บทที่ 4

การดำเนินการวิจัย

ในบทนี้จะกล่าวถึงแนวคิดในการนำการวิเคราะห์เชิงโครงสร้างมาช่วยในการออกแบบระบบเชิงวัตถุ และขั้นตอนการปรับเปลี่ยนระบบเดิมให้เป็นระบบเชิงวัตถุ

พอล ที วอล์ด[7] ได้สรุปบทความที่ทำการศึกษาว่าในการเปลี่ยนระบบที่ออกแบบเชิงโครงสร้างให้เป็นระบบเชิงวัตถุในระบบเวลาจริงนั้น สามารถทำโดยการพิจารณาถึงคุณสมบัติของวัตถุเป็นหลัก และได้ใช้ แผนภาพเอนทิตี-รีเลชันชิพ และแผนภาพกระแสข้อมูล ในการช่วยในการมองระบบ

นอกจากนี้ ชิดนีย์ ซี ไบลิน[10,11] ได้พัฒนาวิธีการวิเคราะห์ความต้องการสำหรับซอฟต์แวร์เชิงวัตถุ โดยให้ชื่อว่า โอโอเอส (OOS หรือ Object-Oriented Requirements Specification) มีจุดประสงค์เพื่อที่จะหาความเข้ากันได้ระหว่างการวิเคราะห์เชิงโครงสร้าง และการออกแบบเชิงวัตถุ โดยในการวิเคราะห์เชิงโครงสร้างก็จะใช้สัญลักษณ์ที่คล้ายกับแผนภาพกระแสข้อมูลในการทำดีคอมโพสิชันระบบ (system decomposition) แต่ในการวิเคราะห์เชิงโครงสร้างทั่วไปกำหนดให้ฟังก์ชันสามารถรวมกลุ่มกันได้ก็ต่อเมื่อฟังก์ชันเหล่านั้นเป็นส่วนประกอบของขั้นตอนในการประมวลผลของฟังก์ชันในระดับสูงขึ้น ในขณะที่โอโอเอสจะกำหนดให้ฟังก์ชันสามารถรวมกลุ่มได้ก็ต่อเมื่อฟังก์ชันเหล่านั้นทำงานบนข้อมูลนามธรรม (data abstraction) เดียวกัน ซึ่งวิธีการนี้สามารถสนับสนุนการเอนแคปซูเลชันของฟังก์ชันและข้อมูลได้

หลักการในการออกแบบระบบจากเชิงโครงสร้างมาเป็นเชิงวัตถุ

เนื่องจากระบบที่ใช้ในการศึกษาได้มีการพัฒนาจนสำเร็จมาแล้ว ดังนั้นการวิจัยจึงจะมุ่งเน้นการนำทรัพยากรที่มีอยู่แล้วมาใช้ให้ได้มากที่สุด เพื่อเป็นการลดเวลาและค่าใช้จ่ายในการพัฒนา โดยมีขั้นตอนดังนี้

1. นำข้อมูลที่ใช้ในระบบมาจากแผนภาพกระแสข้อมูล มาตรวจสอบว่ามีกระแสข้อมูล (data flow) ใดที่มีข้อมูลเดียวกันหรือกลุ่มเดียวกัน
2. ศึกษาและตรวจสอบขั้นตอนการทำงานที่จำเป็นของระบบจากผังเชิงโครงสร้าง เพื่อให้เห็นลำดับขั้นของกระบวนการ ซึ่งจะใช้ประกอบการพิจารณาในการจัดกลุ่มของกระบวนการกับข้อมูล แล้วพิจารณาจัดกลุ่มของขั้นตอนการทำงานที่ได้ทำงานบนข้อมูลเดียวกัน เพื่อนำมาสร้างเป็นวัตถุของระบบ

3. ศึกษาแผนภาพเอนทิตี-รีเลชันชิพ และตารางข้อมูล เพื่อสร้างวัตถุสำหรับตารางข้อมูลนั้น ซึ่งจะช่วยให้การทำงานเป็นระเบียบ และควบคุมได้ง่ายขึ้น จากนั้นพิจารณาสร้างวัตถุสำหรับตารางข้อมูล เพื่อให้ทำหน้าที่จัดการกับแฟ้มข้อมูลต่าง ๆ
4. พิจารณาสร้างวัตถุเพิ่มเติมจากผังการไหล เพื่อให้มีวัตถุครบถ้วนและใช้ทำงานได้ โดยแบ่งพิจารณาซอฟต์แวร์ออกเป็นกลุ่ม ๆ ตามส่วนประกอบของวิธีการออกแบบเชิงวัตถุของปีเตอร์ คอล์ด[8] ได้แก่ ส่วนจัดการข้อมูล ส่วนติดต่อผู้ใช้ ส่วนติดต่อระบบ และส่วนขอบเขตปัญหา ซึ่งจะช่วยในการแยกแยะซอฟต์แวร์ออกเป็นส่วน ๆ ซึ่งจะทำให้ง่ายต่อการค้นหาวัตถุที่ทำงานเกี่ยวข้องในแต่ละส่วนประกอบ
5. หาวัตถุเพิ่มเติมจากขอบเขตปัญหาเดิม เพื่อให้ได้วัตถุที่ยังขาด และจำเป็นต่อการทำงานของระบบ
6. หลังจากได้วัตถุจนพอใจแล้วก็ใช้วิธีการออกแบบเชิงวัตถุทั่วไปได้

<u>การออกแบบโปรแกรมไคลเอนต์</u>

จากหลักการในการออกแบบข้างต้น ได้นำมาใช้ศึกษาโปรแกรมไคลเอนต์เพื่อออกแบบให้เป็นโปรแกรมเชิงวัตถุ ได้ผลดังแผนภาพคลาสรูปที่ 4.1

ในการพิจารณาได้ทำการพิจารณาจากแผนภาพกระแสข้อมูล ผังเชิงโครงสร้าง แผนภาพเอนทิตี-รีเลชันชิพ และตารางข้อมูล โดยทำการพิจารณาทีละส่วนเป็นขั้นตอนดังนี้

1. จากแผนภาพกระแสข้อมูลของส่วนไคลเอนต์ (รูปที่ 2.4) ทำการรวบรวมข้อมูลที่ถูกส่งระหว่างกระบวนการ ทั้งนี้ไม่รวมถึงข้อมูลที่ติดต่อกับเอนทิตีภายนอก และแฟ้มข้อมูล จะได้ดังนี้

