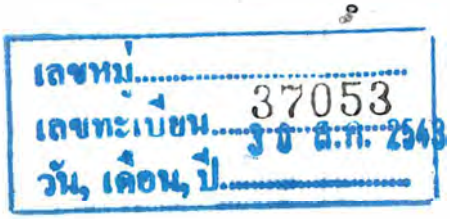
# การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

# Software License Manager


นายสุทัศน์ กันธง

นายสุริยันต์ เลาหประภานนท์




ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2542

# การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

# Software License Manager

โดย

นายสุทัศน์ กันธง

นายสุริยันต์ เลาหประภานนท์

อาจารย์ที่ปรึกษา

อาจารย์อภิเนตร อูนากูล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2542

ปริญญานิพนธ์ปีการศึกษา2542

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

Software License Manager

ผู้จัดทำ

นายสุทัศน์ กันธง รหัส 39014591

นายสุริยันต์ เลาหประภานนท์ รหัส 39014620

อาจารย์ที่ปรึกษา

(อาจารย์อภิเนตร อูนากูล)

# การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

นายสุทัศน์ กันธง 39014591
นายสุริยันต์ เลาหประภานนท์ 39014620
อาจารย์อภิเนตร อูนากูล อาจารย์ที่ปรึกษา
ปีการศึกษา 2542

**บทคัดย่อ**

ปัจจุบันคอมพิวเตอร์ส่วนบุคคลมีการใช้งานกันอย่างแพร่หลาย ผู้ใช้งานแต่ละคนมีความต้องการในการใช้งานซอฟต์แวร์ที่แตกต่างกัน ในการที่ผู้พัฒนาและผู้จำหน่ายซอฟต์แวร์แต่ละรายจะสามารถตอบสนองความต้องการแต่ละระดับของผู้ใช้งานได้นั้น จำเป็นต้องมีระบบที่จะจัดการกับซอฟต์แวร์ให้สามารถใช้งานได้ตามความต้องการของผู้ใช้ ดังนั้น โครงงานนี้จึงได้นำเสนอระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์เพื่อที่จะมาช่วยให้ผู้พัฒนาและผู้จำหน่ายซอฟต์แวร์จัดการกับซอฟต์แวร์ให้สามารถทำงานได้ตามความต้องการของผู้ใช้งานในระดับต่างๆ ได้

**Software License Manager**

Suthat Guntong
Suriyan Laohaprapanon
Apinetr Unakul (Advisor)

**ABSTRACT**

Nowadays, personal computer have been used in almost every kind of business. Every user uses software differently depended on their needs. In order to satisfy such needs, Software developers and software distributors have to employ some kind of system to manage their software. This project purposes a software license management system that can help software developers and software distributors complete their goal, to make their software satisfy customer needs in different level of using software.

# กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้คงไม่อาจสำเร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลายๆ ฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ก็คือ อาจารย์อภิเนตร อูนากูล อาจารย์ที่ปรึกษาวิทยานิพนธ์ ผู้มีวิสัยทัศน์ที่กว้างไกล คอยให้คำแนะนำ ให้ความเอาใจใส่ และช่วยเหลือตลอดมา ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก

และต้องขอขอบพระคุณบุคคลสำคัญที่สุดที่ทำให้ข้าพเจ้ามีวันนี้ ก็คือ บิดา มารดา อันเป็นที่เคารพรักยิ่ง ซึ่งได้เลี้ยงดูผู้เขียนมาเป็นอย่างดี พร้อมทั้งให้โอกาสในการศึกษาอย่างเต็มที่ และยังให้กำลังใจเอาใจใส่เสมอมา ในทุกๆ ด้านอันหาที่เปรียบมิได้ ข้าพเจ้าขอระลึกในพระคุณอันสุดประมาณ และขอกราบขอบพระคุณมา ณ ที่นี้

สุทัศน์ กันธง
สุริยันต์ เลาหประภานนท์

# สารบัญ

# สารบัญตาราง

# สารบัญภาพ

# บทที่ 1
## บทนำ

### 1.1 ที่มาของโครงงาน

อุตสาหกรรมซอฟต์แวร์ในอดีต โดยส่วนใหญ่จะเป็นการพัฒนาซอฟต์แวร์ให้กับกลุ่มลูกค้ารายใหญ่ๆ กล่าวคือ ผู้พัฒนาซอฟต์แวร์จะพัฒนาซอฟต์แวร์ตามความต้องการของลูกค้าเฉพาะราย ผู้ที่มีสิทธิ์ใช้งานซอฟต์แวร์จึงเป็นบุคคลที่อยู่ในหน่วยงานหรือองค์กรที่มีขนาดใหญ่เท่านั้น เนื่องจากลูกค้าในระดับส่วนบุคคลในอดีตยังมีเพียงจำนวนน้อย ซอฟต์แวร์ดังกล่าวจึงถูกพัฒนาเพื่องานเฉพาะ ผู้พัฒนาซอฟต์แวร์จึงสามารถควบคุมสิทธิ์การใช้งานตามความต้องการของผู้ใช้ระบบนั้น โดยไม่จำเป็นต้องพิจารณาถึงความยืดหยุ่นในการใช้งานกับระบบอื่นๆ แนวความคิดในการบริหารและควบคุมสิทธิ์จึงยังไม่ค่อยมีบทบาทมากนัก

ต่อมาเมื่อคอมพิวเตอร์ในระดับส่วนบุคคลเริ่มเข้ามามีบทบาทในชีวิตประจำวันของมนุษย์มากขึ้น ทำให้ลูกค้าในระดับส่วนบุคคลเพิ่มมากขึ้นด้วย อุตสาหกรรมซอฟต์แวร์จึงเริ่มหันมาให้ความสนใจกลุ่มลูกค้าในระดับส่วนบุคคล แต่เนื่องจากความต้องการใช้งานซอฟต์แวร์ต่างๆ ของลูกค้าในระดับนี้โดยส่วนใหญ่จะใกล้เคียงกันโดยจะไม่เฉพาะเจาะจงมากนัก ซอฟต์แวร์สำหรับลูกค้าในระดับนี้จึงเป็นซอฟต์แวร์ที่ค่อนข้างจะมีความยืดหยุ่นในการใช้งาน เป็นผลให้ผู้ใช้บางส่วนใช้งานซอฟต์แวร์ที่ไม่ได้ซื้อมาจากผู้ผลิตซอฟต์แวร์ แต่จะซื้อซอฟต์แวร์เพียงชุดเดียวแล้วนำไปติดตั้งใช้งานหลายชุด ทำให้ผู้พัฒนาซอฟต์แวร์ได้รับผลตอบแทนน้อยกว่าที่ควรจะเป็น ผู้พัฒนาซอฟต์แวร์จึงเริ่มหันมาให้ความสนใจกับการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์มากขึ้น อย่างไรก็ตาม ความต้องการสิทธิ์ในการใช้งานของลูกค้าในระดับส่วนบุคคลค่อนข้างจะมีความหลากหลาย เช่น ลูกค้าบางรายต้องการใช้งานเพียงส่วนใดส่วนหนึ่งของซอฟต์แวร์เท่านั้น หรือต้องการใช้งานเพียงแค่ช่วงระยะเวลาหนึ่ง จากความต้องการที่หลากหลายของลูกค้า จึงก่อให้เกิดปัญหากับผู้พัฒนาซอฟต์แวร์ในการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ของลูกค้า ผู้พัฒนาซอฟต์แวร์บางรายจึงได้หันมาพัฒนาระบบบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์เพื่อนำไปใช้ควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่พัฒนาขึ้น โดยระบบดังกล่าวจะเข้ามาช่วยแบ่งเบาภาระในการกำหนดสิทธิ์การใช้งานของผู้พัฒนาซอฟต์แวร์เอง และยังสามารถควบคุมสิทธิ์การใช้งานได้โดยผู้จำหน่ายซอฟต์แวร์

ในปัจจุบันเมื่ออินเตอร์เน็ตเป็นที่รู้จักกันแพร่หลายในหมู่ผู้ใช้คอมพิวเตอร์ส่วนบุคคล ได้เกิดก่อให้เกิดธุรกิจรูปแบบใหม่ซึ่งทำกันบนอินเตอร์เน็ต ที่รู้จักกันในชื่อ "พาณิชย์กรรมอิเล็กทรอนิกส์" (Electronics Commerce) ธุรกิจซอฟต์แวร์จัดว่าเป็นธุรกิจที่ค่อนข้างจะเอื้ออำนวยต่อการทำธุรกิจในลักษณะนี้ เนื่องจากผู้ใช้ซอฟต์แวร์สามารถทดลองสินค้าได้โดยการดาวน์โหลดไปทดลองใช้ก่อน ถ้าพอใจจึงค่อยตัดสินใจซื้อ อย่างไรก็ตาม การทำธุรกิจซอฟต์แวร์บนอินเตอร์เน็ตจะต้องมีองค์ประกอบหลักที่สำคัญ คือ การแจกจ่ายซอฟต์แวร์ผ่านอินเตอร์เน็ต (Electronics Software Distribution: ESD), การแจกจ่ายสิทธิ์การใช้งานซอฟต์แวร์ผ่านอินเตอร์เน็ต (Electronics Licensing Distribution: ELD) และการจ่ายเงิน

ผ่านอินเตอร์เน็ต (Electronics Payment) ดังนั้น ระบบการบริหารและการควบคุมสิทธิ์การใช้งานซอฟต์แวร์จึงจัดเป็นองค์ประกอบที่ช่วยสนับสนุนให้ธุรกิจในรูปแบบนี้มีความน่าเชื่อถือและมีความยืดหยุ่นมากยิ่งขึ้น

**1.2 ความสำคัญและแนวคิดในการทำโครงงาน**

จากปัญหาดังกล่าวข้างต้นจึงเป็นที่มาของโครงงานนี้โดยมีวัตถุประสงค์ที่จะเป็นแนวทางในการแก้ไขปัญหา การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ เพื่อให้ผู้พัฒนาและผู้จำหน่ายซอฟต์แวร์สามารถปกป้องผลประโยชน์ของตนจากการละเมิดสิทธิ์การใช้งานซอฟต์แวร์ นอกจากนี้แล้วยังช่วยส่งเสริมและสนับสนุนธุรกิจการจำหน่ายซอฟต์แวร์ผ่านทางระบบอินเตอร์เน็ตให้มีประสิทธิ์ภาพดียิ่งขึ้นอีกด้วย โดยโครงงานจะช่วยอำนวยความสะดวกในการสร้างซอฟต์แวร์ให้มีความสามารถทางด้านการจัดการการบริหารและควบคุมสิทธิ์ และช่วยอำนวยความสะดวกในการสร้างระบบบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

**1.3 ขอบเขตและแนวทางการดำเนินงานของโครงงาน**

ขอบเขตและแนวทางการดำเนินงานของโครงงานนี้ คือ เริ่มจากการศึกษาถึงระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่ใช้กันอยู่ในปัจจุบันว่าประกอบไปด้วย องค์ประกอบที่สำคัญอะไรบ้าง, ทฤษฎีและหลักการที่ใช้, ตัวอย่างซอฟต์แวร์ที่ทำหน้าบริหารและควบคุมสิทธิ์, ตัวอย่างซอฟต์แวร์ที่มีการจัดการทางด้านการบริหารและควบคุมสิทธิ์, การแก้ไขซอฟต์แวร์ที่มีการจัดการทางด้านการบริหารและควบคุมสิทธิ์ให้สามารถละเมิดสิทธิ์ได้, แนวทางป้องกันการแก้ไขซอฟต์แวร์, จุดอ่อนของการบริหารและควบคุมสิทธิ์ในรูปแบบต่างๆ เมื่อได้ศึกษาหาความรู้ตามหัวข้อต่างๆ ข้างต้นเสร็จเรียบร้อยแล้ว การดำเนินงานขั้นต่อไปของโครงงานก็คือ การออกแบบระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่จะสามารถนำไปใช้งานได้จริง โดยวิเคราะห์ถึงความเหมาะสมในการนำไปประยุกต์ใช้งานและข้อจำกัดของเวลาที่มีอยู่ในการทำโครงงาน จากนั้นจึงกำหนดขอบเขตของผลงานและลงมือสร้าง โดยโครงงานนี้จะเน้นสร้าง คอมโพเนนท์ (Component) ที่จะสามารถนำไปประยุกต์ใช้งานได้ใน 2 ลักษณะคือ

- ช่วยอำนวยความสะดวกในการสร้างซอฟต์แวร์ให้มีความสามารถในการบริหารและควบคุมสิทธิ์การใช้งาน
- ช่วยอำนวยความสะดวกในการสร้างระบบบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

**1.4 เป้าหมายจากผลสำเร็จของโครงงานที่คาดว่าจะได้รับ**

- ทฤษฎี และหลักการในการพัฒนาระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์
- ความรู้เชิงเทคนิคและแนวทางในการสร้างพัฒนาระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์
- คอมโพเนนท์พื้นฐานสำหรับการสร้างระบบการบริหารและควบคุมสิทธิ์การใช้งานต้นแบบที่สามารถทำงานได้
- สามารถนำคอมโพเนนท์ที่สร้างขึ้นไปประยุกต์ใช้สร้างซอฟต์แวร์ให้มีความสามารถทางด้านการจัดการการบริหารและควบคุมสิทธิ์

# บทที่ 2

## ระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

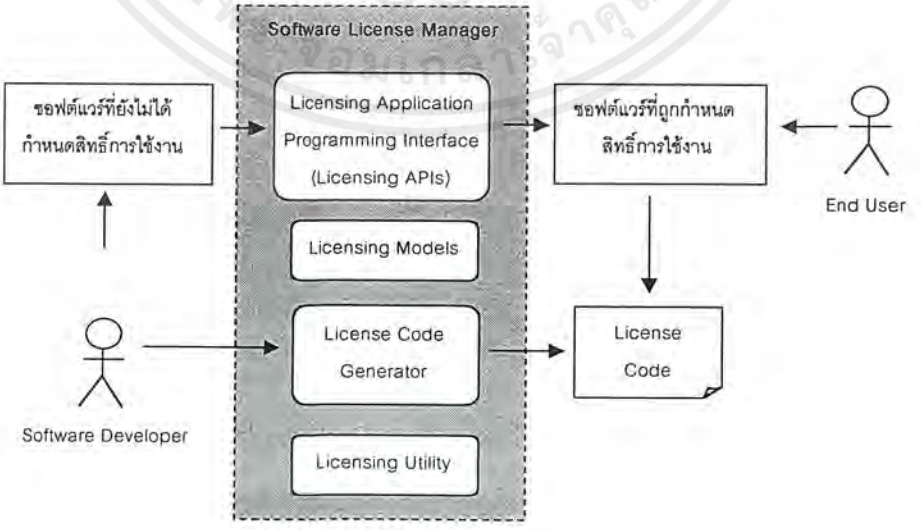
### 2.1 ความหมายของการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ หมายถึง การจัดการสิทธิ์การใช้งานซอฟต์แวร์ของผู้ใช้งาน โดยให้อนุญาตเฉพาะผู้ที่มีสิทธิ์ถูกต้องสามารถใช้งานซอฟต์แวร์ที่ผ่านการจัดการสิทธิ์เท่านั้น โดยขอบเขตของการใช้งานซอฟต์แวร์ของผู้มีสิทธิ์แต่ละคนจะแตกต่างกันไปขึ้นอยู่กับการกำหนดสิทธิ์ที่ได้ตกลงร่วมกันระหว่างผู้จำหน่ายซอฟต์แวร์ และผู้ใช้งานซอฟต์แวร์ซึ่งได้ระบุเอาไว้อย่างชัดเจนในเอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์

### 2.2 รูปแบบของการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

2.2.1 แบบเครื่องเดี่ยว (Stand-Alone Computer)

เป็นการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ ที่ใช้ควบคุมกลุ่มผู้ใช้งานคอมพิวเตอร์ส่วนบุคคล ที่ไม่ได้เชื่อมต่อกันเป็นระบบเน็ตเวิร์ก โดยการบริหารและควบคุมสิทธิ์ จะเริ่มจากผู้จำหน่ายซอฟต์แวร์นำซอฟต์แวร์ที่ลูกค้าต้องการใช้งานมาผ่านกระบวนการควบคุมสิทธิ์ในใช้งานโดยจะตกลงกับลูกค้าในเรื่องของขอบเขตของการใช้งานซอฟต์แวร์เพื่อที่จะนำข้อมูลดังกล่าวไปใช้ในการสร้างเอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์ (License Code) จากนั้นจึงส่งซอฟต์แวร์ดังกล่าวพร้อมเอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์ไปให้กับลูกค้า หลังจากที่ลูกค้าได้ติดตั้งซอฟต์แวร์แล้ว และเรียกใช้งาน ส่วนหนึ่งของซอฟต์แวร์ที่ทำหน้าที่ในการตรวจสอบสิทธิ์การใช้งานจะทำการตรวจสอบซอฟต์แวร์ว่าสามารถอนุญาตให้ผู้ใช้งานใช้งานซอฟต์แวร์ดังกล่าวได้หรือไม่ในระดับไหนซึ่งขึ้นอยู่กับขอบเขตของการใช้งานซอฟต์แวร์ซึ่งได้ระบุเอาไว้ในเอกสารแสดงสิทธิ์ [6]




*รูปที่ 2-1 แสดงรูปแบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์แบบเครื่องเดี่ยว*

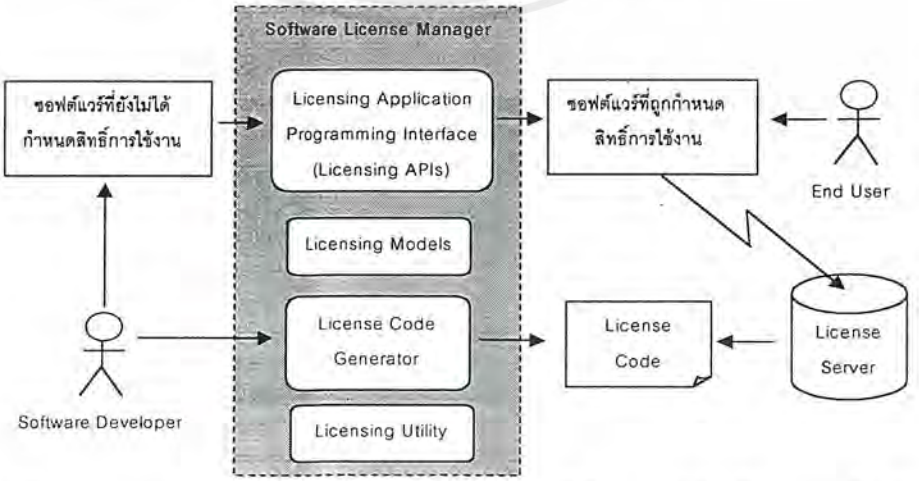
ตัวอย่างการทำงานของระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์แบบเครื่องเดี่ยว เช่น เมื่อมีผู้จำหน่ายซอฟต์แวร์รายหนึ่งต้องการที่จะขายซอฟต์แวร์ประมวลผลคำ ซึ่งซอฟต์แวร์ดังกล่าวประกอบไปด้วยคุณสมบัติหลัก (Feature) ในการใช้งานอยู่ 3 ส่วนคือ

- ส่วนประมวลผลคำ (Word processing)
- ส่วนตรวจสอบการสะกดคำ (Spell checking)
- ส่วนหาพจนานุกรมคำพ้อง/คำตรงข้าม (Thesaurus)

ผู้จำหน่ายซอฟต์แวร์จะนำซอฟต์แวร์ดังกล่าวไปผ่านกระบวนการควบคุมสิทธิ์การใช้งาน เมื่อมีลูกค้าต้องการซื้อซอฟต์แวร์ลูกค้าจะทำการตกลงกับผู้จำหน่ายในเรื่องของการเลือกคุณสมบัติที่ต้องการ สมมติว่า ลูกค้ารายนั้นตกลงที่จะเลือกซื้อใช้เฉพาะ ส่วนประมวลผลคำ และส่วนเช็คการสะกดคำ ต่อจากนั้นผู้จำหน่ายซอฟต์แวร์จะนำข้อมูลดังกล่าวไปสร้างเอกสารสิทธิ์เพื่อกำหนดสิทธิ์การใช้งานซอฟต์แวร์ แล้วทำการส่งซอฟต์แวร์พร้อมทั้งเอกสารสิทธิ์ไปให้กับลูกค้าและลูกค้าก็จะต้องชำระเงินให้กับผู้จำหน่าย เมื่อลูกค้าได้รับซอฟต์แวร์ ได้ติดตั้ง และเรียกใช้งาน ส่วนที่ทำหน้าที่ในการตรวจสอบสิทธิ์การใช้งานซอฟต์แวร์จะทำการตรวจสอบสิทธิ์และอนุญาตให้ลูกค้าสามารถใช้งานซอฟต์แวร์ได้เฉพาะ ส่วนประมวลผลคำ และส่วนการตรวจการสะกดคำ เมื่อลูกค้าเรียกใช้งาน ส่วนพจนานุกรมคำพ้องหรือคำตรงข้าม ส่วนตรวจสอบสิทธิ์จะไม่อนุญาตให้ลูกค้าใช้งานโดยจะแจ้งข้อความเตือนลูกค้าว่า ลูกค้าไม่มีเอกสารสิทธิ์ที่ถูกต้องที่จะอนุญาตให้ใช้งานซอฟต์แวร์ในส่วนนี้ได้

2.2.2 แบบเน็ตเวิร์ก (Network Computer)

เป็นการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ ที่ใช้ควบคุมกลุ่มผู้ใช้งานคอมพิวเตอร์ที่ต่อเชื่อมทำงานร่วมกันเป็นระบบเน็ตเวิร์ก เช่น ระบบคอมพิวเตอร์ในองค์กร หรือในสำนักงาน โดยการบริหารและควบคุมสิทธิ์ จะเริ่มจากการนำซอฟต์แวร์มาผ่านกระบวนการสร้างสิทธิ์เช่นเดียวกับการบริหารแบบแรก แต่ในระบบเน็ตเวิร์กจะมี เครื่องหลักที่ใช้ในการควบคุมการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ (License Server) ซึ่งจะทำหน้าที่ในการตรวจตรวจสอบสิทธิ์การใช้งานซอฟต์แวร์ของเครื่องย่อยในระบบเน็ตเวิร์ก ที่ได้มีการติดตั้งซอฟต์แวร์ที่ผ่านกระบวนการควบคุมสิทธิ์ [6]

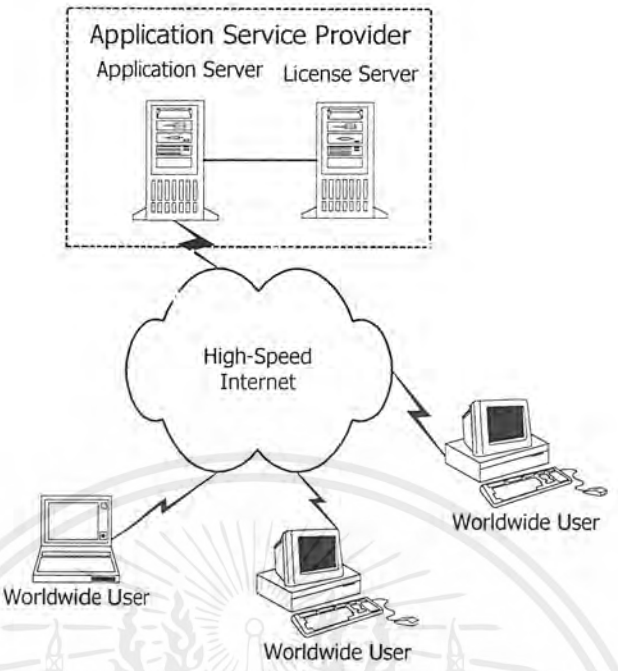



*รูปที่ 2-2 แสดงรูปแบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ในระบบเน็ตเวิร์ก*

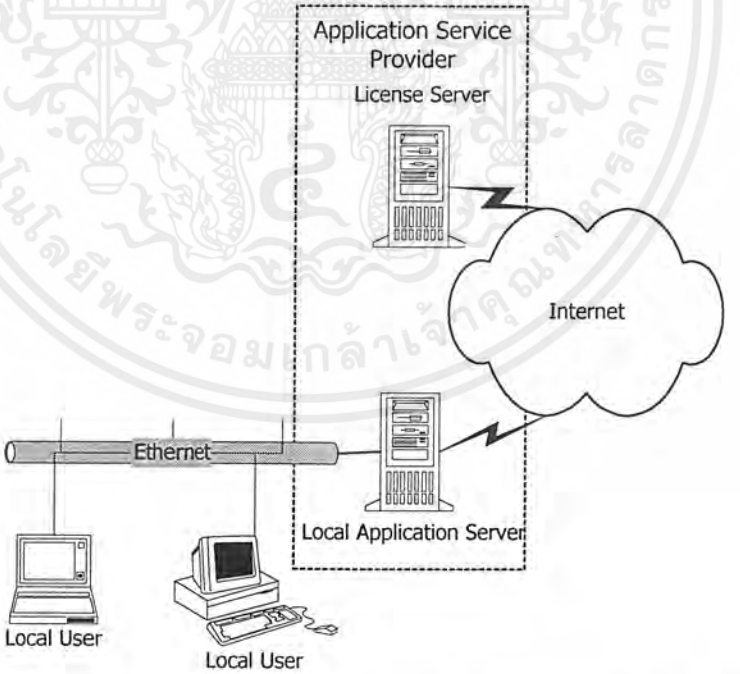
ตัวอย่างการใช้งานระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์แบบเน็ตเวิร์ก เมื่อลูกค้าต้องการใช้งานซอฟต์แวร์แบบที่มีระบบการบริหารและควบคุมสิทธิ์การใช้งานแบบเน็ตเวิร์ก ลูกค้าจะตกลงกับผู้จำหน่ายว่าจะต้องการให้มีการใช้งานซอฟต์แวร์ดังกล่าวกี่เครื่องใช้งานพร้อมๆ กันได้ใน 1 ระบบเน็ตเวิร์กจากนั้นผู้จำหน่ายซอฟต์แวร์ก็จะนำข้อมูลดังกล่าวไปสร้างเอกสารสิทธิ์เพื่อกำหนดสิทธิ์ในการใช้งาน แล้วส่งซอฟต์แวร์พร้อมเอกสารสิทธิ์ไปให้กับลูกค้า ทางด้านลูกค้าก็จะกำหนดให้เครื่องใดเครื่องหนึ่งในระบบเน็ตเวิร์กของลูกค้าใช้เป็นเครื่องหลักที่จะดูแลการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ โดยนำเอกสารสิทธิ์ติดตั้งไว้ที่เครื่องดังกล่าว จากนั้นลูกค้าจะนำซอฟต์แวร์ไปติดตั้งลงเครื่องต่างๆ ในระบบเน็ตเวิร์ก เมื่อเครื่องใดเครื่องหนึ่งที่ได้ติดตั้งซอฟต์แวร์เอาไว้แล้วแล้วต้องการใช้งาน ในตอนเรียกใช้งาน ซอฟต์แวร์จะติดต่อมาที่เครื่องหลักเพื่อขออนุญาตการใช้งานซอฟต์แวร์ สมมติว่าในระบบเน็ตเวิร์กนี้ได้ขอเอกสารสิทธิ์ให้มีการใช้งานซอฟต์แวร์ที่ซื้อมาได้ 10 เครื่องพร้อมๆ กัน ถ้าในระบบยังมีการใช้งานซอฟต์แวร์ที่ซื้อมาไม่ถึง 10 เครื่องที่มาขออนุญาตการใช้งานก็จะได้รับอนุญาตให้ใช้งานได้จนกว่าในระบบนั้นจะมีการใช้งานครบ 10 เครื่องแล้ว เครื่องที่ 11 มาขอใช้งาน เครื่องหลักที่ทำหน้าที่บริหารสิทธิ์จะไม่ยอมให้ใช้จนกว่าจะมีการเลิกใช้งานซอฟต์แวร์ดังกล่าวจากเครื่องที่มีการใช้งานอยู่ก่อนหน้านี้ในจำนวน 10 เครื่อง

