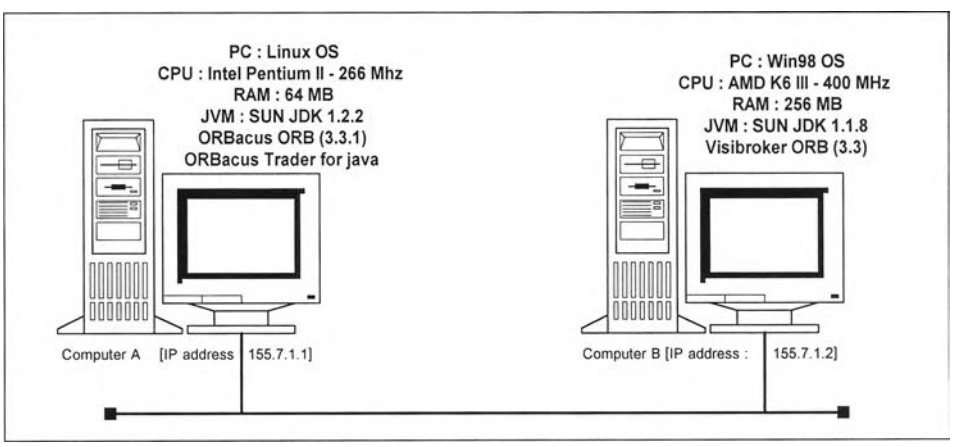
บทที่ 5
การทดสอบโปรแกรมพรีโพรเซสเซอร์และบริการเทรดเดอร์ที่รองรับการเรียกใช้บริการที่เท่าเทียมกัน

ในบทนี้จะกล่าวถึงรายละเอียดเกี่ยวกับวิธีการตลอดจนขั้นตอนที่ใช้ในการทดสอบกลไกส่วนขยายของคอร์บาที่ถูกพัฒนาขึ้นในงานวิจัยนี้ การทดสอบจะถูกแบ่งออกเป็น 2 ส่วนหลัก ส่วนแรกได้แก่การทดสอบการทำงานของโปรแกรมพรีโพรเซสเซอร์ที่ใช้ในการแทรกคำสั่งลงในโปรแกรมของผู้รับบริการตามรายละเอียดที่ได้กล่าวถึงไปแล้วในบทที่ 3 การทดสอบในส่วนนี้จะใช้การเปรียบเทียบผลการทำงานของโปรแกรมผู้รับบริการทั้งก่อนและหลังที่จะถูกแทรกคำสั่งโดยโปรแกรมพรีโพรเซสเซอร์ว่ามีการทำงานที่แตกต่างกันอย่างไร และโปรแกรมของผู้รับบริการที่ผ่านการแทรกคำสั่งการทำงานจะสามารถเรียกใช้งานบริการอื่นที่มีความสามารถเท่าเทียมกันกับบริการที่ต้องการเมื่อออร์บของคอร์บาไม่สามารถค้นหาข้อมูลอ้างอิงของบริการที่ต้องการได้หรือไม่ การทดสอบส่วนที่ 2 นั้นจะเน้นการทดสอบไปยังบริการเทรดเดอร์ที่ผ่านการแก้ไขโปรแกรมให้สามารถนำข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการมาใช้ในกระบวนการอิมพอร์ตข้อเสนอบริการ (ดังรายละเอียดในบทที่ 4) ว่าบริการเทรดเดอร์ใหม่นี้จะช่วยเพิ่มโอกาสในการค้นพบข้อเสนอบริการตามความต้องการของผู้รับบริการได้อย่างไร

สภาพแวดล้อม (Environment) ของเครื่องคอมพิวเตอร์และซอฟต์แวร์ต่างๆที่ใช้ในการทดสอบกลไกส่วนขยายของคอร์บาตามงานวิจัยนี้เป็นดังรูปที่ 5.1



รูปที่ 5.1 สภาพแวดล้อมที่ใช้ในการทดสอบกลไกส่วนขยายของคอร์บาในงานวิจัยนี้[14]

---

[14] โปรแกรมประยุกต์ที่ทำงานอยู่บนออร์บของซอฟต์แวร์คอร์บาต่างผู้ผลิตกันจะสามารถสื่อสารแลกเปลี่ยนข้อมูลกันได้อย่างถูกต้องโดยใช้โพรโตคอลไอออฟ (IIOP Protocol) ตามข้อกำหนดคอร์บาของโอเอ็มจี [1,2]

จากรูปที่ 5.1 จะพบเครื่องคอมพิวเตอร์ 2 เครื่องเชื่อมต่อถึงกันผ่านทางเครือข่ายท้องถิ่น (Local Area Network) โดยคอมพิวเตอร์ทั้ง 2 เครื่องจะติดตั้งระบบปฏิบัติการ (Operating System) ตลอดจนซอฟต์แวร์คอร์บาเพื่อใช้ในการทดสอบที่แตกต่างกัน เครื่องคอมพิวเตอร์ A จะติดตั้งซอฟต์แวร์ออร์บาคัส (ORBacus) และออร์บาคัสเทรดเดอร์ (ORBacus Trader) สำหรับภาษาจาวาพร้อมทั้งโปรแกรมเจดีเค (JDK - Java Development Kit[15]) รุ่น 1.2.2 ของบริษัทซันเพื่อใช้ในการคอมไพล์และใช้งานซอฟต์แวร์คอร์บาที่อยู่ในรูปของจาวาคลาสโดยซอฟต์แวร์ต่างๆที่กล่าวถึงมาทั้งหมดนี้จะทำงานอยู่บนระบบปฏิบัติการลีนุกซ์ (LINUX) งานวิจัยนี้กำหนดให้เครื่องคอมพิวเตอร์ A มีเลขที่อยู่ไอพีเป็น 155.7.1.1

ส่วนเครื่องคอมพิวเตอร์ B จะติดตั้งซอฟต์แวร์วิสิโบรคเกอร์ (Visibroker for Java[16]) รุ่น 3.3 สำหรับภาษาจาวาเพื่อทำหน้าที่เป็นออร์บให้กับโปรแกรมประยุกต์ของคอร์บาโดยภายในซอฟต์แวร์วิสิโบรคเกอร์นี้จะมีโปรแกรมเครื่องมือ (Tools) ต่างๆที่สามารถนำไปใช้ในการพัฒนาโปรแกรมประยุกต์ทั้งฝั่งของผู้ให้บริการและผู้รับบริการ (รายละเอียดการพัฒนาโปรแกรมประยุกต์ด้วยซอฟต์แวร์วิสิโบรคเกอร์อยู่ในภาคผนวก ก.) เครื่องคอมพิวเตอร์ B ที่ใช้ในการทดสอบในงานวิจัยนี้จะติดตั้งระบบปฏิบัติการวินโดวส์ (MS Windows 98) และใช้โปรแกรมเจดีเครุ่น 1.1.8 ของบริษัทซันในการคอมไพล์และใช้งานโปรแกรมของผู้ให้บริการและผู้รับบริการ (ทั้งโปรแกรมของผู้ให้บริการและผู้รับบริการในงานวิจัยนี้จะถูกพัฒนาขึ้นด้วยภาษาจาวาทั้งสิ้น) เครื่องคอมพิวเตอร์ B จะถูกกำหนดให้มีเลขที่อยู่ไอพีเป็น 155.7.1.2

ตัวอย่างที่ใช้ในการทดสอบในบทนี้จะแสดงให้เห็นถึงรูปแบบการแทนที่กันของบริการในระดับอินสแตนซ์ภายใต้สถานการณ์ที่แตกต่างกันออกไปดังรายละเอียดต่อไปนี้

5.1 ตัวอย่างทดสอบการแทนที่กันของบริการในระดับอินสแตนซ์เมื่อบริการทั้ง 2 มีส่วนต่อประสานที่แตกต่างกัน

ตัวอย่างนี้สมมติให้มีบริการ ser1 ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ B โดยบริการนี้จะถูกพัฒนาขึ้นตามส่วนต่อประสาน IDL:service_by_id/account_by_id:1.0 บริการ ser1 จะให้บริการค้นหายอดเงินคงเหลือจากฐานข้อมูลของระบบธนาคารโดยใช้เลขที่บัญชีเป็นคีย์ในการค้นหา โปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้งานบริการ ser1 นี้ได้ตามรูปที่ 5.2

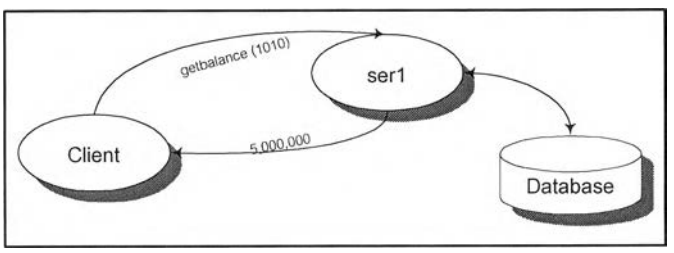
นอกจากบริการ ser1 แล้วงานวิจัยนี้ยังกำหนดให้มีบริการ ser2 ซึ่งเป็นบริการที่ถูกพัฒนาขึ้นตามส่วนต่อประสาน IDL:service_by_name/account_by_name:1.0 ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ B เช่นเดียวกัน บริการ ser2 จะให้บริการค้นหายอดเงินคงเหลือจากฐานข้อมูลเช่นเดียวกันกับบริการ ser1 แตกต่างกันตรงที่บริการ ser2 จะใช้ชื่อเจ้าของบัญชีเป็นคีย์ในการค้นหาแทนการใช้เลขที่บัญชี (ในที่นี้จะ

---

[15] ซอฟต์แวร์เจดีเคสามารถดาวน์โหลดได้จาก http://java.sun.com/products/jdk/

[16] ซอฟต์แวร์วิสิโบรคเกอร์สามารถดาวน์โหลดได้จาก http://www.inprise.com/visibroker/

สมมติให้ชื่อเจ้าของบัญชีภายในฐานข้อมูลเป็นค่าที่ไม่ซ้ำกัน) ถึงแม้ว่าบริการ ser1 และ ser2 จะมีส่วนต่อประสานที่แตกต่างกันก็ตาม ตัวอย่างนี้จะพัฒนาให้บริการทั้ง 2 ชนิดนี้ทำการค้นหาข้อมูลยอดเงินคงเหลือที่ถูกจัดเก็บอยู่ภายในฐานข้อมูลเดียวกัน (Shared Data) จึงกล่าวได้ว่าบริการทั้ง 2 นี้จัดเป็นบริการแบบมีสถานะ[17] (Stateful Service) ที่สามารถถูกใช้งานแทนที่กันได้ ตัวอย่างการทดสอบนี้มีเป้าหมายที่จะแสดงให้เห็นว่ากลไกส่วนขยายของคอร์บาที่ถูกพัฒนาขึ้นในงานวิจัยนี้จะช่วยทำให้เกิดการแทนที่กันของบริการในระดับอินสแตนซ์ได้โดยโปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้บริการ ser2 ให้ทำการค้นหายอดเงินคงเหลือจากฐานข้อมูลแทนการเรียกใช้บริการ ser1 ได้เมื่อบริการ ser1 ไม่พร้อมที่จะให้บริการ



รูปที่ 5.2 โปรแกรมผู้รับบริการเรียกใช้บริการ ser1 เพื่อค้นหายอดเงินจากฐานข้อมูล

การทดสอบในหัวข้อนี้จะใช้งานเครื่องคอมพิวเตอร์ A สำหรับให้บริการเทรดเดอร์ของคอร์บา งานการพัฒนาโปรแกรมประยุกต์ของผู้ให้บริการและผู้รับบริการต่างๆจะกระทำอยู่ภายในเครื่องคอมพิวเตอร์ B ทั้งหมด (ดังรายละเอียดในรูปที่ 5.3) ตัวอย่างนี้จะพัฒนาโปรแกรมในส่วนของผู้ให้บริการโดยใช้ภาษาจาวาทั้งสิ้น 2 โปรแกรมดังรายละเอียดต่อไปนี้

1. **โปรแกรมผู้ให้บริการ** servicebyid

เป็นโปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นสำหรับให้บริการ ser1 กับผู้รับบริการใดๆตามข้อกำหนดคอร์บา บริการ ser1 นี้จะมีส่วนต่อประสานเป็น IDL:service_by_id/account_by_id:1.0 ดังที่กล่าวไปแล้ว (รายละเอียดของส่วนต่อประสานนี้อยู่ในรูปที่ 5.4) งานวิจัยนี้จะสมมติให้บริการ ser1 ทำหน้าที่ให้บริการค้นหายอดเงินคงเหลือจากฐานข้อมูลโดยใช้เลขที่บัญชีเป็นคีย์ในการค้นหา โปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้บริการ ser1 ได้โดยใช้การเรียกไปยังเมทอด getbalance และทำการป้อนเลขที่บัญชีเป็นค่าพารามิเตอร์อินพุตให้กับเมทอด

---

[17] บริการแบบมีสถานะ (Stateful Service) คือบริการใดๆที่มีการจัดเก็บหรือเปลี่ยนแปลงสถานะหรือข้อมูลต่างๆภายหลังการทำงาน โดยสถานะหรือข้อมูลต่างๆเหล่านี้จะเป็นตัวกำหนดรูปแบบการทำงานของบริการต่อไป

งานวิจัยนี้ได้ทำการพัฒนาโปรแกรมขึ้นภายในแฟ้มข้อมูล servicebyid.java โดยจัดสร้างคลาส servicebyid[18] ขึ้นเป็นคลาสหลักของโปรแกรมเพื่อทำงานในหน้าที่ต่างๆดังต่อไปนี้

- รับตัวเลือก -trader จากผู้ใช้งานโปรแกรมผ่านทางค่าพารามิเตอร์อินพุตเพื่อจะระบุให้คลาส servicebyid ทำการเอ็กซ์พอร์ตข้อเสนอบริการของบริการ ser1 ไปยังเทรดเดอร์หรือไม่ ถ้าผู้ใช้งานโปรแกรม servicebyid โดยไม่ได้ระบุตัวเลือกนี้ในขณะเรียกใช้งานโปรแกรมจะมีผลทำให้คลาส servicebyid ทำการลงทะเบียนประกาศการให้บริการ ser1 ไปยังสมาร์ทเอเจนท์เท่านั้น

- สร้างและกำหนดค่าโดยปริยายให้กับออร์บและออบเจ็กต์อะแดปเตอร์ภายในโปรแกรมของผู้ให้บริการ (งานวิจัยนี้กำหนดให้หมายเลขพอร์ตที่ใช้รับคำร้องขอใช้บริการมีค่าเท่ากับ 1500)

- สร้างและกำหนดชื่ออินสแตนซ์ของบริการ ser1 เพื่อทำหน้าที่ให้บริการกับผู้รับบริการโดยอินสแตนซ์ของบริการนี้จะใช้ชื่อส่วนต่อประสานเป็น IDL:service_by_id/account_by_id:1.0

- สร้างการเชื่อมต่อไปยังฐานข้อมูลผ่านทางไดรเวอร์เจดีบีซีของจาวา (JDBC Driver) โดยงานวิจัยนี้จะทำการโหลดไดรเวอร์เจดีบีซีเข้าสู่หน่วยความจำของเครื่องโดยใช้การผ่านค่า [14,16] sun.jdbc.odbc.JdbcOdbcDriver ให้กับเมทอด Class.forName ส่วนเชื่อมต่อที่ได้รับ (Connection) มาจะถูกนำไปใช้ในการค้นหายอดเงินคงเหลือจากฐานข้อมูลต่อไป

- เขียนข้อมูลอ้างอิงของอินสแตนซ์บริการ ser1 ลงในแฟ้มข้อมูล c:\thesis\ior\ser1.ior

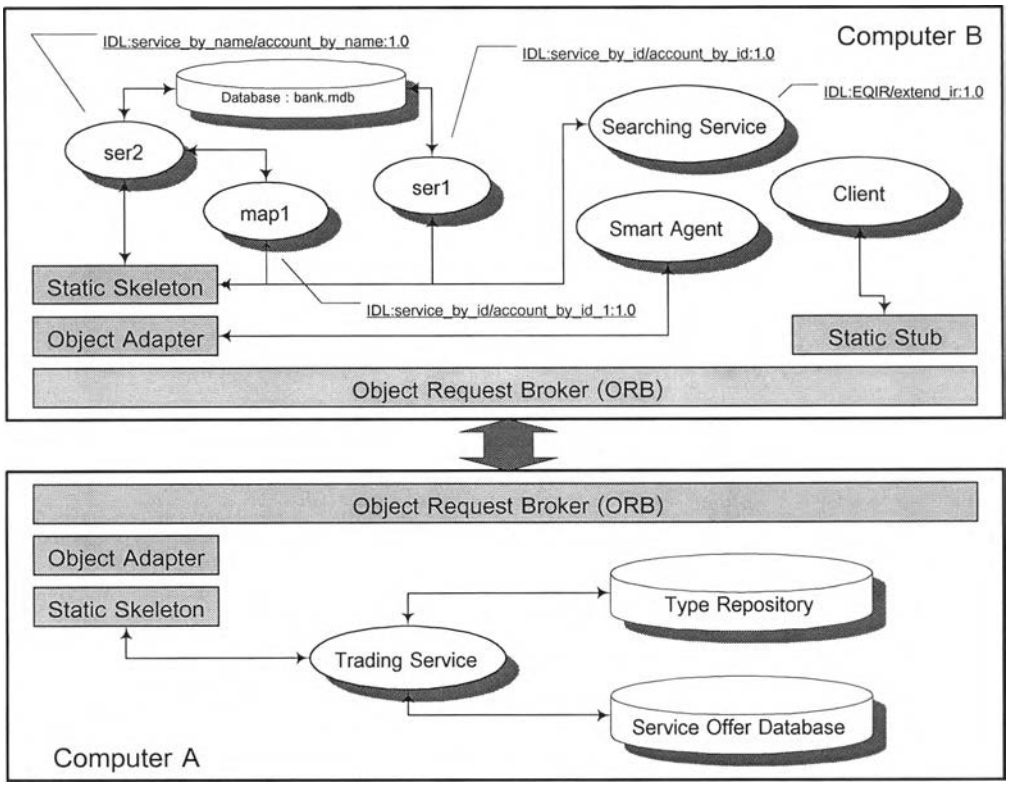
- ถ้าผู้ใช้ระบุตัวเลือก –trader คลาส servicebyid จะทำการเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์ที่ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ A โดยใช้ข้อมูลตามรายละเอียดในตารางที่ 5.1