ระหว่างกระบวนการที่ 1.0 กับ 2.0 มีข้อมูล Subject Code

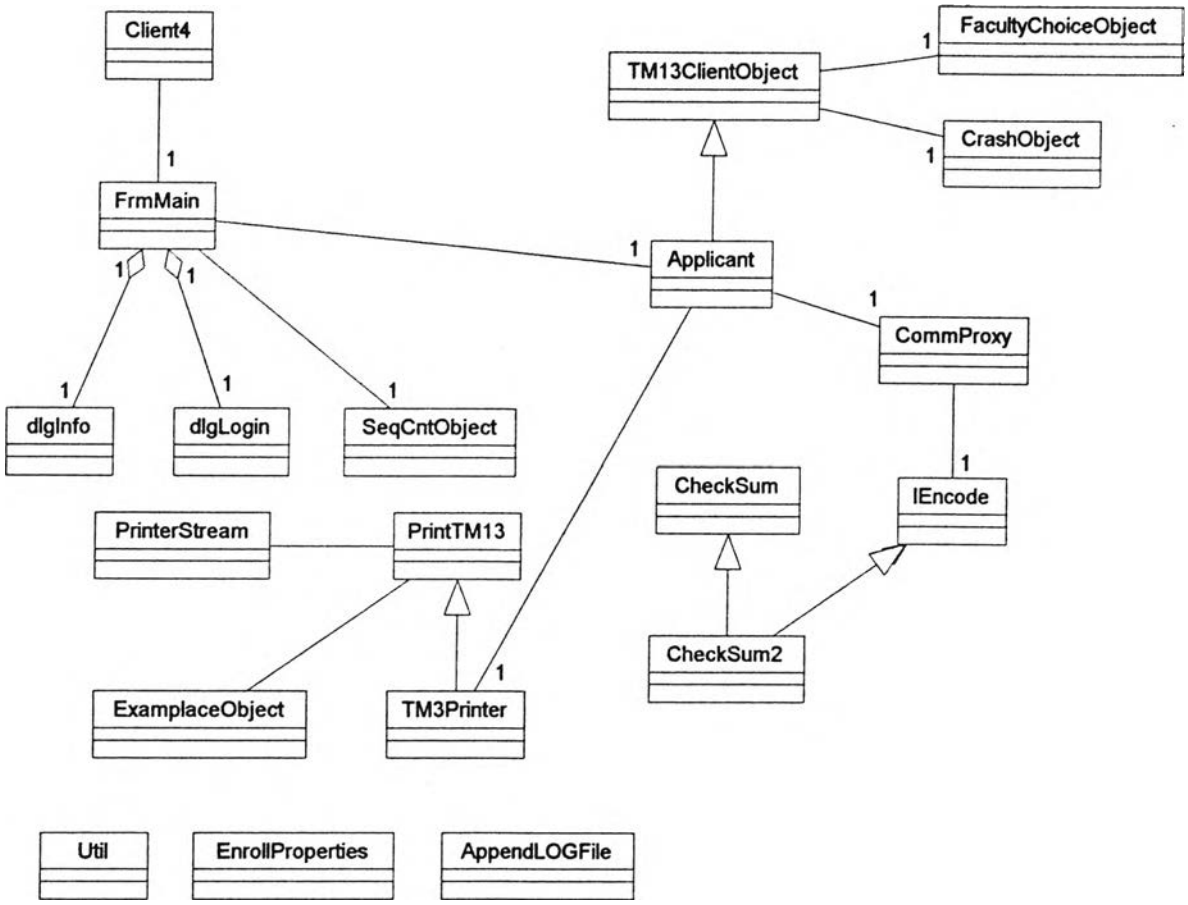
ระหว่างกระบวนการที่ 1.0 กับ 3.0 มีข้อมูล Subject Code

ระหว่างกระบวนการที่ 2.0 กับ 4.0 มีข้อมูล Choices(4), Subject Code

ระหว่างกระบวนการที่ 5.0 กับ 6.0 มีข้อมูล TM13

ระหว่างกระบวนการที่ 6.0 กับ 7.0 มีข้อมูล TM13

จะเห็นว่ามีกระบวนการที่ใช้ข้อมูลแบบเดียวกัน ดังนั้นการจัดกลุ่มของกระบวนการจึงสามารถทำได้เป็น 2 กลุ่มคือกลุ่มที่ใช้ข้อมูล Subject Code และกลุ่มที่ใช้ข้อมูล TM13



รูปที่ 4.1 แสดงแผนภาพคลาสของโปรแกรมไคลเอนต์

การที่ไม่ได้พิจารณาข้อมูลที่ส่งผ่านระหว่างระบบกับเอนทิตีภายนอก และแฟ้มข้อมูล เพราะเอนทิตีภายนอกและแฟ้มข้อมูลจะถูกจัดให้มีวัตถุคอยควบคุมอีกชั้นหนึ่ง ดังนั้นในส่วนเหล่านี้จะถูกใช้งานผ่านวัตถุอยู่แล้ว

2. เมื่อศึกษาถึงผังเชิงโครงสร้างของส่วนไคลเอนต์ ทำให้เห็นภาพชัดเจนมากยิ่งขึ้น เพราะพบว่ากระบวนการเหล่านั้นอยู่คนละกิ่งของลำดับชั้น ในขั้นนี้สามารถตัดสินใจสร้างวัตถุขึ้นพร้อมกัน 2 วัตถุ แต่เมื่อพิจารณาถึงข้อมูลย่อยของข้อมูล TM13 แล้วพบว่า ข้อมูล Subject Code และ Choices เป็นข้อมูลย่อยของข้อมูล TM13

ดังนั้นจึงพิจารณารวมกลุ่มของกระบวนการทั้งหมดนี้เข้าด้วยกันโดยเป็นวัตถุเดียว ที่ทำงานตามหน้าที่ของกระบวนการต่าง ๆ ที่ถูกรวมกันเข้ามา ตั้งชื่อวัตถุนี้ว่า Applicant

3. จากการศึกษาตารางข้อมูลของโปรแกรมไคลเอนต์ ได้ทำการสร้างวัตถุสำหรับแต่ละแฟ้มข้อมูลเพื่อใช้ในการเข้าถึงและปรับปรุงข้อมูลแทนการให้กระบวนการอื่น ๆ เข้ามาใช้งานแฟ้มโดยตรง วิธีนี้ทำให้การควบคุมข้อมูลที่อยู่ในแฟ้มสะดวกขึ้นและสามารถลดข้อผิดพลาดได้

วัตถุที่ถูกสร้างขึ้นเพื่อใช้แทนระเบียนของแฟ้มข้อมูล สรุปได้ดังตารางที่ 4.1

ตารางที่ 4.1 แสดงชื่อของวัตถุที่แทนระเบียนของแฟ้มข้อมูลในโปรแกรมไคลเอนต์

ชื่อแฟ้มข้อมูล	ชื่อวัตถุ	หน้าที่
Faculty -Subject Table	FacultyChoiceObject	ข้อมูลวิชาสอบของแต่ละคณะ
Examplace Name Table	ExamplaceObject	ชื่อของเขตการสอบใช้ในการพิมพ์
Sequnce Count Table	SeqCntObject	เก็บลำดับการรับสมัครที่ไคลเอนต์
Crash Table	CrashObject	ข้อมูลวิชาสอบตรงกัน
TM13 Table	TM13ClientObject	ข้อมูลผู้สมัครทั้งหมด

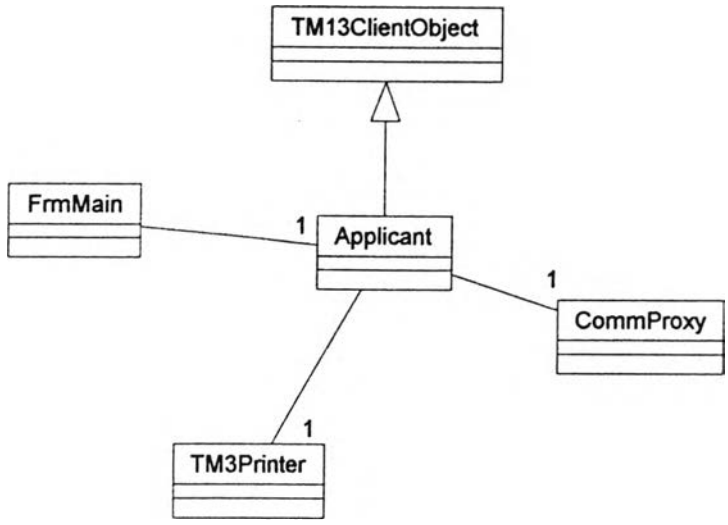
วัตถุต่าง ๆ เหล่านี้จะมีบริการที่เป็นบริการพื้นฐานของคลาส และบริการที่ได้จากการศึกษาแผนภาพกระแสข้อมูล โดยดูว่าแฟ้มข้อมูลนี้สามารถให้ค่าหรือรับค่าแบบใดได้บ้าง

