ภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ

นายอิทธิพล นันทรุจิ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2558

Domain Specific Language for Rendering Visual Components

Mr. Ittipol Nuntaruji

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Software Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2015

หัวข้อวิทยานิพนธ์	ภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ
โดย	นายอิทธิพล นันทรุจิ
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร. ญาใจ ลิ่มปิยะกรณ์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร. สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(ศาสตราจารย์ ดร. บุญเสริม กิจศิริกุล)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(รองศาสตราจารย์ ดร. ญาใจ ลิ่มปิยะกรณ์)

.....กรรมการภายนอกมหาวิทยาลัย
(อาจารย์ ดร. ภาสกร อภิรักษ์วรพินิต)

อิทธิพล นันทรุจิ : ภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ (Domain Specific Language for Rendering Visual Components) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร. ญาใจ ลิ่มปิยะกรณ์, 61 หน้า.

ปัญญาธุรกิจ หรือ บีไอ ประกอบด้วยการใช้เครื่องมือที่หลากหลาย แอปพลิเคชัน และระเบียบวิธีการต่าง ๆ เพื่อช่วยให้องค์กรสามารถรวบรวม และจัดเตรียมข้อมูลสำหรับการวิเคราะห์ พัฒนา ค้นถามข้อมูล และการออกรายงาน แดชบอร์ด การทำจินตทัศน์ข้อมูล เพื่อนำเสนอผลลัพธ์เชิงวิเคราะห์ให้แก่ผู้ใช้ อย่างไรก็ตาม บีไอโซลูชันส่วนใหญ่ในท้องตลาดมีลักษณะการทำงานที่ขึ้นกับแพลตฟอร์ม และต้องใช้ซอฟต์แวร์เฉพาะในการประมวลผลการแสดงผลภาพ งานวิจัยนี้จึงได้นำเสนอแนวทางการทำงานที่ไม่ขึ้นกับแพลตฟอร์มสำหรับการทำจินตทัศน์ข้อมูลและสารสนเทศโดยใช้ภาษาจำเพาะโดเมน หรือดีเอสแอล วากยสัมพันธ์ของดีเอสแอลที่พัฒนาขึ้นในงานนี้ถูกนิยามด้วยอีบีเอ็นเอฟ ประโยชน์ของการใช้แนวทางการแสดงผลภาพซึ่งไม่ขึ้นกับแพลตฟอร์มด้วยสคริปต์ดีเอสแอล เป็นผลอันเนื่องมาจากภาษาเพียงมุ่งเน้นไปที่ข้อมูลหลัก โดยไม่มีข้อจำกัดในการเลือกใช้ไลบรารีแสดงจินตทัศน์ข้อมูล

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2558

5770983421 : MAJOR SOFTWARE ENGINEERING
KEYWORDS: DOMAIN SPECIFIC LANGUAGE / BUSINESS INTELLIGENCE / VISUALIZATION / SEMANTIC MODEL

ITTIPOL NUNTARUJI: Domain Specific Language for Rendering Visual Components. ADVISOR: ASSOC. PROF. YACHAI LIMPIYAKORN, Ph.D., 61 pp.

Business Intelligence (BI) encompasses a variety of tools, applications and methodologies that enable organizations to collect and prepare data for analysis, develop and run queries against the data, and create reports, dashboards, data visualizations to make the analytical results available to end users. However, most BI solutions in the market are platform-dependent and require particular software for rendering. This research thus presents a platform-independent approach for visualizing data and information using domain specific language (DSL). The syntax of DSL developed in this work is defined with EBNF. The use of DSL scripts for rendering provides the benefit of platform-independent as the language simply focuses on primitive data, whereas the library for data visualizations remains intact.

Department: Computer Engineering Student's Signature .....
Field of Study: Software Engineering Advisor's Signature .....
Academic Year: 2015

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความอนุเคราะห์จากรองศาสตราจารย์ ดร.ญาใจ ลิ่มปิยะกรณ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ได้สละเวลาให้ความรู้ คำปรึกษา ตรวจสอบ และแก้ไขข้อผิดพลาดต่าง ๆ ตลอดจนการกำกับดูแล และคอยติดตามความก้าวหน้า ทำให้การวิจัยนี้สำเร็จไปได้ด้วยดี ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอขอบพระคุณ ศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล และ ดร.ภาสกร อภิรักษ์วรพินิต กรรมการสอบวิทยานิพนธ์ ที่กรุณาเสียสละเวลา ให้คำแนะนำ ตรวจสอบ และแก้ไขวิทยานิพนธ์ฉบับนี้

ขอขอบคุณพระคุณบิดา มารดา และญาติพี่น้องที่ให้การสนับสนุน และเป็นกำลังใจที่ดีให้เสมอมา และสนับสนุนด้านทุนทรัพย์ในการศึกษา รวมไปถึงทุกท่านที่มีส่วนช่วยเหลือในการทำวิทยานิพนธ์ครั้งนี้ ซึ่งมิได้กล่าวนามในที่นี้

ท้ายที่สุด ผู้วิจัยขอขอบพระคุณเพื่อน ๆ ทุกคน ที่คอยติดตาม และให้กำลังใจ รวมถึงท่านอื่น ๆ ที่มิได้กล่าวลงนามไว้ ณ ที่นี้ ที่มีส่วนทำให้วิทยานิพนธ์สำเร็จลุล่วงไปได้ด้วยดี ผู้วิจัยหวังเป็นอย่างยิ่งว่า วิทยานิพนธ์ฉบับนี้จะเป็นประโยชน์บ้างไม่มากก็น้อย สำหรับผู้ที่สนใจจะศึกษารายละเอียดต่อไป

สารบัญ

หน้า

หน้า

หน้า

สารบัญรูป

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

การแข่งขันทางธุรกิจที่สูงทำให้หลายองค์กรมีการตื่นตัวที่จะพัฒนาองค์กรเพื่อให้ทัดเทียมกับคู่แข่งขันอยู่เสมอ ปัญญาธุรกิจ (Business Intelligence— BI) ถูกใช้เป็นเครื่องมือหนึ่งในการวิเคราะห์และพัฒนาศักยภาพขององค์กรเพื่อหลีกเลี่ยงการสูญเสียโอกาสทางธุรกิจ ปัญญาธุรกิจเป็นกระบวนการสำหรับวิเคราะห์ข้อมูลและนำเสนอสารสนเทศที่เป็นประโยชน์ เพื่อช่วยให้ผู้บริหารองค์กร ผู้บริหารธุรกิจ และผู้ใช้อื่นๆ สามารถนำไปใช้สนับสนุนการตัดสินใจได้ ในการดำเนินการกระบวนการดังกล่าวจะประกอบด้วยการใช้เครื่องมือที่หลากหลาย แอปพลิเคชัน และระเบียบวิธีการต่างๆ เพื่อช่วยหน่วยงานในการรวบรวม จัดเตรียมข้อมูลสำหรับการวิเคราะห์ พัฒนา ค้นถามข้อมูล การออกรายงาน และการสร้างส่วนประกอบแสดงผลภาพ รวมทั้งการพัฒนา แดชบอร์ด (dashboard) ต่างๆ เพื่อนำเสนอผลลัพธ์เชิงวิเคราะห์ให้แก่ผู้ใช้

ปัญญาธุรกิจในท้องตลาดมีหลากหลาย อาทิเช่น Pentaho[1], Jaspersoft[2], SQL Server Reporting Services[3] ระบบซอฟต์แวร์เหล่านี้ต่างมีฟีเจอร์หลากหลายที่ตอบโจทย์ความต้องการ แต่ทว่ามีลักษณะการทำงานที่ขึ้นกับแพลตฟอร์ม (platform dependent) ดังนั้น การปรับเปลี่ยนแก้ไขจึงขาดความยืดหยุ่น มีข้อจำกัดในการเลือกใช้ซอฟต์แวร์หรือไลบรารี

นอกจากนี้ ปัญญาธุรกิจเป็นกระบวนการที่ขับเคลื่อนด้วยเทคโนโลยี การพัฒนาและปรับปรุงตัวปัญญาธุรกิจ ต้องใช้เวลาและแรงงานค่อนข้างมาก เนื่องจากต้องออกแบบให้รองรับกับเทคโนโลยีใหม่ๆที่หลากหลาย ทั้งเรื่องการจัดเตรียมข้อมูลและการแสดงผล ความจำเป็นที่ต้องมีการปรับปรุงและพัฒนาตัวปัญญาธุรกิจอย่างสม่ำเสมอ เพื่อให้ตอบโจทย์หรือแก้ปัญหาขององค์กร กอปรกับการแสดงผลหรือการออกรายงานแต่ละครั้งมักจะมีขั้นตอนในลักษณะที่คล้ายๆเดิม งานวิจัยนี้ได้เห็นถึงปัญหาของปัญญาธุรกิจดังกล่าว จึงได้นำเสนอการใช้ภาษาจำเพาะโดเมน (Domain Specific Language— DSL) เพื่อแสดงส่วนประกอบแสดงผลภาพ ซึ่งเป็นส่วนหนึ่งของปัญญาธุรกิจ ข้อดีของแนวทางที่นำเสนอ คือ การใช้ภาษาจำเพาะโดเมน จะได้แบบจำลองความหมาย (Semantic model) เพื่อใช้ในการแสดงส่วนประกอบแสดงผลภาพ ทำให้การแสดงผลลัพธ์ไม่ขึ้นกับแพลตฟอร์ม ผู้ใช้มีอิสระมากขึ้นในการเลือกใช้ไลบรารีสำหรับการแสดงผล นอกจากนี้ ผู้เชี่ยวชาญโดเมน (domain expert) สามารถเปลี่ยนแปลงสคริปต์ภาษาจำเพาะโดเมนได้ด้วยตนเอง เมื่อความต้องการเปลี่ยนโดยไม่ต้องรอให้ผู้พัฒนา (developer) ทำการแก้ไขให้ ซึ่งสามารถช่วยลดความผิดพลาดจากการสื่อสารคลาดเคลื่อน และสามารถส่งผลให้ผลิตภาพ (productivity) กระบวนการเพิ่มมากขึ้น

1.2 วัตถุประสงค์ของการวิจัย

เพื่อนำเสนอแนวทางการออกแบบและพัฒนาภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ แบบไม่ขึ้นต่อแพลตฟอร์ม

1.3 ขอบเขตงานวิจัย

1) รองรับส่วนประกอบการแสดงผล ได้แก่ แผนภูมิแท่ง แผนภูมิเส้น แผนภูมิวงกลม ตาราง รูปภาพ และ ตัวอักษร เท่านั้น
2) ประเมินผลงานวิจัยจากความถูกต้องการสร้างส่วนประกอบแสดงผลภาพ ที่มีจำนวนพอเหมาะสมเหตุผล

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1) ได้วิธีการและเครื่องมือสร้างสคริปต์ภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ
2) ได้แนวทางการสร้างส่วนประกอบแสดงผลภาพที่ไม่ขึ้นกับแพลตฟอร์ม

1.5 วิธีดำเนินการวิจัย

1) ศึกษาและทำความเข้าใจทฤษฎีและงานวิจัยที่เกี่ยวข้อง
2) ศึกษาการใช้เครื่องมือที่ใช้สำหรับการพัฒนาโปรแกรม
3) วิเคราะห์และกำหนดระเบียบวิธีวิจัย
4) ออกแบบ ตั้งสมมติฐาน ที่เกี่ยวข้องกับงานวิจัย
5) พัฒนาระบบ
6) ทดสอบและประเมินผลงานวิจัย
7) สรุปผลงานวิจัย
8) ตีพิมพ์ผลงานทางวิชาการ
9) จัดทำวิทยานิพนธ์

1.6 ลำดับการจัดเรียงเนื้อหาในวิทยานิพนธ์

เนื้อหาวิทยานิพนธ์ฉบับนี้แบ่งออกเป็น 6 บท โดยเริ่มจาก บทที่ 1 บทนำจะกล่าวถึงความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ของงานวิจัย ขอบเขตงานวิจัย ประโยชน์ที่คาดว่าจะได้รับ วิธีดำเนินการวิจัย และ ผลงานตีพิมพ์จากวิทยานิพนธ์ บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้องที่นำมาใช้ในงานวิจัยนี้ บทที่ 3 แนวคิดและวิธีวิจัย บทที่ 4 การพัฒนาระบบ สถาปัตยกรรมระบบ สภาพแวดล้อม และเครื่องมือที่ใช้ในการพัฒนา และวิธีการพัฒนาระบบ บทที่ 5 การทดสอบระบบ ทดสอบการทำงานของระบบ และการวิเคราะห์ผล และบทที่ 6 สรุปผลการวิจัย ข้อจำกัด และแนวทางสำหรับการทำวิจัยต่อในอนาคต

