บทที่ 2
ทฤษฎีที่เกี่ยวข้องและแนวคิดการวิจัย

ในบทนี้จะกล่าวถึงทฤษฎีและงานวิจัยต่างๆ ที่เกี่ยวข้อง เพื่อใช้ในการออกแบบและพัฒนาต้นแบบของคลังชนิดของบริการที่รองรับความสัมพันธ์แบบเท่าเทียมกัน นอกจากนั้นยังกล่าวถึงแนวคิดในการทำวิจัยนี้ด้วย

2.1 แนวคิดทฤษฎี

มีดังนี้

- แนวคิดระบบกระจาย (Distributed System Concept)
- การเปลี่ยนรุ่นของบริการในสถาปัตยกรรมต่างๆของระบบกระจาย
- ความสัมพันธ์แบบซับไทป์ปิ้ง (Subtyping Relationship)
- ความสัมพันธ์แบบเท่าเทียมกัน (Equivalence Relationship)
- คลังชนิดของบริการ (Type Repository)

2.1.1 แนวคิดระบบกระจาย (Distributed System Concept)

ความหมายของระบบกระจาย [4] คือระบบที่ประกอบไปด้วยเครื่องคอมพิวเตอร์ หลายเครื่องเชื่อมต่อกันเป็นเครือข่าย โดยที่คอมพิวเตอร์เหล่านี้ทำงานเป็นอิสระจากกันและมีความแตกต่างในหลายๆด้านทั้งทางซอฟต์แวร์ ฮาร์ดแวร์ และภาษาการโปรแกรมที่ใช้ ถึงแม้ว่าจะเป็นอิสระจากกัน แต่เครื่องคอมพิวเตอร์เหล่านี้ก็ทำงานร่วมกัน โดยมีจุดประสงค์หลักคือการใช้ทรัพยากรร่วมกัน อย่างไรก็ตามการทำงานร่วมกันนี้ จะเน้นเพื่อให้เกิดความโปร่งใส (Transparency) ซึ่งก็คือลักษณะการทำงานที่เปรียบเสมือนบริการหรือทรัพยากรต่างๆที่ใช้ อยู่บนเครื่องคอมพิวเตอร์ของผู้ใช้ และถูกพัฒนาขึ้นมาสำหรับผู้ใช้นั้นๆโดยเฉพาะ ทั้งที่ในความเป็นจริงแล้วอาจอยู่ห่างกันและถูกพัฒนาภายใต้สภาพแวดล้อมที่แตกต่างกัน ซึ่งคุณลักษณะของความโปร่งใสก็ยังสามารถแยกเป็นประเด็นย่อยได้อีก แต่ประเด็นที่เกี่ยวข้องกับงานวิจัยนี้คือความโปร่งใสในการเข้าถึง (Access Transparency) และความโปร่งใสในตำแหน่งที่อยู่ (Location Transparency) โดยที่ความโปร่งใสทั้ง 2 ประเด็นนี้เป็นพื้นฐานที่จะมารองรับการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการ นั่นคือผู้ใช้บริการจะสามารถเข้าถึงบริการได้ในรูปแบบเดิมโดยไม่ทราบว่าบริการรุ่นเก่าถูกแทนที่ด้วยบริการรุ่นใหม่แล้ว รวมทั้งไม่ทราบด้วยว่าบริการนั้นอยู่ที่ใด

เมื่อนำแนวคิดเชิงวัตถุ (Object-Oriented Concept) มารองรับการทำงานของระบบกระจาย จะได้ระบบกระจายเชิงวัตถุ (Distributed Object System) เช่น คอร์บา [5 - 7] ดีคอม [8]

และอาร์เอ็ม-โอดีพี [9] ทำให้สามารถนำข้อดีของแนวคิดเชิงวัตถุมาปรับใช้กับระบบกระจายได้ เช่นการติดต่อระหว่างผู้ให้บริการและผู้รับบริการที่เป็นไปในเชิงวัตถุ ส่งผลให้การควบคุมการเข้าถึงทรัพยากรของผู้ให้บริการจากบริการเป็นไปอย่างมีระบบ อันเนื่องมาจากการทำงานในเชิงวัตถุมีการระบุถึงขอบเขตการใช้งานของทรัพยากรต่างๆ ผ่านทางส่วนต่อประสาน โดยไม่สนใจรายละเอียดการพัฒนา ดังนั้นจึงทำให้การทำงานขององค์ประกอบต่างๆในระบบกระจายทั้งผู้ให้บริการและผู้รับบริการเป็นไปได้ด้วยความยืดหยุ่น

คอร์บาเป็นระบบกระจายเชิงวัตถุที่มีการใช้กันอย่างแพร่หลายในปัจจุบันระบบหนึ่ง สาเหตุหนึ่งที่ทำให้คอร์บาเป็นที่นิยมเนื่องจากข้อกำหนดต่างๆ (Specification) ของคอร์บาถูกเขียนอยู่ในรูปแบบของภาษากำหนดนิยามส่วนต่อประสาน (Interface Definition Language - IDL) ซึ่งเป็นรูปแบบของภาษาที่เป็นกลาง ไม่ขึ้นอยู่กับภาษาการโปรแกรมภาษาใดภาษาหนึ่ง ทำให้ในขั้นตอนการทำให้เกิดผล (Implementation) จะได้เป็นองค์ประกอบแบบคอร์บา(CORBA Component) ที่สามารถใช้งานได้บนหลายระบบ (Portable) โดยไม่ยึดติดอยู่กับระบบปฏิบัติการ เครือข่าย ภาษาการโปรแกรม รวมถึงออร์บ (ORB – Object Request Broker) ที่เป็นผลิตภัณฑ์ต่างยี่ห้อกัน และจากการที่ภาษากำหนดนิยามส่วนต่อประสานของคอร์บามีต้นแบบมาจากภาษาซีพลัสพลัส (C++) ซึ่งเป็นภาษาการโปรแกรมในเชิงวัตถุอยู่แล้ว ดังนั้นการออกแบบส่วนต่อประสานตามภาษากำหนดนิยามส่วนต่อประสาน จึงสามารถกำหนดเป็นเชิงวัตถุได้ กล่าวได้ว่าสามารถนำข้อดีของแนวคิดเชิงวัตถุมาใช้ในระบบคอร์บาได้ [7]

2.1.2 การเปลี่ยนรุ่นของบริการในสถาปัตยกรรมต่างๆของระบบกระจาย

ปัจจุบันสถาปัตยกรรมระบบกระจายเชิงวัตถุ (Distributed Object Architectures) ถูกพัฒนาขึ้นมาในหลายรูปแบบ ซึ่งแต่ละรูปแบบต่างก็มีแนวทางที่สนับสนุนการเปลี่ยนรุ่นแตกต่างกันไป ในหัวข้อนี้จะกล่าวถึงสถาปัตยกรรมต่างๆในแง่มุมซึ่งเกี่ยวข้องกับงานวิจัยอันนี้ได้แก่การจัดเก็บข้อมูลชนิดของบริการในรุ่นต่างๆ และวิธีการที่ใช้ในการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการ

