การเชื่อมต่อองค์ประกอบซอฟต์แวร์แบบพลวัต

Dynamic Component Composition

กิตติชาติ รัตนโกมุท และ พรศิริ หมื่นไชยศรี

คณะวิศวกรรมศาสตร์ ภาควิศวกรรมคอมพิวเตอร์

โทร. 218-6992

E-Mail:g41krt@cpu.cp.eng.chula.ac.th

Pornsiri.Mu@chula.ac.th

บทคัดย่อ

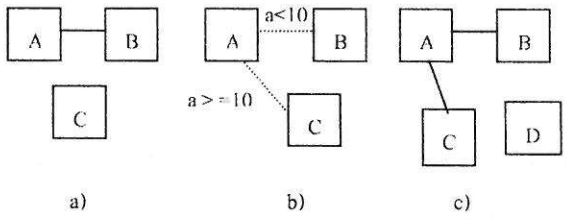
การสร้างซอฟต์แวร์ขึ้นจากองค์ประกอบซอฟต์แวร์ที่มีลักษณะการเชื่อมต่อแบบพลวัตคือการนำองค์ประกอบแต่ละตัวมาเชื่อมต่อกันเองคล้ายกับการสร้างพันธะของอะตอม โดยองค์ประกอบจะเชื่อมต่อกันเองเมื่อองค์ประกอบนั้นมีคุณสมบัติตรงกับกฎของการเชื่อมต่อแทนการกำหนดการเชื่อมต่อโดยผู้พัฒนาโปรแกรม ทั้งนี้เพื่อให้ซอฟต์แวร์นั้นมีคุณลักษณะที่สามารถเปลี่ยนแปลงและเพิ่มเติมองค์ประกอบขณะโปรแกรมทำงานอยู่ได้ การสร้างซอฟต์แวร์ด้วยการเชื่อมต่อแบบพลวัตนี้จะทำให้เกิดประโยชน์ในแง่การดูแลรักษา(Maintenance) การปรับแต่งระบบ(Configuration) และการเฝ้าดูระบบ(Monitoring) ในงานวิจัยนี้จะนำเสนอการออกแบบส่วนประกอบของการเชื่อมต่อและกฎที่ใช้ในการเชื่อมต่อองค์ประกอบแต่ละตัวเข้าด้วยกันพร้อมทั้งยกตัวอย่างโปรแกรมจำลองเครือข่ายไหลเพื่อแสดงให้เห็นถึงการนำไปใช้ขององค์ประกอบที่มีการเชื่อมต่อแบบพลวัตดังกล่าว

1.บทนำ

ในปัจจุบันซอฟต์แวร์มีกลไกการทำงานที่ซับซ้อนมากขึ้น เนื่องจากมีการปรับปรุงพัฒนาอยู่ตลอดเวลาทั้งนี้เพื่อให้สามารถใช้งานได้ถูกต้องตรงกับสถานการณ์ ซอฟต์แวร์จึงมีการพัฒนาให้มีลักษณะที่ยืดหยุ่นมากขึ้นเพื่อให้ง่ายต่อการเปลี่ยนแปลงและแก้ไข การพัฒนาซอฟต์แวร์โดยใช้ องค์ประกอบ(Component) ก็เป็นวิธีแก้ปัญหาอย่างหนึ่ง องค์ประกอบซอฟต์แวร์คือกลุ่มของคำสั่งและชุดข้อมูลของซอฟต์แวร์ ซึ่งสามารถนำมาประกอบเข้ากับองค์ประกอบอื่นเพื่อให้ซอฟต์แวร์ทำงานตามต้องการได้

ในระยะที่ผ่านมาได้มีการศึกษาวิจัยและคิดค้นผลิตภัณฑ์ที่เกี่ยวข้องกับการพัฒนาซอฟต์แวร์ในลักษณะองค์ประกอบ มากมายอาทิเช่น การพัฒนาซอฟต์แวร์แบบPlug&Play[1] งานวิจัยนี้ได้มีการออกแบบกฎในการเชื่อมโดยใช้ส่วนเชื่อมประสาน(Interface) เพื่อให้การเชื่อมต่อนั้นสามารถทำได้สะดวก นอกจากนี้ยังมี Java Bean™ ซึ่งเป็นมาตรฐานการเชื่อมต่อองค์ประกอบที่สร้างโดยภาษา Java™ ซึ่งผู้ใช้สามารถเชื่อมองค์ประกอบต่างๆผ่านทางช่องทาง(Port) โดยใช้ เครื่องมือ(Tool)ในการพัฒนาเช่น Java Studio™ เป็นต้น

เนื่องจากลักษณะการเชื่อมต่อในงานวิจัยและผลิตภัณฑ์ที่กล่าวมานั้นจะมีการเชื่อมต่อระหว่างองค์ประกอบเป็นแบบคงที่คือไม่มีการเปลี่ยนแปลงการเชื่อมต่อขณะที่โปรแกรมทำงานอยู่ ทำให้ซอฟต์แวร์นั้นไม่มีความยืดหยุ่นกล่าวคือ ผู้พัฒนาไม่สามารถเปลี่ยนแปลงการเชื่อมต่อระหว่างองค์ประกอบต่างๆขณะโปรแกรมทำงานอยู่ซึ่งทำให้มีความไม่สะดวกในการพัฒนาโปรแกรมที่มีการเพิ่มจำนวนขององค์ประกอบหรือ มีการเลือกใช้งานรูปแบบขององค์ประกอบที่ไม่ได้กำหนดไว้ก่อนเป็นต้น ยกตัวอย่างเช่น การเชื่อมต่อขององค์ประกอบ A B และ C ตามรูปที่ 1a เมื่อ A เชื่อมต่อกับ B แล้ว A ยังคงต้องต่ออยู่กับ B จนสิ้นสุดการทำงานของโปรแกรม หรือ ถ้าต้องการเชื่อม A กับ C จะต้องมีการกำหนดการเชื่อมต่อใหม่โดยผู้พัฒนาอีกครั้ง