2.2.3 แบบผ่านอินเตอร์เน็ต (Internet)

เมื่ออินเตอร์เน็ตมีบทบาทมากขึ้น การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ จึงเริ่มขยายขอบเขตการทำงานให้สามารถทำงานผ่านอินเตอร์เน็ตได้ ซึ่งการบริหารและควบคุมสิทธิ์ช่วยให้ผู้ให้บริการซอฟต์แวร์ผ่านระบบอินเตอร์เน็ตสามารถควบคุมสิทธิ์การใช้งานของผู้ใช้งานได้ทั่วโลก เนื่องจากให้บริการซอฟต์แวร์ผ่านทางระบบอินเตอร์เน็ต แสดงดังรูปที่ 2 – 3 จำเป็นต้องอาศัยช่องทางการสื่อสารที่มีความเร็วสูง ผู้ให้บริการซอฟต์แวร์ อาจจะติดตั้งซอฟต์แวร์ให้กับผู้ใช้งานในฝั่งของผู้ใช้ แล้วอาศัยการควบคุมสิทธิ์การใช้งานซอฟต์แวร์ผ่านระบบอินเตอร์เน็ตแทน ดังรูปที่ 2 – 4 ก็จะสามารถลดความต้องการของช่องทางการสื่อสารลงได้




*รูปที่ 2-3 แสดงระบบการให้บริการซอฟต์แวร์ผ่านอินเตอร์เน็ตซึ่งจำเป็นต้องมีระบบการบริหาร และควบคุมสิทธิ์การใช้งาน*




*รูปที่ 2-4 แสดงการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ผ่านระบบอินเตอร์เน็ต โดยวิธีการติดตั้งซอฟต์แวร์ในฝั่งผู้ใช้งาน*

**2.3 องค์ประกอบพื้นฐานของระบบบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์**

2.3.1 เอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์ (License Code)

เป็นเอกสารทางดิจิตอลที่สร้างขึ้นจากข้อตกลงร่วมกันระหว่างผู้จำหน่ายซอฟต์แวร์และผู้ใช้งานซอฟต์แวร์เพื่อที่จะกำหนดขอบเขตของการใช้งานซอฟต์แวร์ที่ผู้จำหน่ายอนุญาตให้ผู้ใช้งานสามารถใช้ได้ [6] เอกสารแสดงสิทธิ์โดยทั่วๆไปประกอบไปด้วยส่วนต่างๆ ดังนี้

- รหัสหรือสัญลักษณ์ที่ใช้แสดงถึงผู้จำหน่ายซอฟต์แวร์
- ชื่อคุณสมบัติแต่ละส่วนของซอฟต์แวร์
- รหัสหมายเลขของเครื่องคอมพิวเตอร์ (Digital Fingerprint) ที่จะติดตั้งซอฟต์แวร์ เช่น หมายเลขของอีเทอร์เน็ตการ์ด (Ethernet Address), ข้อมูลเฉพาะของฮาร์ดดิสก์ (Hard disk Identifier), หมายเลขลอจิคอลของดิสก์ (Logical Disk Serial Number), หมายเลขตัวประมวลผล (Processor Serial Number) หรือข้อมูลอื่นๆ ที่ค่อนข้างเฉพาะเจาะจงกับเครื่อง เพื่อนำไปใช้ประโยชน์ในการป้องกันซอฟต์แวร์ให้สามารถใช้งานได้เฉพาะเครื่องใดเครื่องหนึ่ง
- วันหมดอายุการใช้งานซอฟต์แวร์
- จำนวนของเครื่องที่สามารถใช้งานซอฟต์แวร์พร้อมๆ กันได้
- ช่วงระยะเวลาที่เครื่องซึ่งกำลังใช้งานซอฟต์แวร์จะติดต่อกลับไปหา เครื่องหลักที่ทำหน้าที่บริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์เพื่อยืนยันการใช้งาน (ในกรณีที่ระบบทำงานโดยมีเครื่องหลักคอยตรวจสอบสิทธิ์)

2.3.2 เครื่องมือสร้างเอกสารสิทธิ์ (Licensing Utilities)

เป็นโปรแกรม หรือ ฮาร์ดแวร์ที่ใช้ในการสร้างเอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์ทางดิจิตอล [6]

2.3.3 เครื่องมือตรวจเช็คความถูกต้องของเอกสารสิทธิ์

เป็นโปรแกรม หรือ ฮาร์ดแวร์ที่ใช้ในการตรวจสอบความถูกต้องของเอกสารสิทธิ์เพื่อที่จะตัดสินใจว่าซอฟต์แวร์ที่มาพร้อมกับเอกสารสิทธ์ดังกล่าวยังสามารถอนุญาตให้ผู้ใช้งานซอฟต์แวร์ใช้งานได้หรือไม่ โดยเครื่องมือที่ใช้ในการตรวจสอบจะทำการตรวจสอบโดยการถอดรหัสกำหนดสิทธิ์การใช้งาน หรืออาจจะทำโดยการตรวจสอบลายเซ็นต์ดิจิตอลที่กำกับมากับเอกสารแสดงสิทธิ์การใช้งาน [6]

2.3.4 เครื่องหลักที่ทำหน้าที่ในการบริหารสิทธิ์ใช้ในระบบเน็ตเวิร์ก

เป็นเครื่องตัวกลางที่ใช้ในการตรวจสอบเอกสารสิทธิ์ ซึ่งทำให้สามารถบริหารและควบคุมสิทธิ์การใช้งานภายในระบบเน็ตเวิร์กได้ [6]

2.3.5 ช่องทางในการส่งเอกสารสิทธิ์

เป็นช่องทางหรือวิธีที่จะใช้ในการส่งเอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์ไปให้กับผู้ใช้งาน เช่น การส่งไปทางระบบอินเตอร์เน็ต, การส่งไปพร้อมกับซอฟต์แวร์ เป็นต้น [6]

**2.4 แบบจำลองควบคุมสิทธิ์การใช้งานซอฟต์แวร์ (Licensing Models)**

แบบจำลองควบคุมสิทธิ์เป็นรูปแบบการควบคุมสิทธิ์การใช้งานซอฟต์แวร์ ซึ่งใช้ในการจำกัดขอบเขตการใช้งานของผู้ใช้งานซอฟต์แวร์ให้เป็นไปตามความต้องการผู้พัฒนาซอฟต์แวร์ โดยผู้พัฒนาอาจจะกำหนดแบบจำลองในการควบคุมสิทธิ์แบบพื้นฐาน หรือประกอบกันเป็นแบบจำลองที่มีความซับซ้อนขึ้น สำหรับแบบจำลองการควบคุมสิทธิ์พื้นฐานโดยทั่วไป [11], [12] มีดังต่อไปนี้

2.4.1 ซอฟต์แวร์ตัวอย่างให้ทดลองใช้งาน (Demo หรือ Evaluation licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ ที่จำกัดเวลาในการใช้ซอฟต์แวร์ เพื่อทดสอบความพึงพอใจของผู้ใช้ โดยอาจจะจำกัดวันเวลา เริ่มจากวันที่ลงโปรแกรม จนสิ้นสุดถึงวันหมดอายุ หรืออาจจะคิดจากจำนวนครั้งที่ใช้งานซอฟต์แวร์ การควบคุมสิทธิ์การใช้งานแบบนี้ไม่จำเป็นต้องเฉพาะเจาะจงกับเครื่องใดเครื่องหนึ่ง มักจะใช้ในกรณีที่ต้องการทดลองตลาด โดยแจกจ่ายให้ผู้ใช้สามารถทดลองใช้งานได้ทันทีโดยจะให้เอกสารแสดงสิทธิ์การใช้งานไปพร้อมกับซอฟต์แวร์

2.4.2 ควบคุมสิทธิ์ตามคุณสมบัติที่ต้องการใช้งาน (Feature-based licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ ที่จำกัดความสามารถในการใช้คุณสมบัติต่างๆ ที่มีอยู่ของซอฟต์แวร์ให้ผู้ใช้สามารถเลือกใช้เฉพาะคุณสมบัติที่ผู้ใช้ต้องการ ในตอนแรกที่ผู้ใช้ซื้อซอฟต์แวร์ไปจะได้รับซอฟต์แวร์ที่มีคุณสมบัติครบทั้งหมด โดยในแต่ละคุณสมบัติจะถูกควบคุมสิทธิ์แยกจากกัน ผู้ใช้จะสามารถใช้งานได้เฉพาะคุณสมบัติที่ระบุในเอกสารแสดงสิทธิ์การใช้งาน

2.4.3 คิดค่าบริการตามการใช้งาน (Pay-per-use licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ ที่คิดค่าบริการจากปริมาณการใช้งานซอฟต์แวร์ โดยสามารถวัดได้จากจำนวนเวลาทั้งหมดที่ใช้หน่วยประมวลผล ในการประมวลผลหรือเวลาที่ใช้งานซอฟต์แวร์นั้นๆ

2.4.4 ป้องกันให้ทำงานได้เฉพาะระบบ (Node-locked licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ อาจจะจำกัดจำนวนผู้ใช้หรือไม่จำกัดจำนวนผู้ใช้ก็ได้ อาจทำบนเครื่องคอมพิวเตอร์ เพียง 1 เครื่องหรือบนระบบเครือข่ายก็ได้ โดยซอฟต์แวร์ที่ถูกจัดการสิทธิ์จะใช้งานได้เฉพาะภายในเครื่องคอมพิวเตอร์ หรือ ระบบเครือข่ายดังกล่าวเท่านั้น ผู้ใช้ไม่สามารถติดตั้งซอฟต์แวร์บนระบบอื่นหรือเครื่องอื่น ที่ไม่ระบุในเอกสารแสดงสิทธิ์

2.4.5 ควบคุมตามผู้ใช้งาน (User-based licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ ที่จำกัดสิทธิในการใช้ซอฟต์แวร์ ให้สำหรับผู้ที่มีรหัสของผู้ใช้ที่ถูกต้องเท่านั้น โดยในการจะเข้างานซอฟต์แวร์ในแต่ละครั้งผู้ใช้จะต้องล็อกอินก่อน เพื่อทำการป้อนรหัสผ่าน ถ้าทุกอย่างถูกต้องทั้งหมดผู้ใช้ก็จะได้รับอนุญาตให้ใช้งานซอฟต์แวร์ได้

2.4.6 ควบคุมการใช้งานบนไซต์ (Site licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ ที่จำกัดขอบเขตของการใช้ซอฟต์แวร์ ให้อยู่ภายในขอบเขตที่กำหนดไว้เท่านั้น เช่น ภายในบริษัท, หน่วยงาน เป็นต้น

2.4.7 ควบคุมการใช้งานบนเน็ตเวิร์ก (Floating licensing)

เป็นการจัดการสิทธิ์ในการใช้งานซอฟต์แวร์ ที่จำกัดจำนวนผู้ใช้ ซอฟต์แวร์ในขณะหนึ่งขณะใด ในระบบเครือข่ายเดียวกัน กล่าวคือ ซอฟต์แวร์จะถูกใช้โดยเครื่องใดก็ได้บนระบบเครือข่ายนั้น แต่ในขณะใดขณะหนึ่งต้องมีจำนวนผู้ใช้เท่ากับจำนวนผู้ใช้ที่ได้ระบุไว้ในเอกสารแสดงสิทธิ์การใช้งาน

2.4.8 จำกัดเวลาการใช้งาน (Time Limited)

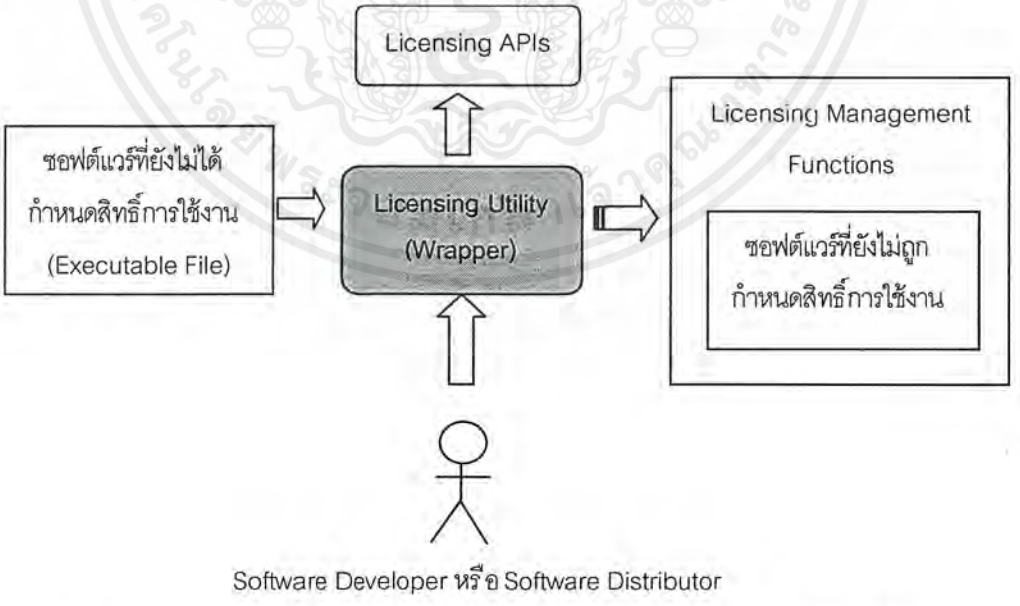
แบบนี้จะกำหนดราคาซอฟต์แวร์ ตามช่วงระยะเวลาการใช้งานซอฟต์แวร์ โดยผู้ใช้สามารถกำหนดระยะเวลาที่ต้องการใช้งานซอฟต์แวร์นี้ แบบจำลองแบบนี้จะเป็นเหมือนการเช่าใช้งานชั่วคราว เมื่อหมดอายุการใช้งาน ผู้ใช้สามารถขอเอกสารแสดงสิทธิ์ใหม่เพื่อต่ออายุการใช้งานได้

**2.5 การปกป้องซอฟต์แวร์ให้สามารถควบคุมสิทธิ์การใช้งานได้**

วิธีการที่นิยมใช้ในการปกป้องซอฟต์แวร์ให้สามารถควบคุมสิทธิ์ได้นั้นสามารถแบ่งออกเป็น 3 วิธี ดังนี้คือ

2.5.1 การป้องกันซอฟต์แวร์ด้วยตัวครอบซอฟต์แวร์ (Wrapper)

เป็นการนำเอาไฟล์ที่สามารถเอ็กซีคิวได้ (Executable file) ซึ่งได้แก่ไฟล์โปรแกรมหรือไดนามิกไลบรารี มาทำการครอบ โดยภายในตัวครอบจะประกอบไปด้วยฟังก์ชันที่ทำหน้าที่ในการจัดการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ ในการทำงานเริ่มต้นเมื่อผู้ใช้เรียกใช้งานซอฟต์แวร์ที่ถูกปกป้องด้วยวิธีนี้ ตัวครอบจะทำงานก่อนโดยตรวจสอบสิทธิ์การใช้งานของซอฟต์แวร์ว่าถูกต้องหรือไม่ ถ้าผ่านการตรวจสอบว่าถูกต้องแล้ว ขั้นตอนต่อไปตัวครอบจะเรียกซอฟต์แวร์ที่ถูกครอบขึ้นมาใช้งาน ในการปกป้องซอฟต์แวร์ด้วยวิธีนี้ ผู้พัฒนาซอฟต์แวร์ไม่จำเป็นต้องแก้โค้ดโปรแกรม [10] แสดงดังรูปที่ 2 - 5



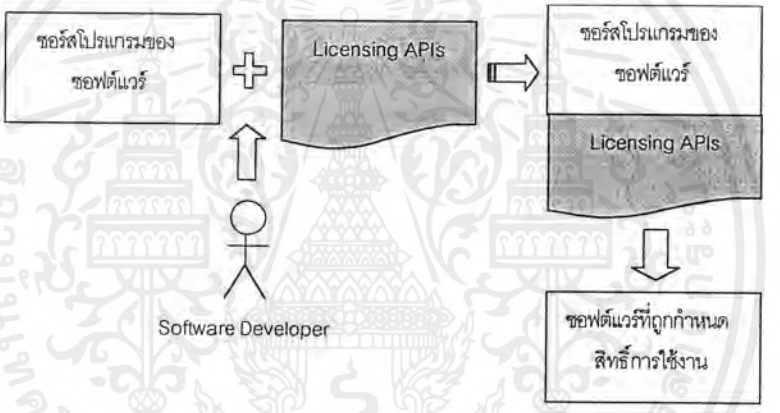

*รูปที่ 2-5 แสดงรูปแบบการครอบซอฟต์แวร์โดยใช้เครื่องมือควบคุมสิทธิ์การใช้งาน*

2.5.2 การปกป้องซอฟต์แวร์โดยการใช้คอมโพเนนท์ควบคุมสิทธิ์

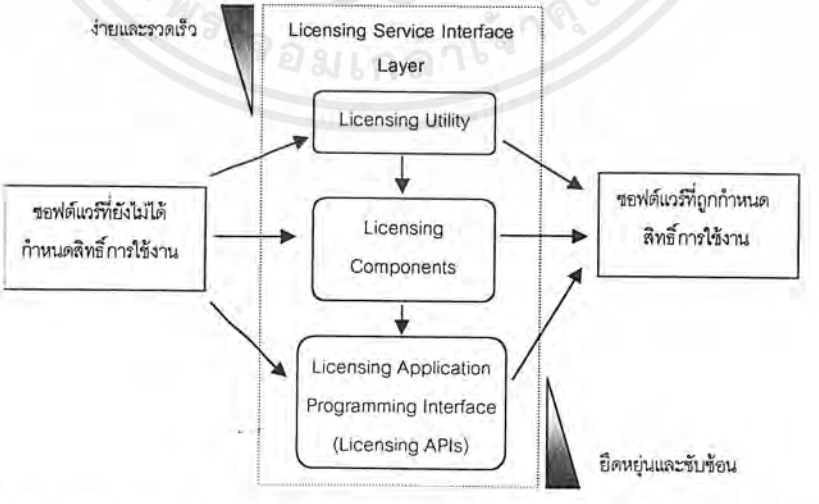
ปัจจุบันการเขียนโปรแกรมเชิงคอมโพเนนท์เริ่มเข้ามามีบทบาทในการพัฒนาซอฟต์แวร์ การป้องกันซอฟต์แวร์โดยวิธีนี้จึงอำนวยความสะดวกแก่ผู้พัฒนาซอฟต์แวร์ในการควบคุมสิทธิ์ โดยไม่จำเป็นต้องศึกษารายละเอียดของฟังก์ชันควบคุมสิทธิ์การใช้งานระดับล่าง เพียงแต่ติดต่อผ่านคอมโพเนนท์ควบคุมสิทธิ์ อย่างไรก็ตามวิธีนี้จะต้องทำโดยผู้พัฒนาซอฟต์แวร์ เนื่องจากต้องมีการแก้ไขซอร์สโปรแกรมของซอฟต์แวร์ นอกจากนี้การป้องกันในระดับนี้จะสามารถทำได้เฉพาะภาษาพัฒนาซอฟต์แวร์ที่สนับสนุน. คอมโพเนนท์

2.5.3 การพัฒนาซอฟต์แวร์โดยเรียกใช้งานฟังก์ชันในการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

เป็นการสร้างซอฟต์แวร์ให้มีความสามารถในการบริหารและควบคุมสิทธิ์โดยตัวมันเอง ทำในขั้นตอนการพัฒนา กำหนดให้มีการเรียกใช้ฟังก์ชันในการบริหารและควบคุมสิทธิ์จากไลบรารี(Library) ที่มีผู้พัฒนาเอาไว้แล้ว ซึ่งได้เตรียมฟังก์ชันที่ใช้ในการบริหารและควบคุมสิทธิ์การใช้งาน[7]




*รูปที่ 2-6 แสดงการสร้างควบคุมสิทธิ์การใช้งานซอฟต์แวร์ โดยการเขียนโปรแกรมติดต่อกับฟังก์ชันควบคุมสิทธิ์โดยตรง*




*รูปที่ 2-7 แสดงการระดับต่างๆ ในการป้องกันซอฟต์แวร์ที่สามารถติดต่อ กับซอฟต์แวร์ควบคุมสิทธิ์การใช้งาน*

จากรูปที่ 2 – 7 แสดงความสามารถในการป้องกันและควบคุมสิทธิ์ซอฟต์แวร์โดยการเรียกใช้บริการในการควบคุมสิทธิ์ของซอฟต์แวร์บริหารและควบคุมสิทธิ์ในระดับต่างๆ กัน เพื่อให้ยืดหยุ่นต่อการใช้งาน กล่าวคือ ในระดับยูทิลิตี้ การป้องกันซอฟต์แวร์สามารถทำให้ง่ายและรวดเร็ว แต่อาจจะไม่สามารถป้องกันและควบคุมสิทธิ์ในการใช้งานในลักษณะที่ซับซ้อน เมื่อเทียบกับการติดต่อกับฟังก์ชันในระดับล่าง ส่วนการทำงานในระดับคอมโพเนนท์ จะอำนวยความสะดวกแก่ผู้พัฒนาซอฟต์แวร์ที่ต้องการป้องกันและควบคุมสิทธิ์ทำได้ให้สะดวกและรวดเร็วขึ้น โดยไม่จำเป็นต้องติดต่อผ่านฟังก์ชันในระดับล่างโดยตรง

**2.6 การเลือกแบบจำลองสิทธิ์การใช้งาน**

การเลือกแบบจำลองสิทธิ์การใช้งานซอฟต์แวร์ สามารถทำได้ 3 ช่วงในกระบวนการป้องกันซอฟต์แวร์ [6] คือ

- เลือกระหว่างพัฒนาซอฟต์แวร์ (Development) เช่น Feature-based
- เลือกระหว่างการสร้างเอกสารแสดงสิทธิ์การใช้งาน (License Code Generate) เช่น Evaluation, Pay-per-use, User-based เป็นต้น
- เลือกโดยผู้ใช้งานระบบ การควบคุมสิทธิ์ในขั้นตอนนี้ ผู้ใช้ที่มีหน้าที่ในการดูแลระบบจะเป็นคนควบคุมสิทธิ์การใช้งาน โดยอาจจะกำหนดกลุ่มของผู้ใช้ เพื่อระบุว่ากลุ่มใดมีสิทธิ์ใช้หรือไม่มีสิทธิ์ใช้ซอฟต์แวร์ตัวใดบ้าง

| รูปแบบ | คำอธิบาย | S/N* | ช่วงที่ทำ |
|---|---|---|---|
| Evaluation | สิทธิ์การใช้งานซอฟต์แวร์จะหมดตามกำหนดเวลาที่แน่นอนหรือตามจำนวนครั้งที่ชัดเจน | S/N | License code |
| Feature-based | จำกัดสิทธิ์การใช้งานคุณลักษณะต่างๆ ของซอฟต์แวร์ | S/N | Development |
| Pay-per-use | คิดค่าบริการการใช้งานซอฟต์แวร์ตามปริมาณการใช้งาน | S/N | License code |
| Node-locked | จำกัดสิทธิ์การใช้งานซอฟต์แวร์ภายในระบบคอมพิวเตอร์ระบบหนึ่ง | S/N | License code |
| User-based | ให้สิทธิ์การใช้งานซอฟต์แวร์กับผู้ใช้ที่มีรหัสผ่านเท่านั้น | N | License code |
| Site | ให้สิทธิ์การใช้งานซอฟต์แวร์กับผู้ใช้ภายในบริเวณบริเวณหนึ่ง เช่น ภายในซับเน็ต | N | License code |
| Floating | ให้สิทธิ์การใช้งานซอฟต์แวร์กับผู้ใช้ภายในระบบเน็ตเวิร์ก | N | License code |

*S/N :- Stand-alone/Network

*ตารางที่ 2-1 แสดงช่วงที่สามารถเลือกแบบจำลองการควบคุมสิทธิ์สำหรับแบบจำลองควบคุมสิทธิ์*

**2.7 องค์ประกอบที่ใช้ในการพิจารณาระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์**

ปัจจัยที่ใช้ในการพิจารณาเลือกระบบบริหารและควบคุมสิทธิ์ [6] มีดังต่อไปนี้

2.7.1 ความหลากหลายของแบบจำลองการควบคุมสิทธิ์การใช้งาน (Licensing Models)

ระบบควบคุมสิทธิ์การใช้งานที่มีแบบจำลองในการควบคุมหลากหลายจะอำนวยความสะดวกในการเลือกแบบจำลองที่เหมาะสมกับซอฟต์แวร์และกลุ่มของลูกค้า เนื่องจากซอฟต์แวร์บางตัวอาจจะมีกลุ่มลูกค้าเป้าหมายหลายระดับ

2.7.2 ความน่าเชื่อถือ (Reliability)

ความน่าเชื่อถือของการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์คือ การที่ผู้ใช้สามารถใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์ ได้ตามสิทธิ์ที่ลูกค้าพึงได้

2.7.3 ความปลอดภัย (Security)

การควบคุมสิทธิ์ที่มีความปลอดภัยจะทำให้ผู้พัฒนาซอฟต์แวร์สามารถควบคุมสิทธิ์การใช้งานซอฟต์แวร์ได้อย่างปลอดภัย กล่าวคือ ผู้ใช้ที่สิทธิ์เท่านั้นจึงจะใช้งานซอฟต์แวร์นั้นได้

2.7.4 ความยืดหยุ่นในการปรับขนาด (Scalability)

เป็นธรรมดาที่ซอฟต์แวร์อาจจะถูกนำไปติดตั้งในระบบที่มีขนาดแตกต่างกัน ระบบบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ จึงต้องสามารถปรับเปลี่ยนได้ง่าย เพื่อให้เหมาะสมกับขนาดของระบบที่ต้องการ

2.7.5 ความเข้ากันได้กับซอฟต์แวร์ที่ต้องการควบคุมสิทธิ์ (Compatibility)

บอกถึงความยืดหยุ่นของการพัฒนาซอฟต์แวร์ให้สามารถควบคุมสิทธิ์การใช้งาน กล่าวคือ ระบบบริหารและควบคุมสิทธิ์สามารถสนับสนุนภาษาซอฟต์แวร์ใดบ้าง

2.7.6 ความสะดวกและรวดเร็วในการพัฒนาซอฟต์แวร์

เป็นปัจจัยที่ผู้พัฒนาสามารถพัฒนาซอฟต์แวร์ของตนให้มีความสามารถในการควบคุมสิทธิ์การใช้งานได้โดยง่ายและรวดเร็ว

2.7.7 ผลกระทบของระบบต่อประสิทธิภาพของซอฟต์แวร์ที่ถูกควบคุมสิทธิ์

เพื่อเพิ่มความปลอดภัยให้กับระบบโดยการป้องกันการแก้ไขดัดแปลงซอฟต์แวร์ เช่น การเข้ารหัส จะมีผลกระทบต่อประสิทธิภาพการทำงานของซอฟต์แวร์เดิม ดังนั้น ระบบการบริหารและควบคุมสิทธิ์ที่ดีจึงควรจะออกแบบฟังก์ชันในการป้องกันต่างๆ ให้มีประสิทธิภาพสูง เพื่อไม่ให้มีผลกระทบต่อประสิทธิภาพของซอฟต์แวร์เดิมมากนัก