2.1.2.1 คอร์บา (CORBA - Common Object Request Broker Architecture)

คอร์บา (CORBA) [5] เป็นข้อกำหนด (Specification) ที่สนับสนุนการทำงานในระบบกระจายที่กำหนดโดยโอเอ็มจี (OMG - Object Management Group) โดยองค์ประกอบที่เกี่ยวข้องกับการจัดเก็บชนิดของบริการ (Service Type) ในคอร์บาคือคลังส่วนต่อประสาน (Interface Repository) ซึ่งทำหน้าที่จัดเก็บและดูแลการใช้ข้อมูลของส่วนต่อประสาน อันได้แก่ นิยามของส่วนต่อประสาน (Interface Definition) ซึ่งรวมถึงชื่อและรุ่นของส่วนต่อประสาน การดำเนินงาน

(Operation) ลักษณะประจำ (Attribute) และความสัมพันธ์แบบซับไทป์ปิ้งกับส่วนต่อประสานอื่น เป็นต้น ส่วนหนึ่งของนิยามส่วนต่อประสานสำหรับวัตถุ InterfaceDef ในคลังส่วนต่อประสานเป็นดังนี้

```
module CORBA {
        interface InterfaceDef;
        typedef sequence <InterfaceDef> InterfaceDefSeq;
        typedef sequence <RepositoryId> RepositoryIdSeq;
        typedef sequence <Operation Description> OpDescriptionSeq;
        typedef sequence <AttributeDescription> AttrDescriptionSeq;
        interface InterfaceDef : Container, Contained, IDLType {

                // read/write interface
                attribute InterfaceDefSeq base_interfaces;
                                ...
                struct FullInterfaceDescription {
                   Identifier name;                 // ชื่อ
                   RepositoryId id;                 // รหัสที่เป็นหนึ่งเดียว
                   RepositoryId defined_in;         // คอนเทนเนอร์ของวัตถุอินเทอร์เฟสเดฟนี้
                   VersionSpec version;             // รุ่น
                   OpDescriptionSeq Operations;     // รายการของการดำเนินงาน
                   AttrDescriptionSeq attributes;   // รายการของลักษณะประจำ
                   RepositoryIdSeq base_interfaces; // รายการของซูเปอร์ไทป์
                   TypeCode type;                   // รหัสสำหรับวัตถุอินเทอร์เฟสเดฟนี้
                };
                FullInterfaceDescription describe_interface();
                                ...
        };
};
```

คอร์บากำหนดให้รุ่นใหม่ของบริการจะต้องมีชื่อของส่วนต่อประสานเหมือนบริการเดิม ต่างกันที่หมายเลขรุ่น แต่อย่างไรก็ดีไม่ได้มีการกล่าวถึงการจัดการปัญหาการเปลี่ยนรุ่นแต่อย่างใด

[6] กล่าวคือไม่ได้มีการกำหนดว่า แนวทางการสนับสนุนการทำงานแทนที่กันได้ข้ามรุ่นเป็นอย่างไร ในกรณีใดที่บริการรุ่นใหม่จะใช้แทนรุ่นเก่าได้ หรือบริการจะถูกเปลี่ยนแปลงในลักษณะใดได้บ้าง จึงจะเรียกว่าเกิดบริการรุ่นใหม่ ไม่ใช่บริการใหม่ ในปัจจุบันบริการหนึ่งจะใช้แทนอีกบริการหนึ่งได้ จะต้องมีส่วนต่อประสานที่เป็นซับไทป์ นั่นคือบริการรุ่นใหม่จะต้องมีความเปลี่ยนแปลงในลักษณะที่มันกลายเป็นซับไทป์ของบริการรุ่นเก่าเท่านั้น จึงจะใช้แทนได้ แต่การเปลี่ยนแปลงในลักษณะนี้จะเป็นการเปลี่ยนแปลงภายในขอบเขตที่จำกัดตามกฎของความสัมพันธ์แบบซับไทป์ปิ้ง บริการรุ่นใหม่ที่ได้จากการเปลี่ยนแปลงอย่างอิสระ (Arbitrary Change)[3] จะไม่สามารถใช้แทนที่บริการรุ่นเก่าได้

2.1.2.2 ดีซีอี (DCE - Distributed Computing Environment)

ดีซีอี [10] เป็นเทคโนโลยีสำหรับการทำงานของระบบกระจายที่กำหนดโดยโอเอสเอ็ฟ (OSF - Open Software Foundation) และมีส่วนที่สนับสนุนการเปลี่ยนรุ่นของชนิดของบริการคือ ส่วนต่อประสานของวัตถุจะระบุโดยใช้ยูยูไอดี (UUID - Universal Unique Identifier) ร่วมกับหมายเลขรุ่น (Version Number) สำหรับหมายเลขรุ่นจะอยู่ในรูปแบบของ หมายเลขหลัก.หมายเลขรอง (major.minor) โดยที่หมายเลขรองจะเพิ่มขึ้นเมื่อส่วนต่อประสานรุ่นใหม่ถูกเพิ่มเติม แต่มิได้แก้ไขสิ่งที่มีอยู่เดิม (Incremental Change) จากการใช้ยูยูไอดีร่วมกับหมายเลขรุ่นทำให้ดีซีอีสนับสนุนการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการได้ โดยเมื่อผู้รับบริการระบุยูยูไอดีและหมายเลขรุ่นของส่วนต่อประสานของบริการที่ต้องการมาให้ หากไม่มีอินสแตนซ์ของบริการนั้นที่จะให้บริการได้ดีซีอีจะทำการค้นหาบริการที่สามารถใช้แทนกันได้ โดยการพิจารณาว่าบริการใดสามารถใช้แทนกันได้ จะพิจารณาจาก 2 ประเด็นคือ

1. เมื่อยูยูไอดีและหมายเลขหลักตรงกัน
2. เมื่อหมายเลขรองของบริการที่จะนำมาใช้แทน มีค่ามากกว่าหรือเท่ากับหมายเลขรองของบริการที่ต้องการ

แม้ว่าดีซีอีจะสนับสนุนการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการ แต่ก็ยังไม่มีคลังชนิดของบริการ (Type Repository) สำหรับเก็บนิยามของส่วนต่อประสานเหมือนกับ

---

[3]การเปลี่ยนบริการรุ่นใหม่ตามแบบซับไทป์ หมายถึงบริการรุ่นใหม่ยังคงต้องมีการดำเนินงานและพารามิเตอร์ ต่างๆของบริการรุ่นเก่าไว้ แต่อาจเพิ่มการดำเนินงานใหม่ได้ (Incremental Change) ในขณะที่**การเปลี่ยนบริการรุ่นใหม่แบบอิสระ** (Arbitrary Change) อาจเป็นการเปลี่ยนชื่อของการดำเนินงานเปลี่ยนจำนวนพารามิเตอร์ หรือชนิดของพารามิเตอร์ที่มีอยู่ในบริการรุ่นเก่าก็ได้ เป็นต้น

