# การสอดส่องความปลอดภัยด้วยเว็บแคม

**Surveillance Using Webcam**

**เบ็ญจวรรณ์ ธนรัตนากร**

**สาคเรศ สายวงศ์**



**ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต**

**ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์**

**สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง**

**ปีการศึกษา 2550**

ปริญญานิพนธ์ปีการศึกษา 2550

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การสอดส่องความปลอดภัยด้วยเว็บแคม

SURVEILLANCE USING WEBCAM

ผู้จัดทำ

1. นางสาวเบ็ญจวรรณ์ ธนรัตนากร รหัสนักศึกษา 47010412

2. นางสาวสาคเรศ สายวงศ์ รหัสนักศึกษา 47010827

_____ อาจารย์ที่ปรึกษา

(อาจารย์อัครเดช วัชระภูพงษ์)

_____ อาจารย์ที่ปรึกษา

(ผศ.ธนา หงษ์สุวรรณ)

_____ อาจารย์ที่ปรึกษา

(อาจารย์ ธนัญชัย ตรีภาค)

# การสอดส่องความปลอดภัยด้วยเว็บแคม


| | | |
|---|---|---|
| นางสาวเบ็ญจวรรณ์ | ธนรัตนากร | 47010412 |
| นางสาวสาคเรศ | สายวงศ์ | 47010827 |
| อาจารย์อัครเดช | วัชระภูพงษ์ | อาจารย์ที่ปรึกษา |
| ผศ.ธนา | หงษ์สุวรรณ | อาจารย์ที่ปรึกษาร่วม |
| อาจารย์ธนัญชัย | ตรีภาค | อาจารย์ที่ปรึกษาร่วม |

ปีการศึกษา 2550


## บทคัดย่อ


ถึงแม้ว่าปัจจุบันจะมีการรักษาความปลอดภัยทางกายภาพด้วยกล้องวงจรปิด(CCTV) แต่เนื่องจากระบบกล้องวงจรปิดมีราคาค่อนข้างสูง ประกอบกับปัจจุบันที่มีกล้องขนาดเล็กสำหรับสนทนาผ่านอินเทอร์เน็ตหรือเว็บแคม (Webcam) เป็นที่นิยมใช้กันอย่างมากมายและมีโปรแกรมประยุกต์ใช้ที่หลากหลายและมีประสิทธิภาพสูง ตัวกล้องมีราคาถูกและติดตั้งได้ง่าย จึงเล็งเห็นโอกาสในการประยุกต์ใช้เว็บแคมให้เกิดประโยชน์ในการสอดส่องดูแลรักษาความปลอดภัยทางกายภาพคล้ายกล้องวงจรปิด อีกทั้งรองรับความสามารถในหลากหลายรูปแบบ ซึ่งรองรับกับความต้องการของผู้ใช้ การใช้งานสะดวกและง่ายต่อการจัดการ อีกทั้งยังมีราคาถูกมากเมื่อเทียบกับการใช้ระบบกล้องวงจรปิดที่ปัจจุบันยังคงมีราคาแพงอยู่มาก

อย่างไรก็ตามโปรแกรมดังกล่าวจะมุ่งพัฒนาโปรแกรมบันทึกภาพนิ่งและภาพเคลื่อนไหวและส่ง-รับข้อมูลบนเครื่อข่ายอินทราเนตได้อย่างปลอดภัย โดยให้ความสะดวกสบายและทำงานได้อย่างมีประสิทธิภาพ

**Surveillance Using Webcam**


| | | |
|---|---|---|
| Ms. Benjawan | Tanaratanakorn | 47010412 |
| Ms. Sakares | Saiwongse | 47010827 |
| Mr. Akkradach | Watcharapupong | Advisor |
| Asst.Prof.Thana | Hongsuwan | Co-Advisor |
| Mr. Thananchai | Treepak | Co-Advisor |


Academic Year 2007

## ABSTRACT


Although surveillance cameras, called CCTV, are used but they still cost very high. Nowadays, webcams are popular for video chatting over hi-speed internet because of lower cost, easy setting and programmable – It is possible adapt ordinary webcam for use as surveillance webcam to serve more people.

The objective of this project intends to develop webcam program for video and recording, still image capturing, and secure data transfer over intranet – With easy for use and high performance.

# กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ได้รับคำแนะนำและคำปรึกษาเกี่ยวกับการวิจัยและการค้นคว้าจาก อาจารย์อัครเดช วัชระภูพงษ์ อาจารย์ผู้ควบคุมปริญญานิพนธ์ ผศ.ธนา หงษ์สุวรรณและอาจารย์ ธนัญชัย ตรีภาค ผู้ควบคุมปริญญานิพนธ์ร่วม ผู้วิจัยรู้สึกซาบซึ้งในความอนุเคราะห์จากอาจารย์ทั้งสองท่านเป็นอย่างสูง

ขอบคุณห้องวิจัย ISAG ภาควิชาคอมพิวเตอร์ที่ได้สนับสนุนในส่วนของอุปกรณ์ เครื่องมือ ตลอดจนหนังสือต่างๆที่มีเอื้อประโยชน์แก่การวิจัยในครั้งนี้ด้วย

ขอกราบขอบพระคุณคณาอาจารย์ รวมถึงผู้ช่วยสอนทุกท่านในภาควิชาวิศวคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่านที่ประสิทธิ์ประสาทวิชาความรู้และประสบการณ์ดีๆให้แก่ข้าพเจ้ามาตลอดระยะเวลา 4 ปีที่ทำการศึกษา

ขอขอบคุณเพื่อนๆ พี่ๆและน้องๆ ในภาควิชาวิศวกรรมคอมพิวเตอร์ที่คอยให้กำลังใจและคำแนะนำ รวมถึงประสบการณ์ต่างๆที่ได้ทำร่วมกันตลอดมา

สุดท้ายนี้ข้าพเจ้าขอขอบพระคุณบิดา มารดาและครอบครัวของข้าพเจ้าที่เป็นกำลังใจและเป็นแรงผลักดันให้ข้าพเจ้าสามารถทำปริญญานิพนธ์นี้ด้วย

อย่างไรก็ตามข้าพเจ้าหวังเป็นอย่างยิ่งว่ารายงานของข้าพเจ้าจะเป็นประโยชน์ต่อทุกท่านและเป็นคำแนะนำแก่นักศึกษารุ่นต่อไปในอนาคตข้างหน้า

นางสาวเบ็ญจวรรณ์ ธนรัตนากร

นางสาวสาคเรศ สายวงศ์

# สารบัญ

หน้า


บทคัดย่อภาษาไทย.....I

บทคัดย่อภาษาอังกฤษ.....II

กิตติกรรมประกาศ.....III

สารบัญ.....IV

สารบัญรูป.....VIII

บทที่ 1 บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา.....1

1.2 ความมุ่งหมายและวัตถุประสงค์.....2

1.3 ขอบเขตของการศึกษา.....2

1.4 ขั้นตอนของการศึกษา.....2

1.5 ประโยชน์ที่คาดว่าจะได้รับ.....3

1.6 ส่วนประกอบของปริญญานิพนธ์.....3

บทที่ 2 โครงสร้างของซอฟต์แวร์

2.1 โครงสร้างซอฟต์แวร์.....4

2.2 ส่วนประกอบที่สำคัญที่ได้จากการพัฒนา.....8

2.2.1 โมเดลไคล์แอนต์ (Client) หรือ IsagCam.....8

2.2.2 โมเดลเซิร์ฟเวอร์ (Server) หรือ IsagCam Manager.....9

บทที่ 3 ทฤษฎีที่เกี่ยวข้อง.....11

3.1 การติดต่อกล้องโดยใช้ AVICAP32.....11

3.1.1 ข้อดีของการใช้ AVICAP32.....11

3.1.2 ข้อเสียของการใช้ AVICAP32.....12

3.1.3 การติดต่อกล้องโดยใช้ไลบรารีอื่น.....12

3.1.3.1 *The Microsoft® Windows® Image Acquisition* (WIA).....12

3.1.3.2 IWiaVideo Interface.....13

# สารบัญ(ต่อ)

หน้า


3.2 RTP (Real time Transport Protocol)..... 14

3.2.1 สถาปัตยกรรมโปรโตคอล (Protocol Architecture)..... 14

3.2.2 ความสำคัญของโปรโตคอลอาร์ทีพี..... 15

3.2.3 ข้อดีและข้อเสียของอาร์ทีพี..... 15

3.2.4 ประโยชน์ของอาร์ทีพี..... 15

3.2.5 หลักการทำงานของอาร์ทีพี..... 15

3.2.5.1 ฝั่งส่ง..... 16

3.2.5.2 ฝั่งรับ..... 16

3.2.6 อาร์ทีพีเอพีไอโปรแกรมมิ่ง(RTP API Programming)..... 17

3.2.6.1 RTPSession, RTP Participant..... 17

3.2.6.2 RtpSender, RTPListener..... 19

3.3 ซ็อกเกต (Socket)..... 20

3.3.1 การใช้งานซ็อกเกต (Socket)..... 21

3.3.2 ชนิดของซ็อกเกต (Socket Type)..... 22

3.3.3 การทำงานกับซ็อกเกตในดอทเน็ต..... 22

3.3.4 การทำงานกับซ็อกเกต..... 23

3.3.4.1 แพสซีฟซ็อกเกต..... 23

3.3.4.2 แอคทีฟซ็อกเกต..... 24

3.3.5 ปัญหาด้านการจัดการข้อมูลที่ส่งผ่านระหว่างเครือข่าย..... 25

3.3.5.1 การจำกัดขนาดของแต่ละชุดข้อมูลที่จะส่งให้มีขนาดเท่ากัน..... 25

3.3.5.2 การส่งขนาดข้อมูลไปพร้อมกันกับชุดข้อมูลนั้น..... 26

3.3.5.3 การใส่สัญลักษณ์เพื่อบอกขอบเขตของข้อมูลแต่ละชุด..... 27

3.4 ภาพขาว-ดำ (Grayscale หรือ Grayscale digital image)..... 27

3.4.1 การแปลงภาพสีมาเป็นภาพขาว-ดำ..... 27

3.4.2 หลักการแปลงภาพสีเป็นขาว-ดำ..... 29

3.5 ความเป็นส่วนตัว (Privacy)..... 30

3.5.1 การเข้ารหัสลับ (Cryptography)..... 30

# สารบัญ(ต่อ)

หน้า


3.5.2 อัลกอริทึมในการเข้ารหัสลับ ..... 30

3.5.2.1 อัลกอริทึมแบบสมมาตร (Symmetric key algorithms) ..... 30

3.5.2.2 อัลกอริทึมแบบอสมมาตร (Asymmetric key algorithms) ..... 30

3.5.3 ปัญหาของอัลกอริทึมแบบสมมาตร ..... 31

3.5.4 ความแข็งแกร่งของอัลกอริทึมสำหรับการเข้ารหัสลับ ..... 32

3.5.5 ความยาวของกุญแจที่ใช้ในการเข้ารหัสลับ ..... 33

3.5.6 อัลกอริทึมในการเข้ารหัสลับแบบสมมาตร ..... 33

3.5.6.1 อัลกอริทึมดีอีเอส (DES) ..... 33

3.5.6.2 อัลกอริทึมทริเปิลดีอีเอส (Triple-DES) ..... 34

3.5.6.2.1 การเข้ารหัสลับโดยใช้ดีอีเอส 3 ครั้งด้วยกุญแจ 3 ค่า ..... 35

3.5.6.2.2 การเข้ารหัสลับโดยใช้ดีอีเอส 3 ครั้งด้วยกุญแจ 2 ค่า ..... 36

บทที่ 4 การทดลองและผลการทดลอง ..... 39

4.1 การเขียนไลบรารีติดต่อเว็บแคม ..... 39

4.1.1 สร้างโปรเจควินโดวส์คอนโทรลไลบรารี (Windows Control Library) ..... 39

4.1.2 Platform Invoke (P/Invoke) และ API Constants ..... 40

4.1.3 การเขียนโค้ดเพื่อติดต่อกับเว็บแคม ..... 42

4.1.3.1 เซ็ตค่าแคปเจอร์วินโดวส์ (Setup a capture window) ..... 42

4.1.3.2 ติดต่อไปยังอุปกรณ์ภาพ (Connect to the capture device) ..... 43

4.1.3.3 เลือกฟอร์เมต (Sets the format of captured video data) ..... 43

4.1.3.4 เซ็ตค่าแฟรมคอลแบก (Set Frame callback function) ..... 43

4.1.4 จัดทำฟังก์ชันที่เป็นอินเตอร์เฟสสำหรับผู้ใช้ ..... 45

# สารบัญ(ต่อ)

หน้า


4.1.5 ตรวจสอบการทำงานของไลบรารี.....45

4.2 การเขียนโปรแกรมส่งข้อมูลคำสั่งโดยใช้ สตรีมซ็อกเกต.....45

4.2.1 การเขียนโปรแกรมส่งข้อมูลคำสั่งโดยใช้ทีซีพีที่ฝั่งเซิร์ฟเวอร์..... .46

4.2.2 การเขียนโปรแกรมส่งข้อมูลคำสั่งโดยใช้ทีซีพีที่ฝั่งไคล์แอนต์.....48

4.3 การเขียนโปรแกรมส่งข้อมูลภาพด้วยอาร์ทีพี.....49

4.4 การออกแบบคอนโทรลไคล์แอนต์ด้วยโปรโตคอลทีซีพีให้ทำตามเงื่อนไขที่หลากหลายจากผู้ใช้งาน.....50

4.5 การเขียนโปรแกรมการจัดการกับดิสก์.....51

4.5.1 การเขียนทับไฟล์ (Overwrite) .....51

4.5.2 หยุดการทำงาน (Stop working) ..... 52

4.6 การเขียนโปรแกรมการบันทึกภาพ.....52

4.6.1 สร้างฟอร์เมตของชื่อไฟล์ที่จะทำการบันทึก.....53

4.6.2 เริ่มต้นทำการบันทึกภาพ.....53

4.7 การจัดการเกี่ยวกับปัญหาเรื่องความเป็นส่วนตัวกระบวนการเข้ารหัสถอดรหัสข้อมูลภาพและข้อมูลคำสั่ง.....55

4.7.1 การออกแบบกระบวนการเข้ารหัสและถอดรหัสข้อมูล.....55

4.7.2 ขั้นตอนการเข้ารหัสข้อมูลคำสั่งและภาพ.....55

บทที่ 5 บทวิจารณ์และสรุป.....58

5.1 บทสรุป.....58

5.2 ปัญหาและอุปสรรคและแนวทางการแก้ไข.....59

5.3 แนวทางการพัฒนาต่อ.....60

บรรณานุกรม..... 61

# สารบัญรูป

**รูปที่** **หน้า**

1.1 แสดงภาพรวมโครงสร้างการรักษาความปลอดภัยด้วยเว็บแคม.....1
2.1 ความสามารถในการรองรับเว็บแคมได้มากกว่า 1 ตัว.....6
2.2 โครงสร้างของโครงงานโดยรวมของโครงการรักษาความปลอดภัยด้วยเว็บแคม.....6
2.3 โครงสร้างโดยละเอียดของโครงการรักษาความปลอดภัยด้วยเว็บแคม.....7
2.4 โครงสร้างของโครงงานฝั่งไคล์แอนต์.....8
2.5 โครงสร้างของโครงงานฝั่งเซิร์ฟเวอร์.....9
3.1 แสดงภาพ Video Capture Messages.....11
3.2 แสดงภาพคำสั่งรีจิสเตอร์ DLL.....12
3.3 แสดงภาพเมทธอดของ IWiaVideo.....13
3.4 แสดงภาพสถาปัตยกรรมโปรโตคอล (Protocol Architecture).....14
3.5 แสดงภาพการส่งข้อมูลผ่านโปรโตคอลอาร์ทีพีที่ฝั่งส่ง.....15
3.6 แสดงภาพการรับข้อมูลผ่านโปรโตคอลอาร์ทีพีที่ฝั่งรับ.....16
3.7 แสดงภาพโครงสร้างของอาร์ทีพีในคอนเฟอร์เรนซ์ เอ็กซ์พี 3.0.....17
3.8 ตัวอย่างโค้ด ที่ใช้ในการเพิ่มและลด ผู้ใช้งานในอาร์ทีพีเซสชั่น.....18
3.9 ตัวอย่างโค้ดที่ใช้ในการรับข้อมูลอาร์ทีพีแพ็กเก็ต 1.....18
3.10 ตัวอย่างโค้ดที่ใช้ในการรับข้อมูลอาร์ทีพีแพ็กเก็ต 2.....18
3.11 ตัวอย่างโค้ด ในการเข้าร่วมเซสชั่น.....19
3.12 ตัวอย่างโค้ด หลังจาก เข้าร่วมเซสชั่นแล้วจะสามารถรับค่าอาร์ทีพีสตรีมบัฟเฟอร์จาก FrameReceived.....19
3.13 ตัวอย่างโค้ดสำหรับการใช้งาน RTPSender.....20
3.14 ไดอะแกรมขั้นตอนการสร้างแพสซีฟซ็อกเกต.....24
3.15 ไดอะแกรมขั้นตอนการสร้างแอคทีฟซ็อกเกต.....24
3.16 ส่วนของโปรแกรมการจำกัดขนาดของแต่ละชุดข้อมูลที่จะส่งให้มีขนาดเท่ากัน.....25
3.17 ส่วนของโปรแกรมการส่งขนาดของชุดข้อมูลไปรวมไปกับชุดข้อมูล.....26

# สารบัญรูป(ต่อ)

**รูปที่** **หน้า**


3.18 ค่าของเปอร์เซนต์ที่ใช้ในการแปลงภาพสี....28

3.19 ภาพสีธรรมชาติ....28

3.20 ภาพขาว-ดำ....28

3.21 เมตริกซ์ที่ใช้ในการเปลี่ยนแปลงสี....29

3.22 การเข้ารหัสลับ....35

3.23 การถอดรหัสลับ....35

3.24 การเข้ารหัสลับ....36

3.25 การถอดรหัสลับ....37

4.1 ลักษณะการใช้งานของคอนโทรลไลบรารี....40

4.2 การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี....40

4.3 การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน capCreateCaptureWindowAจากไลบรารี....41

4.4 การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี....41

4.5 การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี....41

4.6 แสดงการประกาศ API Constant....42

4.7 แสดงการติดต่อกับ handle window....42

4.8 แสดงการติดต่อกับแสดงการติดต่อกับแคปเจอร์ไดรเวอร์....43

4.9 แสดงการตั้งต่าวิดีโอฟอร์เมตในการแคปเจอร์....43

4.10 กำหนดฟังก์ชั่น frame callback....44

4.11 การทำฟังก์ชั่น frame callback....44

4.12 ออกแบบของโปรแกรม IsagCam และ IsagCam Manager....45

4.13 โค้ดการใส่ค่า name space ที่เกี่ยวข้องกับทีซีพี....46

4.14 โค้ดการนำเอาไอพีแอดเดรสที่ของฝั่งเซิรฟ์เวอร์ที่ได้มาสร้างเป็นเอนด์พอยท์....46

4.15 โค้ดการสร้างซ็อกเกต....47

4.16 โค้ดการกำหนดเอนด์พอยท์ให้กับ ซ็อกเกตที่สร้างขึ้น....47

4.17 โค้ดการ Listening พอร์ตเพื่อรอรับการเชื่อมต่อ....47

# สารบัญรูป(ต่อ)

**รูปที่** **หน้า**


4.18 โค้ดการ Accept....47

4.19 โค้ดการ Receive ข้อมูลจากไคล์แอนต์....48

4.20 โค้ดการปิดการเชื่อมต่อ....48

4.21 โค้ดการสร้างการเชื่อมต่อไปยัง End Point ที่กำหนด....48

4.22 แสดงอีเวนท์ที่จัดการการเข้าร่วมเซสชั่นเช่นการเพิ่ม/ลดยูสเซอร์จากเซสชั่นของอาร์ทีพี...49

4.23 แสดงแสดงอีเวนท์ที่จัดการ เพิ่ม/ลด เซสชั่นเช่น การแอคติเวทอาร์ทีพีหลายๆเซสชั่น....49

4.24 แสดงUnhook อีเวนท์.... 50

4.25 แสดงฟอร์แมตของคอนโทรล....50

4.26 แสดงโค้ดการเช็คดิสก์....51

4.27 แสดงโค้ดเพื่อดูไฟล์ทั้งหมดในดิสก์....52

4.28 แสดงโค้ดเพื่อทำการ Sort file.... 52

4.29 แสดงโค้ดเพื่อทำการลบไฟล์....52

4.30 แสดงฟอร์แมตของไฟล์ภาพ.... 53

4.31 แสดงคำสั่งที่ใช้ในการบันทึกภาพ....53

4.32 แสดงภาพและชื่อไฟล์ภาพหลังจากที่ถูกบันทึกลงหน่วยความจำแล้ว....54

4.33 การออกแบบโครงสร้างพื้นฐานเกี่ยวกับการเข้ารหัสลับ-ถอดรหัสลับของโปรแกรม.... 55

4.34 การเข้ารหัสคีย์ด้วยอัลกอริทึมแบบเอ็มดีไฟท์ (MD5)....56

4.35 โค้ดการเข้ารหัสข้อมูลด้วยอัลกอริทึม.... .... 56

4.36 โค้ดการเปลี่ยนค่าเข้ารหัสให้กลับมาอยู่ในรูปแบบของสตริงก่อน....57

4.37 โค้ดการเปลี่ยนค่าสตริงให้กลับมาอยู่ในรูปแบบของ byte[].... .....57

4.38 โค้ดการเข้ารหัสข้อมูลภาพ.....……...57

# บทที่ 1
# บทนำ

## 1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากในปัจจุบันได้มีการใช้อุปกรณ์เว็บแคมเพิ่มขึ้นเป็นจำนวนมาก เช่นเดียวกันโครงการการสอดส่องความปลอดภัยด้วยเว็บแคม (Surveillance Using Webcam) จึงแลเห็นว่าควรที่จะประยุกต์เพื่อใช้เว็บแคม ให้สามารถนำมาใช้ในการสอดส่องความปลอดภัยคล้ายระบบกล้องวงจรปิด (CCTV) โดยที่มุ่งมีความสามารถบันทึกภาพนิ่ง ภาพเคลื่อนไหว และส่ง-รับข้อมูล และการทำงานอื่นๆที่มีความสามารถเหมือนกล้องวงจรปิดบนเครือข่ายอินทราเนต (Intranet)ได้อย่างปลอดภัย โดยให้ความสะดวกสบายแก่ผู้ใช้งานและทำงานได้อย่างมีประสิทธิภาพ

โครงการการสอดส่องความปลอดภัยด้วยเว็บแคม (Surveillance Using Webcam) จึงถือว่าเป็นโปรเจคที่พัฒนาขึ้นมาใหม่ เพื่อพัฒนาต้นแบบในการรักษาความปลอดภัยรูปแบบใหม่และลดค่าใช้จ่ายที่จะสูญเสียในการซื้ออุปกรณ์รักษาความปลอดภัยที่ปัจจุบันยังคงมีราคาแพง

**รูปที่ 1.1** แสดงภาพรวมโครงสร้างการรักษาความปลอดภัยด้วยเว็บแคม

## 1.2 ความมุ่งหมายและวัตถุประสงค์

1. เพื่อศึกษาวิธีเขียนโปรแกรมบันทึกภาพนิ่ง และภาพเคลื่อนไหวผ่านเว็บแคม
2. เพื่อสร้างต้นแบบและพัฒนาระบบสอดส่องความปลอดภัยด้วยเว็บแคม
3. พัฒนาโปรแกรมในการใช้งานระบบรักษาความปลอดภัยบนเว็บแคมให้สอดคล้องกับความต้องการในองค์กร
4. เพื่อลดค่าใช้จ่ายในการซื้อระบบรักษาความปลอดภัยในระบบกล้องวงจรปิด (CCTV) ที่มีราคาแพง อีกทั้งยังเป็นการใช้อุปกรณ์ที่มีอยู่ให้เป็นประโยชน์ยิ่งขึ้น

## 1.3 ขอบเขตของการศึกษา

1. IsagCam และ IsagCam Manager ต้องใช้สามารถใช้งานร่วมกัน
2. จำกัดการใช้งานกับอุปกรณ์เว็บแคม 1 ตัวต่อคอมพิวเตอร์ 1 เครื่อง
3. สามารถใช้งานได้บนแพลตฟอร์มวินโดวส์เท่านั้น
4. จำกัดการใช้งานแค่ภายในเครือข่ายอินทราเนต (Intranet) เท่านั้น