รูปที่ 1 ตัวอย่างการเชื่อมต่อขององค์ประกอบ a) แบบการเชื่อมต่อคงที่ b) แบบการเชื่อมต่อแบบพลวัตที่มีการกำหนดตามเงื่อนไข c) การทำงานขององค์ประกอบที่การเชื่อมต่อแบบพลวัตที่มีการกำหนดตามเงื่อนไขขณะทำงานเมื่อเพิ่มองค์ประกอบ D

เนื่องด้วยปัญหาที่เกิดขึ้นจากการเชื่อมต่อแบบคงที่นั้น จึงมีงานวิจัยแนวทางการพัฒนาการเชื่อมต่อในลักษณะพลวัต อาทิเช่น งานวิจัย[2]ได้กล่าวถึงการทำตัวเชื่อมแบบพลวัต เพื่อให้การเชื่อมต่อเปลี่ยนตามเงื่อนไขที่กำหนดไว้โดยใช้ภาษา IDL (Interface Define Language) หรือในงานวิจัย[3]มีการออกแบบตัวเชื่อม (Connector) โดยให้มีคุณลักษณะ(Behavior)และสถานะ(Status)เพื่อจัดการการเชื่อมต่อระหว่าง

องค์ประกอบ ซึ่งถึงแม้การเชื่อมต่อจะเป็นไปในลักษณะพลวัตแต่ในการใช้งานจะต้องทราบองค์ประกอบในระบบที่แน่นอนทำให้ไม่สามารถที่จะพัฒนาระบบที่ไม่ทราบองค์ประกอบที่แน่นอนได้ ดังรูปที่ 1b เราสามารถสร้างการเชื่อมต่อระหว่างองค์ประกอบ A B และ C ตามเงื่อนไขได้แต่การกำหนดนั้นเป็นการกำหนดก่อนโปรแกรมทำงาน ทำให้โปรแกรมไม่สามารถเชื่อมต่อองค์ประกอบ D ดังรูปที่ 1c ในขณะที่โปรแกรมทำงานอยู่ได้

ในงานวิจัยนี้ได้เสนอการพัฒนาซอฟแวร์องค์ประกอบในลักษณะใหม่โดยองค์ประกอบที่ออกแบบนั้นสามารถที่จะเชื่อมต่อกันเองได้ขณะทำงานโดยผู้พัฒนาไม่ต้องทำการเชื่อมต่อเอง การเชื่อมต่อนั้นจะมีลักษณะเป็นพลวัตกล่าวคือสามารถเปลี่ยนแปลงการเชื่อมต่อได้ตามกฎและเงื่อนไขที่วางเอาไว้ ผู้พัฒนาจะเป็นผู้กำหนดการเชื่อมต่อและคุณสมบัติขององค์ประกอบ การทำงานของโปรแกรมจะคล้ายกับการจับกันของอะตอมซึ่งประโยชน์ที่ได้จากระบบดังกล่าวคือ เราสามารถที่จะปรับเปลี่ยนหรือเพิ่มเติมองค์ประกอบขณะทำงานอยู่ได้ โดยโปรแกรมไม่จำเป็นต้องหยุดทำงาน นอกจากนี้เราสามารถหาข้อผิดพลาดของระบบได้ง่ายเนื่องจากสามารถเห็นการเชื่อมต่อในขณะนั้นๆได้ทำให้เราสามารถตรวจสอบความถูกต้องและดำเนินการได้ทันที ในบทความนี้ส่วนแรกจะกล่าวถึงการออกแบบการเชื่อมต่อแบบพลวัต(Dynamic Composition) และในส่วนที่สองจะเป็นตัวอย่างโปรแกรมที่พัฒนาขึ้นโดยใช้เทคนิควิธีดังกล่าว พร้อมทั้งเปรียบเทียบการเชื่อมต่อแบบพลวัตกับแนวความคิดอื่นๆที่มีลักษณะคล้ายกัน

2. ภาพรวม

ในส่วนนี้เราจะกล่าวถึงการออกแบบระบบงานที่ใช้ในการสร้างองค์ประกอบและระบบที่นำองค์ประกอบที่มีการเชื่อมต่อแบบพลวัตไปใช้ อีกครั้งกล่าวถึงขั้นตอนในการสร้างและใช้องค์ประกอบ

2.1 ระบบที่ใช้ในการสร้างองค์ประกอบและระบบที่นำองค์ประกอบที่มีการเชื่อมต่อแบบพลวัตไปใช้

เนื่องจากองค์ประกอบที่ใช้การเชื่อมต่อแบบพลวัตนี้มีลักษณะพิเศษ ในงานวิจัยจึงมีการออกแบบระบบเพื่อรองรับการสร้างองค์ประกอบและนำองค์ประกอบแบบพลวัตไปใช้

2.1.1 ระบบ RDCS (Runtime Dynamic Component Composition System) คือ ระบบที่ใช้ในการควบคุมและเฝ้าดูการทำงานขององค์ประกอบ ผู้ดูแลระบบสามารถที่จะใช้ระบบ RDCS ในการดำเนินงานต่อไปนี้

- เพิ่ม หรือ ลบองค์ประกอบที่กำลังทำงานอยู่ในระบบ
- หยุด หรือ เริ่มการทำงานขององค์ประกอบแต่ละตัว
- ปรับแต่งองค์ประกอบเพื่อให้พร้อมนำไปใช้งาน
- กำหนดวิธีในการคัดเลือกเพื่อเชื่อมต่อ ในกรณีองค์ประกอบที่ร้องขอบริการนั้นสามารถที่จะเชื่อมต่อกับองค์ประกอบให้บริการได้หลายตัว โดยสามารถจะตั้งค่าให้เป็นการเลือกแบบสุ่ม หรือ กำหนดสิทธิให้แก่องค์ประกอบเป็นตัวเลือกอันดับแรกในการเชื่อมต่อก็ได้
- เฝ้าดูการเชื่อมต่อขององค์ประกอบ โดยผู้ดูแลระบบสามารถที่จะดูการเชื่อมต่อขององค์ประกอบในช่วงขณะใดก็ได้ โดยระบบจะแสดงออกมาในรูปแบบตัวอักษร โดยผู้ดูแลระบบอาจจะนำข้อมูลนี้ไปใช้ในการปรับปรุงโปรแกรมหรือหาข้อผิดพลาดได้

