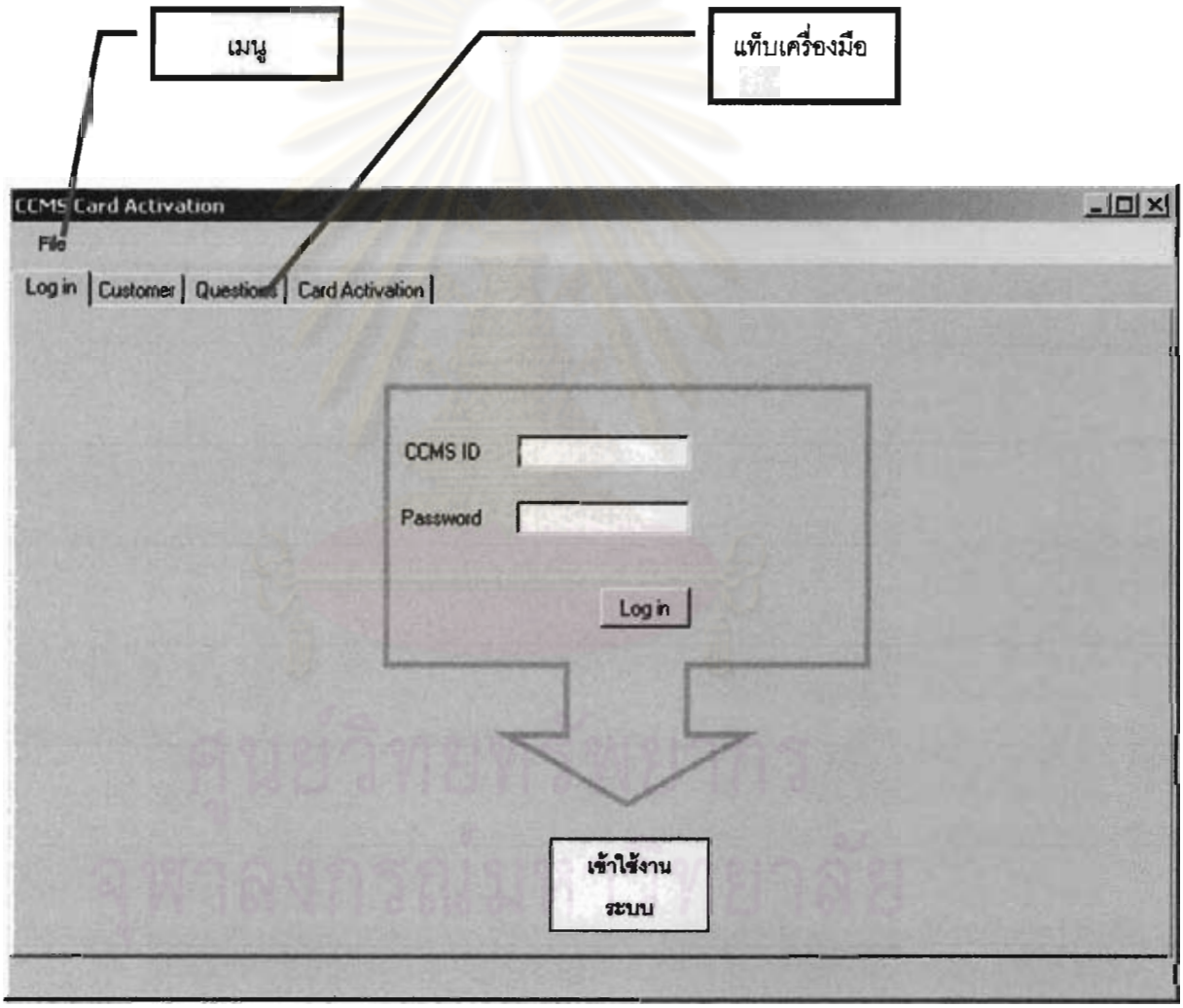


รูปที่ ก.13 ส่วนต่อประสานหน้า Edit ของเมนู Maintenance

ภาคผนวก ข

วิธีการใช้งานโปรแกรม CCMS Card Activation

CCMS Card Activation เป็นโปรแกรมที่สร้างขึ้นครอบโปรแกรม CCMS เพื่อใช้สำหรับหน้าที่การเปิดบัตรเครดิตโดยเฉพาะ (Activate Credit) ดังนั้นผู้ใช้จำเป็นต้องดำเนินการผ่านทางโปรแกรมนี้เท่านั้น โดยหลีกเลี่ยงการใช้งานผ่านโปรแกรม CCMS เดิมที่มีอยู่ หน้าจอหลักของโปรแกรม CCMS Card Activation เป็นไปดังรูปที่ ข.1