## 1.4 ขั้นตอนของการศึกษา

1. ศึกษาเนื้อหารายละเอียดต่างๆที่จำเป็นต้องใช้
    - ศึกษาข้อมูลการติดต่อกับกล้องเว็บแคม และการถ่ายภาพจากเว็บแคม
    - ศึกษาวิธีการเขียนโปรแกรมเพื่อใช้ในการติดต่อกล้อง
    - ศึกษาโครงสร้างและรูปแบบของการเขียนโปรแกรมโดยใช้โปรโตคอลทีซีพี (TCP) ในการส่งข้อมูลคำสั่ง
    - ศึกษาโครงสร้างและรูปแบบของการเขียนโปรแกรมโดยใช้โปรโตคอลอาร์ทีพี (RTP) เพื่อใช้ในการส่งข้อมูลภาพ
    - ศึกษาและหาวิธีการรองรับเว็บแคมได้หลายตัว
    - ศึกษาวิธีการแปลงภาพสีธรรมชาติ เป็นภาพขาว-ดำ (GrayScale)
    - ศึกษาวิธีการตั้งเวลาการถ่ายภาพและเทคนิคอื่นๆที่ใช้ในการคอนโทรลเว็บแคม
    - ศึกษาการจัดการเกี่ยวกับหน่วยความจำ การเข้าถึงข้อมูลของหน่วยความจำ และการลบข้อมูล
    - ศึกษาวิธีการบันทึกภาพ

- ศึกษาการจัดการด้านความปลอดภัยที่เกี่ยวข้องกับโปรเจค
- ศึกษาการจัดการในส่วนของการเพิ่มประสิทธิภาพของโปรแกรม

2. ทดลองโดยการพัฒนาโค้ดที่สามารถนำมาใช้ในการทำโปรเจคได้
   - การทำการศึกษาการเขียนโปรแกรมที่ใช้ในการติดต่อกล้อง
   - การทำการศึกษาโปรแกรมเพื่อใช้ในการคอนโทรลกล้อง
   - การทำการศึกษาวิธีการเขียนโปรแกรมเพื่อให้สามารถติดต่อกล้องได้มากกว่า 1 ตัว
3. ออกแบบในส่วนของหน้าใช้งานของโปรแกรม (User Interface)
4. นำแต่ละฟังก์ชั่นการทำงานมารวมกัน จากนั้นทำการทดสอบโปรแกรมและแก้ไขข้อผิดพลาดของโปรแกรมในส่วนต่างๆ

## 1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. เป็นการสร้างต้นแบบระบบสอดส่องความปลอดภัยด้วยเว็บแคม โดยพัฒนาโปรแกรมบันทึกภาพนิ่ง และภาพเคลื่อนไหวสำหรับแพลตฟอร์มไมโครซอฟท์วินโดวส์ให้สามารถนำมาใช้ประโยชน์ในเรื่องของระบบรักษาความปลอดภัยได้
2. เพิ่มความสะดวกสบายแก่ผู้ใช้งานสามารถเฝ้าดูภาพนิ่ง และภาพเคลื่อนไหวของแว็บแคมที่ใดก็ได้
3. รองรับการใช้งานในรูปแบบที่หลากหลายเพื่อให้ตรงกับความต้องการของผู้งานในองค์กรให้ได้มากที่สุด

## 1.6 ส่วนประกอบของปริญญานิพนธ์

ปริญญานิพนธ์ฉบับนี้ได้แบ่งเนื้อหาออกเป็น 5 บทด้วยกัน ดังต่อไปนี้

บทที่ 1 กล่าวถึงความเป็นมาและความสำคัญของปัญหา ความมุ่งหมายและวัตถุประสงค์ ขอบเขตของการศึกษา ขั้นตอนของการศึกษา ประโยชน์ที่คาดว่าจะได้รับ และส่วนประกอบของปริญญานิพนธ์

บทที่ 2 กล่าวถึงโครงสร้างของซอฟต์แวร์ของโครงการการสอดส่องความปลอดภัยด้วยเว็บแคม (Surveillance Using Webcam) โดยบอกถึงส่วนประกอบที่สำคัญที่ได้พัฒนาขึ้นเอง การทำงานในส่วนหลักและส่วนย่อยต่างๆของโปรแกรมทั้งในฝั่งไคล์แอนต์ (Client) หรือ IsagCam และ ฝั่งเซิร์ฟเวอร์ (Server) หรือ IsagCam Manager

บทที่ 3 แสดงทฤษฎีพื้นฐานที่เกี่ยวข้องกับการทำงานของโปรแกรม โดยประกอบไปด้วย ทฤษฎีที่ใช้

ในการติดต่อกล้อง (AVICAP 32), ทฤษฎีที่ใช้ในการส่งข้อมูล (RTP), ทฤษฎีที่ใช้ในการควบคุมคำสั่ง (Socket), ทฤษฎีที่ใช้ในการทำภาพเคลื่อนไหวธรรมชาติเป็นภาพเคลื่อนไหวขาว-ดำหรือ (Grayscale) และความเป็นส่วนตัวของข้อมูล (Privacy)

บทที่ 4 กล่าวถึงการทดลองและผลการทดลอง เทคนิคที่ใช้ในการเขียนโปรแกรม โดยมีรายละเอียดที่บอกถึงความหมายเหตุผลที่ใช้รวมถึงขั้นตอนการเขียน โดยจะอธิบายถึงขั้นตอนการดำเนินงานของโปรแกรมที่ทดสอบพร้อมทั้งผลการทดลองในแต่ละส่วนเทคนิคและผลการทดสอบโปรแกรม

บทที่ 5 เป็นบทวิจารณ์และสรุปซึ่งกล่าวถึงบทสรุปของโครงงาน ปัญหาและอุปสรรคและแนวทางการแก้ไข แนวทางในการพัฒนาต่อไปในอนาคต

# บทที่ 2
# โครงสร้างของซอฟต์แวร์

## 2.1 โครงสร้างซอฟต์แวร์

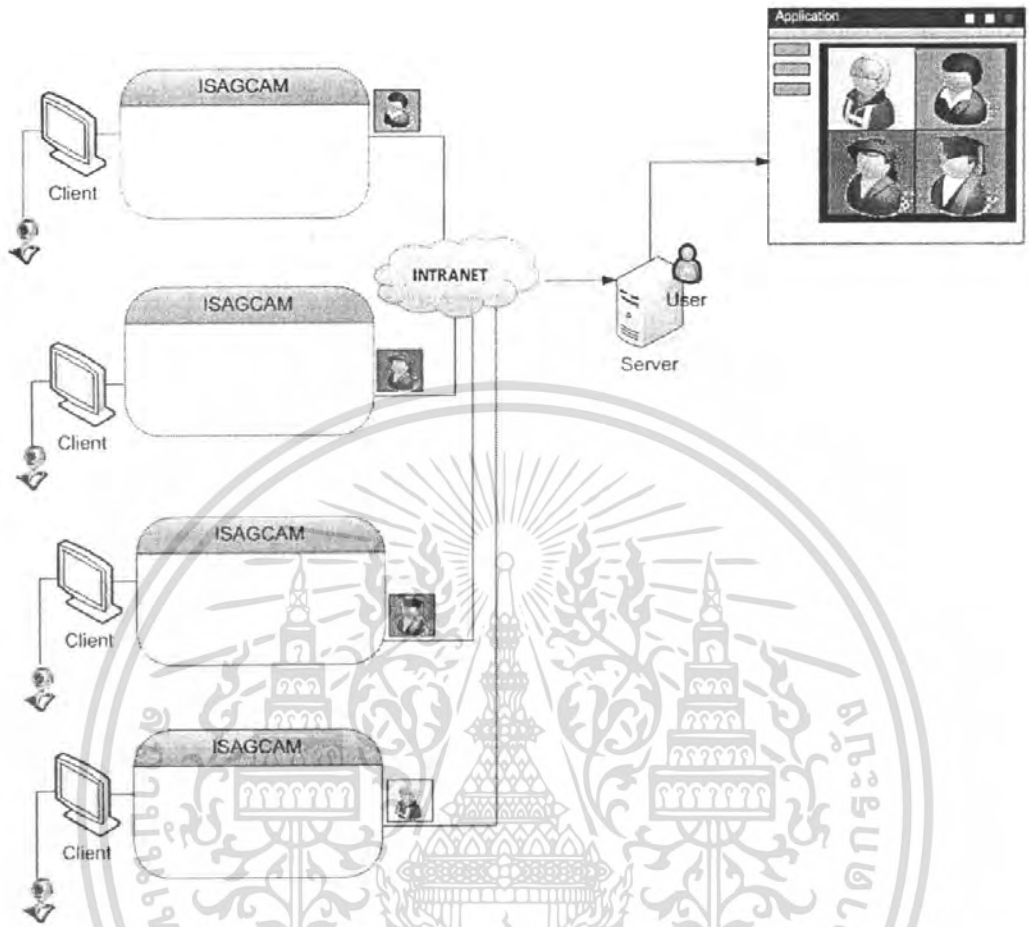
โครงการการสอดส่องความปลอดภัยด้วยเว็บแคม (Surveillance Using Webcam) จะมีโครงสร้างของโปรแกรมที่พัฒนาอยู่ 2 ฝั่งดังต่อไปนี้

ทางฝั่งผู้ใช้ไคล์แอนต์ (Client) จะพัฒนาโปรแกรมชื่อว่า IsagCam คือทำหน้าที่ในการติดต่อกับกล้องเว็บแคมบนเครื่องคอมพิวเตอร์ของผู้ใช้ไคล์แอนต์ (Client) ทำการรับคำสั่งจากฝั่งของเซิร์ฟเวอร์ (Server) และแปลคำสั่งเพื่อใช้ในการควบคุมการทำงานของฝั่งผู้ใช้ไคล์แอนต์ (Client) ในลักษณะต่างๆ เช่นภาพนิ่ง ภาพเคลื่อนไหวขาวดำ ภาพเคลื่อนไหวสีธรรมชาติ การบันทึกตามช่วงเวลาที่กำหนด เป็นต้น และเอาเข้ากระบวนการเข้ารหัส (Encryption) เพื่อการรักษาความปลอดภัย และส่งข้อมูลไปที่ฝั่งเซิร์ฟเวอร์ (Server)

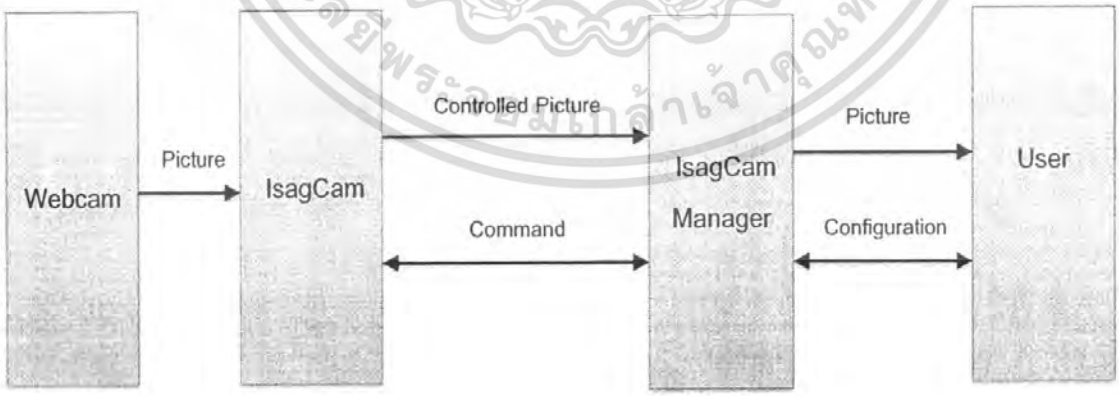
ทางฝั่งผู้ใช้งานเซิร์ฟเวอร์ (Server) จะพัฒนาโปรแกรมชื่อว่า IsagCam Manager คือทำหน้าที่ในการติดต่อกับผู้ใช้ไคล์แอนต์ (Client) ผ่านทางคำสั่งจากหน้าจอของโปรแกรมโดยที่คำสั่งจะเอาเข้ากระบวนการเข้ารหัส (Encryption) เพื่อการรักษาความปลอดภัย นอกจากนี้จะนำข้อมูลภาพไปแสดงผลให้แก่ผู้ใช้งานเซิร์ฟเวอร์ (Server) และบันทึกภาพที่ได้ รวมถึงการจัดการในส่วนของดิสก์

โดยโครงการการสอดส่องความปลอดภัยด้วยเว็บแคม (Surveillance Using Webcam) จะต้องสามารถรองรับเว็บแคมได้มากกว่า 1 ตัว ดังรูปที่ 2.1

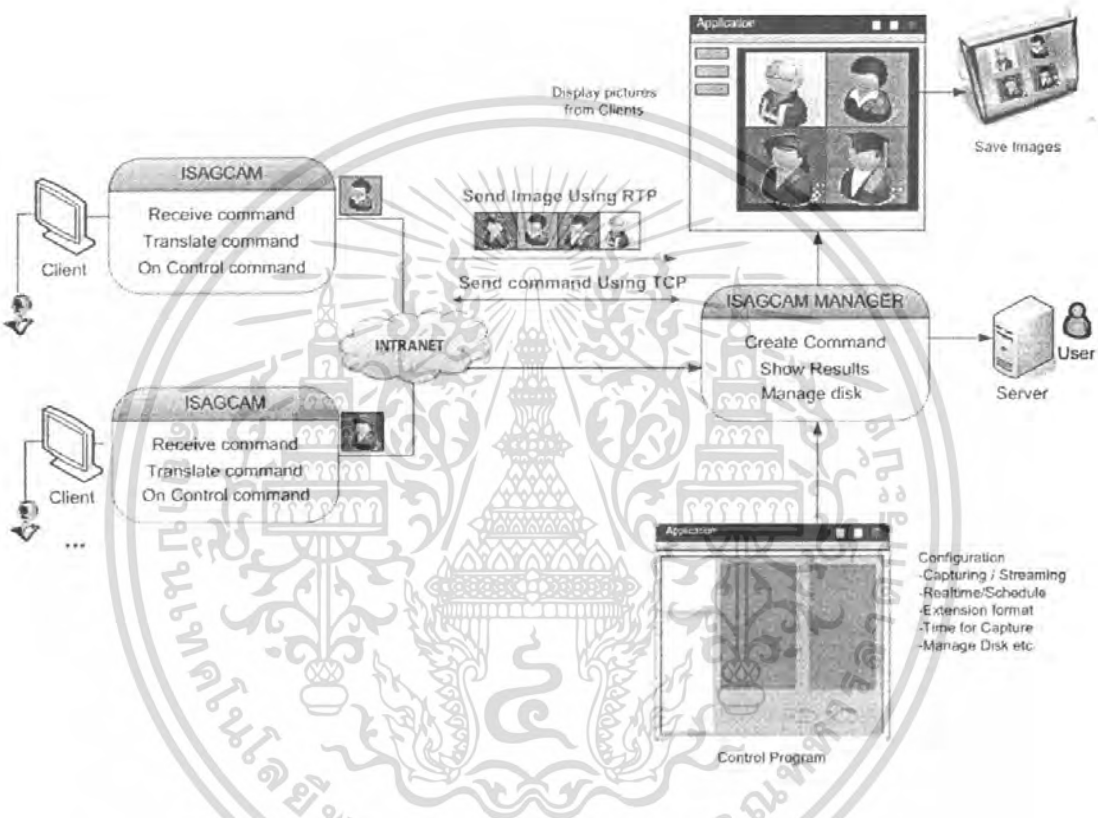
ภาพด้านล่างจะแสดงโครงสร้างโดยรวมดังรูปที่ 2.2 และโครงสร้างโดยละเอียดดังรูปที่ 2.3




**รูปที่ 2.1** ความสามารถในการรองรับเว็บแคมได้มากกว่า 1 ตัว




**รูปที่ 2.2** โครงสร้างของโครงงานโดยรวมของโครงการรักษาความปลอดภัยด้วยเว็บแคม

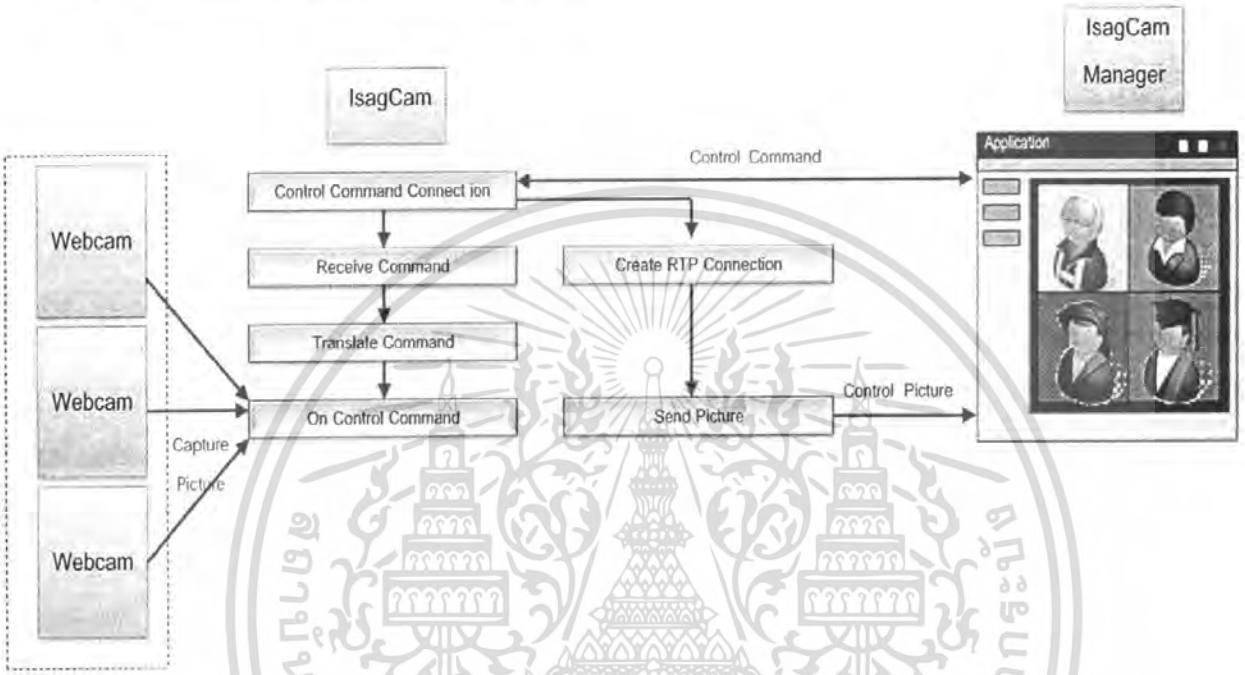



**รูปที่ 2.3** โครงสร้างโดยละเอียดของโครงการรักษาความปลอดภัยด้วยเว็บแคม

## 2.2 ส่วนประกอบที่สำคัญที่ได้จากการพัฒนา

ส่วนประกอบที่ได้พัฒนาในโปรแกรมแบ่งออกเป็น 2 ส่วน ดังต่อไปนี้

### 2.2.1 โมเดลไคล์แอนต์ (Client) หรือ IsagCam



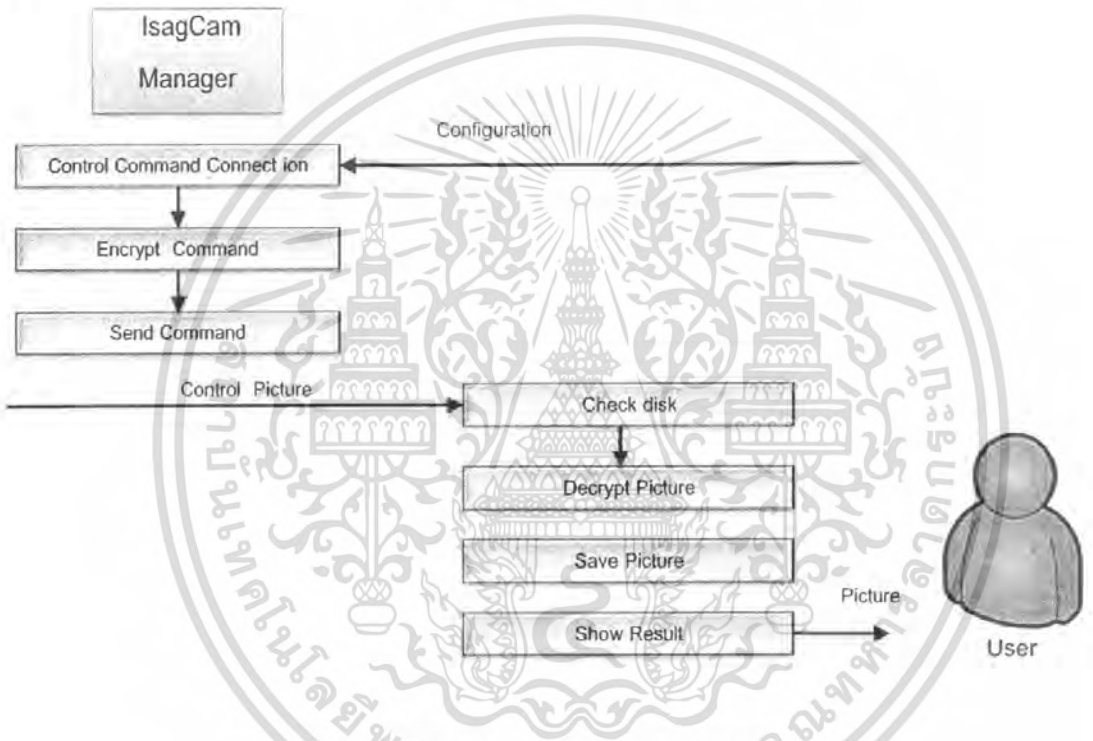

**รูปที่ 2.4** โครงสร้างของโครงงานฝั่งไคล์แอนต์

การทำงานในฝั่งของไคล์แอนต์ (Client) หรือ IsagCam จะมีการทำงานดังต่อไปนี้

- การทำงานของฝั่งไคล์แอนต์ (Client) หรือ IsagCam จะเป็นในลักษณะของสตาทอัพ
- การคอนโทรลส่วนของคำสั่ง ในส่วนนี้จะเป็นส่วนควบคุมคำสั่งต่างๆของโปรแกรม
- การสร้างอาร์ทีพี (RTP) คอนเนคชั่นเพื่อทำการใช้ในการส่งภาพจากเว็บแคมไปยังฝั่งเซิร์ฟเวอร์ (Server)
- การรับข้อมูลคำสั่งต่างๆ จากส่วนควบคุม
- การแปลเป็นคำสั่งต่างๆเช่น
    - คำสั่งในการสร้างคอนเนคชั่น
    - คำสั่งในการจัดการกับรูปภาพในลักษณะต่างๆ เช่น
        - ภาพเคลื่อนไหวขาวดำ สีธรรมชาติ
        - ภาพแคปเจอร์ขาวดำ สีธรรมชาติ
        - การตั้งเวลาในการถ่ายภาพ

- การจัดการตามคำสั่งในส่วนนี้จะรับข้อมูลภาพที่ได้จากเว็บแคมไปทำการประมวลผลตามคำสั่งควบคุมจากฝั่งเซิร์ฟเวอร์ (Server)
- กระบวนการจัดการกับภาพเพื่อทำให้เกิดความปลอดภัย
- การส่งภาพ จะทำการส่งข้อมูลภาพไปยังเซิร์ฟเวอร์ (Server) ด้วยโปรโตคอลอาร์ทีพี (RTP)

### 2.2.2 โมเดลเซิร์ฟเวอร์ (Server) หรือ IsagCam Manager




**รูปที่ 2.5** โครงสร้างของโครงงานฝั่งเซิร์ฟเวอร์

การทำงานในฝั่งของเซิร์ฟเวอร์ (Server) หรือ IsagCam Manager จะมีการทำงานดังต่อไปนี้

- การคอนโทรลส่วนของคำสั่ง ในส่วนนี้จะเป็นส่วนควบคุมคำสั่งการควบคุมต่างๆ อันได้แก่ ค่าที่ใช้ในการคอนฟิคกูเรชั่น
  - การแคปเจอร์/สตรีมมิ่ง
  - ชนิดของไฟล์ภาพที่จะทำการบันทึก
  - พาร์ทในการบันทึกภาพที่ได้รับ
  - เวลาในการแคปเจอร์
  - ภาพสีธรรมชาติ ภาพขาวดำ

- การบันทึกเวลาจริง/การตั้งเวลาในการบันทึก
- ดิสก์และการจัดการกับดิสก์เมื่อเต็ม

ค่าที่ใช้ในการควบคุมอื่นๆ เช่น

- ส่งคำสั่ง
- หยุดการทำงาน

- การนำคำสั่งที่ได้รับมาจากค่าคอนฟิคกูเรชั่น (Configuration) ต่างๆมาสร้างเป็นรูปแบบของการคอนโทรล
- กระบวนการจัดการกับคำสั่งเพื่อทำให้เกิดความปลอดภัย
- การส่งข้อมูลคำสั่งผ่านทางคอนโทรลคอนเนคชั่น
- การจัดการกับดิสก์
- การแสดงข้อมูลของคำสั่งของแต่ละกล้อง
- การบันทึกผลของข้อมูลภาพ