1.7 ผลงานที่ตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์ฉบับนี้ได้รับการตีพิมพ์เป็นบทความทางวิชาการในหัวข้อเรื่อง “DSL for Business Intelligence Visualization” โดย นายอิทธิพล นันทรุจิ และ รศ.ดร.ญาใจ ลิ่มปิยะกรณ์ ในหนังสือรวมบทความการประชุมวิชาการนานาชาติ 2016 6th International Workshop on Computer Science and Engineering (WCSE 2016) ณ กรุงโตเกียว ประเทศญี่ปุ่น วันที่ 17-19 มิถุนายน 2559 หน้า 407-410

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ภาษาจำเพาะโดเมน (Domain Specific Language - DSL)

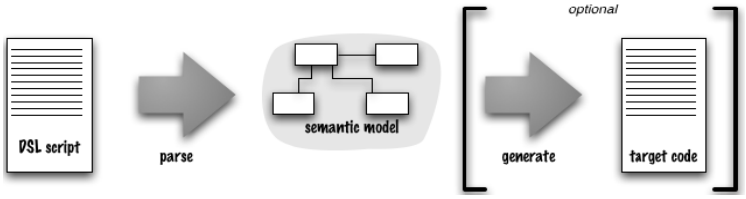
ภาษาจำเพาะโดเมน [4] เป็นลักษณะภาษาคอมพิวเตอร์ที่ถูกสร้างขึ้นมาสำหรับการทำงานในลักษณะใด ลักษณะหนึ่ง โดยเฉพาะเจาะจงตามการทำงานในโดเมนนั้นๆ สามารถแบ่งออกได้เป็น 3 ประเภทหลัก ได้แก่ 1) ภาษาจำเพาะโดเมนภายนอก (External DSL) 2) ภาษาจำเพาะโดเมนภายใน (Internal DSL) และ 3) ตัวปรับแต่งภาษา (Language Workbench)

1. ภาษาจำเพาะโดเมนภายนอก – เป็นภาษาที่ไม่เกี่ยวข้องจากภาษาโปรแกรมหลัก (Host Language) ที่ทำงานอยู่ โดยทั่วไปจะมีไวยากรณ์เป็นของตัวเอง และใช้ ตัวแจงส่วน (parser) ในการประมวลผลไปที่แอปพลิเคชันเพื่อให้ทำงานได้ แต่ก็ยังสามารถใช้ไวยากรณ์ที่มีลักษณะสามัญได้ เช่น XML ตัวอย่างของภาษาจำเพาะโดเมนภายนอก เช่น SQL เป็นต้น
2. ภาษาจำเพาะโดเมนภายใน – เป็นภาษาที่ขยายความสามารถจากภาษาโปรแกรมหลัก มีลักษณะภาษาคล้ายเป็นภาษาที่ถูกกำหนดขึ้นเอง ตัวอย่างของภาษาจำเพาะโดเมนภายใน ได้แก่ LISP
3. ตัวปรับแต่งภาษา – เป็นเครื่องมือที่ช่วยในการพัฒนาภาษาจำเพาะโดเมน ซึ่งไม่เพียงแค่ใช้ในการสร้างโครงสร้างของภาษาจำเพาะโดเมนเท่านั้น แต่ยังรวมถึงการแก้ไขสภาพแวดล้อมของเครื่องมือในการพัฒนาภาษาอีกด้วย

ประโยชน์ของภาษาจำเพาะโดเมนที่เด่นชัดคือ ทำให้ผู้ใช้ที่อยู่ในโดเมนเหล่านั้น สามารถทำความเข้าใจในการใช้ภาษาได้ง่าย ทั้งในด้านการเขียน และ การอ่าน โดยไม่จำเป็นต้องมีความเชี่ยวชาญในการเขียนโปรแกรมมาก่อน และด้วยความที่เป็นภาษาที่สามารถเข้าใจได้ง่าย ทำให้เมื่อพบข้อผิดพลาดในการทำงาน จะสามารถตรวจสอบ และสื่อสารกับทางผู้พัฒนาระบบได้อย่างมีประสิทธิภาพ เป็นการลดช่องว่างระหว่างการสื่อสารของผู้ใช้งานเชิงธุรกิจ และผู้พัฒนาระบบได้[5] นอกจากนี้ยังสามารถเพิ่มคุณภาพทั้งในด้านผลิตภาพ (Productivity) ความสามารถบำรุงรักษา (Maintainability) ความสามารถทำงานร่วม (Interoperability) ความสามารถเคลื่อนย้าย (Portability) ความสามารถทดสอบ (Testability) และความสามารถนำมาใช้ซ้ำ (Reusability) [5-9]

รูปแบบการทำงานแบบหนึ่งในการทำงานของภาษาจำเพาะโดเมน ประกอบด้วยตัวแจงส่วนที่ทำการอ่านสคริปต์ภาษาจำเพาะโดเมน เพื่อประมวลผลสร้างแบบจำลองความหมาย (Semantic

model) ซึ่งใช้พัฒนาเป็นโค้ดปลายทางที่ต้องการ ดังแสดงในรูปที่ 1 โดย Semantic Model ที่เกิดขึ้นจะเป็น Subset ของ Domain Model ที่มีการสร้างไว้ก่อนหน้า



รูปที่ 1 การทำงานของภาษาจำเพาะโดเมน

อย่างไรก็ตาม การออกแบบและพัฒนาภาษาจำเพาะโดเมนมีความท้าทายในเรื่องของการค้นหานามธรรม (abstraction) ที่ถูกต้อง และวากยสัมพันธ์ (syntax) ที่เหมาะสม [10]

2.1.2 ปัญญาธุรกิจ (Business Intelligence - BI)

ปัญญาธุรกิจ [11] คือโปรแกรมประยุกต์ (Application) โครงสร้างพื้นฐาน (Infrastructure) เครื่องมือ (Tools) และ แนวปฏิบัติที่เป็นเลิศ (Best Practice) ที่สามารถเข้าถึงข้อมูล และวิเคราะห์ข้อมูล เพื่อการพัฒนาสมรรถนะ (Performance) และสนับสนุนการตัดสินใจที่เหมาะสม โดยทั่วไปการแสดงผลลัพธ์ปัญญาธุรกิจสามารถแบ่งออกได้เป็น 4 ประเภทหลักๆ ตามความเหมาะสมในการใช้งานข้อมูลของแต่ละประเภทธุรกิจ ได้แก่ 1) Reporting, 2) Analysis, 3) Monitoring, และ 4) Prediction [12] คำอธิบายประเภทการแสดงผลลัพธ์ปัญญาธุรกิจสรุปโดยย่อในตารางที่ 1

ตารางที่ 1 ประเภทการแสดงผลลัพธ์ปัญญาธุรกิจ

Reporting	มุ่งเน้นการรวบรวมนำข้อมูลที่สนใจมาสรุปในรูปแบบของเอกสารรายงาน โดยจะทำการแสดงถึงข้อมูลที่เกิดขึ้นในบริษัทในช่วงเวลาหนึ่งที่สนใจ
Analysis	มุ่งเน้นการแสดงผลข้อมูลที่ถูกนำมาวิเคราะห์ ทำให้สามารถทำความเข้าใจกับข้อมูลที่ได้มา ได้อย่างถูกต้อง โดยสามารถแบ่งออกได้เป็น 3 ประเภทหลัก ดังนี้ 1) Spreadsheet Analysis ทำการวิเคราะห์ข้อมูลที่อยู่ในรูปแบบ spreadsheet ซึ่งมีเป้าหมายในการประเมินสมรรถนะของหน่วยงาน หรือบริษัท 2) Ad-Hoc Query เป็นซอฟต์แวร์ที่ให้ผู้ใช้งานทำการ data query ได้ด้วยตนเอง เช่นการสร้าง data query เพื่อแสดงจำนวนสินค้าที่ขายได้ของสินค้านั้นในช่วงเวลาหนึ่ง

	3) Visualization Tools เป็นซอฟต์แวร์ที่นำข้อมูลดิบมาทำการสร้างจินตทัศน์ (visualization) เพื่อให้ผู้ใช้งานสามารถทำความเข้าใจได้ง่าย เช่น แผนภูมิรูปวงกลม (Pie chart) ฯลฯ
Monitoring	มุ่งเน้นการติดตามข้อมูลแบบ real-time ซึ่งสามารถเลือกดูเฉพาะข้อมูลที่สนใจได้เป็นช่วง โดยสามารถแบ่งออกได้เป็น 3 ประเภทหลักดังนี้ 1) Dashboard เป็นศูนย์รวมของข้อมูลที่สำคัญทั้งหมด โดยมีการแสดงผลในรูปแบบกราฟิกส์เพื่อความเข้าใจง่าย เช่น Google+ dashboard 2) Key Performance Indicators (KPIs) เป็นตัววัดสมรรถนะของโครงการ เช่น Return on Investment (ROI) 3) Business Performance Management เป็นระบบที่ถูกสร้างเพื่อทำให้แน่ใจว่าสมรรถนะของงานในโครงการยังเป็นไปตามเป้าหมายที่ตั้งไว้หรือไม่ เช่น Balanced Scorecard
Prediction	มุ่งเน้นการทำนายว่าจะเกิดเหตุการณ์อะไรในอนาคต หรือแนวโน้มของข้อมูลจะเป็นอย่างไร โดยสามารถแบ่งออกเป็น 2 ประเภทหลักดังนี้ 1) Data Mining เป็นการค้นหารูปแบบและ ความสัมพันธ์ของชุดข้อมูลขนาดใหญ่ โดยเป้าหมายหลักของ data mining คือการจำแนกหรือแปลงข้อมูลเป็นผลลัพธ์ที่สามารถเข้าใจและนำไปใช้ได้ต่อไป 2) Predictive Modelling เป็นแบบจำลองในการทำนายผลลัพธ์ที่ได้ของงานหรือหาความน่าจะเป็นของผลลัพธ์ที่จะได้มา

ตัวอย่างของการแสดงผลปัญญาธุรกิจทั้ง 4 ประเภท: Reporting, Analysis, Monitoring, และ Prediction แสดงดังรูปที่ 2 รูปที่ 3 รูปที่ 4 และ รูปที่ 5 ตามลำดับ

For the year ended December 31 ($ millions)	2002	2001	2000
Cash flows from operating activities			
Earnings from continuing operations	1,898	3,571	973
Adjustments to reconcile earnings from continuing operations to cash flows from operating activities:			
Amortization expense	3,146	3,826	3,352
Net benefit plans credit	(33)	(121)	(109)
Restructuring and other charges	805	915	–
Impairment charge	770	–	–
Net gains on investments	(2,435)	(4,088)	(45)
Future income taxes	602	682	(130)
Non-controlling interest	668	186	323
Other items	(298)	(894)	(545)
Changes in non-cash working capital	(592)	157	(1,613)
	4,531	4,234	2,206
Cash flows from investing activities			
Capital expenditures	(3,771)	(4,999)	(3,652)
Investments	(6,604)	(535)	(4,729)
Divestitures	3,230	4,749	654
Other items	10	(122)	(120)
	(7,135)	(907)	(7,847)
Cash flows from financing activities			
Increase (decrease) in notes payable and bank advances	(210)	(2,744)	3,730
Issue of long-term debt	4,908	2,443	2,447
Repayment of long-term debt	(2,893)	(1,221)	(1,431)
Issue of common shares	2,693	71	36
Costs relating to the issuance of common and preferred shares	(78)	–	–
Purchase of common shares for cancellation	–	(191)	(384)
Issue of preferred shares	510	–	–
Redemption of preferred shares	(306)	–	–
Issue of equity securities and convertible debentures by subsidiaries to non-controlling interest	206	1,459	540
Redemption of preferred shares by subsidiaries	–	(347)	(295)
Dividends paid on common and preferred shares	(1,042)	(1,033)	(928)
Dividends paid by subsidiaries to non-controlling interest	(468)	(357)	(240)
Other items	(46)	72	110
	3,274	(1,848)	3,585
Effect of exchange rate changes on cash and cash equivalents	3	(2)	(19)
Cash provided by (used in) continuing operations	673	1,477	(2,075)
Cash used in discontinued operations	(936)	(1,168)	(60)
Net increase (decrease) in cash and cash equivalents	(263)	309	(2,135)
Cash and cash equivalents at beginning of year	569	260	2,395
Cash and cash equivalents at end of year	306	569	260

รูปที่ 2 ตัวอย่างปัญญาธุรกิจประเภท Reporting (IFRS Balance Sheet)[13]

Time Tracking Log

[Company Name]