- เมื่อทำงานในขั้นตอนดังกล่าวแล้วเสร็จ คลาส servicebyid จะทำการลงทะเบียนประกาศการให้บริการ ser1 ไปยังสมาร์ทเอเจนท์ผ่านการเรียกใช้เมทอด obj_is_ready จากนั้นจึงทำการลูปรอรับคำร้องขอใช้บริการจากผู้รับบริการใดๆโดยใช้การเรียกไปยังเมทอด impl_is_ready ของออบเจ็กต์อะแดปเตอร์ในคอร์บา

จากรูปที่ 5.4 เมื่อผู้รับบริการเรียกใช้เมทอด getbalance ของบริการ ser1 โดยทำการระบุเลขที่บัญชีเป็นพารามิเตอร์อินพุตของเมทอด บริการ ser1 จะนำค่าพารามิเตอร์นี้ไปทำการค้นหายอดเงินคงเหลือจากตาราง accountinfo ภายในฐานข้อมูล bank.mdb ซึ่งเป็นแฟ้มข้อมูลของไมโครซอฟต์แอ็กเซส (MS Access) แฟ้มข้อมูล bank.mdb นี้จะถูกจัดเก็บอยู่ภายในไดเร็คทอรี c:\thesis\database ของเครื่องคอมพิวเตอร์ B อนึ่งการค้นหาข้อมูลจะกระทำผ่านการส่งคำสั่ง SQL ไปยังเมทอด executeQuery ของคลาส java.sql.Statement ในภาษาจาวา [11,14]

---

[18] คลาส servicebyid จะสืบทอดคุณสมบัติจากคลาส service_by_id._account_by_idImplBase ซึ่งเป็นคลาสสเคเลตันที่ได้จากการคอมไพล์แฟ้มข้อมูลไอดีแอลที่ใช้จัดเก็บส่วนต่อประสานตามรูปที่ 5.4 การคอมไพล์แฟ้มข้อมูลไอดีแอลนี้จะกระทำผ่านการเรียกใช้คำสั่ง idl2java ของซอฟต์แวร์วิสิโบรคเกอร์



รูปที่ 5.3 รูปแบบของบริการตลอดจนส่วนประกอบต่างๆที่ใช้ในการทดสอบตัวอย่างที่ 5.1

```
//------------------------------------------------------------------------
// Repository ID : IDL:service_by_id/account_by_id:1.0
// IDL File : c:\thesis\service1\account_by_id.idl
//------------------------------------------------------------------------
module service_by_id {
  interface account_by_id {
     float getbalance(in long long id);
  };
};
```

รูปที่ 5.4 ส่วนต่อประสานของบริการ ser1

ตารางที่ 5.1 ข้อมูลที่ใช้ในการเอ็กซ์พอร์ตข้อเสนอบริการ ser1 ไปยังบริการเทรดเดอร์

ชื่อชนิดบริการ (Service Type Name)	::service_by_id::account_by_id		
ชื่อส่วนต่อประสาน (Interface Name)	IDL:service_by_id/account_by_id:1.0		
คุณสมบัติของบริการ ser1 (Service Property)	search_by	string	"id"
	area	string	"NSEWC"
	cost	float	30.00
	responsetime	float	40.00
ข้อมูลอ้างอิงของบริการ ser1 (Object Reference of Ser1)	IOR:000000000000002449444c3a73657276696365 5f62795f69642f6163636f756e745f62795f69643a312e30000000000 1000000000000005100010000000000a3135352e372e312e320005dc0000003900504d43000000000000002449444c3a73657276696365 5f62795f69642f6163636f756e745f62795f69643a312e3000000000057365723100 **ความหมายจากการถอดรหัสข้อมูลอ้างอิงของบริการ** Interoperable Object Reference Type ID: IDL:service_by_id/account_by_id:1.0 Contains 1 profile. Profile 0-IIOP Profile: Version: 1.0 Host: 155.7.1.2 Port: 1500 Object Key: PersistentId [repId=IDL:service_by_id/account_by_id:1.0,objectName=ser1]		

2. **โปรแกรมผู้ให้บริการ** servicebyname

เป็นโปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นสำหรับให้บริการ ser2 กับผู้รับบริการใดๆตามข้อกำหนดคอร์บา งานวิจัยนี้จะสมมติให้บริการ ser2 ทำหน้าที่ให้บริการค้นหายอดเงินคงเหลือจากฐานข้อมูลโดยใช้ชื่อเจ้าของบัญชีเป็นคีย์ในการค้นหาแทนการใช้เลขที่บัญชี โปรแกรมของผู้รับบริการจะสามารถเรียกใช้บริการ ser2 นี้ได้โดยใช้การเรียกไปยังเมทอด getbalance เช่นเดียวกัน

บริการ ser2 จะมีส่วนต่อประสานเป็น IDL:service_by_name/account_by_name:1.0 (รายละเอียดของส่วนต่อประสานนี้อยู่ในรูปที่ 5.5) ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมขึ้นภายในแฟ้มข้อมูล servicebyname.java โดยจัดสร้างคลาส servicebyname ขึ้นเป็นคลาสหลักของโปรแกรมเพื่อทำงานในหน้าที่ต่างๆที่คล้ายคลึงกับคลาส servicebyid แต่จะมีความแตกต่างกันในรายละเอียดบางอย่างดังต่อไปนี้

- กำหนดให้คลาส servicebyname ใช้หมายเลขพอร์ตเป็น 1501 ในการรับคำร้องขอใช้บริการจากโปรแกรมผู้รับบริการ

- คลาส servicebyname จะเขียนข้อมูลอ้างอิงของอินสแตนซ์บริการ ser2 ลงในแฟ้มข้อมูล c:\thesis\ior\ser2.ior ภายหลังจากที่ได้จัดสร้างอินสแตนซ์ของบริการแล้ว

- ถ้าผู้ใช้งานโปรแกรมระบุตัวเลือก –trader เมื่อเรียกใช้งานโปรแกรม servicebyname คลาส servicebyname จะทำการเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์ที่ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ A โดยใช้ข้อมูลตามรายละเอียดในตารางที่ 5.2

```
//------------------------------------------------------------------------
// Repository ID : IDL:service_by_name/account_by_name:1.0
// IDL File : c:\thesis\service2\account_by_name.idl
//------------------------------------------------------------------------
module service_by_name {
    interface account_by_name {
        float getbalance(in string name);
    };
};
```

รูปที่ 5.5 ส่วนต่อประสานของบริการ ser2

ตารางที่ 5.2 ข้อมูลที่ใช้ในการเอ็กซ์พอร์ตข้อเสนอบริการ ser2 ไปยังบริการเทรดเดอร์

ชื่อชนิดบริการ (Service Type Name)	::service_by_name::account_by_name		
ชื่อส่วนต่อประสาน (Interface Name)	IDL:service_by_name/account_by_name:1.0		
คุณสมบัติของบริการ ser2 (Service Property)	search_by	string	"name"
	area	string	"NSEWC"
	cost	float	20.00
	responsetime	float	80.00

ตารางที่ 5.2 ข้อมูลที่ใช้ในการเอ็กซ์พอร์ตข้อเสนอบริการ ser2 ไปยังบริการเทรดเดอร์ (ต่อ)

ข้อมูลอ้างอิงของบริการ ser2 (Object Reference of Ser2)	IOR:000000000000002849444c3a736572766963655f62795f6e6 16d652f6163636f756e745f62795f6e616d653a312e3000000000 0100000000000000550001000000000000a3135352e372e312e32 0005dd0000003d00504d43000000000000000002849444c3a73657 2766963655f62795f6e616d652f6163636f756e745f62795f6e616 d653a312e3000000000057365723200 **ความหมายจากการถอดรหัสข้อมูลอ้างอิงของบริการ** Interoperable Object Reference Type ID: IDL:service_by_name/account_by_name:1.0 Contains 1 profile. Profile 0-IIOP Profile: Version: 1.0 Host: 155.7.1.2 Port: 1501 Object Key: PersistentId [repId=IDL:service_by_name/account_by_name:1.0 ,objectName=ser2]

จากรูปที่ 5.5 เมื่อผู้รับบริการเรียกใช้เมทอด getbalance ของบริการ ser2 โดยทำการระบุชื่อเจ้าของบัญชีเป็นพารามิเตอร์อินพุตของเมทอด บริการ ser2 จะนำค่าพารามิเตอร์นี้ไปทำการค้นหายอดเงินคงเหลือจากตาราง accountinfo ภายในฐานข้อมูล bank.mdb โดยแฟ้มข้อมูลนี้จะเป็นแฟ้มข้อมูลเดียวกันกับที่ถูกใช้งานโดยบริการ ser1 ส่งผลให้บริการทั้ง 2 บริการนี้จะมีความสัมพันธ์ในลักษณะที่เท่าเทียมกัน กล่าวคือทั้งบริการ ser1 และ ser2 จะส่งกลับยอดเงินคงเหลือเป็นค่าเดียวกันไปให้กับผู้รับบริการ ถ้าหากโปรแกรมของผู้รับบริการทำการค้นหาข้อมูลที่ถูกจัดเก็บอยู่ในตำแหน่งเดียวกัน ถึงแม้ว่าบริการทั้ง 2 นี้จะใช้คีย์ในการค้นหาข้อมูลที่แตกต่างกันก็ตาม ตัวอย่างเช่น (รูปที่ 5.6) ถ้าโปรแกรมของผู้รับบริการทำการค้นหายอดเงินคงเหลือจากเลขที่บัญชี 1010 โดยเรียกใช้บริการ ser1 (ส่งค่า 1010 เป็นค่าพารามิเตอร์ให้กับเมทอด getbalance ของบริการ ser1) หรือต้องการค้นหายอดเงินจากชื่อเจ้าของบัญชี “Robin Company” โดยเรียกใช้บริการ ser2 (ส่งค่า “Robin Company” เป็นค่าพารามิเตอร์ให้กับเมทอด getbalance ของบริการ ser2) ทั้งบริการ ser1 และ ser2 จะส่งกลับยอดเงินคงเหลือเป็นค่าเดียวกันคือ 5,000,000 กลับไปยังโปรแกรมผู้รับบริการ เป็นต้น

เลขที่บัญชี	ชื่อเจ้าของบัญชี	ยอดเงินคงเหลือ
1010	*Robin Company*	*5,000,000*
2078	Verbatim Company	20,000,000
3015	Energizer Company	17,050,000
.....	.....	.....

รูปที่ 5.6 ตัวอย่างข้อมูลภายในตาราง accountinfo ของฐานข้อมูล bank.mdb

นอกจากโปรแกรมทั้ง 2 โปรแกรมที่ได้กล่าวถึงไปแล้วข้างต้น ตัวอย่างนี้ยังได้จัดสร้างคลาส mapserv1 ขึ้นภายในแฟ้มข้อมูล mapserv1.java เพื่อทำงานเป็นตัวดำเนินการแปลง map1 สำหรับใช้ดักและแปลงคำร้องขอใช้บริการที่โปรแกรมของผู้รับบริการส่งไปยังบริการ ser1 ให้สอดคล้องกับส่วนต่อประสานของบริการ ser2 ก่อนที่จะส่งผ่านคำร้องขอใช้บริการนี้ไปทำงานยังบริการ ser2 ต่อไป ตัวดำเนินการแปลง map1 นี้จะถูกเรียกใช้งานโดยโปรแกรมของผู้รับบริการอย่างโปร่งใส เมื่อเกิดการแทนที่บริการ ser1 โดยบริการ ser2 ซึ่งมีส่วนต่อประสานที่แตกต่างกัน

บริการ map1 จะมีส่วนต่อประสานเป็น IDL:service_by_id/account_by_id_1:1.0 โดยส่วนต่อประสานนี้จะเป็นส่วนต่อประสานลูก (Child Interface) ซึ่งได้รับการสืบทอดคุณสมบัติจากส่วนต่อประสาน account_by_id ของบริการ ser1 นั่นเอง (รายละเอียดของส่วนต่อประสานนี้อยู่ในรูปที่ 5.7) คลาส mapserv1 จะทำงานตามขั้นตอนต่างๆคล้ายคลึงกับคลาส servicebyid แต่จะมีความแตกต่างกันในรายละเอียดบางอย่างดังต่อไปนี้

- คลาส mapserv1 จะใช้หมายเลขพอร์ต 1502 ในการดักคำร้องขอใช้บริการที่โปรแกรมผู้รับบริการส่งไปยังบริการ ser1

- คลาส mapserv1 จะเขียนข้อมูลอ้างอิงของอินสแตนซ์บริการ map1 ลงในแฟ้มข้อมูล c:\thesis\ior\map1.ior ภายหลังจากที่ได้จัดสร้างอินสแตนซ์ของบริการแล้ว

- คลาส mapserv1 ซึ่งถูกใช้งานเป็นตัวดำเนินการแปลงจะทำการลงทะเบียนประกาศการให้บริการ map1 ไปยังสมาร์ทเอเจนท์เท่านั้น บริการ map1 จะไม่เอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์ อย่างไรก็ตามโปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้งานตัวดำเนินการแปลง map1 นี้ได้อย่างโปร่งใสโดยอาศัยการทำงานร่วมกันระหว่างบริการค้นหาและคำสั่งที่ถูกแทรกลงในโปรแกรมของผู้รับบริการด้วยโปรแกรมพรีโพรเซสเซอร์หรือจากการทำงานของส่วนเพิ่มขยายของบริการเทรดเดอร์ที่ถูกพัฒนาขึ้นใหม่

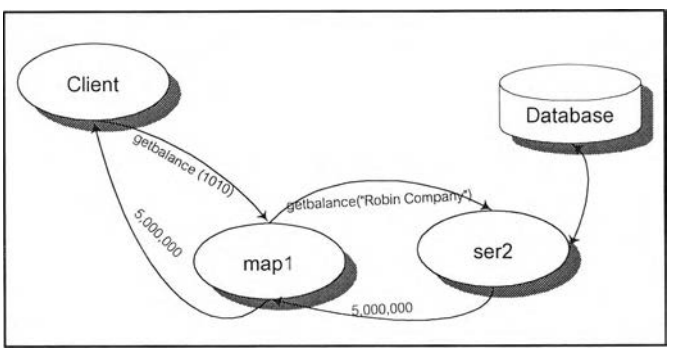
ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมภายในเมทอด getbalance ของบริการ map1 ให้ทำการแปลงค่าพารามิเตอร์อินพุตที่รับเข้ามาเป็นเลขที่บัญชี (ชนิดข้อมูลเป็น long ในภาษาจาวา) ให้เปลี่ยนค่าไปเป็น

ชื่อเจ้าของบัญชี (ชนิดข้อมูลเป็น String ในภาษาจาวา) โดยใช้การเรียกไปยังเมทอด getname_by_id ซึ่งเป็นฟังก์ชันการแปลง ค่าพารามิเตอร์อินพุตนี้จะถูกเปลี่ยนให้มีชนิดและความหมาย (Syntax and Semantics) ที่สอดคล้องกันกับส่วนต่อประสานของบริการ ser2 ก่อนที่จะถูกส่งผ่านไปยังเมทอด getbalance ของบริการ ser2 เพื่อทำการค้นหาข้อมูลต่อไป

เมื่อบริการ ser2 เสร็จสิ้นกระบวนการค้นหาข้อมูลภายในฐานข้อมูลแล้ว บริการ ser2 จะส่งยอดเงินคงเหลือที่พบกลับมายังบริการ map1 ซึ่งบริการ map1 จะส่งผ่านข้อมูลที่ได้รับมานี้กลับไปยังโปรแกรมของผู้รับบริการอีกทอดหนึ่ง โปรแกรมผู้รับบริการจะไม่ทราบเลยว่ายอดเงินคงเหลือที่ได้รับมานี้เป็นผลการค้นหาของบริการใด ทั้งนี้กลไกการแทนที่กันของบริการ ser1 และ ser2 จะเกิดขึ้นอย่างโปร่งใส ตามรูปที่ 5.8

```
//------------------------------------------------------------------------
// Repository ID : IDL:service_by_id/account_by_id_1:1.0
// IDL File : c:\thesis\mapping1\mapserv1.idl
//------------------------------------------------------------------------
module service_by_id {
  interface account_by_id {
        float getbalance(in long long id);
  };
  interface account_by_id_1 : account_by_id {
       string getname_by_id(in long long id);
  };
};
```

รูปที่ 5.7 ส่วนต่อประสานของบริการ map1



รูปที่ 5.8 การเรียกใช้งานบริการ ser2 ผ่านทางตัวดำเนินการแปลง map1

ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมของผู้รับบริการขึ้นภายในแฟ้มข้อมูล client1.java โดยใช้ภาษาจาวา ทั้งนี้โปรแกรม client1 จะใช้งานคลาส client1 ในการเรียกใช้บริการ ser1 ด้วยวิธีการต่างๆดังต่อไปนี้

- ใช้การเรียกผ่านคำสั่ง bind ทั้ง 3 รูปแบบของซอฟต์แวร์วิสิโบรคเกอร์ (ตารางที่ 5.3)

ตารางที่ 5.3 การเรียกใช้งานบริการ ser1 ผ่านคำสั่ง bind ในโปรแกรมผู้รับบริการ client1

รูปแบบคำสั่ง	**คำสั่งและความหมาย**
bind รูปแบบที่ 1	*org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);* *service_by_id.account_by_id ser1= service_by_id.account_by_idHelper.bind(orb);* ความหมายของคำสั่ง bind โปรแกรมของผู้รับบริการต้องการเรียกใช้อินสแตนซ์ของบริการใดๆที่มีส่วนต่อประสานเป็น IDL:service_by_id/account_by_id:1.0
bind รูปแบบที่ 2	*org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);* *String objname="ser1";* *service_by_id.account_by_id ser1 = service_by_id.account_by_idHelper.bind (orb,objname);* ความหมายของคำสั่ง bind โปรแกรมของผู้รับบริการต้องการเรียกใช้อินสแตนซ์ของบริการชื่อ ser1 โดยบริการนี้จะต้องมีส่วนต่อประสานเป็น IDL:service_by_id/account_by_id:1.0
bind รูปแบบที่ 3	*org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);* *String objname="ser1";* *String host = "155.7.1.2";* *org.omg.CORBA.BindOptions bindopt = new org.omg.CORBA.BindOptions();* *service_by_id.account_by_id ser1=service_by_id.account_by_idHelper.bind (orb,objname,host,bindopt);* ความหมายของคำสั่ง bind โปรแกรมของผู้รับบริการต้องการเรียกใช้อินสแตนซ์ของบริการชื่อ ser1 โดยอินสแตนซ์ของบริการนี้จะต้องทำงานอยู่ภายในโฮสต์ที่มีเลขที่อยู่ไอพีเป็น 155.7.1.2 และมีส่วนต่อประสานเป็น IDL:service_by_id/account_by_id:1.0 เท่านั้น

