บทที่ 6

ADO.NET

6.1 สถาปัตยกรรมของ ADO.NET

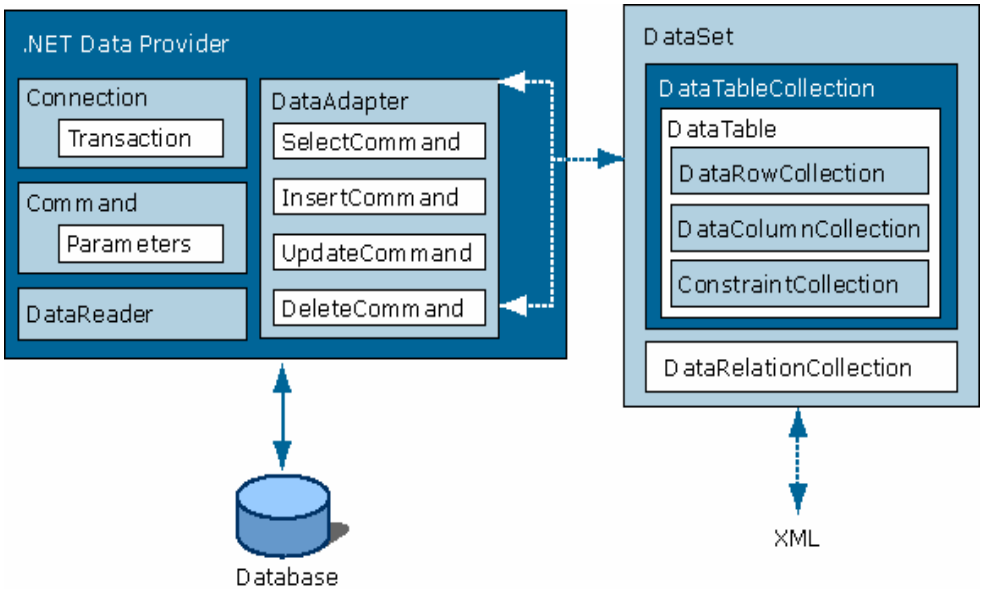
เนื่องจากในการพัฒนาโปรแกรมด้านฐานข้อมูลแต่เดิมนั้นได้ใช้เทคโนโลยี ADO (ActiveX Data Object) เป็นเครื่องมือในการติดต่อและจัดการข้อมูลในแบบ connected database คือจะเปิดการเชื่อมต่อกับฐานข้อมูลไว้ตลอดช่วงเวลาใช้งานซึ่งจะทำให้การเพิ่ม ลบ แก้ไขหรือการกระทำใดๆก็ตามที่เกี่ยวข้องกับข้อมูลจะเป็นการกระทำกับข้อมูลจริงในไฟล์ฐานข้อมูลเป็นจำนวนมากและพร้อมๆกันหรือการประมวลผลข้อมูลที่มีปริมาณมากจนทำให้ระยะเวลาที่ต้องเชื่อมต่อกับ server นานเกินไป ซึ่งจะทำให้อัตราการสิ้นเปลืองทรัพยากรบนเครื่อง server สูง ส่งผลให้ประสิทธิภาพโดยรวมของระบบลดลงได้ ดังนั้นใน Visual Studio .NET จึงได้นำเทคโนโลยีตัวใหม่ล่าสุดคือ ADO.NET ที่พัฒนาจาก ADO มาใช้จัดการข้อมูลแบบ disconnected database คือสามารถจัดการข้อมูลในฐานข้อมูลโดยไม่ต้องเปิดการเชื่อมต่อไว้ตลอดเวลาเหมือนใน ADO เนื่องจากข้อมูลจะถูกอ่านจากฐานข้อมูลมาเก็บไว้ในหน่วยความจำก่อน จากนั้นจะตัดการเชื่อมต่อกับฐานข้อมูลทำให้การกระทำใดๆหลังจากนี้จะเป็นการกระทำกับข้อมูลในหน่วยความจำเท่านั้น เมื่อเสร็จสิ้นกระบวนการแล้วจึงจะบันทึกกลับลงฐานข้อมูล ซึ่งจะช่วยประหยัดทรัพยากรของระบบและเพิ่มประสิทธิภาพการทำงานของเซิร์ฟเวอร์ (ที่จริงใน ADO เองก็มีความสามารถในการทำ disconnected database ได้ แต่โปรแกรมเมอร์จะต้องเป็นผู้ควบคุมด้วยตนเองว่าจะให้มีการเชื่อมต่อหรือยกเลิกการเชื่อมต่อเมื่อใด ซึ่งต่างกับใน ADO.NET ที่จะตัดการเชื่อมต่อให้โดยอัตโนมัติ เมื่อไม่มีความจำเป็นต้องใช้ฐานข้อมูลในแบบออนไลน์ และจะเปิดการเชื่อมต่อให้ไม่เมื่อจำเป็นต้องทำงานร่วมกับฐานข้อมูลจริง)