Name: _____

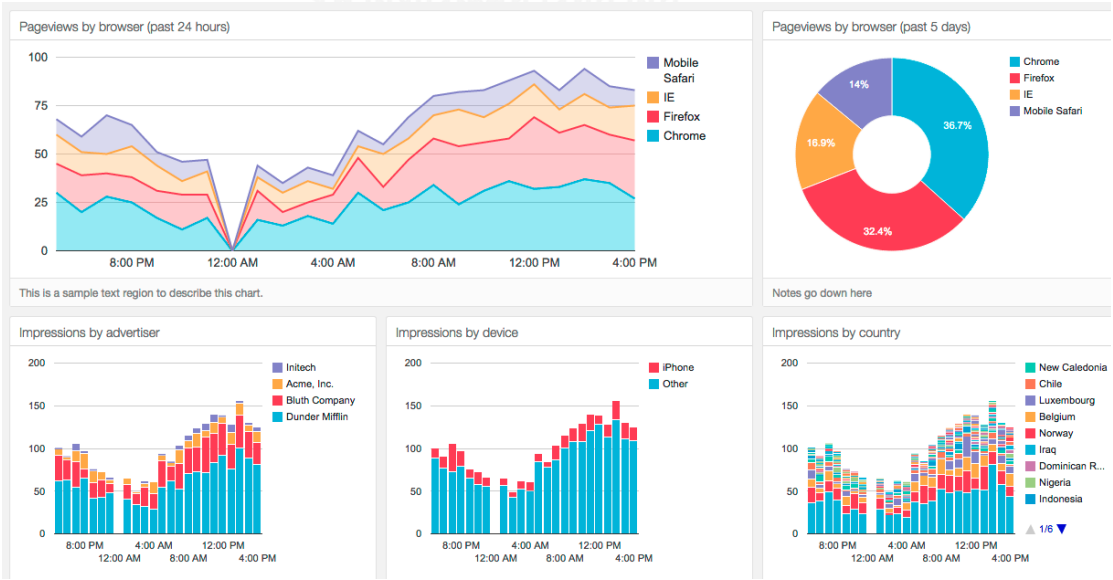
Today	
Date:	2/20/13
HOURS	1.75

Current Period		
Start:	2/18/13	
End	2/25/13	(7 days)
HOURS	14.5	

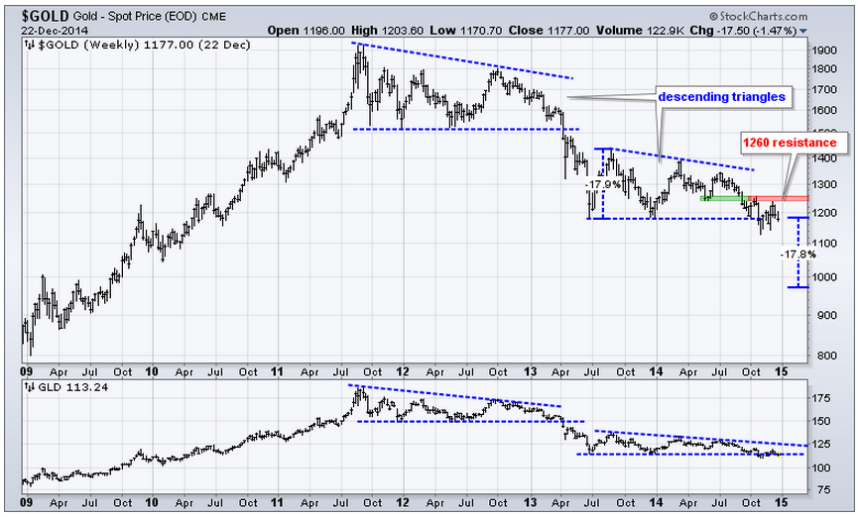
This Week	
2/18	8
2/19	4.75
2/20	1.75
2/21	0
2/22	0
2/23	0
2/24	0
TOTAL	14.5

Date	Project ID	Task ID	Notes	Start Time	Breaks (minutes)	End Time	Hours	Current Period	Billed	Invoice #
2/18/13	Other	Holiday	Presid	9:00 AM		5:00 PM	8.00	yes	X	
2/19/13	Customer 1			9:30 AM	45	3:00 PM	4.75	yes		
2/20/13	Customer 1			8:00 AM	15	10:00 AM	1.75	yes		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		
							0.00	no		

รูปที่ 3 ตัวอย่างปัญญาธุรกิจประเภท Analysis (Timesheet)[14]



รูปที่ 4 ตัวอย่างปัญญาธุรกิจประเภท Monitoring (Dashboard)[15]



รูปที่ 5 ตัวอย่างปัญญาธุรกิจประเภท Prediction (ข้อมูลกราฟจากตลาดหุ้น)[16]

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 A Domain-Specific Visual Language for Report Writing Using Microsoft DSL Tools

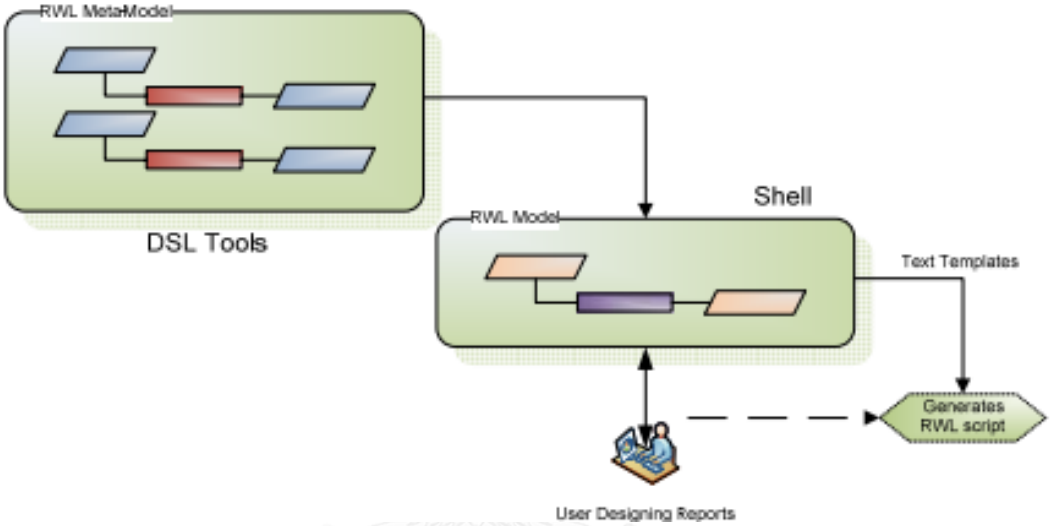
Dantra และคณะ[17] ได้นำเสนอภาษาภาพจำเพาะโดเมน (Domain-Specific Visual Language) สำหรับการออกรายงาน โดยใช้เครื่องมือ Microsoft Domain-Specific Language Tools ในการออกแบบ ซึ่งผลลัพธ์ที่ได้คือ สคริปต์ของ Prism report writer language (RWL) ดังรูปที่ 6 โดย ภาษา RWL เป็นภาษาที่ใช้ในการออกรายงานของกลุ่มบริษัท Prism

```
//---------------------------------------------------------------------------
// <auto-generated>
//     This code was generated by a tool.
//
//     Changes to this file may cause incorrect behavior and will be lost if
//     the code is regenerated.
//
//     Generated on 6/06/2009 1:35:16 p.m.
//     ToolVersion 1.0.0
// </auto-generated>
//---------------------------------------------------------------------------

Code        RW_EXAMPLE_03
Type        Standard
Access      ST$A
Name        "Report Writer Example 03"


Scan RM
    Print RM_CUST + RM_NAME;
    Scan QM
        Choose (QM_CUST_CODE, Match, RM_CUST)

        Print QM_TITLE;
    End
End
```

รูปที่ 6 สคริปต์ภาษา RWL [17]

งานวิจัยได้นำเสนอกระบวนการทำงานดังรูปที่ 7 โดยแบ่งออกเป็น ขั้นตอนหลัก ประกอบด้วย

1. ออกแบบเมตาโมเดล (meta-model) ของภาษา RWL
2. ข้อจำกัดที่ไม่สามารถอธิบายได้ด้วยเมตาโมเดลจะถูกเปลี่ยนไปอยู่ในรูปแบบของกฎ
3. พัฒนาสัญกรณ์ภาพ (visual notation) และ เครื่องมือสำหรับเมตาโมเดล ที่ให้ผู้ใช้งาน (user) สามารถใช้งานได้

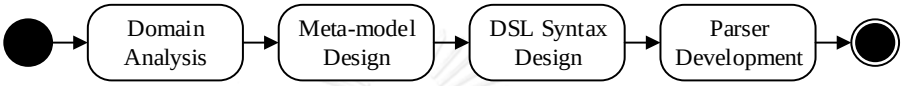


รูปที่ 7 กระบวนการออกแบบรายงานโดยใช้ภาษาภาพจำเพาะโดเมนเชิงพาณิชย์ [17]

อย่างไรก็ตาม งานวิจัยดังกล่าวได้พัฒนาภาษาจำเพาะโดเมนที่สามารถใช้ได้กับการออกรายงาน ซึ่งต้องมีการใช้เครื่องมือที่มีลักษณะเฉพาะเพื่อการออกรายงานอีก เนื่องจากผลลัพธ์ที่ได้เป็นเพียงสคริปต์ที่สามารถใช้ได้กับเครื่องมือของทางกลุ่มบริษัท Prism เท่านั้น

บทที่ 3
แนวคิดและวิธีวิจัย

งานวิจัยนี้ได้นำเสนอแนวทางการออกแบบและพัฒนาภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลประกอบด้วย 4 ขั้นตอนหลัก ได้แก่ 1) การวิเคราะห์โดเมน (Domain Analysis) 2) การออกแบบเมตาโมเดล (Meta-model design) 3) การออกแบบวากยสัมพันธ์ภาษาจำเพาะโดเมน (DSL Syntax design) และ 4) การพัฒนาตัวแจงส่วน (Parser development) ดังแสดงในรูปที่ 8



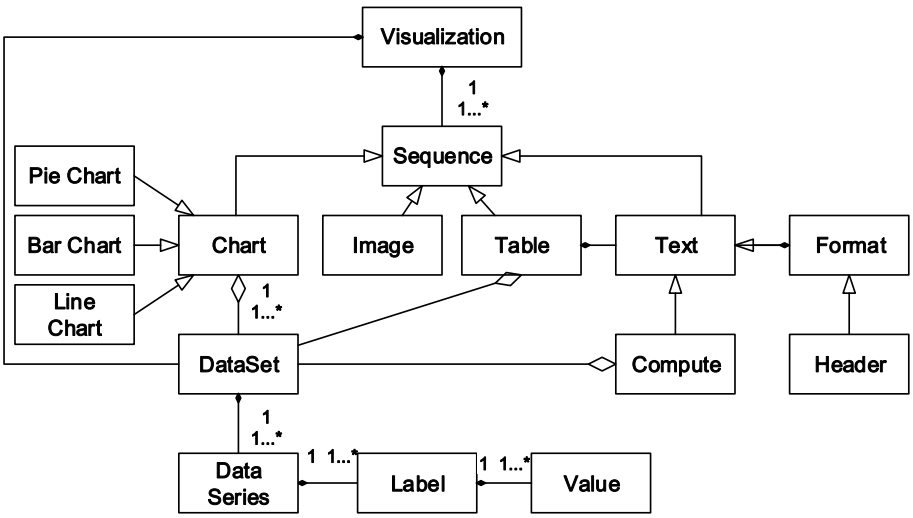
รูปที่ 8 ขั้นตอนการพัฒนาภาษาจำเพาะโดเมนสำหรับสร้างส่วนประกอบแสดงผลภาพ

3.1 การวิเคราะห์โดเมน

เป็นขั้นตอนการสำรวจซอฟต์แวร์ปัญญาธุรกิจที่มีอยู่ในท้องตลาด และนำมาวิเคราะห์โดยให้แยกส่วนของจินตทัศน์ออกจากส่วนการทำงานหลัก เพื่อทำการหาส่วนประกอบดั้งเดิม (Primitive components) รวมถึงข้อมูลอื่นๆ ที่เกี่ยวข้อง ข้อมูลที่ได้จากการวิเคราะห์จะถูกนำมาใช้ในการออกแบบและสร้างเมตาโมเดล วากยสัมพันธ์ และตัวแจงส่วนเพื่อทำการแปลงสคริปต์ภาษาเป็นแบบจำลองความหมาย

3.2 การออกแบบเมตาโมเดล

ส่วนประกอบแสดงผลภาพ (visual components) ของปัญญาธุรกิจที่สนใจ ประกอบด้วย ข้อความ (Text) ตาราง (Table) รูปภาพ (Image) แผนภูมิ (Chart) ในส่วนแผนภูมิจะรองรับแผนภูมิหลักๆ ได้แก่ แผนภูมิแท่ง (Bar Chart) แผนภูมิวงกลม (Pie Chart) และแผนภูมิเส้น (Line Chart) ในการแสดงผลจะมีการแสดงผลเป็นลำดับของส่วนประกอบต่าง ๆ ซึ่งสามารถมีได้มากกว่า 1 ส่วนประกอบ โดยส่วนประกอบที่เป็นข้อความ สามารถเป็นข้อความปกติ และหัวเรื่องได้ เซตของข้อมูลสามารถนำมาประมวลผล และแสดงผลในรูปแบบของแผนภูมิ ตาราง และข้อความคำนวณได้ โดยรูปที่ 9 แสดงการออกแบบเมตาโมเดลของส่วนประกอบแสดงผลภาพในงานวิจัยนี้