ระบบ RDCS นอกจากจะอำนวยความสะดวกในการควบคุมและเฝ้าดูองค์ประกอบแล้ว ระบบ RDCS ยังเป็นตัวที่จัดการการเชื่อมต่อขององค์ประกอบด้วย โดยระบบ RDCS จะเชื่อมองค์ประกอบที่ร้องขอบริการกับองค์ประกอบที่ให้บริการและในกรณีที่สามารถมีผลลัพธ์ในการเชื่อมต่อมากกว่าหนึ่ง ระบบ RDCS จะเป็นตัวตัดสินการเชื่อมต่อนั้น

2.1.2 ระบบ DDCS (Development of Dynamic Component System) คือ ระบบที่อำนวยความสะดวกในการพัฒนาองค์ประกอบที่ใช้การเชื่อมต่อแบบพลวัต โดยผู้พัฒนาจะสามารถใช้ระบบ DDCS พัฒนาองค์ประกอบจนองค์ประกอบนั้นพร้อมใช้งานได้ในระบบ RDCS ในระบบ DDCS นั้นจะมองว่าองค์ประกอบหนึ่งเกิดการนำเอาส่วนประกอบย่อยมารวมกัน โดยผู้พัฒนาสามารถที่จะเก็บรูปแบบของส่วนประกอบย่อยเอาไว้ในไลบารี่ได้เพื่อนำไปใช้ในการพัฒนาองค์ประกอบอื่นๆต่อไป ระบบ DDCS จะมีเครื่องมือในการพัฒนาดังต่อไปนี้

- การทำงานในส่วนบริการและร้องขอบริการขององค์ประกอบ
- ส่วนข้อมูลขององค์ประกอบ
- ส่วนติดต่อกับผู้ใช้ขององค์ประกอบ
- ความสัมพันธ์ขององค์ประกอบกับองค์ประกอบอื่นๆ
- การสร้างและออกแบบกลไกภายในองค์ประกอบ

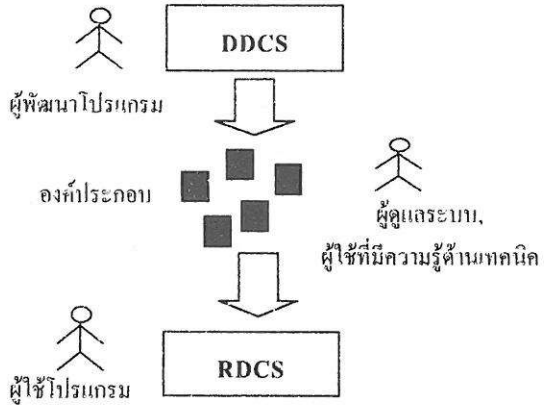
ระบบ DDCS เป็นโปรแกรมสำหรับพัฒนาโดยเฉพาะซึ่งมีลักษณะคล้ายโรงงานผลิต โดยผู้ใช้องค์ประกอบสามารถนำองค์ประกอบไปใช้งานได้ทันที โดยไม่ต้องมีระบบ DDCS ในครั้งต่อ

2.2 ขั้นตอนการสร้างและใช้องค์ประกอบ

ในงานวิจัยนี้จะแบ่งวงจรการพัฒนาและใช้งานองค์ประกอบออกเป็น 3 ส่วน โดยจะเริ่มตั้งแต่การสร้างองค์ประกอบจนถึงผู้ใช้ใช้ซอฟต์แวร์ที่สร้างจากองค์ประกอบที่ใช้การเชื่อมต่อแบบพลวัตดังในรูปที่ 2 โดยมีรายละเอียดของแต่ละส่วนดังนี้

1. ผู้พัฒนาใช้ระบบ DDCS เพื่อสร้างองค์ประกอบแบบต่างๆ
2. ผู้ดูแลระบบ ผู้ใช้ที่มีความสามารถ หรือ สคริปต์ที่มีการกำหนดไว้ล่วงหน้า เลือกองค์ประกอบที่จะใช้เข้าสู่ระบบ RDCS เพื่อใช้องค์ประกอบทำงาน

3. ผู้ใช้ใช้โปรแกรมที่เกิดจากการทำงานขององค์ประกอบต่างๆ ที่ทำงานร่วมกันในระบบ RDCS



รูปที่ 2 ขั้นตอนการสร้างและใช้องค์ประกอบที่มีการเชื่อมต่อแบบพลวัต

3. รูปแบบขององค์ประกอบ

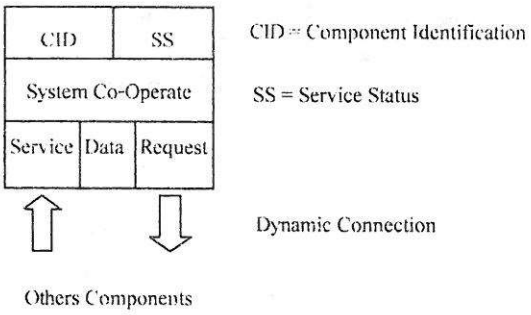
ในส่วนนี้จะกล่าวถึงการออกแบบองค์ประกอบ อาทิเช่น การออกแบบส่วนประกอบภายในองค์ประกอบและการออกแบบกฎการเชื่อมต่อเพื่อรองรับการเชื่อมต่อแบบพลวัต