สถาปัตยกรรมอื่น รวมทั้งการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการก็ยังทำได้อย่างจำกัด โดยที่บริการรุ่นใหม่จะต้องเปลี่ยนในลักษณะที่เป็นการเพิ่มเติมเท่านั้นไม่ใช่การแก้ไข จึงจะใช้แทนบริการรุ่นเก่าได้ การตรวจสอบการใช้แทนกันได้ (Substitutability) จะพิจารณาจากยูยูไอดีและหมายเลขรุ่นเท่านั้น โดยมิได้ตรวจสอบรายละเอียดการทำงานของบริการอย่างแท้จริง

2.1.2.3 ดีคอม (DCOM - Distributed Component Object Model)

ดีคอมเป็นแบบจำลองวัตถุแบบกระจาย (Distributed Object Models) ที่ขยายออกมาจากคอม (COM - Component Object Model) โดยทำให้วัตถุแบบคอมที่อยู่บนเครื่องคอมพิวเตอร์ที่ต่างกัน สามารถสื่อสารกันได้ [8] คอมสนับสนุนเรื่องการเปลี่ยนรุ่นด้วยวิธีการเพิ่มส่วนต่อประสานใหม่รวมไปในวัตถุเดิมโดยไม่ไปกระทบกับส่วนต่อประสานเดิมคือสามารถใช้ส่วนต่อประสานได้ทั้งแบบเก่าและแบบใหม่บนวัตถุนั้น และส่วนต่อประสานทั้งใหม่และเก่า ต่างก็มีตัวระบุส่วนต่อประสาน (IID - Interface Identifier) ที่แตกต่างกันออกไป ซึ่งผู้รับบริการจะสอบถามไปยังวัตถุที่ติดต่อด้วยว่ามีความสามารถที่จะให้บริการตามตัวระบุส่วนต่อประสานที่ระบุหรือไม่ก่อนทุกครั้ง นั่นคือดีคอมไม่ต้องการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการ บริการรุ่นใหม่จะทดแทนบริการรุ่นเก่าไม่ได้ แม้ว่าจะมีความสามารถที่ใช้แทนได้ ดีคอมไม่มีคลังชนิดของบริการเช่นเดียวกับดีซีอี

2.1.2.4 อาร์เอ็ม-โอดีพี (RM-ODP – Reference Model for Open Distributed Processing)

อาร์เอ็ม-โอดีพี [9] เกิดจากความร่วมมือระหว่างองค์การระหว่างประเทศว่าด้วยการมาตรฐาน ไอเอสโอ (ISO - International Organization for Standardization) และไอทียู (ITU - International Telecommunications Union) อาร์เอ็ม-โอดีพีกำหนดคลังชนิดของบริการให้เป็นที่เก็บนิยามของส่วนต่อประสานและความสัมพันธ์ในรูปแบบต่างๆระหว่างส่วนต่อประสาน สำหรับแนวทางการสนับสนุนการเปลี่ยนรุ่นของชนิดของบริการในอาร์เอ็ม-โอดีพีมีแนวคิดของความเข้ากันได้เชิงพฤติกรรม (Behavioral Compatibility) ซึ่งเป็นการยอมให้บริการสามารถใช้แทนกันได้ในกรณีที่บริการมีความสัมพันธ์กันตามกฎซับไทป์ปิ้งของชนิดของบริการ (Service Type Subtyping Rules) [11] (รายละเอียดอยู่ในหัวข้อ 2.1.3) คือบริการที่มีส่วนต่อประสานเป็นซับไทป์ของส่วนต่อประสานของอีกบริการจะสามารถทำงานแทนได้ อย่างไรก็ดีหากบริการรุ่นใหม่ไม่เป็นซับไทป์ของบริการรุ่นเก่าก็จะใช้แทนกันไม่ได้ เช่นเดียวกับในกรณีของคอร์บา

2.1.3 **ความสัมพันธ์แบบซับไทป์ปิ้ง** (Subtyping Relationship) [11]

ในระบบกระจายเชิงวัตถุเราสามารถสร้างบริการให้มีความสัมพันธ์กับบริการอื่นในลักษณะของความสัมพันธ์แบบแม่ลูก โดยทำการสืบทอดคุณลักษณะ (Inheritance) ของส่วนต่อประสานของบริการหนึ่งจากส่วนต่อประสานของอีกบริการหนึ่ง ลักษณะของการเปลี่ยนแปลงจะเป็นแบบเพิ่มขึ้น (Incremental Change) คือเพิ่มเติมคุณสมบัติใหม่เข้าไป โดยการจะพิจารณาว่าบริการ A' มีความสัมพันธ์แบบซับไทป์ปิ้งกับบริการ A จะพิจารณาจากการที่บริการ A' ต้องมีหน้าที่การทำงานแบบที่บริการ A มี ซึ่งจากความสัมพันธ์แบบซับไทป์ปิ้งจะก่อให้เกิดลำดับชั้นของความสัมพันธ์แบบแม่ลูกของชนิดของบริการซึ่งก็คือ แผนภาพที่แสดงถึงความสัมพันธ์ในลักษณะแม่-ลูกระหว่างส่วนต่อประสานต่างๆ ที่มีการสืบทอดคุณลักษณะระหว่างกัน โดยที่ลำดับชั้นของความสัมพันธ์แบบแม่ลูกของชนิดของบริการสามารถนำไปใช้เป็นแนวทางหนึ่งในการพิจารณาให้บริการลูกสามารถทำงานแทนที่บริการแม่ได้ อาร์เอ็ม-โอดีพีกำหนดกฎซับไทป์ปิ้งของชนิดของบริการไว้ดังนี้ [11]

การพิจารณาว่าส่วนต่อประสาน X เป็นซับไทป์ของส่วนต่อประสาน Y หรือไม่ จะต้องครอบคลุมในประเด็นต่อไปนี้

- สำหรับแต่ละการดำเนินงานใน Y จะต้องมีการดำเนินงานใน X ที่สอดคล้องกันโดยต้องมีชื่อของการดำเนินงานเหมือนกัน
- แต่ละการดำเนินงานใน Y และ X ที่สอดคล้องกัน ต้องมีชื่อและจำนวนของพารามิเตอร์เหมือนกัน
- แต่ละการดำเนินงานใน Y และ X ที่สอดคล้องกัน ชนิด (Type) ของอาร์กิวเมนต์พารามิเตอร์ (Argument Parameter) หนึ่งๆใน X สามารถเป็นซูเปอร์ไทป์ (Supertype) ของชนิดอาร์กิวเมนต์พารามิเตอร์เดียวกันใน Y
- แต่ละการดำเนินงานใน Y และ X ที่สอดคล้องกัน ชนิดของพารามิเตอร์ที่คืนกลับ (Return Parameter) หนึ่งๆใน X สามารถเป็นซับไทป์ของชนิดของพารามิเตอร์ที่คืนกลับเดียวกันใน Y

2.1.4 **ความสัมพันธ์แบบเท่าเทียมกัน** (Equivalence Relationship) [3]