- ใช้การเรียกผ่านทางข้อมูลอ้างอิงของบริการที่ถูกจัดเก็บอยู่ภายในแฟ้มข้อมูล ser1.ior

- ใช้การอิมพอร์ตข้อเสนอบริการ ser1 จากบริการเทรดเดอร์ในเครื่องคอมพิวเตอร์ A โดยกำหนดเงื่อนไขการค้นหาเป็น

(searchby == 'id')and('C' ~ area)and('S' ~ area)and(cost <= 500)and(responsetime <1000)

ขั้นตอนการทดสอบตลอดจนผลการทำงานของโปรแกรมผู้รับบริการที่เกิดขึ้น ก่อนที่จะถูกแทรกคำสั่งโดยโปรแกรมพรีโพรเซสเซอร์เป็นดังต่อไปนี้

1. เรียกใช้บริการเทรดเดอร์ที่ติดตั้งอยู่ในเครื่องคอมพิวเตอร์ A โดยใช้คำสั่ง

java com.ooc.CosTrading.Server -i -OAport 5000

บริการเทรดเดอร์จะบันทึกข้อมูลอ้างอิงของบริการลงในแฟ้มข้อมูล ob_ts พร้อมทั้งรอรับคำร้องขอใช้บริการที่พอร์ตหมายเลข 5000 จากนั้นให้ทำการดาวน์โหลดแฟ้มข้อมูล ob_ts มาจัดเก็บลงในไดเร็คทอรี c:\thesis\ior ของเครื่องคอมพิวเตอร์ B โดยใช้โพรโตคอลเอฟทีพี (FTP Protocol) แฟ้มข้อมูลนี้จะถูกใช้ในการอิมพอร์ตและเอ็กซ์พอร์ตบริการจากโปรแกรมของผู้ให้บริการและผู้รับบริการไปยังบริการเทรดเดอร์

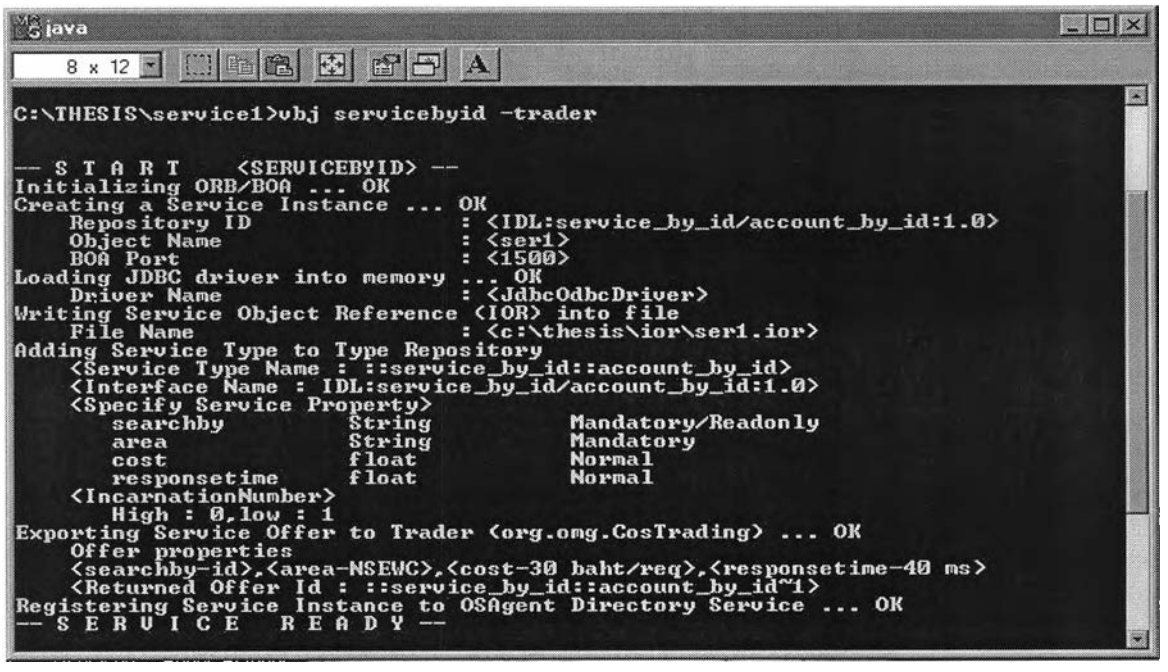
2. เรียกใช้สมาร์ทเอเจนท์โดยคลิกที่ไอคอน (Icon) ของสมาร์ทเอเจนท์ซึ่งอยู่ภายในไดเร็คทอรีที่ติดตั้งซอฟต์แวร์วิสิโบรคเกอร์ของเครื่องคอมพิวเตอร์ B (รูปที่ 5.9)

รูปที่ 5.9 ไอคอนสมาร์ทเอเจนท์ของซอฟต์แวร์วิสิโบรคเกอร์

3. เรียกใช้โปรแกรมผู้ให้บริการ servicebyid เพื่อสร้างอินสแตนซ์และให้บริการ ser1 ตามข้อกำหนดของคอร์บากับโปรแกรมของผู้รับบริการใดๆบนเครื่องคอมพิวเตอร์ B โดยใช้คำสั่งดังนี้

c:\thesis\service1\vbj servicebyid – trader

บริการ ser1 จะถูกลงทะเบียนประกาศการให้บริการไปยังสมาร์ทเอเจนท์และถูกเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์อีกด้วย (ใช้ตัวเลือก -trader) ผลการเรียกใช้งานโปรแกรมของผู้ให้บริการ servicebyid จะแสดงรายละเอียดของการทำงานต่างๆบนหน้าจอ ตามรูปที่ 5.10

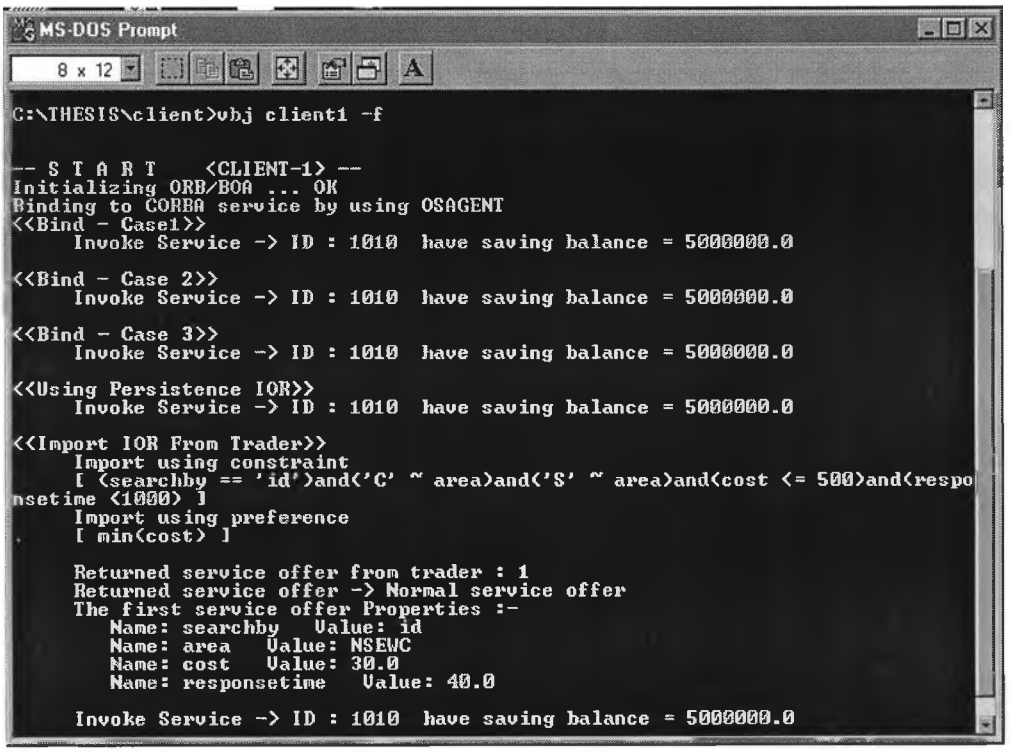


รูปที่ 5.10 ผลการเรียกใช้งานโปรแกรมผู้ให้บริการ servicebyid

4. เรียกใช้โปรแกรมของผู้รับบริการ client1 ในเครื่องคอมพิวเตอร์ B เพื่อสังเกตผลที่เกิดขึ้นจากการค้นหาอินสแตนซ์ของบริการ ser1 โดยใช้การเรียกคำสั่ง bind ทั้ง 3 รูปแบบรวมทั้งการเรียกใช้บริการ ser1 จากข้อมูลอ้างอิงของบริการในแฟ้มข้อมูล c:\thesis\ior\ser1.ior ตลอดจนการอิมพอร์ตข้อเสนอบริการจากบริการเทรดเดอร์ คำสั่งในการเรียกใช้โปรแกรมผู้รับบริการจะเป็นดังนี้ *c:\thesis\client\vbj client1 –f*

ตัวเลือก –f ใช้ในการระบุให้โปรแกรมของผู้รับบริการจัดเก็บ (Logging) ผลการทำงานที่เกิดขึ้นของโปรแกรมลงในแฟ้มข้อมูล c:\thesis\client\client1.result ด้วย (รูปที่ 5.11)

จากรูปที่ 5.11 จะพบว่าโปรแกรมของผู้รับบริการจะสามารถเรียกใช้งานบริการ ser1 เพื่อทำการค้นหายอดเงินคงเหลือของเลขที่บัญชี 1010 จากฐานข้อมูลได้ ไม่ว่าโปรแกรม client1 จะใช้การค้นหาอินสแตนซ์ของบริการ ser1 ด้วยวิธีการใดก็ตาม การเรียกใช้งานบริการ ser1 ในรูปแบบนี้จะเป็นการเรียกใช้บริการในคอร์บาตามรูปแบบปกติ



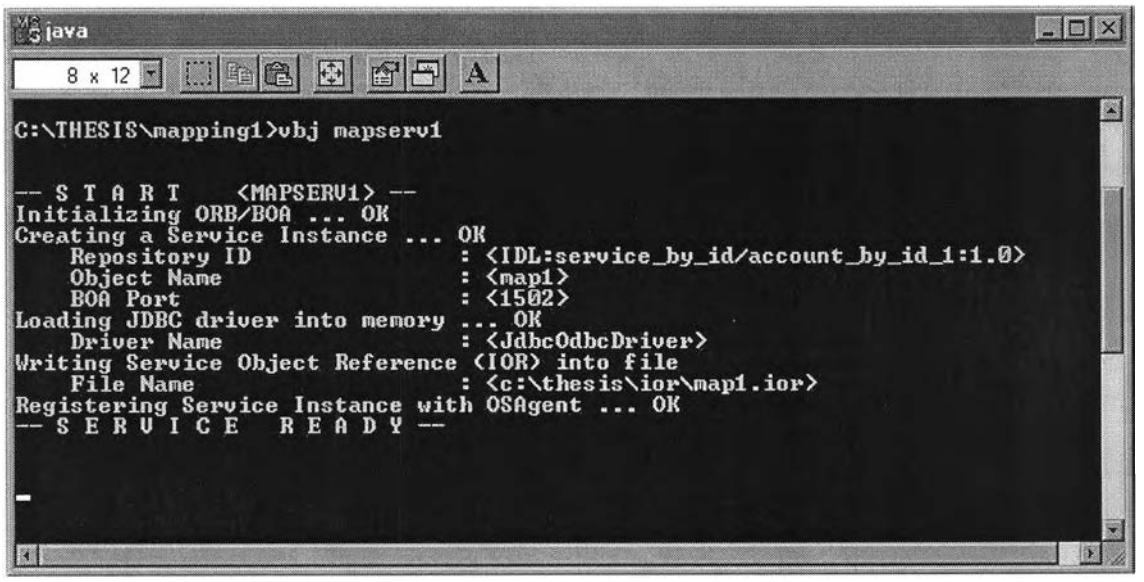
รูปที่ 5.11 ผลการเรียกใช้งานโปรแกรมผู้รับบริการด้วยคำสั่ง vbj client1 -f

5. หยุดการทำงานของโปรแกรมผู้ให้บริการ servicebyid (โดยกด <Ctrl>+ c) และทดลองเรียกใช้งานโปรแกรมผู้รับบริการ client1 อีกครั้ง สังเกตผลที่เกิดขึ้นจากการค้นหาอินสแตนซ์ของบริการ ser1 ในขณะที่อินสแตนซ์ของบริการ ser1 ไม่พร้อมที่จะให้บริการกับผู้ร้องขอ (ขณะโปรแกรม servicebyid หยุดการทำงานอยู่) โปรแกรมผู้รับบริการจะเกิดเอ็กซ์เซพชันขึ้นจากการเรียกคำสั่ง bind (รูปที่ 5.12)

รูปที่ 5.12 ผลการเรียกใช้งานโปรแกรม client1 เมื่อโปรแกรม servicebyid หยุดการทำงาน

6. เรียกใช้โปรแกรมของผู้ให้บริการ servicebyname และโปรแกรม mapserv1 (รูปที่ 5.13 และ 5.14) พร้อมทำการเรียกใช้โปรแกรมของผู้รับบริการ client1 อีกครั้ง (รูปที่ 5.15) เพื่อสังเกตผลลัพธ์ที่เกิดขึ้น (ขณะนี้โปรแกรม servicebyid ยังคงหยุดการทำงานอยู่) ซึ่งผลที่ได้จะแสดงให้เห็นถึงการแทนที่บริการ ser1 โดยใช้บริการ ser2 ผ่านทางตัวดำเนินการแปลง map1 เมื่อโปรแกรมของผู้รับบริการเรียกใช้คำสั่ง bind ในรูปแบบที่ 1 ทั้งนี้เนื่องจากตัวดำเนินการแปลง map1 มีส่วนต่อประสานเป็น account_by_id_1 ซึ่งส่วนต่อประสานนี้จะเป็นส่วนต่อประสานที่สืบทอดมาจากส่วนต่อประสานของบริการ ser1 ที่ผู้รับบริการต้องการและคอร์บาจะรองรับการแทนที่กันของบริการในระดับซนิดได้แต่สำหรับการ bind ในกรณีอื่น (กรณีรูปแบบที่ 2 และ 3) จะเกิดเอ็กซ์เซพชันขึ้น

รูปที่ 5.13 ผลการเรียกใช้งานโปรแกรม servicebyname



รูปที่ 5.14 ผลการเรียกใช้งานโปรแกรม mapserv1

รูปที่ 5.15 ผลการเรียกใช้งานโปรแกรม client1 เมื่อเกิดการแทนที่บริการ ser1 ด้วยบริการ ser2 ในระดับซนิด (กรณี bind รูปแบบที่ 1)

สรุปผลการทดสอบตามขั้นตอนต่างๆข้างต้น

1. โปรแกรมของผู้รับบริการในคอร์บาจะสามารถเรียกใช้งานบริการใดๆของโปรแกรมของผู้ให้บริการได้ ถ้าอินสแตนซ์ของบริการนั้นๆอาทิเช่นอินสแตนซ์ของบริการ ser1 ยังคงทำงานอยู่ตามปกติภายในโปรแกรมของผู้ให้บริการ

2. โปรแกรมของผู้รับบริการจะเกิดเอ็กซ์เซพชันชนิด org.omg.CORBA.NO_IMPLEMENT ขึ้น เมื่อออร์บของคอร์บาไม่สามารถทำการค้นหาและส่งข้อมูลอ้างอิงของบริการ ser1 กลับไปทำงานที่โปรแกรมของผู้รับบริการได้ตามปกติ

3. โปรแกรมของผู้รับบริการจะสามารถเรียกใช้บริการ ser2 ผ่านทางตัวดำเนินการแปลง map1 ได้ ในกรณีที่โปรแกรมของผู้รับบริการเรียกใช้คำสั่ง bind รูปแบบที่ 1 ทั้งนี้เนื่องจากข้อกำหนดคอร์บาจะสนับสนุนการแทนที่กันของบริการในระดับชนิดของบริการในกรณีที่ใช้การ bind แบบไม่เจาะจงอินสแตนซ์ของบริการ (ตามรายละเอียดในบทที่ 1) การแทนที่กันของบริการในลักษณะนี้จะไม่เกิดขึ้นเมื่อโปรแกรมของผู้รับบริการเรียกใช้คำสั่ง bind ในรูปแบบที่ 2 และ 3 รวมถึงกรณีที่โปรแกรมผู้รับบริการพยายามเรียกใช้บริการจากข้อมูลอ้างอิงของบริการภายในแฟ้มข้อมูลอีกด้วย การที่ตัวดำเนินการแปลง map1 ในตัวอย่างการทดสอบนี้สามารถถูกเรียกใช้งานแทนที่บริการ ser1 ได้เพราะตัวดำเนินการแปลง map1 มีชื่อของส่วนต่อประสานเป็นส่วนต่อประสานลูกของส่วนต่อประสานของบริการ ser1 นั่นเอง อย่างไรก็ตามถ้าหากว่าภายในสมาร์ทเอเจนท์มีบริการหลายๆบริการที่สามารถถูกใช้งานแทนที่บริการ ser1 ได้แล้ว โปรแกรมของผู้รับบริการจะไม่สามารถควบคุมให้ออร์บส่งกลับเฉพาะอินสแตนซ์ของบริการที่ต้องการใช้งานกลับมายังโปรแกรมผู้รับบริการได้