3.1 ส่วนประกอบขององค์ประกอบ

เพื่อให้องค์ประกอบสามารถสนับสนุนการเชื่อมต่อแบบพลวัตได้นั้นจึงมีการออกแบบส่วนประกอบต่างๆที่อยู่ในองค์ประกอบดังในรูปที่ 3 โดยจะมีรายละเอียดของแต่ละส่วนประกอบดังนี้

1. ตัวบ่งชี้องค์ประกอบ (Component Identifier : CID) เพื่อแสดงความแตกต่างระหว่างองค์ประกอบต่างๆในขณะทำงาน
2. สถานะการให้บริการ (Service Status : SS) เพื่อบ่งบอกถึงสถานะการบริการขององค์ประกอบซึ่งจะมี 3 สถานะ คือ 1) พร้อมให้บริการ 2)ร้องขอบริการ 3)กำลังทำงานหรือซ่อมแซม
3. ส่วนติดต่อกับระบบ RDCS (System Co-Operate) เป็นส่วนรับข้อมูลจากระบบRDCS หรือส่งข้อมูลไปสู่ระบบ RDCS เพื่อใช้ในการจัดการองค์ประกอบของระบบและการจัดการการเชื่อมต่อ
4. ส่วนข้อมูล (Data) เป็นส่วนเก็บข้อมูลของตัวองค์ประกอบ
5. ส่วนให้บริการ (Service) เป็นส่วนเก็บรูปแบบและการทำงานในด้านให้บริการขององค์ประกอบ
6. ส่วนร้องขอบริการ (Request) เป็นส่วนเก็บรูปแบบและการทำงานในด้านร้องขอบริการขององค์ประกอบ

จากการออกแบบองค์ประกอบให้ประกอบด้วยส่วนย่อยทั้ง 6 ส่วนนั้นผู้วิจัยได้คำนึงถึงการใช้งานขององค์ประกอบและการเปลี่ยนแปลงปรับปรุงส่วนประกอบต่างๆที่อยู่ในองค์ประกอบเพื่อให้มีความชัดเจนในการออกแบบและสามารถเปลี่ยนแปลงเฉพาะส่วนได้ง่าย



รูปที่ 3 ส่วนประกอบภายในองค์ประกอบที่มีการเชื่อมต่อแบบพลวัต

3.2 กฎในการเชื่อมต่อองค์ประกอบ

เมื่อเรานำองค์ประกอบเข้าสู่ระบบ RDCS องค์ประกอบจะเริ่มทำการเชื่อมต่อกันตามเงื่อนไขและรูปแบบที่ผู้พัฒนาออกแบบไว้เพื่อให้ระบบทำงานตามที่ต้องการ โดยในการที่องค์ประกอบจะเชื่อมกันได้นั้นจะเป็นไปตามกฎดังนี้

กฎข้อที่ 1 องค์ประกอบจะเชื่อมต่อกันได้เมื่อองค์ประกอบหนึ่งมีสถานะบริการเป็นให้บริการ(Service) และอีกองค์ประกอบหนึ่งมีสถานะการบริการเป็นสถานะร้องขอบริการ (Request)

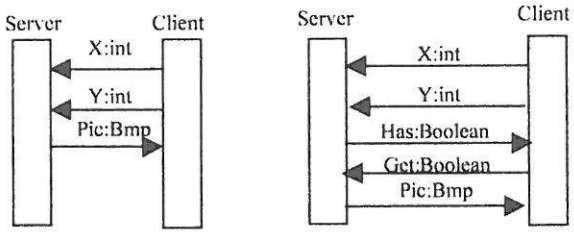
กฎข้อที่ 2 องค์ประกอบที่เชื่อมต่อนั้นจะต้องมีรูปแบบการบริการที่เข้ากันได้คือ มีชื่อบริการและการส่งพารามิเตอร์เหมือนกัน

กฎข้อที่ 3 ในกรณีที่มีองค์ประกอบที่มีคุณสมบัติตรงตามกฎข้อ 1 และ กฎข้อ 2 หลายตัวระบบRDCS จะทำการเลือกองค์ประกอบตามลำดับความสำคัญ โดยผู้นำองค์ประกอบเข้าสู่ระบบจะเป็นผู้กำหนดความสำคัญ

3.3 การสื่อสารระหว่างองค์ประกอบ

เมื่อองค์ประกอบ 2 องค์ประกอบเชื่อมต่อกันองค์ประกอบทั้งสองจะมีการแลกเปลี่ยนข้อมูลระหว่างกันโดยอาจจะมีการแลกเปลี่ยนหลายครั้งก็ได้ แต่มีเงื่อนไขอยู่ว่า องค์ประกอบที่ร้องขอบริการจะต้องส่งข้อมูลก่อน และ องค์ประกอบที่ให้บริการจะส่งข้อมูลกลับเมื่อสิ้นสุดการสื่อสาร รูปที่ 4 แสดงการสื่อสารระหว่างองค์ประกอบServer และ Client โดยรูปที่ 4a จะแสดงการสื่อสารอย่างน้อยที่สุดระหว่างองค์ประกอบโดย Client จะส่งค่าพารามิเตอร์ Xและ Y ให้กับ Server โดย Serverจะส่งค่าผลลัพธ์เป็นรูปภาพกลับมาให้ รูปที่ 4b แสดงการสื่อสารระหว่างองค์ประกอบในลักษณะไปกลับหลายครั้งโดย Client จะส่งพารามิเตอร์ X และ Y แก่ Server โดย Server จะส่งค่า Boolean กลับมาให้จากนั้นฝั่ง

Client จะดำเนินการต่อโดยส่งพารามิเตอร์ Get กลับไปโดยทางฝั่ง Server จะส่งรูปภาพกลับมาให้