จากขั้นตอนนี้พบว่ามีการสร้างวัตถุของแฟ้ม TM13 ซึ่งในการหาวัตถุขั้นแรกสุดได้มีการสร้างวัตถุที่ใช้งานข้อมูล TM13 เช่นกัน ดังนั้นจึงได้กำหนดให้วัตถุ Applicant สืบทอดมาจากวัตถุของแฟ้ม TM13 เพราะจะทำให้วัตถุ Applicant สามารถใช้งานข้อมูล TM13 ได้ทันที สิ่งที่ทำให้วัตถุของข้อมูลTM13 และวัตถุ Applicant แตกต่างกันก็คือบริการที่มีของวัตถุ เนื่องจากกระบวนการในผังเชิงโครงสร้างมีกระบวนการพิมพ์บัตรประจำตัวผู้สอบด้วย ซึ่งเป็นกระบวนการที่ไม่จำเป็นสำหรับแฟ้มข้อมูล (คือแฟ้มข้อมูลจะให้บริการการเก็บและค้นคืนข้อมูล การจะพิมพ์ข้อมูลต้องค้นคืนข้อมูลก่อนแล้วจึงนำข้อมูลไปพิมพ์) ดังนั้นวัตถุ Applicant จึงมีกระบวนการพิมพ์บัตรประจำตัวผู้สอบเป็นบริการหนึ่งด้วย

นอกจากนี้ เนื่องจากในส่วนขอบเขตปัญหา จำเป็นต้องมีวัตถุที่ทำหน้าที่ใช้งานส่วนประกอบต่าง ๆ หรือทำหน้าที่หลักในการทำงานของโปรแกรมนี้ เนื่องจากได้มีการรวมขั้นตอนการทำงานส่วนใหญ่ไปไว้ในวัตถุ TM13ClientObject แล้ว แต่วัตถุนี้ไม่สามารถติดต่อกับวัตถุอื่น ๆ เช่น วัตถุ CommProxy เพื่อส่งข้อมูลไปโปรแกรมเซิร์ฟเวอร์ หรือไม่สามารถส่งข้อมูลไปยังวัตถุ TM3Printer เพื่อพิมพ์บัตรประจำตัวผู้สอบได้ จึงต้องนำวัตถุ Applicant มาใช้เพื่อเชื่อมการทำงานต่าง ๆ เหล่านี้ ดังรูปที่ 4.2

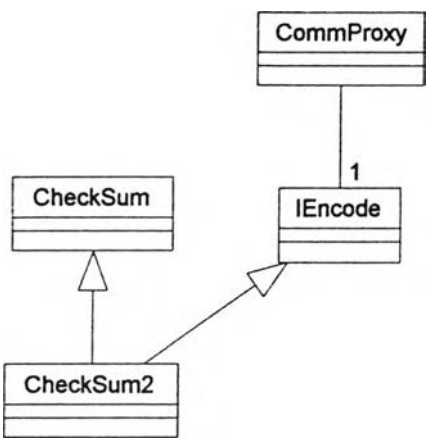
4. จากนั้นเป็นการหาวัตถุที่จะใช้ประกอบในโปรแกรม เพื่อให้โปรแกรมสามารถใช้งานได้ โดยทำการศึกษาผังการไหลเพิ่มเติมกับแผนภาพกระแสข้อมูลและผังเชิงโครงสร้าง

จากผังการไหลพบว่ามีการติดต่อส่งข้อมูลกับโปรแกรมเซิร์ฟเวอร์ ซึ่งในจุดนี้สามารถนำมาทำเป็นวัตถุเพื่อบรรจุโปรโตคอลการสื่อสารได้



รูปที่ 4.2 แสดงความสัมพันธ์ของวัตถุ Applicant กับวัตถุที่เกี่ยวข้อง

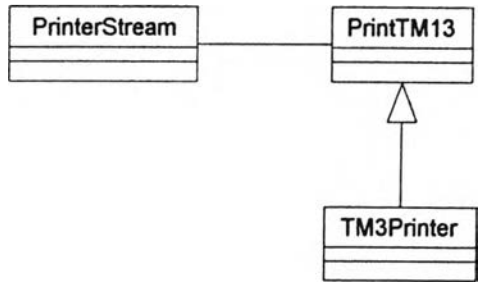
โปรแกรมไคลเอนต์จะทำการติดต่อโปรแกรมเซิร์ฟเวอร์เพื่อขอเลขที่นั่งสอบ และยืนยันการสมัครผ่านโปรโตคอล TCP/IP ดังนั้นวัตถุตัวนี้จึงทำการรับ Socket ที่ใช้มาสร้างช่องทางสื่อสารเพื่อใช้ติดต่อ ส่วนบริการของวัตถุนี้จะมีการส่งข้อความ และรับข้อความเท่านั้น โดยวัตถุจะทำการเข้ารหัสหรือถอดรหัสได้โดยอัตโนมัติ วัตถุนี้มีชื่อว่า CommProxy เมื่อแสดงพร้อมกับวัตถุที่ใช้เข้ารหัสหรือถอดรหัสจะมีโครงสร้างดังรูปที่ 4.3



รูปที่ 4.3 แสดงวัตถุ CommProxy และการประยุกต์ใช้คลาส IEncode

จากรูปที่ 4.3 คลาส CheckSum2 ได้มีการสืบทอดคุณสมบัติมาจากคลาส IEncode ซึ่งเป็นคลาสนามธรรม (abstract class) ทำให้วัตถุ CommProxy สามารถใช้งาน คลาส CheckSum2 ได้โดยผ่านบริการที่ได้นิยามเอาไว้ในคลาส IEncode คลาส CheckSum2 นี้ใช้เพื่อทำการตรวจสอบค่า checksum เพื่อความถูกต้องในการรับหรือส่งข้อมูล

จากผังการไหลมีการพิมพ์บัตรประจำตัวผู้สอบ ซึ่งในการสร้างวัตถุก็สามารถนำมาสร้างคลาสของเครื่องพิมพ์ได้ เพราะการหาวัตถุสามารถหาได้จากอุปกรณ์ที่มีการใช้งานในระบบได้[9]



รูปที่ 4.4 แสดงความสัมพันธ์ของวัตถุกลุ่มเครื่องพิมพ์

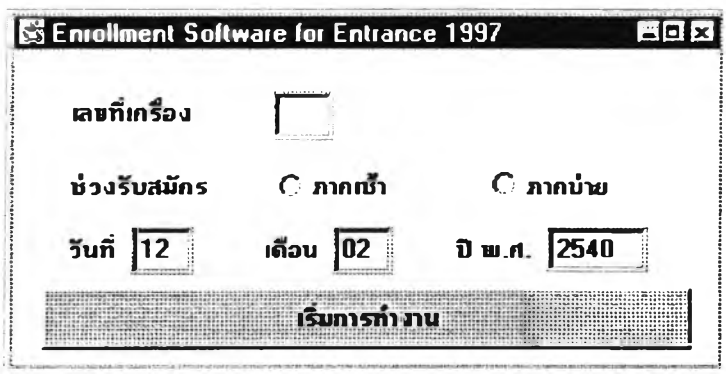
จากรูปที่ 4.4 มีวัตถุเพิ่มเติมดังนี้ คลาส PrinterStream ทำหน้าที่ติดต่อกับเครื่องพิมพ์โดยมองเครื่องพิมพ์เป็นเสมือนแฟ้มข้อมูลแฟ้มหนึ่ง ซึ่งข้อความที่จะพิมพ์เป็นข้อความใดก็ได้ และจะเห็นว่าสามารถนำวัตถุนี้ไปใช้กับโปรแกรมอื่นได้

คลาส PrintTM13 ทำหน้าที่พิมพ์ข้อมูลที่เก็บไว้ในแฟ้ม TM13 หรือจากวัตถุ TM13ClientObject โดยจะทำการจัดตำแหน่งการพิมพ์ต่าง ๆ ให้ถูกต้อง วัตถุนี้มีความสัมพันธ์กับวัตถุ PrinterStream แบบ associate

คลาส TM3Printer เป็นวัตถุที่สืบทอดมาจาก PrintTM13 โดยมีจุดประสงค์เพื่อให้เป็นคลาสเชื่อมต่อระหว่างวัตถุในส่วนของขอบเขตปัญหากับตัววัตถุเครื่องพิมพ์ เพื่อให้สะดวกต่อการแก้ไขภายหลัง