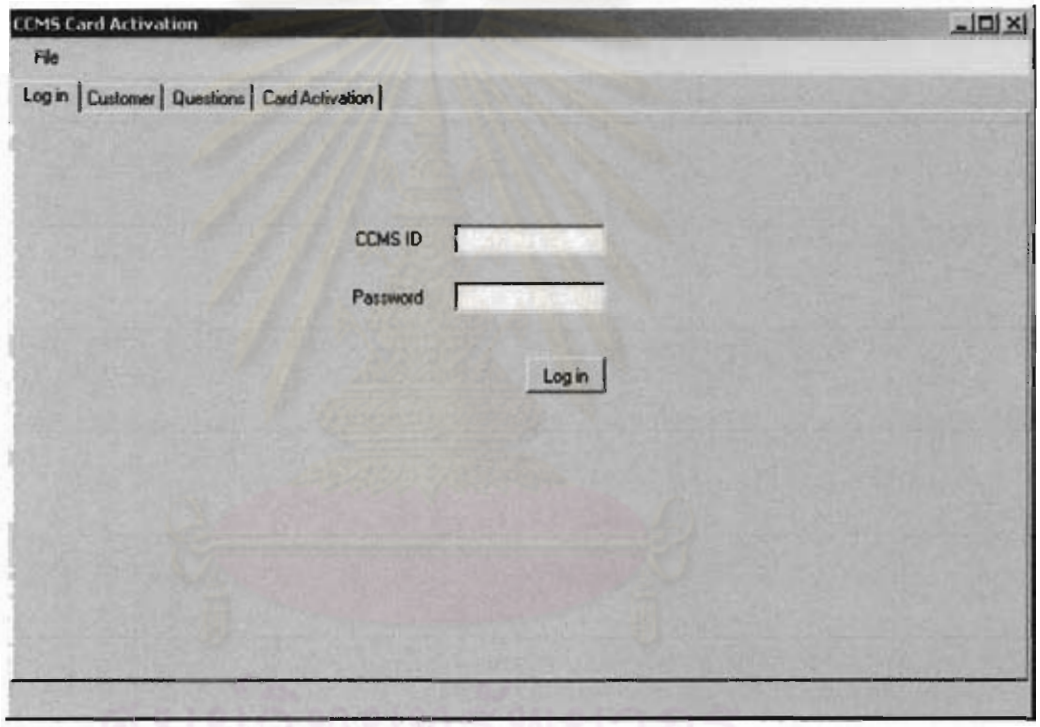
รูปที่ ข.1 หน้าจอหลักโปรแกรม CCMS Card Activation

CCMS Card Activation ประกอบไปด้วยหน้าต่างการใช้งาน 4 หน้า คือ

1. Log in เพื่อเข้าใช้งานโปรแกรม
2. Customer เพื่อค้นหาข้อมูลลูกค้า
3. Questions เพื่อดึงคำถามใช้ในการยืนยันตัวตนลูกค้า
4. Activate เพื่อใช้ในการ activate บัตรเครดิตให้ลูกค้า

ข.1 ส่วนต่อประสานหน้าเข้าสู่ระบบ

เป็นหน้าสำหรับให้เจ้าหน้าที่ล็อคอินเพื่อใช้งานระบบ หากไม่ทำการล็อคอินก่อนการใช้งานโปรแกรมจะไม่สามารถใช้งานได้