# บทที่ 3
# ทฤษฎีที่เกี่ยวข้อง

## 3.1 การติดต่อกล้องโดยใช้ AVICAP32

Audio Video Interleave หรือ AVI ซึ่งเป็นรูปแบบหนึ่งที่ใช้ในการบันทึกภาพมัลติมิเดียถูกคิดค้นขึ้นโดยบริษัทไมโครซอฟต์ในปี 1992 โดยเป็นส่วนหนึ่งของเทคโนโลยีที่ใช้สำหรับวิดิโอในระบบปฏิบัติการวินโดวส์

การใช้คลาสเอวีไอแคปวินโดว์ (AVICap Windows Class) สามารถทำได้บนระบบปฏิบัติการวินโดวส์ โดยเราสามารถจับภาพวิดิโอได้อย่างง่ายดาย คลาส AVICap จะอยู่ในไฟล์ avicap32.dll ซึ่งภายในประกอบด้วยส่วนต่อประสานโดยใช้แมสเสจ (message-based interfaces) ที่ใช้ในการเข้าถึงอุปกรณ์สำหรับบันทึกวิดิโอและออดิโอและมีความสามารถในการบันทึกวิดิโอสตรีมลงบนดิสก์

### 3.1.1 ข้อดีของการใช้ AVICAP32

สามารถจัดการกับคุณสมบัติที่หลากหลายในการบันทึกภาพ ดังภาพด้านล่างแสดงความสามารถในการทำงานของ AVICAP32

| | |
|---|---|
| WM_CAP_ABORT | WM_CAP_PAL_MANUALCREATE |
| WM_CAP_DLG_VIDEOCOMPRESSION | WM_CAP_PAL_OPEN |
| WM_CAP_DLG_VIDEODISPLAY | WM_CAP_PAL_PASTE |
| WM_CAP_DLG_VIDEOFORMAT | WM_CAP_PAL_SAVE |
| WM_CAP_DLG_VIDEOSOURCE | WM_CAP_SEQUENCE |
| WM_CAP_DRIVER_CONNECT | WM_CAP_SEQUENCE_NOFILE |
| WM_CAP_DRIVER_DISCONNECT | WM_CAP_SET_AUDIOFORMAT |
| WM_CAP_DRIVER_GET_CAPS | WM_CAP_SET_CALLBACK_CAPCONTROL |
| WM_CAP_DRIVER_GET_NAME | WM_CAP_SET_CALLBACK_ERROR |
| WM_CAP_DRIVER_GET_VERSION | WM_CAP_SET_CALLBACK_FRAME |
| WM_CAP_EDIT_COPY | WM_CAP_SET_CALLBACK_STATUS |
| WM_CAP_FILE_ALLOCATE | WM_CAP_SET_CALLBACK_VIDEOSTREAM |
| WM_CAP_FILE_GET_CAPTURE_FILE | WM_CAP_SET_CALLBACK_WAVESTREAM |
| WM_CAP_FILE_SAVEAS | WM_CAP_SET_CALLBACK_YIELD |
| WM_CAP_FILE_SAVEDIB | WM_CAP_SET_MCI_DEVICE |
| WM_CAP_FILE_SET_CAPTURE_FILE | WM_CAP_SET_OVERLAY |
| WM_CAP_FILE_SET_INFOCHUNK | WM_CAP_SET_PREVIEW |
| WM_CAP_GET_AUDIOFORMAT | WM_CAP_SET_PREVIEWRATE |
| WM_CAP_GET_MCI_DEVICE | WM_CAP_SET_SCALE |
| WM_CAP_GET_SEQUENCE_SETUP | WM_CAP_SET_SCROLL |
| WM_CAP_GET_STATUS | WM_CAP_SET_SEQUENCE_SETUP |
| WM_CAP_GET_USER_DATA | WM_CAP_SET_USER_DATA |
| WM_CAP_GET_VIDEOFORMAT | WM_CAP_SET_VIDEOFORMAT |
| WM_CAP_GRAB_FRAME | WM_CAP_SINGLE_FRAME |
| WM_CAP_GRAB_FRAME_NOSTOP | WM_CAP_SINGLE_FRAME_CLOSE |
| WM_CAP_PAL_AUTOCREATE | WM_CAP_SINGLE_FRAME_OPEN |
| WM_CAP_PAL_MANUALCREATE | WM_CAP_STOP |

**รูปที่ 3.1** แสดงภาพ Video Capture Messages

### 3.1.2 ข้อเสียของการใช้ AVICAP32

AVICAP32 เป็นส่วนเชื่อมต่อในการเขียนแอพพลิเคชั่น (Application Programming Interface หรือ API) ที่ไม่ได้ถูกขยายเป็นคลาสที่ถูกดูแลจัดการโดยดอทเน็ต (managed class) ดังนั้นเป็นหน้าที่ของนักพัฒนาโปรแกรมที่จำเป็นต้องใช้ Platform Invoke (P/Invoke) เพื่อทำการเรียกใช้ API เหล่านี้

PInvoke เป็นกลไกของภาษาดอทเน็ต (.NET) ที่ใช้ในการเรียกฟังก์ชันที่ไม่ได้ถูกดูแลจัดการโดยดอทเน็ต (unmanaged functions) ใน DLL โดยเฉพาะอย่างยิ่งใช้ประโยชน์สำหรับการเรียกฟังก์ชันของวินโดว์เอพีไอ (Windows API functions) ที่ไม่ได้ถูกเอนแคพซูเลทโดยคลาสของดอทเน็ตเฟรมเวิร์ก (.NET Framework) เช่นเดียวกับฟังก์ชันอื่นๆที่พัฒนาขึ้นโดยนักพัฒนาอิสระ (third-party functions) อื่นๆที่อยู่ใน DLL

### 3.1.3 การติดต่อกล้องโดยใช้ไลบรารีอื่น

นอกจากคลาส AVICap แล้วยังมีไลบรารีอื่นๆที่ใช้ในการติดต่อกับเว็บแคมได้ ไลบรารีที่ได้ศึกษาเพิ่มเติมมีดังนี้

#### 3.1.3.1 The Microsoft® Windows® Image Acquisition (WIA) Automation Layer version 2.0

เป็นคอมโพเนนต์ในการจัดการกับภาพจากกล้องดิจิตอล, สแกนเนอร์, เว็บแคม โดยเราสามารถดาวน์โหลดไฟล์ DLL นี้ คือ wiaaut.dll ได้จากทางเว็บไซต์ของไมโครซอฟต์ซึ่งเมื่อดาวน์โหลดมาแล้วจะได้ไฟล์ Microsoft Compiled HTML Help ด้วย

จากนั้นจึงทำการรีจิสเตอร์ DLL ดังกล่าวโดยใช้คำสั่งข้างใต้นี้ที่คอมมานพรอมต์ (Command Prompt)

```
regsvr32 <path & filename of dll or ocx>
```

**รูปที่ 3.2** แสดงภาพคำสั่งรีจิสเตอร์ DLL

ถ้าใช้เพียงไลบรารี ดังกล่าว สิ่งที่รีเทิร์นกลับมาได้จะเป็นไฟล์บนหน่วยความจำในคอมพิวเตอร์ (storage) แต่เราไม่ต้องการ เราต้องการเพียงแค่ภาพ ซึ่งจะส่งต่อไปยังเครื่องของเซิร์ฟเวอร์และบันทึกที่เซิร์ฟเวอร์ อีกทั้งคำสั่งที่ใช้ในการกำหนดคุณสมบัติของรูปก็มีไม่มาก

### 3.1.3.2 IWiaVideo Interface

เป็นอินเตอร์เฟสที่ได้จัดหาวิธีในการใช้เซอร์วิส Microsoft Windows Image Acquisition (WIA) เพื่อรับภาพจากอุปกรณ์ใดๆก็ตามที่สามารถส่งภาพสตรีมวิดิโอได้ แต่มีข้อจำกัดคือสิ่งที่รีเทิร์นได้จะเป็นไฟล์บนหน่วยความจำในคอมพิวเตอร์จากภาพด้านล่างเป็นตัวอย่างของเมทธอดของ IWiaVideo

| | |
|---|---|
| CreateVideoByDevNum | The IWiaVideo::CreateVideoByDevNum method creates a connection to a streaming video device with the device number obtained from a Directshow enumeration. |
| CreateVideoByName | The IWiaVideo::CreateVideoByName method creates a connection to a streaming video device with the friendly device name obtained from a Directshow enumeration. |
| CreateVideoByWiaDevID | The IWiaVideo::CreateVideoByWiaDevID method creates a connection to a streaming video device from its WIA_DIP_DEV_ID property. |
| DestroyVideo | The IWiaVideo::DestroyVideo method shuts down the streaming video. To restart video playback, the application must call one of the IWiaVideo CreateVideo methods again. |
| GetCurrentState | The IWiaVideo::GetCurrentState method specifies the state of the video stream as a member of the WIAVIDEO_STATE enumeration. |
| ImagesDirectory | The IWiaVideo::ImagesDirectory property specifies the full path and directory where images are stored when calling the IWiaVideo::TakePicture method. |
| Pause | The IWiaVideo::Pause method pauses video playback. |
| Play | Begins playback of streaming video. |
| PreviewVisible | The IWiaVideo::PreviewVisible property specifies whether the video playback is visible in its parent window. This does not affect the WIAVIDEO_STATE of the video. |
| ResizeVideo | The IWiaVideo::ResizeVideo method resizes the video playback to the largest supported resolution that fits inside the parent window. Call this method whenever the parent window is moved or resized. |
| TakePicture | The IWiaVideo::TakePicture method extracts a still image from the video stream, and saves the image as a JPEG file. |

**รูปที่ 3.3** แสดงภาพเมทธอดของ IWiaVideo

จะเห็นได้ว่า 2 วิธีหลังในการติดต่อกับเว็บแคมมีข้อจำกัดในการแคปเจอร์ (capture) ภาพ นั่นคือภาพจะต้องถูกบันทึกลงไฟล์ ดังนั้นในโครงงานนี้จึงเลือกใช้ไลบรารีแรก คือ AVICAP32 ในการพัฒนาโปรแกรม

## 3.2 RTP (Real time Transport Protocol)

### 3.2.1 สถาปัตยกรรมโปรโตคอล (Protocol Architecture)

เมื่อกล่าวถึงการสื่อสารและเครือข่ายสิ่งที่ต้องกล่าวถึงด้วยคือโปรโตคอลและสถาปัตยกรรมโปรโตคอล ทำให้เรามีความเข้าใจว่า โปรโตคอลคือข้อกำหนดการสื่อสารระหว่างคู่สื่อสารซึ่งอาจเป็นระบบคอมพิวเตอร์ อุปกรณ์เครือข่ายหรืออาจเจาะจงไปถึงองค์ประกอบย่อยในระบบ เช่น โปรแกรมประยุกต์ที่ทำงานอยู่บนคอมพิวเตอร์นั้น การสื่อสารจะเกิดขึ้นได้เมื่อทั้งสองระบบต้อง "ใช้ภาษาสื่อสารเดียวกัน" รวมทั้งสิ่งที่จะสื่อสาร วิธีการสื่อสารและจังหวะการสื่อสารจะต้องเป็นไปตามแบบแผนที่โปรโตคอลกำหนด

ทั้งฮาร์ดแวร์และซอฟต์แวร์ในเครือข่ายย่อมมีกลไกและขั้นตอนที่ซับซ้อน การออกแบบโปรโตคอลต้องการแบบอ้างอิงที่แบ่งการทำงานออกเป็นส่วนย่อยและสามารถจัดสรรหน้าที่งานเฉพาะอย่างให้แต่ละส่วนย่อยได้ การแยกย่อยจะช่วยลดความซับซ้อนและยังช่วยให้เข้าใจได้ง่ายกว่าการมองภาพโดยรวมทั้งหมดเป็นสิ่งเดียว โปรโตคอลการสื่อสารจึงมีโครงสร้างเป็นส่วนๆ แต่ละส่วนมีลักษณะเป็นโมดูล (Module) ที่กำหนดแบบแผนการทำงานเฉพาะเรื่อง เราจึงเรียกวิธีการออกแบบและจัดวางโครงสร้างนี้ว่า **สถาปัตยกรรมโปรโตคอล (Protocol Architecture)**

| | | |
|---|---|---|
| 1 | **Application** | โปรเซสและแอพพลิเคชั่นในเครือข่าย |
| 2 | **Presentation** | การแทนข้อมูล รหัสข้อมูล |
| 3 | **Session** | การสื่อสารระหว่างโปรเซส |
| 4 | **Transport** | การเชื่อมต่อระหว่างต้นทางและปลายทาง |
| 5 | **Network** | การเลือกเส้นทาง |
| 6 | **Data Link** | การเข้าใช้สายสัญญาณ |
| 7 | **Physical** | การเชื่อมต่อทางกายภาพ |

**รูปที่ 3.4** แสดงภาพสถาปัตยกรรมโปรโตคอล (Protocol Architecture)

### 3.2.2 ความสำคัญของโปรโตคอลอาร์ทีพี

โปรโตคอลอาร์ทีพีเป็นโปรโตคอลที่มีความสามารถในการรองรับส่งข้อมูลจำพวก ข้อมูล, เสียง/วิดิโอ ผ่านทางเครือข่ายเน็ตเวิร์ค ซึ่งมีจุดมุ่งหมายในการให้บริการเซอร์วิสที่เป็นประโยชน์สำหรับการขนส่งข้อมูลในชั้นแอพพลิเคชั่นเลเยอร์ในแบบตามเวลาจริง (real-time)

### 3.2.3 ข้อดีและข้อเสียของอาร์ทีพี

**ข้อดี**

- ส่งข้อมูลพวกที่ต้องใช้เวลาจริง เช่น วิดิโอ
- สามารถใช้ร่วมกับโปรแกรมประยุกต์โครงข่ายมัลติมีเดียอื่นๆได้

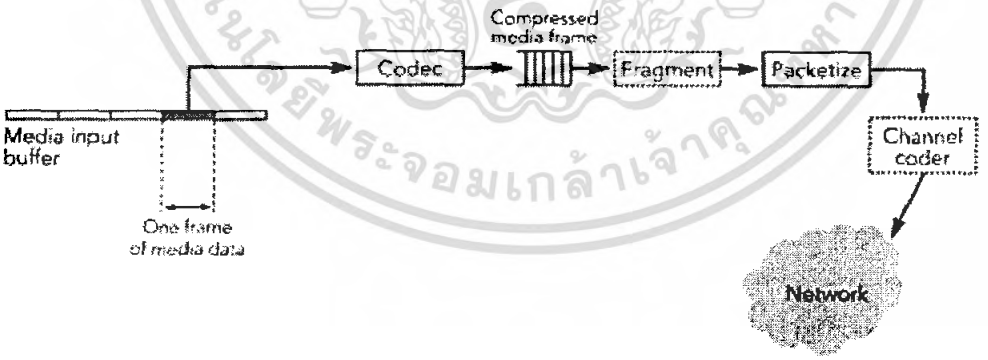
**ข้อเสีย**

- ไม่มีการเรียงลำดับข้อมูล
- มีปัญหาในการลำดับก่อนหลังของเฟรม
- อาร์ทีพีไม่มีการรับประกันของข้อมูลที่ส่ง
- อาร์ทีพีไม่มีกลไกใดๆในการยืนยันข้อมูลว่าส่งได้สำเร็จหรือไม่

### 3.2.4 ประโยชน์ของอาร์ทีพี

- ส่งข้อมูลที่เป็นพวกข้อมูลที่ใช้เวลาจริง
- ควบคุมผลข้อมูลในทันทีที่ข้อมูลถูกส่งเข้าเพื่อให้ได้ผลลัพธ์ที่ต้องการออกมาได้ทันที
- สามารถใช้ร่วมกับโปรแกรมประยุกต์โครงข่ายมัลติมีเดียอื่นๆได้

### 3.2.5 หลักการทำงานของอาร์ทีพี



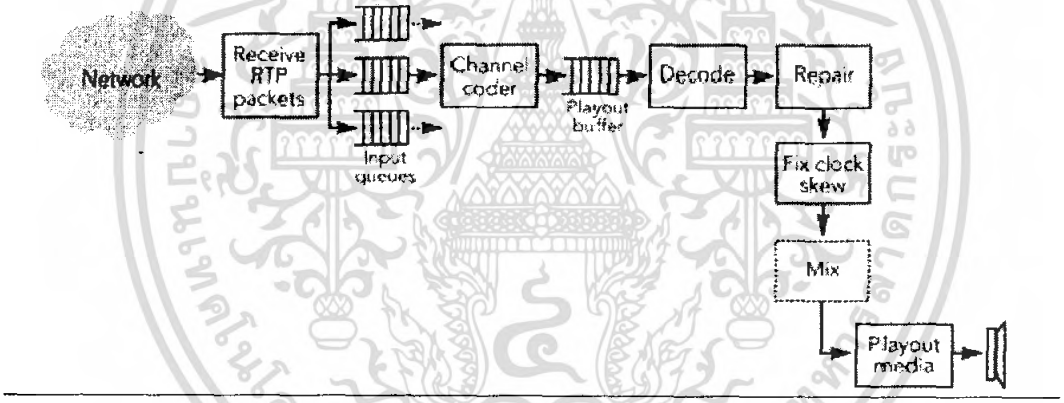

**รูปที่ 3.5** แสดงภาพการส่งข้อมูลผ่านโปรโตคอลอาร์ทีพีที่ฝั่งส่ง

**3.2.5.1 ฝั่งส่ง**

หน้าที่แรกคือการจัดการกับเฟรมข้อมูลให้อยู่ในรูปของแพ็กเก็ตอาร์ทีพีเพื่อให้พร้อมกับการส่งข้อมูล ถ้าเฟรมมีขนาดใหญ่ให้ทำการแฟรคเมนต์ (fragment) ให้เป็นแพ็กเก็ตอาร์ทีพีย่อยๆและในทางตรงกันข้ามถ้าข้อมูลเฟรมมีขนาดเล็กให้ทำการรวมเฟรมข้อมูลต่างๆเข้าด้วยกันเป็นแพ็กเก็ตอาร์ทีพีแพ็กเก็ตเดียว (reassemble)

นอกจากนี้ยังมีการตรวจสอบความผิดพลาด (error correction scheme) เพื่อตรวจสอบความถูกต้องของข้อมูลและส่วนของแชนเนลโค้ดเดอร์ (channel coder) จะทำหน้าที่ในการจัดแพ็กเก็ตขึ้นใหม่ก่อนการส่ง ฝ(error correction packets)

ต่อมาหลังจากที่ข้อมูลแพ็กเก็ตอาร์ทีพีถูกส่งไปแล้ว ข้อมูลในบัฟเฟอร์ก็จะว่างในที่สุด ฝั่งส่งจะไม่ทำการโยนทิ้งแพ็กเก็ตหากตรวจพบว่ามีความผิดพลาดของข้อมูลหรือความผิดพลาดที่เกิดจากการเข้ารหัส เกิดขึ้น นั่นหมายความว่า ฝั่งส่งจะบัฟเฟอร์ บางส่วนหลังจากที่แพ็กเก็ตได้ถูกส่งออกไปขึ้นอยู่กับโคเดค (codec) และการตรวจสอบความผิดพลาดที่ถูกใช้ไป



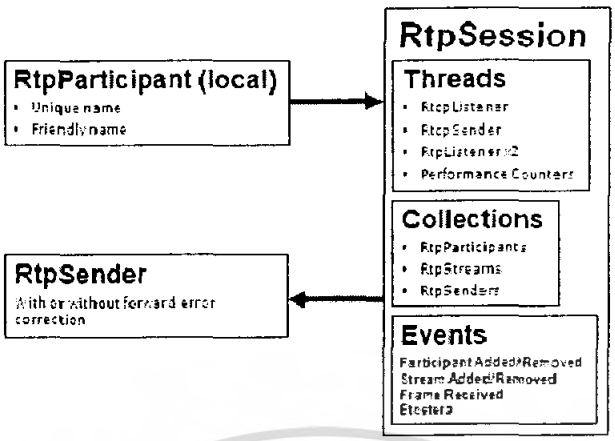

**รูปที่ 3.6** แสดงภาพการรับข้อมูลผ่านโปรโตคอลอาร์ทีพีที่ฝั่งรับ

**3.2.5.2 ฝั่งรับ**

มีหน้าที่ในการรับแพ็กเก็ตที่มาจากเครือข่ายเน็ตเวิร์ก นอกจากนี้ยังทำงานในส่วนของการตรวจสอบบางแพ็กเก็ตที่สูญหาย การกู้เวลาใหม่และการถอดข้อมูลมีเดีย พร้อมทั้งแสดงผลลัพธ์ให้กับผู้ใช้งานได้ทราบอนุญาตให้ผู้ส่งสามารถดัดแปลงการส่งมาถึงผู้รับและทำการดูแลรักษาส่วนที่เป็นฐานข้อมูล (database) ของผู้ใช้งานในเซสชั่น (session) นั้นๆ

**3.2.6 อาร์ทีพีเอพีไอโปรแกรมมิ่ง (RTP API Programming)**

สำหรับ ไมโครซอฟต์จะมีการใช้อาร์ทีพีในรูปแบบของคอนเฟอร์เรนซ์ เอ็กซ์พี (Conference XP) โดยโครงสร้างของอาร์ทีพีในคอนเฟอร์เรนซ์เอ็กซ์พี 3.0 (Conference XP 3.0) จะเป็นดังนี้




รูปที่ 3.7 แสดงภาพโครงสร้างของอาร์ทีพี ในคอนเฟอร์เรนซ์ เอ็กซ์พี 3.0

**3.2.6.1 RTPSession, RTP Participant**

เซสชั่นจะประกอบไปด้วย กลุ่มของผู้ใช้งาน (participant) ซึ่งต้องการติดต่อสื่อสาร โดยใช้อาร์ทีพี ซึ่งผู้ใช้งานดังกล่าวสามารถติดต่อได้กับอาร์ทีพีหลายๆเซสชั่น ยกตัวอย่างเช่น เซสชั่นนึงใช้สำหรับการแลกเปลี่ยนข้อมูลเสียง อีกเซสชั่นนึงอาจใช้สำหรับแลกเปลี่ยนข้อมูลวิดีโอ ในแต่ละผู้ใช้งานเซสชั่นจะถูกระบุโดย

- หมายเลขเน็ตเวิร์กแอดเดรส
- หมายเลขพอร์ต 1 คู่ ซึ่ง หมายเลขพอร์ตดังกล่าวจะใช้ในการไว้ให้ข้อมูลสำหรับในการส่ง-รับหรือทั้งส่งทั้งรับก็สามารถทำได้

โดยทั่วไปหมายเลขพอร์ต มักจะถูกกำหนดเป็นหมายเลขที่ติดกัน โดยหมายเลขที่เป็นเลขคู่ จะถูกกำหนดให้สำหรับการขนส่งข้อมูลด้วยโปรโตคอลอาร์ทีพี ส่วนหมายเลขพอร์ตที่สูงกว่าที่เป็นเลขคี่ มักจะใช้สำหรับอาร์ทีซีพี (RTCP) เพื่อใช้สำหรับคอนโทรลแพ็กเก็ต

สำหรับหมายเลขดีฟอลต์พอร์ตโดยทั่วไปจะให้ใช้ 5004,5005 ซึ่งใช้สำหรับยูดีพี/ไอพี (UDP/IP) แต่ในหลาย แอพพลิเคชั่นจะสร้างเซสชั่นและเพิกเฉยหมายเลขดีฟอลต์พอร์ต

เซสชั่นจะถูกออกแบบสำหรับ ชั้นทรานสปอตเลเยอร์เพื่อส่งมิเดีย สำหรับในกรณีที่เป็นมัลติมิเดีย แต่ละมิเดียจะถูกแยกออกด้วยอาร์ทีพี เซสชั่นหนึ่งๆ

สำหรับคลาส RtpSession และ RtpParticipant จะทำหน้าที่

-จัดการกับออปเจกต์และข้อมูลของอาร์ทีพี

-รอข้อมูลต่างๆจากผู้ใช้

-รับ-ส่งข้อมูลอาร์ทีพี

```
RtpEvents.RtpParticipantAdded += new
RtpEvents.RtpParticipantAddedEventHandler(RtpParticipantAdded);
RtpEvents.RtpParticipantRemoved += new
RtpEvents.RtpParticipantRemovedEventHandler(RtpParticipantRemoved);
```

**รูปที่ 3.8** ตัวอย่างโค้ด ที่ใช้ในการเพิ่มและลด ผู้ใช้งานในอาร์ทีพี เซสชั่น

```
RtpEvents.RtpStreamAdded += new
RtpEvents.RtpStreamAddedEventHandler(RtpStreamAdded);
RtpEvents.RtpStreamRemoved += new
RtpEvents.RtpStreamRemovedEventHandler(RtpStreamRemoved);
```