จากการศึกษางานวิจัยเก่า[2]ประกอบ ทำให้ได้ทราบแนวทางการออกแบบหน้าจอรับสมัคร ซึ่งต้องทำการออกแบบใหม่เพื่อให้ใช้กับภาษาจาวาได้ ในการพิจารณาหน้าจอการทำงานอาจแบ่งวัตถุเป็นวัตถุของแต่ละหน้าต่าง โดยมีองค์ประกอบของหน้าต่างเป็นคุณลักษณะของวัตถุนั้น

1. หน้าจอเริ่มใช้งานโปรแกรมไคลเอนต์ ได้สร้างเป็นวัตถุ ซึ่งสืบทอดมาจากคลาส Dialog ของภาษาจาวา ซึ่งจะต้องแสดงผ่านหน้าต่างโปรแกรม กำหนดให้เป็นวัตถุ dlgLogin ดังรูปที่ 4.5



รูปที่ 4.5 แสดงหน้าจอเริ่มใช้งานโปรแกรมไคลเอนต์

หน้าจอนี้จะทำการรับเลขที่เครื่องรับสมัคร ช่วงเวลาการรับสมัคร และวันที่เดือนปีที่ทำการรับสมัคร ข้อมูลเหล่านี้จะถูกใช้เป็นส่วนหนึ่งของข้อมูลผู้สมัคร โดยหน้าจอนี้จะแสดงเพียงครั้งเดียวตอนเริ่มโปรแกรม

2. หน้าจอรับสมัคร ได้สร้างขึ้นเป็นวัตถุที่สืบทอดมาจากคลาส Frame ของภาษาจาวา โดยจะทำหน้าที่เป็นหน้าต่างหลักในการกรอกข้อมูลผู้สมัคร กำหนดให้เป็นวัตถุ FrmMain ดังรูปที่ 4.6

หน้าจอนี้จะขึ้นมาหลังจากที่เจ้าหน้าที่ได้กรอกข้อมูลจากหน้าจอเริ่มใช้งานเสร็จเรียบร้อยแล้ว หน้าจอแบ่งเป็น 3 ส่วน ซึ่งในขั้นตอนการสมัครก็แบ่งเป็น 3 ส่วนเช่นกัน จากรูปที่ 2.9 พิจารณาได้ดังนี้

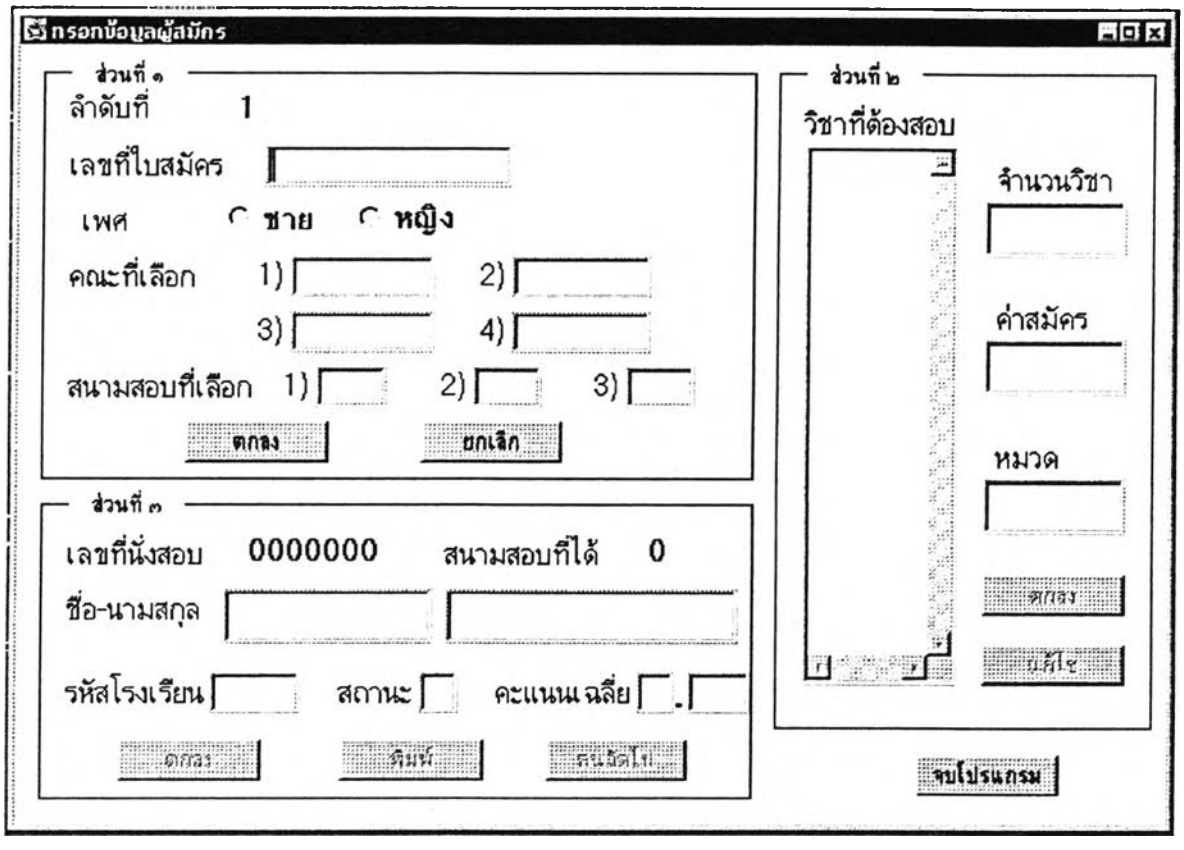
ส่วนที่ 1 เป็นการรับข้อมูลเบื้องต้น เพื่อการพิจารณาหมวดสอบ วิชาที่ใช้สอบ จำนวนวิชาที่ใช้สอบ และจำนวนเงินค่าสมัคร โดยทำการรับข้อมูลเลขที่ใบสมัคร เพศ คณะที่เลือก 4 อันดับ และเขตการสอบที่เลือก 3 อันดับ เมื่อผู้สมัครยืนยันการกรอกข้อมูลแล้วจึงทำการคำนวณต่างๆ ข้างต้น แล้วจึงเข้าสู่ส่วนที่ 2

ส่วนที่ 2 เป็นการแสดงผลที่คำนวณได้ ในกรณีที่ผู้สมัครพบข้อผิดพลาดเช่นมีวิชาที่ไม่ได้สอบขึ้นมา ก็สามารถย้อนกลับไปแก้ไขข้อมูลในส่วนที่ 1 ได้ เมื่อผู้สมัครตอบตกลงในส่วนนี้แล้วโปรแกรมไคลเอนต์ก็จะทำการติดต่อโปรแกรมเซิร์ฟเวอร์เพื่อขอเลขที่นั่งสอบ และเขตการสอบต่อไป เมื่อเซิร์ฟเวอร์ส่งข้อมูลกลับมาแล้วก็จะทำการรับสมัครในส่วนที่ 3 ต่อไป

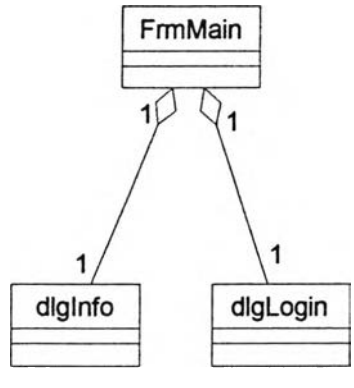
ส่วนที่ 3 เป็นการรับข้อมูลผู้สมัครส่วนที่เหลือได้แก่ ชื่อ นามสกุล รหัสโรงเรียน สถานภาพทางการศึกษา และเกรดเฉลี่ย เมื่อผู้สมัครยืนยันความถูกต้องแล้วก็จะเป็นการพิมพ์บัตรประจำตัวผู้สอบต่อไป ในการพิมพ์บัตรประจำตัวผู้สอบสามารถพิมพ์ซ้ำได้ เมื่อเสร็จสิ้นการสมัครของผู้สมัครแล้วก็จะทำการย้อนไปรับข้อมูลการสมัครจากผู้สมัครคนใหม่จากส่วนที่ 1 อีกครั้ง