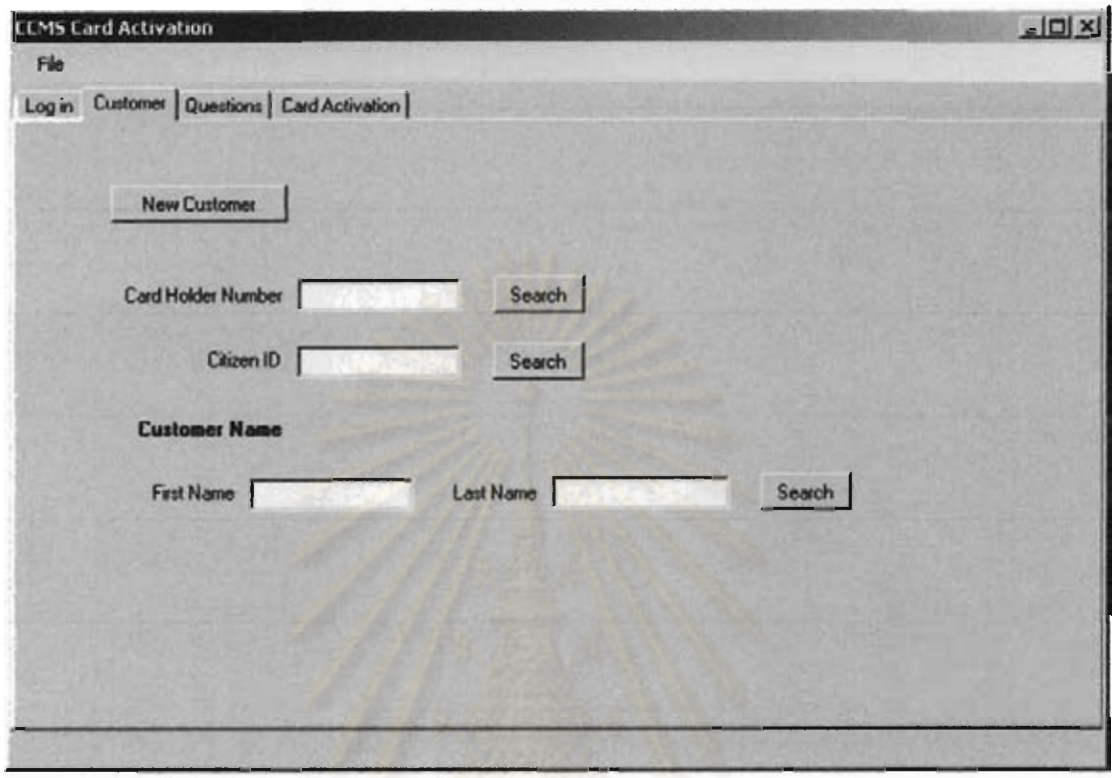
รูปที่ ข.2 หน้าล็อคอินเข้าสู่ระบบ

ขั้นตอนการเข้าสู่ระบบ

1. ผู้ใช้งานกรอก CCMS ID และรหัสผ่าน
2. เมื่อกดปุ่ม Log in และรหัสผ่านนั้นถูกต้อง จะมี dialog box เพื่อให้ใส่รหัสผ่านของโปรแกรม ICBS เป็นขั้นตอนถัดไป ดังรูป
3. ผู้ใช้งานกรอก ICBS ID และรหัสผ่าน

ข.2 หน้า Customer

เป็นหน้าสำหรับใช้ค้นข้อมูลเพื่อนำมายืนยันตัวตนลูกค้า



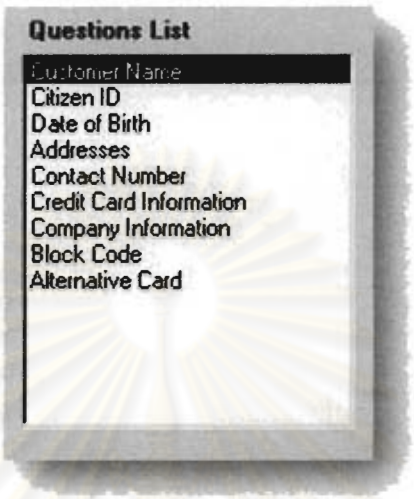
รูปที่ ข.3 หน้า Customer

ขั้นตอนการใช้งาน

1. กดปุ่ม New Customer เมื่อเริ่มต้นลูกค้าคนใหม่
2. รับข้อมูลจากลูกค้าเพื่อดึงคำถามที่จะใช้ยืนยันตัวตนและตรวจสอบลูกค้าในหน้า Questions ซึ่งในการดึงคำถามต้องใช้เลขบัตรเครดิตการ์ด (Card Holder Number)
3. หากลูกค้าไม่ทราบเลขบัตรเครดิต ผู้ใช้สามารถหาข้อมูลเลขบัตรเครดิตได้จาก เลขบัตรประชาชน (Citizen ID)
4. หากลูกค้าไม่ทราบเลขบัตรประชาชน ผู้ใช้สามารถหาข้อมูลเลขบัตรประชาชนได้จาก ชื่อและนามสกุลของลูกค้า
   4.1. การค้นหา ผู้ใช้จำเป็นต้องกรอกชื่อลูกค้าเป็น ภาษาอังกฤษ ให้ถูกต้อง แต่นามสกุลอาจไม่ต้องกรอกให้ครบก็ได้ (แต่การค้นหาจะใช้เวลานานขึ้น ดังนั้นการกรอกข้อมูลครบเป็นทางเลือกที่ดีที่สุด)

ข.3 หน้า Questions

เป็นหน้าแสดงคำถามประเภทต่าง ๆ เพื่อใช้ตรวจสอบและยืนยันตัวตนของลูกค้า โดยคำถามนั้นได้ถูกแบ่งไว้เป็นหมวดหมู่ดังรูป 4 ซึ่งแต่ละหมวดหมู่ก็จะมีหมวดหมู่ย่อยลงไปอีกดังนี้



รูปที่ ข.4 Questions List (หมวดหมู่คำถาม)

1. Customer Name ชื่อของลูกค้า
2. Citizen ID เลขบัตรประชาชนของลูกค้า
3. Date of Birth วัน เดือน ปีเกิด ของลูกค้า
4. Addresses ที่อยู่ของลูกค้า
   - 4.1. Preferred Address ที่อยู่ที่ใช้รับส่งพัสดุจากทางธนาคาร
   - 4.2. Alternative Address ที่อยู่สำรอง
5. Contact Number* หมายเลขติดต่อลูกค้า
6. Credit Card Information แสดงข้อมูลบัตรเครดิต
   - 6.1. General Information ข้อมูลทั่วไป
     - 6.1.1. Card Holder Number แสดงหมายเลขบัตรเครดิตที่แสดงข้อมูล
     - 6.1.2. Open Date วันเปิดใช้งานบัตรเครดิต
     - 6.1.3. Card Fee Date วันชำระค่าธรรมเนียมบัตรเครดิต
     - 6.1.4. Expiration Date วันหมดอายุบัตรเครดิต
     - 6.1.5. Billing Cycle รอบวันชำระค่าใช้จ่าย
   - 6.2. Block Status สถานะ Block code

6.2.1. Duo Account ตรวจสถานะว่าลูกค้ามีบัตรเครดิตอื่นอีกหรือไม่ (D = มี)

6.2.2. Block Code ตรวจสอบสถานะการติด Block code ของบัตรในช่องนี้เป็นไปตามสัญลักษณ์(X,Y)

6.2.2.1. หากมีสัญลักษณ์ Code เกิดที่ตำแหน่ง X หมายถึง บัตรที่กำลังจะเปิดติด Block code ตัวนั้น ๆ อยู่