**รูปที่ 3.9** ตัวอย่างโค้ดที่ใช้ในการรับข้อมูลอาร์ทีพีแพ็กเก็ต 1

```
private void RtpStreamAdded(object sender,
RtpEvents.RtpStreamEventArgs ea)
{
ea.RtpStream.FrameReceived += new
RtpStream.FrameReceivedEventHandler(FrameReceived);
}
private void RtpStreamRemoved(object sender,
RtpEvents.RtpStreamEventArgs ea)
{
ea.RtpStream.FrameReceived -= new
RtpStream.FrameReceivedEventHandler(FrameReceived);
}
```

**รูปที่ 3.10** ตัวอย่างโค้ดที่ใช้ในการรับข้อมูลอาร์ทีพีแพ็กเก็ต 2

```
RtpSession rtpSession;
private void JoinRtpSession(string SessionName,string
name)
{
rtpSession = new RtpSession(ep, new
RtpParticipant(SessionName,
name), true, true);
rtpSender = rtpSession.CreateRtpSenderFec(name,
PayloadType.JPEG,
null, 0, 200);
}
private void LeaveRtpSession()
{
// Clean up all outstanding objects owned by the
RtpSession
rtpSession.Dispose();
}
```

**รูปที่ 3.11** ตัวอย่างโค้ดในการเข้าร่วมเซสชั่น

เราจำเป็นที่จะต้องระบุชนิด (type) ของอาร์ทีพีและรูปแบบของแพ็กเก็ตที่จะใช้ในแต่ละเซสชั่นจะสามารถรองรับได้ 1 รูปแบบ

```
private void FrameReceived(object sender,
RtpStream.FrameReceivedEventArgs ea)
{
System.IO.MemoryStream ms = new
MemoryStream(ea.Frame.Buffer);
pictureBox_Receive.Image = Image.FromStream(ms);
}
```

**รูปที่ 3.12** ตัวอย่างโค้ด หลังจาก เข้าร่วมเซสชั่นแล้วจะสามารถรับค่าอาร์ทีพีสตรีมบัฟเฟอร์

#### 3.2.6.2 RtpSender, RTPListener

ผู้ส่งมีหน้าที่ในการส่งโดยการแคปเจอร์ภาพและการสร้างแพ็กเก็ตอาร์ทีพีซึ่งข้อมูลอาจมาจากไฟล์หรือมาจากการแคปเจอร์ซึ่งจะถูกนำมาบีบอัดก่อนทำการส่ง เช่น การเปลี่ยนจากบิตแมป (bitmap) ไปเป็นเจเปก (jpeg)

สำหรับ RtpSender จะทำหน้าที่

- ส่งข้อมูลผ่านเครือข่ายเน็ตเวิร์ก

-สามารถเลือกที่จะส่งข้อมูลโดยที่มีหรือปราศจากการตรวจสอบความผิดพลาด

สำหรับ RtpListener จะทำหน้าที่

- 1 เทรด (thread) สำหรับรับข้อมูลผ่านเครือข่ายเน็ตเวิร์ค
- 1 เทรดสำหรับกระจายแพ็กเก็ตเพื่อจัดสรรสตรีมเพื่อนำไปประมวลผล

```
RtpSender rtpSender;
MemoryStream ms = new MemoryStream();
pictureBox_sender.Image.Save(ms, ImageFormat.Jpeg);
// Compressed
the captured image as JPEG image format
rtpSender.Send(ms.GetBuffer()); // Send The The
Comressed Image as
Bytes stream
```

**รูปที่ 3.13** ตัวอย่างโค้ด สำหรับการใช้งาน RTPSender

## 3.3 ซ็อกเกต (Socket)

โดยทั่วไปในการติดต่อสื่อสารที่เป็นที่นิยมใช้กันในปัจจุบันจะกระทำบนพื้นฐานของโปรโตคอลทีซีพี / ไอพีและชุดของซอฟต์แวร์ซึ่งทำงานกับแพ็กเก็ตตามมาตรฐานของทีซีพี/ไอพีนั้นจะถูกสร้างขึ้นมาเป็นส่วนหนึ่งของตัวระบบปฏิบัติการ (Operating System, OS) ดังนั้น เมื่อแอพพลิเคชั่นซอฟต์แวร์ใดๆ ที่ต้องการสื่อสารผ่านระบบเครือข่ายตามมาตรฐานโปรโตคอลทีซีพี/ไอพีจะต้องติดต่อกับระบบปฏิบัติการเพื่อขอรับบริการเซอร์วิส

สำหรับทางผู้พัฒนามาตรฐานโปรโตคอลทีซีพี/ไอพีไม่ต้องการให้โปรโตคอลทีซีพี/ไอพีใช้งานได้กับแพลตฟอร์มใดเป็นการเฉพาะหรือใช้งานได้กับเพียงระบบปฏิบัติการระบบใดระบบหนึ่ง ดังนั้นผู้พัฒนาจึงมีความระมัดระวังในการกำหนดมาตรฐานมิให้มีการอ้างถึงข้อมูลการเชื่อมต่อภายในซึ่งเป็นรูปแบบเฉพาะของแพลตฟอร์มใด แพลตฟอร์มหนึ่งหรืออ้างถึงวิธีการเชื่อมต่อกับแอพพลิเคชั่นซอฟต์แวร์ในลักษณะหรือรูปแบบเฉพาะของระบบปฏิบัติการจากผู้ผลิตรายใดรายหนึ่งและด้วยแนวคิดในการออกแบบที่ไม่ยึดติดกับแพลตฟอร์มหรือระบบปฏิบัติการใด ๆ นี้ทำให้อาจกล่าวได้ว่าโปรโตคอลทีซีพี/ไอพีมีลักษณะเป็น Loosely Specified Protocol Software Interface

จากที่กล่าวมาข้างต้นสามารถกล่าวโดยสรุปได้ว่า

*"มาตรฐานทีซีพี/ไอพีนั้นมิได้ระบุรายละเอียดเกี่ยวกับ การเรียกใช้งานซอฟต์แวร์ของโปรโตคอลทีซีพี/ไอพี ของแอพพลิเคชั่นซอฟต์แวร์ หากแต่มีเพียงข้อแนะนำเกี่ยวกับ ฟังก์ชันที่ควรจะมีในชุดของซอฟต์แวร์ของโปรโตคอลทีซีพี/ไอพีเท่านั้น โดยในส่วนของรายละเอียดที่นอกเหนือจากนั้นจะเปิดกว้างให้นักออกแบบระบบเป็นกำหนดและลงรายละเอียดเอง"*

ข้อดีของการออกแบบทีซีพี/ไอพีให้เป็นแบบ Loosely Specified Protocol Software Interface นั้นคือ ความยืดหยุ่น (flexible) และความทนทาน (tolerance) ทั้งนี้เพราะการไม่กำหนดรายละเอียดปลีกย่อยทำให้นักออกแบบระบบสามารถที่จะใส่ทีซีพี/ไอพีลงในระบบปฏิบัติการได้โดยง่ายไม่ว่าระบบปฏิบัติการนั้นจะเป็นระบบปฏิบัติการแบบง่าย ๆ อย่างที่ใช้กันในระบบสมองกลฝังตัว (Embedded System) ไปจนถึงระบบปฏิบัติการที่สลับซับซ้อนอย่างในระบบปฏิบัติการที่ใช้ในเครื่องคอมพิวเตอร์ขนาดใหญ่ (Super Computer) เป็นต้น นอกจากนั้นด้วยการที่ไม่กำหนดวิธีการเชื่อมต่อกับแอพพลิเคชั่นซอฟต์แวร์ทำให้นักออกแบบระบบสามารถเลือกใช้รูปแบบในการเชื่อมต่ออินเตอร์เฟตจึงอย่างอิสระ โดยสามารถใช้ได้ทั้งแบบโพรซิเตอร์รอล (Procedural) หรือการสื่อสารระหว่างวัตถุ (Message-Passing) เป็นต้น แต่ทั้งนี้ Loosely Specified Software Interface นั้นก็ยังมีข้อด้อยเช่นกันกล่าวคือการที่นักออกแบบระบบมีอิสระในการใส่รายละเอียดในส่วนการเชื่อมต่ออินเตอร์เฟสจึงทำให้เกิดความหลากหลายของรูปแบบการเชื่อมต่อ ซึ่งนั่นทำให้การเขียนแอพพลิเคชั่นโปรแกรมเพื่อการใช้งานทีซีพี/ไอพีมีความยุ่งยากและซับซ้อนมากยิ่งขึ้น อีกทั้งยังทำให้แอพพลิเคชั่นโปรแกรมนั้นยากแก่การใช้พอร์ตส่งไปยังระบบปฏิบัติการหรือแพลตฟอร์มอื่น ๆ

แม้ว่านักออกแบบระบบจะสามารถใส่ส่วนการเชื่อมต่อกับซอฟแวร์ (Software Interface) อย่างไรก็ได้ แต่ในทางปฏิบัติกลับกันมีรูปแบบของส่วนเชื่อมต่อของซอฟต์แวร์ทีซีพี/ไอพี (TCP/IP Software Interface) อยู่เพียงไม่กี่แบบเท่านั้นและส่วนเชื่อมต่อของซอฟต์แวร์ทีซีพี/ไอพีที่นิยมใช้งานกันอย่างแพร่หลายจะมีอยู่เพียง 2 รูปแบบเท่านั้นคือ

- ส่วนเชื่อมต่อซ็อกเกต (Socket Interface) เป็นส่วนเชื่อมต่อของซอฟต์แวร์ทีซีพี/ไอพีที่พัฒนาขึ้นโดย University of California ที่เบิร์กเลย์ (Berkley) ซึ่งในตอนต้นได้พัฒนารูปแบบการเชื่อมต่อแบบนี้เพื่อใช้กับระบบปฏิบัติการยูนิกส์ของเบิร์กเลย์ (Berkley Unix)

- ส่วนเชื่อมต่อของชั้นทรานสปอร์ต (Transport Layer Interface) หรือTLI เป็นส่วนเชื่อมต่อของซอฟต์แวร์ทีซีพี/ไอพีที่พัฒนาโดยเอทีแอนด์ที (AT&T) เพื่อใช้งานกับระบบปฏิบัติการยูนิกส์ซิสเต็มวี (System V Unix)

### 3.3.1 การใช้งานซ็อกเกต

ซ็อกเกต (Socket) ที่ถูกสร้างขึ้นนั้นจะมีวัตถุประสงค์ในการใช้งานอยู่เพียง 2 ลักษณะได้แก่

การรอรับการเชื่อมต่อ และการเชื่อมต่อไปยังเครื่องปลายทาง (Initiate a connection) โดยซ็อกเกตที่รอรับการเชื่อมต่อนั้นจะเป็นซ็อกเกตที่ใช้ในเซิร์ฟเวอร์แอพพลิเคชั่นซึ่งซ็อกเกตชนิดนี้มีชื่อเรียกว่าแพสซีฟซ็อกเกต (Passive Socket) ในขณะที่ซ็อกเกตที่พยายามเชื่อมต่อไปยังเครื่องปลายทางซึ่งใช้ในไคลแอนต์แอพพลิเคชั่นนั้นจะมีชื่อเรียกว่าแอคทีฟซ็อกเกต (Active Socket) ทั้งแอคทีฟซ็อกเกตและแพสซีฟซ็อกเกตนั้นในชั้นแรกจะถูกสร้างขึ้นมาด้วยวิธีการเดียวกัน ดังนั้นข้อแตกต่างของแอคทีฟซ็อกเกตและแพสซีฟซ็อกเกตจึงอยู่ที่วัตถุประสงค์การใช้งานเท่านั้น

**3.3.2 ชนิดของซ็อกเกต (Socket Type)**

ซ็อกเกตสามารถแบ่งออกเป็นประเภทตามลักษณะการรับส่งข้อมูลผ่านเครือข่ายได้เป็น 3 ประเภท ดังนี้

- สตรีมซ็อกเกต (Stream Sockets) หรือคอนเนคชั่นโอเรียลเด็ดซ็อกเกต (Connection Oriented Sockets)
- เดต้าแกรมซ็อกเกต (Datagram Sockets) หรือคอนเนคชั่นเลส (Connectionless Sockets)
- รอว์ซ็อกเกต (Raw Sockets)

**3.3.3 การทำงานกับซ็อกเกตในดอทเน็ต**

ในดอทเน็ตการทำงานที่เกี่ยวข้องกับซ็อกเกตจะต้องเรียกใช้งานเนมสเปซ (Namespace) ที่ชื่อว่า System.Net.Sockets ซึ่งเป็น Namespace ที่บรรจุคลาสต่าง ๆ ที่สนับสนุนการทำงานกับซ็อกเกตโดยคลาสที่เกี่ยวข้องกับการทำงานหลัก ๆ กับซ็อกเกตมีดังนี้

- MulticastOption
- NetworkStream
- TcpClient
- TcpListener
- UdpClient
- SocketException
- Socket

โดยคลาสซ็อกเกต (Socket Class) จะมีฟังก์ชันพื้นฐานต่างๆ ที่จำเป็นสำหรับการสร้างแอพพลิเคชั่นซึ่งใช้งานกับซ็อกเกตสำหรับคุณสมบัติ (Properties) ที่สำคัญสำหรับ System.Net.Sockets.Socket ได้แก่

- AddressFamily
- Available
- Blocking
- Connected
- LocalEndPoint

- ProtocolType
- RemoteEndPoint
- SocketType

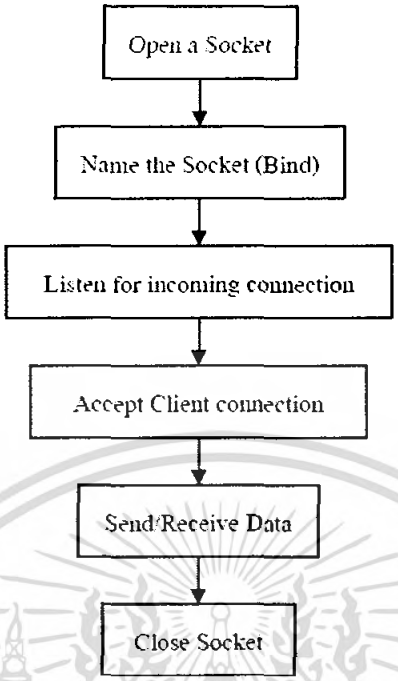
และเมทธอด (Method) ที่สำคัญของ System.Net.Sockets.Socket มีดังนี้

- Accept()
- Bind()
- Close()
- Connect()
- GetSocketOption()
- IOControl()
- Listen()
- Receive()
- Poll()
- Select()
- Send()
- SetSocketOption()
- Shutdown()

**3.3.4 การทำงานกับซ็อกเกต**

**3.3.4.1 แพสซีฟซ็อกเกต**

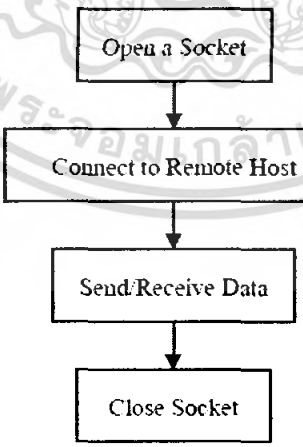
แพสซีฟซ็อกเกตหรือซ็อกเกตซึ่งใช้ในแอพพลิเคชั่นบนเซิร์ฟเวอร์ (Server Application) นั้นมีขั้นตอนการสร้างดังนี้




**รูปที่ 3.14** ไดอะแกรมขั้นตอนการสร้างแพสซีฟซ็อกเกต

**3.3.4.2 แอคทีฟซ็อกเกต**

แอคทีฟซ็อกเกตหรือซ็อกเกตประเภทที่ใช้งานในแอพพลิเคชั่นบนไคลแอนต์ (Client Application) มีขั้นตอนการสร้างดังนี้




**รูปที่ 3.15** ไดอะแกรมขั้นตอนการสร้างแอคทีฟซ็อกเกต

**3.3.5 ปัญหาด้านการจัดการข้อมูลที่ส่งผ่านระหว่างเครือข่าย**

ในการสื่อสารข้อมูลระหว่างเครือข่ายนั้นสิ่งที่อาจเกิดขึ้นได้คือการสูญหายหรือความล่าช้าของข้อมูลที่ส่งจากต้นทางไปยังปลายทาง แม้ว่าโปรโตคอลทีซีพีนั้นจะรับประกันในเรื่องของการส่งข้อมูลว่าจะไม่มีเรื่องของข้อมูลสูญหายระหว่างการรับส่งหรือหากเกิดการสูญหายของข้อมูลขึ้น ข้อมูลชุดที่สูญหายนั้นก็จะถูกส่งใหม่ (retransmit) แต่ทั้งนี้แม้ข้อมูลจะไม่สูญหายแต่ TCP Protocol นั้นเมื่อรับข้อมูลเข้าสู่บัฟเฟอร์ของทีซีพี (TCP Buffer) แล้วนั้นจะทำการเรียงลำดับข้อมูล แล้วรวมข้อมูลที่รับมานั้นเข้าด้วยกันในกรณีที่ข้อมูลนั้นมีขนาดใหญ่แล้วถูกแบ่งออกเป็นส่วนๆ แล้วทยอยส่งมาทีละส่วนหรือในกรณีที่ข้อมูลมีขนาดเล็กก็จะนำข้อมูลนั้นมาเก็บเรียงต่อกันไปภายในบัฟเฟอร์ของทีซีพีโดยไม่มีการทำเครื่องหมายเพื่อระบุถึงขอบเขตของข้อมูลแต่ละชุด

โปรโตคอลทีซีพีนั้นไม่มีการป้องกันขอบเขตของแต่ละชุดข้อมูล (Unprotected Message Boundary) ดังนั้นในการเขียนโปรแกรมเพื่อสื่อสารข้อมูลด้วยโปรโตคอลทีซีพีจึงต้องออกแบบโปรแกรมให้สามารถรองรับปัญหานี้ด้วยสำหรับวิธีการแก้ปัญหาดังกล่าวนั้น โดยทั่วไปนิยมใช้กันอยู่ 3 วิธี ได้แก่

- การจำกัดขนาดของแต่ละชุดข้อมูลที่จะส่งให้มีขนาดเท่ากัน
- การส่งขนาดของชุดข้อมูลไปพร้อมกันกับชุดข้อมูลนั้น
- การใส่สัญลักษณ์เพื่อบอกขอบเขตของข้อมูลแต่ละชุด

ซึ่งในแต่ละวิธีการที่ใช้แก้ปัญหานั้นก็มีทั้งข้อดีและข้อเสียในตัวเองดังที่จะได้กล่าวถึงต่อไป

**3.3.5.1 การจำกัดขนาดของแต่ละชุดข้อมูลที่จะส่งให้มีขนาดเท่ากัน**

เป็นวิธีการแก้ปัญหาที่ง่ายที่สุด เพราะเมื่อขนาดของข้อมูลที่ส่งแต่ละชุดมีขนาดเท่ากัน โปรแกรมจะทราบได้ทันทีว่าข้อมูลมาครบแล้วหรือยังโดยการตรวจสอบขนาดของชุดข้อมูลนั้นนั่นเอง นอกจากนี้แม้ชุดข้อมูลมากกว่า 1 ชุดมาถึงพร้อม ๆ กันก็ยังสามารถแยกชุดข้อมูลออกจากกันได้โดยการนับจำนวนไบต์ในข้อมูลที่รับเข้ามานั้นเทียบกับขนาดมาตรฐานของชุดข้อมูลปกติโดยทั่วไปถ้ามีส่วนของโปรแกรมดังนี้

```
byte[] data = new byte[1024];
....
....
int sent = socket.Send(data);
```

**รูปที่ 3.16** ส่วนของโปรแกรมการจำกัดขนาดของแต่ละชุดข้อมูลที่จะส่งให้มีขนาดเท่ากัน

จะทำให้คิดได้ว่าชุดของข้อมูลขนาด 1024 ไบต์จะถูกส่งไปยังปลายทางทั้งหมด แต่ในทางปฏิบัติข้อมูลอาจถูกส่งไปไม่พร้อมกันในคราวเดียว อันเนื่องมาจากสาเหตุเช่น ขนาดของบัฟเฟอร์ของทีซีพีและปริมาณของข้อมูลโดยรวมซึ่งส่งออกจากเครื่องในเวลานั้น ดังนั้นเมื่อมีโอกาสที่ข้อมูลอาจไม่สามารถส่งออกไปได้ทั้งหมดในคราวเดียวเช่นนี้ นักเขียนโปรแกรมจึงต้องตรวจสอบการทำงานของโปรแกรมว่าทำการส่งข้อมูลออกไปหมดแล้วหรือไม่ซึ่งการตรวจสอบสามารถกระทำได้โดยการใช้ค่าที่ Send() รีเทิร์นให้กลับตัวแปร (ซึ่งในที่นี้ก็คือตัวแปรชื่อ sent) โดยค่าที่รีเทิร์นดังกล่าวนี้จะบอกว่าได้ทำการส่งข้อมูลออกไปเท่าใด ถ้าจำนวนข้อมูลที่ส่งออกนั้นยังน้อยกว่าข้อมูลที่ต้องการส่ง (ในที่นี้คือ 1024 ไบต์) ก็ให้ทำการวนลูป (loop) เพื่อส่งไปเรื่อย ๆ จนกว่าจะส่งข้อมูลออกไปทั้งหมด ดังในส่วนของโปรแกรมตัวอย่างข้างต้นและเช่นเดียวกันสำหรับการรับข้อมูลคือไม่สามารถรับประกันได้ว่าจะสามารถรับข้อมูลจากฝ่ายที่ส่งข้อมูลได้ทั้งหมดในการใช้คำสั่ง Receive() เพียงรอบเดียว ดังนั้นการวนลูปเพื่อรับข้อมูลจึงจำเป็น แต่ทั้งนี้ทั้งนั้นการวนลูปเพื่อรับในลักษณะนี้จะสามารถกระทำได้ก็ต่อเมื่อทราบขนาดที่แน่นอนของข้อมูลที่จะถูกส่งมา ซึ่งส่วนของโปรแกรมข้างบนนั้นจะทำหน้าที่ในการรับข้อมูลจากการส่งข้อมูลแบบที่กำหนดขนาดไว้แล้ว (*fixed size*)

**3.3.5.2 การส่งขนาดข้อมูลไปพร้อมกันกับชุดข้อมูลนั้น**

จากหัวข้อที่แล้วการส่งข้อมูลด้วยการจำกัดขนาดของข้อมูลที่จะส่งให้มีขนาดเท่ากันทุกแพ็กเก็ตนั้นจะสามารถแก้ปัญหาในเรื่องของขอบเขตของแมสเสจ (message boundaries) ได้ แต่ข้อเสียของวิธีการดังกล่าวคือถ้าข้อมูลที่จะส่งนั้นมีขนาดที่ยาวมาก ก็จำเป็นที่จะต้องแบ่งข้อมูลนั้นออกเป็นส่วน ๆ ที่เท่า ๆ กันและทำการส่งข้อมูลออกไปครั้งละ 1 ชุด ซึ่งการส่งข้อมูลออกไปหลาย ๆ ชุด แทนที่จะเป็นชุดเดียวยาว ๆ เช่นที่ยกตัวอย่างมานี้จะมีผลให้เกิดการสิ้นเปลืองแบนวิดท์ (bandwidth) ของระบบสื่อสารโดยใช่เหตุด้วยเช่นกัน

เพื่อแก้ปัญหานี้จึงได้มีการคิดค้นวิธีการที่จะให้สามารถส่งชุดข้อมูลที่มีขนาดต่างกันได้ แต่การจะส่งชุดข้อมูลซึ่งมีขนาดแตกต่างกันไปได้นั้นทางฝั่งรับจะต้องทราบขนาดของชุดข้อมูลที่จะส่งไปด้วย ซึ่งก็สามารถกระทำได้โดยการส่งขนาดของชุดข้อมูลไปรวมไปกับชุดข้อมูลที่จะส่งนั้น ตัวอย่างเช่น

9message 1

**รูปที่ 3.17** *ส่วนของโปรแกรมการส่งขนาดของชุดข้อมูลไปรวมไปกับชุดข้อมูล*

ในที่นี้ขนาดของชุดข้อมูลที่จะส่งไปด้วยคือ 9 สำหรับตัวข้อมูลคือ message 1 แต่การใช้ตัวอักษรเพื่อบอกขนาดถ้าชุดข้อมูลมีขนาดใหญ่ก็ต้องใช้หลายไบต์ในการบอกขนาด (1 ตัวอักษรต่อ 1 ไบต์) เพื่อลดปริมาณข้อมูลที่จะส่ง