ในอีกประเด็นหนึ่งคือ ปัจจุบันการส่งข้อมูลเป็นลักษณะผ่านเว็บมากขึ้น ภาษา XML มีบทบาทมากขึ้น โดยส่วนของ ADO.NET ก็จะมีความสามารถในการรับส่งข้อมูลเป็น XML อีกด้วย

องค์ประกอบของคอมโพเนนต์ใน ADO.NET

ADO.NET ประกอบด้วย 2 คอมโพเนนต์หลักคือ ดาต้าเซ็ต (DataSet) และ ดอทเน็ต ดาต้า โพไวด์เดอร์ (.NET data provider) ซึ่งประกอบด้วย กลุ่มของ คอมโพเนนต์คือ คอนเน็กชั่น (Connection), คอมมาน (Command), ดาต้ารีดเดอร์ (DataReader) และ ดาต้าอเด็บเตอร์ (DataAdapter) ดังรูปที่ 6-1

ADO.NET DataSet เป็นคอมโพเนนต์หลักในการเชื่อมต่อข้อมูลแบบดีสคอนเน็กเตต (Disconnected), ดาต้าเซ็ต (DataSet) ถูกออกแบบมาให้มีความเป็นอิสระจากแหล่งข้อมูลด้วยเหตุนี้ มันจึงสามารถใช้ได้ในดาต้าซอร์ส (Data Source) หลายประเภท



รูปที่ 6-1 สถาปัตยกรรม ADO.NET

ดาต้าเซ็ต ประกอบด้วยกลุ่มของ ดาต้าเทเบิล (DataTable) ซึ่ง เทเบิล (Table) ที่ว่านี้ก็เกิดมาจากการส่ง Query ไปดึงมาจาก ฐานข้อมูล โดยดาต้าเทเบิลก็จะประกอบไปด้วย DataRow, DataColumn รวมถึง primary key, foreign key, ข้อจำกัดต่างๆ พวกข้อมูล (Data) Integrity และ ความสัมพันธ์ของข้อมูล (DataRelation) ใน ดาต้าเทเบิลเอง ในส่วนของการส่งข้อมูล ดาต้าเซ็ต ได้ใช้ XML เป็นตัวกลางในการส่งข้อมูลระหว่าง Tier ทำให้สามารถนำไปใช้กับการทำเว็บเซอร์วิส ได้

ส่วนหลักอีกส่วนหนึ่งคือ .NET Data Provider ซึ่งสร้างมาเพื่อการประมวลผลข้อมูลแบบ รวดเร็วและแบบส่งไปข้างหน้าอย่างเดียว (forward-only) และเป็นการเข้าถึงข้อมูลแบบอ่านอย่างเดียว (read-only) โดยประกอบด้วยส่วนย่อยๆ ดังต่อไปนี้คือ