6.2.2.2. หากมีสัญลักษณ์ Code เกิดที่ตำแหน่ง Y หมายถึง มีบัตรเครดิตของลูกค้าติด Block code ตัวนั้น ๆ อยู่

เช่น หากเกิดสัญลักษณ์(H,) แสดงว่า มีบัตรเครดิตบางใบติด Block code H อยู่ การตรวจสอบข้อมูลว่าบัตรไหนติด Block code นั้น ต้องเข้าดูข้อมูลในหมวด Block Code

6.3. Other Credit Card ชื่อผู้ถือบัตรเครดิตอื่น ๆ

7. Company Information ข้อมูลทั่วไปของบริษัทที่ลูกค้าทำงานอยู่

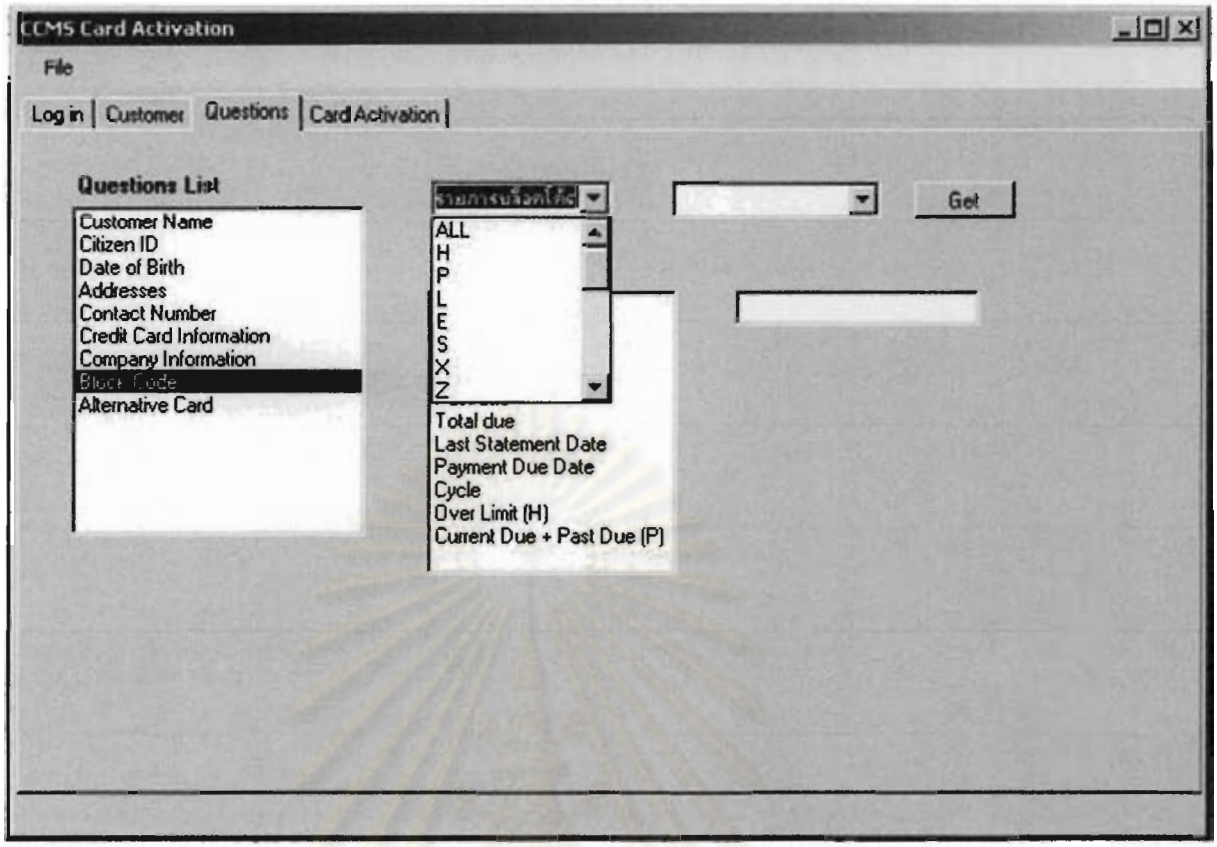
8. Block Code* แสดงรายละเอียดบัตรเครดิตที่เกี่ยวข้องกับ Block code ชนิดต่าง ๆ