3. หน้าจอแสดงข้อผิดพลาด ได้สร้างเป็นวัตถุ ซึ่งสืบทอดมาจาก Dialog คลาสเช่นกัน โดยเป็นหน้าต่างแสดงข้อความผิดพลาดเท่านั้น ผู้กรอกข้อมูลต้องกดปุ่มเพื่อปิดหน้าจอนี้ก่อนจะกลับไปแก้ไขข้อผิดพลาดที่หน้าจอรับสมัคร

เนื่องจากทั้งหน้าจอจะจัดอยู่ในส่วนติดต่อผู้ใช้ ดังนั้นโครงสร้างความสัมพันธ์ของหน้าจอทั้งหมดสามารถเขียนได้ดังรูปที่ 4.7



รูปที่ 4.6 แสดงหน้าจอรับสมัคร



รูปที่ 4.7 แสดงความสัมพันธ์ระหว่างวัตถุในส่วนการติดต่อผู้ใช้

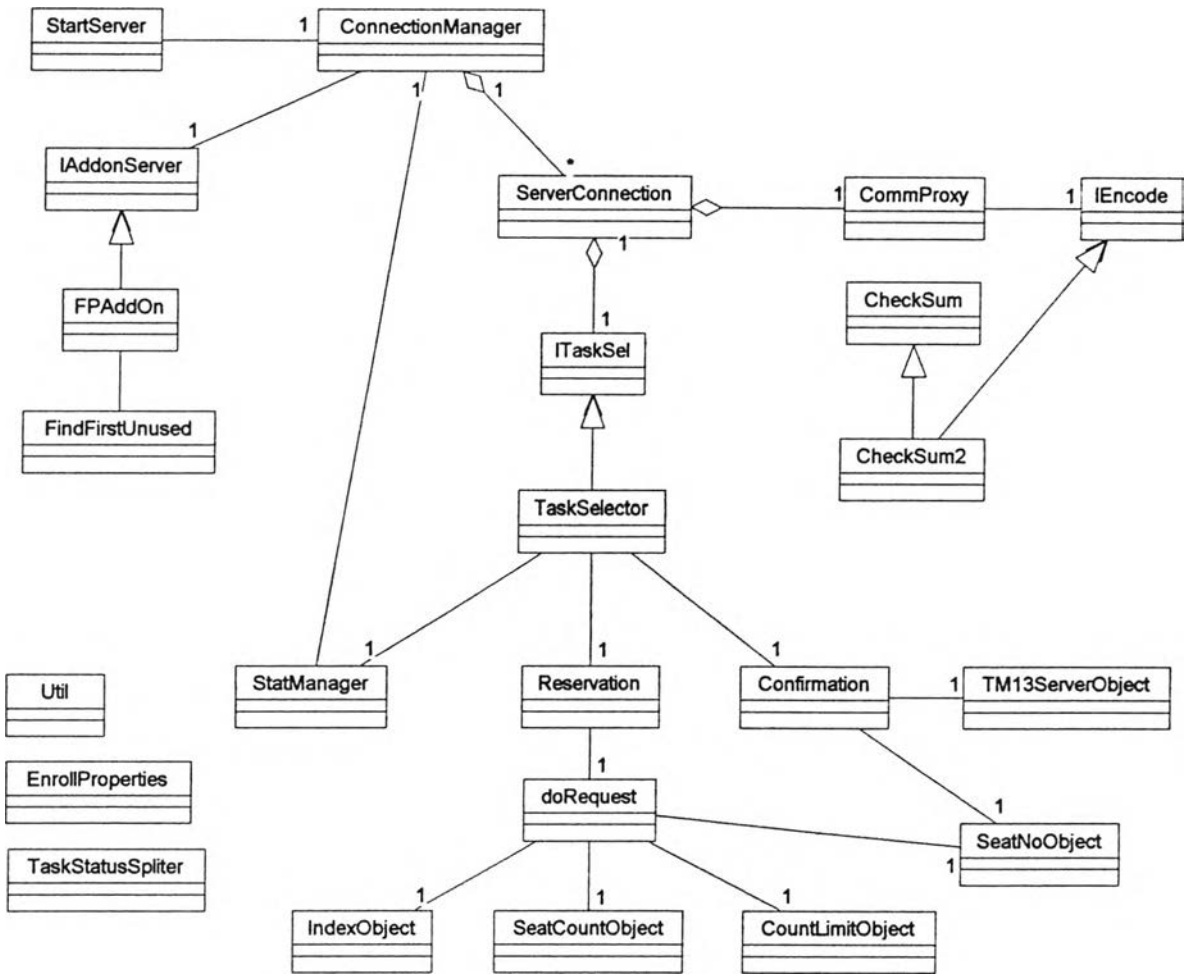
จากรูปที่ 4.7 หน้าจอรับสมัคร (คลาส FrmMain) จะเป็นหลักในการทำงาน โดยมีหน้าจอเริ่มใช้งาน (คลาส dlgLogin) และหน้าจอแสดงความผิดพลาด (คลาส dlgInfo) เป็นส่วนประกอบ

จากรูปที่ 4.1 ยังมีวัตถุที่ยังไม่ได้กล่าวถึงดังนี้ วัตถุ Client4 เป็นวัตถุที่มีฟังก์ชัน main เพื่อใช้ในการเริ่มต้นโปรแกรมนี้ โดยจะทำหน้าที่สร้างวัตถุและตั้งค่าต่าง ๆ วัตถุ Util เป็นวัตถุที่มีบริการลักษณะอรรถประโยชน์ ใช้ในการตรวจสอบขนาดของข้อมูลที่กรอกลงในช่องรับข้อมูล วัตถุ EnrollProperties เป็นวัตถุที่ใช้ในการอ่านข้อมูลการตั้งค่าการทำงานต่าง ๆ จากแฟ้ม Enroll.conf วัตถุ AppendLOGFile เป็น

วัตถุที่ใช้เพื่อบันทึกข้อความผิดพลาดที่เกิดขึ้นในระหว่างโปรแกรมทำงานลงแฟ้มข้อมูล รายละเอียดของคลาสสำหรับโปรแกรมไคลเอนต์อยู่ในภาคผนวก ค.

การออกแบบโปรแกรมเซิร์ฟเวอร์

ในการออกแบบโปรแกรมเซิร์ฟเวอร์ ก็มีขั้นตอนคล้ายกับการออกแบบโปรแกรมไคลเอนต์เช่นกัน ได้ผลดังแผนภาพคลาสรูปที่ 4.8



รูปที่ 4.8 แสดงแผนภาพคลาสของโปรแกรมเซิร์ฟเวอร์

1. พิจารณาแผนภาพกระแสข้อมูลรูปที่ 2.5 แสดงข้อมูลที่ส่งระหว่างกระบวนการได้ดังนี้

ระหว่างกระบวนการที่ 1.0 กับ 2.0 มีข้อมูล Examplace, Subject Type

ระหว่างกระบวนการที่ 2.0 กับ 3.0 มีข้อมูล Examplace, Subject Type, Seat No Sequence

ระหว่างกระบวนการที่ 5.0 กับ 7.0 มีข้อมูล TM13

ระหว่างกระบวนการที่ 7.0 กับ 8.0 มีข้อมูล TM13

จะเห็นได้ว่ามีกระบวนการที่ใช้ข้อมูล TM13 เหมือนกัน และมีกระบวนการที่ใช้ข้อมูล Examplace และ Subject Type เหมือนกันเช่นกัน นอกจากนี้ยังพบว่าข้อมูล Examplace, Subject Type และ Seat No Sequence ก็เป็นข้อมูลย่อยของข้อมูล TM13 ด้วย แต่ในที่นี้จะยังไม่รวมกระบวนการเป็นกลุ่มเดียวกันตามแบบโปรแกรมไคลเอนต์ เพราะยังต้องพิจารณาผังเชิงโครงสร้างอีก