**2.8 การบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ในปัจจุบัน**

ผลิตภัณฑ์ในการบริหารและควบคุมสิทธิ์ที่มีชื่อเสียงและมีผู้นิยมใช้ผลิตภัณฑ์หนึ่งก็คือ SentinelLM License Manager ของบริษัท Rainbow Technology [9], [10] ผลิตภัณฑ์ตัวนี้ทำหน้าที่ในการปกป้องซอฟต์แวร์จากการละเมิดสิทธิ์การใช้งาน และยังอำนวยความสะดวกในการทำธุรกิจอิเล็กทรอนิกส์เพื่อจำหน่ายซอฟต์แวร์ทางอินเตอร์เน็ตอีกด้วย โดยจะอำนวยความสะดวกในการแจกจ่ายซอฟต์แวร์ และการกำหนดสิทธิ์การใช้งานซอฟต์แวร์ นอกจากนี้ยังมีทางเลือกให้ลูกค้าในการใช้งาน อุปกรณ์กำหนดสิทธิ์

อัตโนมัติ (Client Activator) ซึ่งจะทำให้ลูกค้าสามารถกำหนดสิทธิ์ผ่านทางระบบอินเตอร์เน็ตได้โดยสะดวก และยังช่วยขยายตลาดให้กับผู้จำหน่ายซอฟต์แวร์โดยมีคุณสมบัติในการกำหนดสิทธิ์การใช้งานซอฟต์แวร์เป็นแบบทดลองใช้ (Evaluation) ให้ผู้ที่สนใจซอฟต์แวร์นำไปทดลองใช้งานดูก่อนเมื่อพอใช้แล้วจึงค่อยตัดสินใจซื้อที่หลัง โดยการกำหนดสิทธิ์รูปแบบดังกล่าวจะจำกัดเวลาในการใช้งานซอฟต์แวร์ให้สามารถใช้งานได้ภายในระยะเวลาหนึ่งเท่านั้น เอกสารแสดงสิทธิ์หนึ่งเอกสารสามารถเปลี่ยนจากแบบทดลองใช้ไปเป็นแบบซื้อไปใช้งานถาวรโดย ลูกค้าไม่จำเป็นต้องทำการติดตั้งซอฟต์แวร์ใหม่ สำหรับการบริหารและการจัดการสิทธิ์แบบเน็ตเวิร์ก ก็ยังอำนวยความสะดวกให้สามารถจัดการซอฟต์แวร์ให้เป็นส่วนกลาง (Centralize) พร้อมสำหรับการนำไปประยุกต์ใช้ในระบบธุรกิจและองค์กร โดยมีอุปกรณ์ช่วย (Tools) ต่างๆ มากมาย

SentinelLM License Manager สามารถใช้ในการกำหนดสิทธิ์รูปแบบต่างๆ ที่นิยมใช้ [6] มีดังนี้คือ

- แบบใช้งานพร้อมกัน (Concurrent License) เป็นรูปการกำหนดสิทธิ์ที่ใช้กันในระบบเน็ตเวิร์กเพื่อจำกัดจำนวนผู้ใช้ในงานซอฟต์แวร์ที่ปกป้องให้สามารถใช้งานได้พร้อมกันตามจำนวนที่กำหนด
- แบบทดลองใช้งาน (Demo License) แบ่งออกเป็นอีก 2 ประเภทย่อยคือ แบบจำกัดคุณสมบัติการใช้งานและแบบจำกัดเวลาในการใช้งานวัตถุประสงค์ของการจัดการสิทธิ์แบบนี้ก็คือ ต้องการให้ลูกค้านำซอฟต์แวร์ไปทดลองใช้งานก่อนเพื่อทดสอบความพอใจก่อนที่จะตัดสินใจซื้อซอฟต์แวร์ ในแบบจำกัดคุณสมบัตินั้นซอฟต์แวร์ที่ให้ทดลองจะอนุญาตให้ลูกค้าใช้งานเฉพาะบางคุณสมบัติเท่านั้น ถ้าลูกค้าต้องการใช้งานคุณสมบัติที่ปิดไม่ให้ใช้ลูกค้าจะต้องทำการขอสิทธิ์การใช้งานแบบถาวรกล่าวคือซื้อซอฟต์แวร์ตัวกล่าว เพื่อที่จะสามารถใช้งานได้ครบทุกคุณสมบัติ อีกแบบหนึ่งคือแบบจำกัดเวลา ซอฟต์แวร์ที่ให้ทดลองจะอนุญาตให้ลูกค้าสามารถใช้งานซอฟต์แวร์ได้ภายในช่วงเวลาที่กำหนดเท่านั้น เมื่อซอฟต์แวร์หมดอายุการใช้งานแล้วหากลูกค้าต้องการจะใช้งานอีก ลูกค้าจะต้องทำการขอสิทธิ์การใช้งานแบบถาวร การจำกัดสิทธิ์แบบนี้มักจะไม่ป้องกันซอฟต์แวร์ไว้กับเครื่องใดเครื่องหนึ่งโดยเฉพาะ บางครั้งอาจจะเรียกการจำกัดสิทธิ์แบบนี้ว่าการจำกัดสิทธิ์แบบชั่วคราว เพราะผู้จำหน่ายซอฟต์แวร์บางรายจะอนุญาตให้ลูกค้าใช้งานซอฟต์แวร์ได้ครบทุกคุณสมบัติภายในช่วงเวลาที่กำหนด สำหรับในการกำหนดเวลาใช้งานนั้นสามารถ กำหนดได้ 2 วิธีคือ การกำหนดแบบสัมพัทธ์ (Relative Time-Limited) คือการกำหนดจำนวนวันที่อนุญาตให้ใช้งาน และการกำหนดแบบสัมบูรณ์ (Absolute Time-Limited) คือการกำหนดวันหมดอายุที่แน่นอน สำหรับการกำหนดแบบสัมบูรณ์นั้น SentinelLM ใช้เทคโนโลยีที่มีความสามารถในการตรวจสอบเมื่อ มีการลบซอฟต์แวร์ตัวเก่าที่หมดอายุออก แล้วจะติดตั้งซอฟต์แวร์ดังกล่าวลงไปใหม่ SentinelLM จะทำการตรวจและแจ้งให้ทราบพร้อมทั้งป้องกันไม่ให้มีการติดตั้งใหม่
- แบบใช้ร่วมกันหรือแบบผสม (Shared License) เป็นการกำหนดสิทธิ์ที่ใช้ในระบบเน็ตเวิร์กใช้กำหนดสิทธิ์ซอฟต์แวร์ที่มีหลายคุณสมบัติซึ่งใช้งานพร้อมๆ กัน ทำให้ไม่ต้องทำเอกสารสิทธิ์หลายๆ เอกสาร

# บทที่ 3

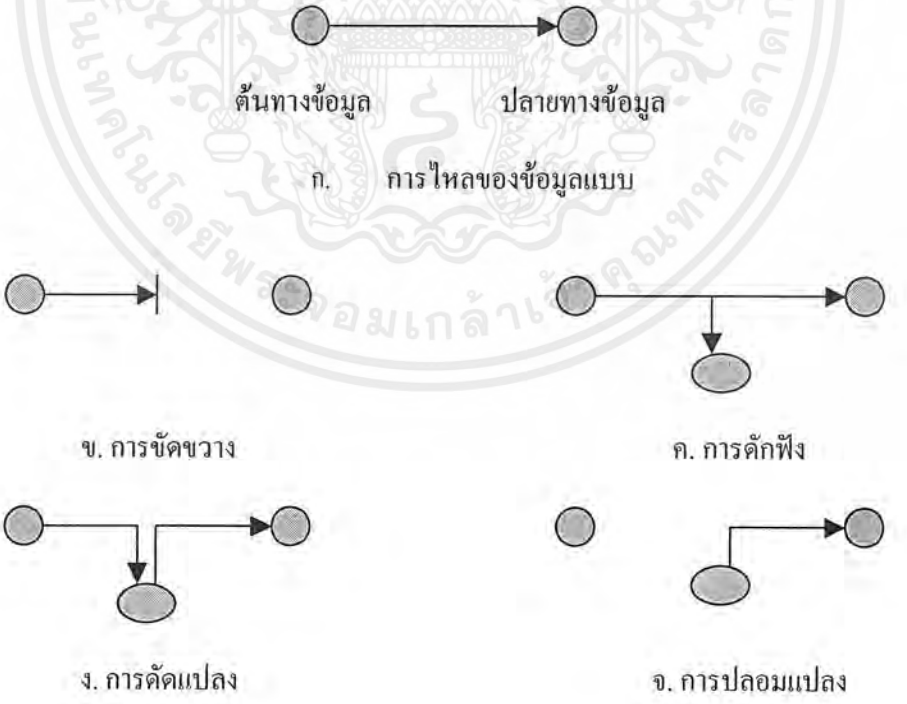
## การสร้างรหัสความปลอดภัยให้กับข้อมูลในระบบคอมพิวเตอร์ (Cryptography)

### 3.1 ภาพรวมของการสร้างรหัสความปลอดภัย

3.1.1 ปัญหาต่างๆ ที่เกิดกับข้อมูลในระบบคอมพิวเตอร์

ปัญหาต่างที่สามารถเกิดขึ้นระหว่างการส่งข้อมูลระหว่างเครื่องในระบบคอมพิวเตอร์ ทั้งภายในวงเน็ตเวิร์กเดียวกัน หรือระหว่างวงเน็ตเวิร์ก [4] สามารถแบ่งเป็นประเภทต่างๆ ได้ดังนี้คือ

- การขัดขวาง (Interruption) คือ การที่ข้อมูลที่จะส่งเกิดการสูญหาย สาเหตุอาจเกิดจากสภาพของฮาร์ดดิสก์ ที่ใช้เก็บข้อมูลเสื่อม, เส้นทางติดต่อสื่อสารที่ใช้ในการส่งข้อมูลขาดหรือเสียหาย
- การดักฟัง (Interception) คือ การที่ผู้ที่ไม่ได้รับอนุญาตให้รับรู้ข้อมูลทำการดักเอาข้อมูลดังกล่าวไปใช้ประโยชน์
- การดัดแปลง (Modification) คือ การที่ผู้ที่ไม่ได้รับอนุญาตให้รับรู้ข้อมูลทำการดักเอาข้อมูลดังกล่าวไปทำการดัดแปลงแก้ไขแล้วส่งข้อมูลนั้นต่อไปให้กับผู้รับ
- การปลอมแปลง (Fabrication) คือ การที่มีผู้แอบอ้างส่งข้อมูลปลอมไปให้กับผู้รับโดยอ้างว่ามาจากแหล่งอื่น




*รูปที่ 3-1 แสดงปัญหาต่างๆ ที่อาจเกิดกับข้อมูลในระบบคอมพิวเตอร์*

3.1.2 รูปแบบในการแก้ไขปัญหา

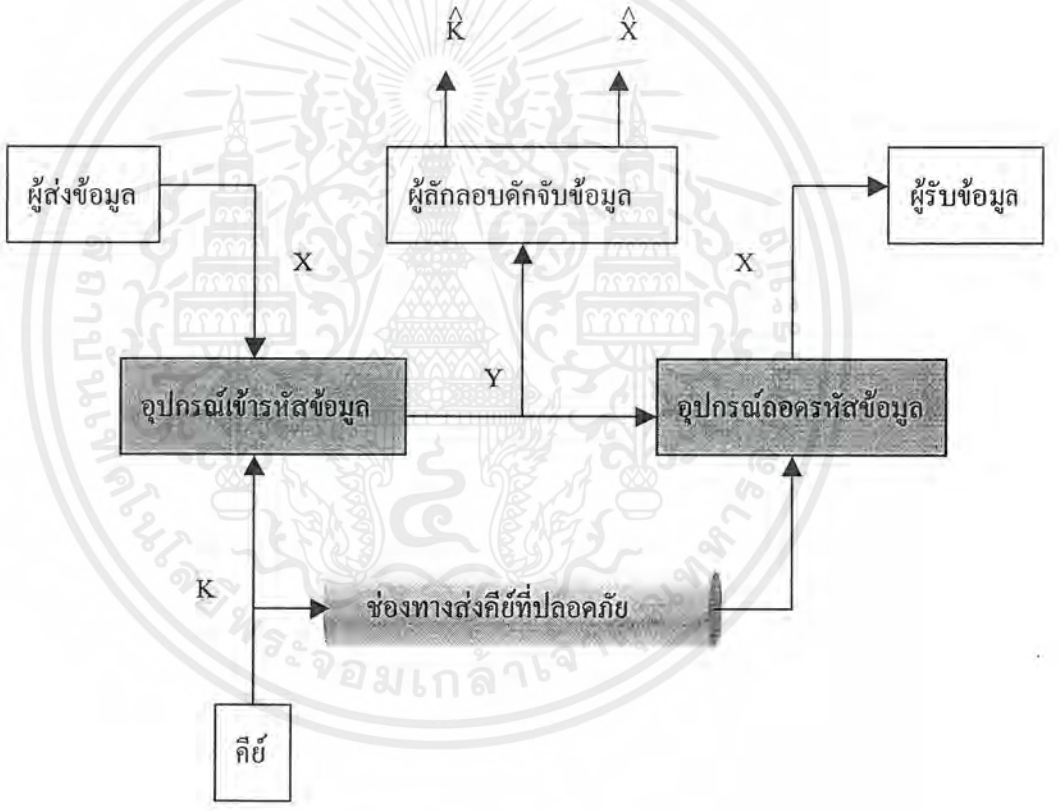
- การเข้ารหัสและถอดรหัสข้อมูล (Data Encryption/Data Decryption)

การเข้ารหัสข้อมูล หมายถึง การทำให้ข้อมูลที่อยู่ในรูปแบบปกติสามารถอ่านเพื่อตีความหมายได้ (Plaintext) แปลงไปเป็นข้อมูลสุ่มที่ไร้ความหมายไม่สามารถอ่านเพื่อตีความหมายได้ (Cipher text) [4]

การถอดรหัสข้อมูล หมายถึง การทำให้ข้อมูลสุ่มที่ไร้ความหมายไม่สามารถอ่านเพื่อตีความหมายได้ ซึ่งได้จากการเข้ารหัสข้อมูล แปลงกลับเป็นข้อมูลในรูปแบบปกติสามารถอ่านเพื่อตีความหมายได้ [4]

- การเข้ารหัสและถอดรหัสแบบสมมาตร (Symmetric Data Encryption)

การเข้ารหัสและถอดรหัสแบบสมมาตร เป็นการเข้ารหัสและถอดรหัสที่ใช้คีย์ (Key) ในการเข้ารหัสและถอดรหัสตัวเดียวกัน ดังนั้น ความปลอดภัยของการเข้ารหัสและถอดรหัสแบบนี้จึงขึ้นอยู่กับประสิทธิภาพการทำงานของ อัลกอริธึมที่ใช้และความลับในการเก็บคีย์ [4]




*รูปที่ 3-2 แสดงการทำงานของการเข้ารหัสและถอดรหัสแบบสมมาตร*

อัลกอริธึมที่ใช้จะต้องมีประสิทธิภาพดีเพียงพอที่จะทำให้ ผู้ลักลอบดักจับข้อมูลไม่สามารถถอดรหัสจากข้อมูลสุ่มที่ไร้ความหมาย มาเป็นข้อมูลปกติได้แม้จากทราบรายละเอียดของอัลกอริธึมในการเข้ารหัสนั้นก็ตาม เนื่องจากอัลกอริธึมที่เป็นมาตราฐานมักจะถูกตีพิมพ์เผยแพร่อย่างเปิดเผย จากคุณสมบัตินี้จึงทำให้มีการผลิตชิป (Chip) ที่ใช้สำหรับการเข้ารหัสและถอดรหัสขึ้นใช้โดยเฉพาะ อัลกอริธึมที่เป็นที่นิยมใช้กันได้แก่ อัลกอริธึมการเข้ารหัสถอดรหัสมาตรฐาน (The Data Encryption Standard: DES)

จากรูปที่ 3-2 แสดงให้เห็นถึงลักษณะการทำงานของการเข้ารหัสและถอดรหัสแบบสมมาตร สามารถอธิบายได้ดังนี้ เมื่อผู้ส่งข้อมูลต้องการส่งข้อมูล $X = [X_1, X_2, X_3, ..., X_M]$ ไปให้กับผู้รับข้อมูลโดยใช้คีย์ $K = [K_1, K_2, K_3, ..., K_J]$ ในการเข้ารหัสข้อมูล คีย์ดังกล่าวอาจจะถูกสร้างจากฝั่งของผู้ส่งข้อมูลใช้ในการเข้ารหัสข้อมูลแล้วส่งไปให้ฝั่งของผู้รับข้อมูลโดยผ่านช่องทางส่งคีย์ที่ปลอดภัย เพื่อที่จะให้ผู้รับข้อมูลนำไปถอดรหัสข้อมูลที่ส่งไป อีกทางเลือกหนึ่งสำหรับการจัดการคีย์ก็คือ การจัดให้มีส่วนกลางที่เชื่อถือได้ ทำหน้าที่ในการสร้างคีย์ขึ้นมาแล้วส่งไปให้กับทั้งสองฝ่ายคือผู้รับและผู้ส่งข้อมูล

เมื่อฝั่งของผู้ส่ง ส่งข้อมูล X และคีย์ K ไปที่อุปกรณ์เข้ารหัสข้อมูล ผลลัพธ์ที่ได้จะเป็นข้อมูลสุ่มที่ไร้ความหมายไม่สามารถอ่านเพื่อตีความหมายได้ $Y = [Y_1, Y_2, Y_3, ..., Y_N]$ ตามสมการข้างล่าง

$$Y = E_K(X)$$

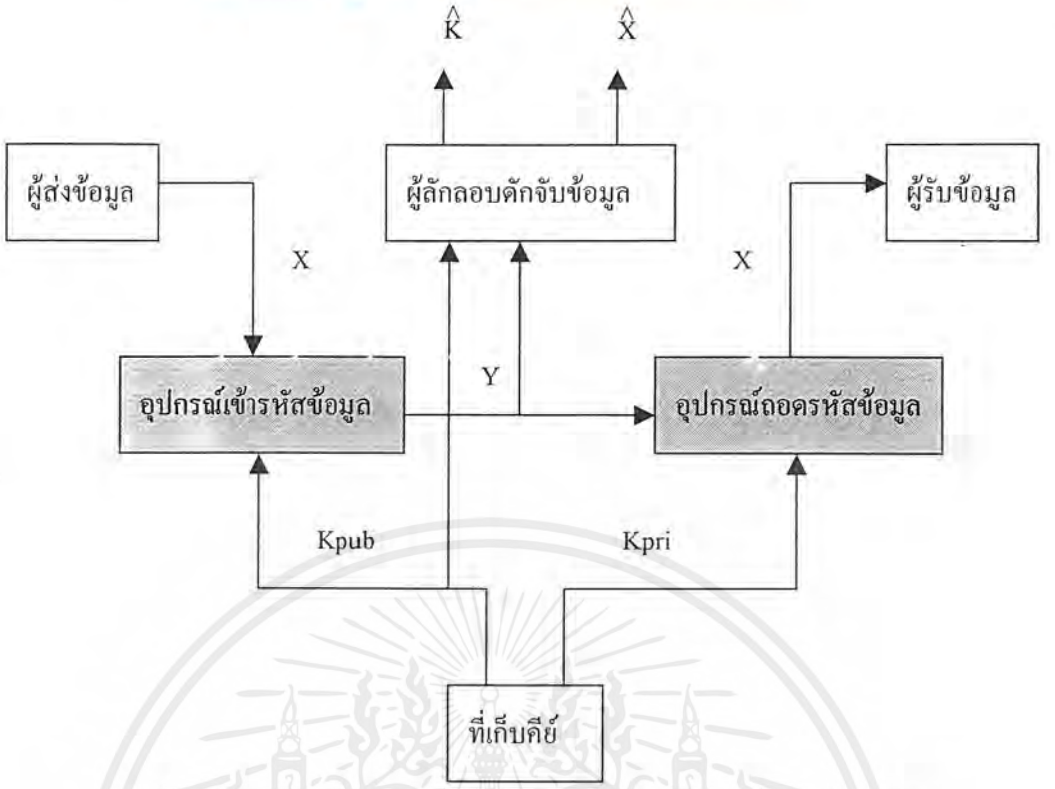
เมื่อฝั่งของผู้รับ รับข้อมูล Y และคีย์ K เข้าไปที่อุปกรณ์ถอดรหัสข้อมูล ผลลัพธ์ที่ได้จะเป็นข้อมูลปกติสามารถอ่านเพื่อตีความหมายได้ X ตามสมการข้างล่าง

$$X = D_K(Y)$$

ผู้ลักลอบดักจับข้อมูลสามารถดักจับข้อมูลที่ผ่านการเข้ารหัส Y ได้แต่อาจจะไม่สามารถดักจับคีย์ K ได้เพราะคีย์ดังกล่าวถูกส่งผ่านช่องทางส่งคีย์ที่ปลอดภัย ผู้ลักลอบอาจจะต้องการทราบค่าข้อมูล X จากข้อมูลสุ่ม Y หรือ อาจจะต้องการทราบค่าคีย์ K จากข้อมูลสุ่มเพื่อนำไปใช้ในการแอบลักลอบถอดรหัสข้อมูลครั้งต่อไป

- การเข้ารหัสและถอดรหัสแบบไม่สมมาตร (Asymmetric Data Encryption or Public-Key Encryption)

การเข้ารหัสและถอดรหัสแบบไม่สมมาตร เป็นการเข้ารหัสและถอดรหัสที่มีคีย์ใช้งานอยู่ 2 คีย์ซึ่งเกี่ยวข้องเป็นคู่กันคือ คีย์ส่วนตัว (Private Key) และ คีย์สาธารณะ (Public Key) โดยคีย์ส่วนตัวจะถูกเก็บไว้ที่เจ้าของคีย์เป็นความลับ ในขณะที่คีย์สาธารณะจะถูกเปิดเผยให้บุคคลภายนอกรับทราบ เมื่อคีย์ใดคีย์หนึ่งที่เป็นคู่กันถูกนำไปใช้ในการเข้ารหัสข้อมูล อีกคีย์หนึ่งจะถูกนำไปใช้ในการถอดรหัส ลักษณะการนำการเข้ารหัสและถอดรหัสแบบนี้ไปใช้งานคือ นำไปใช้ในการเข้ารหัสข้อมูลที่จะใช้ในการส่งไปให้กับผู้อื่นหรืออาจจะใช้สำหรับการสร้างและตรวจสอบลายเซ็นต์ดิจิตอลซึ่งจะกล่าวถึงในรายละเอียดในหัวข้อต่อไป คุณสมบัติที่สำคัญของการเข้ารหัสและถอดรหัสแบบนี้คือ เมื่อทราบค่าของคีย์ใดคีย์หนึ่ง พร้อมทั้งเข้าใจถึงอัลกอริธึมที่ใช้แล้ว จะยังคงไม่สามารถคำนวนหาค่าคีย์อีกคีย์หนึ่งที่เป็นคีย์ที่คู่กันได้ จึงทำให้การเข้ารหัสและถอดรหัสแบบนี้มีความปลอดภัยและน่าเชื่อถือมากยิ่งขึ้น อัลกอริธึมที่นิยมใช้กันอย่างแพร่หลาย เช่น อัลกอริธึมการเข้ารหัสถอดรหัส อาร์เอสเอ (The RSA Public-Key Encryption Algorithm) [13]




*รูปที่ 3-3 แสดงการส่งข้อมูลโดยใช้การเข้ารหัสและถอดรหัสแบบไม่สมมาตร*

รูปที่ 3-3 แสดงให้เห็นถึงลักษณะการทำงานในการส่งข้อมูลโดยใช้การเข้ารหัสและถอดรหัส. แบบไม่สมมาตร สามารถอธิบายได้ดังนี้ เมื่อผู้ส่งต้องการจะส่งข้อมูลไปให้กับผู้รับ ผู้ส่งจะต้องทราบคีย์สาธารณะ Kpub ของผู้รับ ดังนั้นในขั้นตอนแรกผู้รับจะต้องทำการสร้างคีย์คู่ คีย์ส่วนตัวและคีย์สาธารณะขึ้นมา 1 คู่เสียก่อน จากนั้นผู้รับจะต้องเก็บคีย์ส่วนตัว Kpri อันนี้เอาไว้ให้เป็นความลับ ส่วนคีย์สาธารณะ Kpub ผู้รับจะเผยแพร่ต่อสาธารณะเพื่อที่จะให้บุคคลอื่นนำคีย์ดังกล่าวไปใช้งานได้ เมื่อผู้ส่งทราบคีย์สาธารณะ Kpub ของผู้รับแล้ว ผู้ส่งก็พร้อมที่จะส่งข้อมูลไปให้ผู้รับ เมื่อผู้ส่งเตรียมข้อมูลที่จะส่ง X พร้อมแล้ว ผู้ส่งจะส่งข้อมูล X ดังกล่าวและคีย์สาธารณะ Kpub ของผู้รับไปยังอุปกรณ์เข้ารหัสได้ข้อมูลสุ่ม Y ส่งไปให้กับผู้รับดังสมการข้างล่าง

$$Y = E_{Kpub}(X)$$

เมื่อผู้รับได้รับก็จะสามารถถอดรหัสข้อมูลสุ่มกลับมาเป็นข้อมูลปกติโดยใช้คีย์ส่วนตัวของตัวเอง ดังสมการข้างล่าง

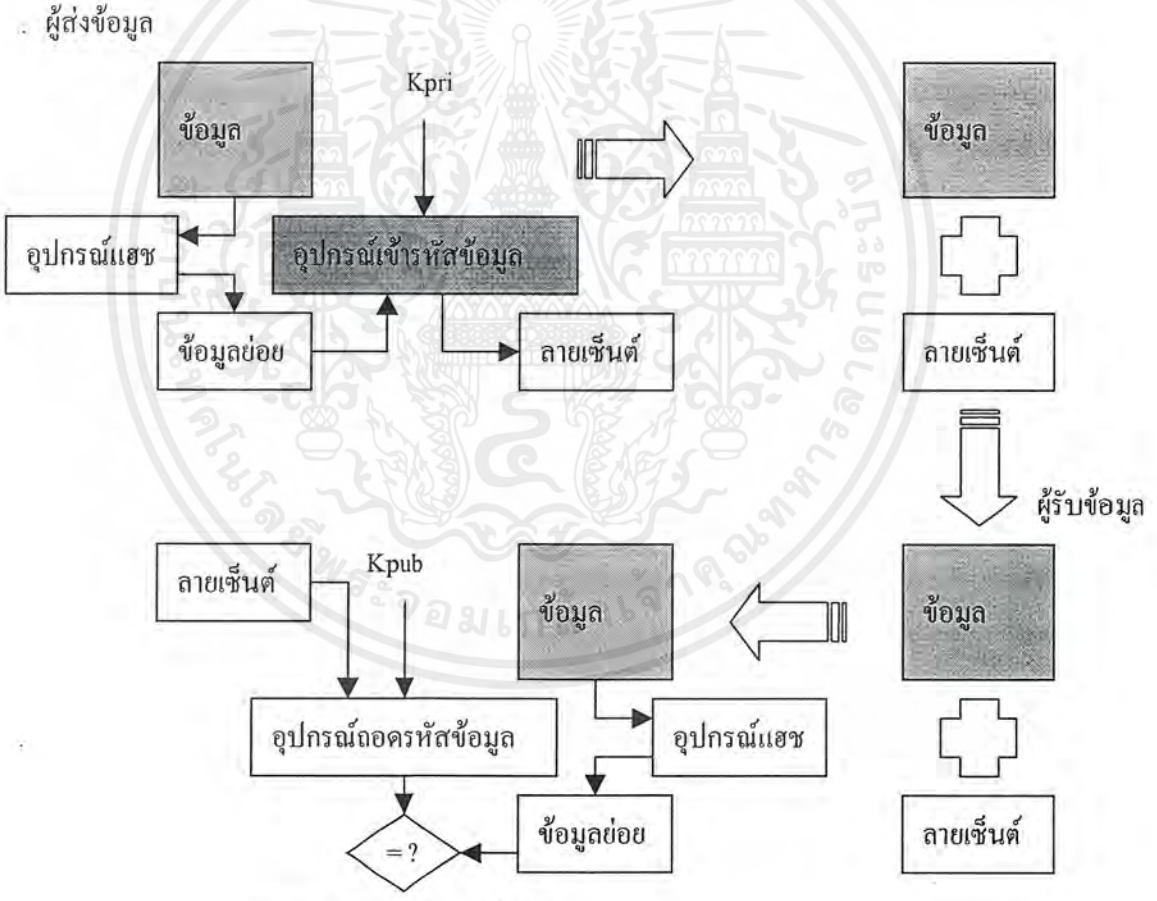
$$X = D_{Kpri}(Y)$$

ผู้ลักลอบดักจับข้อมูลสามารถดักจับข้อมูลที่ผ่านการเข้ารหัส Y และคีย์สาธารณะ Kpubได้แต่จะไม่สามารถคำนวณหาค่าคีย์ส่วนตัว Kpri เพื่อมาถอดรหัสข้อมูล Y ได้ถึงแม้จะทราบรายละเอียดของอัลกอริธึมที่ใช้ในการเข้ารหัสและถอดรหัสเป็นอย่างดีก็ตาม