โดยทั่วไปจึงไม่ใช้ตัวอักษรในการบอกขนาดแต่จะใช้ข้อมูลชนิดไบต์ในการบอกขนาดแทนเช่นถ้าใช้ 2 ไบต์แรกของชุดข้อมูล จะได้ว่าชุดข้อมูลนั้นสามารถมีขนาดได้มากที่สุด 216 ไบต์หรือ64 กิโลไบต์นั่นเอง ส่วนของโปรแกรมต่อไปนี้จะเป็นตัวอย่างของการทำงานรับ-ส่งชุดข้อมูลที่ไม่จำกัดขนาด

**3.3.5.3 การใส่สัญลักษณ์เพื่อบอกขอบเขตของข้อมูลแต่ละชุด**

เป็นวิธีการแก้ไขปัญหาเกี่ยวกับขอบเขตของแมสเสจวิธีสุดท้าย เป็นวิธีการแก้ปัญหาโดยการใส่สัญลักษณ์ที่ได้กำหนดไว้ล่วงหน้าเพื่อบอกจุดสิ้นสุดของข้อมูลแต่ละชุด เมื่อรับข้อมูลเข้ามาโปรแกรมจะต้องทำการตรวจข้อมูลนั้นทีละตัวอักษรเพื่อหาสัญลักษณ์บอกจุดสิ้นสุด

สำหรับวิธีการนี้จะมีข้อด้อยอยู่ 2 ประการคือ ต้องกำหนดสัญลักษณ์ที่ใช้ระบุขอบเขตของชุดข้อมูลซึ่งจะต้องไม่ซ้ำกันกับข้อมูลที่จะส่งและการที่จะต้องตรวจสอบข้อมูลที่ได้รับเข้ามาทีละ 1 ตัวอักษรก็จะทำให้เกิดโอเวอร์เฮด (overhead) ขึ้น

ใน C# มีคลาสซึ่งช่วยในการทำงานในลักษณะของการใส่สัญลักษณ์เพื่อบอกขอบเขตของข้อมูล โดยที่นักเขียนโปรแกรมไม่ต้องเขียนโค้ดเพื่อทำการตรวจสอบตัวอักษรทีละตัวเพื่อหาสัญลักษณ์ คลาสซึ่งช่วยทำงานดังกล่าวได้แก่คลาสชื่อ NetworkStream ซึ่งทำหน้าที่เป็นส่วนเชื่อมต่อกับซ็อกเกต, คลาสชื่อ StreamReader และ StreamWriter ทำหน้าที่อ่าน/เขียนข้อมูลชนิดเทคที่ได้จาก NetworkStream

## 3.4 ภาพขาว-ดำ (Grayscaleหรือ*Grayscale digital image*)

ภาพเคลื่อนไหวขาว-ดำ (grayscaleหรือgrayscale digital image) คือภาพที่ซึ่งค่าของแต่ละพิกเซล (pixel) เป็นซิงเกิลแซมเปิล (single sample) โดยภาพที่แสดงออกมาจะเป็นลักษณะของการประกอบกันของเฉดสีเทา โดยมีตั้งแต่สีดำในที่ที่มีความหนาแน่นน้อยที่สุดไปจนถึงสีขาวในที่ที่มีความหนาแน่นมากที่สุด

องค์ประกอบสำคัญที่แซมเปิลสามารถถูกแสดงอกมาในเฉดสีใดเฉดสีหนึ่งหรือสีที่แสดงออกมาสามารถเปลี่ยนแปลงไปตามค่าความเข้มข้น ภาพเคลื่อนไหวขาวดำจะอยู่ในช่วง ดำ->ขาว ซึ่งองค์ประกอบในภาพจะประกอบด้วยสีเพียง 2 สี เท่านั้นคือ ดำและขาว ซึ่งในบางครั้งเฉดสีเทาอาจจะมีชื่อเรียกว่า "black-and-white photography"

ภาพขาว-ดำ คำนวนได้มาจากการวัดความเข้มข้นของแสงแต่ละพิกเซล (pixel) โดยอาศัยหลักการของอิเล็กโตรแมกเนติกสเปกตรัม (electromagnetic spectrum)

### 3.4.1 การแปลงภาพสีมาเป็นภาพขาว-ดำ

ในการแปลงสีใดๆก็ตามให้เป็นระดับต่างๆของสีเทา อันดับแรกคือจะต้องจัดรูปสีดังกล่าวให้เป็นในรูปแบบของอาร์จีบี (RGB) เสียก่อน ต่อมา คือการเพิ่มค่าของอาร์จีบีดังรูปที่ 2.18

| | | |
|---|---|---|
| -R | เพิ่มขึ้น | 30% |
| -G | เพิ่มขึ้น | 59% |
| -B | เพิ่มขึ้น | 11% |

**รูปที่ 3.18** ค่าของเปอร์เซนต์ที่ใช้ในการแปลงภาพสี

อย่างไรก็ตามแต่เสกลอาจจะอยู่ในช่วงของ 0.0-0.1, 0 ถึง 255, หรือ 0%-100% ยังคงใช้หลักการเดิม โดยหลักเปอร์เซนต์ดังกล่าวเป็นการเลือกค่าที่เหมาะสมที่สามารถทำให้ตาของมนุษย์โดยทั่วไปเห็นเป็นภาพขาว-ดำ

**รูปที่ 3.19** ภาพสีธรรมชาติ

**รูปที่ 3.20** ภาพสีขาว-ดำ

**3.4.2 หลักการแปลงภาพสีเป็นขาว-ดำ**

หลักการคือจะใช้เทคนิคของคัลเลอร์เมตริกซ์ (ColorMatrix) เข้ามาใช้ในโปรแกรม คัลเลอร์เมตริกซ์คือ 5x5 เมตริกซ์ซึ่งสามารถสร้างขึ้นมาเพื่อใช้ในการแก้ไขภาพสี

คัลเลอร์เมตริกซ์เป็น 5x5 เมตริกซ์ที่ใช้บรรจุค่าโคออร์ดิเนตของ RGBA คือ red, green, blue, w โดยค่า w มักเป็นค่า 1 สำหรับการใส่ค่าลงไปในเมตริกซ์จะเป็น อยู่ในช่วง 0.0-1.0 โดยช่วงที่มีความเข้มข้นที่สุดคือ 255 ให้มีค่าเป็น 1. 0 ซึ่งโดยทั่วไปแล้วเมตริกซ์จะสามารถอยู่ในรูป 4x4 ได้ ถ้าอยู่ในลักษณะของลิเนียร์ (linear) สามารถจัดการในเรื่องสเกลโรเตชั่น (scaling rotation) เป็นต้น อย่างไรก็ตามการจัดการในเรื่องของสี การเปลี่ยนแปลงของสี จะต้องการค่าของสีที่เพิ่ม ซึ่งนอนลิเนียร์ (non linear) จะสามารถจัดการในเรื่องนี้ได้ ดังนั้นอิลิเมนต์ (element) สุดท้ายที่เพิ่มเข้ามาเป็นตัวที่จะบอกว่าจะเป็นลิเนียร์หรือนอนลิเนียร์เป็นคล้ายตัวดัมมี่ (dummy) คือมีค่า 0 หรือ 1 ได้เท่านั้น

ImageAttributes เป็นออบเจกต์ (object) ที่รวมข้อมูลบิตแมป, เมตาไฟล์คัลเลอร์ (metafile color) ที่ถูกจัดการระหว่างถอดความ ซึ่งออบเจกต์ ImageAttributes จะดูแลในส่วนของ การปรับเปลี่ยนสี เช่น ภาพขาว-ดำ (Grayscale) , แกมมา (gamma) , color (ภาพสี) โดยผ่านเมทธอด SetColorMatrix

การใช้เมกตริกซ์จะเป็นผลดีเพราะไม่จำเป็นจะต้องทราบข้อมูลเกี่ยวกับ pixel ของรูปนั้นๆและโค้ดค่อนข้างสั้นอีกด้วย

จากตัวอย่างด้านล่าง สเกลเวกเตอร์สี [255,128,102,255] เปลี่ยนเป็น .5 และเพิ่มค่า 26 เข้าไปในองค์ประกอบของ RGB จากนั้นค่าของ A ให้เป็นค่าของ ความเข้มข้นสูงที่สุดคือ 1.0 ในส่วนของตัวสุดท้ายให้เพิกเฉยไว้

$$\begin{bmatrix} 1.0 & 0.5 & 0.4 & 1.0 & 1.0 \end{bmatrix} \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0.1 & 0.1 & 0.1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.6 & 0.35 & 0.3 & 1.0 & 1.0 \end{bmatrix} \Rightarrow \begin{bmatrix} 153 & 89 & 77 & 255 \end{bmatrix}$$

**รูปที่ 3.21** เมตริกซ์ที่ใช้ในการเปลี่ยนแปลงสี

จากตัวอย่างจะเห็นได้ว่าจะสามารถเปลี่ยนจากสี เป็นสี

ในการนำหลักการดังกล่าวไปใช้จะค่อนข้างง่าย โดยการใช้งานจะใช้ออปเจคคัลเลอร์เมตริกซ์และออปเจกต์ ImageAttributes ร่วมกันจากนั้นให้เอาค่าของออปเจกต์ ImageAttributes เข้าไปเป็นพารามิเตอร์ของ method Graphics.DrawImage เพื่อวาดภาพตามสีที่ต้องการ

## 3.5 ความเป็นส่วนตัว (Privacy)

การทำให้ข้อมูลเป็นความลับ (Confidentiality) เพื่อป้องกันไม่ให้ผู้ที่ไม่มีสิทธิ์ในการเข้าถึงข้อมูลสามารถเข้าถึงข้อมูลได้

### 3.5.1 การเข้ารหัสลับข้อมูล (Cryptography)

การเข้ารหัสลับข้อมูลโดยพื้นฐานแล้วจะเกี่ยวข้องกับวิธีการทางคณิตศาสตร์เพื่อใช้ในการป้องกันข้อมูลหรือข้อความตั้งต้นที่ต้องการส่งไปถึงผู้รับ ข้อมูลตั้งต้นจะถูกแปรเปลี่ยนไปสู่ข้อมูลหรือข้อความอีกรูปแบบหนึ่งที่ไม่สามารถอ่านเข้าใจได้โดยใครก็ตามที่ไม่มีกุญแจสำหรับเปิดดูข้อมูลนั้น เราเรียกกระบวนการในการแปรรูปของข้อมูลตั้งต้นว่าการเข้ารหัสลับข้อมูล (Encryption) และกระบวนการในการแปลงข้อความที่ไม่สามารถอ่านและทำความเข้าใจให้กลับไปสู่ข้อความดั้งเดิม ว่าการถอดรหัสข้อมูล (Decryption)

### 3.5.2 อัลกอริทึมในการเข้ารหัสลับข้อมูล

อัลกอริทึมในการเข้ารหัสลับข้อมูลมี 2 ประเภทหลัก คือ

#### 3.5.2.1 อัลกอริทึมแบบสมมาตร (Symmetric key algorithms)

อัลกอริทึมแบบนี้จะใช้กุญแจที่เรียกว่า กุญแจลับ (Secret key) ซึ่งมีเพียงหนึ่งเดียวเพื่อใช้ในการเข้าและถอดรหัสข้อความที่ส่งไป อัลกอริทึมยังสามารถแบ่งย่อยออกเป็น 2 ประเภท ได้แก่ แบบบล็อค (Block Algorithms) ซึ่งจะทำการเข้ารหัสทีละบล็อค 1 บล็อคประกอบด้วยหลายไบต์ เช่น 64 ไบต์ เป็นต้นและแบบสตรีม (Stream Algorithms) ซึ่งจะทำการเข้ารหัสทีละไบต์อัลกอริทึมแบบนี้จะใช้กุญแจที่เรียกว่า กุญแจลับ (Secret key) ซึ่งมีเพียงหนึ่งเดียวเพื่อใช้ในการเข้าและถอดรหัสข้อความที่ส่งไป

#### 3.5.2.2 อัลกอริทึมแบบอสมมาตร (Asymmetric key algorithms)

อัลกอริทึมนี้จะใช้กุญแจสองตัวเพื่อทำงาน ตัวหนึ่งใช้ในการเข้ารหัสและอีกตัวหนึ่งใช้ในการถอดรหัสข้อมูลที่เข้ารหัสมาโดยกุญแจตัวแรก อัลกอริทึมกลุ่มสำคัญในแบบอสมมาตรนี้คือ อัลกอริทึมแบบกุญแจสาธารณะ (Public keys Algorithms) ซึ่งใช้กุญแจที่เรียกกันว่า กุญแจสาธารณะ (Public keys) ในการเข้ารหัสและใช้กุญแจที่เรียกกันว่า กุญแจส่วนตัว (Private keys) ในการถอดรหัสข้อมูลนั้น กุญแจสาธารณะนี้สามารถส่งมอบให้กับผู้อื่นได้ เช่น เพื่อนร่วมงานที่เราต้องการติดต่อด้วยหรือแม้กระทั่งวางไว้บนเว็บไซต์เพื่อให้ผู้อื่นสามารถดาวน์โหลดไปใช้งานได้ สำหรับกุญแจส่วนตัวนั้นต้องเก็บไว้กับผู้เป็นเจ้าของกุญแจส่วนตัวเท่านั้นและห้ามเปิดเผยให้ผู้อื่นทราบโดยเด็ดขาด

อัลกอริทึมแบบกุญแจสาธารณะยังสามารถประยุกต์ใช้ได้กับการลงลายมือชื่ออิเล็กทรอนิกส์ (ซึ่งเปรียบเสมือนการลงลายมือชื่อของเราที่ใช้กับเอกสารสำนักงานทั่วไป การลงลายมือชื่อนี้จะเป็นการพิสูจน์ความเป็นเจ้าของและสามารถใช้ได้กับการทำธุรกรรมต่างๆ บนอินเทอร์เน็ต เช่น การซื้อสินค้า เป็นต้น วิธีการใช้งานคือ ผู้เป็นเจ้าของกุญแจส่วนตัวลงลายมือชื่อของตนกับข้อความที่ต้องการส่งไปด้วยกุญแจส่วนตัว แล้วจึงส่ง

ข้อความนั้นไปให้กับผู้รับ เมื่อได้รับข้อความที่ลงลายมือชื่อมา ผู้รับสามารถใช้กุญแจสาธารณะที่เป็นของกุญแจส่วนตัวคู่นั้นเพื่อตรวจสอบว่าเป็นข้อความที่มาจากผู้ส่งนั้นหรือไม่

**3.5.3 ปัญหาของอัลกอริทึมแบบสมมาตร**

อัลกอริทึมแบบสมมาตรมีความสำคัญไม่ด้อยไปกว่าอัลกอริทึมแบบอสมมาตร ทั้งนี้เนื่องจากอัลกอริทึมแบบแรกทำงานได้รวดเร็วกว่าและง่ายต่อการใช้งานกว่าแบบหลัง อย่างไรก็ตามอัลกอริทึมแบบสมมาตรยังมีปัญหาที่สำคัญ 3 ประการ ซึ่งเป็นข้อจำกัดในการใช้งานอัลกอริทึมนี้

- ในการใช้งานอัลกอริทึมนี้ สองกลุ่มที่ต้องการแลกเปลี่ยนข้อมูลกันเช่น องค์กร ก และ ข จำเป็นต้องแลกเปลี่ยนกุญแจลับกันก่อน ซึ่งอาจหมายถึงส่งมอบกุญแจลับให้กับอีกกลุ่มหนึ่ง การแลกเปลี่ยนกุญแจนั้นอาจทำได้อย่างยุ่งยากและไม่สะดวก

- ทั้งสองกลุ่มต้องรักษากุญแจลับนั้นไว้เป็นอย่างดี ห้ามเปิดเผยให้ผู้อื่นล่วงรู้โดยเด็ดขาด การที่กุญแจถูกเปิดเผยออกไปสู่ผู้อื่น อาจก่อให้เกิดปัญหากับกลุ่มที่ไม่ทราบนี้ได้ เช่น กลุ่มนี้อาจส่งข้อความที่เป็นความลับไปให้กับอีกกลุ่มหนึ่ง แต่ข้อความนี้อาจถูกเปิดเผยได้โดยใช้กุญแจลับที่ล่วงรู้โดยผู้อื่น

- สำหรับสองกลุ่มที่ต้องการติดต่อกัน จำเป็นต้องใช้กุญแจลับเป็นจำนวน 1 กุญแจเพื่อติดต่อกัน สมมติว่ามีผู้ที่ต้องติดต่อกันเป็นจำนวน n กลุ่ม จำนวนกุญแจลับทั้งหมดที่ต้องแลกเปลี่ยนกันคิดเป็นจำนวนทั้งหมด C 2n หรือเท่ากับ n (n-1)/2 กุญแจ ซึ่งจะเห็นได้ว่าจำนวนกุญแจมีมากมายเกินไป ซึ่งอาจก่อให้เกิดปัญหาด้านการรักษาความปลอดภัยให้กับกุญแจเหล่านี้

อัลกอริทึมแบบกุญแจสาธารณะ (ซึ่งเป็นแบบอสมมาตร) ช่วยแก้ปัญหาเหล่านี้ได้ทั้งหมด ผู้ใช้ที่ถือกุญแจส่วนตัวและต้องการให้บุคคลอื่นที่ตนติดต่อด้วยส่งเอกสารหรือข้อความที่เข้ารหัสมาหาตน สามารถเผยแพร่กุญแจสาธารณะของตนไว้บนเว็บไซต์หรือในที่สาธารณะซึ่งผู้อื่นสามารถเข้ามาดาวน์โหลดไปใช้งานได้ วิธีการใช้งานคือให้บุคคลอื่นที่มาดาวน์โหลดกุญแจไปนั้นทำการเข้ารหัสข้อความที่ต้องการส่งด้วยกุญแจสาธารณะ แล้วจึงส่งข้อความที่เข้ารหัสไปให้กับผู้เป็นเจ้าของกุญแจสาธารณะ โดยวิธีนี้จะไม่มีผู้อื่นสามารถเปิดดูข้อความที่เข้ารหัสนั้นได้ยกเว้นผู้ที่ถือกุญแจส่วนตัวที่เป็นคู่ของกุญแจสาธารณะสามารถเปิดข้อความนี้ดูได้

การเผยแพร่กุญแจสาธารณะในสถานที่ต่างๆ ได้ทำให้ลดความยุ่งยากในการแลกเปลี่ยนกุญแจกันซึ่งเป็นปัญหาข้อแรกของการเข้ารหัสแบบสมมาตร สำหรับปัญหาที่ว่าทั้งสองกลุ่มจะต้องรักษากุญแจลับไว้เป็นอย่างดีนั้น วิธีการของกุญแจสาธารณะจะทำให้ผู้ที่ต้องรับผิดชอบเหลือเพียงผู้เดียว กล่าวคือ ผู้ถือกุญแจส่วนตัวซึ่งห้ามให้ผู้อื่นล่วงรู้โดยเด็ดขาด

สำหรับปัญหาที่สามที่ว่าจำนวนกุญแจลับที่จำเป็นต้องใช้มีมากมายเกินไป วิธีการของกุญแจสาธาณะจะใช้จำนวนกุญแจที่ประหยัดกว่า เนื่องจากกุญแจสาธารณะ 1 กุญแจของกลุ่มๆ หนึ่งจะสามารถเผยแพร่ให้กับกี่กลุ่มก็ได้ที่เราต้องการติดต่อด้วย แทนที่จะเป็น 1 กุญแจลับต่อสองกลุ่มที่ต้องการติดต่อกัน ดังนั้นถ้ามีกลุ่มที่ต้อง

ติดต่อกันจำนวน n กลุ่ม จำนวนกุญแจส่วนตัวที่ต้องระวังรักษาก็คือ n กุญแจ ซึ่งจะเห็นได้ว่าลดลงไปได้เป็นจำนวนมาก

ข้อเสียที่สำคัญของระบบกุญแจสาธารณะที่สำคัญคือ ต้องใช้เวลาในการคำนวณการเข้าและถอดรหัสเมื่อเทียบกับระบบกุญแจสมมาตร และอาจใช้เวลาเป็นพันเท่าของเวลาที่ใช้โดยระบบกุญแจสมมาตร

**3.5.4 ความแข็งแกร่งของอัลกอริทึมสำหรับการเข้ารหัสลับ**

ความแข็งแกร่งของอัลกอริทึมหมายถึงความยากในการที่ผู้บุกรุกจะสามารถถอดรหัสข้อมูลได้โดยปราศจากกุญแจที่ใช้ในการเข้ารหัส ซึ่งจะขึ้นอยู่กับปัจจัยดังนี้

- การเก็บกุญแจเข้ารหัสไว้อย่างเป็นความลับ ผู้เป็นเจ้าของกุญแจลับหรือส่วนตัวต้องระมัดระวังไม่ให้กุญแจสูญหายหรือล่วงรู้โดยผู้อื่น

- ความยาวของกุญแจเข้ารหัส ปกติกุญแจเข้ารหัสจะมีความยาวเป็นบิต ยิ่งจำนวนบิตของกุญแจยิ่งมากยิ่งทำให้การเดาเพื่อสุ่มหากุญแจที่ถูกต้องเป็นไปได้ยากยิ่งขึ้น ดเช่น กุญแจขนา1 บิต จะสามารถแทนตัวเลขได้ 2 ค่าคือ 0 กับ 1 กุญแจขนาด 2 บิต จะเป็นไปได้ 4 ค่าคือ 0, 1, 2, 3 เป็นต้น

- ความไม่เกรงกลัวต่อการศึกษาหรือดูอัลกอริทึมเพื่อหารูปแบบของการเข้ารหัส อัลกอริทึมที่ดีต้องเปิดให้ผู้รู้ทำการศึกษาในรายละเอียดได้โดยไม่เกรงว่าผู้ศึกษาจะสามารถจับรูปแบบของการเข้ารหัสได้

- การมีประตูลับในอัลกอริทึม อัลกอริทึมที่ดีต้องไม่แฝงไว้ด้วยประตูลับที่สามารถใช้เป็นทางเข้าไปสู่อัลกอริทึม แล้วอาจใช้เพื่อทำการถอดรหัสข้อมูลได้ ประตูลับนี้ทำให้ไม่จำเป็นต้องใช้กุญแจในการถอดรหัส

- ความไม่เกรงกลัวต่อปัญหาการหาความสัมพันธ์ในข้อมูลที่ได้รับ กล่าวคือเมื่อผู้บุกรุกทราบข้อมูลบางอย่างที่เป็นข้อมูลดั้งต้นซึ่งยังไม่ได้เข้ารหัส รวมทั้งมีข้อมูลที่เข้ารหัสแล้ว (ของข้อมูลดั้งต้นนั้น) ผู้บุกรุกอาจจะสามารถหาความสัมพันธ์ระหว่างข้อความทั้งสองนั้นได้ ซึ่งจะเป็นวิธีการในการถอดรหัสข้อมูลได้ ปัญหานี้เรียกกันว่า Known plaintext attack คำว่า plaintext หมายถึงข้อความตั้งต้นที่ยังไม่ได้ผ่านการเข้ารหัส

- คุณสมบัติของข้อความตั้งต้น คุณสมบัตินี้อาจใช้เป็นช่องทางในการถอดรหัสข้อมูลได้ อัลกอริทึมที่ดีต้องไม่ใช้คุณสมบัติของข้อความเป็นกลไกในการเข้ารหัสข้อมูล

คำแนะนำในการเลือกใช้อัลกอริทึมคือให้ใช้อัลกอริทึมที่ได้มีการใช้งานมาเป็นระยะเวลานานแล้ว ทั้งนี้เนื่องจากหากปัญหาของอัลกอริทึมนี้มีจริง ก็คงเกิดขึ้นมานานแล้วและก็คงเป็นที่ทราบกันแล้ว นั่นคืออย่างน้อยที่สุดจวบจนกระทั่งถึงปัจจุบัน ก็ยังไม่มีการบุกรุกที่ทำให้อัลกอริทึมนั้นไม่สามารถใช้งานได้อย่างปลอดภัยเป็นที่ประจักษ์ ดังนั้นจึงไม่ควรใช้อัลกอริทึมใหม่ๆ ที่เพิ่งได้มีการนำเสนอกันสู่สาธารณะ เพราะอาจมีช่องโหว่แฝงอยู่และยังไม่เป็นที่ทราบในขณะนี้

**3.5.5 ความยาวของกุญแจที่ใช้ในการเข้ารหัสลับ**

ความยาวของกุญแจเข้ารหัสมีหน่วยนับเป็นบิต หนึ่งบิตในคอมพิวเตอร์เป็นตัวเลขฐานสองที่ประกอบด้วยค่า 0 และ 1 กุญแจที่มีความยาว 1 บิต ตัวเลขที่เป็นไปได้เพื่อแทนกุญแจนั้น จึงอาจมีค่าเป็น 0 หรือ 1 กุญแจที่มีความยาว 2 บิต ตัวเลขที่เป็นไปได้จึงเป็น 0, 1, 2 และ 3 ตามลำดับ กุญแจที่มีความยาว 3 บิต ตัวเลขที่เป็นไปได้จะอยู่ระหว่าง 0 ถึง 7 ดังนั้นเมื่อเพิ่มความยาวของกุญแจทุกๆ 1 บิต ค่าที่เป็นไปได้ของกุญแจจะเพิ่มขึ้นเป็นสองเท่าตัวหรือจำนวนกุญแจที่เป็นไปได้จะเพิ่มขึ้นเป็น 2 เท่าตัวนั่นเอง