2. จากการศึกษาผังโครงสร้างของเซิร์ฟเวอร์พบว่ามีกระบวนการหลักอยู่ 2 งานซึ่งเป็นการทำงานแยกจากกัน โดยเมื่อเทียบกับแผนภาพกระแสข้อมูลแล้วเป็นการรับและส่งค่าคืนไปยังเอนทิตีภายนอกทั้ง 2 กระบวนงานซึ่งทำให้ยังไม่สามารถสรุปให้เหมาะสมได้ว่าควรจะรวมกลุ่มของกระบวนการทั้ง 2 นี้หรือไม่

3. ทำการสร้างวัตถุเพื่อจัดการกับแฟ้มข้อมูล โดยสรุปการสร้างวัตถุเพื่อจัดการกับแฟ้มข้อมูลจะได้วัตถุดังตารางที่ 4.2

ตารางที่ 4.2 แสดงชื่อของวัตถุที่แทนระเบียนของแฟ้มข้อมูลในโปรแกรมเซิร์ฟเวอร์

ชื่อแฟ้มข้อมูล	ชื่อวัตถุ	หน้าที่
Index Table	IndexObject	ข้อมูลดรรชนีเลขที่นั่งสอบว่างในแฟ้ม SeatNo Table
SeatNo Table	SeatNoObject	ข้อมูลเลขที่นั่งสอบ
Count Limit Table	CountLimitObject	จำนวนจำกัดในแต่ละเขตการสอบ
Seat Count Table	SeatCountObject	จำนวนผู้สมัครในแต่ละเขตการสอบ
TM13 Table	TM13ServerObject	ข้อมูลผู้มัครทั้งหมด

จะสังเกตุได้ว่าในขั้นตอนนี้ได้มีการสร้างวัตถุของแฟ้ม TM13 เช่นกันแต่ไม่ได้มีการนำบริการที่ได้จากการพิจารณาผังโครงสร้างมารวมด้วย เพราะกระบวนการเหล่านั้นไม่เกี่ยวข้องกับแฟ้มข้อมูลโดยตรง ส่วนมากเป็นการทำงานกับแฟ้มข้อมูลอื่นเพื่อให้ได้ผลลัพธ์ที่ต้องการ

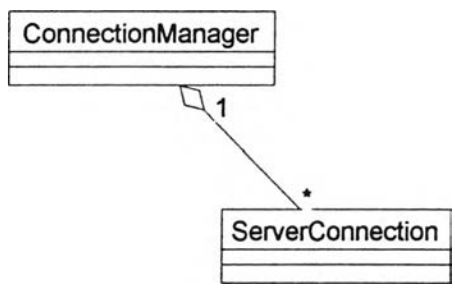
ในส่วนที่เป็นปัญหานี้จำเป็นต้องมีการศึกษาผังการไหลเพิ่มเติม จะพบว่าลักษณะของโปรแกรมเป็นเซิร์ฟเวอร์ซึ่งให้บริการโปรแกรมไคลเอนต์ตามการร้องขอ ซึ่งการทำงานของโปรแกรมนี้แบ่งเป็น 2 งานชัดเจน ซึ่งในการสร้างเป็นวัตถุก็สามารถสร้างวัตถุจากกระบวนการดำเนินงานได้ (สังเกตุได้ว่าที่ผังเชิงโครงสร้างก็เป็นการแบ่งตามการดำเนินงานชัดเจน) โดยวัตถุที่ใช้สำหรับการขอเลขที่นั่งสอบให้ชื่อว่าวัตถุ Reservation และวัตถุที่ใช้ยืนยันการสมัครให้ชื่อว่าวัตถุ Confirmation

4. ศึกษาผังการไหลของโปรแกรมเซิร์ฟเวอร์ (รูปที่ 2.8) เพื่อหาวัตถุเพิ่มเติม พบว่าเมื่อมีการติดต่อจากโปรแกรมไคลเอนต์ โปรแกรมเซิร์ฟเวอร์จึงจะทำการสร้างโปรเซสใหม่และให้บริการได้ เนื่องจากการสื่อสารจำเป็นต้องมีโปรโตคอลในการติดต่อชนิดเดียวกัน ดังนั้นจึงสามารถใช้วัตถุที่ได้ออกแบบไว้จากโปรแกรมไคลเอนต์ได้

วัตถุ CommProxy เป็นวัตถุที่ควบคุมการสื่อสารระหว่างโปรแกรมไคลเอนต์และโปรแกรมเซิร์ฟเวอร์ไว้ทั้งหมด ดังนั้นจึงสะดวกมากในการนำไปใช้งานเพราะมีรูปแบบที่ซับซ้อนน้อยลง และยังสะดวกในการปรับปรุงโปรโตคอล เพราะถูกแยกออกจากส่วนอื่นของโปรแกรมโดยเด็ดขาด

จากผังการไหลรูปที่ 2.8 ในช่วงแรกเป็นการรอคอยการติดต่อของโปรแกรมไคลเอนต์ การทำงานแบบนี้เป็นการทำงานของโปรแกรมเซิร์ฟเวอร์ทั่ว ๆ ไป ดังนั้นจึงสามารถนำคลาสของวัตถุที่เป็นโปรแกรมเซิร์ฟเวอร์ที่ได้มีการพัฒนามาแล้วมาใช้ได้

วัตถุที่เกี่ยวข้องในการทำงานแบบเซิร์ฟเวอร์นี้มี 2 วัตถุได้แก่ วัตถุ ConnectionManager และวัตถุ ServerConnection ดังรูปที่ 4.9



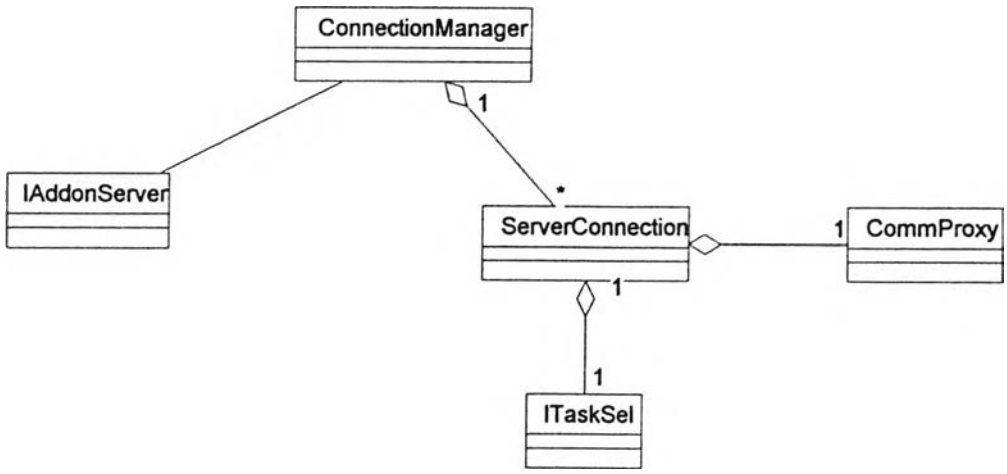
รูปที่ 4.9 แสดงวัตถุของโปรแกรมเซิร์ฟเวอร์ทั่วไป

วัตถุ ConnectionManager จะเป็นกระบวนการหลักที่ทำการรอคอยการติดต่อจากโปรแกรมไคลเอนต์ เมื่อได้รับการติดต่อก็จะทำการสร้างวัตถุ ServerConnection และส่งผ่านการติดต่อไปให้กับวัตถุ ServerConnection นี้ทำการให้บริการตามคำร้องขอของโปรแกรมไคลเอนต์ต่อไป

ในการนำมาใช้ในงานวิจัยนี้จำเป็นต้องมีการดัดแปลงเพิ่มเติมอีกเล็กน้อย เพื่อให้สามารถใช้งานได้ครบตามวัตถุประสงค์ โดยเพิ่มเติมวัตถุเข้าไปอีกดังรูปที่ 4.10