เป็นความสัมพันธ์อีกรูปแบบหนึ่งระหว่างบริการในระบบกระจาย โดยจะระบุว่าบริการหนึ่งมีความสามารถที่เท่าเทียมกับอีกบริการและสามารถทำงานแทนบริการนั้นได้หรือไม่ การทดแทนกันได้จะพิจารณาจากความสามารถในการทำงาน (Functionality) ของบริการมากกว่าหน้าตา (Signature) ของบริการตามนิยามของส่วนต่อประสาน ตัวอย่างเช่น บริการรุ่นใหม่ควรที่จะถูกใช้งานแทนที่บริการรุ่นเก่าได้ แม้ว่าหน้าตาของส่วนต่อประสานจะเปลี่ยนไปจากเดิม (เช่น ชื่อของการดำเนินงานเปลี่ยนไป) แต่ยังคงความสามารถโดยทั่วไปของบริการรุ่นเก่าไว้ รูปที่

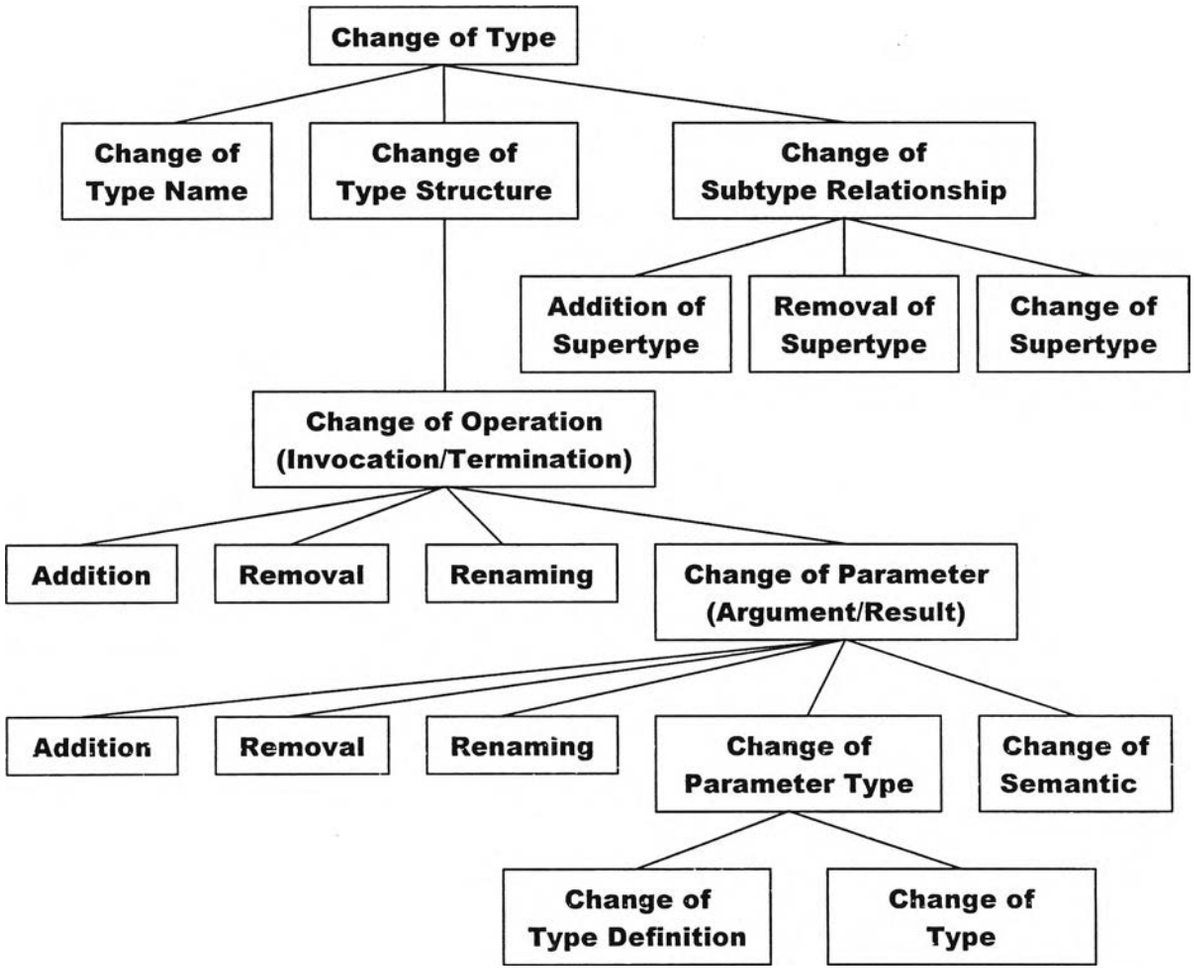
2.1 แสดงความเปลี่ยนแปลงแบบอิสระที่ยอมให้เกิดขึ้นได้ในส่วนต่อประสานของบริการรุ่นใหม่ [1] ในความสัมพันธ์รูปแบบนี้ จำเป็นต้องอาศัยข้อมูลพิเศษจำนวนหนึ่ง (Mapping Information) ที่ระบุว่าบริการรุ่นใหม่จะถูกทำให้ใช้งานแทนบริการรุ่นเก่าได้อย่างไร

2.1.5 **คลังชนิดของบริการ** (Type Repository) [12]

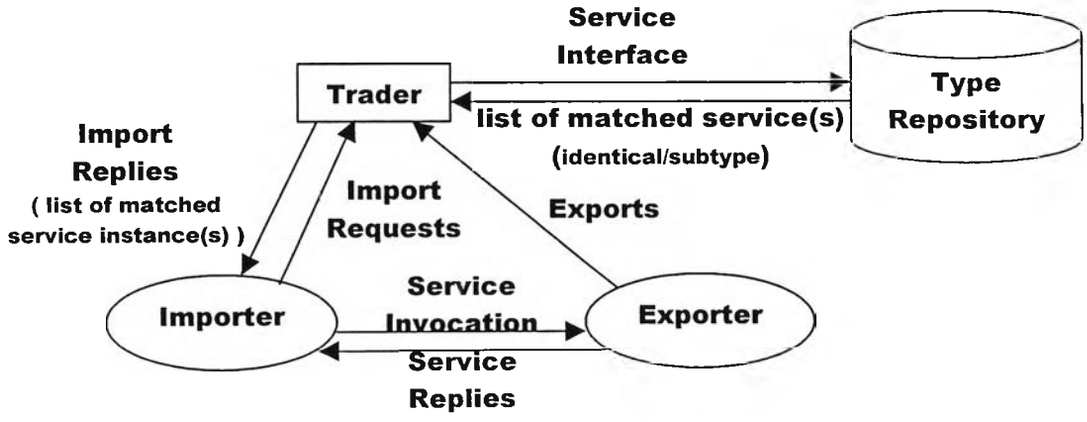
คือส่วนจัดเก็บข้อมูลต่างๆที่เกี่ยวข้องกับชนิดของบริการ (Service Type) ซึ่งอาจอยู่ในรูปของแฟ้มข้อมูล หรือฐานข้อมูลก็ได้ในทางปฏิบัติ โดยส่วนใหญ่ข้อมูลที่จัดเก็บในคลังชนิดของบริการของแต่ละสถาปัตยกรรมนั้นมีความคล้ายคลึงกัน เช่น ตัวระบุส่วนต่อประสานของบริการ (Service Interface Identifier) ชื่อของบริการ (Service Name) คำอธิบายชนิดของบริการ (Service Type Description) ส่วนต่อประสานของบริการ (Service Interface or Signature) นอกจากนี้อาจมีข้อมูลอื่นๆเพิ่มเติมตามความเหมาะสม ขึ้นอยู่กับว่าแต่ละสถาปัตยกรรมระบบกระจายจะออกแบบอย่างไร เช่น ข้อมูลรุ่น และข้อมูลของความสัมพันธ์แบบซับไทป์ปิ้ง ซึ่งข้อมูลเหล่านี้จะถูกใช้ สำหรับการค้นหาบริการในกรณีที่ต้องการคุณสมบัติการแทนที่กันได้ (Substitutability) อาร์เอ็ม-โอดีพีกำหนดหน้าที่หลักของคลังชนิดของบริการที่ทำงานร่วมกับเทรดเดอร์[4] (Trader) ดังรูปที่ 2.2