รูปที่ 9 เมตาโมเดลของส่วนประกอบแสดงผลภาพ

3.3 การออกแบบวากยสัมพันธ์

ในการออกแบบวากยสัมพันธ์ ไวยากรณ์ของภาษาถูกสร้างขึ้นด้วย Extended Backus–Naur Form (EBNF) [18] โดยการออกแบบจะอ้างอิงตามส่วนประกอบข้อมูลหลักที่ได้จากการวิเคราะห์ในขั้นตอนที่ 1 และ เมตาโมเดลในขั้นตอนที่ 2 ในส่วนประกอบแสดงผลภาพ ประกอบไปด้วยข้อมูลที่อยู่ในรูปแบบข้อความ ตาราง รูปภาพ และแผนภูมิ รวมถึงการระบุเซตของข้อมูลเพื่อการนำมาแสดงผลในข้อความคำนวณ ตาราง และแผนภูมิ โดยที่เซตของข้อมูลจะถูกระบุด้วยคำสั่งในภาษาเอสคิวแอล (SQL) ส่วนประกอบที่เป็นรูปภาพจะทำการระบุข้อมูลเป็นลิงค์ของรูปภาพ ส่วนประกอบที่เป็นตารางสามารถระบุชื่อตาราง จำนวนบรรทัดที่แสดง และฟิลด์ของข้อมูลที่นำมาแสดง รวมถึงสามารถกำหนดชื่อหัวตารางได้ ส่วนประกอบที่เป็นแผนภูมิสามารถระบุชื่อเรื่อง ชนิดของแผนภูมิ ชุดข้อมูล(Data Series) โดยชุดข้อมูลสามารถแสดงผลได้หลายชุดข้อมูล ส่วนประกอบที่เป็นข้อความสามารถระบุข้อมูลข้อความปกติ ข้อความหัวเรื่อง และข้อความคำนวณได้ โดยข้อความคำนวณจะครอบคลุมผลนับ(Count) ผลรวม(Summary) ผลเฉลี่ย(Average) ผลลัพธ์ที่น้อยที่สุด(Minimum) และผลลัพธ์ที่มากที่สุด(Maximum) โดยรูปที่ 10 แสดงวากยสัมพันธ์ภาษาจำเพาะโดเมนที่กำหนดขึ้นสำหรับการสร้างส่วนประกอบแสดงผลภาพในงานวิจัยนี้

```
visual  = 'visualization' '['{visual_attrs} ']';
visual_attrs = datasource
               |image
               |table
               |chart
               |text ;
datasource = 'data' '=' data_value;
image = 'image' '=' data_value;
table = 'table' '['{table_attrs} ']';
table_attrs =   table_title
               |table_paging
               |datasource
               |table_select_cols;
table_title = 'title' '=' data_value;
table_paging = 'num-row' '=' data_num_value;
table_select_cols = 'select' '=' table_cols;
table_cols =  '('[data_field_name]data_field_value{','[data_field_name]data_field_value}')';
chart = chart_type '['chart_attrs{chart_attrs} ']';
chart_attrs =   chart_title
               |datasource
               |chart_series;
chart_type =    'barchart'
               |'piechart'
               |'linechart';
chart_title = 'title' '=' data_value;
chart_series = 'series-set' id '['series_attrs{series_attrs} ']';
series_attrs =  datasource
               |series_label series_value ;
series_label = 'label' '=' data_field_value;
series_value = 'value' '=' data_field;
text = ('text'|'header' header_style) '=' data_field;
header_style =  'H1'
               |'H2'
               |'H3';
data_field =    aggregate_field
               |data_field_value
               |data_value;
aggregate_field = (Sum|Count|Avg|Min|Max)'('data_field')';
data_num_value = integer;
data_field_value = '<<'string'>>';
data_value =    string
               |integer;
```

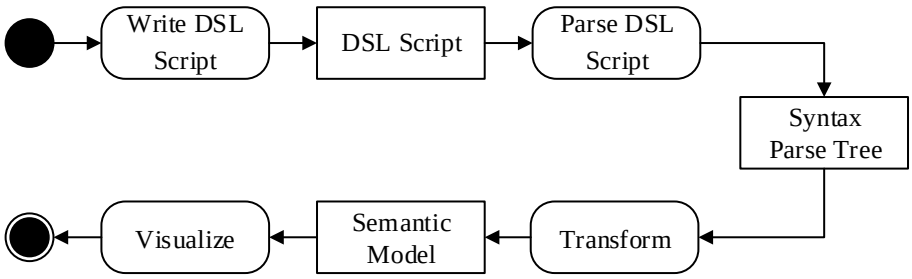
รูปที่ 10 วากยสัมพันธ์ของภาษาจำเพาะโดเมนสำหรับสร้างส่วนประกอบแสดงผลภาพ

3.4 พัฒนาตัวแจงส่วน

ในการพัฒนาตัวแจงส่วน จะทำการเลือกใช้เครื่องมือในการสร้างตัวแจงส่วน ซึ่งจะนำเข้าสคริปต์ภาษา และแปลงให้อยู่ในรูปของแบบจำลองความหมาย

3.5 ตัวอย่างการใช้งาน

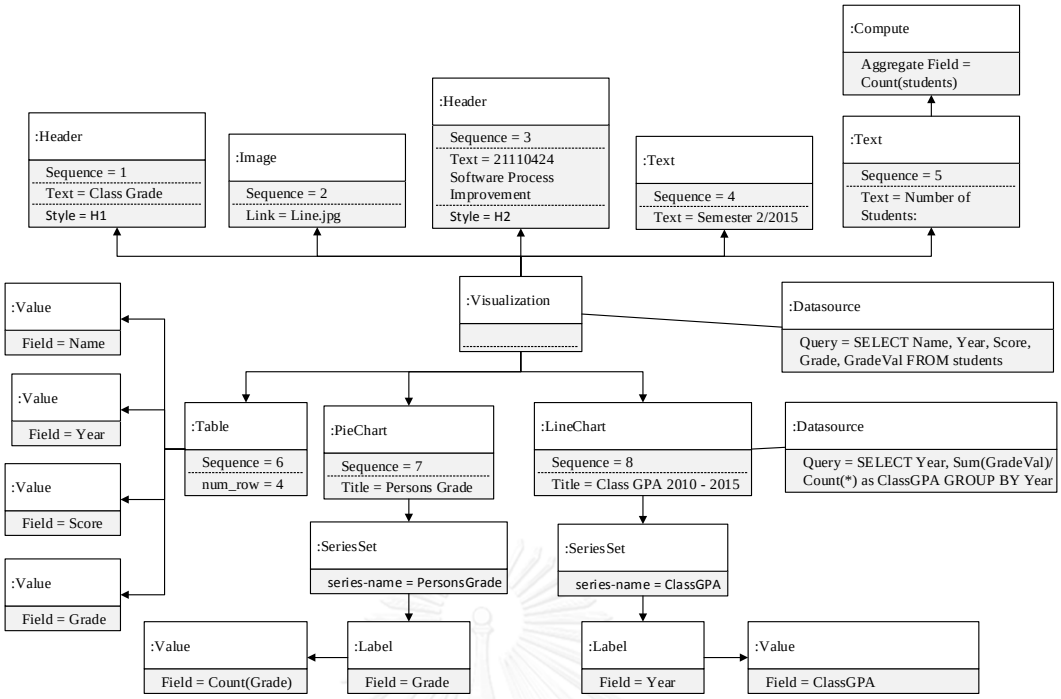
รูปที่ 11 แสดงขั้นตอนการสร้างส่วนประกอบแสดงผลภาพจากสคริปต์ภาษาจำเพาะโดเมน ตัวอย่างสคริปต์ (รูปที่ 12) ประกอบด้วยส่วนประกอบการมองเห็น ได้แก่ หัวข้อหลัก (Header) รูปภาพ (Image) หัวข้อรอง (Sub header) ข้อความปกติ (Normal Text) ตาราง แผนภูมิวงกลม และแผนภูมิแท่ง เป็นต้น การแจงส่วนสคริปต์จะได้แบบจำลองความหมาย (รูปที่ 13) ซึ่งเมื่อผ่านการประมวลผลภาพแล้วจะได้ส่วนประกอบแสดงผลภาพ (รูปที่ 14)



รูปที่ 11 ขั้นตอนการสร้างส่วนประกอบแสดงผลภาพจากสคริปต์ภาษาจำเพาะโดเมน

```
visualization
[
    data = "SELECT Name, Year, Score, Grade, GradeVal FROM students"
    header H1 = "Grade Report"
    image = "Line.jpg"
    header H2 = "2110424 Software Process Improvement"
    text = "Semester 2/2015"
    text = "Number of Students : " Count(<<Name>>)
    table
    [
        num-row = 4
        select = (<<Name>>,<<Year>>,<<Score>>,<<Grade>>)
    ]
    piechart
    [
        title = "Persons Grade"
        series-set PersonsGrade
        [
            label = <<Grade>>
            value = Count(<<Grade>>)
        ]
    ]
    linechart
    [
        data = "SELECT Year, Sum(GradeVal)/Count(*) as ClassGPA GROUP BY Year"
        title = "Class GPA 2010 - 2015"
        series-set ClassGPA
        [
            label = <<Year>>
            value = <<ClassGPA>>
        ]
    ]
]
```

รูปที่ 12 ตัวอย่างสคริปต์ภาษาจำเพาะโดเมน



รูปที่ 13 แบบจำลองความหมายของตัวอย่างสคริปต์ภาษาจำเพาะโดเมน

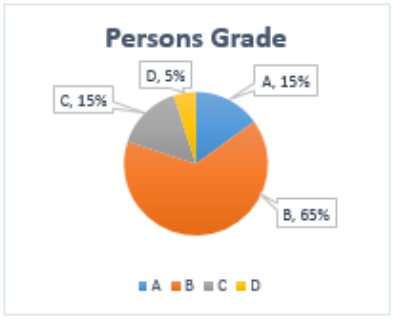
Class Grade

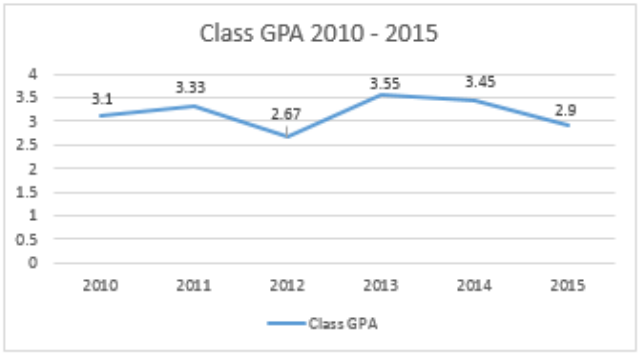
2110424 Software Process Improvement

Semester 2/2015
Number of Students: 20

Name	Year	Score	Grade
John	4	91	A
Carol	4	87	A
Willey	4	85	A
Tommy	3	77	B

1 2 3 4 5



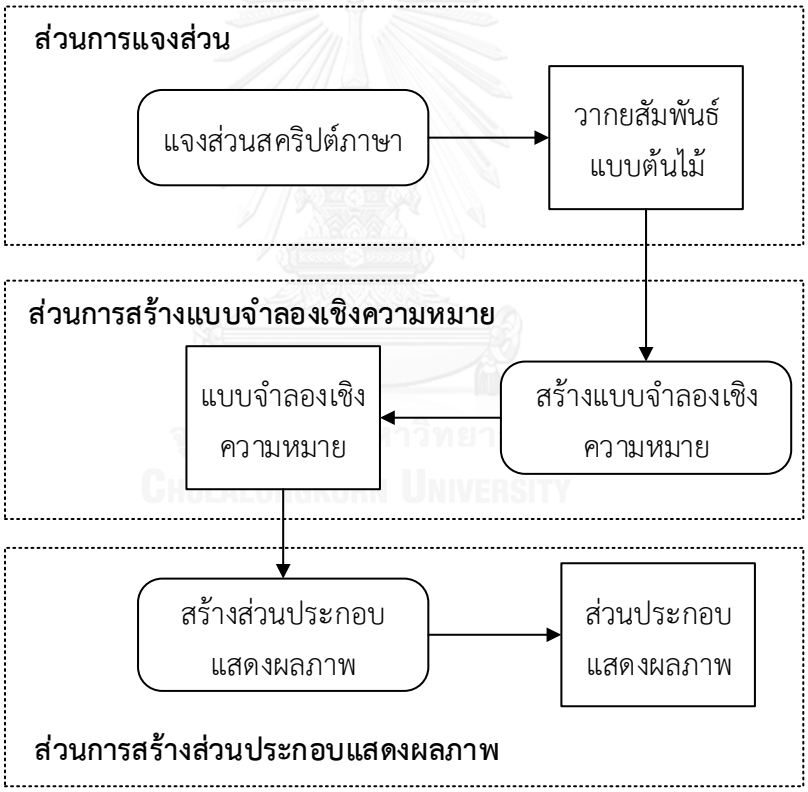


รูปที่ 14 การแสดงผลของตัวอย่างสคริปต์ภาษาจำเพาะโดเมน

บทที่ 4
การพัฒนาระบบ

4.1 สถาปัตยกรรมระบบ

เครื่องมือที่พัฒนาแบ่งออกเป็นสามส่วนหลัก ได้แก่ ส่วนการแจงส่วนภาษาจำเพาะโดเมน ส่วนการสร้างแบบจำลองความหมาย และส่วนการแสดงส่วนประกอบแสดงผลภาพ โดยทั้งสามส่วนนั้นมีความสัมพันธ์กันดังรูปที่ 15 การแจงส่วนภาษาจำเพาะโดเมน จะทำหน้าที่แปลงสคริปต์ภาษาให้อยู่ในรูปแบบวากยสัมพันธ์แบบต้นไม้ จากนั้นจะนำวากยสัมพันธ์แบบต้นไม้มาสร้างเป็นแบบจำลองความหมาย โดยทำให้อยู่ในรูปแบบเอกซ์เอ็มแอล และจึงนำแบบจำลองความหมายมาสร้างส่วนประกอบแสดงผลภาพ โดยแสดงผลอยู่ในรูปแบบเว็บแอปพลิเคชัน