ฉะนั้นจะเห็นได้ว่ากุญแจยิ่งมีความยาวมาก โอกาสที่ผู้บุกรุกจะสามารถคาดเดากุญแจที่ตรงกับหมายเลขที่ถูกต้องของกุญแจจะยิ่งยากมากขึ้นตามลำดับ ในการที่ผู้บุกรุกลองผิดลองถูกกับกุญแจโดยใช้กุญแจที่มีหมายเลขต่างๆ กัน เพื่อหวังที่จะพบกุญแจที่ถูกต้องและสามารถใช้ถอดรหัสข้อมูลได้ การลองผิดลองถูกนี้เราเรียกกันว่าคีย์เสิร์ช (Key search)หรือการค้นหากุญแจนั่นเอง ทฤษฎีได้กล่าวไว้ว่าการลองผิดลองถูกนี้โดยเฉลี่ยจะต้องทดลองกับกุญแจเป็นจำนวนครึ่งหนึ่งของกุญแจทั้งหมดก่อนที่จะพบกุญแจที่ถูกต้อง

ความยาวของกุญแจที่มีขนาดเหมาะสมจึงขึ้นอยู่กับความเร็วในการค้นหากุญแจของผู้บุกรุกและระยะเวลาที่ต้องการให้ข้อมูลมีความปลอดภัย ตัวอย่างเช่น ถ้าผู้บุกรุกสามารถลองผิดลองถูกกับกุญแจเป็นจำนวน 10 กุญแจภายในหนึ่งวินาทีแล้ว กุญแจที่มีความยาว 40 บิต จะสามารถป้องกันข้อมูลไว้ได้ 3,484 ปี ถ้าผู้บุกรุกสามารถลองได้เป็นจำนวน 1 ล้านกุญแจในหนึ่งวินาที เทคโนโลยีปัจจุบันสามารถทำได้กุญแจที่มีความยาว 40 บิตจะสามารถป้องกันข้อมูลไว้ได้เพียง 13 วันเท่านั้น ซึ่งอาจไม่เพียงพอสำหรับในบางลักษณะงานด้วยเทคโนโลยีในปัจจุบันหากผู้บุกรุกสามารถทดลองได้เป็นจำนวน 1,000 ล้านกุญแจในหนึ่งวินาที กุญแจขนาด 128 บิตจะสามารถป้องกันข้อมูลไว้ได้ 1022 ปี ดังนั้นด้วยลักษณะงานทั่วไปกุญแจขนาด 128 บิตจะพอเพียงต่อการรักษาความลับของข้อมูลเอาไว้ได้

**3.5.6 อัลกอริทึมในการเข้ารหัสลับแบบสมมาตร**

อัลกอริทึมสำหรับการเข้ารหัสแบบสมมาตรในปัจจุบันมีเป็นจำนวนมาก ข้างล่างนี้จะนำเสนอเพียงจำนวนหนึ่งซึ่งเป็นอัลกอริทึมที่เป็นที่รู้จักกันดีในวงการของการเข้ารหัสข้อมูล

**3.5.6.1 อัลกอริทึมดีอีเอส (DES)**

DES ย่อมาจาก Data Encryption Standard อัลกอริทึมนี้ได้รับการรับรองโดยรัฐบาลสหรัฐอเมริกาในปี ค .ศ.1977 ให้เป็นมาตรฐานการเข้ารหัสข้อมูลสำหรับหน่วยงานของรัฐทั้งหมด ในปี 1981 อัลกอริทึมยังได้รับการกำหนดให้เป็นมาตรฐานการเข้ารหัสข้อมูลในระดับนานาชาติตามมาตรฐาน ANSI (American National Standards) อีกด้วย

DES เป็นอัลกอริทึมแบบบล็อกซึ่งใช้กุญแจที่มีขนาดความยาว 56 บิตและเป็นอัลกอริทึมที่มีความแข็งแกร่ง แต่เนื่องด้วยขนาดความยาวของกุญแจที่มีขนาดเพียง 56 บิต ซึ่งในปัจจุบันถือได้ว่าสั้นเกินไป ผู้บุกรุก

อาจใช้วิธีการลองผิดลองถูกเพื่อค้นหากุญแจที่ถูกต้องสำหรับการถอดรหัสได้

ในปี 1998 ได้มีการสร้างเครื่องคอมพิวเตอร์พิเศษขึ้นมาซึ่งมีมูลค่า 250,000 เหรียญสหรัฐ เพื่อใช้ในการค้นหากุญแจที่ถูกต้องของการเข้ารหัสข้อมูลหนึ่งๆ ด้วย DES และพบว่าเครื่องคอมพิวเตอร์นี้สามารถค้นหากุญแจที่ถูกต้องได้ภายในระยะเวลาไม่ถึงหนึ่งวัน

**2.5.6.2 อัลกอริทึมทริเปิลดีอีเอส (Triple-DES)**

นับตั้งแต่เริ่มมีการประยุกต์ใช้งานอัลกอริทึมดีอีเอส เป็นมาตรฐานในการเข้ารหัสลับข้อมูลได้มีผู้ให้ความสนใจถึงขีดความสามารถของ ดีอีเอส ว่ามีความปลอดภัยเพียงพอสำหรับงานที่ต้องการความปลอดภัยสูงหรือสามารถทนทานต่อการโจมตีในรูปแบบต่างๆได้มากน้อยเพียงใด ประเด็นของ DES ที่มีการกล่าวถึงกันมากที่สุดคือในเรื่องขนาดกุญแจที่ใช้ เพราะดีอีเอส กำหนดให้ใช้กุญแจที่มีเพียง 56 บิตซึ่งทำให้จำนวนกุญแจทั้งหมดที่เป็นไปได้คือ $2^{56}$ หรือ $7.2 \times 10^{16}$ ดังนั้นคำถามจึงมีอยู่ว่าขนาดกุญแจขนาด 56 บิตนี้สามารถต้านทานการโจมตีแบบตะลุยหรือการค้นหากุญแจทุกค่าได้หรือไม่

ในปี 1977 Diffie Hellman คาดการณ์ว่ามีความเป็นไปได้ที่จะสร้างคอมพิวเตอร์แบบขนานที่สามารถค้นหากุญแจทั้งหมดได้ในเวลา 10 ชั่วโมงซึ่งต้องใช้งบประมาณสร้างประมาณ 20 ล้านเหรียญสหรัฐดังนั้นขนาดกุญแจ 56 บิตไม่จัดว่าปลอดภัยสำหรับงานที่ต้องการความปลอดภัยสูง ในปีค.ศ. 1997 คำทำนายนี้ก็กลายเป็นความจริง เมื่อมีนักวิจัยใช้เครื่องคอมพิวเตอร์แบบขนานประมาณ 3500 เครื่องทำงานแบบขนาน เพื่อคุ้นหากุญแจ ดีอีเอส โดยใช้เวลาทั้งสิ้นประมาณ 4 เดือนและต่อมาในปีค.ศ. 1998 Electronic Frotier Foundation (EFF) ได้สร้างเครื่องเจาะรหัส ดีอีเอส (ดีอีเอส Cracker) สำหรับค้นหากุญแจของ ดีอีเอส โดยเฉพาะ โดยใช้งบประมาณในการสร้างประมาณ 250,000 เหรียญสหรัฐซึ่งสามารถค้นหากุญแจของ ดีอีเอส ได้ภายใน 4 วัน จึงเป็นการพิสูจน์อย่างเห็นได้ชัดว่า กุญแจขนาด 56 บิตของ ดีอีเอ ไม่ปลอดภัยต่องานที่ต้องการใช้ความปลอดภัยสูง

อีกประเด็นหนึ่งที่เกี่ยวข้องกับการรักษาความปลอดภัยของ ดีอีเอส คืออัลกอริทึมของ ดีอีเอส เองซึ่งมีผู้ตั้งข้อสงสัยว่าอัลกอริทึมของ ดีอีเอส มีจุดอ่อนหรือจุดบกพร่องซึ่งสามารถโดนโจมตีได้ง่ายด้วยวิธีการวิเคราะห์รหัสลับหรือไม่ จึงมีผู้พยายามศึกษาถึงวิธีการทำงานของอัลกอริทึมโดยละเอียดและค้นพบว่าอัลกอริทึมของ ดีอีเอส นั้นยังไม่มีจุดบกพร่องที่รุนแรงเพียงพอให้ผู้โจมตีสามารถใช้ในการวิเราะห์รหัสลับดีอีเอส ได้ง่ายขึ้นดังนั้นจึงสามารถสรุปได้ว่าอัลกอริทึมของ ดีอีเอส จัดได้ว่ามีความปลอดภัยพอสมควรจากการโจมตีด้วยวิธีวิเคราห์รหัสลับ

จากที่กล่าวมาจะเห็นว่า อัลกอริทึม ดีอีเอส ถูกพิจารณาว่าไม่สามารถให้ความปลอดภัยที่เพียงพอสำหรับงานที่ต้องการความปลอดภัยสูงในยุคปัจจุบัน ดังนั้นเพื่อแก้ไขข้อจำกัดของดีอีเอส จึงได้มีการศึกษาถึงทางเลือก 2 ทาง คือ

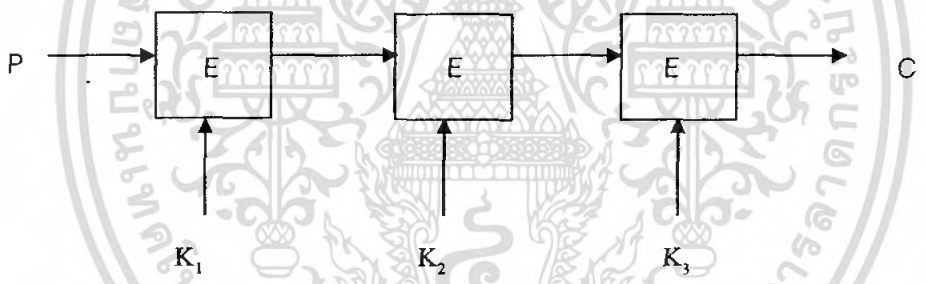
- พัฒนาอัลกอริทึมใหม่ที่มีความปลอดภัยมากกว่า ดีอีเอส เช่น มีขนาดกุญแจที่ใหญ่ขึ้นหรือทำอัลกอริทึมที่มีความปลอดภัยสูงขึ้น ตัวอย่างวิธีนี้ได้แก่ อัลกอริทึมเออีเอส (AES)

- พัฒนาวิธีการใช้งานอัลกอริทึมดีอีเอส ให้มีความแข็งแกร่งมากขึ้นด้วยการนำมาใช้เข้ารหัสลับมากกว่า 1 ครั้ง วิธีนี้ไม่ต้องพัฒนาอัลกอริทึมขึ้นมาใหม่ เพราะยังคงใช้อัลกอริทึมดีอีเอส เดิม ข้อดีคือประหยัดค่าใช้จ่าย เนื่องจาก อัลกอริทึมดีอีเอส มีการใช้งานมานานมากและมีการลงทุนเพื่อพัฒนาอาร์ดแวร์และซอฟต์แวร์สำหรับดีอีเอส จำนวนมาก ดังนั้นเราสามารถใช้ฮาร์ดแวร์และซอฟต์แวร์ของดีอีเอส ที่มีอยู่แล้วให้เกิดประโยชน์ได้

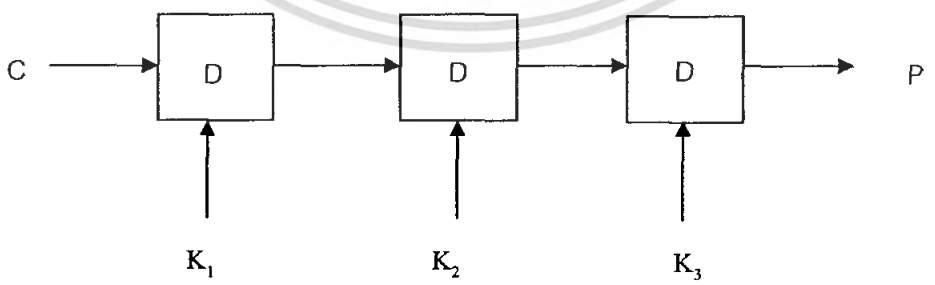
การเข้ารหัสอัลกอริทึมดีอีเอส มากกว่า 1 ครั้งสามารถทำได้สองวิธีคือดีอีเอส 2 ครั้งดับเบิ้ลดีอีเอส (Double DES) หรือดีอีเอส 3 ครั้งทริเปิลดีอีเอส (Triple DES) ซึ่งจะกล่าวถึงต่อไป

**3.5.6.2.1 การเข้ารหัสลับโดยใช้ดีอีเอส 3 ครั้งด้วยกุญแจ 3 ค่า**

การเข้ารหัสโดยใช้ดีอีเอส 3 ครั้งวิธีนี้ จะทำโดยนำข้อความต้นฉบับ P ผ่านเข้าอัลกอริทึมเข้ารหัสดีอีเอส 3 ครั้ง โดยแต่ละครั้งใช้ค่ากุญแจที่ต่างกัน คือ $k_1$, $k_2$, $k_3$ ดังรูป




**รูปที่ 3.22** การเข้ารหัสลับ




**รูปที่ 3.23** การถอดหัสลับ

เราสามารถแสดงนิยามทางคณิตศาสตร์ได้ว่า

$$C= E_{k3}[E_{k2}[E_{k1}[P]]]$$

ส่วนการถอดรหัสก็จะทำย้อนกลับ กล่าวคือจะถอดรหัสด้วยค่ากุญแจ $k_3, k_2, k_1$ ตามลำดับซึ่งสามารถแสดงนิยามทางคณิตศาสตร์ได้ว่า

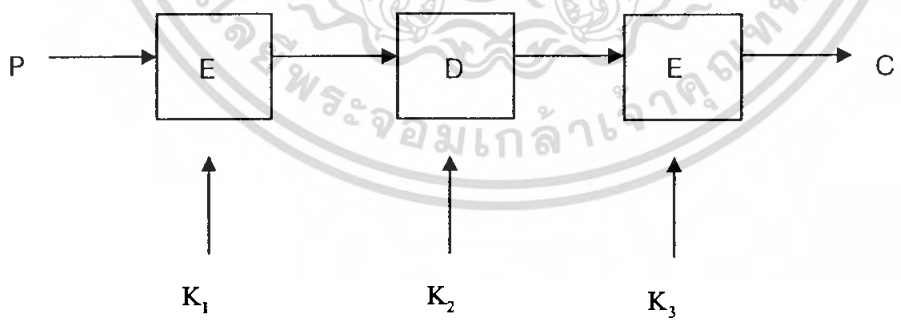
$$P= D_{k3}[D_{k2}[D_{k1}[C]]]$$

**3.5.6.2.1 การเข้ารหัสลับโดยใช้ดีอีเอส 3 ครั้งด้วยกุญแจ 2 ค่า**

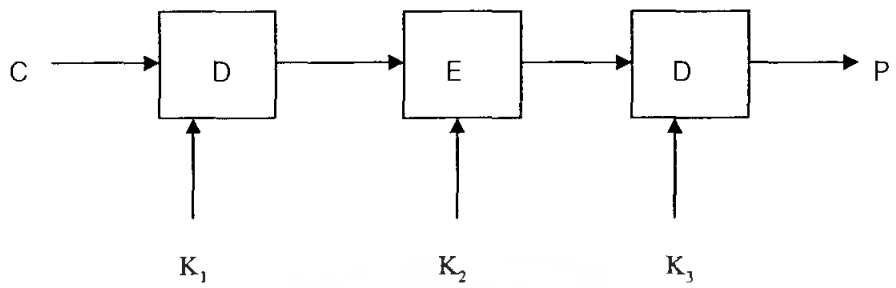
การใช้ดีอีเอส 3 ครั้งด้วยกุญแจ 3 ค่านี้จัดว่าเป็นการเข้ารหัสที่มีความปลอดภัยมากที่สุดวิธีหนึ่งในปัจจุบันเนื่องจากต้องใช้กุญแจ 3 ค่าดังนั้น ขนาดของกุญแจคือ 56 x 3 = 168 บิตซึ่งทำให้การโจมตีแบบตะลุยโดยการค้นหาค่ากุญแจทั้งหมดไม่สามารถทำได้ในเทคโนโลยีปัจจุบัน แต่การใช้กุญแจถึง 3 ค่าในการเข้ารหัสและถอดรหัสนี้ ทำให้เกิดความไม่สะดวกในการใช้งาน จึงมีผู้เสนอรูปแบบการเข้ารหัสโดยใช้ดีอีเอส 3 ครั้งด้วยกุญแจ 2 ค่าแทนซึ่งทำให้ขนาดของกุญแจเท่ากับ 56 x 2 = 112 บิตซึ่งจำนวนบิตเท่ากับการใช้ดีอีเอส 2 ครั้งแต่มีความปลอดภัยสูงกว่า

การเข้ารหัสดีอีเอส 3 ครั้งด้วยกุญแจ 2 ค่ามีขั้นตอนดังนี้

- นำข้อความต้นฉบับ P มาเข้ารหัสดีอีเอส ครั้งแรกโดยใช้กุญแจ $K_1$ ทำ $E_{k1}[P]$
- นำข้อความไซเฟอร์ที่มาจากขั้นแรก ไปถอดรหัสด้วยกุญแจ $K_2$ ทำให้ได้ $D_{k2}[E_{k1}[P]]$
- นำข้อความไซเฟอร์ที่มาจากขั้นที่สอง เข้ารหัสด้วยกุญแจ $K_1$ เป็นครั้งที่ 3 ทำให้ได้ข้อความไซเฟอร์ C




**รูปที่ 3.24** การเข้ารหัสลับ




**รูปที่ 3.25** การถอดหัสลับ

เราสามารถแสดงนิยามทางคณิตศาสตร์ได้ว่า

$$C = E_{k3}[D_{k2}[E_{k1}[P]]]$$

การถอดรหัสดีอีเอส 3 ครั้งด้วยกุญแจ 2 ค่ามีขั้นตอนดังนี้

- นำข้อความไซเฟอร์ P มาเข้ารหัสดีอีเอส ครั้งแรกโดยใช้กุญแจ $K_1$ ทำ $D_{k1}[C]$
- นำข้อความผลลัพธ์ที่มาจากขั้นแรก ไปถอดรหัสด้วยกุญแจ $K_2$ ทำให้ได้ $E_{k2}[D_{k1}[C]]$
- นำข้อความผลลัพธ์ที่มาจากขั้นที่สอง เข้ารหัสด้วยกุญแจ $K_1$ เป็นครั้งที่ 3 ทำให้ได้ข้อความไซเฟอร์ C

ดังแสดงในรูปข้างล่าง ทำให้ได้ข้อความต้นฉบับ P ออกมาตามเดิม

เราสามารถแสดงนิยามทางคณิตศาสตร์ได้ว่า

$$P = D_{k3}[E_{k2}[D_{k1}[P]]]$$

การเข้ารหัสดีอีเอส 3 ครั้งด้วยกุญแจ 2 ค่า แบบนี้เรียกว่าการเข้ารหัสแบบ เข้ารหัสลับ ถอดรหัสลับ เข้ารหัสลับ (encrypt-decrypt-encrypt)หรือที่เรียกสั้นๆว่า อีดีอี (EDE) ส่วนการเข้ารหัสโดยใช้ ดีอีเอส 3 ครั้งด้วยกุญแจ 3 ค่าเรียกว่า การทำงานแบบ เข้ารหัสลับ เข้ารหัสลับ เข้ารหัสลับ (encrypt- encrypt -encrypt) หรือที่เรียกสั้นๆว่าอีอีอี(EEE) จุดประสงค์ของการทำงานแบบอีดีอี คือระบบนี้สามารถนำไปใช้กับ ดีอีเอสแบบดั้งเดิมซึ่งใช้ค่ากุญแจเดียว เพราะถ้าเรากำกนดว่ากุญแจ $K_1 = K_2$ ในนิยามทางคณิตศาสตร์จะได้

$$C = E_{k3}[D_{k2}[E_{k1}[P]]] = E_{k1}[P]$$

ซึ่ง $C = E_{k1}[P]$ คือการเข้ารหัสแบบปกตินั่นเอง ดังนั้นการทำงานแบบอีดีเอส จึงมีจุดประสงค์เพื่อสามารถนำไปใช้งานกับระบบดีอีเอส เดิมที่มีอยู่แล้ว ทำให้การเข้ารหัสโดยวิธีนี้เป็นที่นิยมแพร่หลายในปัจจุบัน

# บทที่ 4
# การทดลองและผลการทดลอง

## 4.1 การเขียนไลบรารีติดต่อเว็บแคม

ในการทดลองเขียนไลบรารีเพื่อใช้ในการติดต่อเว็บแคม เราจะใช้เทคโนโลยีของ AVICAP32 ซึ่งเป็นรูปแบบหนึ่งที่ใช้ในการบันทึกภาพมัลติมิเดียถูกคิดค้นขึ้นโดยบริษัทไมโครซอฟต์โดยมีขั้นตอนในการเขียนดังต่อไปนี้

### 4.1.1 สร้างโปรเจควินโดวส์คอนโทรลไลบรารี (Windows Control Library)

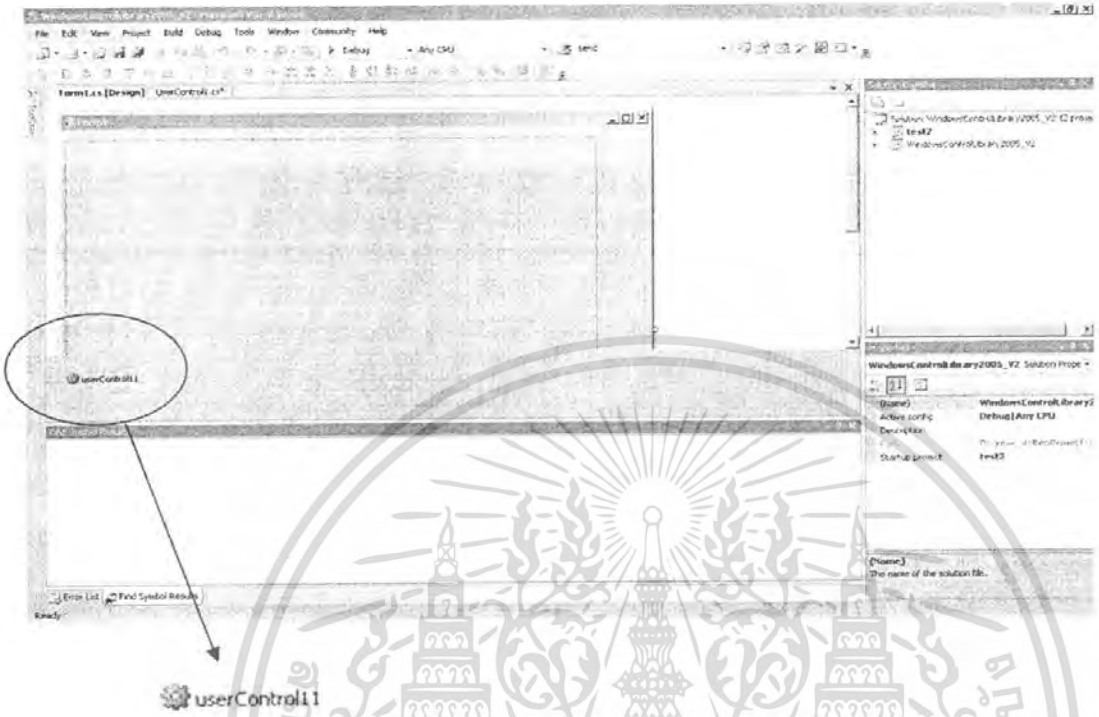
ทำการสร้างไลบรารีในการติดต่อกับเว็บแคมขึ้นมาโดยใช้โปรแกรม Microsoft Visual Studio ในการพัฒนาและใช้รูปแบบโปรเจคประเภทวินโดวส์คอนโทรลไลบรารี เพื่อสร้างไลบรารีที่มีลักษณะเป็นคอนโทรลสำหรับถูกเรียกใช้โดยโปรเจคอื่นๆ รูปด้านล่างเป็นลักษณะการใช้งานของคอนโทรลไลบรารีดังกล่าว

ในโครงงานนี้จะสร้างคอนโทรลที่มีลักษณะไม่ปรากฏบนพาเรนท์วินโดวส์ ใดๆ เพื่อ ให้ลดข้อจำกัดในการใช้งานคอนโทรลในเรื่อง การขึ้นอยู่กับคอนโทรลอื่นๆ เช่น ถ้าคอนโทรลอยู่บน ฟอร์มถ้าฟอร์มถูกซ่อน (Hide Method) คอนโทรลอาจมีการทำงานที่ผิดจากวัตถุประสงค์