---

[4]เทรดเดอร์ (Trader) [11, 12] คือ องค์ประกอบของระบบกระจายที่เกี่ยวข้องกับการดูแล (ค้นหาและประกาศ) และจัดการกับอินสแตนซ์ (Instance) ของบริการที่มีในแต่ละโดเมน หรือแม้แต่ที่อยู่ต่างโดเมนกัน โดยมีหน้าที่การทำงานเบื้องต้นได้แก่ อิมพอร์ต (Import หรือ Discover) สำหรับผู้รับบริการที่ต้องการค้นหาบริการ และ เอ็กพอร์ต (Export หรือ Advertise) สำหรับผู้ให้บริการที่จะโฆษณาอินสแตนซ์ของบริการของตน



รูปที่ 2.1 การเปลี่ยนแปลงแบบอิสระที่มีได้ในนิยามของส่วนต่อประสานในอาร์เอ็ม-โอดีพี



รูปที่ 2.2 ความสัมพันธ์ระหว่างการทำงานของเทรดเดอร์และคลังชนิดของบริการ

เมื่อเทรดเดอร์รับคำขอใช้บริการจากอิมพอร์ตเตอร์ (Importer) ซึ่งก็คือผู้รับบริการก็จะไปตรวจสอบกับคลังชนิดของบริการว่ามีบริการดังกล่าวหรือไม่ โดยคลังชนิดของบริการจะส่งรายการของบริการ ซึ่งมีส่วนต่อประสานที่จะทำงานได้ตามที่ผู้รับบริการต้องการกลับมาให้ อันได้แก่บริการที่ตรงกับที่ขอมาหรือบริการที่เป็นซับไทป์ จากนั้นเทรดเดอร์ก็จะคัดเลือกอินสแตนซ์ของบริการเหล่านั้นซึ่งมีเอ็กพอร์ตเตอร์ (Exporter) ซึ่งก็คือผู้ให้บริการมาโฆษณาไว้ แล้วส่งกลับบริการที่ได้รับการคัดเลือกให้ผู้รับบริการไป เนื่องจากคลังชนิดของบริการดูแลเฉพาะความสัมพันธ์แบบซับไทป์-ปิ้งนี้เองที่ทำให้เกิดข้อจำกัดในการค้นหาบริการที่เหมาะสมมาใช้แทนที่กัน เนื่องจากความสัมพันธ์แบบซับไทป์ปิ้งไม่สอดคล้องกับการเปลี่ยนบริการรุ่นใหม่แบบอิสระ

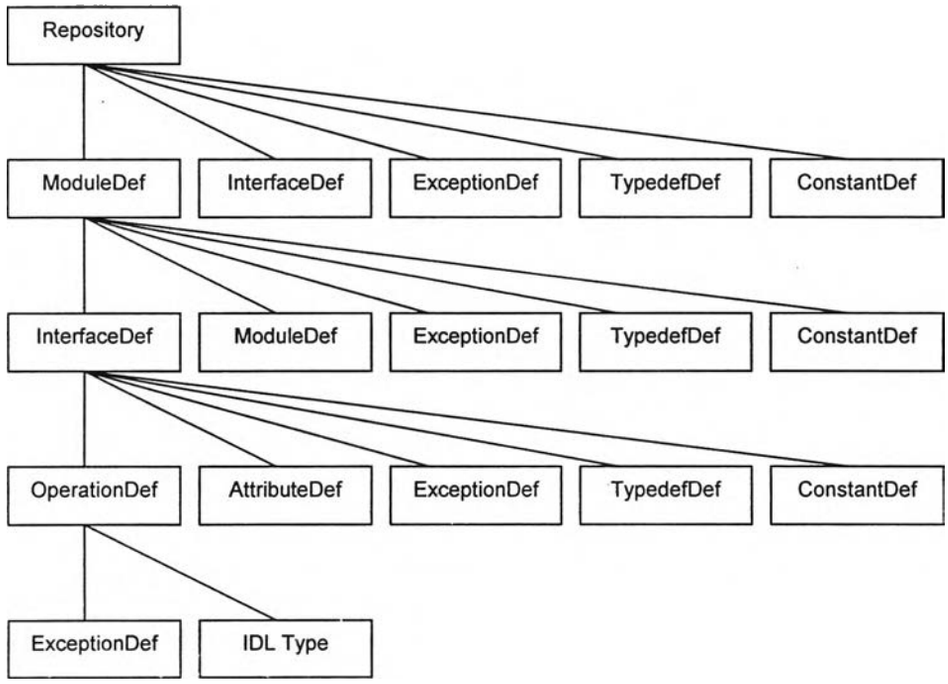
2.1.6 **คลังส่วนต่อประสาน** (Interface Repository) [5-7]

นิยามส่วนต่อประสาน (Interface Definition) เป็นข้อมูลส่วนหนึ่งของคำอธิบายชนิดของบริการ โดยทั่วไปจะเก็บอยู่ในเทรดเดอร์ แต่สำหรับระบบคอร์บา ได้แยกการจัดเก็บส่วนของนิยามส่วนต่อประสานออกมาเก็บที่อีกองค์ประกอบหนึ่งที่เรียกว่าคลังส่วนต่อประสาน (Interface Repository) แต่การทำงานของเทรดเดอร์ในคอร์บา ยังคงเรียกใช้ข้อมูลนิยามส่วนต่อประสานที่เก็บอยู่ในคลังส่วนต่อประสาน รูปแบบการจัดเก็บข้อมูลในคลังส่วนต่อประสานของคอร์บามีความสัมพันธ์ในลักษณะการบรรจุ (Containment Relationship) ดังรูปที่ 2.3