9. Alternative Card* บัตรอื่น ๆ ของลูกค้าที่เป็นเจ้าของอยู่นอกเหนือไปจากบัตรเครดิต

การดูรายละเอียดของบัตรเครดิตที่ติด Block Code

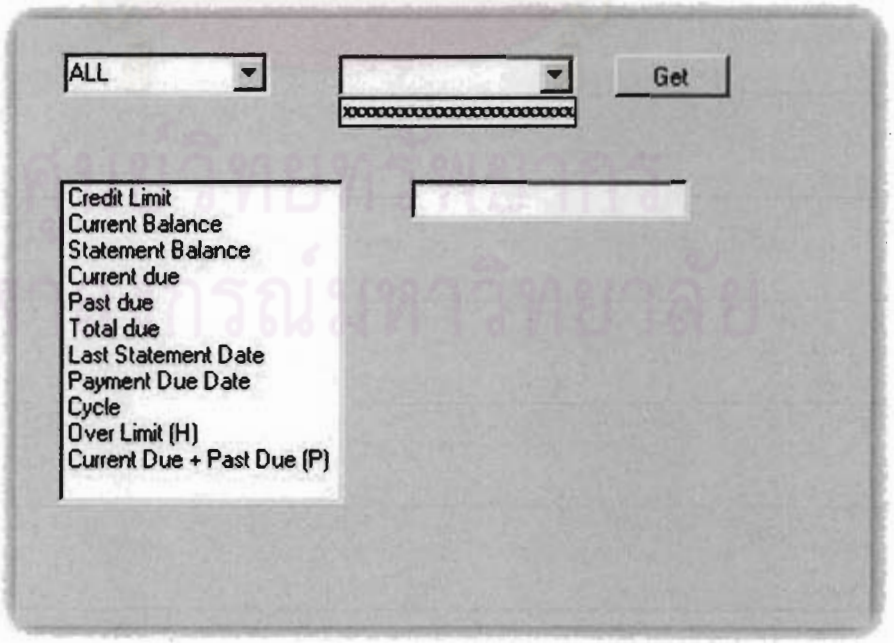
การดูรายละเอียดบัตรเครดิตที่ติด Block code สามารถดูได้ผ่านหมวด "Block code" โดยเราจะทราบสถานะของ Block code ที่เกิดขึ้นได้จากหมวด Credit Card Information > Block Code แล้วว่า มีสถานะเป็น Code อะไร จากให้ทำตามขั้นตอนต่างๆ ดังนี้

1. เลือกหมวด "Block Code" จาก Question List ด้านซ้ายมือของโปรแกรม จากนั้น เลือกรายการสถานะ Block code ที่เกิดขึ้น จะแสดงรายการบัตรเครดิตที่ติด Block code ชนิดนั้นๆ ทางคอลโทรลลิสต์ด้านขวามือ



รูปที่ ข.5 รายการของ Block Code

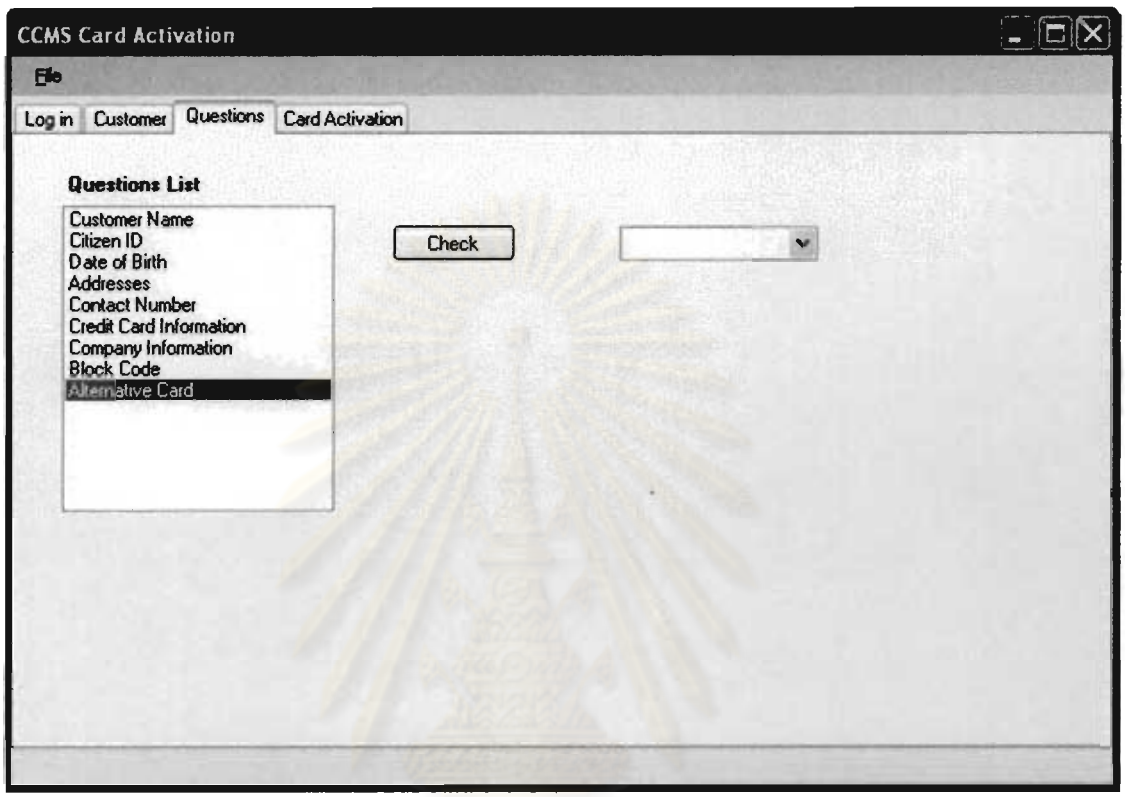
2. เลือกเลขบัตรเครดิตที่เราต้องการเรียกดูข้อมูล