- ลายเซ็นต์ดิจิตอล (Digital Signature)

ลายเซ็นต์ดิจิตอล หมายถึง รหัสข้อมูลทางดิจิตอลที่ใช้บ่งบอกตัวตนของเจ้าของลายเซ็นต์ (Authentication) ว่าเป็นเจ้าของข้อมูลที่แท้จริงที่ถูกกำกับโดยลายเซ็นต์นั้น โดยอาจจะส่งรวมเป็นไฟล์ (File) เดียวกับข้อมูล หรืออาจจะส่งเป็นไฟล์แยกกับข้อมูลก็ได้ [4]

- ○ แฮชชิ่ง (Hashing) แฮชชิ่งในความหมายเชิงการทำลายเซ็นต์ดิจิตอล หมายถึง การนำข้อมูลที่จะทำการเซ็นต์ลายเซ็นต์มาประมวลผลผ่านฟังก์ชันการประมวลผลให้ได้ ข้อมูลย่อย (Message Digest) ซึ่งมีขนาดที่เหมาะสมกับการนำไปทำลายเซ็นต์ดิจิตอล
- ○ การสร้างลายเซ็นต์ดิจิตอล




*รูปที่ 3-4 แสดงการสร้างและพิสูจน์ลายเซ็นต์ดิจิตอล*

จากรูปที่ 3-4 แสดงให้เห็นถึงขั้นตอนต่างๆในการสร้างและพิสูจน์ลายเซ็นต์ดิจิตอล ขั้นตอนการสร้างลายเซ็นต์มีดังนี้คือในขั้นแรกผู้ที่จะสร้างลายเซ็นต์ดิจิตอลจะต้องมีคู่คีย์ของ คีย์ส่วนตัวและคีย์

สาธารณะเสียก่อน จากนั้นจึงนำข้อมูลที่จะทำการเซ็นต์ลายเซ็นต์มาแฮชเพื่อให้ได้ข้อมูลย่อย แล้วนำข้อมูลย่อยดังกล่าวไปเซ็นต์ลายเซ็นต์ซึ่งก็คือการนำไปเข้ารหัสนั้นเองด้วยคีย์ส่วนตัวของตน สุดท้ายจึงแนบลายเซ็นต์ดังกล่าวไปพร้อมกับข้อมูลที่จะส่ง

สำหรับการพิสูจน์ลายเซ็นต์มีขั้นตอนดังนี้คือ เมื่อผู้รับรับข้อมูลที่มีลายเซ็นต์แนบมาด้วยแล้วผู้รับจากนำเฉพาะข้อมูลไปทำการแฮชให้ได้ข้อมูลย่อย จากนั้นนำข้อมูลย่อยดังกล่าวไปเปรียบเทียบกับ ค่าที่ได้จากการนำลายเซ็นต์ดิจิตอลมาถอดรหัสด้วยคีย์สาธารณะของผู้ส่งถ้าเท่ากันแสดงว่าข้อมูลดังกล่าวถูกส่งมาจากผู้ส่งที่เป็นเจ้าของคีย์สาธารณะที่ใช้ถอดรหัสจริง

**3.2 ฟังก์ชันของไมโครซอฟท์ที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูล (Microsoft Cryptographic Application Programming Interfaces)**

3.2.1 แบบจำลองของฟังก์ชันที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อม ของไมโครซอฟท์ (Microsoft CryptoAPI Programming Model)

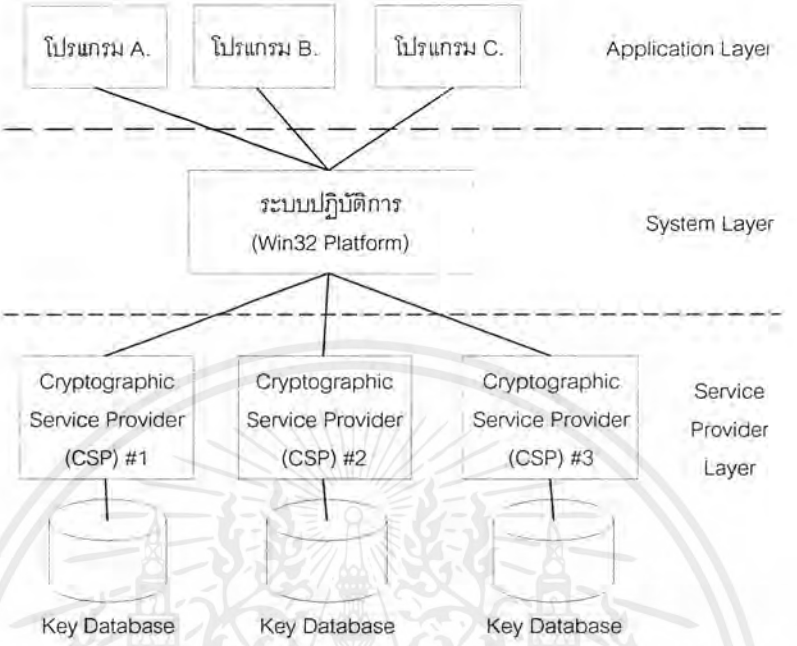
ฟังก์ชันของไมโครซอฟท์ที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูล (CryptoAPI) เป็นกลุ่มของฟังก์ชันซึ่งอำนวยความสะดวกให้แอพพลิเคชัน (Application) สามารถสร้างรหัสความปลอดภัยให้กับข้อมูล เช่น การเข้ารหัสข้อมูล, การสร้างลายเซ็นต์ดิจิตอล ได้อย่างยืดหยุ่นและรวดเร็ว ในขณะเดียวกันก็มีจัดเก็บคีย์ส่วนตัวไว้เป็นอย่างดี [14]

การทำงานในการสร้างรหัสความปลอดภัยทุกการทำงานจะกระทำอย่างเป็นอิสระต่อกัน คือเป็นโมดูล (Module) ที่แยกจากกัน ซึ่งเรียกว่า ส่วนให้บริการการสร้างรหัสความปลอดภัย (Cryptographic Service Providers (CSPs)) ตัวอย่างของส่วนให้บริการดังกล่าวได้แก่ ส่วนให้บริการการสร้างรหัสความปลอดภัยของไมโครซอฟท์ (Microsoft RSA Base Provider) ซึ่งถูกนำมารวมกับระบบปฏิบัติการของไมโครซอฟท์เองด้วย

ส่วนให้บริการการสร้างรหัสความปลอดภัยที่สร้างมาจากผู้ให้บริการที่ต่างกัน จะทำการจัดการ (Implement) ฟังก์ชันของไมโครซอฟท์ที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยแตกต่างกัน บางผู้ให้บริการใช้อัลกอริธึมที่มีความปลอดภัยสูง บางผู้ให้บริการจัดการโดยใช้ฮาร์ดแวร์เช่น สมาร์ตการ์ด (Smart card) บางผู้ให้บริการก็อาจจะติดต่อกับผู้ใช้โดยตรงเช่นการสร้างลายเซ็นต์ดิจิตอลโดยใช้คีย์ส่วนตัว

แบบจำลองการทำงานของฟังก์ชันของไมโครซอฟท์ที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูล สามารถเปรียบได้กับแบบจำลองของการติดต่อกับอุปกรณ์แสดงผลของวินโดว์ (Windows GDI Model) ซึ่งส่วนให้บริการการสร้างรหัสความปลอดภัย สามารถเปรียบได้กับ ตัวควบคุมการทำงานของอุปกรณ์กราฟฟิก (Graphics Device Drivers) และในกรณีที่ผู้ให้บริการการสร้างรหัสความปลอดภัยทำการอิมพลิเมนต์การจัดการสร้างรหัสความปลอดภัยโดยใช้ฮาร์ดแวร์ด้วย ส่วนที่เป็นฮาร์ดแวร์ดังกล่าวก็จะเปรียบได้กับ อุปกรณ์ฮาร์ดแวร์ของส่วนกราฟฟิกแสดงผล แอพพลิเคชันที่ดีจะไม่ยินยอมให้มีการติดต่อกับ ตัวควบคุมการทำงานของอุปกรณ์กราฟฟิกและอุปกรณ์ฮาร์ดแวร์ของส่วนกราฟฟิกแสดงผลโดยตรง เช่นเดียวกัน แอพพลิเคชันที่ดีก็จะ ไม่ยินยอมให้มีการติดต่อกับส่วนให้บริการการสร้าง

รหัสความปลอดภัยทั้งที่เป็นซอฟต์แวร์และฮาร์ดแวร์โดยตรง แต่ควรที่จะติดต่อโดยผ่านฟังก์ชันที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัย




*รูปที่ 3-5 แสดงแบบจำลองของฟังก์ชันที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูลของไมโครซอฟท์*

ระบบการสร้างรหัสความปลอดภัยให้กับข้อมูลของไมโครซอฟท์ประกอบไปด้วยส่วนต่างๆ ดังนี้คือ ส่วนแอพพลิเคชัน, ส่วนระบบปฏิบัติการ และส่วนบริการการสร้างรหัสความปลอดภัย

แอพพลิเคชันจะติดต่อกับระบบปฏิบัติการผ่านฟังก์ชันที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูลของไมโครซอฟท์ จากนั้นระบบปฏิบัติการจะติดต่อกับส่วนบริการการสร้างรหัสความปลอดภัยผ่านฟังก์ชันส่วนติดต่อกับส่วนบริการการสร้างรหัสความปลอดภัยของผู้สร้างส่วนบริการการสร้างรหัสความปลอดภัย (Cryptographic Service Provider Interface: CryptoSPI)

3.2.2 ฟังก์ชันที่สนับสนุนในการเขียนโปรแกรม

ฟังก์ชันของไมโครซอฟท์ที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูลสามารถแบ่งกลุ่มตามรูปแบบการใช้ได้ ดังนี้

- กลุ่มฟังก์ชันคอนเท็กซ์ (Context Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่ติดต่อกับส่วนที่ให้บริการการสร้างรหัสความปลอดภัยให้กับข้อมูลโดยสามารถเลือกส่วนที่ให้บริการดังกล่าวจากผู้ให้บริการต่างๆได้หลากหลาย ตามความเหมาะสมและระดับความปลอดภัยที่ต้องการ
- กลุ่มฟังก์ชันสร้างคีย์ (Key Generation Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่สร้างคีย์ที่จะใช้กับแอพพลิเคชันในการสร้างความปลอดภัย

- กลุ่มฟังก์ชันแลกเปลี่ยนคีย์ (Key Exchange Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่จัดการทางด้านการแลกเปลี่ยนคีย์ ให้มีความปลอดภัย เช่น ฟังก์ชันที่ใช้ในการนำคีย์เข้าหรือออกจาก CSPs
- กลุ่มฟังก์ชันเข้ารหัสและถอดรหัสข้อมูล (Data Encryption Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่ในการเข้ารหัสและถอดรหัสข้อมูล
- กลุ่มฟังก์ชันแฮช, สร้างลายเซ็นต์ดิจิตอล และตรวจสอบลายเซ็นต์ (Hashing and Signature Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่ในการแฮชและสร้างลายเซ็นต์ดิจิตอลให้กับข้อมูล รวมทั้งการตรวจสอบลายเซ็นต์ดิจิตอล

สำหรับรายละเอียดของฟังก์ชันสามารถดูเพิ่มเติมได้ในภาคผนวก ค

# บทที่ 4
## การออกแบบ สร้าง และพัฒนาโครงงาน

### 4.1 การวิเคราะห์ระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

โครงงานนี้ได้ทำการวิเคราะห์และออกแบบระบบโดยใช้ Unify Modeling Language (UML) [1], [2] ซึ่งได้แสดงออกมาในรูปของไดอะแกรมต่างๆ ที่สำคัญ

4.1.1 ความต้องการเบื้องต้นของระบบ

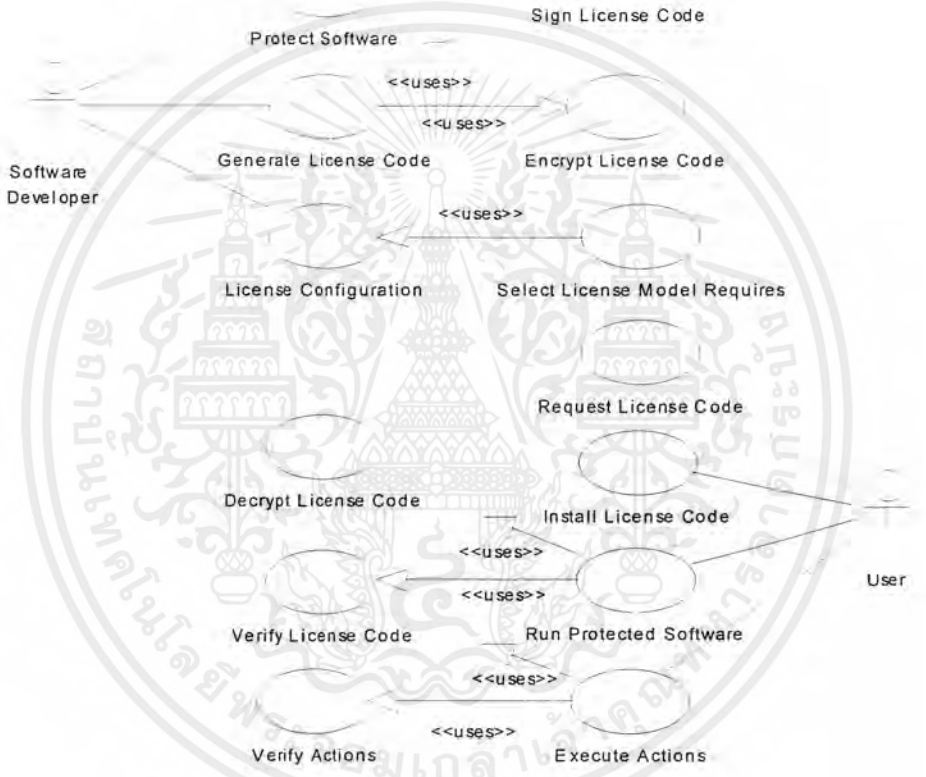
เนื่องจากระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ เป็นระบบที่ค่อนข้างจะมีความซับซ้อนประกอบด้วยองค์ประกอบหลายส่วนด้วยกัน ในโครงงานนี้จึงนำเสนอระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ขนาดเล็ก เพื่อแสดงให้เห็นถึงหลักการทำงานของระบบการบริหารและควบคุมสิทธิ์การใช้งาน โดยได้กำหนดความต้องการเบื้องต้นของระบบดังนี้

- ผู้ใช้งานระบบนี้ คือ
  - ผู้พัฒนาซอฟต์แวร์ คือ ผู้พัฒนาซอฟต์แวร์ที่ต้องการควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่ตนพัฒนาขึ้น
  - ผู้ใช้ซอฟต์แวร์ที่ถูกควบคุมสิทธิ์ คือ ผู้ใช้ที่ใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์การใช้งานโดยระบบนี้
- ระบบจะต้องสามารถทำการควบคุมสิทธิ์การใช้งานซอฟต์แวร์ได้ โดยพิจารณาจากเอกสารแสดงสิทธิ์
- ผู้พัฒนาซอฟต์แวร์หรือผู้ที่ได้รับอนุญาตเท่านั้นที่สามารถสร้างเอกสารแสดงสิทธิ์การใช้งานได้ โดยเอกสารแสดงสิทธิ์ที่สร้างจากผู้แอบอ้างจะไม่สามารถนำไปใช้งานได้
- ใช้ระบบสร้างรหัสแบบไม่สมมาตรในการสร้างลายเซ็นต์ดิจิตอลกำกับเอกสารแสดงสิทธิ์การใช้งาน
- มีแบบจำลองในการควบคุมสิทธิ์ที่สำคัญ คือ แบบทดลองใช้งานชั่วคราว, แบบเฉพาะเจาะจงเครื่อง, และแบบควบคุมแยกส่วนตามคุณสมบัติ หรืออาจจะรวมแบบจำลองต่างๆ เข้าด้วยกัน
- ระบบสามารถสร้างเอกสารแสดงสิทธิ์ให้กับผู้ใช้เพื่อขยายหรือจำกัดสิทธิ์การใช้งานซอฟต์แวร์ได้โดยผู้ใช้งานไม่จำเป็นต้องติดตั้งโปรแกรมใหม่
- ผู้ใช้สามารถเลือกกำหนดสิทธิ์การใช้งานในแต่ละส่วนของซอฟต์แวร์ได้เอง หรือผู้พัฒนาซอฟต์แวร์เป็นผู้กำหนดต้นแบบเบื้องต้นให้
- ระบบจะต้องมีส่วนเพิ่มความปลอดภัยเพื่อป้องกันการแก้ไขซอฟต์แวร์หรือเอกสารแสดงสิทธิ์การใช้งาน เช่น การตรวจสอบ CRC ของโปรแกรม, การเข้ารหัสเอกสารแสดงสิทธิ์การใช้งาน
- ระบบจะต้องไม่เพิ่มความซับซ้อนในการใช้งานซอฟต์แวร์ให้กับผู้ใช้งาน

- ระบบจะต้องสามารถช่วยอำนวยความสะดวกแก่ผู้พัฒนาซอฟต์แวร์ให้สามารถควบคุมสิทธิ์การใช้งานได้อย่างรวดเร็ว

4.1.2 Use Case Diagram

จากความต้องการเบื้องต้นของระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ สามารถกำหนดฟังก์ชันหลัก (Primary Functions) ที่จำเป็นต้องมีในการพัฒนาระบบ โดยนำเสนอเป็น ไดอะแกรม Use Case ได้ดังรูป




*รูปที่ 4-1 แสดง Use Case ไดอะแกรมของระบบ*

จากรูปที่ 4 – 1 ไดอะแกรม Use Case สามารถแบ่งฟังก์ชันของระบบออกเป็น 2 ส่วน คือ ส่วนที่ถูกใช้งานโดยผู้พัฒนาซอฟต์แวร์หรือผู้สร้างเอกสารแสดงสิทธิ์การใช้งาน และส่วนที่ถูกใช้งานโดยผู้ใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์การใช้งาน

4.1.3 ฟังก์ชันสำหรับผู้พัฒนาซอฟต์แวร์

- Protect Software เป็นฟังก์ชันที่ทำหน้าที่ในการป้องกันซอฟต์แวร์จากการแก้ไขหรือดัดแปลงส่วนใดส่วนหนึ่งของซอฟต์แวร์

- Sign License Code เป็นฟังก์ชันที่ใช้ในการสร้างลายเซ็นต์ดิจิตอลเพื่อกำกับเอกสารแสดงสิทธิ์ โดยลายเซ็นต์ดังกล่าวจะเป็นส่วนที่ช่วยให้ระบบสามารถตรวจสอบที่มาของเอกสารแสดงสิทธิ์การใช้งาน เพื่อป้องกันการแอบอ้าง
- Encrypt License Code ฟังก์ชันนี้ใช้สำหรับการเข้ารหัสเอกสารแสดงสิทธิ์เพื่อเพิ่มความปลอดภัยให้กับเอกสารแสดงสิทธิ์ในการใช้งานในอีกระดับหนึ่ง
- Generate License Code เป็นฟังก์ชันที่ใช้ในการสร้างเอกสารแสดงสิทธิ์การใช้งานให้กับผู้ใช้ โดยฟังก์ชันนี้จะมีการเรียกใช้งานฟังก์ชันการสร้างลายเซ็นต์ดิจิตอล และการเข้ารหัส ดังกล่าวข้างต้น
- License Configuration ใช้ในการกำหนดค่าต่างๆ ให้กับระบบ เช่น การแบ่งแยกส่วนของซอฟต์แวร์ การกำหนดแบบจำลองในการควบคุมสิทธิ์การใช้งานในแต่ละส่วนของซอฟต์แวร์ อัลกอริธึมที่ใช้ในการเข้ารหัส การสร้างลายเซ็นต์ดิจิตอล เป็นต้น

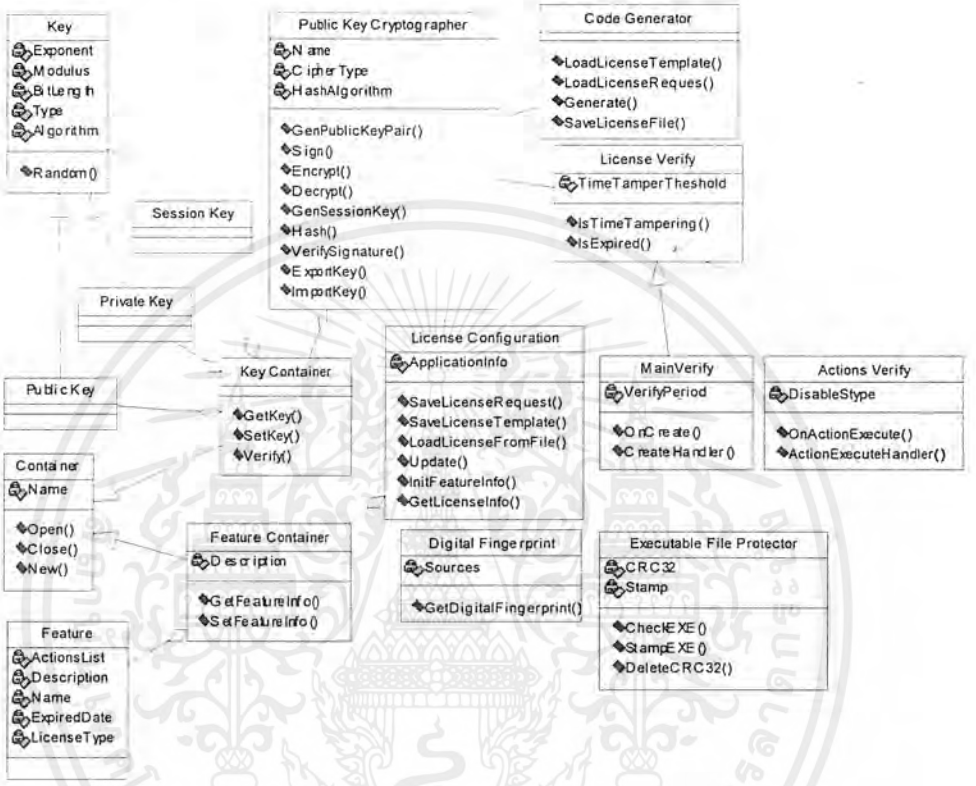
4.1.4 ฟังก์ชันสำหรับผู้ใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์

- Decrypt License Code ฟังก์ชันนี้ใช้สำหรับการถอดรหัสเอกสารแสดงสิทธิ์ที่ถูกเข้ารหัส
- Verify License Code เป็นฟังก์ชันที่ใช้สำหรับการตรวจสอบลายเซ็นต์ดิจิตอลที่กำกับในเอกสารแสดงสิทธิ์การใช้งาน
- Verify Actions ใช้สำหรับตรวจสอบและอนุญาตให้ผู้ใช้งานซอฟต์แวร์สามารถใช้งานฟังก์ชันใดฟังก์ชันหนึ่งในซอฟต์แวร์ โดยพิจารณาจากเอกสารแสดงสิทธิ์การใช้งาน เช่น พิจารณาจากวันหมดอายุการใช้งาน
- Select License Model Requires เป็นฟังก์ชันที่อนุญาตให้ผู้ใช้สามารถเลือกแบบจำลองที่ใช้ในการควบคุมสิทธิ์และค่าต่างๆ เช่น วันหมดอายุการใช้งาน ฟังก์ชันที่ต้องใช้งาน ตามความต้องการ
- Request License Code ฟังก์ชันนี้ใช้สำหรับการสร้างเอกสารข้อมูลผู้ใช้และความต้องการของผู้ใช้ เพื่อให้ผู้พัฒนาซอฟต์แวร์สามารถสร้างเอกสารแสดงสิทธิ์การใช้งานได้ตามความต้องการของผู้ใช้
- Install License Code เป็นฟังก์ชันที่อำนวยความสะดวกแก่ผู้ใช้ในการติดตั้งเอกสารแสดงสิทธิ์ ให้กับซอฟต์แวร์ที่ถูกควบคุมสิทธิ์
- Run Protected Software เป็นฟังก์ชันที่ผู้ใช้เรียกใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์ เมื่อเกิดฟังก์ชันนี้ระบบก็จะทำการตรวจสอบเอกสารแสดงสิทธิ์ เพื่อพิจารณาอนุญาตให้ผู้ใช้งานสามารถใช้งานซอฟต์แวร์ดังกล่าวได้หรือไม่
- Execute Actions ฟังก์ชันนี้เกิดขึ้นเมื่อผู้ใช้งานเรียกใช้งานฟังก์ชันใดฟังก์ชันหนึ่งที่ได้ถูกควบคุมสิทธิ์ไว้ ส่วนของฟังก์ชัน Verify Action ก็จะทำการตรวจสอบ เพื่อพิจารณาอนุญาตให้ผู้ใช้สามารถใช้งานฟังก์ชันนั้นหรือไม่

## 4.2 การออกแบบระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์

### 4.2.1 คลาสไดอะแกรมของระบบที่ออกแบบ (Class Diagram)

จากการวิเคราะห์ความต้องการเบื้องต้นของระบบและฟังก์ชันหลักๆ ที่ระบบจะต้องสามารถตอบสนองให้กับผู้ใช้งานได้ จึงได้ทำการออกแบบระบบในเชิงวัตถุ (Object-Oriented Design) ซึ่งประกอบด้วยคลาสต่างๆ ที่สำคัญดังรูป