ตัวอย่างนี้ได้ทำการเรียกใช้โปรแกรมพรีโพรเซสเซอร์เพื่อทำการแทรกคำสั่งการทำงานลงในโปรแกรมของผู้รับบริการและจัดเก็บโปรแกรมที่ผ่านการแทรกคำสั่งแล้วลงในแฟ้มข้อมูลชื่อ eqclient1.java (รูปที่ 5.16) จากนั้นจึงทำการคอมไพล์และทดลองเรียกใช้งานโปรแกรมของผู้รับบริการอีกครั้งเพื่อเปรียบเทียบผลการทำงานที่เกิดขึ้นว่าคำสั่งที่แทรกโดยโปรแกรมพรีโพรเซสเซอร์จะช่วยทำให้โปรแกรมผู้รับบริการสามารถเรียกใช้งานบริการอื่นๆเพื่อทำงานแทนที่บริการที่ต้องการเมื่อบริการที่ต้องการไม่พร้อมที่จะให้บริการในขณะเรียกใช้งานได้หรือไม่

การทดสอบในส่วนนี้จะสมมติให้ผู้ให้บริการแทนที่ทำการกำหนดข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการลงในแฟ้มข้อมูล c:\thesis\config\config.eq (รูปที่ 5.17) ข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูลนี้จะถูกเรียกใช้งานโดยบริการค้นหาแทนการเรียกใช้ข้อมูลจากคลังจัดเก็บส่วนต่อประสานในงานวิจัย [4]

รูปที่ 5.16 การแทรกคำสั่งและจัดสร้างโปรแกรม eqclient1 โดยโปรแกรมพรีโพรเซสเซอร์

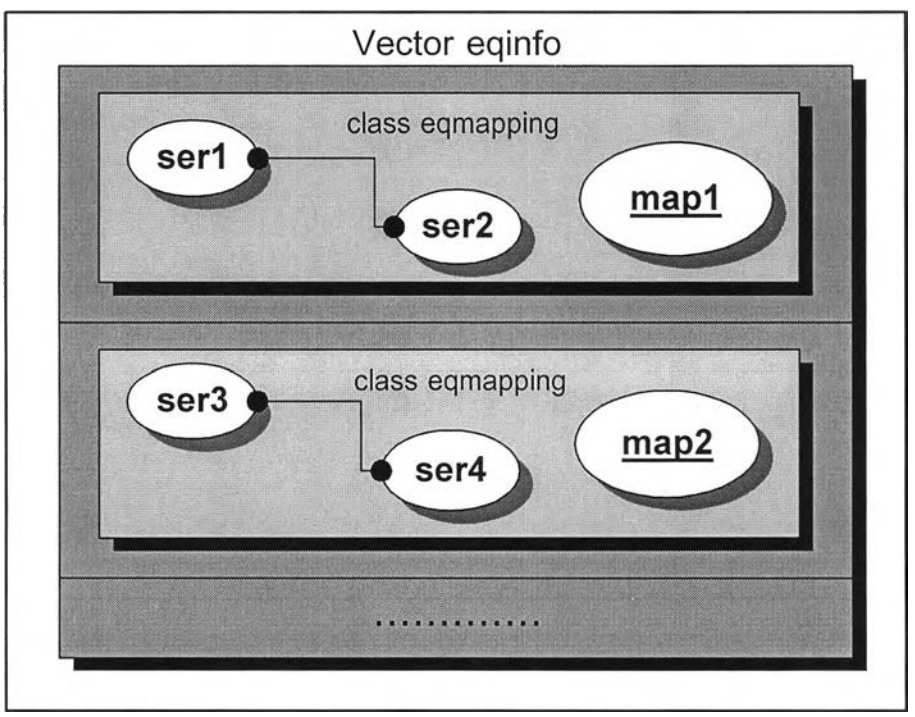
งานวิจัยนี้ได้ทำการพัฒนาโปรแกรม irserv ขึ้นโดยใช้ภาษาจาวาเพื่อทำหน้าที่เป็นบริการค้นหาสำหรับใช้ค้นหาบริการแทนที่จากข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูล config.eq งานวิจัยนี้กำหนดให้โปรแกรม irserv ใช้งานคลาส irserv ซึ่งเป็นคลาสหลักของโปรแกรมในการสร้างอินสแตนซ์ของบริการชื่อ irserv1 ตามส่วนต่อประสานชื่อ IDL:EQIR/extend_ir:1.0 (รายละเอียดกล่าวอยู่ในบทที่ 3) คลาส irserv จะสร้างและจัดเก็บข้อมูลอ้างอิงของบริการ irserv1 ลงในแฟ้มข้อมูล c:\thesis\ior\irserv.ior ทั้งนี้เพื่อให้โปรแกรมของผู้รับบริการนำข้อมูลในแฟ้มข้อมูลนี้ไปใช้ในกระบวนการสร้างการเชื่อมต่อตลอดจนเรียกใช้งานบริการค้นหาได้ต่อไป แฟ้มข้อมูล irserv.ior นี้จะถูกดาวน์โหลดไปที่เครื่องคอมพิวเตอร์ A เพื่อใช้ในการสร้างการเชื่อมต่อระหว่างบริการเทรดเดอร์และบริการค้นหาเช่นเดียวกัน งานวิจัยนี้กำหนดให้บริการ irserv1 รอรับคำร้องขอใช้บริการจากโปรแกรมผู้รับบริการที่พอร์ตหมายเลข 1600

เมื่อเรียกใช้งานโปรแกรม irserv คลาส irserv จะใช้เมทอด loadconfig ในการโหลดข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูล c:\thesis\config\config.eq ลงในคุณลักษณะ eqinfo ของคลาส irserv (คุณลักษณะนี้จะมีชนิดของข้อมูลเป็นเวคเตอร์) ซึ่งภายในแต่ละเอลิเมนท์ของคุณสมบัติ eqinfo จะใช้จัดเก็บข้อมูลความสัมพันธ์ระหว่างบริการ 2 บริการใดๆที่ถูกกำหนดค่าโดยผู้ให้บริการแทนที่ ข้อมูลความสัมพันธ์นี้จะถูกแทนความหมายโดยใช้คลาส eqmapping ตามรูปที่ 5.18

```
<start>
IDL:service_by_id/account_by_id:1.0,ser1,155.7.1.2
IDL:service_by_name/account_by_name:1.0,ser2,155.7.1.2
c:\thesis\ior\map1.ior
::service_by_id::account_by_id,4
searchby,string,id
area,string,NSEWC
cost,float,30.00
responsetime,float,40.00
<end>
```

รูปที่ 5.17 การกำหนดข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการ ser1 และ ser2

บริการค้นหาจะทำการค้นหาบริการแทนที่จากข้อมูลความสัมพันธ์ที่ถูกจัดเก็บอยู่ภายในคุณลักษณะ eqinfo ของคลาส irserv ตามเงื่อนไขที่โปรแกรมของผู้รับบริการระบุมาในเมทอดต่างๆที่ถูกเรียกใช้งาน อาทิเช่นเมทอด search_equ เป็นต้น



รูปที่ 5.18 การแทนข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการในบริการค้นหา

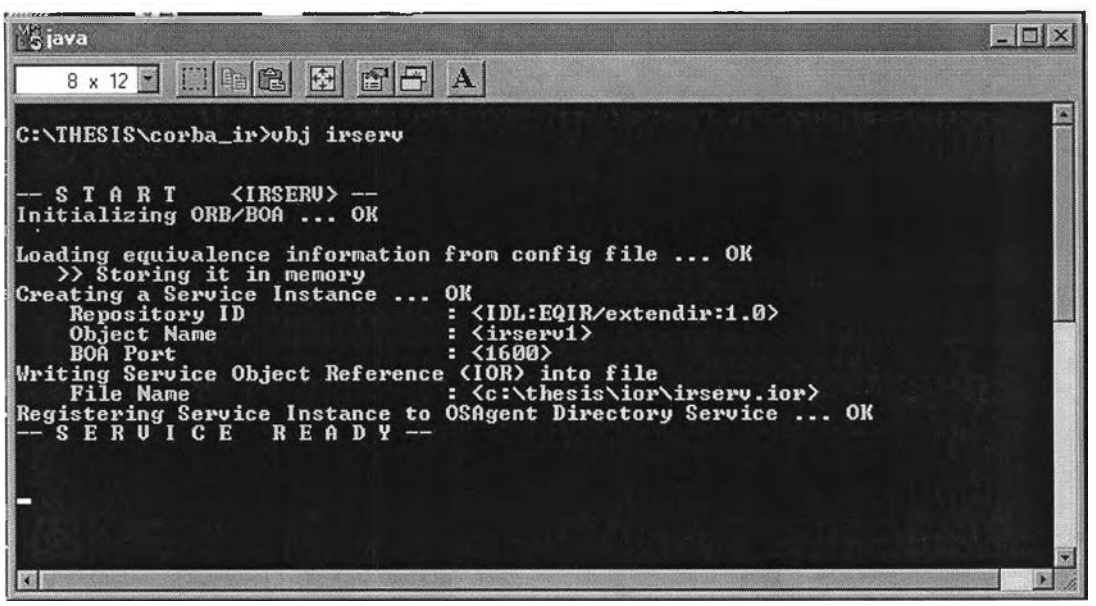
ขั้นตอนการทดสอบโปรแกรมของผู้รับบริการที่เกิดขึ้นภายหลังการเรียกใช้โปรแกรมพรีโพรเซสเซอร์เพื่อทำการแทรกคำสั่งการทำงานแล้วเป็นดังต่อไปนี้

1. ลบข้อเสนอบริการ ser1 ออกจากบริการเทรดเดอร์ที่ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ A

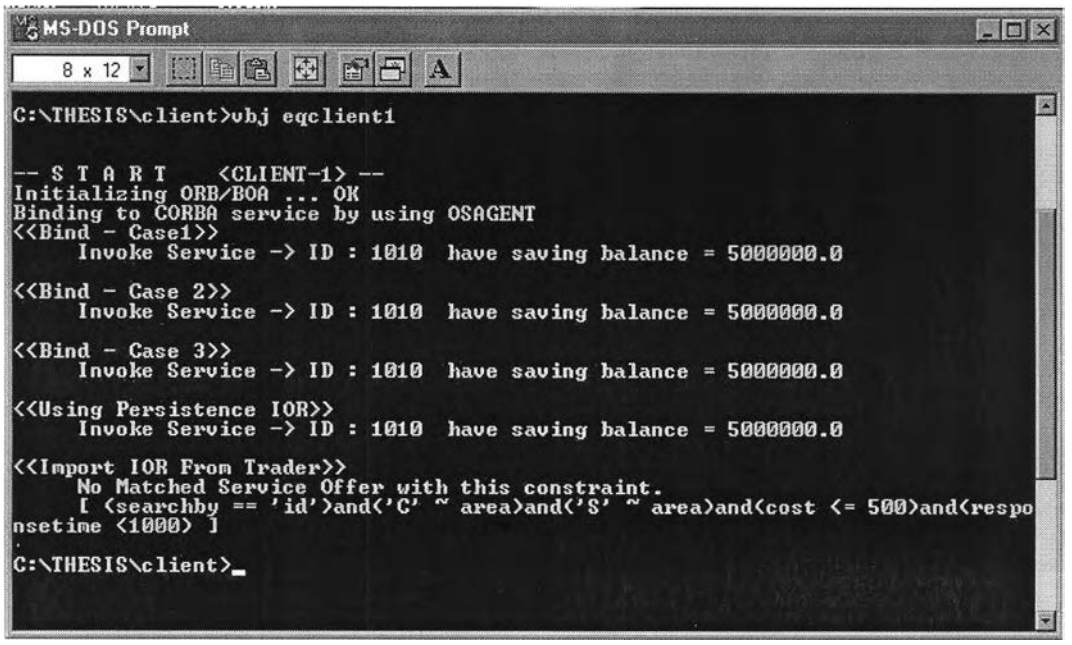
2. เรียกใช้โปรแกรม irserv เพื่อสร้างอินสแตนซ์ของบริการค้นหา (irserv1) บนเครื่องคอมพิวเตอร์ B โดยใช้คำสั่ง *c:\thesis\corba_ir\vbj irserv*

บริการ irserv1 จะถูกลงทะเบียนประกาศการให้บริการไปยังสมาร์ทเอเจนท์ ผลการเรียกใช้งานโปรแกรม irserv จะแสดงรายละเอียดของการทำงานต่างๆบนหน้าจอ ตามรูปที่ 5.19

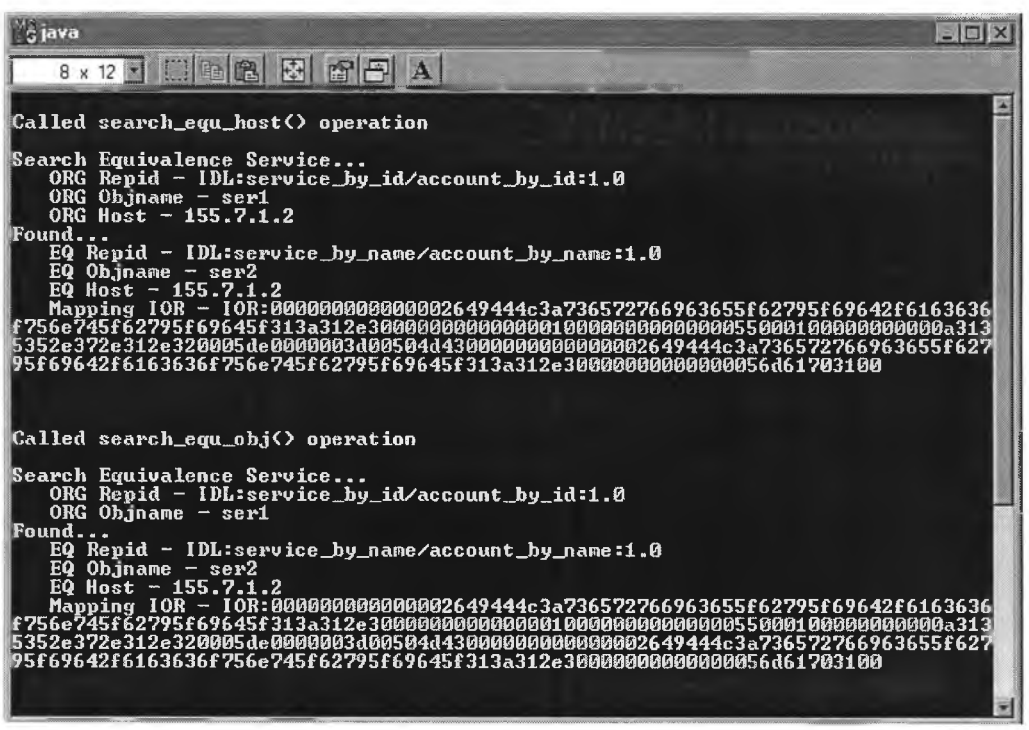
3. ที่เครื่องคอมพิวเตอร์ A ยังคงมีโปรแกรมผู้ให้บริการ servicebyname และโปรแกรม mapserv1 ทำงานอยู่ จากนั้นให้ทำการคอมไพล์และเรียกใช้โปรแกรมของผู้รับบริการ eqclient1.java สังเกตผลลัพธ์ในการค้นหาบริการ ser1 ของโปรแกรม eqclient1 ที่เกิดขึ้น ในขณะที่โปรแกรม servicebyid ไม่ได้ถูกเรียกใช้งาน (บริการ ser1 ไม่พร้อมให้บริการ) ผลการทำงานของโปรแกรมผู้รับบริการ eqclient1 จะเป็นตามรูปที่ 5.20



รูปที่ 5.19 ผลการเรียกใช้งานโปรแกรม irserv ซึ่งเป็นบริการค้นหา



รูปที่ 5.20 ผลการเรียกใช้งานโปรแกรม eqclient1 ที่ผ่านการแทรกคำสั่งแล้ว



รูปที่ 5.21 การค้นหาบริการแทนที่ของบริการค้นหา

รูปที่ 5.22 map1 ทำการแปลงคำร้องขอใช้บริการและเรียกใช้บริการ ser2

จากการทดสอบสามารถสรุปผลการทำงานของโปรแกรมผู้รับบริการภายหลังจากที่ได้ผ่านการแทรกคำสั่งโดยโปรแกรมพรีโพรเซสเซอร์แล้วว่าโปรแกรมของผู้รับบริการใหม่จะสามารถเรียกใช้งานบริการ ser2 เพื่อใช้งานแทนที่บริการ ser1 ผ่านทางตัวดำเนินการแปลง map1 ได้ในขณะที่บริการ ser1 ไม่พร้อมที่จะให้บริการ (รูปที่ 5.20-5.23) การแทนที่ในลักษณะนี้จะเกิดขึ้นเมื่อโปรแกรมของผู้รับบริการใช้การค้นหาบริการที่ต้องการผ่านทางคำสั่ง bind ทั้ง 3 รูปแบบหรือใช้การเรียกผ่านข้อมูลอ้างอิงของบริการภายในแฟ้มข้อมูล อันแสดงให้เห็นว่าการแทรกคำสั่งด้วยโปรแกรมพรีโพรเซสเซอร์สามารถทำให้คอร์บารองรับการแทนที่ของบริการในระดับอินสแตนซ์เพิ่มเติมได้ อย่างไรก็ตามโปรแกรมของผู้รับบริการจะไม่ได้รับข้อเสนอบริการ ser2 ไปใช้งานแทนที่ข้อเสนอบริการ ser1 เมื่อบริการเทรดเดอร์ไม่พบข้อเสนอบริการตามเงื่อนไขที่อิมพอร์ตเตอร์ระบุมาในเมทอด query ในกรณีที่โปรแกรมของผู้รับบริการทำการค้นหาบริการที่ต้องการผ่านทางบริการเทรดเดอร์ปกติที่ไม่สนับสนุนการแทนที่บริการในระดับอินสแตนซ์