รูปที่ ข.6 รายการเลขบัตรเครดิตตามสถานะของ Block code

3. กดปุ่ม Get เพื่อดึงข้อมูล
4. รายการข้อมูลจะถูกแสดงตามหัวข้อลิสต์ด้านล่าง

การเรียกดูชนิดบัตรเครดิตอื่น ๆ นอกเหนือจากบัตรเครดิต



รูปที่ ข.7 หมวดหมู่ Alternative Card

1. เลือกหมวด Alternative Card
2. กดปุ่ม Check
3. เลือกรายการด้านขวาของปุ่ม Check

ข.4 หน้า Activation

รูปที่ ข.8 หน้า Card Activation

การแอกติเวทบัตรเครดิต

1. กรอกเลขที่บัตรเครดิตที่ช่อง "Card Holder Number"
2. กรอกชื่อเจ้าหน้าที่ ผู้ทำการ Activate บัตรเครดิตที่ช่อง "Activated By"
3. กดปุ่ม "Activate" เพื่อดำเนินการ

การบันทึกรายละเอียดการเปิดบัตรเครดิต

1. กรอกเลขที่บัตรเครดิตที่ช่อง "Card Holder Number"
2. กรอก Memo ที่ช่อง "Memo"
3. กดปุ่ม "Save" เพื่อบันทึกข้อมูล

ข.5 การเข้าถึงส่วนต่าง ๆ ของโปรแกรมด้วยคีย์ลัดบนแป้นพิมพ์

เนื่องด้วยโปรแกรมนี้ เป็นโปรแกรมสำหรับคนตาบอด จึงจำเป็นต้องทำความเข้าใจคีย์ลัดที่จำเป็นบนคีย์บอร์ดเพื่อใช้งานได้อย่างประสิทธิภาพ

Menu Bar

- กด Alt + ตัวอักษรที่ได้รับการขีดเส้นใต้ , เลื่อนด้วยลูกศร
  R = reset, F = file, O = log out, E = exit

Collection Tab (Log in, Customer, Questions, Card Activation)

- เคลื่อนที่ไปด้านหน้า กด Ctrl + Tab
- เคลื่อนที่ไปด้านหลัง กด Ctrl + Shift + Tab

Control ต่าง ๆ

- กด Tab เพื่อเลื่อนไปยัง Control ถัดไป
- กด Shift + Tab เพื่อเลื่อนไปยัง Control ก่อนหน้า
- กด Space หรือ Enter หรือ
  Alt + ตัวอักษรที่ถูกขีดเส้นใต้ เพื่อสั่งงาน Control

หน้า Log in

ปุ่ม Log in = Alt + L

หน้า Customer

ปุ่ม New Customer = Alt + N

ปุ่ม Search = Alt + S

หน้า Questions

ปุ่ม Change Info = Alt + C

กด Page Up เพื่อโฟกัสไปยังหมวดคำถามหลัก

หน้า Card Activation

ปุ่ม Activate = Alt + A

ปุ่ม Save = Alt + S

ภาคผนวก ค

High Level Language Application Programming Interface Function

ส่วนต่อประสานโปรแกรมประยุกต์นี้ ถูกพัฒนาเริ่มแรกโดย IBM ซึ่งมีหน้าที่เอาไว้ติดต่อสื่อสารส่งข้อมูลระหว่างโปรแกรมที่เราพัฒนาและโปรแกรมเลียนแบบเครื่องปลายทาง การใช้งานจะเป็นการเรียกฟังชั่นก์ผ่านไลบรารี่ HLLAPI โดยมีเมธอดที่ใช้ในการเรียกหนึ่งเมธอด ขึ้นอยู่กับว่าเราจะส่งหมายเลขฟังก์ชั่นไหนไปเพื่อร้องขอการใช้งานคำสั่งนั้น ซึ่งการเรียกฟังก์ชั่นหนึ่ง ๆ อาจต้องการการเรียกฟังก์ชั่นก่อนหน้าขึ้นอยู่กับตารางที่ 3 ดังต่อไปนี้

ตารางที่ ค.1 หมายเลขฟังก์ชั่นที่ใช้เรียก (Calling Function)

No.	Function	Prerequisite Function Call
1	CONNECT PRESENTATION SPACE	None
2	DISCONNECT PRESENTATION SPACE	CONNECT PRESENTATION SPACE (1)
3	SEND KEY	CONNECT PRESENTATION SPACE (1)
4	WAIT	CONNECT PRESENTATION SPACE (1)
5	COPY PRESENTATION SPACE	CONNECT PRESENTATION SPACE (1)
6	SEARCH PRESENTATION SPACE	CONNECT PRESENTATION SPACE (1)
7	QUERY CURSOR LOCATION	CONNECT PRESENTATION SPACE (1)
8	COPY PRESENTATION SPACE TO STRING	CONNECT PRESENTATION SPACE (1)
9	SET SESSION PARAMETERS	None
10	QUERY SESSIONS	None
11	RESERVE	CONNECT PRESENTATION SPACE (1)
12	RELEASE	CONNECT PRESENTATION SPACE (1)
13	COPY OIA	CONNECT PRESENTATION SPACE (1)
14	QUERY FIELD ATTRIBUTE	CONNECT PRESENTATION SPACE (1)
15	COPY STRING TO PRESENTATION SPACE	CONNECT PRESENTATION SPACE (1)
18	PAUSE	None
20	QUERY SYSTEM	None