1. Connection ทำหน้าที่ติดต่อกับฐานข้อมูล SQL Server หรือ OLE DB เช่นเดียวกับออบเจ็ค Connection ใน ADO

2. Command จะใช้ร่วมกับออบเจ็คพารามิเตอร์ (Parameter) ในการประมวลผลคำสั่ง SQL, Query หรือ Stored Procedure ในแบบที่สามารถส่งผลลัพธ์คืนกลับมาหรือจะไม่ส่งคืนก็ได้เช่นเดียวกับการใช้ออบเจ็ค Command ใน ADO

3. DataReader จะใช้สำหรับการดึงข้อมูลแบบทีละเรคอร์ดจากฐานข้อมูลบนเซิร์ฟเวอร์ โดย server cursor จะเป็นแบบ forward-only คือเดินหน้าอย่างเดียวถอยหลังไม่ได้และไม่สามารถแก้ไขข้อมูลได้ด้วย (read-only) ทำให้ความเร็วในการเข้าถึงข้อมูลสูง

4. DataAdapter จะเป็นเหมือนสะพานเชื่อมต่อระหว่างดาต้าเซ็ต และ ดาต้าซอร์ส โดยการทำงานคือเมื่อมีการเปิด Connection และกำหนด Query String SQL แล้ว DataAdapter ใช้ Command Object เพื่อประมวลผลคำสั่งภาษา SQL และทำการดึงข้อมูลลงมาที่ดาต้าเซ็ต เพื่อใช้งานต่อไป โดย .NET Framework มี .NET data provider ให้ใช้ 2 แบบคือ

1. SQL Server .NET data provider ใช้ติดต่อและทำงานร่วมกับไฟล์ฐานข้อมูล Microsoft SQL Server ประกอบด้วยคลาส SqlConnection, SqlCommand, SqlDataReader และ SqlDataAdapter

2. OLE DB .NET data provider ใช้ติดต่อและทำงานร่วมกับไฟล์ฐานข้อมูลเช่น Microsoft Access, Oracle เป็นต้น ประกอบด้วยคลาส OleDbConnection, OleDbCommand, OleDbDataReader และ OleDataAdapter

รูปที่ 6-2 เปรียบเทียบระหว่าง SQL Server .NET Data Provider และ OLEDB .NET Data Provider

ก็คือ ถ้าใช้ SQL Server .NET Data Provider ต้องใช้กับ Microsoft SQL Server 7.0 หรือใหม่กว่าเท่านั้น โดย คลาสของ SQL Server .NET Data Provider กำหนดโดย namespace System.Data.SqlClient สำหรับเวอร์ชันที่เก่ากว่าก็ใช้ OLE DB .NET Data Provider แทน

ในการเลือกใช้ DataReader หรือ ดาต้าเซ็ตใน แอปพลิเคชั่นชัน นั้น ควรพิจารณาถึงชนิดของหน้าที่การทำงานของแอปพลิเคชั่น โดยจะเลือกใช้ดาต้าเซ็ต ก็ต่อเมื่อ

- มีการส่งข้อมูลระหว่างเทียร์ หรือมีต้องมีการใช้ XML Web service
- มีการติดต่อข้อมูลแบบไดนามิค โดยผูกติดกับวินโดว์ฟอร์ม (Window From)หรือ การรวมข้อมูลที่เกี่ยวข้องกันมาใช้งานจากแหล่งข้อมูลหลายๆที่
- ให้มีการเก็บข้อมูลลงในแคชของเครื่อง
- มีการประมวลผลข้อมูลโดยไม่ต้องเปิดการเชื่อมต่ออยู่ตลอดเวลา ซึ่งทำให้ไคลเอนต์อื่นๆ สามารถเข้ามาใช้งานฐานข้อมูลได้มากขึ้น