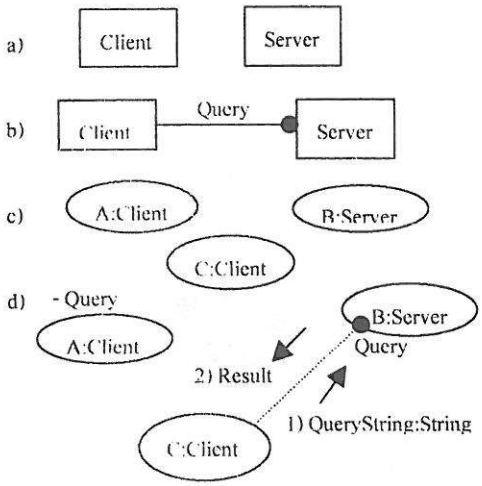
a) b)

รูปที่ 4 ตัวอย่างการติดต่อขององค์ประกอบ a)การติดต่อครั้งเดียว
b)การติดต่อแบบหลายครั้ง

3.4 สัญลักษณ์ที่ใช้ในการอธิบายการทำงานขององค์ประกอบ

เพื่อให้ผู้พัฒนาสามารถเข้าใจถึงการทำงานขององค์ประกอบในการเชื่อมต่อแบบพลวัตจึงมีการกำหนดสัญลักษณ์ขององค์ประกอบดังนี้

1. **ชนิดขององค์ประกอบ** ใช้รูปสี่เหลี่ยมและด้านในเขียนชนิดขององค์ประกอบนั้นเพื่อกำหนดชนิดขององค์ประกอบที่จะใช้ในระบบ ดังในรูปที่ 5a เป็นการกำหนดองค์ประกอบชนิด Client และ Server
2. **ความสัมพันธ์ระหว่างชนิดองค์ประกอบ** ใช้เส้นทึบโดยปลายด้านที่เป็นการให้บริการจะมีวงกลมอยู่ที่ปลายเพื่อแสดงความสัมพันธ์ระหว่างองค์ประกอบ ในกรณีที่มีลำดับความสัมพันธ์ขององค์ประกอบให้ระบุเลขลำดับไว้ใต้เส้น ดังในรูปที่ 5b เป็นการแสดงความสัมพันธ์ระหว่างองค์ประกอบ Client และ Server
3. **องค์ประกอบขณะทำงานอยู่** ใช้รูปวงกลมและด้านในเขียน ชื่อที่ตามด้วยชนิดขององค์ประกอบนั้น ในกรณีที่จะแสดงสถานะการบริการให้เขียน เครื่องหมาย + เพื่อแสดงถึงการร้องขอบริการ (Request) และเครื่องหมาย - แสดงการให้บริการ(Service) โดยในกรณีที่จะระบุชื่อของบริการให้เขียนต่อจากเครื่องหมายนั้น ถ้าต้องการแสดงถึงข้อมูลให้เขียนข้อมูลลงภายในสี่เหลี่ยมขอบมนและ ทำลูกศรชี้ไปยังองค์ประกอบนั้น ดังในรูปที่ 3c เป็นการกำหนดองค์ประกอบ A , C เป็นองค์ประกอบ Client และ B เป็นองค์ประกอบชนิด Server
4. **การเชื่อมต่อระหว่างองค์ประกอบ** ใช้เส้นประเพื่อแสดงการเชื่อมต่อและปลายด้านที่เชื่อมต่อกับองค์ประกอบให้เขียนวงกลมทึบไว้เพื่อแสดงการให้บริการขององค์ประกอบนั้น ในกรณีที่ต้องการระบุค่าที่ส่งผ่านการเชื่อมต่อให้เขียนไว้เหนือเส้นประพร้อมระบุทิศทางโดยใช้ลูกศร ในกรณีที่ต้องระบุลำดับให้เขียนไว้หน้าค่าของพารามิเตอร์นั้น ดังในรูปที่ 5d แสดงการเชื่อมต่อแบบพลวัตและการส่งข้อมูลถึงกันระหว่างองค์ประกอบ B และ C



รูปที่ 5 ตัวอย่างการใช้สัญลักษณ์เพื่อนำเสนอองค์ประกอบ

4. ตัวอย่าง

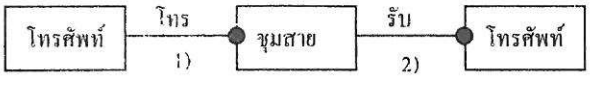
เพื่อให้เห็นถึงการทำงานขององค์ประกอบแบบพลวัตได้ชัดเจนในบทความนี้ได้ใช้ระบบจำลองเครือข่ายโทรศัพท์มาเป็นตัวอย่าง โดยระบบดังกล่าวจะประกอบด้วยเครื่องโทรศัพท์และชุมสายโทรศัพท์ โดยโทรศัพท์และชุมสายจะไม่ได้ถูกเชื่อมกันไว้ก่อนแต่จะเชื่อมกันเมื่อมีการสื่อสารระหว่างคู่สายเท่านั้น

4.1 การออกแบบการเชื่อมต่อ

เพื่อให้โปรแกรมสามารถทำงานได้ตามลักษณะดังกล่าวเราจึงกำหนดการทำงานเมื่อมีการเชื่อมต่อดังนี้

1. เครื่องต้นทางร้องขอบริการจากชุมสาย โดยส่งข้อมูลเลขปลายทาง
2. ชุมสายเชื่อมต่อกับเครื่องต้นทางและส่งข้อมูลบอกการรอแก่เครื่องต้นทาง
3. ชุมสายร้องขอบริการ โดยตั้งเงื่อนไขการเชื่อมต่อกับเครื่องโทรศัพท์ที่มีเบอร์ปลายทางตรงกับที่กำหนด
4. เครื่องปลายทางเชื่อมต่อกับชุมสาย พร้อมส่งข้อมูลบอกว่าพร้อมสำหรับการสื่อสาร
5. ชุมสายเชื่อมเครื่องปลายทางและต้นทางสื่อสารกัน
6. ถ้ามีฝ่ายใดฝ่ายหนึ่งวาง ชุมสายจะทำการตัดการเชื่อมต่อ
7. ชุมสายและเครื่องโทรศัพท์พร้อมที่จะทำงานต่อไป