*รูปที่ 4-2 แสดงคลาสไดอะแกรมของระบบ*

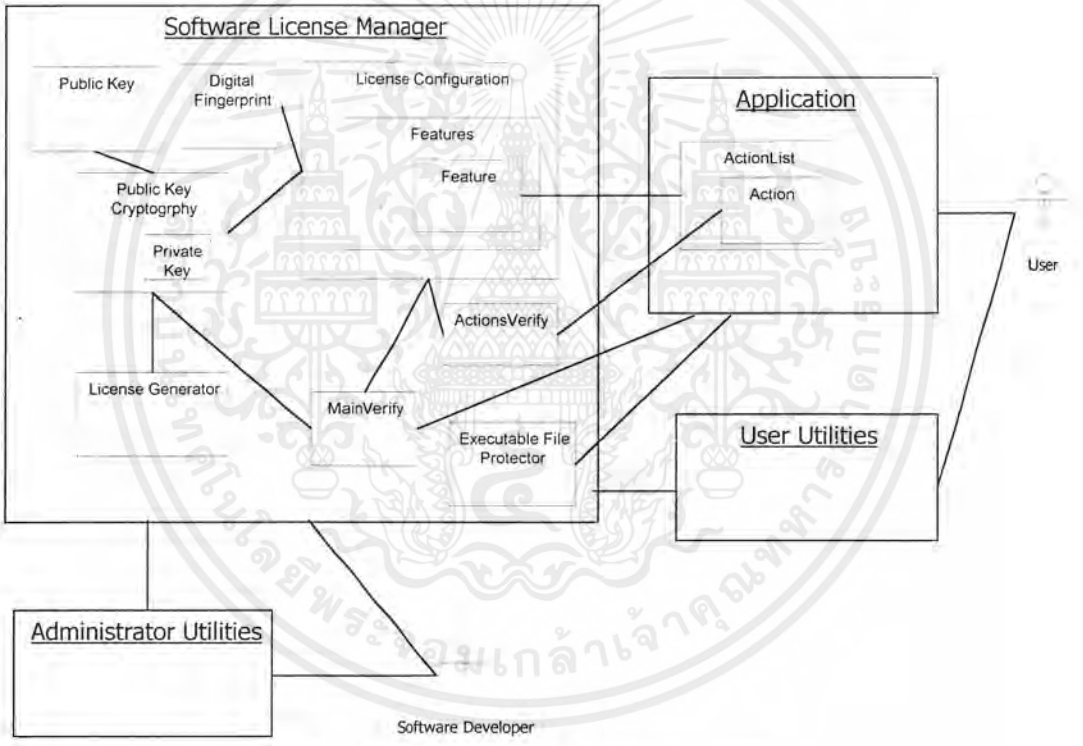
จากรูปที่ 4 – 2 ประกอบด้วยคลาสต่างๆ ที่สำคัญ ดังนี้

- Public Key Cryptographer คลาสนี้มีหน้าที่หลักในการสร้างรหัสความปลอดภัยให้กับระบบ โดยสามารถบริหารและควบคุมคีย์ต่างๆ ที่ใช้ในการสร้างรหัส โดยอาศัยคลาส Key Container ทำหน้าที่ในการเก็บคีย์ต่างๆ เช่น Private/Public Key Pair และ Session Key
- License Configuration เป็นคลาสที่ใช้ในการแยกส่วนฟังก์ชันภายในซอฟต์แวร์ โดยฟังก์ชันต่างๆ จะถูกแยกออกเป็น Feature เพื่อให้สามารถกำหนดสิทธิ์การใช้งานที่แตกต่างกันในแต่ละ Feature ได้
- Code Generator ทำหน้าที่ในการสร้างเอกสารแสดงสิทธิ์การใช้งาน
- License Verify คลาสนี้ช่วยในการตรวจสอบเอกสารแสดงสิทธิ์การใช้งาน ซึ่งทำโดยการตรวจสอบลายเซ็นต์ที่กำกับมากับเอกสารแสดงสิทธิ์ รวมทั้งทำการตรวจสอบ เพื่ออนุญาตให้ผู้ใช้ซอฟต์แวร์สามารถใช้งานฟังก์ชันใด ฟังก์ชันหนึ่งได้หรือไม่

- Digital Fingerprint เป็นคลาสที่ใช้สำหรับการดึงค่ารหัสของเครื่องที่ติดตั้งซอฟต์แวร์ที่ถูกควบคุมสิทธิ์ เมื่อต้องการล็อคให้ผู้ใช้สามารถติดตั้งซอฟต์แวร์ดังกล่าวเฉพาะบนเครื่องใดเครื่องหนึ่ง
- Executable File Protector เป็นคลาสที่ทำหน้าที่ในการป้องกันไฟล์ซอฟต์แวร์จากการแก้ไขหรือดัดแปลง โดยการตรวจสอบ CRC ของไฟล์

4.2.2 ออบเจ็กต์ไดอะแกรม (Object Diagram)

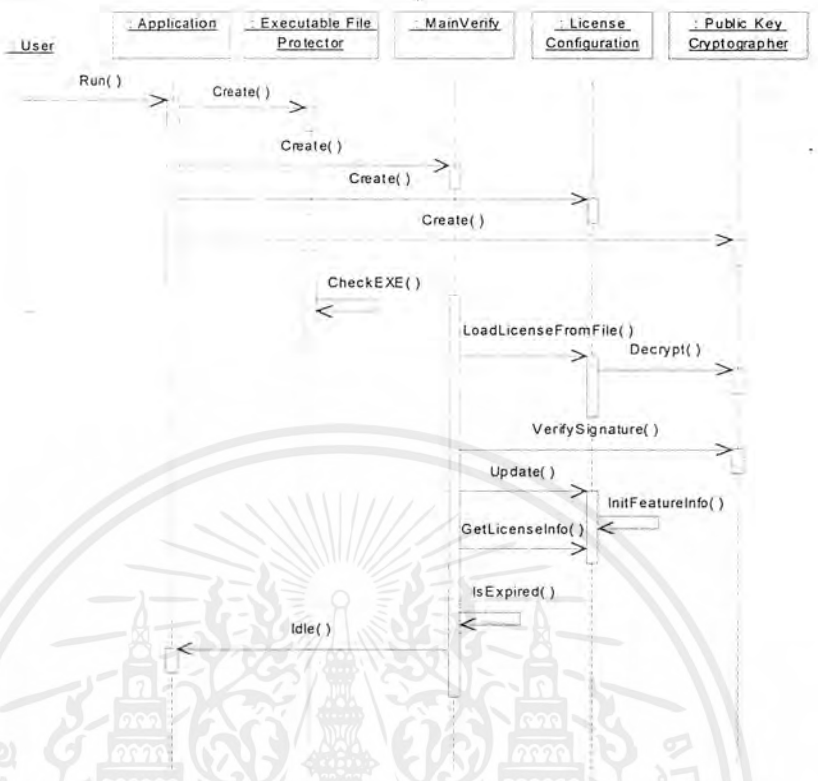
จากคลาสไดอะแกรมของระบบ สามารถแสดงความสัมพันธ์ระหว่าง ออบเจ็กต์ต่างๆ ในระบบและภายนอกระบบที่เกี่ยวข้อง ดังรูปที่ 4 – 3 โดยจะเห็นว่าระบบจะถูกนำไปใช้งานใน 3 ส่วน หลักด้วยกัน คือ ส่วนของซอฟต์แวร์ที่ต้องการควบคุมสิทธิ์, ส่วนของเครื่องมือการบริหารและควบคุมสิทธิ์การใช้งานทั้งในฝั่งของผู้พัฒนาซอฟต์แวร์เอง และฝั่งของผู้ใช้งานซอฟต์แวร์ถูกควบคุมสิทธิ์



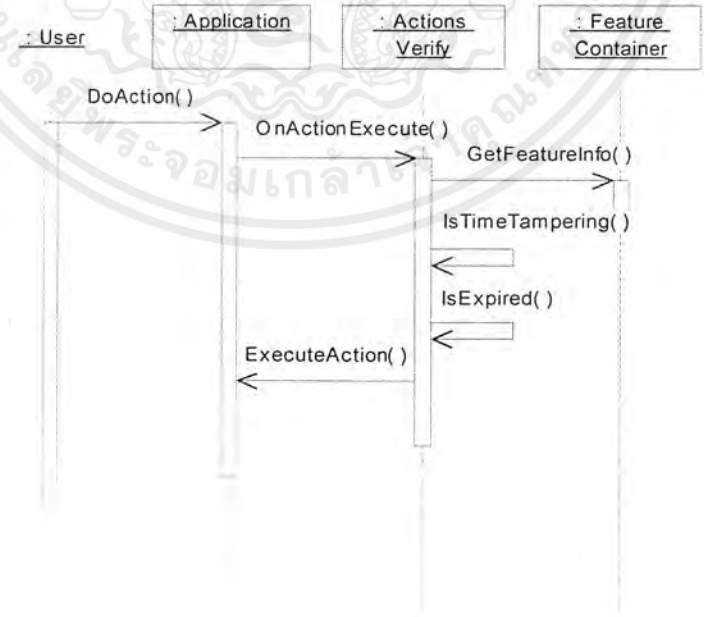

*รูปที่ 4-3 แสดงออบเจ็กต์ไดอะแกรมของระบบ*

4.2.3 ตัวอย่าง Scenarios ที่สำคัญในระบบ

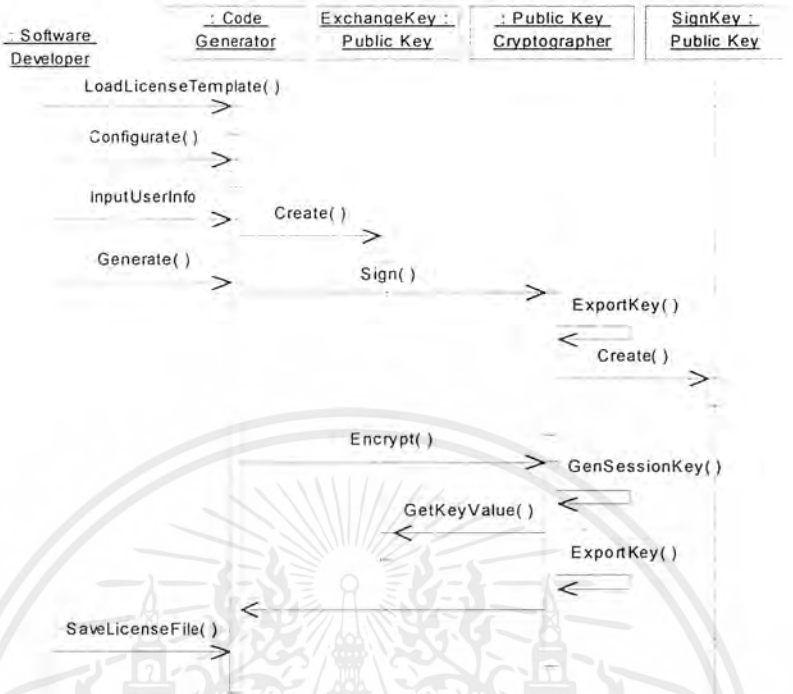
- Run Protected Application เป็น Scenario เมื่อผู้ใช้เริ่มใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์การใช้งาน แสดงโดย Sequence ไดอะแกรม ดังรูปที่ 4 – 4
- Execute Actions เป็น Scenario เมื่อผู้ใช้เรียกใช้งานฟังก์ชันที่ถูกควบคุมสิทธิ์การใช้งาน ภายในซอฟต์แวร์ ดังรูปที่ 4 – 5
- Generate License Code เป็น Scenario ในการสร้างเอกสารแสดงสิทธิ์การใช้งานของผู้พัฒนาซอฟต์แวร์ ดังรูปที่ 4 – 6




*รูปที่ 4-4 แสดง Scenario ในการเรียกใช้งานโปรแกรมที่ถูกควบคุมสิทธิ์การใช้งาน*




*รูปที่ 4-5 แสดง Scenario ในการเรียกใช้งานฟังก์ชันที่ถูกควบคุมสิทธิ์การใช้งาน*

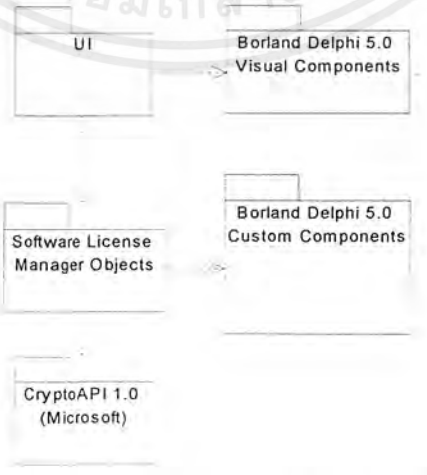



*รูปที่ 4-6 แสดง Scenario ในการสร้างเอกสารแสดงสิทธิ์การใช้งาน*

**4.3 การสร้าง และพัฒนาระบบ**

4.3.1 ส่วนประกอบหลักในการสร้าง และพัฒนาระบบ

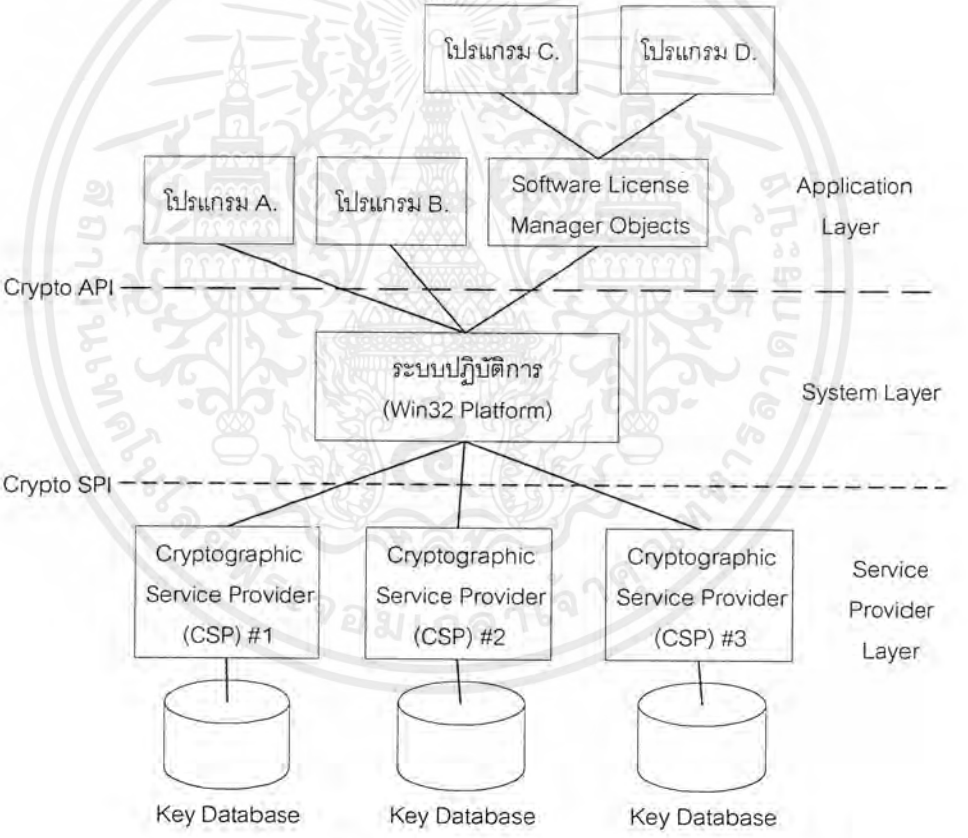
โครงงานนี้ได้นำเสนอระบบการบริหารและควบคุมสิทธิ์การใช้งาน ในระดับคอมโพเนนท์ ซึ่งจะพัฒนาโดยใช้ Borland Delphi 5.0 โดยมีส่วนประกอบหลักในการสร้างระบบ ดังรูปที่ 4 – 7




*รูปที่ 4-7 แสดงส่วนประกอบหลักของระบบที่สร้างขึ้น*

จากรูปที่ 4 – 7 ส่วนประกอบในการสร้างระบบจะแบ่งเป็นดังนี้

- UI (User Interface) เป็นส่วนที่ใช้สำหรับติดต่อกับผู้ใช้งาน เพื่อให้สามารถใช้งานระบบได้โดยง่าย ได้แก่ ฟอร์มที่ใช้ในการตั้งค่าต่างๆ ให้กับระบบ ฟอร์มที่ใช้แสดงข้อความผิดพลาดที่เกิดขึ้นภายในระบบ เป็นต้น โดยส่วนนี้จะสร้างโดยใช้ Visual Components ของ Borland Delphi 5.0
- Software License Manager Objects เป็นส่วนที่นำเอาคลาสต่างๆ ที่ได้ออกแบบข้างต้น มาพัฒนาในระดับคอมโพเนนท์ เพื่อให้สามารถใช้งานได้ง่าย
- CryptoAPI 1.0 (Microsoft) ส่วนนี้จะเป็น APIs ในระดับล่างที่ใช้ในการสร้างรหัสความปลอดภัยให้กับระบบ ซึ่งประกอบด้วยฟังก์ชันต่างๆ ทำอำนวยความสะดวกในการสร้าง Public Key Cryptography ซึ่งสามารถแสดงความสัมพันธ์ระหว่าง Software License Manager Objects กับ Microsoft CryptoAPI ดังรูปที่ 4 – 8




*รูปที่ 4-8 แสดงความสัมพันธ์ของระบบกับ Microsoft CryptoAPI 1.0*

4.3.2 คลาสไดอะแกรมของระบบที่สร้างขึ้น


*รูปที่ 4-9 แสดงคลาสไดอะแกรมของระบบที่ได้สร้างและพัฒนาขึ้น*

จากรูปที่ 4 – 9 แสดงคลาสไดอะแกรมของระบบที่สร้าง และพัฒนาขึ้นโดยใช้ Borland Delphi 5.0 ซึ่งประกอบด้วยคลาสต่างๆ ที่สำคัญ ดังนี้

- TCryptoManager เป็นคอมโพเนนท์ที่ทำหน้าที่สร้างรหัสความปลอดภัยในระบบแบบ Public Key Cryptographic โดยจะเรียกใช้งานฟังก์ชันของ Microsoft CryptoAPI 1.0 นอกจากนี้ TCryptoManager ยังมีฟังก์ชันในอ่านรหัสหมายเลขของเครื่องจาก 4 แหล่งด้วยกัน คือ หมาย

เลขของอีเทอร์เน็ตการ์ด, ข้อมูลเฉพาะของฮาร์ดดิสก์, หมายเลขลอจิคอลของดิสก์, หมายเลขตัวประมวลผล เพื่อใช้ในแบบจำลองการควบคุมสิทธิ์ที่เฉพาะเจาะจงเครื่องที่ติดตั้ง

- TLicConfig เป็นคอมโพเนนท์ที่อำนวยความสะดวกให้กับผู้พัฒนาโปรแกรมในการกำหนดค่าต่างๆ ในการควบคุมสิทธิ์ เช่น การจัดแบ่งฟังก์ชันออกเป็นส่วนๆ โดยทำงานร่วมกับ TActionList ซึ่งเป็นคอมโพเนนท์ที่มีอยู่แล้วใน Borland Delphi 5.0 ภายในคอมโพเนนท์นี้จะประกอบด้วยคลาสย่อย TFeatures ซึ่งจะเก็บค่าการควบคุมสิทธิ์ของแต่ละกลุ่มของฟังก์ชันที่ถูกแบ่ง
- TVerify คอมโพเนนท์สืบทอดมาจาก TCustomApplicationEvent โดยจะสามารถดักจับเหตุการณ์ทั้งหมดที่เกิดขึ้นกับแอพพลิเคชัน โดยได้เพิ่มให้มีความสามารถในการตรวจสอบการแก้ไขเวลาของระบบ เพื่อป้องกันการย้อนเวลา
- TActionsVerify จะทำงานร่วมกับ TLicConfig ทำหน้าที่ดักจับเหตุการณ์ที่เกิดขึ้นเมื่อมีการเรียกใช้งานฟังก์ชันภายในแอพพลิเคชันของผู้ใช้งานซอฟต์แวร์ที่ถูกควบคุมสิทธิ์ เพื่อพิจารณาอนุญาตให้ผู้ใช้สามารถใช้งานฟังก์ชันนั้นหรือไม่
- TMainVerify จะทำงานร่วมกับ TCryptoManager เพื่อถอดรหัสและตรวจสอบความถูกต้องของลายเซ็นต์ที่กำกับมากับเอกสารแสดงสิทธิ์การใช้งาน ทุกครั้งที่มีการเรียกใช้งานแอพพลิเคชัน หากเอกสารถูกต้องก็จะอนุญาตให้ TLicConfig สามารถนำค่าไปใช้งานได้ นอกจากนี้ยังทำหน้าที่ในการตรวจสอบเอกสารแสดงสิทธิ์ตามช่วงเวลาที่กำหนด
- TPublicKey เป็นคอมโพเนนท์ที่เก็บค่าต่างๆ ของ Public Key โดยจะฝังไปแอพพลิชัน ซึ่งจะถูกเรียกใช้โดย TCryptoManager ในการตรวจสอบลายเซ็นต์ที่กำกับเอกสารแสดงสิทธิ์การใช้งาน
- TProtectEXE เป็นคอมโพเนนท์ที่ทำหน้าที่ช่วยฝัง CRC32 ให้กับแอพพลิเคชัน และทำการตรวจสอบ CRC32 ทุกครั้งที่มีการเรียกใช้งานแอพพลิเคชัน เพื่อป้องกันการแก้ไขดัดแปลงส่วนใดส่วนหนึ่งของซอฟต์แวร์
- TCodeGenerator เป็นคอมโพเนนท์ที่ใช้ในการสร้างเอกสารแสดงสิทธิ์การใช้งาน โดยทำงานร่วมกับ TCryptoManager ในการสร้างลายเซ็นต์และเข้ารหัสเอกสารแสดงสิทธิ์

# บทที่ 5

## การทดสอบระบบและตัวอย่างการใช้งานจริง

### 5.1 การทดสอบความต้องการของระบบกับระบบจริงที่พัฒนาขึ้น

หลังจากที่ได้สร้างและพัฒนาระบบขึ้น ในการทดสอบระบบจะเริ่มจากการพิจารณาความต้องการเบื้องต้นของระบบ โดยแยกเป็นข้อๆ และทดสอบดูว่าระบบสามารถตอบสนองต่อความต้องการดังกล่าวได้หรือไม่ โดยวิธีใด และสรุปคลาสที่เกี่ยวข้อง รายละเอียดแสดงดังตารางที่ 5 – 1

| ความต้องการของระบบ | การตอบสนองระบบ | คลาสที่เกี่ยวข้อง |
|---|---|---|
| ระบบจะต้องสามารถทำการควบคุมสิทธิ์การใช้งานซอฟต์แวร์ได้ โดยพิจารณาจากเอกสารแสดงสิทธิ์ | มีไฟล์เอกสารแสดงสิทธิ์สำหรับเก็บรายละเอียดต่างๆ | TMainVerify<br>TLicConfig<br>TCodeGenerator |
| ผู้พัฒนาซอฟต์แวร์หรือผู้ที่ได้รับอนุญาตเท่านั้นที่สามารถสร้างเอกสารแสดงสิทธิ์การใช้งานได้ โดยเอกสารแสดงสิทธิ์ที่สร้างจากผู้แอบอ้างจะไม่สามารถนำไปใช้งานได้ | ใช้การสร้างลายเซ็นต์ดิจิตอลกำกับเอกสารแสดงสิทธิ์ เพื่อระบุตรวจสอบที่มาของเอกสารสิทธิ์ได้ นอกจากนี้ยังสามารถเข้ารหัสเพื่อระบุปลายทางผู้รับเอกสารสิทธิ์ได้ด้วย | TCodeGenerator<br>TCryptoManager<br>TPublicKey |
| ใช้ระบบสร้างรหัสแบบไม่สมมาตรในการสร้างลายเซ็นต์ดิจิตอลกำกับเอกสารแสดงสิทธิ์การใช้งาน | ทำงานผ่าน Microsoft CryptoAPI 1.0 เพื่อสร้างรหัสแบบไม่สมมาตรโดยใช้ RSA Public Key Cryptography | TCryptoManager |
| มีแบบจำลองในการควบคุมสิทธิ์ที่สำคัญ คือ แบบทดลองใช้งานชั่วคราว, แบบเฉพาะเจาะจงเครื่อง, และแบบควบคุมแยกส่วน หรืออาจจะรวมแบบจำลองต่างๆ เข้าด้วยกัน | มีการกำหนดวันหมดอายุการใช้งานตั้งแต่ในระดับแอพพลิเคชัน หรืออาจจะแบ่งแอ็กชันออกเป็นฟีเจอร์แล้วกำหนดรูปแบบการควบคุมในแต่ละฟีเจอร์ | TLicConfig<br>TcryptoMananger<br>(ฟังก์ชัน GetDFP() ใช้อ่านค่าลายมือดิจิตอลของเครื่อง |
| ระบบสามารถสร้างเอกสารแสดงสิทธิ์ให้กับผู้ใช้เพื่อขยายหรือจำกัดสิทธิ์การใช้งานซอฟต์แวร์ได้โดยผู้ใช้งานไม่จำเป็นต้องติดตั้งโปรแกรมใหม่ | มีเครื่องมือที่ใช้ในการสร้างไฟล์เอกสารแสดงสิทธิ์ได้ตามความต้องการของผู้ใช้ โดยผู้ใช้สามารถนำไฟล์ดังกล่าวแทนไฟล์เดิมโดยไม่จำเป็นต้องติดตั้งโปรแกรมใหม่ | TCodeGenerator<br>TLicConfig |
| ผู้ใช้สามารถเลือกกำหนดสิทธิ์การใช้งานในแต่ละส่วนของซอฟต์แวร์ได้เอง หรือผู้พัฒนาซอฟต์แวร์เป็นผู้กำหนดต้นแบบ | มีเครื่องมือที่ช่วยในการเลือกฟีเจอร์และรูปแบบการควบคุมสิทธิ์ตามความต้องการของผู้ใช้ | TLicConfig |

| ความต้องการของระบบ | การตอบสนองระบบ | คลาสที่เกี่ยวข้อง |
|---|---|---|
| เบื้องต้นให้ | | |
| ระบบจะต้องมีส่วนเพิ่มความปลอดภัยเพื่อป้องกันการแก้ไขซอฟต์แวร์หรือเอกสารแสดงสิทธิ์การใช้งาน เช่น การตรวจสอบ CRC ของโปรแกรม, การเข้ารหัสเอกสารแสดงสิทธิ์การใช้งาน | มีเครื่องมือที่ใช้ในการฝังค่า CRC32 ลงในตัวซอฟต์แวร์ เพื่อสามารถตรวจจับการแก้ไขและดัดแปลงซอฟต์แวร์ได้ และยังสามารถเข้ารหัสเอกสารแสดงสิทธิ์การใช้งานไม่ให้อ่านได้ | TProtectEXE<br>TCryptoManager |
| ระบบจะต้องไม่เพิ่มความซับซ้อนในการใช้งานซอฟต์แวร์ให้กับผู้ใช้งาน | มีเครื่องมืออำนวยความสะดวก ในการติดตั้งเอกสารสิทธิ์ | เครื่องมือสำหรับผู้ใช้งานซอฟต์แวร์ |
| ระบบจะต้องสามารถช่วยอำนวยความสะดวกแก่ผู้พัฒนาซอฟต์แวร์ให้สามารถควบคุมสิทธิ์การใช้งานได้อย่างรวดเร็ว | ผู้พัฒนาสามารถควบคุมสิทธิ์ซอฟต์แวร์โดยการใช้งานคอมโพเนนท์ โดยอาจจะไม่ต้องเขียนโค้ดเพิ่มเติม | TLicConfig<br>TCodeGenerator<br>TCryptoManager<br>TPublicKey<br>TProtectEXE<br>TMainVerify<br>TActionsVerify |

*ตารางที่ 5-1 แสดงการทดสอบความต้องการเบื้องต้นของระบบกับระบบที่พัฒนาขึ้น*

**5.2 ตัวอย่างการทดสอบโดยการประยุกต์ใช้งาน**

5.2.1 วัตถุประสงค์ในการทดสอบ

- เพื่อทดสอบการทำงานของคอมโพเนนท์แต่ละคอมโพเนนท์ว่าสามารถทำงานได้ถูกต้องหรือไม่
- เพื่อสร้างแอพพลิเคชันอย่างง่ายและทดสอบการทำงานของคอมโพเนนท์ที่สัมพันธ์กันว่าสามารถทำงานได้ตามลำดับขั้นตอนที่ได้ออกแบบไว้หรือไม่
- เพื่อแสดงให้เห็นถึงรูปแบบการจัดการสิทธิ์การใช้งานซอฟต์แวร์อย่างง่ายที่เป็นที่นิยมใช้งานกัน
- เพื่อแสดงให้เห็นถึงการทำงานของระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ของโครงงาน