รูปที่ 5.23 บริการ ser2 ทำงานแทนที่บริการ ser1

ในขั้นตอนถัดไปจะเน้นการทดสอบไปยังบริการเทรดเดอร์ใหม่ที่ได้ผ่านการแก้ไขโปรแกรมให้สามารถรองรับการเรียกใช้งานข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูล config.eq ได้ การทดสอบนี้จะสมมติให้โปรแกรมของผู้รับบริการทำการเรียกใช้บริการเทรดเดอร์โดยทำการอิมพอร์ตข้อเสนอบริการที่มีชนิดของบริการเป็น ::service_by_id::account_by_id จากเทรดเดอร์ งานวิจัยนี้จะสมมติให้ไม่มีเอ็กซ์พอร์ตเตอร์ใดเลยมาลงทะเบียนประกาศการให้บริการชนิดนี้กับบริการเทรดเดอร์ การทดสอบในขั้นตอนนี้จะแสดงให้เห็นถึงผลลัพธ์ของการอิมพอร์ตข้อเสนอบริการจากบริการเทรดเดอร์ที่แตกต่างกันเมื่อบริการเทรดเดอร์ถูกเรียกใช้งานโดยทำการระบุหรือไม่ระบุตัวเลือก –eq ทั้งนี้ บริการเทรดเดอร์ที่ถูกเรียกใช้งานโดยระบุตัวเลือก –eq จะมีความสามารถในการค้นหาข้อเสนอบริการใดๆ ที่ทำงานได้เท่าเทียมกันกับข้อเสนอบริการที่มีชนิดของบริการและคุณสมบัติต่างๆตามค่าที่ผู้ใช้งานระบุมาในเมทอด query (รายละเอียดการเรียกใช้บริการเทรดเดอร์ใหม่อยู่ในบทที่ 4) ขั้นตอนที่ใช้ในการทดสอบบริการเทรดเดอร์ใหม่จะเป็นดังนี้

1. ปิดและเรียกใช้บริการเทรดเดอร์อีกครั้งโดยระบุตัวเลือก –eq เพื่อกำหนดให้บริการเทรดเดอร์สนับสนุนความสามารถในการค้นหาข้อเสนอบริการแทนที่จากข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูล config.eq ด้วย ผู้ใช้งานบริการเทรดเดอร์จะเรียกใช้คำสั่งดังนี้

java com.ooc.CosTrading.Server –i -OAport 5000 –eq

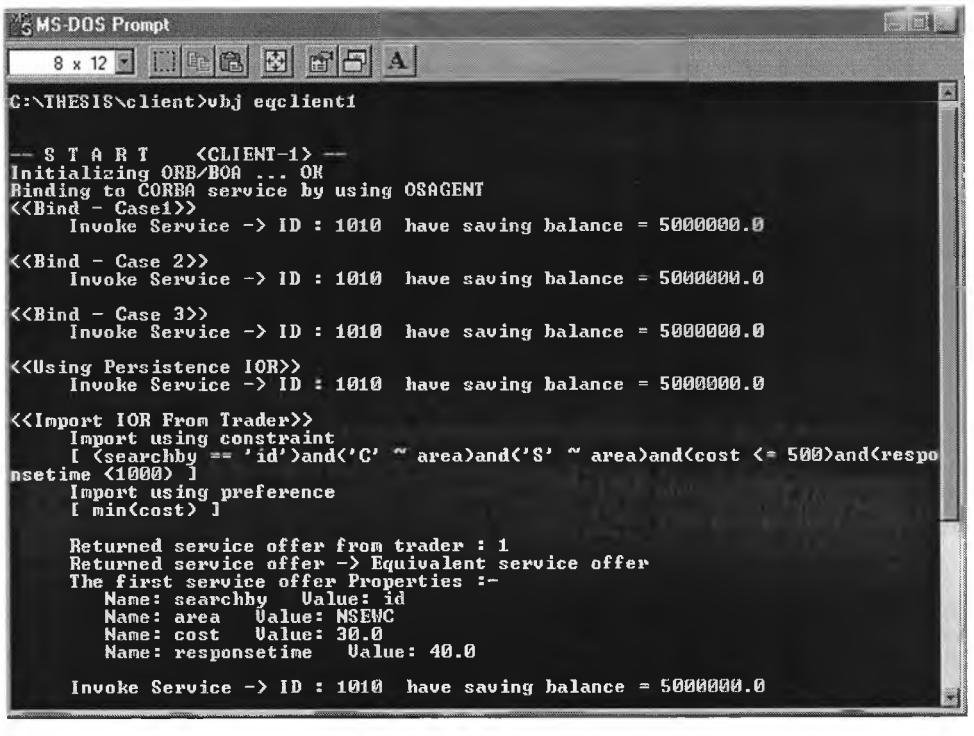
2. เพิ่มชนิดของบริการ ::service_by_id::account_by_id ลงในคลังชนิดบริการของเทรดเดอร์

3. เรียกใช้โปรแกรมของผู้ให้บริการ servicebyname ให้ทำการเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์โดยกำหนดรายละเอียดตามข้อมูลในตารางที่ 5.2 ผู้ใช้งานโปรแกรมจะเรียกคำสั่งดังนี้ *c:\thesis\service2\vbj servicebyname –trader*

4. เรียกใช้โปรแกรม mapserv1 เพื่อสร้างอินสแตนซ์ map1 สำหรับทำหน้าที่เป็นตัวดำเนินการแปลงด้วยคำสั่ง *c:\thesis\mapping1\vbj mapserv1*

5. เรียกใช้โปรแกรม irserv เพื่อสร้างอินสแตนซ์และให้บริการค้นหา (irserv1) ด้วยคำสั่ง *c:\thesis\corba_ir\vbj irserv*

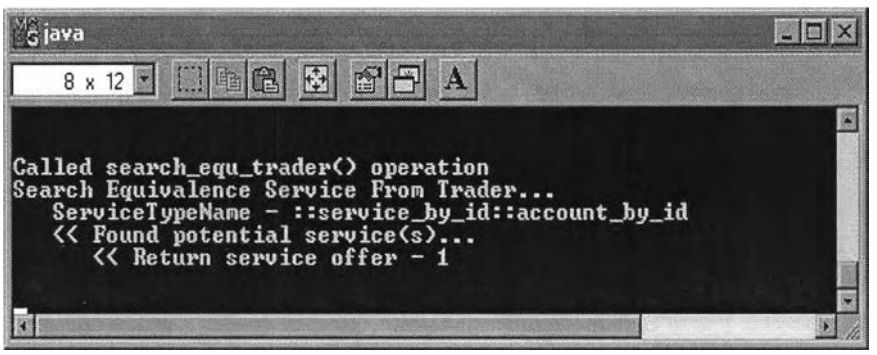
6. เรียกใช้โปรแกรมของผู้รับบริการให้ทำการอิมพอร์ตข้อเสนอบริการ ser1 จากบริการเทรดเดอร์โดยระบุชนิดของบริการเป็น ::service_by_id::account_by_id (รูปที่ 5.24) การทดสอบบริการเทรดเดอร์ในครั้งนี้จะพบว่าโปรแกรมของผู้รับบริการสามารถทำการอิมพอร์ตข้อเสนอบริการแทนที่จากบริการเทรดเดอร์ได้ทั้งนี้ผลการทำงานของโปรแกรมผู้รับบริการจะเป็นดังรูปที่ 5.24



รูปที่ 5.24 ผลการอิมพอร์ตข้อเสนอบริการจากบริการเทรดเดอร์ที่ผ่านการแก้ไขโปรแกรมแล้ว

เมื่อบริการเทรดเดอร์ไม่พบข้อเสนอบริการใดเลยที่สอดคล้องกับเงื่อนไขที่ผู้ใช้งานระบุมาในเมทอด query บริการเทรดเดอร์จะทำการเชื่อมต่อไปยังบริการค้นหา (รูปที่ 5.25) ในเครื่องคอมพิวเตอร์ B และทำการเรียกใช้เมทอด search_equ_trader (รายละเอียดอยู่ในบทที่ 4) เพื่อให้บริการค้นหาส่งกลับกลุ่มของ

ข้อเสนอบริการที่สามารถทำงานแทนที่ข้อเสนอบริการที่ผู้ใช้งานระบุมาได้ บริการเทรดเดอร์จะทำการคัดเลือกข้อเสนอบริการที่รับมาจากบริการค้นหาเพื่อส่งกลับเฉพาะข้อเสนอบริการที่มีคุณสมบัติตรงตามเงื่อนไขการค้นหาที่ผู้ใช้งานระบุมาในรูปของภาษาอธิบายเงื่อนไขกลับไปยังโปรแกรมของผู้รับบริการต่อไป จากผลการทดสอบดังกล่าวข้างต้นสามารถสรุปผลได้ว่าบริการเทรดเดอร์ใหม่ที่ได้ผ่านการแก้ไขโปรแกรมให้สนับสนุนการนำข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการมาใช้ในกระบวนการอิมพอร์ตข้อเสนอบริการในงานวิจัยนี้จะช่วยเพิ่มโอกาสในการค้นพบข้อเสนอบริการที่มีคุณสมบัติตรงตามความต้องการของผู้รับบริการมากยิ่งขึ้น



รูปที่ 5.25 บริการเทรดเดอร์เรียกใช้เมทอด search_equ_trader ของบริการค้นหา

ตัวอย่างทดสอบการแทนที่กันของบริการในระดับอินสแตนซ์เมื่อบริการทั้ง 2 มีส่วนต่อประสานที่แตกต่างกันในหัวข้อนี้ จะสามารถสรุปผลการทดสอบกลไกส่วนขยายของคอร์บาที่ถูกพัฒนาขึ้นได้ดังรายละเอียดในตารางที่ 5.4

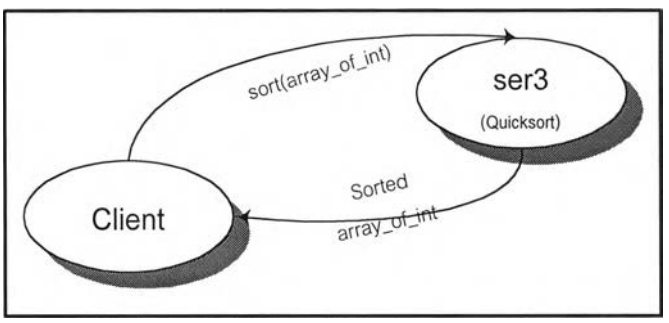
ตารางที่ 5.4 สรุปผลการทดสอบกลไกส่วนขยายของคอร์บาจากตัวอย่างการทดสอบที่ 5.1

ขั้นตอน	บริการที่ทำงาน	บริการที่หยุดทำงาน	เรียกโปรแกรมผู้รับบริการ	ผลที่เกิดขึ้นกับโปรแกรมของผู้รับบริการ
1	เครื่องคอมพิวเตอร์ A :- บริการ Trader <ปกติ> เครื่องคอมพิวเตอร์ B :- บริการ ser1 <ระบุให้ Export ข้อเสนอบริการไปยังเทรดเดอร์ด้วย>	-	Client1	สามารถเรียกใช้บริการผ่านทาง 1. คำสั่ง bind ทั้ง 3 รูปแบบ 2. ใช้การแปลงข้อมูลอ้างอิงของบริการจากแฟ้มข้อมูล c:\thesis\ior\ser1.ior 3. อิมพอร์ตข้อเสนอบริการจากเทรดเดอร์
2	เครื่องคอมพิวเตอร์ A :- บริการ Trader <ปกติ>	เครื่องคอมพิวเตอร์ B :- บริการ ser1	Client1	ไม่สามารถเรียกใช้บริการ ser1 ได้ในทุกกรณี <เกิด Exception ขึ้น>
3	เครื่องคอมพิวเตอร์ A :- บริการ Trader <ปกติ> เครื่องคอมพิวเตอร์ B :- ตัวดำเนินการแปลง map1 บริการ ser2 <ระบุให้ Export ข้อเสนอบริการไปยังเทรดเดอร์ด้วย>	เครื่องคอมพิวเตอร์ B :- บริการ ser1	Client1	สามารถเรียกใช้บริการ ser2 แทนที่บริการ ser1 ผ่านตัวดำเนินการแปลง map1 ได้เฉพาะกรณีที่เรียกใช้คำสั่ง bind รูปแบบที่ 1 เท่านั้น <คอร์บารองรับการแทนที่บริการในระดับชนิด>

4	เครื่องคอมพิวเตอร์ A :- ลบข้อเสนอบริการ ser1 ออกจากบริการ Trader <ปกติ> เครื่องคอมพิวเตอร์ B :- ตัวดำเนินการแปลง map1 บริการ ser2 บริการค้นหา <irserv1>	เครื่องคอมพิวเตอร์ B :- บริการ ser1	Eqclient1	สามารถเรียกใช้บริการผ่านทาง 1. คำสั่ง bind ทั้ง 3 รูปแบบ 2. ใช้การแปลงข้อมูลอ้างอิงของบริการจากแฟ้มข้อมูล c:\thesis\ior\ser1.ior หมายเหตุ บริการเทรดเดอร์ <ปกติ> ไม่พบข้อเสนอบริการตามที่อิมพอร์ตเตอร์ต้องการ และไม่ทำการค้นหาข้อเสนอบริการอื่นที่เท่าเทียมกันจากบริการค้นหา <No Matched Service Offer>
5	เครื่องคอมพิวเตอร์ A :- ปิดและเรียกใช้บริการ Trader <ระบุตัวเลือกให้เทรดเดอร์รองรับการค้นหาบริการแทนที่> เครื่องคอมพิวเตอร์ B :- ตัวดำเนินการแปลง map1 บริการ ser2 บริการค้นหา	เครื่องคอมพิวเตอร์ B :- บริการ ser1 <Add Service Type ไปยังบริการเทรดเดอร์อีกครั้ง>	Eqclient1	สามารถเรียกใช้บริการผ่านทาง 1. คำสั่ง bind ทั้ง 3 รูปแบบ 2. ใช้การแปลงข้อมูลอ้างอิงของบริการจากแฟ้มข้อมูล c:\thesis\ior\ser1.ior 3. อิมพอร์ตข้อเสนอบริการจากเทรดเดอร์ หมายเหตุ เทรดเดอร์ส่งกลับข้อเสนอบริการแทนที่ (ส่วน Object Reference เป็นข้อมูลอ้างอิงของตัวดำเนินการแปลง map1) ไปยังอิมพอร์ตเตอร์

5.2 ตัวอย่างทดสอบการแทนที่กันของบริการในระดับอินสแตนซ์เมื่อบริการทั้ง 2 มีส่วนต่อประสานเดียวกันและเป็นบริการแบบไม่มีสถานะ[19]

ตัวอย่างนี้จะสมมติให้มีบริการ ser3 ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ B โดยบริการนี้จะถูกพัฒนาขึ้นตามส่วนต่อประสาน IDL:sorting:1.0 บริการ ser3 จะให้บริการเรียงลำดับ (Sorting) ข้อมูลที่เป็นเลขจำนวนเต็มซึ่งถูกจัดเก็บอยู่ภายในอาเรย์ (Array of Integer) จากค่าน้อยเรียงไปหามากโดยใช้อัลกอริทึม (Algorithm) แบบ Quick Sort โปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้งานบริการ ser3 นี้ได้ตามรูปที่ 5.26



รูปที่ 5.26 โปรแกรมผู้รับบริการ client เรียกใช้บริการ ser3

นอกจากบริการ ser3 แล้วตัวอย่างนี้ยังกำหนดให้มีบริการ ser4 ซึ่งเป็นบริการที่ถูกพัฒนาขึ้นด้วยส่วนต่อประสานเดียวกันกับบริการ ser3 ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ B บริการ ser4 จะทำหน้าที่ให้บริการเรียงลำดับข้อมูลที่เป็นเลขจำนวนเต็มเช่นเดียวกันกับบริการ ser3 แตกต่างกันตรงที่บริการ ser4 จะใช้อัลกอริทึมแบบ Insertion Sort แทนการใช้อัลกอริทึมแบบ Quick Sort ซึ่งจะมีผลทำให้บริการ ser4 ใช้เวลาในการประมวลผลเพื่อเรียงลำดับข้อมูลนานกว่าบริการ ser3 เมื่อต้องทำการเรียงลำดับข้อมูลที่มีจำนวนเอลิเมนท์ภายในอาเรย์เท่ากัน[20] ถึงแม้ว่าทั้งบริการ ser3 และ ser4 จะมีส่วนต่อประสานเดียวกันก็ตาม ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมของบริการทั้งสองนี้ด้วยวิธีการที่แตกต่างกัน (Different Implementation) ตามอัลกอริทึมที่ใช้ บริการทั้งสองจะรับข้อมูลที่อยู่ในรูปของอาเรย์ของเลขจำนวนเต็มจากผู้รับบริการเพื่อนำมาเรียงลำดับข้อมูลจากค่าน้อยไปมากก่อนที่จะส่งกลับข้อมูลที่ผ่านการเรียงลำดับแล้วนั้นกลับไปยังผู้รับบริการอีกครั้ง การทำงานของบริการทั้ง 2 นี้จะเป็นแบบไม่มีสถานะ

---

[19] บริการแบบไม่มีสถานะ (Stateless Service) คือบริการใดๆที่มีการรับข้อมูลเข้ามาเพื่อใช้ในการประมวลผลและจัดส่งผลลัพธ์ที่ได้กลับไปยังผู้เรียกใช้บริการโดยไม่มีการจัดเก็บหรือเปลี่ยนแปลงสถานะหรือข้อมูลใดๆภายในบริการนั้น

[20] การเรียงลำดับข้อมูลด้วยอัลกอริทึมแบบ Quick Sort จะใช้เวลาในการประมวลผลโดยเฉลี่ย (Average Running Time) เป็น O(N log N) ในขณะที่อัลกอริทึมแบบ Insetion Sort จะใช้เวลาในการประมวลผลโดยเฉลี่ยถึง $O(N^2)$ [15]

การทดสอบในหัวข้อนี้มีเป้าหมายที่จะแสดงให้เห็นว่ากลไกส่วนขยายของคอร์บาที่ถูกพัฒนาขึ้นในงานวิจัยนี้จะช่วยทำให้เกิดการแทนที่กันของบริการในระดับอินสแตนซ์เมื่อบริการทั้งสองมีส่วนต่อประสานเดียวกันแต่ใช้การพัฒนาโปรแกรมที่แตกต่างกันได้ โปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้บริการ ser4 ให้ทำการเรียงลำดับข้อมูลแทนการเรียกใช้บริการ ser3 ได้เมื่อบริการ ser3 ไม่พร้อมที่จะให้บริการ