ข้อมูลชนิดหลักที่จัดเก็บอยู่ในคลังส่วนต่อประสาน มีทั้งสิ้น 8 ชนิด ดังนี้

1. Repository เป็นส่วนจำเพาะ (module) ในระดับบนสุดเพื่อใช้เป็นตัวบรรจุ (Container) ข้อมูลค่าคงที่ (constants) นิยามชนิด (typedefs) ข้อยกเว้น (exceptions) นิยามส่วนต่อประสาน และส่วนจำเพาะ
2. ModuleDef เป็นตัวบรรจุในเชิงตรรกะของส่วนต่อประสาน นอกจากนั้นยังเป็นตัวบรรจุของข้อมูลค่าคงที่ นิยามชนิด ข้อยกเว้น นิยามส่วนต่อประสาน และส่วนจำเพาะอื่นๆ
3. InterfaceDef คือนิยามส่วนต่อประสาน เป็นตัวบรรจุรายการข้อมูลค่าคงที่ นิยามชนิด (Type Definition) ข้อยกเว้น (Exception) การดำเนินการ และลักษณะประจำ
4. AttributeDef คือนิยามลักษณะประจำของส่วนต่อประสาน
5. OperationDef คือนิยามการดำเนินการของส่วนต่อประสาน เป็นตัวบรรจุรายการพารามิเตอร์และข้อยกเว้นที่อาจเกิดขึ้นจากการดำเนินการ
6. TypedefDef เป็นส่วนต่อประสานพื้นฐานสำหรับนิยามของชื่อชนิด (named type) ที่ไม่ใช่ส่วนต่อประสาน
7. ConstantDef คือนิยามลักษณะประจำของชื่อค่าคงที่ (named constant)

8. ExceptionDef คือนิยามข้อยกเว้นที่อาจเกิดขึ้นจากการดำเนินการ



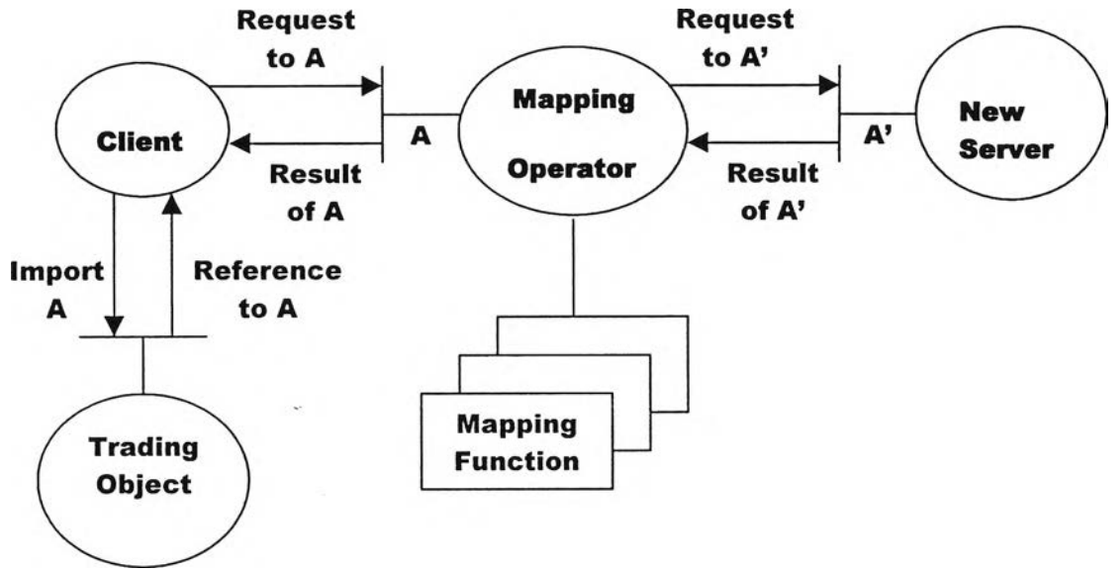
รูปที่ 2.3 ความสัมพันธ์ในลักษณะการบรรจุ (Containment Relationship) ของข้อมูลในคลังส่วนต่อประสาน

จากข้อกำหนดเกี่ยวกับนิยามส่วนต่อประสานของคอร์บา [5] ข้อมูลที่ได้กำหนดไว้แล้วข้างต้นในคลังส่วนต่อประสาน เป็นเพียงข้อมูลพื้นฐาน หากต้องการเพิ่มความสามารถของคลังส่วนต่อประสานให้สามารถรองรับข้อมูลส่วนต่อประสานที่มีลักษณะเฉพาะได้ (proprietary) การออกแบบและการทำให้เกิดผล ควรเป็นไปในรูปแบบของการขยายหรือสืบทอดคุณลักษณะของคลังส่วนต่อประสาน โดยไม่ควรเปลี่ยนแปลงนิยามส่วนต่อประสาน ซึ่งวิทยานิพนธ์นี้ได้ปฏิบัติตามแนวทางดังกล่าว

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 Evolution Transparency for Distributed Service Types [1]

งานวิจัยนี้ได้ตระหนักถึงปัญหาที่จะเกิดขึ้นหากมีการเปลี่ยนรุ่นของบริการแล้วขาดการควบคุมที่ดีในระบบกระจาย ซึ่งงานวิจัยนี้ได้นำเสนอการทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการโดยใช้วิธีการยกระดับคำขอใช้บริการ (Request Upgrade) ซึ่งก็คือการทำการแปลง จากบริการรุ่นเก่าสู่บริการรุ่นใหม่ ดังรูปที่ 2.4



รูปที่ 2.4 รูปแบบการทำงานของฟังก์ชันการแปลง

การทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการของงานวิจัยนี้ ได้มีการสร้างตัวดำเนินการแปลง (Mapping Operator) ขึ้นมาโดยอัตโนมัติเมื่อมีการเปลี่ยนรุ่นของบริการ ซึ่งข้อมูลในการเปลี่ยนรุ่น (ดูหัวข้อ 3.1) จะได้มาจากผู้ที่ทำการเปลี่ยนรุ่น (Evolver) เมื่อมีการขอใช้บริการเก่า (คือ A ในรูปที่ 2.4) คำขอใช้บริการนั้นจะถูกแปลงโดยตัวดำเนินการแปลง ซึ่งจะใช้ฟังก์ชันการแปลงในการแปลงคำขอใช้บริการนั้นให้อยู่ในรูปแบบที่บริการใหม่ (คือ A' ในรูปที่ 2.4) เข้าใจได้ ทั้งนี้ตัวดำเนินการแปลงจะถูกสร้างให้เสมือนกับเป็นบริการรุ่นเก่าเพื่อที่จะสามารถดัก (Intercept) คำขอใช้บริการรุ่นเก่าได้ ทั้งนี้แต่ละคู่ของอินสแตนซ์ของบริการรุ่นเก่ากับอินสแตนซ์ของบริการรุ่นใหม่ที่ถูกจับคู่ในการทำงานทดแทนกันอาจใช้ฟังก์ชันการแปลงที่ต่างกันได้ ขั้นตอนการทำงานดังกล่าวนี้จะช่วยผู้รับบริการให้สามารถเรียกใช้บริการผ่านส่วนต่อประสานในลักษณะเดิมได้ โดยที่ผู้รับบริการไม่จำเป็นต้องรู้ว่าบริการได้เปลี่ยนรุ่นไปแล้ว เป็นการแก้ปัญหาไปก่อนในช่วงเวลาหนึ่งในขณะที่ผู้รับบริการยังไม่พร้อมที่จะเปลี่ยนการทำงานของตนไปตามบริการรุ่นใหม่ เพราะการเปลี่ยนรุ่นของบริการในระบบกระจาย ไม่สามารถเปลี่ยนให้ครอบคลุมและทั่วถึงได้ในช่วงเวลาอันสั้น ทำให้การทำงานของผู้รับบริการไม่ต้องหยุดชะงักลงในทันที กลไกดังกล่าวนี้จำเป็นต้องอาศัยฟังก์ชันการแปลงในการแปลงคำขอใช้บริการ ซึ่งข้อมูลส่วนนี้ผู้ทำการเปลี่ยนรุ่นของบริการจะเป็นผู้ระบุ ทั้งนี้เนื่องจากว่าบริการรุ่นใหม่สามารถถูกเปลี่ยนแปลงได้อย่างอิสระจากบริการรุ่นเก่า ทำให้ความสัมพันธ์แบบเท่าเทียมกันนั้นไม่สามารถพิจารณาได้จากหน้าตาของส่วนต่อประสานของบริการแต่เพียงอย่างเดียว ผู้ทำการเปลี่ยนรุ่นจะต้องคำนึงถึงการแทนที่กันได้ทางความสามารถ (Functionality Substitutability) แล้วระบุฟังก์ชันการแปลงให้เหมาะสม