รูปที่ 15 ภาพรวมการทำงานของระบบ

4.2 สภาพแวดล้อมและเครื่องมือที่ใช้ในการพัฒนา

สภาพแวดล้อม และเครื่องมือที่ใช้ในการพัฒนาระบบ ประกอบด้วยรายการฮาร์ดแวร์ และซอฟต์แวร์ดังต่อไปนี้

4.2.1 สภาพแวดล้อม

1. หน่วยประมวลผลอินเทล คอร์ ไอ 7-5500U 2.40 กิกะเฮิร์ต (CPU Intel Core i7-5500U 2.4GHz)

2. หน่วยความจำ 8 กิกะไบต์ (8 GB RAM)

3. ฮาร์ดดิสก์ความจุ 500 กิกะไบต์ (500 GB HDD)

4. ระบบปฏิบัติการไมโครซอฟต์วินโดว์ 7 อัลติเมต เซอร์วิสแพค 1 (Microsoft Windows 7 Ultimate Service Pack 1) แบบ 64 บิท

4.2.2 เครื่องมือที่ใช้ในการพัฒนา

1. ไมโครซอฟต์ วิชวลสตูดิโอ 2010 เซอร์วิสแพค 1 (Microsoft Visual Studio Service Pack 1)

2. ไมโครซอฟต์ ดอทเน็ตเฟรมเวิร์ค 4.0 (Microsoft .NET Framework 4.0)

3. ไอโรนี 2013_12_12 (Irony - .NET Language Workbench _2013_12_12)

4. เอเอสพีดอทเน็ต 4.0 (ASP.NET 4.0)

4.3 การพัฒนาระบบ

4.3.1 การพัฒนาตัวแจงส่วน

การพัฒนาตัวแจงส่วนจะใช้เครื่องมือพัฒนาไอโรนี เวอร์ชัน 2013_12_12 เพื่อทำการสร้างไวยากรณ์ และตัวแจงส่วนของระบบ ซึ่งสามารถพัฒนาโดยใช้ไวยากรณ์ตามที่ได้ออกแบบไว้แล้วในบทที่ 3 โดยในการพัฒนาจะเริ่มต้นจากการสร้างคลาสไวยากรณ์ของภาษา ดังรูปที่ 16

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Irony.Parsing;
using Irony.Interpreter;
using Irony.Interpreter.Ast;
namespace Chula.GrammarSample
{
    [Language("VizDSL", "1.0", "Visuallization DSL")]
    public class VizDSLGrammar : Grammar
    {
        public VizDSLGrammar()
            : base(caseSensitive: false)
        {

        }
    }
}
```

รูปที่ 16 ตัวอย่างการสร้างคลาสไวยากรณ์ภาษา

ทำการกำหนดชุดของเทอร์มินอล (ไม่สามารถแบ่งต่อไปได้) ดังรูปที่ 17

```
// 1. Terminals
StringLiteral str = new StringLiteral("string", "\"", StringOptions.AllowsDoubledQuote);
StringLiteral strField = new StringLiteral("dataField");
strField.AddStartEnd("<<", ">>", StringOptions.NoEscapes);

NumberLiteral num = new NumberLiteral("num", NumberOptions.IntOnly);
IdentifierTerminal identifier = new IdentifierTerminal("identifier");
```

รูปที่ 17 ตัวอย่างการกำหนดชุดของเทอร์มินอล

ทำการกำหนดคำสำคัญ (Keyword) เพื่อให้การพัฒนาส่วนของไวยากรณ์อ่านง่ายขึ้น ดังรูปที่

```
var visualization = ToTerm("visualization");
var data = ToTerm("data");
var image = ToTerm("image");
var table = ToTerm("table");
var title = ToTerm("title");
var num_row = ToTerm("num-row");
var comma = ToTerm(",");
var select = ToTerm("select");
var barchart = ToTerm("barchart");
var piechart = ToTerm("piechart");
var linechart = ToTerm("linechart");
var series_set = ToTerm("series-set");
var label = ToTerm("label");
var value = ToTerm("value");
var text = ToTerm("text");
var header = ToTerm("header");
var H1 = ToTerm("H1");
var H2 = ToTerm("H2");
var H3 = ToTerm("H3");
var Sum = ToTerm("Sum");
var Count = ToTerm("Count");
var Avg = ToTerm("Avg");
var Min = ToTerm("Min");
var Max = ToTerm("Max");
var equal = ToTerm("=");
```

รูปที่ 18 ตัวอย่างการกำหนดคำสำคัญ

จากนั้นทำการกำหนดชุดของนอนเทอร์มินอล (สามารถแบ่งต่อไปได้) ดังรูปที่ 19

```
//// 2. Non-Terminal
NonTerminal visual = new NonTerminal("visual");
NonTerminal visual_attr = new NonTerminal("visual_attr");
NonTerminal visual_attrs = new NonTerminal("visual_attrs");
NonTerminal datasource = new NonTerminal("datasource");
NonTerminal img = new NonTerminal("image");
NonTerminal tbl = new NonTerminal("table");
NonTerminal otitle = new NonTerminal("title");
NonTerminal table_attr = new NonTerminal("table_attr");
NonTerminal table_attrs = new NonTerminal("table_attrs");
NonTerminal table_paging = new NonTerminal("table_paging");
NonTerminal table_selection = new NonTerminal("table_selection");
NonTerminal table_select_col = new NonTerminal("table_select_col");
NonTerminal table_select_cols = new NonTerminal("table_select_cols");
NonTerminal table_cols = new NonTerminal("table_cols");
NonTerminal chart = new NonTerminal("chart");
NonTerminal chart_attrs = new NonTerminal("chart_attrs");
NonTerminal chart_attr = new NonTerminal("chart_attr");
NonTerminal chart_type = new NonTerminal("chart_type");
NonTerminal chart_series = new NonTerminal("chart_series");
NonTerminal series_attrs = new NonTerminal("series_attrs");
NonTerminal series_attr = new NonTerminal("series_attr");
NonTerminal series_label = new NonTerminal("series_label");
NonTerminal series_value = new NonTerminal("series_value");
NonTerminal text_value = new NonTerminal("text");
NonTerminal _header = new NonTerminal("header");
NonTerminal header_style = new NonTerminal("header_style");
NonTerminal data_field = new NonTerminal("data_field");
NonTerminal data_fields = new NonTerminal("data_fields");
NonTerminal data_field_value = new NonTerminal("data_field_value");
NonTerminal aggregate_field = new NonTerminal("aggregate_field");
NonTerminal data_value = new NonTerminal("data_value");
NonTerminal data_num_value = new NonTerminal("data_num_value");
NonTerminal _label = new NonTerminal("label");
```

รูปที่ 19 ตัวอย่างการกำหนดชุดของนอนเทอร์มินอล

จากนั้นจึงทำการสร้างไวยากรณ์ ดังรูปที่ 20

```
this.Root = visual;
visual.Rule = visualization + "[" + visual_attrs + "]";
visual_attrs.Rule = MakeStarRule(visual_attrs, visual_attr);
visual_attr.Rule = datasource | img | tbl | chart | text_value;
datasource.Rule = data + equal + data_value;
img.Rule = image + equal + data_value;
tbl.Rule = table + "[" + table_attrs + "]";
table_attrs.Rule = MakeStarRule(table_attrs, table_attr);
table_attr.Rule = otitle | datasource | table_selection | table_paging;
otitle.Rule = title + equal + data_value;
table_paging.Rule = num_row + equal + data_num_value;
table_selection.Rule = select + equal + "(" + table_select_cols + ")";
table_select_cols.Rule = MakePlusRule(table_select_cols, comma, table_select_col);
table_select_col.Rule = _label + data_field_value;
_label.Rule = Empty | str;
chart.Rule = chart_type + "[" + chart_attrs + "]";
chart_attrs.Rule = MakeStarRule(chart_attrs, chart_attr);
chart_attr.Rule = otitle | datasource | chart_series;
chart_type.Rule = barchart | piechart | linechart;
chart_series.Rule = series_set + identifier + "[" + series_attrs + "]";
series_attrs.Rule = MakePlusRule(series_attrs, series_attr);
series_attr.Rule = series_label + series_value | datasource;
series_label.Rule = label + equal + data_field_value;
series_value.Rule = value + equal + data_fields;
text_value.Rule = text + equal + data_fields | header + header_style + equal + data_fields;
header_style.Rule = H1 | H2 | H3;
data_fields.Rule = MakeStarRule(data_fields, data_field);
data_field.Rule = aggregate_field | data_field_value | data_value;
aggregate_field.Rule =  Sum + "(" + data_field + ")" |
                        Count + "(" + data_field + ")" |
                        Avg + "(" + data_field + ")" |
                        Min + "(" + data_field + ")" |
                        Max + "(" + data_field + ")";
data_field_value.Rule = strField;
data_value.Rule = str | num;
data_num_value.Rule = num;
```

รูปที่ 20 การสร้างไวยากรณ์ของไอโรนี

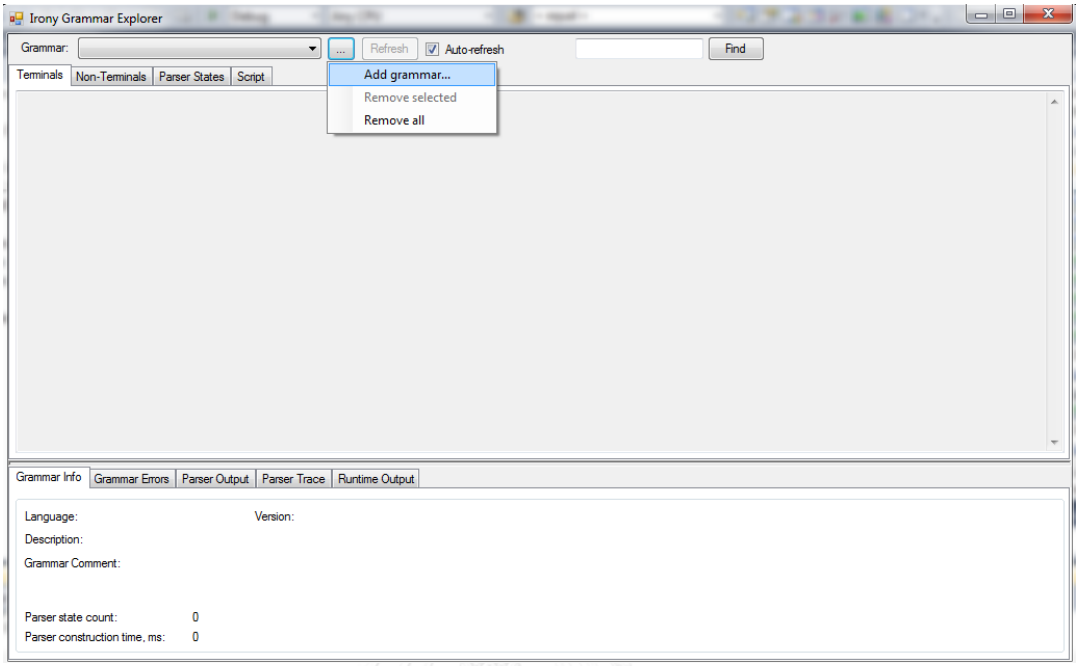
และสุดท้าย ทำการรีจิสเตอร์เครื่องหมาย เพื่อไม่ให้ไอโรนี นำเครื่องหมายเหล่านี้ไปอยู่ในวากยสัมพันธ์แบบต้นไม้ ดังรูปที่ 21