การทดสอบในหัวข้อนี้จะใช้งานเครื่องคอมพิวเตอร์ A สำหรับให้บริการเทรดเดอร์ของคอร์บา งานการพัฒนาโปรแกรมประยุกต์ของผู้ให้บริการและผู้รับบริการต่างๆจะกระทำอยู่ภายในเครื่องคอมพิวเตอร์ B ทั้งหมด (ดังรูปที่ 5.27) งานวิจัยนี้ได้ทำการพัฒนาโปรแกรมในส่วนของผู้ให้บริการโดยใช้ภาษาจาวาทั้งสิ้น 2 โปรแกรมดังรายละเอียดต่อไปนี้

1. **โปรแกรมผู้ให้บริการ** qsort

เป็นโปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นสำหรับให้บริการ ser3 กับผู้รับบริการใดๆตามข้อกำหนดของคอร์บา บริการ ser3 นี้จะมีชื่อส่วนต่อประสานเป็น IDL:sorting:1.0 (รายละเอียดของส่วนต่อประสานนี้อยู่ในรูปที่ 5.28) ตัวอย่างนี้จะสมมติให้บริการ ser3 ทำหน้าที่ให้บริการเรียงลำดับข้อมูลที่เป็นเลขจำนวนเต็มซึ่งถูกจัดเก็บอยู่ภายในอาเรย์จากค่าน้อยเรียงไปหามากโดยใช้อัลกอริทึมแบบ Quick Sort โปรแกรมของผู้รับบริการจะสามารถเรียกใช้งานบริการ ser3 นี้ได้โดยใช้การเรียกไปยังเมทอด sort และส่งผ่านอาเรย์ของเลขจำนวนเต็มเป็นพารามิเตอร์ให้กับเมทอด

งานวิจัยนี้ได้ทำการพัฒนาโปรแกรมขึ้นภายในแฟ้มข้อมูล qsort.java โดยจัดสร้างคลาส qsort[21] ขึ้นเป็นคลาสหลักของโปรแกรมเพื่อทำงานในหน้าที่ต่างๆดังต่อไปนี้

- รับตัวเลือก -trader จากผู้ใช้งานโปรแกรมผ่านทางค่าพารามิเตอร์อินพุตเพื่อจะระบุให้คลาส qsort ทำการเอ็กซ์พอร์ตข้อเสนอบริการของบริการ ser3 ไปยังบริการเทรดเดอร์หรือไม่ ถ้าผู้ใช้งานโปรแกรม qsort ไม่ได้ระบุตัวเลือกนี้ในขณะเรียกใช้งานโปรแกรมจะมีผลทำให้คลาส qsort ทำการลงทะเบียนประกาศการให้บริการ ser3 ไปยังสมาร์ทเอเจนท์เท่านั้น

- สร้างและกำหนดค่าโดยปริยายให้กับออร์บและออบเจ็กต์อะแดปเตอร์ภายในโปรแกรมของผู้ให้บริการ (งานวิจัยนี้กำหนดให้หมายเลขพอร์ตที่ใช้รับคำร้องขอใช้บริการมีค่าเท่ากับ 1503)

- สร้างและกำหนดชื่ออินสแตนซ์ของบริการ ser3 เพื่อทำหน้าที่ให้บริการกับผู้รับบริการโดยอินสแตนซ์ของบริการนี้จะใช้ชื่อส่วนต่อประสานเป็น IDL:sorting:1.0

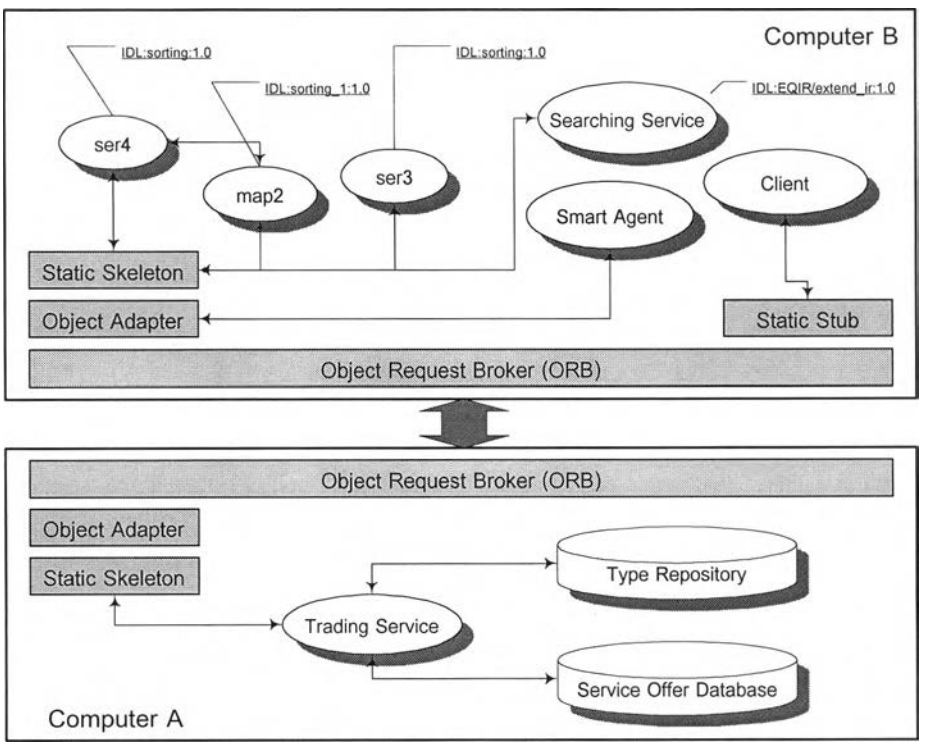
- เขียนข้อมูลอ้างอิงของอินสแตนซ์บริการ ser3 ลงในแฟ้มข้อมูล c:\thesis\ior\ser3.ior

---

[21] คลาส qsort จะสืบทอดคุณลักษณะจาก _sortingImplBase ซึ่งเป็นคลาสสเคเลตันที่ได้จากการคอมไพล์ส่วนต่อประสานตามรูปที่ 5.28

- ถ้าผู้ใช้ระบุตัวเลือก –trader คลาส qsort จะทำการเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์ที่ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ A โดยใช้ข้อมูลตามรายละเอียดในตารางที่ 5.5

- เมื่อทำงานในขั้นตอนดังกล่าวแล้วเสร็จ คลาส qsort จะทำการลงทะเบียนประกาศการให้บริการ ser3 ไปยังสมาร์ทเอเจนท์ผ่านการเรียกใช้เมทอด obj_is_ready จากนั้นจึงทำการลูปรอรับคำร้องขอใช้บริการจากผู้รับบริการใดๆโดยใช้การเรียกไปยังเมทอด impl_is_ready ของออบเจ็กต์อะแดปเตอร์



รูปที่ 5.27 รูปแบบของบริการตลอดจนส่วนประกอบต่างๆที่ใช้ในการทดสอบตัวอย่างที่ 5.2

```
//------------------------------------------------------------------------
// Repository ID : IDL:sorting:1.0
// IDL File : c:\thesis\service3\qsort.idl
//------------------------------------------------------------------------
typedef sequence<long> l_ub_seq;
interface sorting {
      l_ub_seq sort(in l_ub_seq a);
};
```

รูปที่ 5.28 ส่วนต่อประสานของบริการ ser3 และ ser4

ตารางที่ 5.5 ข้อมูลที่ใช้ในการเอ็กซ์พอร์ตข้อเสนอบริการ ser3 ไปยังบริการเทรดเดอร์

ชื่อชนิดบริการ (Service Type Name)	::sorting		
ชื่อส่วนต่อประสาน (Interface Name)	IDL:sorting:1.0		
คุณสมบัติของบริการ ser3 (Service Property)	datatype	string	"integer"
	algorithm	string	"quicksort"
	cost	float	10.00
	max_element	long	20000
ข้อมูลอ้างอิงของบริการ ser3 (Object Reference of Ser3)	IOR:000000000000001049444c3a736f7274696e673a312e3000000000010000000000000003d000100000000000a3135352e372e312e320005df0000002500504d4300000000000001049444c3a736f7274696e673a312e3000000000057365723300 **ความหมายจากการถอดรหัสข้อมูลอ้างอิงของบริการ** Interoperable Object Reference Type ID: IDL:sorting:1.0 Contains 1 profile. Profile 0-IIOP Profile: Version: 1.0 Host: 155.7.1.2 Port: 1503 Object Key: PersistentId [repId=IDL:sorting:1.0,objectName=ser3]		

จากรูปที่ 5.28 เมื่อผู้รับบริการเรียกใช้เมทอด sort ของบริการ ser3 โดยทำการผ่านค่าพารามิเตอร์เป็นอาเรย์ของเลขจำนวนเต็มชุดใดๆที่ยังไม่ได้ผ่านการเรียงลำดับมายังเมทอด sort บริการ ser3 จะนำค่าพารามิเตอร์นี้ไปทำการตรวจสอบจำนวนของเอลิเมนท์ภายในอาเรย์ให้มีจำนวนไม่เกินค่าขอบเขตสมาชิก (max_element) ซึ่งบริการนี้จะสามารถประมวลผลได้ ค่าขอบเขตสมาชิกนี้จะเป็นคุณสมบัติหนึ่งของบริการ ser3 ที่จะถูกเอ็กซ์พอร์ตไปยังบริการเทรดเดอร์โดยค่านี้จะถูกเอ็กซ์พอร์ตในรูปของค่า max_element เมื่อตรวจสอบค่าพารามิเตอร์นี้แล้วเสร็จ เมทอด sort จะทำการเรียงลำดับข้อมูลซึ่งเป็นเอลิเมนท์ภายในอาเรย์จากตามลำดับจากค่าน้อยไปหามากโดยใช้การเรียกไปยังเมทอด quicksort ของคลาส qsort เพื่อทำงานตามอัลกอริทึม Quick Sort ต่อไป

2. **โปรแกรมผู้ให้บริการ** insertionsort

เป็นโปรแกรมประยุกต์ที่ถูกพัฒนาขึ้นสำหรับให้บริการ ser4 กับผู้รับบริการใดๆตามข้อกำหนดคอร์บา ตัวอย่างนี้จะสมมติให้บริการ ser4 ทำหน้าที่ให้บริการเรียงลำดับข้อมูลที่เป็นเลขจำนวนเต็มเช่น

เดียวกันกับบริการ ser3 แตกต่างกันตรงที่บริการ ser4 จะใช้อัลกอริทึมแบบ Insertion Sort บริการ ser4 มีส่วนต่อประสานชื่อเดียวกับบริการ ser3 คือ IDL:sorting:1.0 ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมภายในแฟ้มข้อมูล insertionsort.java ด้วยภาษาจาวาโดยจัดสร้างคลาส insertionsort ขึ้นให้เป็นคลาสหลักของโปรแกรมเพื่อทำงานในหน้าที่ต่างๆที่คล้ายคลึงกับคลาส qsort แต่จะมีความแตกต่างกันในรายละเอียดบางอย่างดังต่อไปนี้

- กำหนดให้คลาส insertionsort ใช้หมายเลขพอร์ตเป็น 1504 ในการรับคำร้องขอใช้บริการ

- คลาส insertionsort จะเขียนข้อมูลอ้างอิงของอินสแตนซ์บริการ ser4 ลงในแฟ้มข้อมูล c:\thesis\ior\ser4.ior ภายหลังจากที่ได้จัดสร้างอินสแตนซ์ของบริการแล้ว

- ถ้าผู้ใช้งานโปรแกรมระบุตัวเลือก –trader เมื่อเรียกใช้งานโปรแกรม insertionsort คลาส insertionsort จะทำการเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์ที่ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ A โดยใช้ข้อมูลตามรายละเอียดในตารางที่ 5.6

ตารางที่ 5.6 ข้อมูลที่ใช้ในการเอ็กซ์พอร์ตข้อเสนอบริการ ser4 ไปยังบริการเทรดเดอร์

ชื่อชนิดบริการ (Service Type Name)	::sorting		
ชื่อส่วนต่อประสาน (Interface Name)	IDL:sorting:1.0		
คุณสมบัติของบริการ ser4 (Service Property)	datatype	string	"integer"
	algorithm	string	"insertionsort"
	cost	float	5.00
	max_element	long	10000
ข้อมูลอ้างอิงของบริการ ser4 (Object Reference of Ser4)	IOR:000000000000001049444c3a736f7274696e673a312e3000 0000000100000000000003d000100000000000a3135352e372 e312e320005e0000002500504d430000000000000001049444c 3a736f7274696e673a312e3000000000057365723400 **<u>ความหมายจากการถอดรหัสข้อมูลอ้างอิงของบริการ</u>** Interoperable Object Reference Type ID: IDL:sorting:1.0 Contains 1 profile. Profile 0-IIOP Profile: Version: 1.0 Host: 155.7.1.2 Port: 1504 Object Key: PersistentId [repId=IDL:service_by_name/account_by_name:1.0 ,objectName=ser4]		

งานวิจัยนี้จะกำหนดให้บริการ ser3 และ ser4 มีความสัมพันธ์ในลักษณะที่เท่าเทียมกัน กล่าวคือ ทั้งบริการ ser3 และ ser4 จะให้บริการตามส่วนต่อประสานเดียวกันกับโปรแกรมของผู้รับบริการแตกต่างกันตรงที่บริการทั้ง 2 นี้จะใช้รูปแบบในการพัฒนาโปรแกรมด้วยอัลกอริทึมที่แตกต่างกัน ถ้าโปรแกรมของผู้รับบริการต้องการเรียกใช้บริการ ser3 แต่บริการ ser3 ไม่พร้อมที่จะให้บริการ โปรแกรมของผู้รับบริการจะสามารถเรียกใช้บริการ ser4 เพื่อทำงานแทนที่ ser3 ได้โดยบริการทั้ง 2 นี้จะส่งกลับข้อมูลที่ผ่านการเรียงลำดับแล้วด้วยค่าที่เหมือนกันกลับไปให้โปรแกรมของผู้รับบริการ (รูปที่ 5.29)

สมมติให้โปรแกรมผู้รับบริการส่งผ่านค่าพารามิเตอร์ต่อไปนี้ไปยังเมทอด sort ของบริการ ser3 และ ser4
{100,23,20,78,16,10,1,4,46,120,90,44,77,12,12}
ทั้งบริการ ser3 และ ser4 จะส่งกลับข้อมูลที่ผ่านการเรียงลำดับแล้ว (มีค่าเหมือนกัน) กลับไปยังผู้รับบริการ
{1,4,10,12,12,16,20,23,44,46,77,78,90,100,120}

รูปที่ 5.29 ตัวอย่างผลการเรียงลำดับข้อมูลโดยใช้บริการ ser3 และ ser4

นอกจากโปรแกรมทั้ง 2 โปรแกรมที่ได้กล่าวถึงไปแล้วข้างต้น งานวิจัยนี้ยังได้จัดสร้างคลาส mapserv2 ขึ้นภายในแฟ้มข้อมูล mapserv2.java เพื่อทำงานเป็นตัวดำเนินการแปลง map2 สำหรับใช้ดักและส่งผ่านคำร้องขอใช้บริการที่โปรแกรมของผู้รับบริการส่งไปยังบริการ ser3 ให้ไปเรียกใช้งานบริการ ser4 ได้โดยตรงในลักษณะของการส่งผ่านแบบไม่ทำการแปลง (Direct Mapping) บริการ map2 จะมีส่วนต่อประสานเป็น IDL:sorting_1:1.0 โดยส่วนต่อประสานนี้จะเป็นส่วนต่อประสานลูกซึ่งได้รับการสืบทอดคุณสมบัติจากส่วนต่อประสาน sorting ของบริการ ser3 นั่นเอง (รายละเอียดของส่วนต่อประสานนี้อยู่ในรูปที่ 5.30) คลาส mapserv2 จะทำงานตามขั้นตอนต่างๆคล้ายคลึงกับคลาส qsort แต่จะมีความแตกต่างกันในรายละเอียดบางอย่างดังนี้

- คลาส mapserv2 จะใช้หมายเลขพอร์ต 1505 ในการดักคำร้องขอใช้บริการที่โปรแกรมผู้รับบริการส่งไปยังบริการ ser3

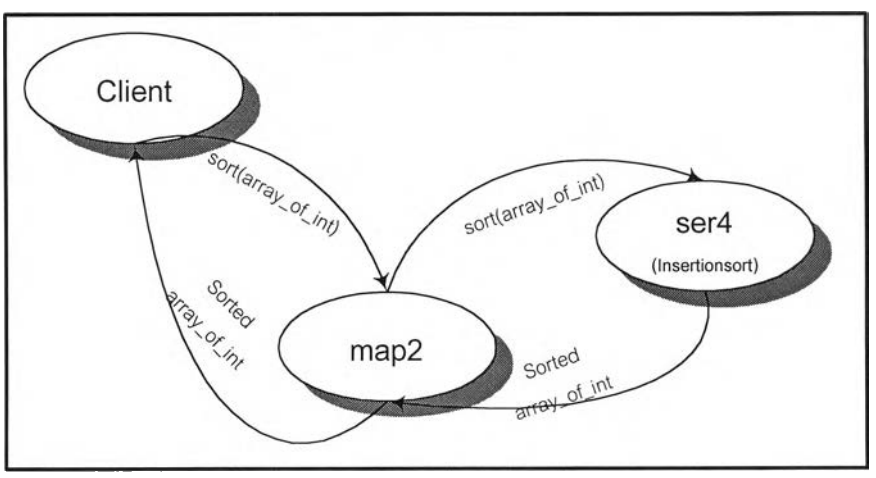
- คลาส mapserv2 จะเขียนข้อมูลอ้างอิงของอินสแตนซ์บริการ map2 ลงในแฟ้มข้อมูล c:\thesis\ior\map2.ior ภายหลังจากที่ได้จัดสร้างอินสแตนซ์ของบริการแล้ว

- คลาส mapserv2 ซึ่งถูกใช้งานเป็นตัวดำเนินการแปลงจะทำการลงทะเบียนประกาศการให้บริการ map2 ไปยังสมาร์ทเอเจนท์เท่านั้น บริการ map2 จะไม่เอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์ อย่างไรก็ตามโปรแกรมของผู้รับบริการใดๆจะสามารถเรียกใช้งานตัวดำเนินการแปลง map2 นี้ได้อย่างโปร่งใส

ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมภายในเมทอด sort ของบริการ map2 ให้ทำการเรียกไปยังเมทอด sort ของบริการ ser4 เพื่อทำการเรียงลำดับข้อมูลที่ถูกส่งมาเป็นค่าพารามิเตอร์ (อาเรย์ของเลขจำนวนเต็ม) ของเมทอด sort โดยโปรแกรมของผู้รับบริการ เมื่อบริการ ser4 เสร็จสิ้นกระบวนการเรียงลำดับข้อมูลในอาเรย์แล้วบริการ ser4 จะส่งกลับอาเรย์ที่ผ่านการเรียงลำดับแล้วนั้นกลับมายังบริการ map2 ซึ่งบริการ map2 จะส่งผ่านข้อมูลที่ได้รับมานี้กลับไปให้โปรแกรมของผู้รับบริการอีกทอดหนึ่ง ทั้งนี้กลไกการแทนที่กันของบริการ ser3 และ ser4 จะเกิดขึ้นอย่างโปร่งใสตามรูปที่ 5.31

```
//----------------------------------------------------------------------------
// Repository ID : IDL:sorting_1:1.0
// IDL File : c:\thesis\mapping2\mapserv2.idl
//----------------------------------------------------------------------------
typedef sequence<long> l_ub_seq;
interface sorting {
      l_ub_seq sort(in l_ub_seq a);
};
interface sorting_1  : sorting {
};
```

รูปที่ 5.30 ส่วนต่อประสานของบริการ map2



รูปที่ 5.31 การใช้งานบริการ ser4 ผ่านทางตัวดำเนินการแปลง map2

ตัวอย่างนี้ได้ทำการพัฒนาโปรแกรมของผู้รับบริการขึ้นภายในแฟ้มข้อมูล client2.java โดยใช้ภาษาจาวา ทั้งนี้โปรแกรม client2 จะใช้งานคลาส client2 ในการเรียกใช้บริการ ser3 ด้วยวิธีการต่างๆเช่นเดียวกันกับโปรแกรม client1 ของตัวอย่างการทดสอบที่ 5.1 คือ

- ใช้การเรียกผ่านคำสั่ง bind ทั้ง 3 รูปแบบของซอฟต์แวร์วิสิโบรคเกอร์ (ตารางที่ 5.7)

ตารางที่ 5.7 การเรียกใช้งานบริการ ser3 ผ่านคำสั่ง bind ในโปรแกรมผู้รับบริการ client2

รูปแบบคำสั่ง	**คำสั่งและความหมาย**
bind รูปแบบที่ 1	*org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);* *sorting ser3 = sortingHelper.bind(orb);* <u>ความหมายของคำสั่ง bind</u> โปรแกรมของผู้รับบริการต้องการเรียกใช้อินสแตนซ์ของบริการใดๆที่มีส่วนต่อประสานเป็น IDL:sorting:1.0
bind รูปแบบที่ 2	*org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);* *String objname="ser3";* *sorting ser3 = sortingHelper.bind(orb,obname);* <u>ความหมายของคำสั่ง bind</u> โปรแกรมของผู้รับบริการต้องการเรียกใช้อินสแตนซ์ของบริการชื่อ ser3 โดยบริการนี้จะต้องมีส่วนต่อประสานเป็น IDL:sorting:1.0
bind รูปแบบที่ 3	*org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,null);* *String objname="ser3";* *String host = "155.7.1.2";* *org.omg.CORBA.BindOptions bindopt = new org.omg.CORBA.BindOptions();* *sortingser3 = sortingHelper.bind(orb,objname,host,bindopt);* <u>ความหมายของคำสั่ง bind</u> โปรแกรมของผู้รับบริการต้องการเรียกใช้อินสแตนซ์ของบริการชื่อ ser3 โดยอินสแตนซ์ของบริการนี้จะต้องทำงานอยู่ภายในโฮสต์ที่มีเลขที่อยู่ไอพีเป็น 155.7.1.2 และมีส่วนต่อประสานเป็น IDL:sorting:1.0 เท่านั้น

- ใช้การเรียกผ่านทางข้อมูลอ้างอิงของบริการที่ถูกจัดเก็บอยู่ภายในแฟ้มข้อมูล ser3.ior

- ใช้การอิมพอร์ตข้อเสนอบริการ ser3 จากบริการเทรดเดอร์ในเครื่องคอมพิวเตอร์ A โดยกำหนดเงื่อนไขการค้นหาเท่ากับ

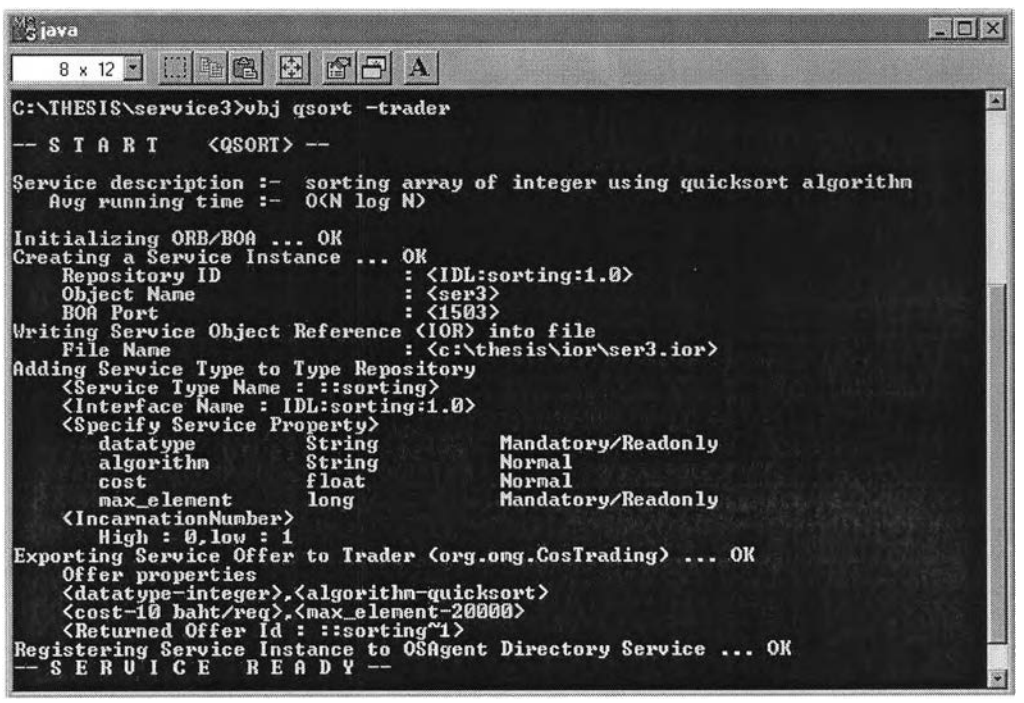
(algorithm=='quicksort')and(datatype == 'integer')and(cost <= 20)and(max_element >=1000)

ขั้นตอนการทดสอบตลอดจนผลการทำงานของโปรแกรมผู้รับบริการที่เกิดขึ้น ก่อนที่จะถูกแทรกคำสั่งโดยโปรแกรมพรีโพรเซสเซอร์เป็นดังต่อไปนี้

1. เรียกใช้บริการเทรดเดอร์ที่ติดตั้งอยู่ในเครื่องคอมพิวเตอร์ A โดยใช้คำสั่ง

java com.ooc.CosTrading.Server -i -OAport 5000

2. เรียกใช้สมาร์ทเอเจนท์และโปรแกรมผู้ให้บริการ qsort เพื่อสร้างอินสแตนซ์และให้บริการ ser3 โดยใช้คำสั่ง *c:\thesis\service3\vbj qsort – trader* ผลการเรียกใช้งานโปรแกรมของผู้ให้บริการ qsort จะแสดง รายละเอียดของการทำงานต่างๆบนหน้าจอตามรูปที่ 5.32



รูปที่ 5.32 ผลการเรียกใช้งานโปรแกรมผู้ให้บริการ qsort

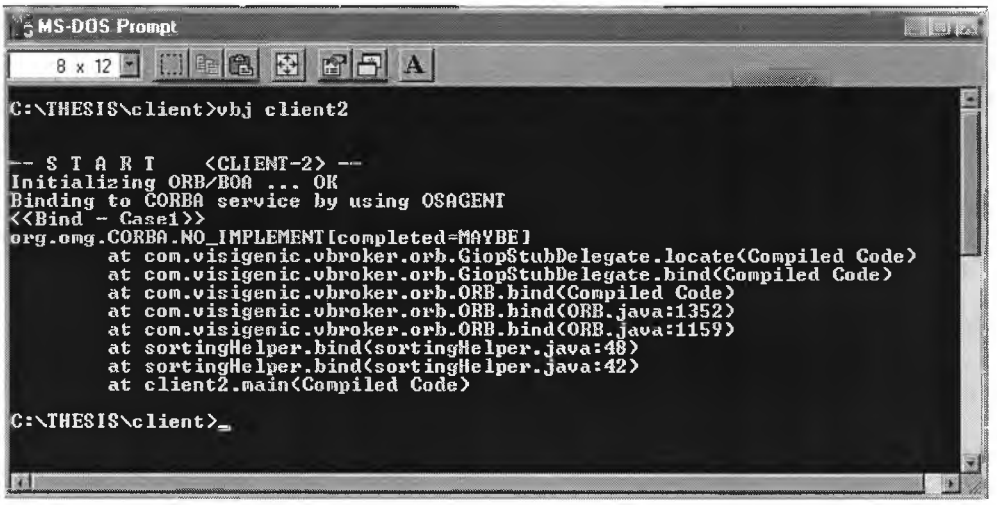
3. เรียกใช้โปรแกรมของผู้รับบริการ client2 ในเครื่องคอมพิวเตอร์ B เพื่อสังเกตผลที่เกิดขึ้นจากการค้นหาอินสแตนซ์ของบริการ ser3 โดยใช้การเรียกคำสั่ง bind ทั้ง 3 รูปแบบรวมทั้งการเรียกใช้บริการ ser3 จากข้อมูลอ้างอิงของบริการในแฟ้มข้อมูล c:\thesis\ior\ser3.ior ตลอดจนการอิมพอร์ตข้อเสนอบริการ

จากบริการเทรดเดอร์ คำสั่งในการเรียกใช้โปรแกรมผู้รับบริการจะเป็นดังนี้ *c:\thesis\client\vbj client2* (รูปที่ 5.33)

รูปที่ 5.33 ผลการเรียกใช้งานโปรแกรมผู้รับบริการด้วยคำสั่ง vbj client2

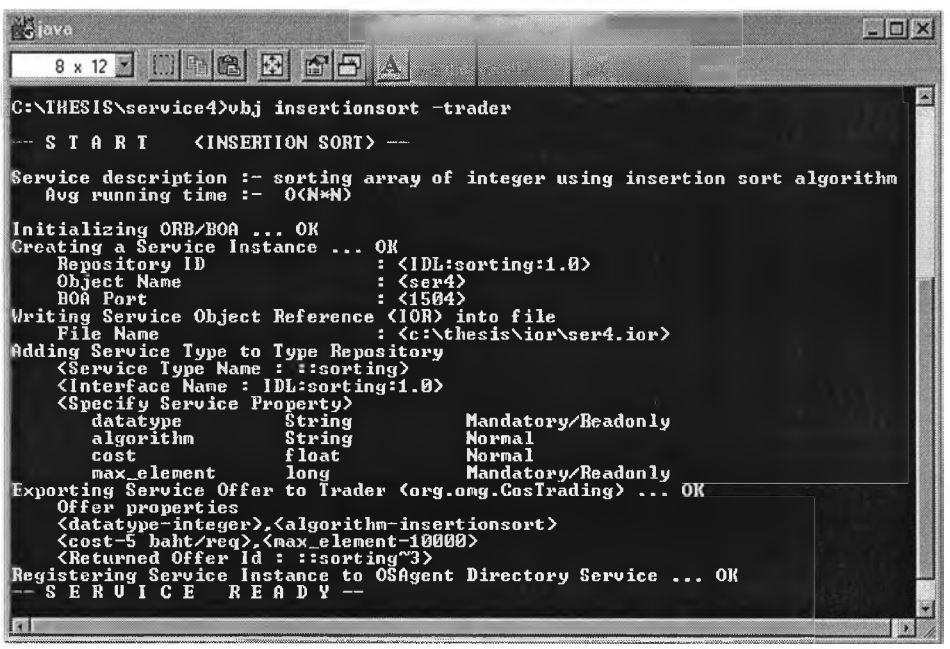
จากรูปที่ 5.33 จะพบว่าโปรแกรมของผู้รับบริการจะสามารถเรียกใช้งานบริการ ser3 เพื่อทำการเรียงลำดับเลขจำนวนเต็มค่า 50-1 ให้กลายเป็นค่าที่เรียงจากน้อยไปหามากคือ 1-50 ได้ไม่ว่าโปรแกรม client1 จะใช้การค้นหาอินสแตนซ์ของบริการ ser3 ด้วยวิธีการใดก็ตาม การเรียกใช้งานบริการ ser3 ในรูปแบบนี้จะเป็นการเรียกใช้บริการในคอร์บาตามรูปแบบปกติ

4. หยุดการทำงานของโปรแกรมผู้ให้บริการ qsort (โดยกด <Ctrl>+c) และทดลองเรียกใช้งานโปรแกรมผู้รับบริการ client2 อีกครั้ง สังเกตผลที่เกิดขึ้นจากการค้นหาอินสแตนซ์ของบริการ ser3 เมื่อโปรแกรมของผู้รับบริการเรียกใช้คำสั่ง bind ทั้ง 3 รูปแบบ ในขณะที่อินสแตนซ์ของบริการ ser3 ไม่พร้อมที่จะให้บริการกับผู้ร้องขอ โปรแกรมผู้รับบริการจะเกิดเอ็กซ์เซพชันขึ้นจากการเรียกคำสั่ง bind ดังรูปที่ 5.34

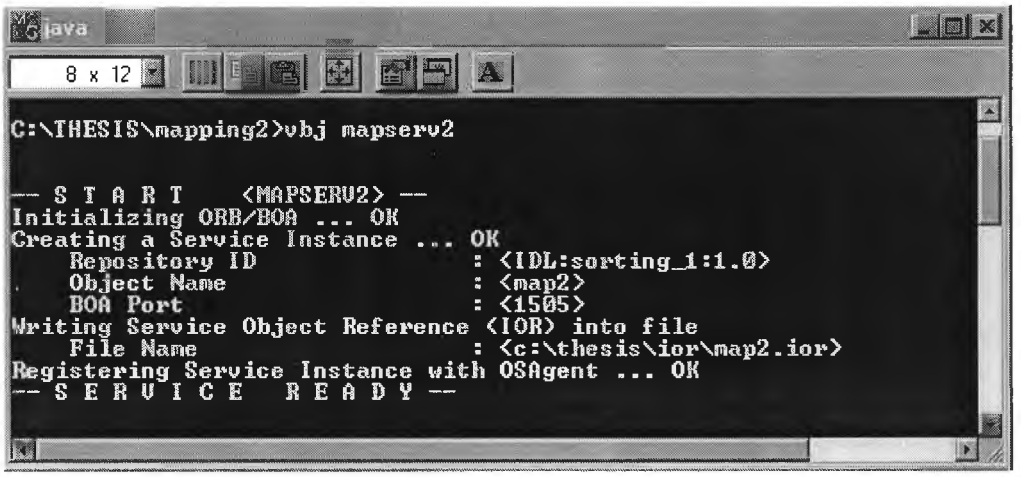


รูปที่ 5.34 ผลการเรียกใช้งานโปรแกรมผู้รับบริการเมื่อโปรแกรม qsort หยุดทำงาน

5. เรียกใช้โปรแกรมของผู้ให้บริการ insertionsort และโปรแกรม mapserv2 (รูปที่ 5.35 และ 5.36) พร้อมทำการเรียกใช้โปรแกรมของผู้รับบริการ client2 อีกครั้ง (รูปที่ 5.37) เพื่อสังเกตผลลัพธ์ที่เกิดขึ้น (ขณะนี้โปรแกรม qsort ยังคงหยุดการทำงานอยู่) ซึ่งผลที่ได้จะแสดงการแทนที่กันของบริการในระดับชนิด เมื่อโปรแกรมผู้รับบริการใช้การ bind รูปแบบที่ 1 เช่นเดียวกันกับการทดสอบในตัวอย่างที่ 5.1



รูปที่ 5.35 ผลการเรียกใช้งานโปรแกรมผู้ให้บริการ insertionsort



รูปที่ 5.36 ผลการเรียกใช้งานโปรแกรม mapserv2

รูปที่ 5.37 ผลการเรียกใช้งานโปรแกรม client2 เมื่อเกิดการแทนที่ของบริการในระดับชนิด

ตัวอย่างนี้ได้ทำการเรียกใช้โปรแกรมพรีโพรเซสเซอร์เพื่อทำการแทรกคำสั่งการทำงานลงในโปรแกรมของผู้รับบริการและจัดเก็บโปรแกรมที่ผ่านการแทรกคำสั่งแล้วลงในแฟ้มข้อมูลชื่อ

eqclient2.java (รูปที่ 5.38) จากนั้นจึงทำการคอมไพล์และทดลองเรียกใช้งานโปรแกรมของผู้รับบริการอีกครั้งเพื่อเปรียบเทียบผลการทำงานที่เกิดขึ้นว่าคำสั่งที่แทรกโดยโปรแกรมพรีโพรเซสเซอร์จะช่วยทำให้โปรแกรมผู้รับบริการสามารถเรียกใช้งานบริการอื่นๆที่มีส่วนต่อประสานเดียวกันแต่มีความแตกต่างในแง่ของการพัฒนาโปรแกรมเพื่อทำงานแทนที่บริการที่ต้องการได้หรือไม่

การทดสอบในส่วนนี้จะสมมติให้ผู้ให้บริการแทนที่ทำการกำหนดข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการ ser3 และ ser4 ลงในแฟ้มข้อมูล c:\thesis\config\config.eq (รูปที่ 5.39) ข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูลนี้จะถูกเรียกใช้งานโดยบริการค้นหาแทนการเรียกใช้ข้อมูลจากคลังจัดเก็บส่วนต่อประสานในงานวิจัย [4]