เมื่อได้ทราบถึงการทำงานของระบบแล้วเราสามารถออกแบบการเชื่อมต่อระหว่างองค์ประกอบได้ดังรูปที่ 6



รูปที่ 6 แผนภาพความสัมพันธ์ระหว่างองค์ประกอบในระบบจำลองเครือข่ายโทรศัพท์

4.2 การออกแบบองค์ประกอบ

เมื่อเราได้ออกแบบการทำงานของระบบแล้ว เราก็จะออกแบบการทำงานขององค์ประกอบแต่ละตัวโดยในระบบจะมีองค์ประกอบดังนี้

4.2.1 องค์ประกอบโทรศัพท์

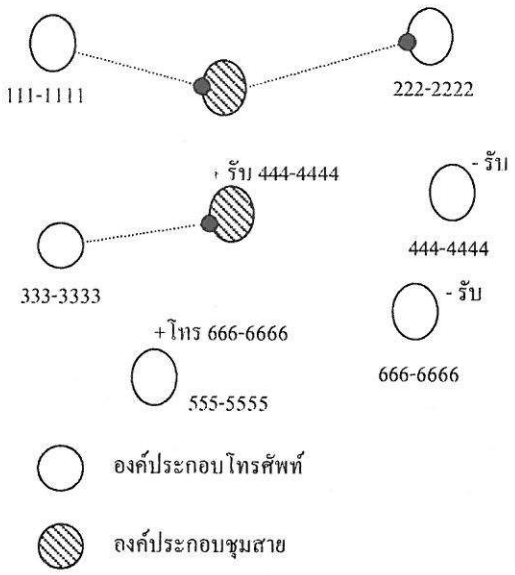
- ในสภาวะปรกติจะสามารถให้บริการ "รับ" กับชุมสาย
- เมื่อผู้ใช้ ใช้องค์ประกอบโทรศัพท์โทรหาโทรศัพท์อีกเครื่องหนึ่ง องค์ประกอบจะเปลี่ยนสถานะเป็นร้องขอ "โทร" กับชุมสาย
- เมื่อร้องขอ "โทร" กับชุมสาย องค์ประกอบจะส่งข้อมูล เบอร์โทรศัพท์ของเครื่องปลายทางไปให้ชุมสาย พร้อมรอรับข้อมูลผลลัพธ์จากองค์ประกอบชุมสาย
- เมื่อองค์ประกอบให้บริการ "รับ" องค์ประกอบจะรับข้อมูลเบอร์ต้นทางจากชุมสาย และจะส่งผลลัพธ์การรับโทรศัพท์ของผู้ใช้ว่ารับหรือไม่

4.2.2 องค์ประกอบชุมสาย

- ในสภาวะปรกติจะให้บริการ "โทร" กับโทรศัพท์
- เมื่อมีการให้บริการ "โทร" ชุมสายจะรับข้อมูลเบอร์ปลายทางจากนั้น ร้องขอบริการ "รับ" โดยมีเงื่อนไขการเชื่อมต่อว่าโทรศัพท์นั้นจะต้องมีเบอร์ตรงกับเบอร์ปลายทาง
- ในกรณีที่ไม่มีโทรศัพท์เครื่องใดมีเบอร์จะส่งผลลัพธ์เป็นเชื่อมต่อไม่สำเร็จ
- ในกรณีเจอโทรศัพท์ปลายทางจะส่งข้อมูลเครื่องต้นทางไปให้เครื่องปลายทาง
- ในกรณีที่เครื่องปลายทางส่งผลลัพธ์ไม่มีผู้รับสายชุมสายจะส่งข้อมูลติดต่อไม่สำเร็จกับเครื่องต้นทาง
- ในกรณีที่มีผู้รับสาย ชุมสายจะทำการเชื่อมการสื่อสารระหว่าง 2 เครื่องจนกว่าจะมีเครื่องใดวางสาย

4.3 การนำองค์ประกอบที่สร้างไปใช้งาน

หลังจากที่เรานำองค์ประกอบที่สร้างขึ้นมานั้นทำให้อยู่ในรูปแบบแลชฐานสอง เราก็จะนำองค์ประกอบที่สร้างขึ้นมานั้นนำไปใช้ในระบบ RDCS โดยในรูปที่ 7 จะเป็นการทำงานขณะหนึ่งของระบบจำลองเครือข่ายโทรศัพท์



รูปที่ 7 การทำงานขององค์ประกอบโทรศัพท์และชุมสายในระบบ RDCS

จากตัวอย่างในรูปที่ 7 จะเห็นการทำงานขององค์ประกอบโทรศัพท์ 6 ตัวที่มีเบอร์ต่างๆกันและองค์ประกอบชุมสาย 2 ตัวโดยชุมสายตัวแรก(ด้านบนของรูป)กำลังให้บริการคู่สายขององค์ประกอบโทรศัพท์เบอร์ 111-1111 กับเบอร์ 222-2222 อยู่ ส่วนองค์ประกอบชุมสายตัวที่ 2 (ด้านล่างของรูป) กำลังร้องขอบริการ "รับ" จากองค์ประกอบโทรศัพท์โดยมีเงื่อนไขว่าเบอร์ประจำตัวต้องเป็นเบอร์ 444-4444 เท่านั้น ส่วนในองค์ประกอบโทรศัพท์ 555-5555 นั้นกำลังจะร้องขอบริการ "โทร" อยู่แต่ปรากฏว่าไม่มีชุมสายใดว่างเพื่อบริการการร้องขอนั้น ส่วนในองค์ประกอบโทรศัพท์ที่ 444-4444 และ 666-6666มีสถานะพร้อมจะให้บริการรับเมื่อมีการร้องขอจากชุมสาย