ในประเด็นของการจัดการกับข้อมูลฟังก์ชันการแปลง รูปแบบของฟังก์ชันการแปลงก็เป็นดังลักษณะพื้นฐานของฟังก์ชันทั่วๆไปในภาษาการโปรแกรมต่างๆ เช่น ลักษณะของค่าปริยาย (Default Values) นิพจน์ (Expression) การเรียกใช้ฟังก์ชันหรือวัตถุอื่น (call to other function or object) เป็นต้น สำหรับฟังก์ชันการแปลงในงานวิจัย [1] มีไว้เพื่อการแปลงในรูปแบบต่างๆดังนี้

- แปลงการดำเนินการ (Map Operation) คือ ฟังก์ชันการแปลงจะระบุว่าการดำเนินงานหนึ่งๆ ของส่วนต่อประสานเดิม สามารถถูกทำงานแทนด้วยการดำเนินงานหนึ่งหรือกลุ่มของการดำเนินงานใดบ้างในส่วนต่อประสานใหม่
- แปลงพารามิเตอร์ (Map Parameter) คือ ฟังก์ชันการแปลงจะระบุว่าการเปลี่ยนแปลงลักษณะของคำขอใช้บริการ เช่น เปลี่ยนชื่อ เปลี่ยนชนิด (Type) รวมไปถึงการเปลี่ยนจำนวนของพารามิเตอร์ จากคำขอใช้บริการของส่วนต่อประสานเดิมไปสู่คำขอใช้บริการในส่วนต่อประสานใหม่จะทำได้อย่างไร
- แปลงอินสแตนซ์ (Map Instance) คือ ฟังก์ชันการแปลงจะระบุว่าตัวดำเนินการแปลงจะตัดสินใจอย่างไรในการเลือกอินสแตนซ์ของบริการรุ่นใหม่มารองรับการทำงานแบบเก่า

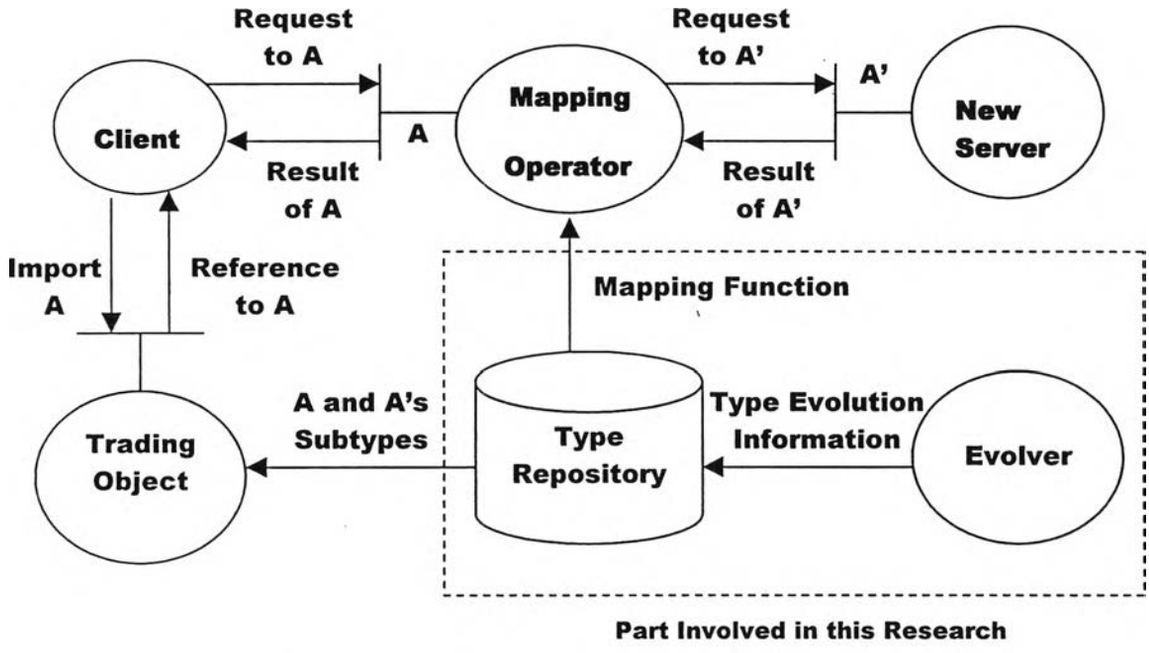
รูปแบบของฟังก์ชันการแปลงเหล่านี้จะสนับสนุนการเปลี่ยนแปลงแบบอิสระ

อย่างไรก็ดีในงานวิจัย [1] ยังไม่ได้คำนึงถึงการจัดเก็บข้อมูลนี้ให้เป็นระบบระเบียบ ข้อมูลฟังก์ชันการแปลงดังกล่าวยังเก็บอยู่ในแฟ้มข้อมูลต่างๆที่กระจัดกระจายอยู่ในระบบแฟ้มข้อมูลทั้งๆที่ควรเก็บรวบรวมเป็นส่วนหนึ่งของข้อมูลที่เกี่ยวกับบริการซึ่งจัดเก็บอยู่แล้วโดยคลังชนิดของบริการ

2.3 แนวคิดในการทำวิจัย

จากการที่แต่ละสถาปัตยกรรมระบบกระจายยังไม่มีมาตรฐานในการรองรับปัญหา การเปลี่ยนรุ่นของบริการและการทำงานแทนที่กันได้ข้ามรุ่นอย่างเพียงพอ เพื่อให้งานวิจัย [1] ซึ่งพยายามแก้ปัญหาเหล่านี้โดยทำให้เกิดความโปร่งใสในการเปลี่ยนแปลงชนิดของบริการสำหรับการเปลี่ยนบริการรุ่นใหม่แบบอิสระนั้น ทำงานได้สมบูรณ์ยิ่งขึ้น ควรที่จะต้องมีระบบจัดเก็บและจัดการข้อมูลที่ใช้สำหรับการทำการแปลงที่ดีขึ้น ดังนั้นงานวิจัยนี้จึงมีแนวคิดที่จะออกแบบคลังชนิดของบริการให้มีความสามารถเพิ่มเติมในการเก็บข้อมูลความสัมพันธ์แบบเท่าเทียมกัน และข้อมูลฟังก์ชันการแปลงที่จำเป็นสำหรับงานวิจัย [1] นอกเหนือไปจากการจัดเก็บข้อมูลเกี่ยวกับนิยามของส่วนต่อประสานโดยทั่วไป ดังรูปที่ 2.5