เมื่อเขียนไลบรารีเสร็จเรียบร้อยแล้ว โปรเจคอื่นสามารถเรียกใช้ได้ดังนี้ที่เมนูบาร์ของ Microsoft Visual Studio

- เลือก Tools เลือก Toolbox Items
- ที่แท็บ .Net Framework Components ให้คลิกปุ่ม Browse... เพื่อทำการเลือกไฟล์ DLL ที่สร้างได้จาก Windows Control Library Project ขึ้นมาก่อน
- เลือก checkbox ไลบรารีนั้น แล้ว Control library ดังกล่าวก็จะปรากฏบริเวณ Toolbox
- drag และdrop ที่ฟอร์มของโปรเจคที่ต้องการเรียกใช้ไลบรารีนี้

Control ดังกล่าวก็จะปรากฏดังรูปด้านล่าง




**รูปที่ 4.1** ลักษณะการใช้งานของคอนโทรลไลบรารี

### 4.1.2 Platform Invoke (P/Invoke) และ API Constants

**Platform Invoke (P/Invoke)**

จากที่ได้กล่าวมาแล้วว่าไลบรารี APICAP32.DLL เป็นเอพีไอที่ไม่ได้ถูกขยายเป็น managed class จึงต้องใช้ PInvoke ซึ่งเป็นกลไกของภาษาดอทเน็ตที่ใช้ในการเรียก unmanaged functions ใน DLLs มีโค้ดดังต่อไปนี้

```
[DllImport("user32", EntryPoint = "SendMessage")]
public static extern int SendMessage(int hWnd, uint Msg, int wParam, int lParam);
```

**รูปที่ 4.2** การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี

จากรูปข้างต้นเป็นการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี user32.dll โดยส่งอาร์กิวเมนต์ไป 4 ค่า มีชนิดเป็น int, uint, int และint ตามลำดับ

```
[DllImport("avicap32.dll", EntryPoint = "capCreateCaptureWindowA")]
public static extern int capCreateCaptureWindowA
(string lpszWindowName, int dwStyle, int X, int Y, int nWidth, int nHeight, int hwndParent, int nID);
```

**รูปที่ 4.3** การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน capCreateCaptureWindowA จากไลบรารี

จากรูปข้างต้นเป็นการเรียกใช้ฟังก์ชัน capCreateCaptureWindowA จากไลบรารี avicap32.dll โดยส่งอาร์กิวเมนต์ไป 8 ค่า มีชนิดเป็น ค่าแรกเป็น string ค่าที่ 2 ถึงค่าที่ 8 เป็น int

ในรูปด้านล่างจะเป็นโค้ดเพิ่มเติมในส่วนของ P/Invoke ที่ใช้ในการทดลอง จะแสดงให้เห็นถึงโอเวอร์โหลดเมทธอด (Overloading Method) ซึ่งเป็นฟีเจอร์ใน object oriented programming languages ที่ยอมให้เกิดการสร้างฟังก์ชันที่มีชื่อเดียวกันแต่มีชนิดของอินพุทและเอาท์พุทที่ต่างกัน

```
[DllImport("User32.dll")]
public static extern bool SendMessage(int hWnd, uint wMsg, int wParam, FrameEventHandler lParam);
```

**รูปที่ 4.4** การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี

```
[DllImport("User32.dll")]
public static extern bool SendMessage(int hWnd, uint wMsg, int wParam, ref BITMAPINFO lParam);
```

**รูปที่ 4.5** การใช้ P/Invoke เพื่อเตรียมการเรียกใช้ฟังก์ชัน SendMessage จากไลบรารี

จะเห็นได้ว่าการใช้กลไก P/Invoke มีข้อจำกัดคือ การที่จะสามารถเรียกใช้ฟังก์ชันใดมาใช้ได้ เราต้องรู้ชื่อไฟล์ DLL ที่มีฟังก์ชันดังกล่าว

**API Constants**

การติดต่อกับเว็บแคมได้ นอกจากต้องอาศัยฟังก์ชันจากไลบรารี APICAP32.DLL แล้ว ยังต้องอาศัยค่าคงที่ด้วยเช่น ค่าคงที่ของแมสเสจดังรูปด้านล่าง

```
public const int WM_USER = 1024;
public const int WM_CAP_START = WM_USER;
public const int WM_CAP_CONNECT = WM_CAP_START+  10;
public const int WM_CAP_DISCONNECT = WM_CAP_START + 11;
public const int WM_CAP_GET_FRAME = WM_CAP_START + 60;
public const int WM_CAP_GET_VIDEOFORMAT = WM_CAP_START + 44;
public const int WM_CAP_SET_VIDEOFORMAT = WM_CAP_START + 45;
public const int WM_CAP_SET_PREVIEW = WM_CAP_START + 50;
public const int WM_CAP_SET_CALLBACK_FRAME = WM_CAP_START + 5;
```

**รูปที่ 4.6** แสดงการประกาศ API Constant

จากรูป จะเป็นการประกาศค่าค่าคงที่ของ (API Constants) โดยค่าที่ประกาศ เป็นค่าของแมสเสจที่ใช้ในการติดต่อกับเว็บแคม โดยที่ค่าเหล่านี้สามารถทราบได้จากไฟล์ Vfw.h ในพาท C:\Program Files\Microsoft-Visual Studio 8\VC\PlatformSDK\Include (จะพบถ้าติดตั้งโปรแกรม Microsoft Visual Studio 2005)

**4.1.3 การเขียนโค้ดเพื่อติดต่อกับเว็บแคม**

**4.1.3.1 เซ็ตค่าแคปเจอร์วินโดวส์ (Setup a capture window)**

การที่จะติดต่อกับเว็บแคมได้ เราต้องทราบ handle ของคอนโทรล (control) ที่ใช้ในการติดต่อ คือ handle ของคอนโทรลนั้นๆเอง ดังรูปด้านล่าง

```
// setup a capture window
mCapHwnd = capCreateCaptureWindowA
          ("WebCap", 0, 0, 0, m_Width, m_Height, this.Handle.ToInt32(), 0);
```

**รูปที่ 4.7** แสดงการติดต่อกับ handle window

ซึ่งค่าที่รีเทิร์นออกมาจะเป็นค่าของ handle ของคอนโทรล

**4.1.3.2 ติดต่อไปยังอุปกรณ์ภาพ (Connect to the capture device)**

ติดต่อกับ Capture driver โดยการส่งแมสเสจไปติดต่อดังรูปด้านล่าง ถ้าค่าที่รีเทิร์นออกมาเป็น true แสดงว่าติดต่อได้ แต่ถ้ารีเทิร์นเป็น false แสดงว่าติดต่อไม่ได้

```
SendMessage(mCapHwnd, WM_CAP_CONNECT, 0, 0);
```

**รูปที่ 4.8** แสดงการติดต่อกับแคปเจอร์ไดรเวอร์

**4.1.3.3 เลือกฟอร์เมต (Sets the format of captured video data)**

เนื่องจากการส่งแมสเสจไปตั้งค่าฟอร์แมตของวิดิโอ ต้องส่งออปเจค ของ BITMAPINFO ไปด้วย ดังนั้นจึงทำการสร้างออปเจคของ Bitmap ก่อนและตั้งค่าคุณสมบัติที่ต้องการ จากนั้นจึงส่งไปเป็นอาร์กิวเมนต์ร่วมกับแมสเสจ WM_CAP_SET_VIDEOFORMAT ในฟังก์ชัน SendMessage()

```
BITMAPINFO bitmapinfo = new BITMAPINFO();
bitmapinfo.bmiHeader.biSize = Marshal.SizeOf(bitmapinfo.bmiHeader);
bitmapinfo.bmiHeader.biWidth = m_Width;
bitmapinfo.bmiHeader.biHeight = m_Height;
bitmapinfo.bmiHeader.biPlanes = 1;
bitmapinfo.bmiHeader.biBitCount = 24;
SendMessage(mCapHwnd, WM_CAP_SET_VIDEOFORMAT, Marshal.SizeOf(bitmapinfo), ref bitmapinfo);
```

**รูปที่ 4.9** แสดงการตั้งค่าวิดิโอฟอร์เมตในการแคปเจอร์

**4.1.3.4 เซ็ตค่าแฟรมคอลแบก callback function**

AVICap จะเรียกฟังก์ชันนี้เมื่อมีการแคปเจอร์เฟรมเกิดขึ้น เราประกาศ frame callback ฟังก์ชั่นได้ดังรูปข้างล่าง

```
this.mFrameEventHandler = new FrameEventHandler(FrameCallBack);
SendMessage(mCapHwnd, WM_CAP_SET_CALLBACK_FRAME, 0, mFrameEventHandler);
```

**รูปที่ 4.10** กำหนดฟังก์ชั่น frame callback

จากนั้นก็ทำการสร้าง Function frame callback

```
private void FrameCallBack(IntPtr lwnd, IntPtr lpVHdr)
{
    VIDEOHDR videoHeader = new VIDEOHDR();
    byte[] VideoData;
    videoHeader = (VIDEOHDR)Marshal.PtrToStructure(lpVHdr, videoHeader.GetType());
    VideoData = new byte[videoHeader.dwBytesUsed];
    Marshal.Copy(new IntPtr(videoHeader.lpData), VideoData, 0, VideoData.Length);
    System.Drawing.Bitmap bmp = new Bitmap(m_Width, m_Height); //edit
    System.Drawing.Imaging.BitmapData bmpData = bmp.LockBits(new Rectangle(0, 0, m_Width, m_Heig
    int startAddr = bmpData.Scan0.ToInt32();
    for (int row = m_Height - 1; row >= 0; row--)
    {
        // ** Woot! Dirty trick, may not work on all platforms.... egrath
        Marshal.Copy(VideoData, (row * m_Width * 3), new IntPtr(startAddr), m_Width * 3);
        startAddr += m_Width * 3;
    }
    bmp.UnlockBits(bmpData);
    if (this.RecievedFrame != null)
    {
        WebCameraEventArgs e = new WebCameraEventArgs(VideoData, bmp);
        this.RecievedFrame(this, e);
    }
```

**รูปที่ 4.11** การทำฟังก์ชั่น frame callback

โค้ดภายในฟังก์ชั่น frame callback จะเป็นโค้ดในการเข้าถึงข้อมูลที่แคปเจอร์ได้และคัดลอกข้อมูลส่วนนั้นออกมาเป็นออปเจคบิตแมป (Object Bitmap) โดยมีขั้นตอนดังนี้

- สร้างออปเจคของ VIDEOHDR ขึ้นมาแล้วจึง marshal ข้อมูลจากหน่วยความจำในส่วนที่เป็น unmanaged block ซึ่งก็คือ lpVHdr ไปยัง managed object ซึ่งคือ videoHeader
- คัดลอกข้อมูลจาก unmanaged memory pointer ไปยัง managed 8-bit unsigned integer array
- สร้างออปเจกต์ของบิตแมปคัดลอกข้อมูลจากอาเรย์ไปเป็นบิตแมปโดยต้องคำนึงถึงโครงสร้างของบิตแมปเองด้วย

**4.1.4 จัดทำฟังก์ชันที่เป็นอินเตอร์เฟสสำหรับผู้ใช้**

จัดทำฟังก์ชั่นต่างๆเพื่อให้ผู้ใช้สามารถติดต่อกล้องผ่านทางอินเทอร์เฟสต่างๆ เช่น

-public void Start(ulong FrameNum)

สำหรับเริ่มการทำงานของเว็บแคม

-public void Stop()

สำหรับหยุดการทำงานของเว็บแคม

-public int CaptureWidth

กำหนดค่าความกว้างของภาพที่จะแคปเจอร์

-public int CaptureHeight

กำหนดค่าความยาวของภาพที่จะแคปเจอร์

-public int TimeToCapture_milliseconds

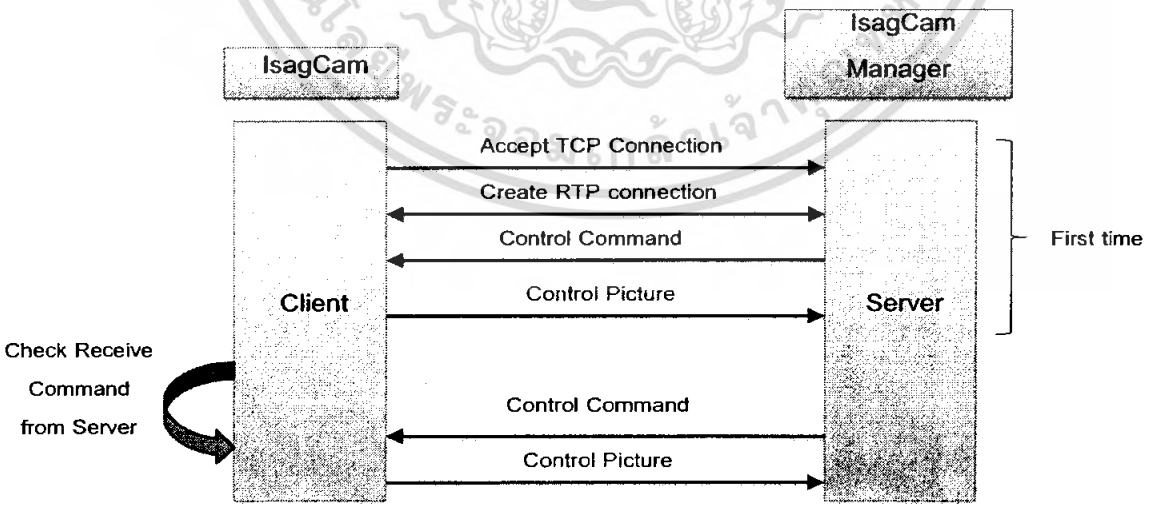
กำหนดช่วงเวลาในการแคปเจอร์

**4.1.5 ตรวจสอบการทำงานของไลบรารี**

ทำการสร้างอีกโปรเจคขึ้นมาทดสอบการทำงานของไลบรารี เพื่อดูว่าได้ผลตรงตามที่ต้องการหรือไม่

## 4.2 การเขียนโปรแกรมส่งข้อมูลคำสั่งโดยใช้ทีซีพีสตรีมซ็อกเกต

จากภาพด้านล่างแสดงการออกแบบของโปรแกรม IsagCam และ IsagCam Manager ซึ่งจะมีรายละเอียดดังนี้




**รูปที่ 4.12** ออกแบบของโปรแกรม IsagCam และ IsagCam Manager

- จะเริ่มต้นด้วยการเขียนโปรแกรมเพื่อติดต่อกันระหว่าง IsagCam และ IsagCam Manager สามารถทำได้โดยเริ่มต้นการติดต่อสื่อสารโดยโปรโตคอลทีซีพี ซึ่งฝั่งเซิร์ฟเวอร์เมื่อกดปุ่มทำงานจะทำการเริ่มสถานะ listen เพื่อรอการเชื่อมต่อและเมื่อไคล์แอนต์เปิดโปรแกรมจะทำการ accept และเริ่มต้นการส่งข้อมูลคอนโทรลอันแรก

- จากนั้นฝั่งไคล์แอนต์จะรับข้อมูลไปแปลคำสั่งและประมวลผลส่งกลับมาให้เซิร์ฟเวอร์ซึ่งต่อจากนั้นฝั่งไคล์แอนต์จะใช้ timer ในการวนตรวจว่าเซิร์ฟเวอร์ได้ส่งข้อมูลใดๆมาหรือไม่ ซึ่งถ้าพบก็จะทำการแปลคำสั่งและประมวลผลไปเรื่อยๆ

**4.2.1 การเขียนโปรแกรมส่งข้อมูลคำสั่งโดยใช้ทีซีพีสตรีมซ็อกเก็ต (TCP Stream Socket) ฝั่งเซิร์ฟเวอร์**

- กำหนดเนมสเปสที่เกี่ยวข้อง

```
using System;
using System.Net.Sockets;
using System.Net;
using System.Text;
```

**รูปที่ 4.13** โค้ดการใส่ค่า name space ที่เกี่ยวข้องกับทีซีพี

- นำเอาไอพีแอดเดรส (IP Address) ที่ของฝั่งเซิร์ฟเวอร์ที่ได้มาสร้างเป็นเอนด์พอยท์ (End Point) สำหรับใช้งานโดยการกำหนดพอร์ตที่จะใช้ติดต่อเพิ่มเข้าไป ในการกำหนดเอนด์พอยท์นั้นจะใช้วิธีสร้างออปเจคชนิด IPEndPoint ขึ้นมาแล้วกำหนดค่าไอพีแอดเดรสและหมายเลขพอร์ตให้กับออปเจ็กต์ชนิด IPEndPoint ที่สร้างขึ้นมานั้น

```
IPEndPoint ipEndPoint = new IPEndPoint("192.168.1.2", 11000);
```

**รูปที่ 4.14** โค้ดการนำเอาไอพีแอดเดรสที่ของฝั่งเซิรฟ์เวอร์ที่ได้มาสร้างเป็นเอนด์พอยท์

- การสร้างซ็อกเกตขึ้นมาใช้งานโดยการสร้างอินสแตนท์ (Instance) ของคลาสซ็อกเกต (Socket class) ดังนี้

```
Socket sListener = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
```

**รูปที่ 4.15** โค้ดการสร้างซ็อกเกต

- ในการติดต่อเป็นการกำหนดเอนด์พอยท์ให้กับซ็อกเกตที่สร้างขึ้น (Bind) และสั่งเปิดพอร์ตเพื่อรอรับการเชื่อมต่อ (Listening) การกำหนดเอนด์พอยท์ให้กับซ็อกเกตที่สร้างขึ้นเกิดขึ้นเมื่อใช้เมทธอด Bind () ดังนี้

```
sListener.Bind(ipEndPoint);
```

**รูปที่ 4.16** โค้ดการกำหนดเอนด์พอยท์ให้กับซ็อกเกตที่สร้างขึ้น

```
sListener.Listen(10);
```

**รูปที่ 4.17** โค้ดการ Listening พอร์ตเพื่อรอรับการเชื่อมต่อ

โดยพารามิเตอร์ในวงเล็บหลังเมทธอด Listen นั้นจะเป็นตัวบ่งบอกว่าซ็อกเกตนี้จะรองรับการเชื่อมต่อได้ทั้งหมดกี่คอนเนคชั่น เป็นการรอรับการเชื่อมต่อจากไคล์แอนต์โดยในการรอการเชื่อมต่อนี้จะใช้เมทธอด Accept () ซึ่งเมทธอดนี้จะทำการบล็อกเทรดซึ่งเรียกใช้งานให้หยุดรอจนกว่าจะมีไคล์แอนต์เชื่อมต่อเข้ามา

```
Socket handler = sListener.Accept();
```

**รูปที่ 4.18** โค้ดการ Accept

- เมื่อการเชื่อมต่อต่าง ๆ พร้อมที่จะใช้งานแล้วก็จะทำการสื่อสารข้อมูล (Send/Receive) บนช่องทางสื่อสารที่ทำการสร้างขึ้น โดยในโปรแกรมตัวอย่างจะเป็นการรับข้อมูลจากไคล์แอนต์โดยการใช้เมทธอด Receive() ดังนี้

```
int bytesRec = handler.Receive(bytes)
```

**รูปที่ 4.19** โค้ดการรับข้อมูลจากไคล์แอนต์

โดยที่พารามิเตอร์ของเมทธอด Receive() จะเป็นตัวแปรซึ่งจะนำค่าที่ได้รับไปเก็บไว้ แล้วเมทธอดนี้จะส่งค่ากลับเป็นปริมาณของข้อมูลซึ่งได้รับมา

- เมื่อการสื่อสารระหว่างไคล์แอนต์และเซิร์ฟเวอร์สิ้นสุดลง ก็ต้องทำการปิดการเชื่อมต่อแล้วคืนทรัพยากรให้แก่ระบบ โดยเมทธอดที่ใช้เพื่อจบการเชื่อมต่อจะมีอยู่ด้วยกัน 2 เมทธอดคือ Shutdown และ Close และมีวิธีการใช้ดังนี้

```
handler.Shutdown(SocketShutdown.Both);
handler.Close();
```

**รูปที่ 4.20** โค้ดการปิดการเชื่อมต่อ

**4.2.2 การเขียนโปรแกรมส่งข้อมูลคำสั่งโดยใช้ทีซีพีสตรีมซ็อกเกตฝั่งไคล์แอนต์**

การทำงานจะคล้ายฝั่งเซิร์ฟเวอร์แต่จะมีความแตกต่างกันในข้อที่ 4 ที่ฝั่งไคล์แอนต์จะเป็นการสร้างการเชื่อมต่อไปยังเอนด์พอยท์ที่กำหนด โดยใช้เมทธอด Connect() ดังนี้

```
sender.Connect(ipEndPoint);
```

**รูปที่ 4.21** โค้ดการสร้างการเชื่อมต่อไปยังเอนด์พอยท์ที่กำหนด

เป็นการสร้างช่องทางการเชื่อมต่อระหว่างโปรแกรมไคล์แอนต์ และโปรแกรมเซิร์ฟเวอร์เมื่อการเชื่อมต่อเกิดขึ้นโดยสมบูรณ์ ก็สามารถใช้คำสั่งสำหรับ รับ/ส่ง ข้อมูลได้

## 4.3 การเขียนโปรแกรมส่งข้อมูลภาพด้วยอาร์ทีพี

การส่งข้อมูลภาพโดยใช้โปรโตคอลอาร์ทีพีซึ่งจะมีการสร้างโปรแกรมทั้ง 2 ฝั่งโดยการสร้างเซสชั่นสำหรับการติดต่อสื่อสารและทำการส่งข้อมูล ในขณะเดียวกันเมื่อต้องการปิดการเชื่อมต่อสื่อสารเซสชั่นดังกล่าวจะต้องสามารถปิดการเชื่อมต่อตัวเองลงได้ด้วย

จากการทดลองเขียนโปรแกรมส่งข้อมูลภาพ โดยเริ่มจะมีหลักการทำงานคร่าวๆดังต่อไปนี้

- Hook RTP events ส่วนนี้จะเป็นส่วนเริ่มแรกสุดคือการระบุอีเวนท์ที่เกี่ยวข้องกับการส่งข้อมูลผ่านสตรีมซึ่งอีเวนท์ต่างๆที่ใช้นั้นประกอบด้วย

```
RtpEvents.RtpParticipantAdded += new
RtpEvents.RtpParticipantAddedEventHandler(RtpParticipantAdded);
RtpEvents.RtpParticipantRemoved += new
RtpEvents.RtpParticipantRemovedEventHandler(RtpParticipantRemoved);
```

**รูปที่ 4.22** แสดงอีเวนท์ที่จัดการการเข้าร่วมเซสชั่นเช่น การ เพิ่ม/ลด ยูสเซอร์จากเซสชั่นของอาร์ทีพี

```
RtpEvents.RtpStreamAdded += new
RtpEvents.RtpStreamAddedEventHandler(RtpStreamAdded);
RtpEvents.RtpStreamRemoved += new
RtpEvents.RtpStreamRemovedEventHandler(RtpStreamRemoved);
```

**รูปที่ 4.23** แสดงอีเวนท์ที่จัดการ เพิ่ม/ลด เซสชั่นเช่น การแอคติเวทอาร์ทีพีหลายๆเซสชั่น

- เข้าร่วมอาร์ทีพีเซสชั่น
- ระบุชนิดของแพ็กเกตอาร์ทีพีจะใช้ในการส่งโดยแต่ละเซสชั่นจะสามารถระบุชนิดของตัวข้อมูล (payload) ได้ 1 ชนิด ในที่นี้ระบุเป็น chat
- เริ่มทำการส่ง-รับ
- Unhook RTP events ส่วนนี้จะเป็นส่วนท้ายสุดคือการลบอีเวนท์ที่เกี่ยวข้องกับการส่งข้อมูลผ่านสตรีมซึ่งอีเวนท์ต่างๆที่ใช้นั้นจะถูกลบดังนี้

```
RtpEvents.RtpParticipantAdded -= new
RtpEvents.RtpParticipantAddedEventHandler(RtpParticipantAdded);
            RtpEvents.RtpParticipantRemoved -= new
RtpEvents.RtpParticipantRemovedEventHandler(RtpParticipantRemoved);
            RtpEvents.RtpStreamAdded -= new
RtpEvents.RtpStreamAddedEventHandler(RtpStreamAdded);
            RtpEvents.RtpStreamRemoved -= new
RtpEvents.RtpStreamRemovedEventHandler(RtpStreamRemoved);
```

**รูปที่ 4.24** แสดง Unhook อีเวนท์

ทำการลบเซสชั่นออกโดยการเคลียร์ค่าเซสชั่นของอาร์ทีพีและผู้ส่ง (Sender) ให้มีค่าเป็น null