5.2.2 กรรมวิธีการทดสอบ

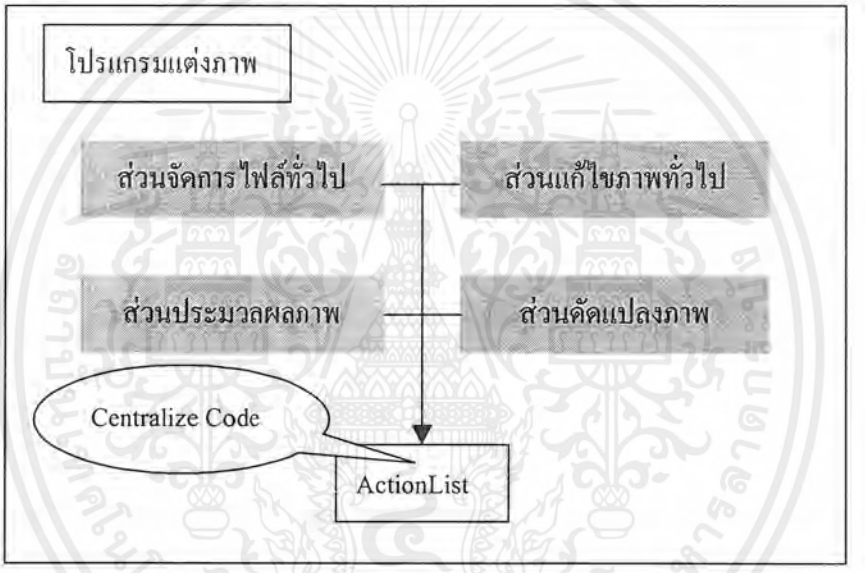
วิธีการทดสอบจะทำตามขั้นตอนดังต่อไปนี้

- ทำการกำหนดรูปแบบของการจัดการสิทธิ์การใช้งานซอฟต์แวร์ที่จะทำการทดสอบ
- ทำการพัฒนาแอพพลิเคชันตัวอย่างโดยนำคอมโพเนนท์ที่ทำหน้าในการบริหารและควบคุมสิทธิ์มาใช้
- ทำการทดสอบการทำงานของแอพพลิเคชันในสถานะปกติ

- ทำการทดสอบการทำงานของแอพพลิเคชันเมื่อมีการละเมิดเงื่อนไขที่ได้กำหนดเอาไว้ในเอกสารสิทธิ์
- ทำการทดสอบการขอเอกสารสิทธิ์
- ทำการทดสอบการเซ็นต์และจ่ายเอกสารสิทธิ์

5.2.3 สร้างตัวอย่างการทดสอบ

จากคอมโพเนนท์ต่างๆ ที่ได้สร้างขึ้นทั้ง 6 คอมโพเนนท์ ดูรายละเอียดในภาคผนวก ก สามารถนำไปควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่พัฒนาไว้แล้ว โดยในตัวอย่าง (ดูรายละเอียดการสร้างตัวอย่างสำหรับทดสอบระบบได้ในภาคผนวก ข) ได้สร้างซอฟต์แวร์สำหรับทดสอบอย่างง่าย ซึ่งมีเฉพาะส่วนติดต่อกับผู้ใช้ โดยใช้ TActionList ในการ Centralize Code ของ Main Menu กับ Toolbars เข้าด้วยกัน ซึ่งสามารถแสดงไดอะแกรมของฟังก์ชันต่างๆ ที่มีในซอฟต์แวร์ได้ดังรูป

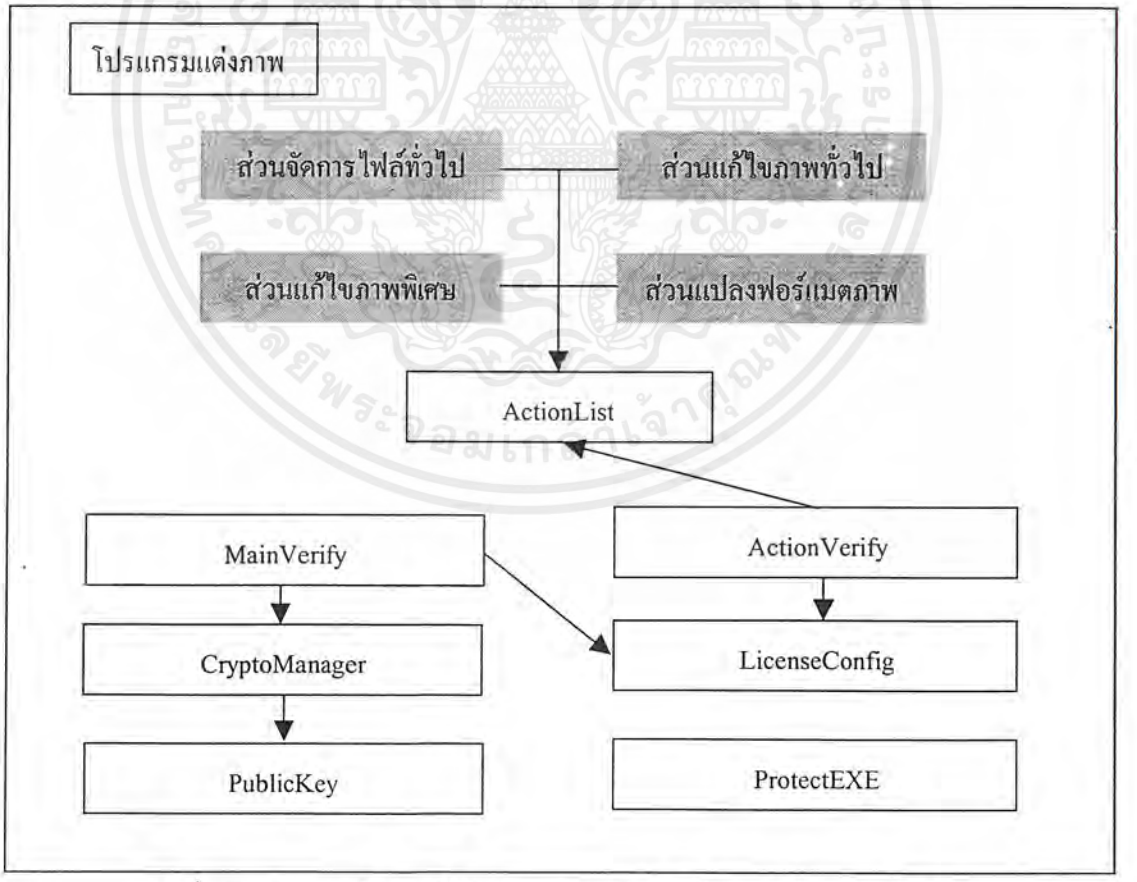



*รูปที่ 5-1 แสดงองค์ประกอบของโปรแกรมแต่งภาพที่มีคุณสมบัติหลัก 4 คุณสมบัติ*

ซอฟต์แวร์ตัวอย่างเป็นโปรแกรมแต่งภาพที่ประกอบไปด้วยคุณสมบัติหลักทั้งหมด 4 คุณสมบัติด้วยกัน คือ ส่วนจัดการไฟล์ทั่วไป ส่วนแก้ไขภาพทั่วไป ส่วนประมวลผลภาพ ส่วนดัดแปลงภาพ ในแต่ละคุณสมบัติจะประกอบด้วยฟังก์ชันย่อยต่างๆ เช่น ในส่วนจัดการไฟล์ทั่วไป จะประกอบฟังก์ชันในการสร้างไฟล์ใหม่, เปิดไฟล์เก่า, บันทึกไฟล์ เป็นต้น โดยฟังก์ชันต่างๆ เหล่านี้จะถูก Centralize โดย TActionList

การควบคุมสิทธิ์การซอฟต์แวร์โดยใช้คอมโพเนนท์ที่พัฒนาขึ้น จะเริ่มจาก การวางคอมโพเนนท์ TCryptoManager ลงไปฟอร์มหลัก (ฟอร์มที่มีส่วนเชื่อมต่อกับผู้ใช้ในการเรียกใช้งานคุณสมบัติต่างๆ ที่ต้องการควบคุมสิทธิ์) ในแอพพลิเคชันเพื่อที่จะนำไปจัดการด้านการสร้างรหัสความปลอดภัยให้กับแอพพลิเคชัน จากนั้นจะเพิ่มคอมโพเนนท์ TPublicKey สำหรับฝังคีย์สาธารณะของผู้มีสิทธิ์ในการสร้างเอกสารสิทธิ์ เพื่อใช้ในการพิสูจน์ลายเซ็นต์ของเอกสารสิทธิ์ ซึ่งทำงานโดยคอมโพเนนท์

TCryptoManager คอมโพเนนท์ต่อไปที่เพิ่มเข้าไปก็คือคอมโพเนนท์ TLicenseConfig เป็นคอมโพเนนท์ที่ใช้ในการกำหนดคุณสมบัติ (Features) และจัดแบ่ง Actions ให้แต่ละ Feature โดยคอมโพเนนท์ตัวนี้จะทำงานร่วมกับคอมโพเนนท์อีกตัวหนึ่งที่เพิ่มเข้าไปก็คือ คอมโพเนนท์ TMainVerify จะเป็นคอมโพเนนท์ที่ทำการตรวจสอบความถูกต้องของเอกสารสิทธิ์จากไฟล์ ก่อนนำมาเก็บไว้ที่ คอมโพเนนท์ TLicenseConfig นอกจากนี้ยังทำการตรวจสอบความถูกต้องของเอกสารสิทธิ์อีกเป็นระยะๆ ตามช่วงเวลาที่ได้กำหนดเอาไว้ คอมโพเนนท์ต่อไปที่จะเพิ่มเข้าไปคือ TActionsVerify ซึ่งเป็นคอมโพเนนท์ที่ในการตรวจสอบความถูกต้องและอายุการใช้งานของคุณสมบัติต่างๆ ในแอพพลิเคชัน ในขณะที่มีการเรียกใช้คุณสมบัตินั้นๆ โดยในซอฟต์แวร์ทดลองใช้งานในตัวอย่างนี้ก็คือ โปรแกรมแต่งภาพมีการกำหนดเอกสารสิทธิ์ให้ คุณสมบัติส่วนจัดการไฟล์ทั่วไปและ ส่วนแก้ไขภาพทั่วไปสามารถใช้งานได้ภายในเวลาที่กำหนดคือจะกำหนดวันหมดอายุเอาไว้ ส่วนคุณสมบัติ ส่วนประมวลผลภาพ และ ส่วนดัดแปลงภาพจะกำหนดไม่ให้สามารถใช้งานได้ ถ้าหากลูกค้าต้องการจะใช้งานในส่วนดังกล่าวจะต้องทำการร้องขอเอกสารสิทธิ์ใหม่ที่เป็นแบบถาวร คอมโพเนนท์ตัวสุดท้ายที่จะเพิ่มเข้าไปก็คือ TProtectEXE ทำหน้าที่ในการตรวจสอบการแก้ไขไฟล์ที่จะนำไปเอ็กซีคิว ว่ามีการเปลี่ยนแปลงแก้ไขส่วนหนึ่งส่วนใดหรือเปล่า โดยเฉพาะส่วนของคีย์สาธารณะที่จะใช้ในการพิสูจน์ลายเซ็นต์ของเอกสารสิทธิ์ที่ได้ฝังเอาไว้กับตัวแอพพลิเคชัน หากถูกแก้ไขจะทำให้สามารถแอบอ้างสร้างเอกสารสิทธิ์ได้




*รูปที่ 5-2 แสดงคอมโพเนนท์ต่างๆ และความสัมพันธ์ในการสร้างแอพพลิเคชัน*

จากนั้นทำการบันทึก License Template เก็บลงไฟล์ไว้ (โดยใช้ TLicenseConfig) ซึ่งไฟล์ดังกล่าวจะใช้เป็นโครงร่างของเอกสารสิทธิ์ สำหรับเอกสารสิทธิ์แบบทดลองใช้งาน (Demo Version) ซึ่งสามารถติดตั้งบนเครื่องใดก็ได้ แต่เอกสารสิทธิ์ประเภทนี้จะสามารถใช้งานได้ชั่วคราวเท่านั้น ขึ้นอยู่กับวันหมดอายุที่กำหนดในเอกสารสิทธิ์

ในกรณีที่ผู้ใช้ต้องการขยายสิทธิ์การใช้งานสามารถขอเอกสารสิทธิ์ใหม่ โดยการส่งไฟล์ License Request ให้แก่ผู้พัฒนาซอฟต์แวร์ เพื่อให้ผู้พัฒนาซอฟต์แวร์ สร้างเอกสารสิทธิ์ใหม่ให้ตามความต้องการ โดยเอกสารสิทธิ์ประเภทนี้จะสามารถใช้งานได้เฉพาะเจาะจงเครื่อง และมีการเข้ารหัสข้อมูลด้วย

5.2.4 ผลการทดสอบ

จากการทดสอบตามขั้นตอนต่างๆ ที่ผ่านมาทั้งหมดพบว่าระบบที่ออกแบบสามารถทำงานได้ถูกต้องตามขั้นตอนที่ออกแบบเอาไว้ โดยสามารถป้องกันการเรียกใช้งานคุณสมบัติที่หมดอายุการใช้งานได้, สามารถตรวจสอบได้ว่ามีการตั้งเวลาย้อนถอยหลังหรือไม่, สามารถสร้างเอกสารแสดงสิทธิ์เพื่อเพิ่มหรือจำกัดคุณสมบัติของซอฟต์แวร์ได้โดยไม่จำเป็นต้องติดตั้งโปรแกรมใหม่ โดยสามารถสรุปผลการทดสอบได้ดังตาราง

| | เหตุการณ์ | การตอบสนองของระบบ |
|---|---|---|
| เอกสารสิทธิ์แบบทดลองใช้ (Demo Version) | โปรแกรมหมดอายุการใช้งาน | แจ้งข้อความบ่งบอกสถานะหมดอายุการใช้งานของโปรแกรม |
| | เรียกใช้งานคุณสมบัติที่หมดอายุการใช้งาน หรือถูกกำหนดให้สามารถใช้งานได้ | แจ้งข้อความบ่งบอกสถานะไม่อนุญาตให้ใช้งานคุณสมบัตินั้น |
| | เอกสารสิทธิ์ถูกแก้ไข | แจ้งข้อความบ่งบอกถึงความผิดพลาดของเอกสารสิทธิ์ หรือโปรแกรมแฮงค์ |
| | โปรแกรมถูกแก้ไข (แก้ไขไฟล์ EXE) | โปรแกรมจะเกิดเหตุการณ์ตามความผิดพลาดของ EXE เช่น ขนาดผิดพลาด, CRC ผิดพลาด |
| | โปรแกรมถูกเรียกใช้งานโดยไม่มีไฟล์เอกสารสิทธิ์ | แจ้งข้อความเอกสารสิทธิ์ไม่ถูกต้อง |
| เอกสารสิทธิ์แบบล็อกกับเครื่อง | คัดลอกไฟล์เอกสารสิทธิ์มาจากที่อื่น ไม่ได้ขอโดยตรง | แจ้งข้อความเอกสารสิทธิ์ไม่ถูกต้อง |

*ตารางที่ 5-2 แสดงผลการทดสอบการทำงานของโครงงาน*

# บทที่ 6

## บทวิจารณ์และสรุปของโครงงาน

### 6.1 ผลที่ได้รับจากโครงงาน

- ทฤษฎี หลักการ และความรู้พื้นฐานสำหรับพัฒนาระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์
- แนวทางและความรู้เชิงเทคนิคในการพัฒนาระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์
- คอมโพเนนท์ที่อำนวยความสะดวกในการป้องกันซอฟต์แวร์ให้สามารถบริหารและควบคุมสิทธิ์การใช้งานได้
- ฟังก์ชันในระดับล่างที่ใช้อ่านค่ารหัสหมายเลขประจำเครื่องจากแหล่งต่างๆ เพื่อป้องกันซอฟต์แวร์ให้ใช้ได้เฉพาะเครื่อง ดูรายละเอียดในภาคผนวก ง

### 6.2 ความสามารถของระบบพัฒนาขึ้นเมื่อแยกพิจารณาตามลักษณะที่สำคัญ

เนื่องจาก ผู้จัดทำโครงงานนี้ ไม่สามารถนำระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่มีอยู่ในปัจจุบันมาทดสอบการทำงาน เพื่อเปรียบเทียบผลได้ จึงแสดงความสามารถของระบบโดยแยกพิจารณาตามลักษณะที่สำคัญของระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ ผลการพิจารณาแสดงดังตารางที่ 6 – 1

| ลักษณะที่พิจารณา | ความสามารถของระบบ |
|---|---|
| แบบจำลองควบคุมสิทธิ์ | 1. แบบทดลองใช้งาน (โดยการกำหนดวันหมดอายุการใช้งาน)<br>2. แบบกำหนดคุณสมบัติการใช้งาน<br>3. แบบกำหนดวันหมดอายุการใช้งาน<br>4. ผสมกันระหว่างแบบต่างๆ ข้างต้น |

| | |
|---|---|
| ความปลอดภัย | 1. ใช้ RSA Public Key Cryptography โดยผ่านทาง Microsoft CryptoAPI 1.0<br>- Microsoft Cryptographic Service Provider 1.0<br>- Hash Algorithm MD2, MD4 และ MD5 128 บิต<br>- 512 บิต Public Key<br>- 40 บิต Session Key<br>2. เลือกป้องกันเฉพาะเจาะจงเครื่องได้โดยสามารถอ่านค่ารหัสเครื่องได้จาก 4 แหล่ง หรือผสมกันก็ได้<br>- Ethernet Address (ไม่ซ้ำ)<br>- Harddisk ID. (ไม่ซ้ำ)<br>- Processor Serial Number (Intel Pentium III) (ไม่ซ้ำ)<br>- Logical Disk Serial Number (มีโอกาสซ้ำ) |
| ความน่าเชื่อถือ | มีความน่าเชื่อถือในระดับปานกลาง เนื่องจากเป็นระบบต้นแบบที่เพิ่งพัฒนาขึ้น |
| ความเข้ากันได้กับซอฟต์แวร์ที่ต้องการควบคุมสิทธิ์ | ตอนนี้ยังจำกัดเฉพาะ Borland Delphi 5.0 เนื่องจากได้ทำการสร้างและพัฒนาระบบในระดับคอมโพเนนท์ |
| ความสะดวกรวดเร็วในการใช้งาน | สามารถสร้างซอฟต์แวร์ให้ได้รับการควบคุมสิทธิ์ได้อย่างรวดเร็ว เนื่องจากทำงานในระดับคอมโพเนนท์ โดยอาจจะไม่จำเป็นต้องเขียนโค้ดเพิ่มเติม |
| ผลกระทบของระบบต่อประสิทธิภาพของซอฟต์แวร์ | - ปกป้องการแก้ไขซอฟต์แวร์โดยใช้ CRC32 ซึ่งสามารถทำงานได้เร็ว<br>- การถอดรหัสและตรวจสอบเอกสารสิทธิ์จะทำเป็นคาบเวลา<br>- ประสิทธิภาพจะแปรผกผันกับจำนวนฟีเจอร์และแอ็กชันที่ถูกควบคุมสิทธิ์ในซอฟต์แวร์ |
| ระบบปฏิบัติการที่รองรับ | MS Windows 9x (Internet Explorer 3.0 ขึ้นไป)<br>MS Windows NT 4.0<br>MS Windows 2000 |

*ตารางที่ 6-1 แสดงความสามารถของระบบที่พัฒนาขึ้นโดยพิจารณาตามลักษณะที่สำคัญ*

**6.3 ข้อจำกัดของซอฟต์แวร์**

- ระบบยังไม่มีฟังก์ชันการทำงานในระดับล่างที่ชัดเจน เนื่องจากระบบถูกพัฒนาในระดับคอมโพเนนท์ โดยเรียกใช้งานฟังก์ชันของ Microsoft CryptoAPI 1.0 เป็นหลัก จึงไม่สามารถนำไปใช้งานในโปรแกรมภาษาอื่นได้

- ระบบยังขาดความสามารถในการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ผ่านระบบเน็ตเวิร์กหรือระบบอินเตอร์เน็ตได้เองโดยอัตโนมัติ
- การควบคุมสิทธิ์การใช้งานซอฟต์แวร์ จะต้องทำตั้งแต่พัฒนาซอฟต์แวร์ (โดยมีโค้ดของซอฟต์แวร์) ไม่สามารถทำการควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่ผ่านการคอมไพล์แล้ว

**6.4 สิ่งที่สามารถพัฒนาต่อเนื่อง**

จากข้อจำกัดของระบบที่พัฒนาขึ้น ในหัวข้อที่ 6.3 ระบบนี้สามารถพัฒนาปรับปรุงต่อเนื่องในส่วนต่างๆ ที่สำคัญดังนี้

- Licensing APIs เป็นฟังก์ชันในระดับล่างที่อำนวยความสะดวกในการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ เพื่อให้ระบบสามารถทำงานร่วมกับโปรแกรมภาษาอื่นๆ ที่สนับสนุนการอินเตอร์เฟสกับฟังก์ชันระดับล่างได้
- Licensing Server เพิ่มขีดความสามารถให้ระบบสามารถบริหารและควบคุมสิทธิ์การใช้งานในระบบเน็ตเวิร์กหรือระบบอินเตอร์เน็ตได้ ซึ่งจะทำให้สามารถนำไปประยุกต์ใช้งานในการสร้างระบบที่เกี่ยวข้อง เช่น ระบบขายซอฟต์แวร์บนอินเตอร์เน็ต หรือระบบการบริการซอฟต์แวร์บนอินเตอร์เน็ต เป็นต้น
- เครื่องมือที่ช่วยในการควบคุมสิทธิ์ของซอฟต์แวร์โดยไม่จำเป็นต้องการแก้ไขโค้ดของซอฟต์แวร์ เพื่อให้สามารถบริหารและควบคุมสิทธิ์การใช้งานได้กับซอฟต์แวร์ไม่ว่าจะพัฒนามาจากภาษาใดก็ตาม ที่มีรูปแบบของไฟล์เอ็กซีคิวที่เหมือนกัน

**6.5 บทสรุปของโครงงาน**

โครงงานนี้ได้นำเสนอถึงความสำคัญ, ทฤษฎี, และหลักการในการพัฒนาระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ เนื่องจากระบบการบริหารและควบคุมสิทธิ์การใช้งานที่สมบูรณ์ประกอบด้วยองค์ประกอบหลายส่วนที่ทำงานร่วมกัน โครงงานนี้ไม่สามารถนำเสนอระบบการบริหารและควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่สมบูรณ์แบบ แต่นำเสนอตัวอย่างระบบต้นแบบที่สามารถทำงานได้จริง เพื่อพิสูจน์แนวคิดและหลักการทำงานของระบบที่ได้ออกแบบ

# ภาคผนวก ก

## คอมโพเนนต์ควบคุมสิทธิ์การใช้งานซอฟต์แวร์ที่สร้างขึ้นจากโครงงาน

คอมโพเนนท์ด้านการบริหารและควบคุมสิทธิ์ต่างๆ และหน้าที่การทำงานของแต่ละคอมโพเนนท์โดยสรุปมีดังต่อไปนี้

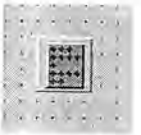
- ตัวจัดการรหัสความปลอดภัย (TCryptoManager) เป็นคอมโพเนนท์ที่ทำหน้าที่ในการจัดการรหัสความปลอดภัยเช่น การเข้ารหัสข้อมูล การถอดรหัสข้อมูล การสร้างลายเซ็นต์กำกับข้อมูล

*รูปแสดงคอมโพเนนท์ TCryptoManager*

- ตัวจัดการคีย์สาธารณะที่ใช้ในการสร้างรหัสความปลอดภัย (TPublicKey) เป็นคอมโพเนนท์ที่ทำหน้าในการจัดการเก็บคีย์สาธารณะที่ใช้ในการเข้ารหัสข้อมูลของเอกสารแสดงสิทธิ์การใช้งานซอฟต์แวร์ หรือ คีย์สาธารณะที่ใช้ในการพิสูจน์ลายเซ็นดิจิตอลให้ฝังอยู่กับตัวแอพพลิเคชัน

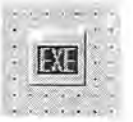
*รูปแสดงคอมโพเนนท์ TPublicKey*

- ตัวตั้งค่าสิทธิ์การใช้งาน (TLicenseConfig) เป็นคอมโพเนนท์ที่ทำหน้าที่ในการเก็บนำค่าต่างๆ ที่เกี่ยวกับรายละเอียดของลูกค้าและขอบเขตอำนาจสิทธิ์การใช้งานซอฟต์แวร์แล้วเขียนลงไฟล์เอกสารร้องขอสิทธิ์ จากนั้นฝั่งที่ทำหน้าที่ในการอนุมัติสิทธิ์ คอมโพเนนท์ดังกล่าวจะทำหน้าที่ในการอ่านไฟล์เอกสารร้องขอสิทธิ์ขึ้นมาเก็บไว้ ตรวจเช็คความถูกต้องหรืออาจจะมีการปรับแต่งเพิ่ม จากนั้นฝั่งนี้จะทำการอนุมัติไฟล์เอกสารร้องขอสิทธิ์ให้เป็นไฟล์เอกสารสิทธิ์ที่มีสิทธิ์ถูกต้องในการใช้งานซอฟต์แวร์



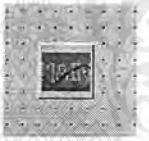
*รูปแสดงคอมโพเนนท์ TLicenseConfig*

- ตัวจัดการปกป้องไฟล์เอ็กซีคิว (TProtectEXE) เป็นคอมโพเนนท์ที่ทำหน้าที่ในการตรวจเช็คการแก้ไขไฟล์ที่จะนำไปเอ็กซีคิว ว่ามีการเปลี่ยนแปลงแก้ไขส่วนหนึ่งส่วนใดหรือเปล่า หากมีการแก้ไขก็จะทำให้ไฟล์ดังกล่าวไม่สามารถนำไปใช้งานได้อีกต่อไป



*รูปแสดงคอมโพเนนท์ TProtectEXE*

- ตัวจัดการตรวจเช็คความถูกต้องของเอกสารแสดงสิทธิ์หลัก (TMainVerify) เป็นคอมโพเนนท์ที่ทำหน้าที่ในการ ตรวจเช็คหาไฟล์เอกสารสิทธิ์ พร้อมทั้งโหลดไฟล์ดังกล่าวมาเก็บไว้ในตัวตั้งค่าสิทธิ์ในการใช้งานในตอนเริ่มรันแอพพลิเคชัน และจะทำการตรวจเช็คดูว่าเอกสารสิทธิ์ดังกล่าวถูกต้องหรือเปล่า เงื่อนไขการใช้งานซอฟต์แวร์ยังคงเป็นไปตามเงื่อนไขที่ได้ระบุเอาไว้ในเอกสารแสดงสิทธิ์หรือไม่ จากนั้นจะทำการเช็คความถูกต้องของเอกสารสิทธิ์เป็นช่วงเวลาตามที่ได้มีการกำหนดไว้ตามความเหมาะสม