No.	Function	Prerequisite Function Call
21	RESET SYSTEM	None
22	QUERY SESSION STATUS	None
23	START HOST NOTIFICATION	None
24	QUERY HOST UPDATE	START HOST NOTIFICATION (23)
25	STOP HOST NOTIFICATION	START HOST NOTIFICATION (23)
30	SEARCH FIELD	CONNECT PRESENTATION SPACE (1)
31	FIND FIELD POSITION	CONNECT PRESENTATION SPACE (1)
32	FIND FIELD LENGTH	CONNECT PRESENTATION SPACE (1)
33	COPY STRING TO FIELD	CONNECT PRESENTATION SPACE (1)
34	COPY FIELD TO STRING	CONNECT PRESENTATION SPACE (1)
40	SET CURSOR	CONNECT PRESENTATION SPACE (1)
41	START CLOSE INTERCEPT	None
42	QUERY CLOSE INTERCEPT	START CLOSE INTERCEPT (41)
43	STOP CLOSE INTERCEPT	START CLOSE INTERCEPT (41)
50	START KEYSTROKE INTERCEPT	None
51	GET KEY	START KEYSTROKE INTERCEPT (50)
52	POST INTERCEPT STATUS	START KEYSTROKE INTERCEPT (50)
53	STOP KEYSTROKE INTERCEPT	START KEYSTROKE INTERCEPT (50)
90	SEND FILE	None
91	RECEIVE FILE	None
99	CONVERT POSITION or CONVERT ROWCOL	None
208	Copy Selected Text to Clipboard	CONNECT PRESENTATION SPACE (1)
209	Copy and Append Selected Text to Clipboard	CONNECT PRESENTATION SPACE (1)
210	Paste Text From Clipboard to Presentation Space	CONNECT PRESENTATION SPACE (1)
211	Select all Text Within Current Presentation Space	CONNECT PRESENTATION SPACE (1)

ภาคผนวก ง

Terminal Emulator Wrapper Aplication Programming Interface

ง.1 Package Index

- com.tewapi.context
- com.tewapi.database
- com.tewapi.exception
- com.tewapi.factory
- com.tewapi.utils
- com.tewapi.wrapper
- com.tewapi.xml

ง.2 คลาส Wrapper

Class com.tewapi.wrapper.Wrapper	
Public class Wrapper	
เป็นคลาสที่ในการจัดการกับข้อมูลหน้าจอโปรแกรมเลียนแบบเครื่องปลายทาง โดยได้แบ่งประเภทของข้อมูลออกเป็นวัตถุต่าง ๆ ดังต่อไปนี้ Session, Screen, Field, Table, Error	
Method Index	
initial()	เปิดการใช้งานเซสชันที่กำหนด
initialAll()	เปิดการใช้งานเซสชันทั้งหมด
destroy()	ปิดการใช้งานเซสชันที่กำหนด
destroyAll()	ปิดการใช้งานเซสชันทั้งหมด
setValue()	กำหนดค่าให้กับระบบเก่าของวัตถุนั้น ๆ
getValue()	ร้องขอค่าจากระบบเก่าของวัตถุนั้น ๆ
getValueList()	ร้องขอกลุ่มของค่าจากระบบเก่าของวัตถุนั้น ๆ
getAllValue()	ร้องขอค่าทั้งหมดจากระบบเก่าของวัตถุนั้น ๆ
getArrtibute()	ร้องขอแอททริบิวส์ที่เลือกของวัตถุนั้น ๆ
getArrtbuteiList()	ร้องขอกลุ่มของแอททริบิวส์ที่เลือกของวัตถุนั้น ๆ
getAllArrtibute()	ร้องขอแอททริบิวส์ทั้งหมดที่เลือกของวัตถุนั้น ๆ
Methods	
initial	

- public boolean initial(String selecter)
  - selecter = session
  - เปิดการใช้งานเซสชันที่ได้ระบุชื่อลงในพารามิเตอร์ selecter
  - Returns: true เมื่อเปิดเซสชันสำเร็จ, false เปิดเซสชันไม่สำเร็จ

initialAll

- public boolean initialAll( )
  - selecter = session
  - เปิดการใช้งานเซสชันทั้งหมดของระบบเก่า
  - Returns: true เมื่อเปิดเซสชันสำเร็จ, false เปิดเซสชันไม่สำเร็จ