รูปที่ 5.38 การแทรกคำสั่งเพื่อสร้างโปรแกรม eqclient2 โดยโปรแกรมพรีโพรเซสเซอร์

```
<start>
IDL:sorting:1.0,ser3,155.7.1.2
IDL:sorting:1.0,ser4,155.7.1.2
c:\thesis\ior\map2.ior
::sorting,4
datatype,string,integer
algorithm,string,quicksort
cost,float,10.00
max_element,long,20000
<end>
```

รูปที่ 5.39 การกำหนดข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการ ser3 และ ser4

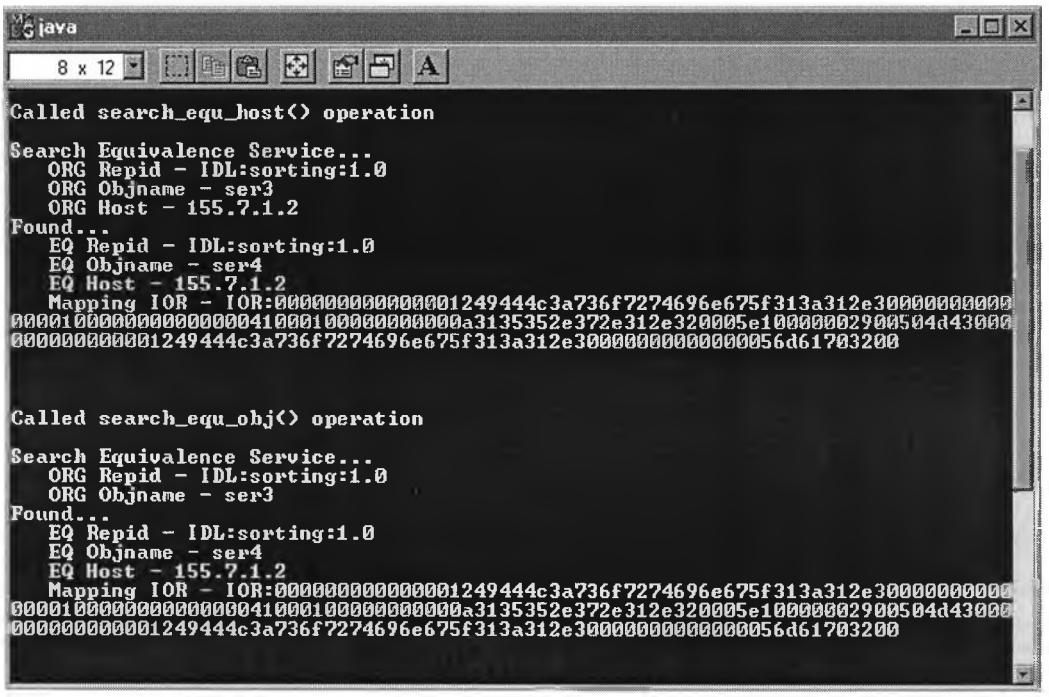
ขั้นตอนการทดสอบโปรแกรมของผู้รับบริการที่เกิดขึ้นภายหลังการเรียกใช้โปรแกรมพรีโพรเซสเซอร์เพื่อทำการแทรกคำสั่งการทำงานแล้วเป็นดังต่อไปนี้

1. ลบข้อเสนอบริการ ser3 ออกจากบริการเทรดเดอร์ที่ทำงานอยู่ภายในเครื่องคอมพิวเตอร์ A

2. เรียกใช้โปรแกรม irserv เพื่อสร้างอินสแตนซ์ของบริการค้นหา (irserv1) บนเครื่องคอมพิวเตอร์ B โดยใช้คำสั่ง *c:\thesis\corba_ir\vbj irserv*

3. ที่เครื่องคอมพิวเตอร์ A ยังคงมีโปรแกรมผู้ให้บริการ insertionsort และโปรแกรม mapserv2 ทำงานอยู่ จากนั้นให้ทำการคอมไพล์และเรียกใช้โปรแกรมของผู้รับบริการ eqclient2.java สังเกตผลลัพธ์ในการค้นหาบริการ ser3 ของโปรแกรม eqclient2 ที่เกิดขึ้น ในขณะที่โปรแกรม qsort ไม่ได้ถูกเรียกใช้งาน (บริการ ser3 ไม่พร้อมให้บริการ) ผลการทำงานของโปรแกรมผู้รับบริการ eqclient2 จะเป็นตามรูปที่ 5.40

รูปที่ 5.40 ผลการเรียกใช้งานโปรแกรม eqclient2 ที่ผ่านการแทรกคำสั่งแล้ว



รูปที่ 5.41 การค้นหาบริการแทนที่ของบริการค้นหา

รูปที่ 5.42 map2 ทำการดักคำร้องขอใช้บริการและเรียกใช้บริการ ser4

จากการทดสอบสามารถสรุปผลการทำงานของโปรแกรมผู้รับบริการภายหลังจากที่ได้ผ่านการแทรกคำสั่งโดยโปรแกรมพรีโพรเซสเซอร์แล้วว่าโปรแกรมของผู้รับบริการใหม่จะสามารถเรียกใช้งานบริการ ser4 เพื่อใช้งานแทนที่บริการ ser3 ผ่านทางตัวดำเนินการแปลง map2 ได้ในขณะที่บริการ ser3 ไม่พร้อมที่จะให้บริการ (รูปที่ 5.40-5.43) การแทนที่ในลักษณะนี้จะเกิดขึ้นเมื่อโปรแกรมของผู้รับบริการใช้การค้นหาบริการที่ต้องการผ่านทางคำสั่ง bind ทั้ง 3 รูปแบบหรือใช้การเรียกผ่านข้อมูลอ้างอิงของบริการภายในแฟ้มข้อมูล อันแสดงให้เห็นว่าการแทรกคำสั่งด้วยโปรแกรมพรีโพรเซสเซอร์สามารถทำให้คอร์บารองรับการแทนที่ของบริการในระดับอินสแตนซ์เพิ่มเติมได้ อย่างไรก็ตามโปรแกรมของผู้รับบริการจะไม่ได้รับข้อเสนอบริการ ser4 ไปใช้งานแทนที่ข้อเสนอบริการ ser3 เมื่อบริการเทรดเดอร์ไม่พบข้อเสนอบริการตามเงื่อนไขที่อิมพอร์ตเตอร์ระบุมาในเมทอด query ในกรณีที่โปรแกรมของผู้รับบริการทำการค้นหาบริการที่ต้องการผ่านทางบริการเทรดเดอร์ปกติที่ไม่สนับสนุนการแทนที่บริการในระดับอินสแตนซ์

รูปที่ 5.43 บริการ ser4 ทำงานแทนที่บริการ ser3

ในขั้นตอนถัดไปจะเน้นการทดสอบไปยังบริการเทรดเดอร์ใหม่ที่ได้ผ่านการแก้ไขโปรแกรมให้สามารถรองรับการเรียกใช้งานข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูล

config.eq ได้ การทดสอบนี้จะกำหนดให้โปรแกรมของผู้รับบริการทำการอิมพอร์ตข้อเสนอบริการที่มีชนิดของบริการเป็น ::sorting จากเทรดเดอร์โดยจะสมมติให้ไม่มีเอ็กซ์พอร์ตเตอร์ใดเลยมาลงทะเบียนประกาศการให้บริการชนิดนี้โดยระบุให้ค่าคุณสมบัติ algorithm ของบริการเท่ากับ "quicksort" การทดสอบนี้จะแสดงให้เห็นถึงผลลัพธ์ของการอิมพอร์ตข้อเสนอบริการจากบริการเทรดเดอร์ที่แตกต่างกันเมื่อบริการเทรดเดอร์ถูกเรียกใช้งานโดยทำการระบุหรือไม่ระบุตัวเลือก –eq ทั้งนี้บริการเทรดเดอร์ที่ถูกเรียกใช้งานโดยระบุตัวเลือก –eq จะมีความสามารถในการค้นหาข้อเสนอบริการใดๆที่ทำงานได้เท่าเทียมกันกับข้อเสนอบริการที่มีชนิดของบริการและคุณสมบัติต่างๆตามค่าที่ผู้ใช้งานระบุมาในเมทอด query ขั้นตอนที่ใช้ในการทดสอบบริการเทรดเดอร์ใหม่จะเป็นดังนี้

1. ปิดและเรียกใช้บริการเทรดเดอร์อีกครั้งโดยระบุตัวเลือก –eq เพื่อกำหนดให้บริการเทรดเดอร์สนับสนุนความสามารถในการค้นหาข้อเสนอบริการแทนที่จากข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการภายในแฟ้มข้อมูล config.eq ด้วย ผู้ใช้งานบริการเทรดเดอร์จะเรียกใช้คำสั่งดังนี้

java com.ooc.CosTrading.Server –i -OAport 5000 –eq

2. เพิ่มชนิดของบริการ ::sorting ลงในคลังชนิดบริการของเทรดเดอร์

3. เรียกใช้โปรแกรมของผู้ให้บริการ insertionsort ให้ทำการเอ็กซ์พอร์ตข้อเสนอบริการไปยังบริการเทรดเดอร์โดยกำหนดรายละเอียดตามข้อมูลในตารางที่ 5.6 ผู้ใช้งานโปรแกรมจะเรียกคำสั่งดังนี้

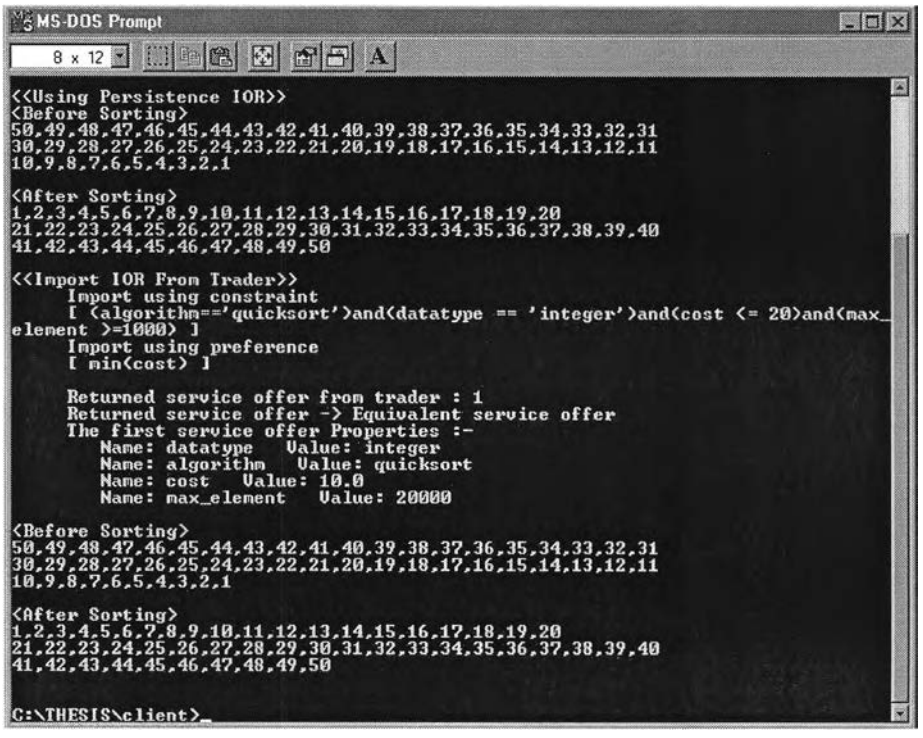
c:\thesis\service4\vbj insertionsort –trader

4. เรียกใช้โปรแกรม mapserv2 เพื่อสร้างอินสแตนซ์ map2 สำหรับทำหน้าที่เป็นตัวดำเนินการแปลงด้วยคำสั่ง *c:\thesis\mapping2\vbj mapserv2*

5. เรียกใช้โปรแกรม irserv เพื่อสร้างอินสแตนซ์และให้บริการค้นหา (irserv1) ด้วยคำสั่ง *c:\thesis\corba_ir\vbj irserv*

6. เรียกใช้โปรแกรมของผู้รับบริการให้ทำการอิมพอร์ตข้อเสนอบริการจากบริการเทรดเดอร์โดยระบุชนิดของบริการที่ต้องการเป็น ::sorting และกำหนดให้เงื่อนไขการค้นหาที่ใช้ในการอิมพอร์ตบริการจากเทรดเดอร์เป็นค่าดังนี้ *(algorithm=='quicksort') and (datatype == 'integer') and (cost <= 20) and (max_element >=1000)* จะพบว่าโปรแกรมของผู้รับบริการได้รับข้อเสนอบริการ ser4 จากบริการเทรดเดอร์มาใช้งานแทนที่ได้ตามรูปที่ 5.44

เมื่อบริการเทรดเดอร์ไม่พบข้อเสนอบริการใดเลยที่สอดคล้องกับเงื่อนไขที่ผู้ใช้งานระบุมาในเมทอด query บริการเทรดเดอร์จะทำการเรียกใช้เมทอด search_equ_trader ของบริการค้นหาเพื่อให้บริการค้นหาส่งกลับกลุ่มของข้อเสนอบริการที่สามารถทำงานแทนที่ข้อเสนอบริการที่ผู้ใช้งานระบุมาได้ บริการเทรดเดอร์จะทำการคัดเลือกข้อเสนอบริการที่รับมาจากบริการค้นหาเพื่อส่งกลับเฉพาะข้อเสนอบริการที่มีคุณสมบัติตรงตามเงื่อนไขการค้นหาที่ผู้ใช้งานระบุมาในรูปของภาษาอธิบายเงื่อนไขกลับไปยังโปรแกรมของผู้รับบริการต่อไป



รูปที่ 5.44 ผลการอิมพอร์ตข้อเสนอบริการจากบริการเทรดเดอร์ที่ผ่านการแก้ไขโปรแกรมแล้ว

จากผลการทดสอบดังกล่าวข้างต้นสามารถสรุปผลได้ว่าบริการเทรดเดอร์ใหม่ที่ได้ผ่านการแก้ไขโปรแกรมให้สนับสนุนการนำข้อมูลความสัมพันธ์แบบเท่าเทียมกันของบริการมาใช้ในกระบวนการอิมพอร์ตข้อเสนอบริการในงานวิจัยนี้จะช่วยเพิ่มโอกาสในการค้นพบข้อเสนอบริการที่มีคุณสมบัติตรงตามความต้องการของผู้รับบริการมากยิ่งขึ้นดังรายละเอียดการทดสอบในตารางที่ 5.8

ตารางที่ 5.8 สรุปผลการทดสอบกลไกส่วนขยายของคอร์บาจากตัวอย่างการทดสอบที่ 5.2

ขั้นตอน	บริการที่ทำงาน	บริการที่หยุดทำงาน	เรียกโปรแกรมผู้รับบริการ	ผลที่เกิดขึ้นกับโปรแกรมของผู้รับบริการ
1	เครื่องคอมพิวเตอร์ A :- บริการ Trader <ปกติ> เครื่องคอมพิวเตอร์ B :- บริการ ser3 <ระบุให้ Export ข้อเสนอบริการไปยังเทรดเดอร์ด้วย>	-	Client2	สามารถเรียกใช้บริการผ่านทาง 1. คำสั่ง bind ทั้ง 3 รูปแบบ 2. ใช้การแปลงข้อมูลอ้างอิงของบริการจากแฟ้มข้อมูล c:\thesis\ior\ser3.ior 3. อิมพอร์ตข้อเสนอบริการจากเทรดเดอร์
2	เครื่องคอมพิวเตอร์ A :- บริการ Trader <ปกติ>	เครื่องคอมพิวเตอร์ B :- บริการ ser3	Client2	ไม่สามารถเรียกใช้บริการ ser3 ได้ในทุกกรณี <เกิด Exception ขึ้น>
3	เครื่องคอมพิวเตอร์ A :- บริการ Trader <ปกติ> เครื่องคอมพิวเตอร์ B :- ตัวดำเนินการแปลง map2 บริการ ser4 <ระบุให้ Export ข้อเสนอบริการไปยังเทรดเดอร์ด้วย>	เครื่องคอมพิวเตอร์ B :- บริการ ser3	Client2	สามารถเรียกใช้บริการ ser4 แทนที่บริการ ser3 ผ่านตัวดำเนินการแปลง map2 ได้เฉพาะกรณีที่เรียกใช้คำสั่ง bind รูปแบบที่ 1 เท่านั้น <คอร์บารองรับการแทนที่บริการในระดับชนิด>

4	เครื่องคอมพิวเตอร์ A :- ลบข้อเสนอบริการ ser3 ออกจากบริการ Trader <ปกติ> เครื่องคอมพิวเตอร์ B :- ตัวดำเนินการแปลง map2 บริการ ser4 บริการค้นหา <irserv1>	เครื่องคอมพิวเตอร์ B :- บริการ ser3	Eqclient2	สามารถเรียกใช้บริการผ่านทาง 1. คำสั่ง bind ทั้ง 3 รูปแบบ 2. ใช้การแปลงข้อมูลอ้างอิงของบริการจากแฟ้มข้อมูล c:\thesis\ior\ser3.ior หมายเหตุ บริการเทรดเดอร์ <ปกติ> ไม่พบข้อเสนอบริการตามที่อิมพอร์ตเตอร์ต้องการ และไม่ทำการค้นหาข้อเสนอบริการอื่นที่เท่าเทียมกันจากบริการค้นหา <No Matched Service Offer>
5	เครื่องคอมพิวเตอร์ A :- ปิดและเรียกใช้บริการ Trader <ระบุตัวเลือกให้เทรดเดอร์รองรับการค้นหาบริการแทนที่> เครื่องคอมพิวเตอร์ B :- ตัวดำเนินการแปลง map2 บริการ ser4 บริการค้นหา	เครื่องคอมพิวเตอร์ B :- บริการ ser3 <Add Service Type ไปยังบริการเทรดเดอร์อีกครั้ง>	Eqclient2	สามารถเรียกใช้บริการผ่านทาง 1. คำสั่ง bind ทั้ง 3 รูปแบบ 2. ใช้การแปลงข้อมูลอ้างอิงของบริการจากแฟ้มข้อมูล c:\thesis\ior\ser3.ior 3. อิมพอร์ตข้อเสนอบริการจากเทรดเดอร์ หมายเหตุ เทรดเดอร์ส่งกลับข้อเสนอบริการแทนที่ (ส่วน Object Reference เป็นข้อมูลอ้างอิงของตัวดำเนินการแปลง map2) ไปยังอิมพอร์ตเตอร์