จากแนวคิดดังกล่าวส่งผลให้การออกแบบและพัฒนาคลังชนิดของบริการเพื่อให้สามารถรองรับความสัมพันธ์แบบเท่าเทียมกันได้นั้น สามารถแบ่งแนวคิดออกเป็น 2 ประเด็นใหญ่ๆคือ ข้อมูลที่ต้องจัดเก็บ (ดูหัวข้อ 2.3.1) และการจัดการกับข้อมูลเหล่านั้น (ดูหัวข้อ 2.3.2)



รูปที่ 2.5 แบบจำลองการทำงานของฟังก์ชันการแปลงร่วมกับคลังชนิดของบริการ

2.3.1 ข้อมูลสนับสนุนความสามารถในการรองรับความสัมพันธ์แบบเท่าเทียมกัน

ข้อมูลเพิ่มเติมที่ทำให้คลังชนิดของบริการเดิมสามารถรองรับความสัมพันธ์แบบเท่าเทียมกันได้นั้น เป็นข้อมูลที่งานวิจัย [1] นำไปสร้างตัวดำเนินการแปลง ข้อมูลดังกล่าวได้แก่ ข้อมูลการเปลี่ยนรุ่นของส่วนต่อประสาน ข้อมูลของอินสแตนซ์ของบริการต่างรุ่นที่มีความสัมพันธ์แบบเท่าเทียมกัน รวมทั้งข้อมูลฟังก์ชันการแปลงที่อินสแตนซ์ของบริการต่างรุ่นใช้ ข้อมูลเหล่านี้จะได้รับการจัดเก็บไว้ในคลังชนิดของบริการที่จะพัฒนาขึ้นและเนื่องจากวิทยานิพนธ์นี้ได้คำนึงถึงการนำข้อมูลความสัมพันธ์แบบเท่าเทียมกันเหล่านี้กลับมาใช้ได้ใหม่ได้ทั้งในการเปลี่ยนรุ่นระดับอินสแตนซ์และระดับชนิด จึงได้ออกแบบให้การจัดเก็บข้อมูลมีโครงสร้างความสัมพันธ์ในลักษณะการบรรจุถึง 3 ระดับ อันได้แก่ข้อมูลชนิด EquivalenceDef, MappingDef และ MappingFunctionDef ซึ่งในข้อมูลแต่ละตัวก็จะมีการจัดเก็บข้อมูลพื้นฐานและข้อมูลเฉพาะในแต่ละชนิดข้อมูล เช่นข้อมูลเฉพาะของ MappingDef ได้แก่ ข้อมูลรายละเอียดของคู่อินสแตนซ์ของส่วนต่อประสานที่มีความสัมพันธ์แบบเท่าเทียมกัน (EquivalenceSourceObjectPair และ EquivalenceDestinationObjectPair) เป็นต้น ส่วนข้อมูลเฉพาะของ MappingFunctionDef ได้แก่ข้อมูลวิธีการแปลง (MapFnBody) เป็นต้น ซึ่งจะกล่าวถึงรายละเอียดในบทที่ 3

2.3.2 แนวทางการจัดการข้อมูลสนับสนุนความสามารถในการรองรับความสัมพันธ์แบบเท่าเทียมกัน

คลังชนิดของบริการที่จะออกแบบจะสามารถนำไปพัฒนาและปรับใช้กับคลังชนิดของบริการที่มีอยู่ในสถาปัตยกรรมต่างๆได้ตามความเหมาะสม โดยรวมแล้วคลังชนิดของบริการใหม่นี้จะมีความสามารถในการจัดการกับข้อมูลที่กล่าวในหัวข้อ 2.3.1 ดังเช่น

- จัดเก็บนิยามของส่วนต่อประสานของบริการต่างๆ เช่น ตัวระบุส่วนต่อประสานของบริการ (Interface Identifier) ส่วนต่อประสานของบริการ และ ความสัมพันธ์แบบซับไทป์ปิ้งเป็นอย่างน้อย
- จัดเก็บข้อมูลความสัมพันธ์แบบเท่าเทียมกัน ซึ่งรวมถึงฟังก์ชันการแปลงระหว่างอินสแตนซ์ของบริการรุ่นเก่ากับอินสแตนซ์ของบริการรุ่นใหม่แต่ละคู่
- แสดงรายละเอียดของนิยามของส่วนต่อประสานของบริการต่างๆ และแสดงความสัมพันธ์แบบซับไทป์ปิ้ง
- แสดงรายละเอียดของความสัมพันธ์แบบเท่าเทียมกัน ซึ่งรวมถึงฟังก์ชันการแปลง
- ลบข้อมูลนิยามของส่วนต่อประสาน
- ลบข้อมูลความสัมพันธ์แบบเท่าเทียมกัน ซึ่งรวมถึงฟังก์ชันการแปลง (สามารถใช้ในกรณีที่ผู้รับบริการได้ทำการแก้ไขโปรแกรมของตนให้สอดคล้องกับบริการรุ่นใหม่แล้ว)

ต้นแบบของคลังชนิดของบริการที่รองรับความสัมพันธ์แบบเท่าเทียมกันนี้ จะทำงานร่วมกับส่วนต่อประสานสำหรับผู้ใช้ (User Interface) ที่พัฒนาขึ้นมาด้วย สำหรับให้ผู้เปลี่ยนรุ่นสามารถบันทึกข้อมูลที่ใช้ในการเปลี่ยนรุ่นลงไปได้ รวมทั้งสามารถแสดงข้อมูลต่างๆที่มีการจัดเก็บอยู่ในคลังด้วย

สำหรับการวัดผล หลังจากได้พัฒนาคลังชนิดของบริการขึ้นมาตามที่ออกแบบแล้ว จะมีการทดสอบว่าคลังชนิดของบริการนั้นสามารถจัดเก็บและจัดการข้อมูลความสัมพันธ์แบบเท่าเทียมกัน ทั้งในระดับชนิดและระดับอินสแตนซ์ ซึ่งสอดคล้องกับการเปลี่ยนแปลงส่วนต่อประสานดังรูปที่ 2.1 ด้วย รวมทั้งตัวอย่างของการนำข้อมูลที่จัดเก็บไปใช้ในบางกรณีของรูปที่ 2.1

ในบทถัดไป จะอธิบายถึงนิยามของคลังชนิดของบริการที่รองรับความสัมพันธ์แบบเท่าเทียมกัน ที่ได้ออกแบบตามแนวคิดการวิจัยดังที่กล่าวไปแล้วในบทนี้ และแนวทางปฏิบัติที่ได้กล่าวไว้ในนิยามส่วนต่อประสานของคอร์บา