```
MarkPunctuation("[", "]", "(", ")");
RegisterBracePair("[", "]");
RegisterBracePair("(", ")");
```

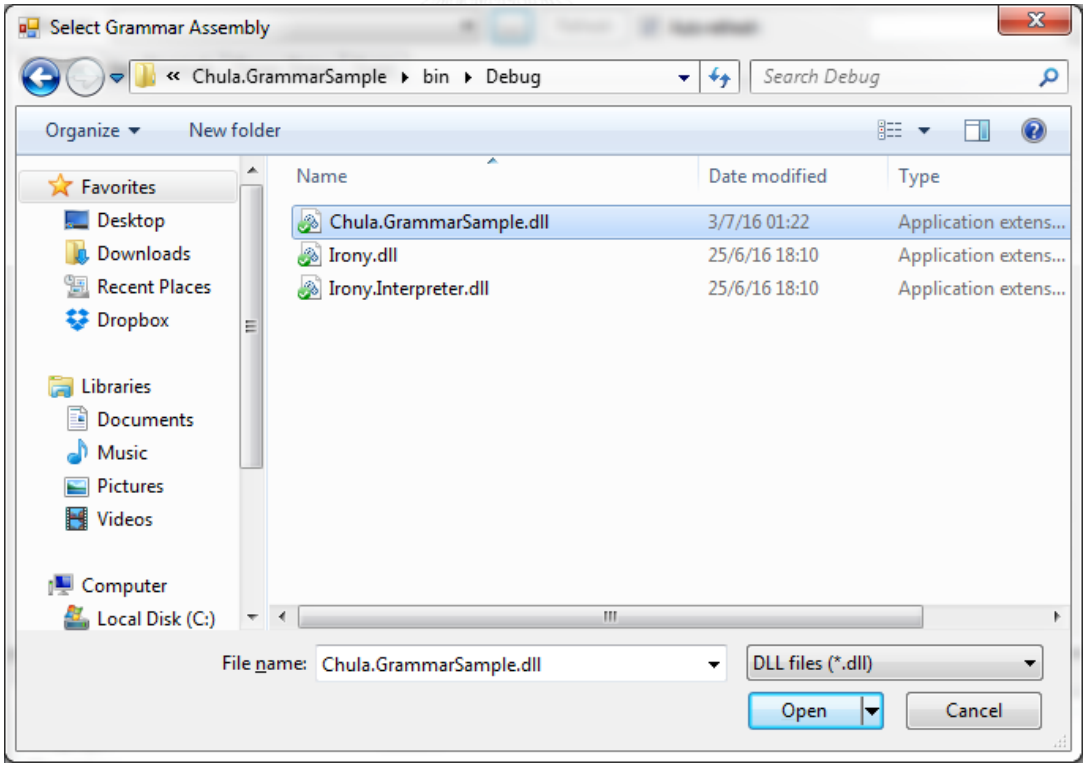
รูปที่ 21 ชุดคำสั่งในการรีจิสเตอร์เครื่องหมาย

เมื่อทำการสร้างไวยากรณ์เรียบร้อยแล้ว สามารถทำการทดสอบไวยากรณ์ได้จากการใช้แกรมมาเอกซ์พลอเรอร์ (Grammar Explorer) ของไอโรนี โดยเริ่มจากการเปิดโปรแกรม ทำการเลือก

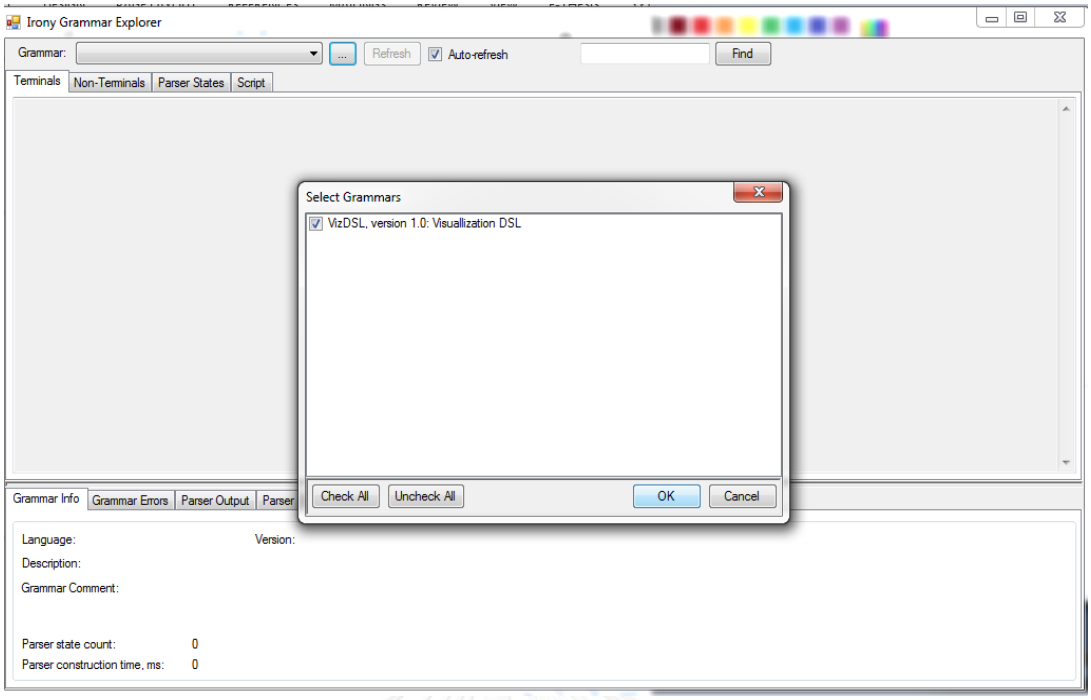
ไลบรารีของไวยากรณ์ที่ได้ทำการคอมไพล์ไว้แล้ว จากนั้นนำสคริปต์ภาษาที่ใช้ทดสอบมาใส่ไว้ในกรอบข้อความ และกดปุ่ม Parse เพื่อทดสอบ ดังรูปที่ 22 รูปที่ 23 รูปที่ 24 และ รูปที่ 25 ตามลำดับ



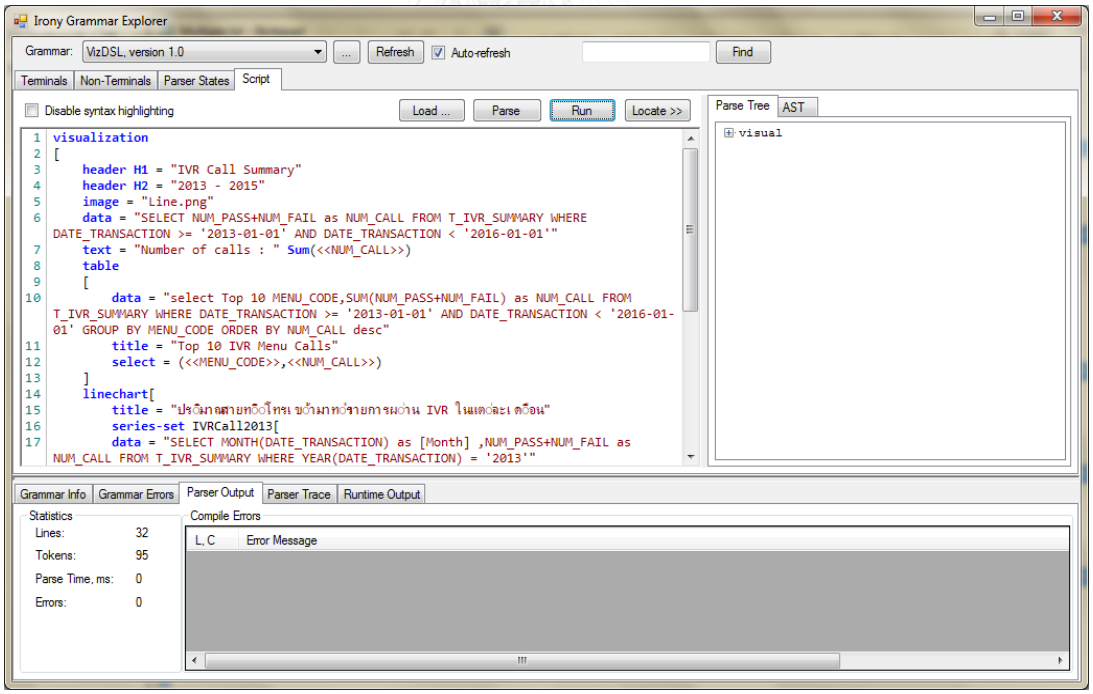
รูปที่ 22 การเพิ่มไวยากรณ์ในแกรมมาเอกซ์พลอเรอร์



รูปที่ 23 หน้าจอเลือกไวยากรณ์ที่สร้างขึ้น



รูปที่ 24 หน้าจอการเลือกไวยากรณ์เพื่อใช้ในแกรมมาเอกซ์พลอเรอร์



รูปที่ 25 หน้าจอแกรมมาเอกซ์พลอเรอร์

4.3.2 การพัฒนาแบบจำลองความหมาย

ในการพัฒนาแบบจำลองความหมายนั้น จะทำการสร้างจากวากยสัมพันธ์แบบต้นไม้ของไอโรนี ให้อยู่ในรูปแบบเอกซ์เอ็มแอลแบบกำหนดเอง โดยจะต้องไปเก็บข้อมูลในแต่ละโนดของต้นไม้ที่ไอโรนีสร้างขึ้น เพื่อนำข้อมูลมาใช้สร้างเอกซ์เอ็มแอลตามที่ได้กำหนดไว้ ดังรูปที่ 26

```
//travel in visual_attrs
XElement visual = new XElement("Visualization");
foreach (var childs in node.Root.ChildNodes[1].ChildNodes)
{
    var node_visual_attr = childs.ChildNodes[0];
    switch (node_visual_attr.Term.Name)
    {
        case "datasource":
            XElement datasource = new XElement("Datasource", node_visual_attr.ChildNodes[2].ChildNodes[0].Token.ValueString);
            visual.Add(datasource);
            continue;
        case "image":
            XElement image = new XElement("Image", node_visual_attr.ChildNodes[2].ChildNodes[0].Token.ValueString);
            visual.Add(image);
            continue;
        case "table":
            XElement table = new XElement("Table");
            var node_table_attrs = node_visual_attr.ChildNodes[1];
            foreach (var table_attr in node_table_attrs.ChildNodes)
            {
                table = WriteTableAttr(table_attr.ChildNodes[0], table);
            }
            visual.Add(table);
            continue;
        case "chart":
            XElement chart;
            var node_chart_attrs = node_visual_attr.ChildNodes[1];
            var node_chart_type = node_visual_attr.ChildNodes[0].ChildNodes[0];
            switch (node_chart_type.Term.Name)
            {
                case "barchart": chart = new XElement("Barchart"); break;
                case "linechart": chart = new XElement("Linechart"); break;
                case "piechart": chart = new XElement("Piechart"); break;
                default: chart = new XElement("Chart"); break;
            }

            foreach (var chart_attr in node_chart_attrs.ChildNodes)
            {
                chart = WriteChartAttr(chart_attr.ChildNodes[0], chart);
            }
            visual.Add(chart);
            continue;
        case "text":
            var node_text_type = node_visual_attr.ChildNodes[0];
            if (node_text_type.Term.Name == "header")
            {
                var header_style = node_visual_attr.ChildNodes[1].ChildNodes[0].Term.Name;
                XElement header = new XElement("Header");
                XAttribute hStyle = new XAttribute("header-style", header_style);
                header.Add(hStyle);

                var node_data_fields = node_visual_attr.ChildNodes[3];
                foreach (var data_field in node_data_fields.ChildNodes)
                {
                    header = WriteDataField(data_field.ChildNodes[0], header);
                }
                visual.Add(header);
            }
            else
            {
                XElement text = new XElement("Text");
                var node_data_fields = node_visual_attr.ChildNodes[2];
                foreach (var data_field in node_data_fields.ChildNodes)
                {
                    text = WriteDataField(data_field.ChildNodes[0], text);
                }
                visual.Add(text);
            }
            continue;
    }
```

รูปที่ 26 ชุดคำสั่งการเก็บข้อมูลเพื่อทำการสร้างเอกซ์เอ็มแอล

และเมื่อได้ข้อมูลที่ต้องการครบถ้วนแล้ว จะนำมาส่งออกเป็นเอกซ์เอ็มแอลไฟล์ ดังรูปที่ 27