## 4.4 การออกแบบคอนโทรลไคล์แอนต์ด้วยโปรโตคอลทีซีพีให้ทำตามเงื่อนไขที่หลากหลายจากผู้ใช้งาน


**รูปที่ 4.25** แสดงฟอร์แมตของคอนโทรล

เมื่อทำการเชื่อมต่อคอนเนคชั่นของฝั่งไคล์แอนต์และฝั่งเซิร์ฟเวอร์ได้ด้วยโปรโตคอลทีซีพีและอาร์ทีพีแล้วในส่วนของการคอนโทรลข้อมูลเมื่อผู้ใช้งานฝั่งเซิร์ฟเวอร์จะมีการรับคำสั่งจากอินเทอร์เฟตของโปรแกรม

IsagCam Managerโปรแกรมจะทำการสร้างคำสั่งคอนโทรลและส่งข้อมูลคำสั่งคอนโทรลไปยังฝั่งไคล์แอนต์ ซึ่งรูปแบบของคำสั่ง ได้ออกแบบและพัฒนาเอง โดยมีรูปแบบดังต่อไปนี้

**ฟิลล์ที่ 0** ชนิดของข้อมูลแบ่งออกเป็น 2 แบบคือ

- Control packet ดูแลจัดการส่วนคอนโทรลในรูปแบบต่างๆที่ผู้ใช้ต้องการทั้งหมดโดยเป็นค่าคอนฟิคจากอินเทอร์เฟสของ IsagCam Manager
- Stop packet เมื่อกล้องเว็บแคมหยุดการทำงาน

**ฟิลล์ที่ 1** Schedule/Realtime ดูแลรูปแบบของการส่งให้เป็นแบบตั้งเวลาได้ หรือเริ่มต้นทำงานทันที

**ฟิลล์ที่ 2-3** Fromtime/Totime เวลาเริ่มต้นและสิ้นสุดกรณีตั้งเวลา (จัดให้อยู่ในรูป Datetime)

**ฟิลล์ที่ 4** Capturing/Streaming ดูแลรูปแบบของการส่งให้เป็นแบบการถ่ายภาพตามช่วงเวลาหรือเป็นแบบข้อมูลสตรีม

**ฟิลล์ที่ 5** Time interval ดูแลจัดการส่วนของเวลาที่จะใช้ในการถ่ายภาพทุกๆ 5, 15, 30 วินาที

**ฟิลล์ที่ 6** Color/Grayscale ดูแลจัดการเปลี่ยนภาพสีธรรมชาติให้เป็นภาพขาวดำ

คำสั่งที่ทำการรวมนั้นจะแยกแต่ละฟิลล์ด้วยเครื่องหมาย ","และส่งไปยังฝั่งเซิร์ฟเวอร์ผ่านทางโปรโตคอลทีซีพี

## 4.5 การเขียนโปรแกรมการจัดการกับดิสก์

การจัดการกับดิสก์ในโปรแกรมฝั่งเซิร์ฟเวอร์หรือ IsagCam Manager จะกระทำเมื่อมีข้อมูลภาพเข้ามาและจะเริ่มต้นทำการบันทึกจะมีการเช็คก่อนว่าดิสก์ที่ใช้ในการเก็บข้อมูลในแต่ละกล้องนั้นยังมีความจุเพียงพอกับขนาดของรูปหรือไม่ ซึ่งถ้าไม่เพียงพอจะมีรูปแบบในการจัดการอยู่ 2 ส่วนด้วยกันคือ

- Overwrite ทำการลบไฟล์ภาพที่เก่าที่สุดออกและทำการบันทึกภาพใหม่ลงดิสก์แทน
- Stop working เมื่อตรวจสอบได้ว่าดิสก์เต็มจะทำการหยุดการทำงาน

### 4.5.1 การเขียนทับไฟล์ (Overwrite)

- การเช็คดิสก์ว่าเหลือน้อยกว่าความจุที่กำหนดหรือไม่ ทำได้โดย

```
DriveInfo drvD = new System.IO.DriveInfo(getdrive);
      long drivespaceavailable =
      long.Parse(drvD.AvailableFreeSpace.ToString());
```

**รูปที่ 4.26** แสดงโค้ดการเช็คดิสก์

getdrive คือชื่อไดร์ฟที่เราใช้บันทึกภาพ เช่น C , D เป็นต้น

- ดูไฟล์ทั้งหมดในดิสก์ที่ใช้ในการบันทึกภาพ (แต่ละกล้องจะมีพาร์ทในการบันทึกภาพที่ต่างกัน)

```
DirectoryInfo dir = new DirectoryInfo(@pathfile);
                    FileInfo[] files = dir.GetFiles("*.*");
```

**รูปที่ 4.27** แสดงโค้ดเพื่อดูไฟล์ทั้งหมดในดิสก์

จะได้ไฟล์ทึกประเภทที่อยู่ในพาร์ทที่เราต้องการ

- ทำการจัดเรียงไฟล์ (Sort file) เพื่อเรียงตามลำดับเวลาก่อน-หลังของการบันทึกภาพโดยใช้ฟังก์ชั่น Sort ซึ่งเป็นฟังก์ชั่นที่มีอยู่แล้ว

```
Array.Sort<FileInfo>(files, delegate(FileInfo a,
FileInfo b) { return
a.LastWriteTime.CompareTo(b.LastWriteTime); });
```

**รูปที่ 4.28** แสดงโค้ดเพื่อทำการจัดเรียงไฟล์

- ทำการวนลูปเพื่อเริ่มต้นการลบไฟล์โดยใช้คำสั่ง Delete ไฟล์ในส่วนนี้โปรแกรมได้ออกแบบไว้ให้ลบทีละ 10 เปอร์เซนต์เพื่อจะได้ไม่เสียเวลาในการเข้ามาทำการเช็คและเรียงลำดับไฟล์ข้อมูลหลายๆครั้ง

```
files[0].Delete();
```

**รูปที่ 4.29** แสดงโค้ดเพื่อทำการลบไฟล์

### 4.5.2 หยุดการทำงาน (Stop working)

การทำงานคล้ายกับการทำ Overwrite เพียงแต่เมื่อพบว่าดิสก์เหลือน้อยกว่าพื้นที่ที่กำหนดจะหยุดการทำงานโดยให้กล้องเลิกบันทึกภาพและส่งเมสเซจเพื่อเตือนให้ผู้ใช้ทราบเพื่อทำการเปลี่ยนดิสก์

## 4.6 การเขียนโปรแกรมการบันทึกภาพ

การบันทึกภาพที่รับมาจากฝั่งไคล์แอนต์จะทำงานที่ตัว timer โดยแต่ละกล้องก็จะมี timer เป็นของตัวเองซึ่งจะทำการบันทึกภาพใน events Tick ซึ่งตัวโปรแกรมจะสามารถเริ่มทำงานส่วนการบันทึกได้ก็ต่อเมื่อทำการเช็คดิสก์แล้วพบว่าดิสก์มีพื้นที่ว่างเหลืออยู่ โดยโปรแกรมจะเริ่มเข้าส่วนของการบันทึกด้วยวิธีการดังนี้

### 4.6.1 สร้างฟอร์เมตของชื่อไฟล์ที่จะทำการบันทึก

ในที่นี้การบันทึกชื่อไฟล์จะทำการบันทึกเป็นลักษณะบันทึกตามช่วงเวลา (time format) โดยเพื่อประโยชน์สำหรับการเรียกดูไฟล์ภายหลังจะทำให้ทราบว่าภาพบันทึก ณ เวลาใดๆ โดยฟอร์เมตของเวลาที่จะทำการบันทึกจะจัดรูปแบบดังตัวอย่างด้านล่าง

| วันเดือนปี_ชั่วโมงนาทีวินาที AM/PM + extension (gif, jpeg, png, bmp) |
|---|

**รูปที่ 4.30** แสดงฟอร์เมตของไฟล์ภาพ

### 4.6.2 เริ่มต้นทำการบันทึกภาพ

- คลาสไลบรารีที่ใช้ คือ System โดยใช้ Structure Datetime และกำหนดคุณสมบัติเป็น Now ค่าที่รีเทิร์นได้เป็นค่าออปเจคของ DateTime เป็นวันเวลาปัจจุบันของคอมพิวเตอร์ตามเวลาของ local time
- ทำการบันทึกภาพโดยภาพที่จะทำการบันทึกจะบันทึกเป็นบิตแมปโดยใช้คำสั่ง

| Bitmap.Save("ตำแหน่งเก็บไฟล์") |
|---|

**รูปที่ 4.31** แสดงคำสั่งที่ใช้ในการบันทึกภาพ


**รูปที่ 4.32** แสดงภาพและชื่อไฟล์ภาพหลังจากที่ถูกบันทึกลงหน่วยความจำแล้ว

ไฟล์ภาพที่จัดเก็บเรียบร้อยแล้วและเก็บในรูปแบบของ วันเดือนปี_ชั่วโมงนาทีวินาที AM/PM + extension (gif, jpeg, png, bmp) ในพาร์ทที่เลือกไว้

## 4.7 การจัดการเกี่ยวกับปัญหาเรื่องความเป็นส่วนตัว (Privacy) กระบวนการเข้ารหัสถอดรหัสข้อมูลภาพและข้อมูลคำสั่ง

ในการจัดการเกี่ยวกับเรื่องความเป็นส่วนตัวเพื่อป้องกันการถูกรุกรานจากบุคคลภายนอกเราจะมีการทำในเรื่องของการรักษาความปลอดภัยของข้อมูลภาพและคำสั่ง นั่นคือจะมีการเข้ารหัสของข้อมูลทุกครั้งก่อนทำการส่ง โดยจะใช้ อัลกอริทึมของ Triple DES ซึ่งเหมาะกับงานที่เอามาใช้โดยไม่ต้องคำนึงถึงความปลอดภัยมากนัก

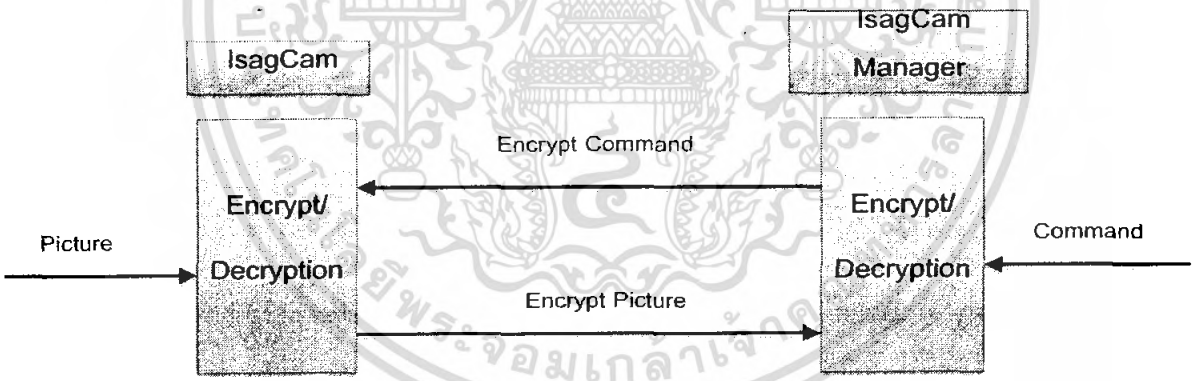
### 4.7.1 การออกแบบกระบวนการเข้ารหัสและถอดรหัสข้อมูล

กระบวนการการถอดความจะแบ่งออกเป็น 2 ส่วนที่สำคัญคือ การเข้ารหัส (encryption) และถอดรหัส (decryption) ซึ่งจะมีอยู่ทั้งฝั่งไคล์แอนต์และเซิร์ฟเวอร์

ที่ฝั่งเซิร์ฟเวอร์จะมีการส่งคำสั่งที่ผ่านการเข้ารหัส เช่นเดียวกันจะรับภาพที่ผ่านการเข้ารหัสมาจากฝั่งไคล์แอนต์และทำการถอดรหัสเพื่อให้ได้ข้อมูลภาพและแสดงผล

ที่ฝั่งไคล์แอนต์จะมีการรับคำสั่งที่ผ่านการเข้ารหัสมาจากเซิร์ฟเวอร์และทำการถอดรหัสเพื่อนำคำสั่งไปประมวลผลเช่นเดียวกันจะส่งภาพที่ผ่านการเข้ารหัสไปให้เซิร์ฟเวอร์

จากผลการออกแบบจะได้ตามภาพตัวอย่างด้านล่าง




**รูปที่ 4.33** การออกแบบโครงสร้างพื้นฐานเกี่ยวกับการเข้ารหัสลับ-ถอดรหัสลับของโปรแกรม

### 4.7.2 ขั้นตอนการเข้ารหัสข้อมูลคำสั่งและภาพ

- เริ่มต้นจะทำการเพิ่มส่วนของเนมสเปสที่เกี่ยวข้องกับการรักษาความปลอดภัย

```
using System.Security;
using System.Security.Cryptography;
using System.Runtime.InteropServices;
```

การทำการเข้ารหัสข้อมูลสตริง

- จะเป็น overload function โดยจะแยกกันระหว่างการเข้ารหัสข้อมูลซึ่งเป็นชนิดสตริง (string) และข้อมูลเสียง ซึ่งเป็นไบท์อาเรย์ (byte array) โดยมีขั้นตอนดังนี้
- นำคีย์ที่ได้จากไฟล์กำหนดค่า (configuration file) มาผ่านกระบวนการเข้ารหัสแบบ MD5 ดูตัวอย่างโค้ดด้านล่าง

```
string key=System.Configuration.ConfigurationManager.AppSettings["Security"];
MD5CryptoServiceProvider hashmd5 = new MD5CryptoServiceProvider();
keyArray = hashmd5.ComputeHash(UTF8Encoding.UTF8.GetBytes(key));
```

**รูปที่ 4.34** การเข้ารหัสคีย์ด้วยอัลกอริทึมแบบเอ็มดีไฟท์ (MD5)

- ทำการเข้ารหัสข้อมูลด้วย Tripple DES ด้วยคลาส TripleDESCryptoServiceProvider

```
tdes.Key = keyArray;
//mode of operation. there are other 4 modes.
//We choose ECB(Electronic code Book)
tdes.Mode = CipherMode.ECB;
//padding mode(if any extra byte added)
tdes.Padding = PaddingMode.PKCS7;

ICryptoTransform cTransform = tdes.CreateEncryptor();
//transform the specified region of bytes array to resultArray
byte[] resultArray =
cTransform.TransformFinalBlock(toEncryptArray, 0,
toEncryptArray.Length);
```

**รูปที่ 4.35** โค้ดการเข้ารหัสข้อมูลด้วยอัลกอริทึม

- การนำไปใช้ให้ทำการเปลี่ยนค่าให้กลับมาอยู่ในรูปแบบของสตริงก่อน

```
Convert.ToBase64String(resultArray, 0, resultArray.Length);
```

**รูปที่ 4.36** โค้ดการเปลี่ยนค่าเข้ารหัสให้กลับมาอยู่ในรูปแบบของสตริงก่อน

การทำถอดรหัสข้อมูลสตริง

- ทำเช่นเดียวกันกับเข้ารหัส เพียงแต่ต่างที่มีการเปลี่ยนข้อมูลที่เข้าเป็นข้อมูลที่ผ่านการเข้ารหัสด้วยวิธีการดังรูปข้างล่างแทนที่จะเป็นสตริงธรรมดาและผ่านกระบวนการเปลี่ยนเป็น byte[]

```
byte[] toEncryptArray = Convert.FromBase64String(cipherString);
```

**รูปที่ 4.37** โค้ดการเปลี่ยนค่าสตริงให้กลับมาอยู่ในรูปแบบของ byte[]

การทำการเข้ารหัสข้อมูลภาพ

- ทำเช่นเดียวกันกับเข้ารหัสข้อมูลคำสั่งเพียงแต่เปลี่ยนข้อมูลเป็น memorystream แทน

```
TripleDES alg = TripleDES.Create();
CryptoStream cs = new CryptoStream(ms,
alg.CreateEncryptor(),CryptoStreamMode.Write);
```

**รูปที่ 4.38** โค้ดการเข้ารหัสข้อมูลภาพ

การทำการถอดรหัสข้อมูลภาพ

- ทำเช่นเดียวกันกับเข้ารหัสข้อมูลภาพ

# บทที่ 5
# บทวิจารณ์และสรุป

## 5.1 บทสรุป

โครงการการสอดส่องความปลอดภัยด้วยเว็บแคม (Surveillance Using Webcam) จะมีโครงสร้างของโปรแกรมที่พัฒนาอยู่ 2 ฝั่งดังต่อไปนี้

1. ทางฝั่งผู้ใช้ไคล์แอนต์ (Client) จะพัฒนาโปรแกรมชื่อว่า IsagCam การทำงานในฝั่งของไคล์แอนต์ (Client) หรือ IsagCam จะมีการทำงานดังต่อไปนี้

- การทำงานของฝั่งไคล์แอนต์ (Client) หรือ IsagCam จะเป็นในลักษณะของสตาทอัพโปรแกรม
- การคอนโทรลส่วนของคำสั่ง ในส่วนนี้จะเป็นส่วนควบคุมคำสั่งต่างๆของโปรแกรม
- การสร้างอาร์ทีพีคอนเนคชั่นเพื่อทำการใช้ในการส่งภาพจากเว็บแคมไปยังฝั่งเซิร์ฟเวอร์ (Server)
- การรับข้อมูลคำสั่งต่างๆ จากส่วนควบคุม
- การแปลเป็นคำสั่งต่างๆเช่น
- คำสั่งในการสร้างคอนเนคชั่น
- คำสั่งในการจัดการกับรูปภาพในลักษณะต่างๆ เช่น
    - ภาพเคลื่อนไหวขาวดำ สีธรรมชาติ
    - ภาพแคปเจอร์ขาวดำ สีธรรมชาติ
    - การตั้งเวลาในการถ่ายภาพ
- การจัดการตามคำสั่งในส่วนนี้จะรับข้อมูลภาพที่ได้จากเว็บแคมไปทำการประมวลผลตามคำสั่งควบคุมจากฝั่งเซิร์ฟเวอร์ (Server)
- กระบวนการจัดการกับภาพเพื่อทำให้เกิดความปลอดภัย
- การส่งภาพ จะทำการส่งข้อมูลภาพไปยังเซิร์ฟเวอร์ (Server) ด้วยโปรโตคอลอาร์ทีพี

2. ทางฝั่งผู้ใช้งานเซิร์ฟเวอร์(Server) จะพัฒนาโปรแกรมชื่อว่า IsagCam Manager การทำงานในฝั่งของเซิร์ฟเวอร์ (Server) หรือ IsagCam Manager จะมีการทำงานดังต่อไปนี้

- การคอนโทรลส่วนของคำสั่ง ในส่วนนี้จะเป็นส่วนควบคุมคำสั่งการควบคุมต่างๆ อันได้แก่

ค่าที่ใช้ในการคอนฟิคกูเรชั่น

- การแคปเจอร์/สตรีมมิ่ง
- ชนิดของไฟล์ภาพที่จะทำการบันทึก
- พาร์ทในการบันทึกภาพที่ได้รับ
- เวลาในการแคปเจอร์
- ภาพสีธรรมชาติ ภาพขาวดำ
- การบันทึกเวลาจริง/การตั้งเวลาในการบันทึก
- ดิสก์และการจัดการกับดิสก์เมื่อเต็ม

ค่าที่ใช้ในการควบคุมอื่นๆ เช่น

- ส่งคำสั่ง
- หยุดการทำงาน

- การนำคำสั่งที่ได้รับมาจากค่าการคอนฟิคกูเรชั่นต่างๆมาสร้างเป็นรูปแบบของการคอนโทรล
- กระบวนการจัดการกับคำสั่งเพื่อทำให้เกิดความปลอดภัย
- การส่งข้อมูลคำสั่งผ่านทางคอนโทรลคอนเนคชั่น
- การจัดการกับดิสก์
- การแสดงข้อมูลของคำสั่งของแต่ละกล้อง
- การบันทึกผลของข้อมูลภาพ

## 5.2 ปัญหาและอุปสรรคและแนวทางการแก้ไข

1. ปัญหาในเรื่องของทราฟฟิคของการส่งข้อมูลเนื่องจากข้อมูลมาจาก 4 ต้นทางและฝั่งปลายทางมีเพียงเครื่องเดียวที่จะทำการรับภาพและแสดงผล ทำให้การส่งบางครั้งมีความล่าช้า

การแก้ไข

- การจัดการเรื่องเวลาในการแคปเจอร์ โดยที่ฝั่งไคล์แอนต์ไม่จำเป็นที่จะต้องแคปเจอร์ตลอดเวลา แต่จะทำตามคำสั่งของเซิร์ฟเวอร์เท่านั้นจะเป็นการลดภาระส่วนหนึ่งของฝั่งเซิร์ฟเวอร์
- ที่ฝั่งไคล์แอนต์ทำการตรวจเช็คทีละบิตข้อมูลของภาพที่แคปเจอร์ใหม่และภาพเดิมที่เคยแคปเจอร์ไว้ ก่อนทำการส่งจริง ถ้าเกิดว่าภาพใหม่มีความแตกต่างมากกว่า 30 % จะทำการส่งใหม่ แต่ถ้าไม่จะถือว่าเป็นภาพเดียวกันจะไม่ส่งภาพที่ฝั่งเซิร์ฟเวอร์จะทำการบันทึกรูปเดิมที่ช่วงเวลาใหม่

2. ปัญหาการตีกันของสตรีมภาพ เนื่องจากอาร์ทีพีในส่วนของ Join Rtpsession พบว่าเมื่อมีการส่งข้อมูลกล้องเข้ามาใหม่จะมีการเพิ่มผู้ใช้งานซ้ำจากของเดิม อันเนื่องมาจากการทำงานของอาร์ทีพีเป็นแบบ มัลติคาสต์ทำให้เกิดการซ้อนกันของสตรีมภาพที่หน้าจอแสดงผล

การแก้ไข

- ทำการเช็คที่ส่วนของ Join Rtpsession ว่าถ้ามีข้อมูลเข้ามาใหม่ให้ทำการลบ participant ที่ไม่ใช่ออกจากกลุ่มผู้ใช้งาน จะทำให้แก้ไขการตีกันของสตรีมภาพได้

## 5.3 แนวทางการพัฒนาต่อ

ในอนาคตข้างหน้าอาจมีการพัฒนาโครงการการสอดส่องความปลอดภัยด้วยเว็บแคมให้สามารถใช้งานได้อย่างมีประสิทธิภาพกว่าเดิมซึ่งอาจเพิ่งส่วนของการใช้งานต่างๆเพิ่มขึ้นดังต่อไปนี้

การดักจับภาพเคลื่อนไหว

1. การตรวจจับภาพเคลื่อนไหว (Motion detection) โดยการรวมความสามารถของ Image processing มาใช้ในการตรวจสอบความเคลื่อนไหวเพื่อใช้ในการทำสัญญาณกันขโมย
2. การบันทึกเสียง
3. การบันทึกภาพเป็นแบบวิดีโอ และสามารถเล่นไฟล์ภาพที่บันทึกได้ในตัวโปรแกรม

# บรรณานุกรม

Karli Watson, David Espinosa, Zach Greenvoss, Jacob Hammer Pederson, Christian nagel , Jon D.Reid, Matthew Reynoalds, Morgan Skinner ,Eric White สุวัฒนา สุขสมจิตน์ . "**คัมภีร์การใช้ Visual Studio.NET**." พิมพ์ครั้งที่ 1 กรุงเทพฯ: บริษัท โปรวิชั่น จำกัด

พฤกษ์ ธนานุสนธิ์,วรวิทย์ ลีลาประเสริฐวงศ์. 2549. **"โปรแกรมคัดกรองข้อมูลเว็บ. "**ปริญญานิพนธ์วิศวกรรมศาสตบัณฑิต สาขาวิชาวิศวกรรม คอมพิวเตอร์ คณะวิศวกรรมศาสตร์, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

**"MSDN Documentation"** , [online]. Available:
http://msdn2.microsoft.com/en-us/default.aspx

**"The Code Project-Free Source Code and Tutorials"**, [online]. Available:
http://www.codeproject.com

**"The Number one the number one developer site"** , [online]. Available:
http://www.codeguru.com

**"C# Corner-C-Sharp C#.NET CSharp VB.NET ASP.NET Visual Studio .NET",** [online]. Available:
http://www.c-sharpcorner.com

Mark Strawmyer **"Creating a Windows Service in .NET"** , [online].Available:
http://www.developer.com/net/csharp/article.php/2173801

Fadi Dot net **"Welcome to our Network Programming Examples & Tutorials"**, [online].Available:
http://www.fadidotnet.org/online_book/Network_Programming_online.htm

Jacob Grass **"How to convert a colour image to grayscale"**, [online].Available:
http://www.bobpowell.net/grayscale.htm