Destroy

- public boolean destroy( )
  - selecter = session
  - ปิดการใช้งานเซสชันที่ได้ระบุชื่อลงในพารามิเตอร์ selecter
  - Returns: true เมื่อปิดเซสชันสำเร็จ, false เมื่อปิดเซสชันไม่สำเร็จ

destroyAll

- public boolean destroyAll( )
  - selecter = session
  - ปิดการใช้งานเซสชันทั้งหมดของระบบเก่า
  - Returns: true เมื่อเปิดเซสชันสำเร็จ, false เปิดเซสชันไม่สำเร็จ

setValue

- public boolean setValue(String selecter, String value)
  - selecter = session, field, table
  - กำหนดค่าให้กับวัตถุที่ระบุลงในพารามิเตอร์ selecter โดยที่
    - Field = กำหนดค่าให้กับฟิลด์หนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ
      - #ชื่อหน้าจอที่ต้องการ
      - #ชื่อฟิลด์ที่ต้องการ
    - Table = กำหนดค่าให้กับตารางหนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ
      - #ชื่อหน้าจอที่ต้องการ
      - #ชื่อตารางที่ต้องการ
      - แอททริบิวส์ [column,row]
    - Session = กำหนดหน้าจอที่ต้องการจะไปถึง โดยระบุค่าต่าง ๆ ดังนี้ คือ
      - #ชื่อเซสชันที่ต้องการ
      - #หน้าจอเป้าหมาย
      - #หน้าจอที่ต้องกำหนดค่าที่ผู้ใช้งานส่งมา

• Returns: true เมื่อกำหนดค่าสำเร็จ, false เมื่อกำหนดค่าไม่สำเร็จ
getValue
- public String getValue(String selecter, String value) • selecter = field, table • ร้องขอค่าของวัตถุที่ระบุลงในพารามิเตอร์ selecter โดยที่ ▪ Field = ร้องขอค่าฟิลด์หนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ • #ชื่อหน้าจอที่ต้องการ • #ชื่อฟิลด์ที่ต้องการ ▪ Table = กำหนดค่าให้กับตารางหนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ • #ชื่อหน้าจอที่ต้องการ • #ชื่อตารางที่ต้องการ • แอททริบิวส์ [column,row] • Returns: ค่าของวัตถุที่ร้องขอไป
getValueList
- public HashMap getValueList(String selecter, String value) • selecter = field, table • ร้องขอค่าของวัตถุที่ระบุลงในพารามิเตอร์ selecter โดยที่ ▪ Field = ร้องขอค่าฟิลด์หนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ • #ชื่อหน้าจอที่ต้องการ • #ชื่อฟิลด์ที่ต้องการ ▪ Table = กำหนดค่าให้กับตารางหนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ • #ชื่อหน้าจอที่ต้องการ • #ชื่อตารางที่ต้องการ • ชื่อคอลัมน์ที่ต้องการ • Returns: กลุ่มของค่าของวัตถุที่ร้องขอไปในรูปแบบ String
getAllValue
- public HashMap getAllValue( ) • selecter = field, table • ร้องขอค่าทั้งหมดของวัตถุที่ระบุลงในพารามิเตอร์ selecter โดยที่ ▪ Field = ร้องขอค่าฟิลด์ทั้งหมดในหน้าจอปัจจุบัน ▪ Table = ร้องขอค่าทั้งหมดของตารางในหน้าจอปัจจุบัน • Returns: กลุ่มของค่าของวัตถุที่ร้องขอไปในรูปแบบ String
getArrtibute

- public String getArrtibute(String selecter, String value)
  - selecter = Session, Screen, field, table
  - ร้องขอค่าแอททริบิวส์ของวัตถุที่ระบุลงในพารามิเตอร์ selecter
  - Returns: ค่าของแอททริบิวส์ที่ร้องขอไป

getArrtibuteList

- public HashMap getArrtibuteList(String selecter, String value)
  - selecter = Session, Screen, field, table
  - ร้องขอกลุ่มของค่าแอททริบิวส์ของวัตถุที่ระบุลงในพารามิเตอร์ selecter โดยที่
    - Field = ร้องขอค่าฟิลด์หนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ
      - #ชื่อหน้าจอที่ต้องการ
      - #ชื่อฟิลด์ที่ต้องการ
    - Table = กำหนดค่าให้กับตารางหนึ่ง โดยระบุค่าต่าง ๆ ดังนี้คือ
      - #ชื่อหน้าจอที่ต้องการ
      - #ชื่อตารางที่ต้องการ
      - ชื่อคอลัมน์ที่ต้องการ
  - Returns: กลุ่มของค่าแอททริบิวส์ของวัตถุที่ร้องขอไปในรูปแบบ String

getAllArrtibute

- public HashMap getAllArrtibute ( )
  - selecter = Session, Screen, field, table
  - ร้องขอค่าแอททริบิวส์ทั้งหมดของวัตถุทั้งหมด
  - Returns: กลุ่มของค่าแอททริบิวส์ของวัตถุที่ร้องขอไป

ประวัติผู้เขียนวิทยานิพนธ์

นายถาวร ลิ้มวัฒนาชัย เกิดเมื่อวันที่ 16 มกราคม พ.ศ. 2528 สำเร็จการศึกษาระดับปริญญาบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ จากคณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี ในปีการศึกษา 2549 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2550

โดยระหว่างการศึกษาในหลักสูตรวิทยาศาสตรมหาบัณฑิต งานวิจัยชิ้นนี้ของผู้วิจัย และคณะได้รับการคัดเลือกให้ถูกตีพิมพ์เป็นบทความวิชาการระดับประเทศเรื่อง "Terminal Emulator Wrapper Application Programming Interface" ในงาน "NCCIT" ซึ่งจัดขึ้น ที่มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ประเทศไทย ระหว่างวันที่ 11-12 พฤษภาคม พ.ศ. 2554