เมื่อระบบที่เราพัฒนาขึ้นมาแล้วนั้นสามารถทำงานได้แล้ว และเราจำเป็นต้องมีการเพิ่มองค์ประกอบอาทิเช่น องค์ประกอบโทรศัพท์ เราสามารถเพิ่มเติมได้ทันทีโดยไม่ต้องหยุดระบบ หรือ ในกรณีที่ชุมสายมีไม่พอต่อความต้องการเราก็สามารถจะเพิ่มองค์ประกอบนั้นๆเข้าไปได้ในทันที

หรือในอนาคตนั้นได้มีการพัฒนาองค์ประกอบโทรศัพท์ที่สามารถส่งรูปขณะพูดคุยได้ ซึ่งองค์ประกอบโทรศัพท์รุ่นใหม่นี้จะสื่อสารได้ก็ต่อเมื่อมี ชุมสายที่รองรับและองค์ประกอบปลายทางรองรับด้วย เราก็สามารถจะปรับปรุงระบบได้โดยการเพิ่มองค์ประกอบแบบใหม่เข้าสู่ระบบทันที ซึ่งก็สามารถจะทำงานร่วมกับองค์ประกอบที่มีอยู่เดิมได้

5. งานวิจัยที่เกี่ยวข้อง

งานวิจัย [1] เป็นการวิจัยการทำองค์ประกอบของซอฟต์แวร์ให้สามารถเปลี่ยนแปลงได้ง่าย โดยองค์ประกอบที่ออกแบบนั้นจะมีลักษณะคือ 1)องค์ประกอบแต่ละตัวมีความเป็นอิสระต่อกัน 2) องค์

ประกอบแต่ละตัวไม่ยึดติดกับตำแหน่งหรือสถานที่ที่นำไปใช้อาทิเช่น เครื่องหรือ ตำแหน่งของหน่วยความจำ 3) สามารถทำงานร่วมกับองค์ประกอบอื่นที่มีความหลากหลายได้ 4) สามารถเปลี่ยนแปลงองค์ประกอบได้ง่าย โดยในงานวิจัยดังกล่าวได้ใช้การกำหนดตัวเชื่อมประสานเพื่อเป็นข้อกำหนดในการเชื่อมต่อกันขององค์ประกอบ โดยผู้พัฒนาสามารถเพิ่มเติมลักษณะพิเศษขององค์ประกอบด้วยการสืบทอดลักษณะของตัวเชื่อมประสานได้ ส่วนในการกำหนดตัวขององค์ประกอบจะกำหนดการบริการตัวเชื่อมประสานนั้น หรือ การร้องขอบริการตัวเชื่อมประสานนั้นโดยองค์ประกอบที่มีการบริการและการร้องขอบริการอันเดียวกันเท่านั้นที่จะเชื่อมต่อกันได้

งานวิจัย[6] เป็นงานวิจัยที่กล่าวถึงการกำหนดการเชื่อมต่อองค์ประกอบแตกต่างจากแบบพลวัตโดยกำหนดการเชื่อมต่อโดยใช้ภาษาในการอธิบายขั้นตอนการเชื่อมต่อโดยมีเงื่อนไขการเชื่อมต่อในลักษณะต่างๆ แต่ในภาษาดังกล่าวนั้นจำเป็นที่จะต้องทราบองค์ประกอบทุกตัวในระบบเพื่อที่จะกำหนดการเชื่อมต่อแบบพลวัตได้

งานวิจัย[3] เป็นงานวิจัยที่คิดค้นตัวเชื่อมที่ผู้พัฒนาสามารถกำหนดเงื่อนไขการทำงานได้อีกทั้งมีสถานะในการทำงานอีกด้วยเพื่อจัดการด้านการเชื่อมต่อขององค์ประกอบที่ต่ออยู่กับตัวเชื่อมซึ่งมีลักษณะการแก้ปัญหาคล้ายกับงานวิจัย[6]แต่การกำหนดการจัดการจะถูกบรรจุอยู่ในตัวเชื่อม

งานวิจัย [7] เป็นการใช้แผนภาพ FERD เพื่อแสดงการเชื่อมต่อกันระหว่างชนิดองค์ประกอบ ในงานวิจัยนั้นได้พัฒนาระบบที่ใช้ในการสร้างองค์ประกอบและระบบที่นำองค์ประกอบที่สร้างขึ้นมาไปใช้ เมื่อเราเชื่อมต่อองค์ประกอบ 2 ตัวเข้าด้วยกันแล้วองค์ประกอบจะเริ่มทำงานทันที

นอกจากนี้ยังมีทฤษฎีและรูปแบบระบบที่เกี่ยวข้องเช่น การออกแบบเชิงวัตถุ(Object Oriented Design) ใน [2] และ ระบบมัลติซิสเต็มเอเจนท์ (Multisystem Agent) ในงานวิจัย [4] และ [5] ในการออกแบบเชิงวัตถุนั้นคือ การมองโปรแกรมเป็นวัตถุ(Object)ต่างๆมาทำงานร่วมกัน โดยสามารถแบ่งวัตถุออกเป็นหมวดหมู่(Class)ที่มีความสัมพันธ์ระหว่างกันคล้ายกับการออกแบบโปรแกรมที่มีการเชื่อมต่อแบบพลวัต แต่ในการออกแบบการทำงานของโปรแกรมหรือในระดับวัตถุนั้น ในการออกแบบเชิงวัตถุจะมีการกำหนดวัตถุที่มีในโปรแกรมทั้งหมดพร้อมกับการเรียกใช้บริการระหว่างกันซึ่งมีลักษณะแน่นอน แต่ในการออกแบบโปรแกรมแบบพลวัตจะใช้วิธีการจำลองสถานการณ์ขณะหนึ่งเท่านั้นเพราะการเชื่อมต่อไม่ตายตัวและเปลี่ยนแปลงได้ ส่วนระบบมัลติซิสเต็มเอเจนท์คือระบบที่ประกอบไปด้วยโปรแกรมร้องขอบริการและโปรแกรมให้บริการ ซึ่งมีลักษณะคล้ายกับการร้องขอบริการขององค์ประกอบแบบพลวัต แต่ในการเชื่อมต่อแบบพลวัตนั้นจะอนุญาตให้องค์ประกอบมีชื่อบริการที่เหมือนกันได้ ซึ่งทำให้ผลลัพธ์ในการเชื่อมต่อมีมากกว่าหนึ่งซึ่งโปรแกรมแบบพลวัตจะมีกลไกในการคัดเลือกการเชื่อมต่อ ซึ่งแตกต่างจากระบบมัลติซิสเต็มเอเจนท์ที่ไม่อนุญาตให้มีชื่อบริการซ้ำกันและไม่มีระบบคัดเลือกการเชื่อมต่อ