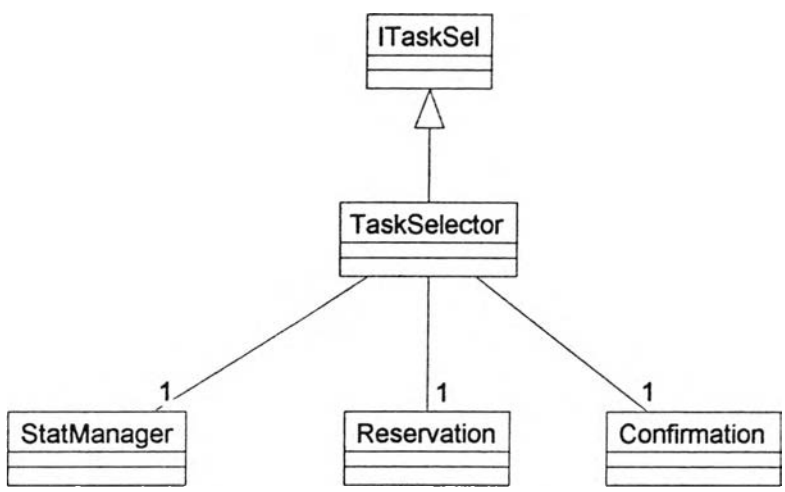
จากรูปที่ 4.10 ได้มีการเพิ่มเติมคลาสนามธรรม IAddonServer คลาสนามธรรม ITaskSel และวัตถุ CommProxy โดยคลาส IAddonServer จะถูกสืบทอดเพื่อสร้างวัตถุที่จะทำงานเฉพาะตอนเริ่มโปรแกรมเซิร์ฟเวอร์เท่านั้น ในที่นี้จากรูปที่ 2.8 คือขั้นตอนการตรวจสอบข้อมูลเลขที่นั่งสอบที่ว่างเป็นเลขแรก ส่วนคลาส ITaskSel จะถูกสืบทอดเพื่อสร้างวัตถุที่สามารถแบ่งแยกงานตามคำร้องขอของโปรแกรมไคลเอนต์ ทั้งนี้เพื่อให้สะดวกในการปรับปรุงแก้ไขเมื่อมีการเพิ่มเติมบริการให้กับโปรแกรมเซิร์ฟเวอร์ ส่วน

วัตถุ CommProxy จะเกิดขึ้นเมื่อได้รับการส่งต่อการติดต่อจากโปรแกรมไคลเอนต์แล้ว เพื่อเป็นช่องทางการติดต่อกับวัตถุ ServerConnection และโปรแกรมไคลเอนต์



รูปที่ 4.10 แสดงวัตถุของโปรแกรมเซิร์ฟเวอร์ที่ใช้

เมื่อพิจารณาขั้นตอนการทำงานจากผังเชิงโครงสร้าง (รูปที่ 2.7) และผังการไหล (รูปที่ 2.8) พบว่าการให้บริการของโปรแกรมเซิร์ฟเวอร์นี้แบ่งเป็น 2 ส่วนชัดเจน กล่าวคือ ส่วนแรกเป็นการขอเลขที่นั่งสอบและเขตการสอบ ส่วนที่สองเป็นการยืนยันการรับสมัครและบันทึกข้อมูลผู้สมัครลงแฟ้มข้อมูล TM13 บนเครื่องเซิร์ฟเวอร์ ซึ่งได้สร้างวัตถุเพื่อแบ่งการให้บริการคือวัตถุ TaskSelector ดังรูปที่ 4.11

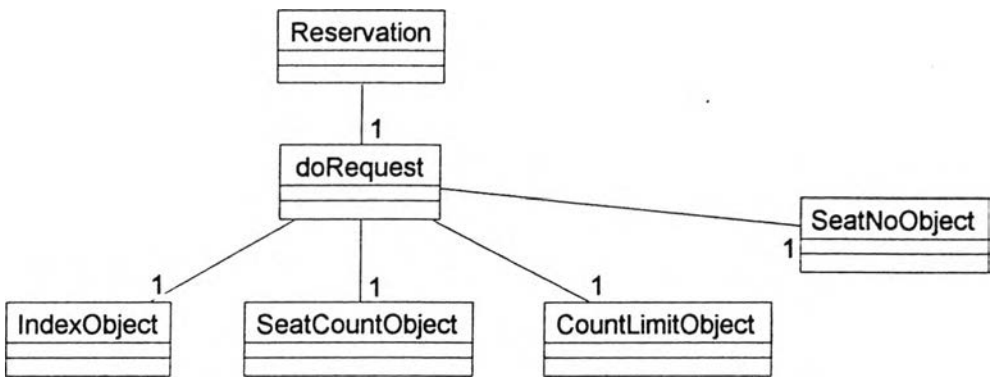


รูปที่ 4.11 แสดงวัตถุ TaskSelector สำหรับแบ่งการให้บริการ

จากรูปที่ 4.11 ได้มีการเพิ่มเติมวัตถุ StatManager เข้าไปภายหลังเพื่อให้บริการการตรวจสอบการติดต่อระหว่างโปรแกรมเซิร์ฟเวอร์และโปรแกรมไคลเอนต์ และยังสามารถตรวจสอบจำนวนการให้บริการของแต่ละไคลเอนต์ได้อีกด้วย

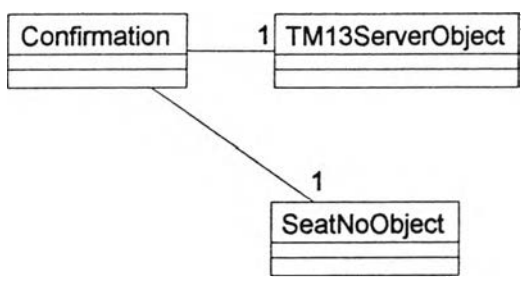
ในส่วนการขอเลขที่นั่งสอบ และเขตการสอบ ได้สร้างวัตถุขึ้นมาเพื่อทำงานตามขั้นตอนต่าง ๆ ที่จำเป็น ได้แก่วัตถุ Reservation จากขั้นตอนการทำงานพบว่า ช่วงแรกเป็นการขอเขตการสอบที่ว่าง ซึ่งจะ

มีการใช้งานแฟ้ม Count Limit Table และ Seat Count Table ในขั้นตอนนี้จำเป็นต้องมีการป้องกันการทำงานพร้อมกันของกระบวนการที่ให้บริการ ดังนั้นจึงต้องมีการจัดลำดับการขอบริการโดยได้สร้างวัตถุตัวกลางขึ้นมาวัตถุหนึ่งให้ชื่อว่า doRequest ทำหน้าที่จัดลำดับการขอเขตการสอบที่ว่าง โดยใช้คำสงวน synchronized และ static ของภาษาจาวาในการนิยามฟังก์ชัน คำสงวนนี้จะบังคับให้มีการทำงานที่กำหนดได้ทีละกระบวนการเดียว ดังนั้นในการขอเขตการสอบที่ว่างจึงจะไม่มีการทำงานพร้อมกัน นอกจากนี้ในการขอเลขที่นั่งสอบก็จำเป็นต้องป้องกันการแย่งกันทำงานของกระบวนการเช่นกัน จึงใช้เทคนิคเดียวกันนี้ในการจัดลำดับการใช้งาน โดยฟังก์ชันนี้จะถูกสร้างไว้ในวัตถุ doRequest แต่การขอเลขที่นั่งสอบนั้นถ้าผู้สมัครทำการสมัครหมวดวิชาและสนามสอบที่ไม่ตรงกัน ก็สามารถของเลขที่นั่งสอบพร้อมกันได้ ดังนั้นเพื่อเป็นการเพิ่มความสามารถของการให้บริการ ฟังก์ชันขอเลขที่นั่งสอบนี้จึงถูกแบ่งออกเป็น 28 ฟังก์ชัน (เนื่องจากมี 7 เขตการสอบ และ 4 หมวดวิชา) ที่สามารถทำงานได้พร้อมกันดังรูปที่ 4.12