```
string configXMLOutputFile = ConfigurationManager.AppSettings["XMLOutputFile"];
visual.Save(configXMLOutputFile);
```

รูปที่ 27 ตัวอย่างการส่งออกเอกซ์เอ็มแอลไฟล์

ในการพัฒนานั้นเอกซ์เอ็มแอลที่ได้จะมีส่วนประกอบ และคุณลักษณะต่าง ๆ ดังตารางที่ 2 และตัวอย่างของเอกซ์เอ็มแอลที่ได้มาดังรูปที่ 28

ตารางที่ 2 ส่วนประกอบและคุณลักษณะของเอกซ์เอ็มแอลแบบจำลองความหมาย

Element	Sub Element	Attribute
Visualization	Datasource, Image, Header, Text, Table, Piechart, Barchart, Linechart	-
Datasource	-	-
Image	-	-
Header	TextValue, Compute	header-style
Text	TextValue, Compute	-
Table	Datasource, Column	title, num-row
Piechart	Datasource, SeriesSet	title
Barchart	Datasource, SeriesSet	title
Linechart	Datasource, SeriesSet	title
TextValue	-	-
Compute	Compute, Field	Function
Field	-	-
Column	Label, Value	-
Label	TextValue, Field	-
Value	Field, Compute	-
SeriesSet	Datasource, Label, Value	series-name

```
<?xml version="1.0" encoding="utf-8"?>
<Visualization>
  <Header header-style="H1">
    <TextValue>IVR Call Summary</TextValue>
  </Header>
  <Header header-style="H2">
    <TextValue>2013 - 2015</TextValue>
  </Header>
  <Image>Line.png</Image>
  <Datasource>SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE
  DATE_TRANSACTION &gt;= '2013-01-01' AND DATE_TRANSACTION &lt; '2016-01-01'</Datasourc
  <Text>
    <TextValue>Number of calls : </TextValue>
    <Compute Function="sum">
      <Field>NUM_CALL</Field>
    </Compute>
  </Text>
  <Table title="Top 10 IVR Menu Calls">
    <Datasource>select Top 10 MENU_CODE,SUM(NUM_PASS+NUM_FAIL) as NUM_CALL FROM
    T_IVR_SUMMARY WHERE DATE_TRANSACTION &gt;= '2013-01-01' AND DATE_TRANSACTION &lt;
    '2016-01-01' GROUP BY MENU_CODE ORDER BY NUM_CALL desc</Datasource>
    <Column>
      <Label>
        <TextValue>MENU_CODE</TextValue>
      </Label>
      <Value>
        <Field>MENU_CODE</Field>
      </Value>
    </Column>
    <Column>
      <Label>
        <TextValue>NUM_CALL</TextValue>
      </Label>
      <Value>
        <Field>NUM_CALL</Field>
      </Value>
    </Column>
  </Table>
  <Linechart title="ปริมาณสายที่โทรเข้ามาทำรายการผ่าน IVR ในแต่ละเดือน">
    <SeriesSet series-name="IVRCall2013">
      <Datasource>SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL as
      NUM_CALL FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2013'</Datasource>
      <Label>
        <Field>Month</Field>
      </Label>
      <Value>
        <Compute Function="sum">
          <Field>NUM_CALL</Field>
        </Compute>
      </Value>
    </SeriesSet>
    <SeriesSet series-name="IVRCall2014">
      <Datasource>SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL as
      NUM_CALL FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2014'</Datasource>
      <Label>
        <Field>Month</Field>
      </Label>
      <Value>
        <Compute Function="sum">
          <Field>NUM_CALL</Field>
        </Compute>
      </Value>
    </SeriesSet>
    <SeriesSet series-name="IVRCall2015">
      <Datasource>SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL as
      NUM_CALL FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2015'</Datasource>
      <Label>
        <Field>Month</Field>
      </Label>
      <Value>
        <Compute Function="sum">
          <Field>NUM_CALL</Field>
        </Compute>
      </Value>
    </SeriesSet>
  </Linechart>
</Visualization>
```

รูปที่ 28 ตัวอย่างเอกซ์เอ็มแอลไฟล์แสดงแบบจำลองความหมาย

สำหรับการเรียกใช้งานในส่วนนี้ จะทำการดัดแปลงแกรมมาเอกซ์พลอเรอร์ ให้ปุ่ม Run ใช้ในการ สร้างแบบจำลองความหมาย โดยเรียกใช้ส่วนของการสร้างแบบจำลองความหมาย ดังรูปที่ 29

```
private void RunSample() {
    //Custom
    XmlDocument xml = new XmlDocument();
    ParseTree result = GetParseTree();
    ParseSemantic semantic = new ParseSemantic();
    semantic.WriteSemantic(result);
}
```

รูปที่ 29 ชุดคำสั่งการเรียกใช้ส่วนการสร้างแบบจำลองความหมาย

4.3.3 การพัฒนาส่วนประกอบแสดงผลภาพ

การพัฒนาส่วนประกอบแสดงผลภาพ ในงานวิจัยนี้ได้เลือกใช้เทคโนโลยีเว็บแอปพลิเคชันเอเอสพีดอทเน็ต 4.0 โดยเริ่มจากการอ่านไฟล์เอกซ์เอ็มแอลแบบจำลองความหมาย ดังรูปที่ 30

```
XmlDocument doc = new XmlDocument();
string configXMLOutputFile = ConfigurationManager.AppSettings["XMLOutputFile"];
doc.Load(configXMLOutputFile);
```

รูปที่ 30 ตัวอย่างการอ่านไฟล์เอกซ์เอ็มแอล

เมื่อทำการอ่านไฟล์แล้ว จะทำการนำส่วนประกอบ และคุณสมบัติ ของแบบจำลองความหมาย ที่อยู่ในรูปแบบเอกซ์เอ็มแอล มาสร้างส่วนประกอบแสดงผลภาพโดยภาพรวมของฟังก์ชัน ดังรูปที่ 31

```
private void Render(XmlNode componentNode)
{
    switch (componentNode.Name)
    {
        case "Image":
            PlaceHolder1.Controls.Add(GetImage(componentNode.InnerText));
            break;
        case "Header":
            var datafield = GetDatafield(componentNode);
            PlaceHolder1.Controls.Add(GetText(datafield, componentNode.Attributes[0].Value));
            break;
        case "Text":
            datafield = GetDatafield(componentNode);
            PlaceHolder1.Controls.Add(GetText(datafield));
            break;
        case "Datasource":
            string query = componentNode.InnerText.Replace("&gt;", ">").Replace("&lt;", "<");
            ViewState.Add("TableData", DataAdaptor.GetData(query));
            break;
        case "Table":
            string numrow = "";
            foreach (XmlAttribute tableAttribute in componentNode.Attributes)
            {
                if (tableAttribute.Name == "title")
                {
                    HtmlGenericControl control = new HtmlGenericControl();
                    control.TagName = "H3";
                    control.InnerText = tableAttribute.Value;
                    PlaceHolder1.Controls.Add(control);
                }
                else if (tableAttribute.Name == "num-row")
                {
                    numrow = tableAttribute.Value;
                }
            }
            PlaceHolder1.Controls.Add(GetTable(componentNode,numrow));
            break;
        case "Barchart":
            PlaceHolder1.Controls.Add(GetChart("Barchart", componentNode));
            break;
        case "Linechart":
            PlaceHolder1.Controls.Add(GetChart("Linechart", componentNode));
            break;
        case "Piechart":
            PlaceHolder1.Controls.Add(GetChart("Piechart", componentNode));
            break;

    }
}
```

รูปที่ 31 ฟังก์ชันการใช้งานแบบจำลองความหมายเพื่อสร้างส่วนประกอบแสดงผลภาพ

เมื่อพบส่วนประกอบของรูปภาพจากแบบจำลองความหมาย สามารถสกัดเอาข้อมูล เป็นคุณสมบัติของรูปภาพ ได้ดังรูปที่ 32

```
private Image GetImage(string imgLink)
{
    Image img = new Image();
    img.ImageUrl = "Images/" + imgLink;
    return img;
}
```

รูปที่ 32 ฟังก์ชันการกำหนดคุณสมบัติรูปภาพ

เมื่อพบส่วนประกอบของหัวเรื่อง และข้อความ สามารถสกัดเอาข้อมูล เป็นคุณสมบัติของข้อความได้ดังรูปที่ 33 และหากเป็นข้อความคำนวณจะสามารถสกัดเอาข้อมูลได้ดังรูปที่ 34

```
private HtmlGenericControl GetText(string text, string headerStyle = "")
{
    HtmlGenericControl control = new HtmlGenericControl();
    if (headerStyle == "")
        control.TagName = "div";
    else
        control.TagName = headerStyle;
    control.InnerText = text;
    return control;
}
```

รูปที่ 33 ฟังก์ชันการกำหนดคุณสมบัติข้อความ

```
private string GetDatafield(XmlNode componentNode)
{
    string data = "";
    foreach (XmlNode nodeDatafield in componentNode)
    {
        switch (nodeDatafield.Name)
        {
            case "TextValue":
                data += nodeDatafield.ChildNodes[0].Value;
                break;
            case "Compute":
                if (ViewState["TableData"] != null)
                {
                    string func = nodeDatafield.Attributes[0].Value;
                    string field = nodeDatafield.ChildNodes[0].ChildNodes[0].Value;
                    DataTable dt = (DataTable)ViewState["TableData"];
                    switch (func.ToLower())
                    {
                        case "count":
                            data += dt.Compute("Count(" + field + ")", "");
                            break;
                        case "sum":
                            data += dt.Compute("Sum(" + field + ")", "");
                            break;
                        case "avg":
                            data += dt.Compute("Avg(" + field + ")", "");
                            break;
                        case "min":
                            data += dt.Compute("Min(" + field + ")", "");
                            break;
                        case "max":
                            data += dt.Compute("Max(" + field + ")", "");
                            break;
                    }
                }

                break;
        }
    }
    return data;
}
```

รูปที่ 34 ฟังก์ชันการสกัดข้อมูลจากข้อความคำนวณ

เมื่อพบส่วนประกอบของตาราง สามารถสกัดข้อมูล เป็นคุณสมบัติ และข้อมูลแสดงผลของตารางได้ดังรูปที่ 35

```
private GridView GetTable(XmlNode tableNode,string numrow = "")
{
    GridView gridview = new GridView();
    List<string> labels = new List<string> { };
    List<string> values = new List<string> { };
    foreach (XmlNode node in tableNode)
    {
        if (node.Name == "Datasource")
        {
            string query = node.InnerText.Replace("&gt;", ">").Replace("&lt;", "<");
            ViewState.Add("TableData", DataAdaptor.GetData(query));
        }
        else if (node.Name == "Column")
        {
            foreach (XmlNode columnType in node.ChildNodes)
            {
                if (columnType.Name == "Label")
                    labels.Add(columnType.ChildNodes[0].ChildNodes[0].Value);
                else if (columnType.Name == "Value")
                    values.Add(columnType.ChildNodes[0].ChildNodes[0].Value);
            }
        }
    }
    DataTable dt = (DataTable)ViewState["TableData"];

    for (int i = 0; i < labels.Count; i++)
    {
        BoundField bf = new BoundField();
        bf.DataField = values[i];
        bf.HeaderText = labels[i];
        gridview.Columns.Add(bf);
    }

    if (numrow != "")
    {
        dt= dt.AsEnumerable().Take(int.Parse(numrow)).CopyToDataTable();
    }
    gridview.AutoGenerateColumns = false;
    gridview.DataSource = dt;
    gridview.DataBind();

    return gridview;
}
```

รูปที่ 35 ฟังก์ชันการสกัดข้อมูลชนิดตาราง

เมื่อพบส่วนประกอบแผนภูมิ สามารถทำการสกัดข้อมูล ได้ โดยเริ่มจากประกาศตัวแปรสำหรับสร้างแผนภูมิ และทำการสกัดคุณลักษณะ ดังรูปที่ 36 จากนั้นทำการสกัดส่วนประกอบในแผนภูมิ ดังรูปที่ 37 รูปที่ 38 และ รูปที่ 39

```
DataTable dt = new DataTable();
Chart chart = new Chart();
Title title = new Title();
string label = "";
string value = "";
string func = "";
foreach (XmlAttribute attribute in chartNode.Attributes)
{
    if (attribute.Name == "title")
    {
        title.Font = new System.Drawing.Font(System.Drawing.FontFamily.GenericSansSerif, (float)16, System.Drawing.FontStyle.Bold);
        title.Name = attribute.Value.Replace(" ", "");
        title.Text = attribute.Value;
        chart.Titles.Add(title);
    }
}
```

รูปที่ 36 ส่วนการประกาศตัวแปรและ การสกัดข้อมูลคุณลักษณะของแผนภูมิ