นอกจากนี้ยังมีผลิตภัณฑ์ที่เกี่ยวข้องกับการพัฒนาแบบองค์ประกอบอาทิเช่น Visual Basic™ , Java Bean™ และ Visual Age for Java™ ซึ่งเป็นเครื่องมือพัฒนาที่ใช้ประโยชน์จากลักษณะขององค์ประกอบซอฟต์แวร์ซึ่งใน Java Bean™ และ Visual Age for Java™ นั้นผู้พัฒนาจะกำหนดการเชื่อมต่อระหว่างองค์ประกอบโดยใช้เส้นโยงระหว่างขั้วต่อ แต่การเชื่อมต่อนั้นเป็นในลักษณะคงที่ ส่วนใน Visual Basic™ จะใช้ภาษาเป็นตัวเชื่อมต่อโดยการเรียกใช้ วิธีการ(Method), คุณสมบัติ(Property) และ รับเหตุการณ์(Event) จากอีกองค์ประกอบหนึ่งซึ่งในการเชื่อมต่อนั้นอาจมีลักษณะเป็นพลวัต แต่การเชื่อมต่อที่เกิดขึ้นจะจำกัดเฉพาะองค์ประกอบที่มีอยู่ในโปรแกรมเท่านั้น ซึ่งผู้ใช้ไม่สามารถเปลี่ยนหรือเพิ่มองค์ประกอบในขณะใช้งานได้

6. สรุป

ในบทความนี้ได้นำเสนอการเชื่อมต่อแบบพลวัต โดยออกแบบให้องค์ประกอบนั้นสามารถเชื่อมต่อกันเองได้ โดยการเชื่อมต่อจะเกิดขึ้นเมื่อ 1) องค์ประกอบหนึ่งมีสถานะบริการเป็นให้บริการ(Service) อีกองค์ประกอบหนึ่งมีสถานะ บริการเป็นร้องขอบริการ (Request) 2) มีชนิดของบริการอันเดียวกัน 3) เป็นอันดับแรกในการกำหนดจากระบบในกรณีที่มีหลายองค์ประกอบที่ตรงตามข้อ 1 และ ข้อ2 โดยเมื่อการทำงานระหว่างองค์ประกอบสิ้นสุดลง องค์ประกอบจะสลายการเชื่อมต่อเอง

ในการใช้งานและพัฒนาองค์ประกอบนั้นจะใช้ระบบในการพัฒนา 2 ระบบ คือ 1)ระบบ DDCS ใช้ในการสร้างองค์ประกอบที่มีการเชื่อมต่อแบบพลวัต 2)ระบบ RDCS เป็นระบบที่นำองค์ประกอบที่พัฒนาจากระบบ DDCS มาใช้งาน

ข้อดีที่ได้จากการนำรูปแบบองค์ประกอบที่มีการเชื่อมต่อแบบพลวัตไปใช้คือ สามารถบำรุงรักษา(Maintainance) และปรับแต่ง(Configuration) ระบบได้ง่ายเนื่องจากสามารถเพิ่ม ลด หรือ ปรับแต่งองค์ประกอบต่างๆได้โดยไม่ต้องหยุดระบบขณะทำงาน และ ไม่ต้องนำเอาองค์ประกอบมาเชื่อมเข้าด้วยกัน นอกจากนี้ยังสามารถเฝ้าดูระบบ(Monitoring) เพื่อดูประสิทธิภาพและหาข้อผิดพลาดได้ง่ายเพราะสามารถแสดงข้อมูลการเชื่อมต่อในขณะต่างๆได้

เอกสารอ้างอิง

[1] F. Bronsard , D. Bryan , W. (Votek)Kozaczynski , E. S. Liongosari, J. Q.Ning, A. Olafsson and J. W.Wetterstrand, **"Toward Software Plug-and-Play"**, *SOFTWARE ENGINEERING NOTES ACM PRESS*, JUN 1997 pp.19-29

[2] G. Booch, **Object-Oriented Analysis and Design with Application**. California: Addison-Wesley, 1997.

[3] S. Ducasse and T. Richner, **"Executeable Connectors: Toward Reusable Design Elements"**, *SOFTWARE ENGINEERING NOTES ACM PRESS*, JAN 1997 pp.483-499

[4] W. Jiao and Z. Shi, **"A Dynamic Architecture for Multi-Agent Systems"** , *31 st International Conference on Technology of Object-Oriented Language and Systems*,Nanjing,China,September 1999.

[5] C. N. Linn, **"A Multi-Agent System for Cooperative Document Indexing and Querying in Distributed Environments"** , *1999 International Workshops on Parallel Processing* ,Wakamatsu,Japan,September 1999.

[6] J. Magee and J. Kramer, **"Dynamic Structure in Software Architectures"**, *Proceedings of the Fourth ACM SIGSOFT Symposium on the Foundations of Software Engineering*, v.21 n.6, Nov 1996 pp.3-14.

[7] P. Muenchaisri and Toshimi Minoura, **"Software Composition with Extended Entity-Relationship Diagrams"**,*Second Conference on Object-Oriented Technology and Systems*,Toronto,Canada,June 1996.