*รูปแสดงคอมโพเนนท์ TMainVerify*

- ตัดจัดการตรวจเช็คสิทธิ์ในการใช้งานคุณสมบัติต่างๆ ของแอพพลิเคชัน(TActionsVerify) เป็นคอมโพเนนท์ที่ทำหน้าที่ในการตรวจเช็คความถูกต้องของคุณสมบัติต่างๆของแอพพลิเคชัน ในขณะที่มีการเรียกใช้คุณสมบัตินั้นๆ ว่าควรจะอนุญาตให้ใช้ ไม่ควรอนุญาตให้ใช้ หรืออนุญาตให้ใช้แต่จำกัดขอบไว้เพียงใดซึ่งก็ขึ้นอยู่กับเอกสารสิทธิ์การใช้งานซอฟต์แวร์

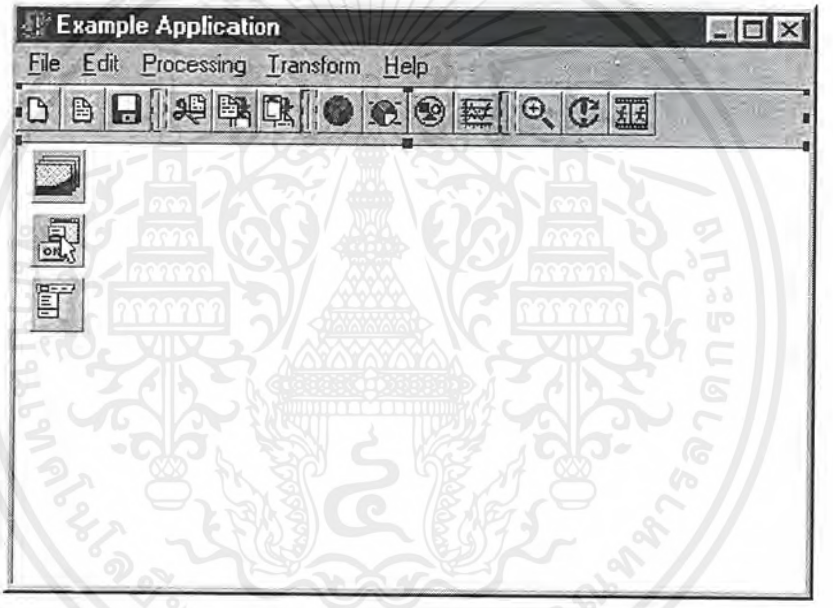
*รูปแสดงคอมโพเนนท์ TActionsVerify*

# ภาคผนวก ข

## ขั้นตอนการสร้างตัวอย่างสำหรับการทดสอบระบบ

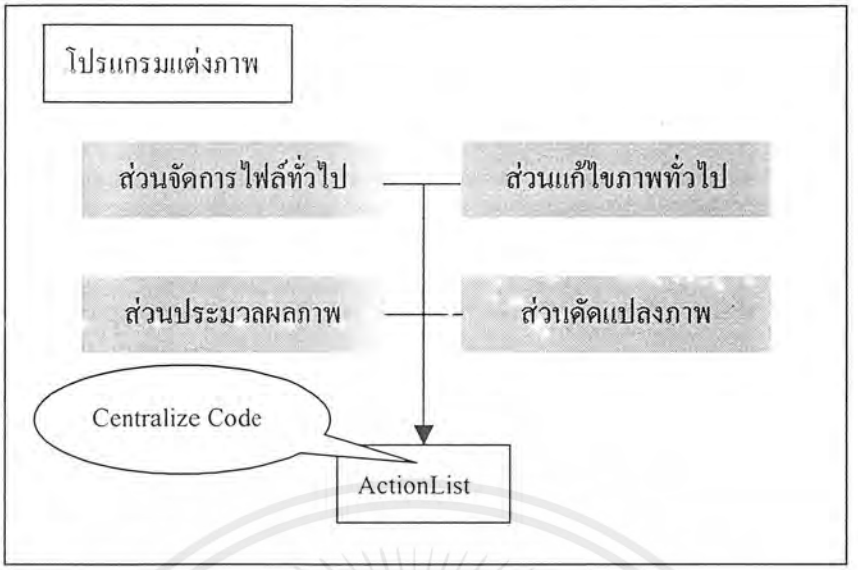
จากคอมโพเนนท์ต่างๆ ที่ได้สร้างขึ้นทั้ง 6 คอมโพเนนท์สามารถนำไปประยุกต์ใช้งานได้ดังตัวอย่าง โดยตัวอย่างแรกจะเป็นการสร้างแอพพลิเคชันในรูปแบบที่ให้ลูกค้าทดลองนำไปใช้งาน (Demo Version) จากนั้นจะเป็นตัวอย่างการสร้างแอพพลิเคชันในรูปแบบที่จำหน่ายให้ลูกค้านำไปใช้งาน โดยใช้ได้เฉพาะ เครื่องคอมพิวเตอร์ที่ลูกค้าใช้อยู่ กล่าวคือจะป้องกันซอฟต์แวร์ดังกล่าวเข้ากับเครื่องของลูกค้า ลูกค้าจะไม่สามารถนำซอฟต์แวร์ดังกล่าวไปติดตั้งเพื่อใช้งานกับเครื่องอื่นได้

**1. ตัวอย่างการสร้างซอฟต์แวร์ทดลองใช้งาน (Demo Version)**



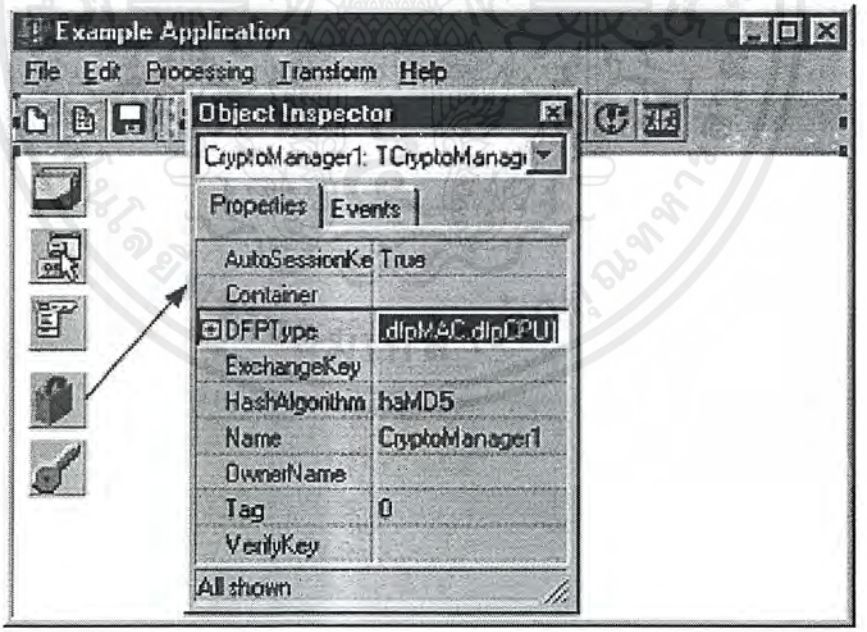

*รูปแสดงฟอร์มหลักของโปรแกรมที่สร้างขึ้นเป็นตัวอย่างการทดสอบ*

โปรแกรมตัวอย่างที่สร้างขึ้นมาเพื่อทดสอบการทำงานของโครงงาน เป็นโปรแกรมตกแต่งภาพ ซึ่งประกอบไปด้วยคุณสมบัติหลักดังนี้ คือ การจัดการไฟล์ทั่วไป การแก้ไขภาพทั่วไป การประมวลผลภาพ และการดัดแปลงภาพ ต่อจากนั้นจะทำการ Centralize Code ของแต่ละฟังก์ชันย่อยๆ โดยใช้ ActionList



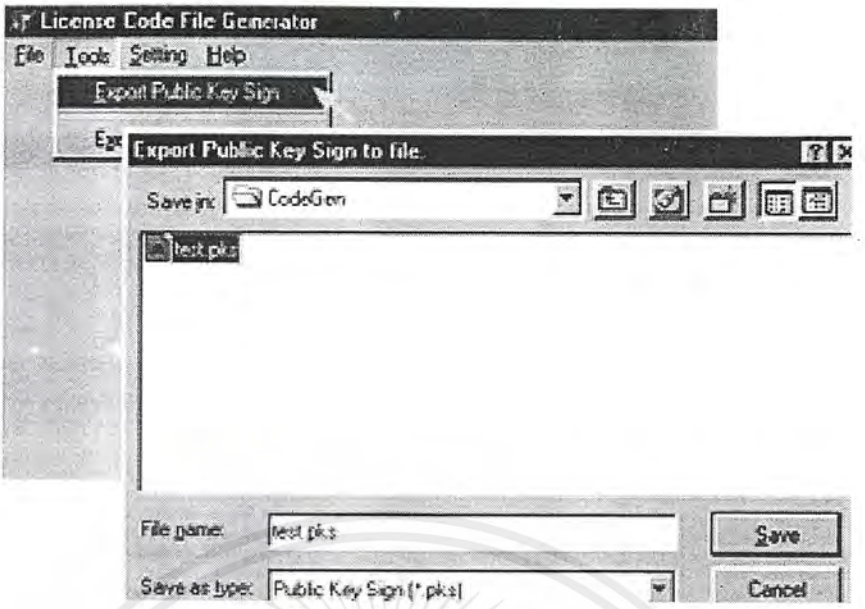

*รูปแสดงการ Centralize Code ของโปรแกรมโดยใช้ ActionList*

จากนั้นจะเพิ่มคอมโพเนนท์ TCryptoManager และ TPublicKey เพื่อที่จะนำ คีย์สาธารณะที่ใช้ในการพิสูจน์ลายเซ็นต์มาฝังเก็บเอาไว้กับตัวโปรแกรม



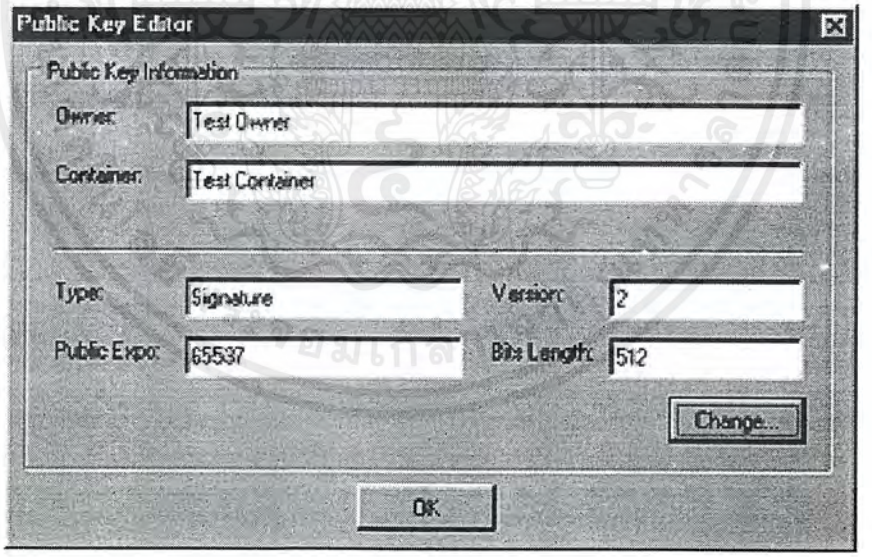

*รูปแสดงฟอร์มหลักหลังจากเพิ่มคอมโพเนนท์ TCryptoManager และ TPublicKey*

จากนั้นจะใช้โปรแกรมในการสร้างเอกสารสิทธิ์ดึงเอาคีย์คีย์สาธารณะที่ใช้ในการพิสูจน์ลายเซ็นต์ออกมาเก็บไว้ในไฟล์



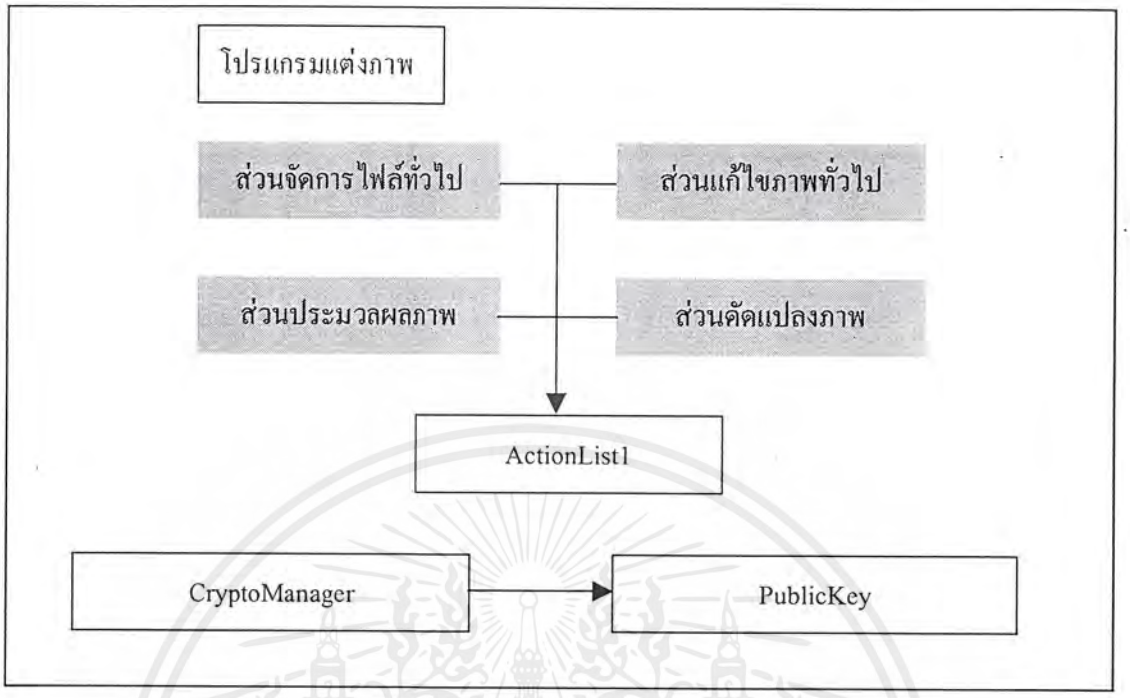

*รูปแสดงการดึงเอาคีย์สาธารณะที่ใช้สำหรับการพิสูจน์ลายเซ็นต์จากฝั่งสร้างเอกสารสิทธิ์ออกมา*

ต่อมาทำการนำเอาคีย์สาธารณะดังกล่าวไปเก็บไว้ในคอมโพเนนท์ TPublicKey เพื่อที่จะฝังเอาไว้กับตัวโปรแกรม และใช้ในการตรวจสอบลายเซ็นดิจิตอลในทุกๆ ครั้งที่ต้องการตรวจสอบเอกสารสิทธิ์



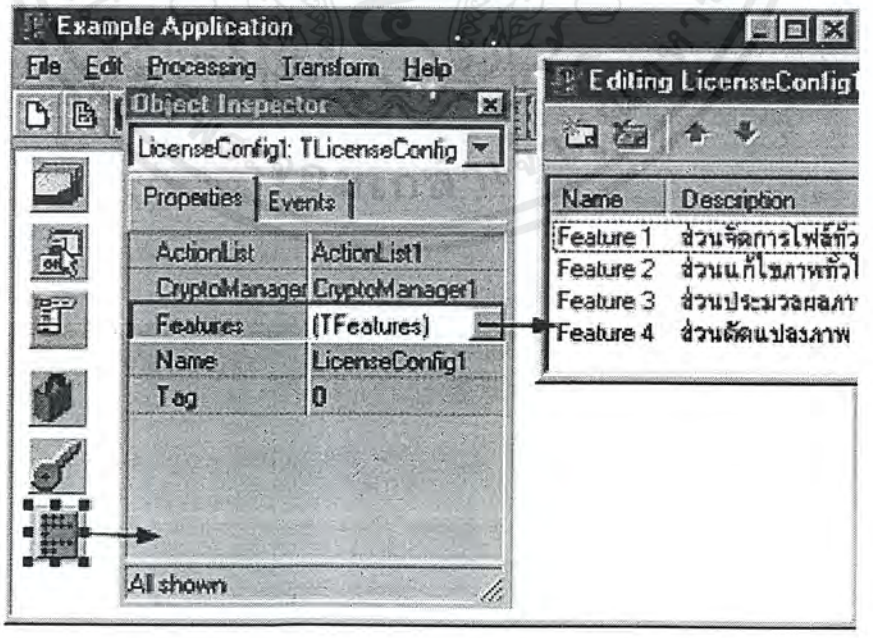

*รูปแสดงการนำเอาเอาคีย์สาธารณะไปเก็บไว้ในคอมโพเนนท์ TPublicKey*

เราสามารถแสดงความสัมพันธ์ระหว่างคอมโพเนนท์ TCryptoManager และ TPublicKey ได้ดังนี้ คือ คีย์สาธารณะที่เก็บเอาไว้ที่คอมโพเนนท์ TPublicKey จะถูกนำไปใช้งานในการพิสูจน์ลายเซ็นต์โดยคอมโพเนนท์ TCryptoManager

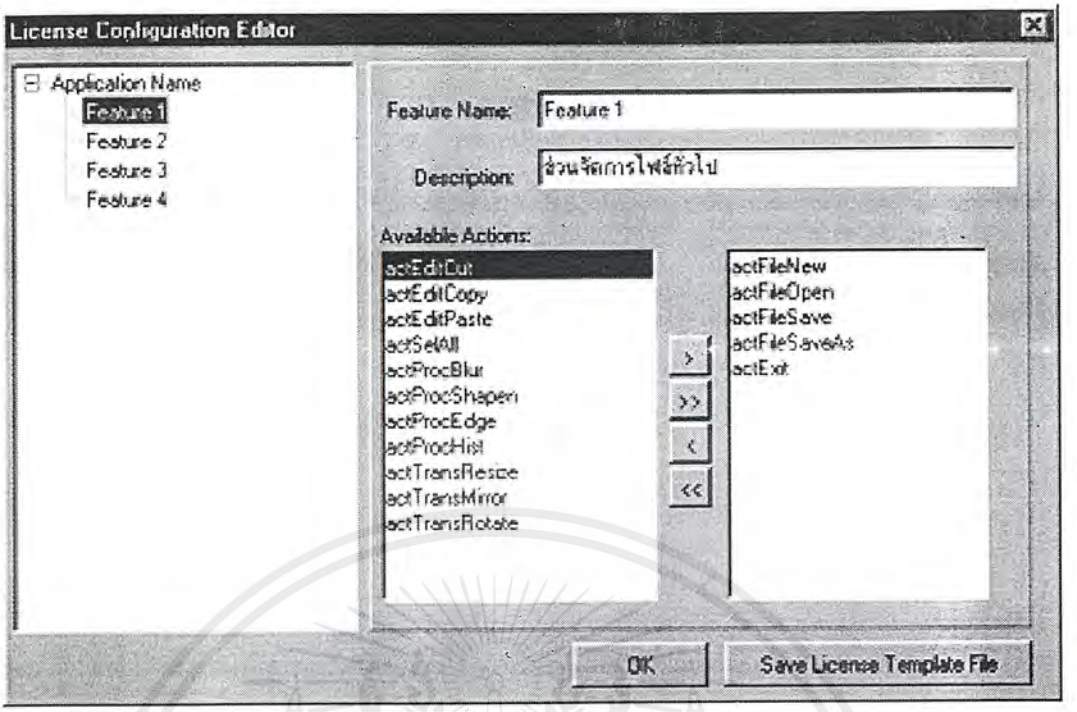



*รูปแสดงความสัมพันธ์ระหว่างคอมโพเนนท์ CryptoManager และ คอมโพเนนท์ PublicKey*

จากนั้นจะเพิ่มคอมโพเนนท์ TLicenseConfig เพื่อสร้าง Feature สำหรับแบ่งฟังก์ชันออกเป็นหมวดหมู่



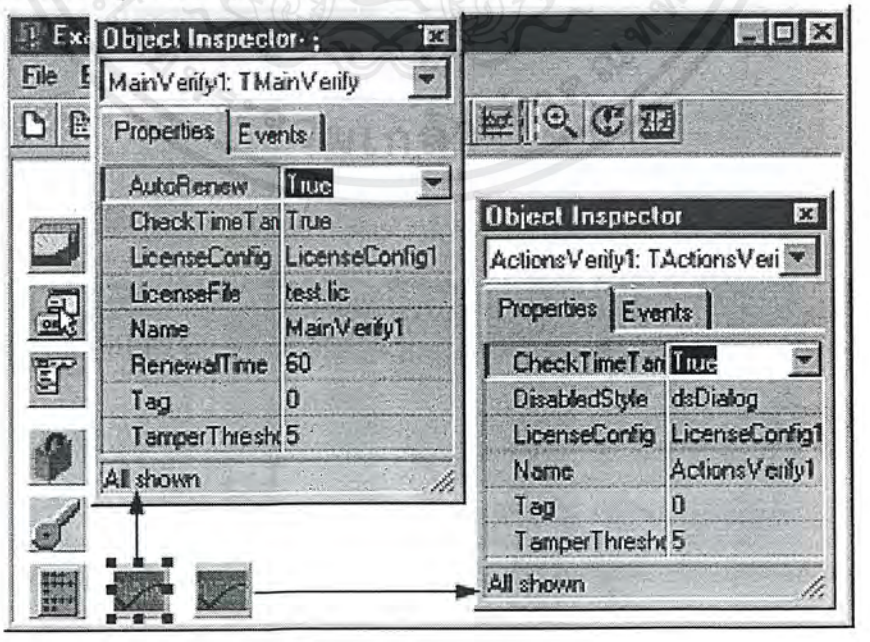

*รูปแสดงการสร้าง Feature สำหรับแบ่งฟังก์ชันออกเป็นหมวดหมู่*




*รูปแสดงการจัดแบ่งคุณสมบัติย่อยๆ ให้กับ Feature ต่างๆ โดยใช้ Component Editor ของ TLicenseConfig*

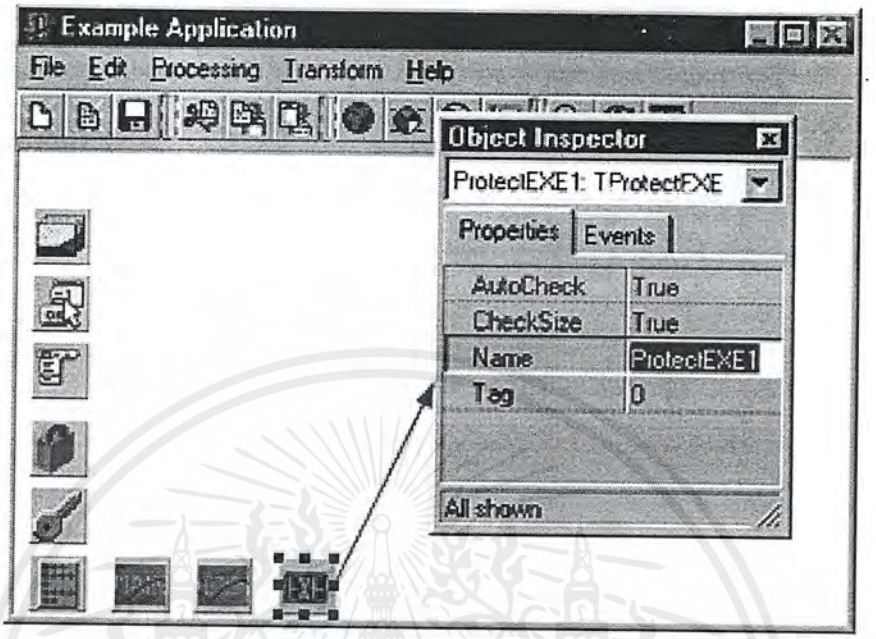
หลังจากที่ได้จัดแบ่ง Actions ต่างๆ ให้กับ Feature แล้ว จะต้องทำการ Save License Template File เพื่อเก็บไว้สำหรับเป็นโครงร่างในการสร้างไฟล์เอกสารสิทธิ์แบบทดลองใช้งาน (Demo Version) ให้กับโปรแกรม

จากนั้นก็ทำการเพิ่มคอมโนเนนท์ TMainVerify และ TActionsVerify ซึ่งเป็นคอมโพเนนท์ที่ใช้ในการตรวจสอบความถูกต้องของสิทธิ์การใช้งานซอฟต์แวร์



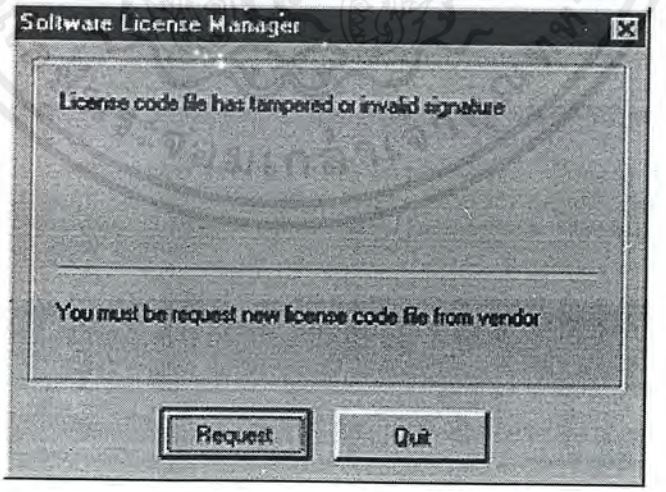

*รูปแสดงการตั้งค่าต่างๆ ให้กับ TMainVerify และ TActionsVerify*

คอมโพเนนท์สุดท้ายที่วางลงไปในฟอร์มหลักคือ TProtectEXE แสดงดังรูป




*รูปแสดงการตั้งค่าให้กับ TProtectEXE*

เมื่อเพิ่มคอมโพเนนท์ดังกล่าวลงในฟอร์มหลักเรียบร้อย ก็สามารถทดลองรันโปรแกรมดังกล่าว จะปรากฏหน้าจอดังรูป เนื่องจากโปรแกรมยังไม่มีไฟล์เอกสารแสดงสิทธิ์การใช้งาน




*รูปแสดงข้อความเมื่อรันโปรแกรมครั้งแรก*

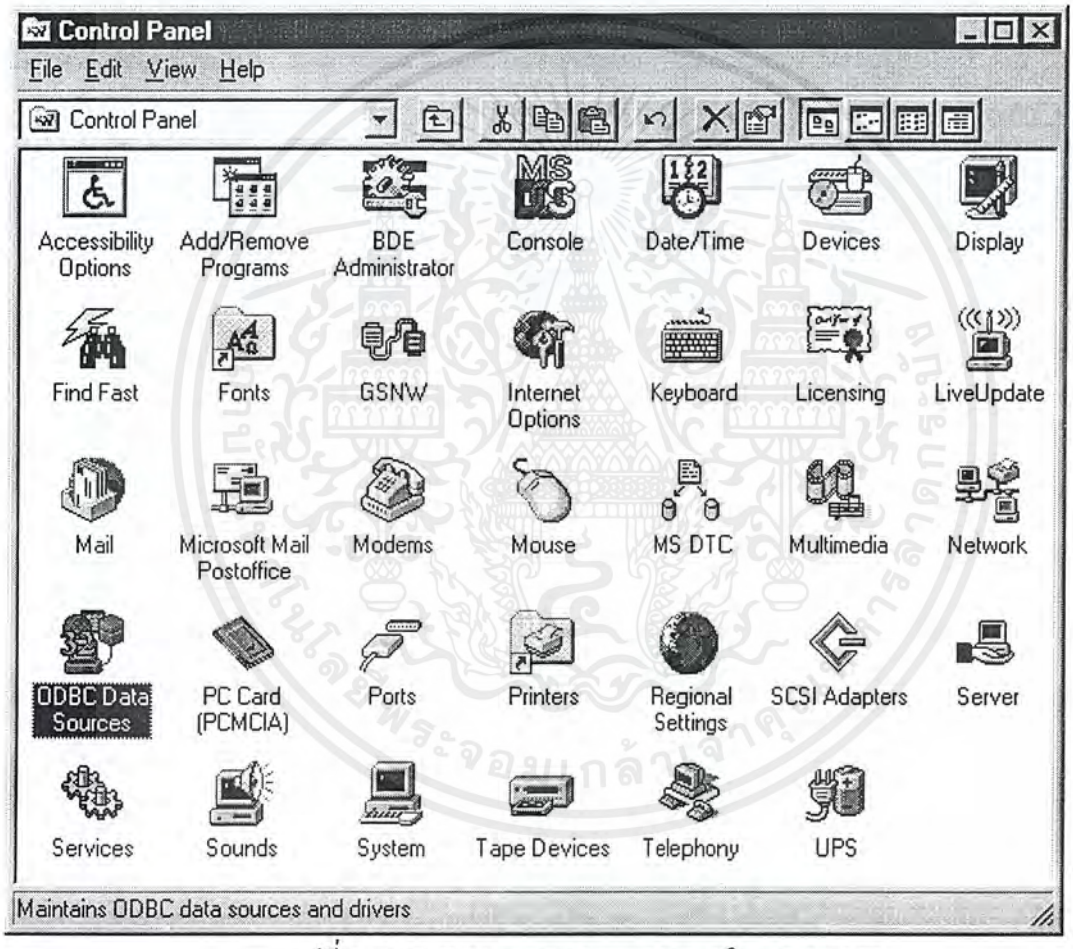
## ภาคผนวก ข.