โดยถ้าไม่มีความจำเป็นเหล่านี้ ก็สามารถใช้ DataReader แทนได้ โดยการทำงานจะเป็นแบบ Forward-Only and read-only คือเป็นการส่งข้อมูลให้อย่างเดียว และส่งไปข้างหน้าเท่านั้น และเป็นแบบอ่านได้อย่างเดียว คือ Update ข้อมูลกลับไปไม่ได้ โดยการใช้ DataReader จะเป็นการประหยัดหน่วยความจำลงไปได้มาก

สรุปข้อเปรียบเทียบระหว่าง ADO และ ADO.NET

1. ในการใช้งานกับ Web ซึ่งเป็นลักษณะ ดึงข้อมูลเว็บเพจนั้นเสร็จแล้วก็ทำการตัดสายอัตโนมัติ ซึ่งส่งผลทำให้การทำงานเร็วขึ้น ใช่ว่า ADO จะไม่มี แต่ ADO.NET ทำให้มันสะดวกขึ้นโดยทำโดยอัตโนมัติแทนที่โปรแกรมเมอร์ต้องไปควบคุมทุกขั้นตอน

   <u>หมายเหตุ</u> (ADO.NET จะทำ snapshot ของฐานข้อมูลที่เรากำลังติดต่ออยู่เก็บไว้ในหน่วยความจำของเครื่อง โดยการทำ Snapshot นั้นไม่ได้ทำทั้งฐานข้อมูล แต่ทำเฉพาะส่วนที่เรา query เท่านั้น โดยเมื่อดึงข้อมูลเสร็จก็จะทำการ Disconnect ทันที)
2. ส่วนการติดต่อกับฐานข้อมูลและส่วน การจัดการ client cursor (ภาษา ADO เรียกว่า Recordset) นั้นจะเป็นอิสระต่อกันโดยสิ้นเชิง ทำให้ทั้งสองส่วนพัฒนาได้โดยอิสระ
3. ใน ADO เมื่อก่อน Recordset ไม่สามารถพัฒนาได้มากนักเพราะว่าต้องผูกติดกับ Database Server ที่ตัวเองติดต่อด้วย ดังนั้นเพื่อให้สามารถรองรับได้กับ Database ทุกประเภท ทาง Recordset จึงจำเป็นต้องถูกออกแบบมาให้ทำงานเป็นแบบกลางๆ ทำให้ความสามารถด้อยลงไป ดังนั้น ใน ADO.NET จึงแยกในส่วนการติดต่อฐานข้อมูลออกมา โดยจะมีหนึ่งคลาสต่อหนึ่งต่อหนึ่งประเภทฐานข้อมูล เช่น ถ้าเราต่อกับ Oracle ต้องใช้คลาส หนึ่ง และถ้าต่อกับ Access ต้องใช้อีกคลาสไม่เหมือนกับ ADO ที่ใช้ OLEDB , คลาส ADODB.Connection เพื่อติดต่อกับทุก Database ข้อดีข้อการแยกก็จะทำให้การเชื่อมต่อกับ DBMS แต่ละยี่ห้อเป็นไปได้โดยอิสระและสามารถดึงเอาประสิทธิภาพมาใช้ได้อย่างเต็มที่
4. ส่วนของ Client Cursor นั้นปรับปรุงใหม่ แทนที่จะเป็น Recordset ซึ่งรองรับเพียงแค่ตารางเดียวเหมือนเมื่อก่อน แต่ใน ADO.NET สามารถรองรับทั้งฐานข้อมูล และแต่ละเทเบิล สามารถมี reference integrity ด้วย
5. การส่งผ่านข้อมูลระหว่าง ADO.NET และ DBMS นั้น แทนที่จะส่งโดยใช้ DCOM ซึ่งจะมีปัญหากับ Firewall ทำให้การสื่อสารยุ่งยาก แต่ ADO.NET นั้นจะแปลงข้อมูลอยู่ในรูปของ XML ก่อนแล้วค่อยส่งออกไป ทำให้ใช้พอร์ต (port) ของเว็บมาตรฐาน ทะลุกำแพง firewall ได้