รูปที่ 4.12 แสดงวัตถุ Reservation และวัตถุที่เกี่ยวข้อง

ในส่วนที่สองเป็นการยืนยันการสมัคร และบันทึกข้อมูลผู้สมัคร ซึ่งจะมีการบันทึกข้อมูลผู้สมัครลงแฟ้ม TM13 และมีการแก้ไขสถานะการสมัครในแฟ้ม Seat No Table การสร้างวัตถุสำหรับขั้นตอนการทำงานนี้ชื่อวัตถุ Confirmation ดังรูปที่ 4.13



รูปที่ 4.13 แสดงวัตถุ Confirmation และวัตถุที่เกี่ยวข้อง

จากรูปที่ 4.8 ยังมีวัตถุที่ยังไม่ได้กล่าวถึงดังนี้ วัตถุ StartServer เป็นวัตถุที่มีฟังก์ชัน main เพื่อใช้ในการเริ่มต้นโปรแกรมนี้ โดยจะทำหน้าที่สร้างวัตถุและตั้งค่าต่าง ๆ วัตถุ FPAddOn และวัตถุ FindFirstUnused เป็นวัตถุที่ทำหน้าที่ในการตรวจสอบข้อมูลเลขที่นั่งสอบที่ว่างเป็นเลขแรก ซึ่งจะทำเมื่อ

เริ่มต้นการทำงานของโปรแกรม และค่าที่ได้จะถูกใช้ตลอดการทำงานของโปรแกรม โดยจะถูกใช้ในส่วนของวัตถุ Reservation และวัตถุ Confirmation นอกจากนี้ยังมีวัตถุ TaskStatusSpliter ซึ่งเป็นวัตถุที่ช่วยแปลความหมายของค่าสถานะที่ได้จากวัตถุ TaskSelector ซึ่งถูกใช้โดยวัตถุ StatManager เพื่อให้ได้ค่าทางสถิติที่สนใจ รายละเอียดของคลาสสำหรับโปรแกรมเซิร์ฟเวอร์อยู่ในภาคผนวก ง.

จากการออกแบบพบว่าในโปรแกรมไคลเอนต์มีวัตถุดังต่อไปนี้

1. Client4 ใช้เป็นวัตถุเริ่มต้นการทำงาน
2. FrmMain ใช้เป็นหน้าจอรับสมัคร
3. dlgInfo ใช้เป็นหน้าจอแสดงความผิดพลาด
4. dlgLogin ใช้เป็นหน้าจอเริ่มเข้าโปรแกรม
5. SeqCntObject ใช้เป็นวัตถุที่นับลำดับของผู้สมัคร
6. Applicant ใช้เป็นวัตถุที่ดำเนินการรับสมัคร
7. TM13ClientObject ใช้เป็นวัตถุเก็บข้อมูลผู้สมัคร
8. FacultyChoiceObject ใช้เป็นวัตถุให้ค่าวิชาที่ใช้สอบในแต่ละคณะที่เลือก
9. CrashObject ใช้สำหรับตรวจสอบวิชาที่สอบว่ามีวันสอบตรงกันหรือไม่
10. CommProxy ใช้เป็นตัวแทนในการติดต่อกับโปรแกรมเซิร์ฟเวอร์
11. IEncode เป็นคลาสนามธรรมสำหรับใช้กับวัตถุ CommProxy
12. CheckSum เป็นวัตถุที่คำนวณค่า checksum
13. CheckSum2 เป็นวัตถุที่ใช้คำนวณค่า checksum และถูกใช้โดยวัตถุ CommProxy
14. TM3Printer ใช้เป็นวัตถุในการพิมพ์บัตรประจำตัวผู้สอบ
15. PrintTM13 ใช้เป็นวัตถุในการพิมพ์บัตรประจำตัวผู้สอบ
16. PrinterStream เป็นวัตถุช่วยในการติดต่อเครื่องพิมพ์
17. ExamplaceObject เป็นวัตถุที่ให้ชื่อของเขตการสอบ ใช้สำหรับพิมพ์บัตรประจำตัวผู้สอบ
18. Util เป็นวัตถุอรรถประโยชน์

19. EnrollProperties เป็นวัตถุใช้อ่านค่าตั้งต้นจากแฟ้ม Enroll.conf

20. AppendLOGFile เป็นวัตถุที่ทำการบันทึกข้อผิดพลาดในการทำงานลงแฟ้มข้อมูล

นอกจากนี้ยังพบว่าในโปรแกรมเซิร์ฟเวอร์มีวัตถุดังต่อไปนี้

1. StartServer ใช้เป็นวัตถุเริ่มต้นการทำงาน

2. ConnectionManager เป็นวัตถุที่รอการติดต่อจากโปรแกรมไคลเอนต์

3. ServerConnection เป็นวัตถุที่ทำการให้บริการแก่โปรแกรมไคลเอนต์ที่ร้องขอมา

4. IAddonServer เป็นคลาสนามธรรมสำหรับใช้กับวัตถุ ConnectionManager

5. FPAddOn เป็นวัตถุที่ใช้หาค่าดรรชนีของข้อมูลเลขที่นั่งสอบที่ว่าง

6. FindFirstUnused เป็นวัตถุที่ใช้หาค่าดรรชนีของข้อมูลเลขที่นั่งสอบที่ว่าง

7. CommProxy ใช้เป็นตัวแทนในการติดต่อกับโปรแกรมเซิร์ฟเวอร์

8. IEncode เป็นคลาสนามธรรมสำหรับใช้กับวัตถุ CommProxy

9. CheckSum เป็นวัตถุที่คำนวณค่า checksum

10. CheckSum2 เป็นวัตถุที่ใช้คำนวณค่า checksum และถูกใช้โดยวัตถุ CommProxy

11. ITaskSel เป็นคลาสนามธรรมสำหรับใช้กับวัตถุ ServerConnection

12. TaskSelector เป็นวัตถุที่แบ่งการให้บริการประเภทต่าง ๆ ให้แก่วัตถุที่ให้บริการนั้น

13. StatManager เป็นวัตถุที่ให้บริการด้านสถิติเบื้องต้น

14. Reservation เป็นวัตถุที่ให้บริการการเลือกเขตการสอบและการขอเลขที่นั่งสอบ

15. Confirmation เป็นวัตถุที่ให้บริการยืนยันการรับสมัครและจัดเก็บข้อมูลผู้สมัคร

16. doRequest เป็นวัตถุที่จัดการลำดับการใช้งานวัตถุที่ไม่ต้องการให้ใช้งานพร้อมกัน

17. IndexObject เป็นวัตถุที่จัดการข้อมูลของแฟ้มดรรชนี

18. SeatCountObject เป็นวัตถุที่จัดการข้อมูลของแฟ้มจำนวนผู้สมัครในแต่ละเขตการสอบ

19. CountLimitObject เป็นวัตถุที่จัดการข้อมูลของแฟ้มจำนวนจำกัดของผู้สมัครในแต่ละเขตการสอบ

20. SeatNoObject เป็นวัตถุที่จัดการข้อมูลของแฟ้มเลขที่นั่งสอบ

21. TM13ServerObject เป็นวัตถุที่จัดการข้อมูลของแฟ้มข้อมูลผู้สมัคร

22. Util เป็นวัตถุอรรถประโยชน์

23. EnrollProperties เป็นวัตถุใช้อ่านค่าตั้งต้นจากแฟ้ม Enroll.conf

24. TaskStatusSpliter เป็นวัตถุที่ช่วยในการอ่านค่าสถิติ

หลังจากนั้นก็นำผลที่ได้จากการออกแบบไปทำการสร้างโปรแกรมโดยใช้ภาษาจาวา ดังตัวอย่างโปรแกรมในภาคผนวก จ. และภาคผนวก ฉ. และได้นำโปรแกรมที่สมบูรณ์ไปผ่านการแปลโปรแกรม แล้วนำไปใช้ในการรับสมัครได้ผลดี