## การ Setup ODBC

### การ Setup ODBC บน Server ที่เป็นระบบ WindowNT

การติดต่อกับ Database ที่เราสร้างขึ้นมีหลายวิธีด้วยกันแต่วิธีที่ง่ายที่สุดก็คือการติดต่อโดยผ่าน ODBC ซึ่งมีขั้นตอนดังต่อไปนี้

1. ทำการเปิด Folder ที่ชื่อ Control panel ขึ้นมาหลังจากนั้นดับเบิ้ลคลิ๊กไปที่ไอคอนของ ODBC ที่มีชื่อว่า ODBC Data Sources ดังรูป



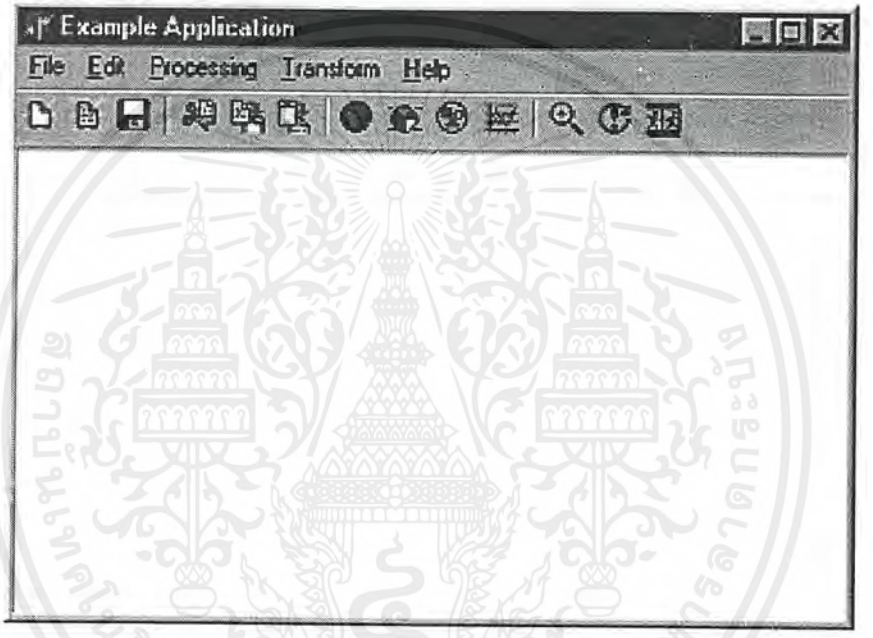

*รูปที่ 1 แสดง Folder ของ Control Panel ใน windowNT*

2. หลังจากที่เราดับเบิ้ลคลิ๊กที่ไอคอนแล้วจะขึ้นไดอะล็อค ดังรูป เพื่อรอการตั้งค่าต่างๆ

จากรูปได้กำหนดให้โปรแกรมหมดอายุการใช้งานวันที่ 27 เมษายน 2543 และกำหนดไม่อนุญาตให้ใช้งานคุณสมบัติการประมวลผลภาพได้

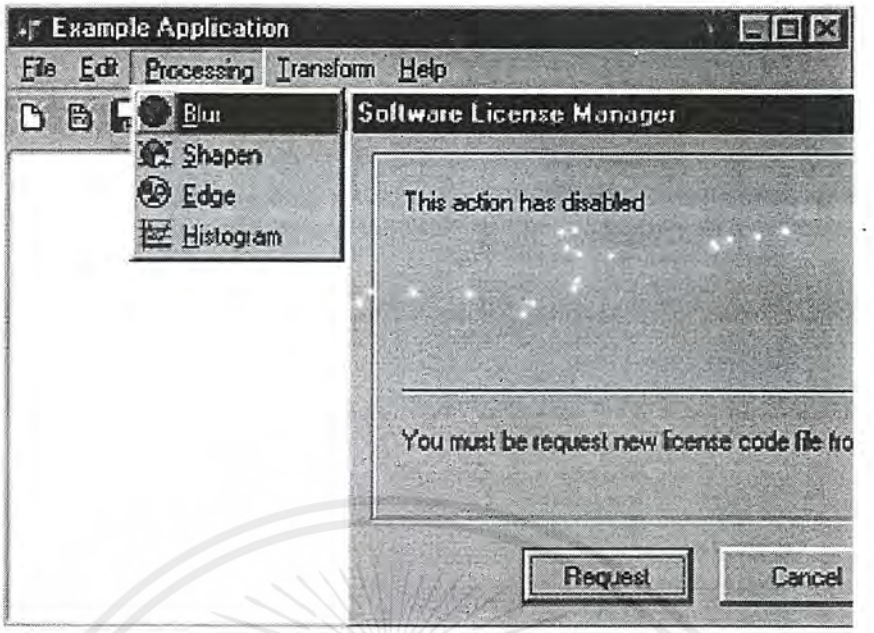
หลังจากตั้งค่าให้กับเอกสารสิทธิ์จนเป็นที่พอใจก็ทำการสร้างเอกสารสิทธิ์โดย File->Generate License File แล้วบันทึกในชื่อ test.lic (ชื่อไฟล์จะต้องตรงกับที่กำหนดในคอมโพเนนท์ TMainVerify) ให้อยู่ในโฟลเดอร์เดียวกันกับโปรแกรม

เป็นอันเสร็จสิ้นขั้นตอนในการพัฒนาแอพพลิเคชันให้มีการจัดการสิทธิ์การใช้งานแบบทดลองใช้งาน ขั้นตอนต่อไปของการทดลอง คือจะแจกจ่ายซอฟต์แวร์ดังกล่าวพร้อมเอกสารสิทธิ์ไปให้แก่ผู้ใช้งานเมื่อผู้ใช้งานได้รับก็จะเริ่มเรียกใช้งานโปรแกรมที่ถูกเรียกขึ้นมาจะมีลักษณะดังนี้




*รูปแสดงโปรแกรมแต่งภาพที่ถูกเรียกใช้งานโดยใช้เอกสารสิทธิ์แบบทดลองใช้งาน*

เมื่อผู้ใช้เรียกใช้งานคุณสมบัติไม่อนุญาตให้ใช้งานได้ จะแสดงข้อความดังรูป



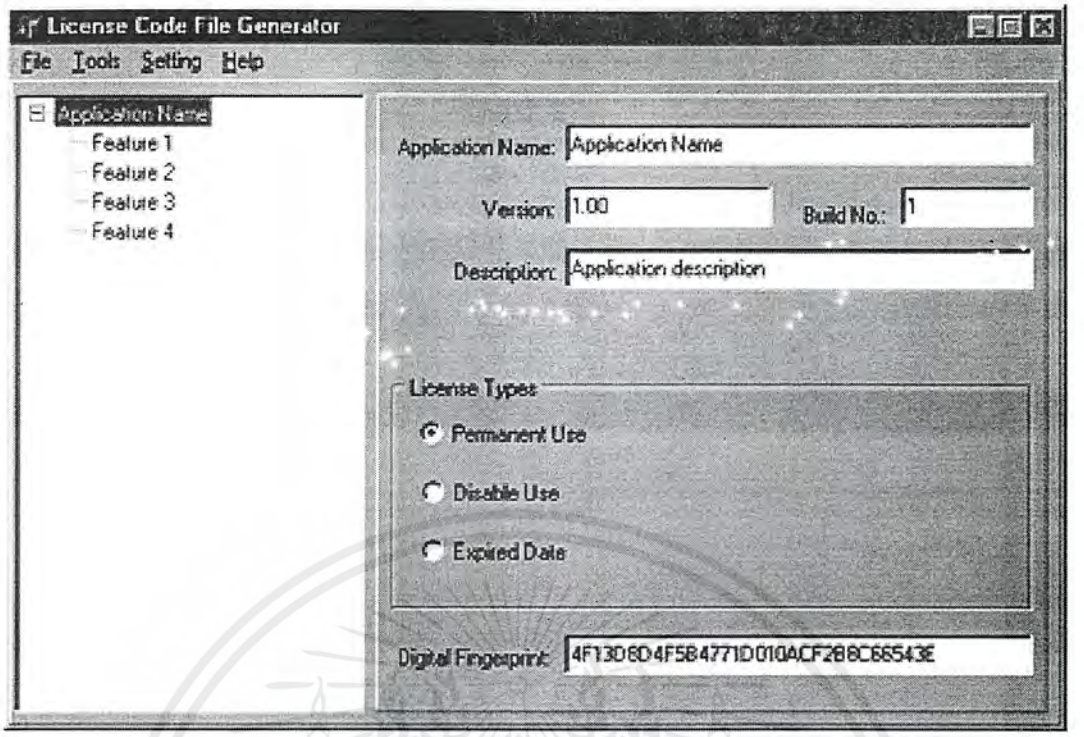

*รูปแสดงข้อความเมื่อผู้ใช้เรียกใช้งานคุณสมบัติที่ไม่อนุญาตให้ใช้งาน*

หากมีการแก้ไขเวลาของระบบโดยการตั้งเวลาถอยหลัง เพื่อยืดอายุการใช้งานโปรแกรม จะแสดงข้อความไม่อนุญาตให้โปรแกรมสามารถทำงานต่อได้ดังรูป


*รูปแสดงข้อความเตือนกรณีที่มีการแก้ไขเวลาของระบบย้อนหลังและจะหยุดการทำงานของโปรแกรม*

**2. ตัวอย่างการควบคุมสิทธิ์การใช้งานซอฟต์แวร์แบบล็อกกับเครื่อง (Node Locked)**

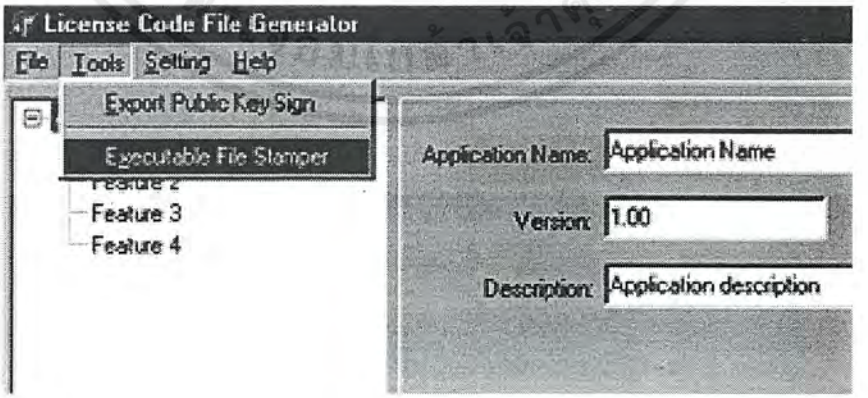
มีลักษณะการพัฒนาคล้ายกับการพัฒนาซอฟต์แวร์แบบทดลองใช้แต่จะต่างกันตรงที่ ในฝั่งของผู้ใช้งานจะมีการสร้างไฟล์ร้องขอเอกสารสิทธิ์และส่งไปให้กับฝั่งผู้สร้างเอกสารสิทธิ์ โดยในไฟล์ร้องของเอกสารสิทธิ์ของฝั่งผู้ใช้จะต้องส่งข้อมูลที่เฉพาะเจาะจงของเครื่องผู้ใช้ด้วยเช่น ฮาร์ดดิสก์ซีเรียลนัมเบอร์, ลอจิคอลดิสก์นัมเบอร์, อีเทอร์เน็ตแอดเดรส, โปรเซสเซอร์ซีเรียลนัมเบอร์ หรือ คีย์สาธารณะที่ใช้ในการแลกเปลี่ยนคีย์เข้ารหัส ของลูกค้าด้วยเพื่อที่จะให้ ฝั่งที่สร้างเอกสารสิทธิ์นำข้อมูลดังกล่าวไปใช้ในกระบวนการเข้ารหัสเอกสารสิทธิ์ด้วย เมื่อทางฝั่งผู้สร้างเอกสารสิทธิ์ได้รับไฟล์ร้องขอเอกสารสิทธิ์ก็สามารถนำมาสร้างเอกสารสิทธิ์การใช้งานแบบล็อกกับเครื่องส่งไปให้ผู้ใช้ ซึ่งแสดงการสร้างเอกสารสิทธิ์ดังรูป




*รูปแสดงการสร้างเอกสารสิทธิ์แบบล็อกให้ใช้งานได้เฉพาะเครื่องจาก License Request File ที่ได้จากผู้ใช้*

เมื่อผู้ใช้คัดลอกเอกสารสิทธิ์ดังกล่าวไปใช้งานในเครื่องอื่น ที่ไม่ใช่เครื่องที่ขอเอกสารสิทธิ์ โปรแกรมจะไม่สามารถทำงานได้ โดยจะแจ้งว่าเอกสารสิทธิ์ที่มีอยู่ไม่ถูกต้อง

ในกรณีที่ต้องการป้องกันการแก้ไขไฟล์ EXE ของโปรแกรมจะต้องทำการฝังขนาดและ CRC ของไฟล์ลงในโปรแกรมด้วย โดยใช้เครื่อง Executable File Stamper แสดงดังรูป




*รูปแสดงการเรียกใช้งานเครื่องมือ Executable File Stamper*

โดยการฝังขนาด และ CRC ของไฟล์ ลงในไฟล์ EXE ของโปรแกรมจะทำได้เฉพาะโปรแกรมที่ได้เพิ่มคอมโพเนนท์ TProtectEXE ลงไปแล้ว ซึ่งคอมโพเนนท์ดังกล่าวจะทำหน้าที่ในการจองตำแหน่งสำหรับฝังขนาดและ CRC ของไฟล์ แล้วทำการตรวจสอบเมื่อโปรแกรมถูกรัน

ในการฝังขนาดและ CRC อาจจะเพิ่มความปลอดภัยให้มากขึ้นโดยการเพิ่มออปชั่น Erase Marker เพื่อลบ Marker สำหรับ Stamp ขนาดและ CRC แต่โปรแกรมที่ถูก Stamp โดยวิธีดังกล่าว จะไม่สามารถลบ หรือ Stamp ขนาดและ CRC ใหม่ได้


*รูปแสดงเครื่องมือสำหรับการฝังขนาดและ CRC ลงภายในไฟล์ EXE ของโปรแกรม*

# ภาคผนวก ค

## Microsoft CryptoAPI 1.0 Function Reference

ฟังก์ชันของไมโครซอฟท์ที่สนับสนุนการเขียนโปรแกรมในการสร้างรหัสความปลอดภัยให้กับข้อมูลสามารถแบ่งกลุ่มตามรูปแบบการใช้ได้ดังนี้

- กลุ่มฟังก์ชันคอนเท็กซ์ (Context Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่ติดต่อกับส่วนที่ให้บริการการสร้างรหัสความปลอดภัยให้กับข้อมูลโดยสามารถเลือกส่วนที่ให้บริการดังกล่าวจากผู้ให้บริการต่างๆได้หลากหลาย ตามความเหมาะสมและระดับความปลอดภัยที่ต้องการ
- กลุ่มฟังก์ชันสร้างคีย์ (Key Generation Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่สร้างคีย์ที่จะใช้กับแอพพลิเคชันในการสร้างความปลอดภัย
- กลุ่มฟังก์ชันแลกเปลี่ยนคีย์ (Key Exchange Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่จัดการทางด้านการแลกเปลี่ยนคีย์ ให้มีความปลอดภัย เช่น ฟังก์ชันที่ใช้ในการนำคีย์เข้าหรือออกจาก CSPs
- กลุ่มฟังก์ชันเข้ารหัสและถอดรหัสข้อมูล (Data Encryption Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่ในการเข้ารหัสและถอดรหัสข้อมูล
- กลุ่มฟังก์ชันแฮช, สร้างลายเซ็นต์ดิจิตอล และตรวจสอบลายเซ็นต์ (Hashing and Signature Functions) เป็นกลุ่มฟังก์ชันที่ทำหน้าที่ในการแฮชและสร้างลายเซ็นต์ดิจิตอลให้กับข้อมูล รวมทั้งการตรวจสอบลายเซ็นต์ดิจิตอล

โดยในแต่ละกลุ่มของฟังก์ชันจะประกอบด้วยฟังก์ชันและหน้าที่โดยสรุป ดังต่อไปนี้

| ฟังก์ชัน | คำอธิบาย |
|---|---|
| Context Funtions | |
| CryptAcquireContext | Acquires a handle to the current user's key container within a particular CSP. |
| CryptGetProvParam | Retrieves attributes of a CSP. |
| CryptReleaseContext | Releases the handle acquired by CryptAcquireContext |
| CryptSetProvider | Specifies the user default CSP for a particular CSP type. |
| CryptSetProvParam | Specifies attributes of a CSP. |
| Key Generation Functions | |
| CryptDeriveKey | Create a key derived from a password. |
| CryptGenKey | Create a random key. |

| ฟังก์ชัน | คำอธิบาย |
|---|---|
| Key Exchange Functions | |
| CryptDestroyKey | Destroy a key. |
| CryptExportKey | Transfer a key from the CSP into a key blob in the application's memory space. |
| CryptGenRandom | Generate random data. |
| CryptGetKeyParam | Retrieve a key's parameters. |
| CryptGetUserKey | Get a handle to the key exchange or signature key. |
| CryptImportKey | Transfer a key from a key blob to a CSP. |
| CryptSetKeyParam | Specify a key's parameters. |
| Data Encryption Functions | |
| CryptEncrypt | Encrypt a section of plaintext using the specified encryption key. |
| CryptDecrypt | Decrypt a section of cipher text using the specified encryption key. |
| Hashing and Digital Signature Functions | |
| CryptCreateHash | Create an "empty" hash object. |
| CryptDestroyHash | Destroy a hash object. |
| CryptGetHashParam | Retrieve a hash object parameter. |
| CryptHashData | Hash a block of data, adding it to the specified hash object. |
| CryptHashSessionKey | Hash a session key, adding it to the specified hash object. |
| CryptSetHashParam | Set a hash object parameter. |
| CryptSignHash | Sign the specified hash object. |
| CryptVerifySignature | Verify a digital signature, given a handle to the hash object that was supposedly signed. |

*ตารางสรุปฟังก์ชัน CryptoAPI 1.0*

สำหรับรายละเอียดการใช้งานฟังก์ชันต่างๆ สามารถศึกษาเพิ่มเติมได้จาก Win32 Programmer's Reference หรือ http://msdn.microsoft.com

# ภาคผนวก ง

## ฟังก์ชันในการอ่านค่ารหัสหมายเลขเครื่อง (Digital Fingerprint API 1.0)

เป็นฟังก์ชันระดับล่างที่ถูกพัฒนาขึ้นเพื่อใช้กับโครงงาน ซึ่งประกอบด้วยฟังก์ชันที่ใช้ในการอ่านค่ารหัสหมายเลขเครื่องจาก 4 แหล่งด้วยกัน คือ

1. MAC Address

   function GetMACAddress(MACAddr: PChar; MaxLen: Integer): Integer; stdcall;

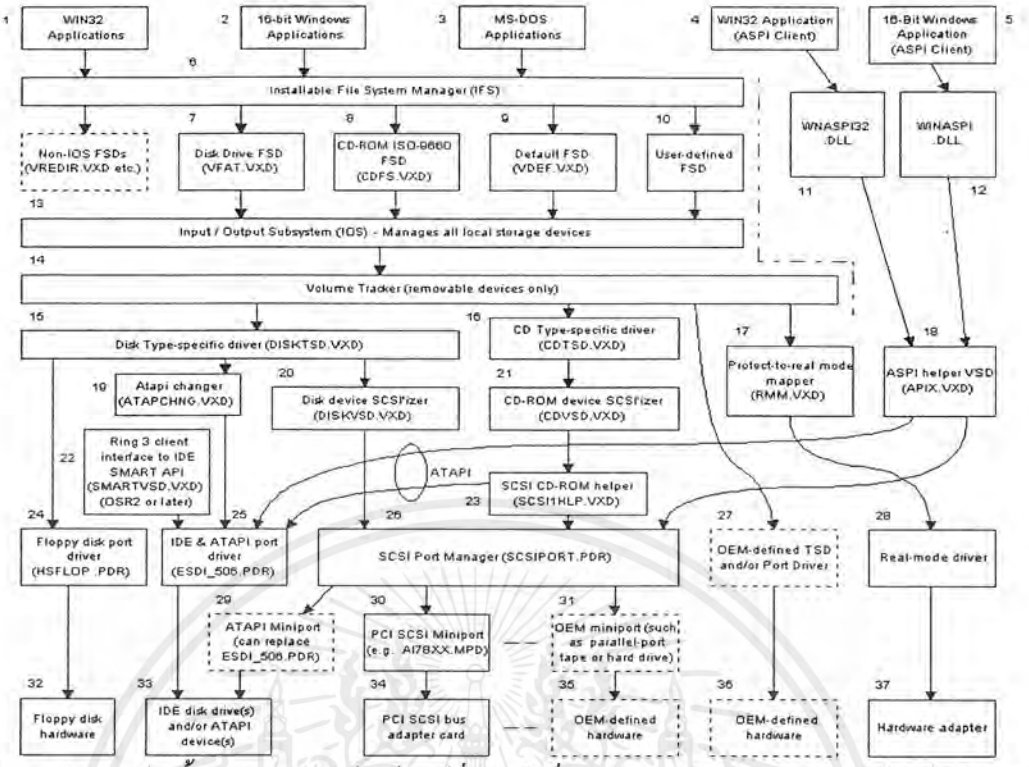
   ฟังก์ชันนี้จะทำการอ่านค่าหมายเลขของการ์ดเน็ตเวิร์กซึ่งเป็นหมายเลขที่ไม่ซ้ำกัน แต่จะมีข้อจำกัดสำหรับเครื่องไม่มีการ์ดเน็ตเวิร์กจะไม่สามารถนำค่านี้มาใช้งานได้

2. Harddisk Identify Information

   function GetHarddiskDFP(DriveID: Byte; HarddiskDFP: PChar; MaxLen: Integer): Integer; stdcall;

   ฟังก์ชันนี้จะทำการอ่านข้อมูลรายละเอียดทางเทคนิคของฮาร์ดดิสก์ แล้วนำเอาเฉพาะส่วนที่เป็น Factory Model, Factory Serial Number และ Factory Revision มาต่อเข้าด้วยกันก็จะได้รหัสหมายเลขที่ไม่ซ้ำกันสำหรับฮาร์ดดิสก์แต่ละตัว

   เนื่องจากการอ่านค่าข้อมูลทางเทคนิคของฮาร์ดดิสก์ ค่อนข้างจะซับซ้อนเนื่องจากระบบปฏิบัติการ (MS Windows) จะมีการป้องกันไม่สามารถติดต่อกับฮาร์ดดิสก์โดยตรง การอ่านค่าดังกล่าวจึงทำโดยการอ่านผ่านดีไวซ์ไดรเวอร์ SMARTVSD.VXD [16] (Self Monitoring and Recovery Technique Virtual Device Driver) ซึ่งเป็นดีไวซ์ไดรเวอร์ที่มีฟังก์ชันรองรับการคำสั่ง Harddisk Identify Command (0xEC) [15]




*รูปแสดงโครงสร้างทั้งหมดของดีไวซ์ไดร์เวอร์ที่ทำหน้าที่ควบคุมการทำงานของอุปกรณ์จัดเก็บข้อมูลในระบบปฏิบัติการวินโดว์*

จากรูปจะเห็นว่า SMARTVSD.VXD (หมายเลข 22) ติดต่อกับ IDE Disk Drive (หมายเลข 33) โดยทำงานผ่าน IDE & ATAPI port driver (หมายเลข 25)

3. Processor Serial Number [17]

function GetProcessorDFP(ProcessorDFP: PChar; MaxLen: Integer): Integer; stdcall;

ฟังก์ชันนี้จะทำการอ่านค่า Processor Serial Number ซึ่งเป็นรหัสขนาด 96 บิต ซึ่งฝังอยู่ใน Intel Pentium III Processor ขึ้นไป โดยการอ่านค่าดังกล่าวสามารถทำได้โดยการใช้คำสั่ง CPUID

อย่างไรก็ตามค่า Processor Serial Number อาจจะถูกป้องกันไม่ให้สามารถอ่านได้โดยเจ้าของเครื่อง ถึงแม้จะเป็นเครื่องที่ใช้ Intel Pentium III Processor ดังนั้น หากต้องการอ่านค่า Processor Serial Number จะต้องได้รับการยินยอมจากผู้ใช้

4. Logical Disk Serial Number

function GetLogicalDiskDFP(DriveID: Char; LogicalDiskDFP: PChar; MaxLen: Integer): Integer; stdcall;

ฟังก์ชันนี้จะอ่านค่า Logical Disk Serial Number โดยผ่าน Win32 API.

# บรรณานุกรม


[1] Hans-Erik Eriksson, Magnus Penker. *UML Toolkit.* USA: John Wiley & Sons, Inc., 1998.

[2] Bruce Powel Douglass. *Real-Time UML Developing Efficient Objects fro Embedded Systems.* USA: Addison-Wesley, 1998.

[3] Xavier Pacheco, Steve Teixeira. *Delphi 4.0 Developer's Guide.* USA: Sams Publishing, 1998.

[4] William Stallings. *Data and Computer Communication (Fifth Edition).* USA: Prentice Hall, 1997.

[5] Borland Delphi 5.0 Developer's Guide On-line Manual.

[6] SentinelLM Developer's Guide On-line Manual.

[7] SentinelLM Programmer's Reference On-line Manual.

[8] CORBA Licensing Service On-line Reference.

[9] SentinelLM.

http://www.rainbow.com/Products/SentinelLM/index.html

[10] SentinelLM Shell.

http://www.rainbow.com/Products/SentinelLMShell/index.html

[11] บทความเกี่ยวกับ Software License Management.

http://www.globetrotter.com/artindx.htm

[12] FlexLM.

http://www.globetrotter.com/flexlm/flexlm.shtml

[13] Public Key Cryptography Standard.

http://www.rsasecurity.com/rsalabs/pkcs

[14] Microsoft CryptoAPI.

http://www.microsoft.com/security/tech/CryptoAPI/default.asp

[15] ATA Specifications.

ftp://fission.dt.wdc.com/pub/standards/ata/

ftp://fission.dt.wdc.com/pub/standards/ata/ata-3/ata3-r6.doc

[16] Microsoft Smart IOCTL API Specifications.

http://premium.microsoft.com/msdn/library/specs/d4/iocltapi.htm

http://www.microsoft.com/hwdev/download/respec/iocltapi.rtf

[17] Processor Serial Number.

http://www.intel.com