```
    if (node.Name == "Datasource"){
        string query = node.InnerText.Replace("&gt;", ">").Replace("&lt;", "<");
        ViewState.Add("TableData", DataAdaptor.GetData(query));
        //dt = (DataTable)ViewState["TableData"];
    }
    else if (node.Name == "SeriesSet"){
        chart.Series.Add(GetSeries(node, chartType,ref label,ref value,ref func));
        // Add point to Chart
        dt = (DataTable)ViewState["TableData"];
        // check yData is Compute Type
        if (func != "")  {
            foreach (DataRow row in dt.Rows) {
                string xDat = row[label].ToString();
                string yDat = "";
                switch (func.ToLower()){
                    case "count":
                        yDat = dt.Compute("Count(" + value + ")", label + "= '" + xDat + "'").ToString();
                        break;
                    case "sum":
                        yDat = dt.Compute("Sum(" + value + ")", label + "= '" + xDat + "'").ToString();
                        break;
                    case "avg":
                        yDat = dt.Compute("Avg(" + value + ")", label + "= '" + xDat + "'").ToString();
                        break;
                    case "min":
                        yDat = dt.Compute("Min(" + value + ")", label + "= '" + xDat + "'").ToString();
                        break;
                    case "max":
                        yDat = dt.Compute("Max(" + value + ")", label + "= '" + xDat + "'").ToString();
                        break;
                }
                if (!dictionary.Keys.Contains(xDat)) {
                    dictionary.Add( xDat, yDat );
                }
            }
            foreach (var pair in  dictionary){
                chart.Series[node.Attributes[0].Value].Points.AddXY(pair.Key, pair.Value);
            }
        }
        else{
            foreach (DataRow row in dt.Rows){
                chart.Series[node.Attributes[0].Value].Points.AddXY(row[label].ToString(), row[value].ToString());
            }
        }
        chart.Series[node.Attributes[0].Value].SmartLabelStyle.Enabled = true;
    }
}
```

รูปที่ 37 การสกัดส่วนประกอบในแผนภูมิ

```
private Series GetSeries(XmlNode node,string chartType,ref string label,ref string value, ref string func )
{
    foreach (XmlNode seriesNode in node.ChildNodes)
    {
        if (seriesNode.Name == "Datasource")
        {
            string query = seriesNode.InnerText.Replace("&gt;", ">").Replace("&lt;", "<");
            ViewState.Add("TableData", DataAdaptor.GetData(query));
            // dt = (DataTable)ViewState["TableData"];
        }
        else if (seriesNode.Name == "Label")
            label = seriesNode.ChildNodes[0].ChildNodes[0].Value;
        else if (seriesNode.Name == "Value")
        {
            if (seriesNode.ChildNodes[0].Name == "Field")
                value = seriesNode.ChildNodes[0].ChildNodes[0].Value;
            else if (seriesNode.ChildNodes[0].Name == "Compute")
            {
                func = seriesNode.ChildNodes[0].Attributes[0].Value;
                value = seriesNode.ChildNodes[0].ChildNodes[0].ChildNodes[0].Value;
            }
        }
    }
    // Define Data Series
    Series series = new Series(node.Attributes[0].Value);
    series.IsValueShownAsLabel = true;
    switch (chartType)
    {
        case "Barchart":
            series.ChartType = SeriesChartType.Column; break;
        case "Piechart":
            series.ChartType = SeriesChartType.Pie; break;
        case "Linechart":
            series.ChartType = SeriesChartType.Line; break;
    }
    return series;
}
```

รูปที่ 38 การสกัดข้อมูลของซีรี่ย์

```
//// Define the chart area
ChartArea chartArea = new ChartArea();
chartArea.AxisX.Title = label;
chartArea.AxisY.Title = value;
chart.ChartAreas.Add(chartArea);
chart.ChartAreas[0].AxisX.MajorGrid.LineColor = System.Drawing.Color.LightGray;
chart.ChartAreas[0].AxisY.MajorGrid.LineColor = System.Drawing.Color.LightGray;
Legend legend = new Legend();
chart.Legends.Add(legend);
chart.Width = new System.Web.UI.WebControls.Unit(500, System.Web.UI.WebControls.UnitType.Pixel);
chart.Height = new System.Web.UI.WebControls.Unit(350, System.Web.UI.WebControls.UnitType.Pixel);
chart.DataBind();
return chart;
```

รูปที่ 39 การเพิ่มคุณลักษณะอื่นให้แผนภูมิ

เมื่อทำการเขียนสคริปต์ภาษา และทำการรันสคริปต์แล้ว ตัวอย่างผลลัพธ์จากการแสดงผลส่วนประกอบแสดงผลภาพแสดงได้ ดังรูปที่ 40

IVR Call Summary

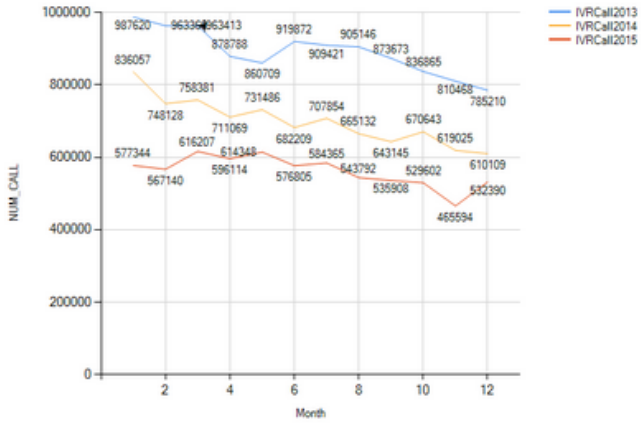
2013 - 2015

Number of calls : 25817392

Top 10 IVR Menu Calls

MENU_CODE	NUM_CALL
1.1.1	18844750
1.1.2.1	1150669
1	1026954
1.3.1	910488
1.1.2.2	831073
**11*	812998
1.1.2	643403
1.1	478252
1.3	338787
1.1.3.3	278883

ปริมาณสายที่โทรเข้ามาทำรายการผ่าน IVR ในแต่ละเดือน



รูปที่ 40 ตัวอย่างส่วนประกอบแสดงผลภาพ

บทที่ 5
การทดสอบและการวิเคราะห์ผล

5.1 วัตถุประสงค์ของการทดสอบ

เพื่อสนับสนุนแนวทางการสร้างภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ ที่ได้พัฒนาในบทที่ 4 โดยการทดสอบเริ่มต้นจากการเขียนสคริปต์ภาษา ทำการส่งออกเอกซ์เอ็มแอลไฟล์ และสุดท้ายผลทดสอบสำเร็จคือ การได้ส่วนประกอบแสดงผลภาพที่มีข้อมูลถูกต้องตามสคริปต์ภาษา

5.2 การทดสอบระบบ

วิธีการในการใช้ทดสอบระบบ จะใช้การทดสอบแบบแบล็กบ็อก (Black Box Testing) ซึ่งจะเป็นการทดสอบที่ใช้การดูผลลัพธ์ที่ได้ออกมา ว่าถูกต้องตามผลลัพธ์ที่คาดหวังหรือไม่ โดยจะทำการทดสอบโดยใช้สคริปต์ภาษาแบ่งออกเป็นแต่ละประเภทตามส่วนประกอบแสดงผลภาพที่ได้กำหนดไว้ในบทที่ 3

5.2.1 ทดสอบการแสดงข้อความ

การทดสอบจะทำการเขียนสคริปต์ภาษา สำหรับการแสดงผลข้อความ ทั้งในแบบ ข้อความหัวเรื่องหลัก หัวเรื่องรอง ข้อความปกติ โดยสคริปต์ทดสอบเป็นดังรูปที่ 41 รูปที่ 42 รูปที่ 43 ตามลำดับ รวมถึงข้อความคำนวณที่มีฟังก์ชัน Count, Sum, Avg, Min, Max ดังรูปที่ 44 รูปที่ 45 รูปที่ 46 รูปที่ 47 และ รูปที่ 48 ตามลำดับ

```
visualization
[
    header H1 = "IVR Call Summary"
]
```

รูปที่ 41 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลหัวเรื่องหลัก

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
]
```

รูปที่ 42 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลหัวเรื่องรอง

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    text = "Number of call :"
]
```

รูปที่ 43 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลข้อความปกติ

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    data = "SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE
DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01' AND
MENU_CODE = '1.3.5' AND NUM_PASS+NUM_FAIL = 0"
    text = "Days of no call in menu 1.3.5 : " Count(<<NUM_CALL>>)
]
```

รูปที่ 44 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลข้อความการคำนวณฟังก์ชัน Count

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    data = "SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE
DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01'"
    text = "Number of calls : " Sum(<<NUM_CALL>>)
]
```

รูปที่ 45 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลข้อความการคำนวณฟังก์ชัน Sum

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    data = "SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE
DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01'"
    text = "Average day calls : " Avg(<<NUM_CALL>>)
]
```

รูปที่ 46 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลข้อความการคำนวณฟังก์ชัน Avg

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    data = "SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE
DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01'"
    text = "Minimum no. of calls : " Min(<<NUM_CALL>>)
]
```

รูปที่ 47 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลข้อความการคำนวณฟังก์ชัน Min

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    data = "SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE
DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01'"
    text = "Maximum no. of calls : " Max(<<NUM_CALL>>)
]
```

รูปที่ 48 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลข้อความการคำนวณฟังก์ชัน Max

5.2.2 ทดสอบการแสดงรูปภาพ

การทดสอบจะทำการเขียนสคริปต์ภาษา สำหรับการแสดงผลรูปภาพ โดยข้อมูลรูปภาพจะมีเตรียมไว้อยู่ในพาธที่กำหนด โดยสคริปต์ทดสอบเป็นดังรูปที่ 49

```
visualization
[
    image = "data_visualisation.jpg"
]
```

รูปที่ 49 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลรูปภาพ

5.2.3 ทดสอบการแสดงผลตาราง

การทดสอบจะทำการเขียนสคริปต์ภาษา สำหรับการแสดงผลตาราง โดยแสดงผลแบบมีการกำหนดหัวชื่อหัวตาราง และแบบไม่กำหนดชื่อหัวตาราง และ กำหนดจำนวนแถวในตาราง โดยข้อมูลจะถูกนำเข้าจากฟิลด์ข้อมูลในฐานข้อมูล โดยสคริปต์ทดสอบเป็นดังรูปที่ 50 รูปที่ 51 และ รูปที่ 52

```
visualization
[
    data = "select Top 10 MENU_CODE,SUM(NUM_PASS+NUM_FAIL) as NUM_CALL
FROM T_IVR_SUMMARY WHERE DATE_TRANSACTION >= '2013-01-01' AND
DATE_TRANSACTION < '2016-01-01' GROUP BY MENU_CODE ORDER BY NUM_CALL desc
"
    table
    [
        title = "Top 10 IVR Menu Calls"
        select = (<<MENU_CODE>>,<<NUM_CALL>>)
    ]
]
```

รูปที่ 50 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลตารางแบบไม่กำหนดชื่อหัวตาราง

```
visualization
[
    data = "select Top 10 MENU_CODE,SUM(NUM_PASS+NUM_FAIL) as NUM_CALL
FROM T_IVR_SUMMARY WHERE DATE_TRANSACTION >= '2013-01-01' AND
DATE_TRANSACTION < '2016-01-01' GROUP BY MENU_CODE ORDER BY NUM_CALL desc
"
    table
    [
        title = "Top 10 IVR Menu Calls"
        select = ("Menu Code"<<MENU_CODE>>,"No. of Calls"<<NUM_CALL>>)
    ]
]
```

รูปที่ 51 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลตารางแบบกำหนดชื่อหัวตาราง

```
visualization
[
    data = "select MENU_CODE,SUM(NUM_PASS+NUM_FAIL) as NUM_CALL FROM T_IVR_SUMMARY WHERE
DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01' GROUP BY MENU_CODE ORDER
BY NUM_CALL desc"
    table
    [
        num-row = 10
        title = "Top 10 IVR Menu Calls"
        select = ("Menu Code"<<MENU_CODE>>,"No. of Calls"<<NUM_CALL>>)
    ]
]
```

รูปที่ 52 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลตารางแบบกำหนดจำนวนแถว

5.2.4 ทดสอบการแสดงผลแผนภูมิ

การทดสอบจะทำการเขียนสคริปต์ภาษา สำหรับการแสดงผลแผนภูมิ โดยแบ่งเป็น แผนภูมิแท่ง แผนภูมิเส้น และแผนภูมิวงกลม ตัวอย่างสคริปต์ที่ใช้ทดสอบดังรูปที่ 53 รูปที่ 54 และ รูปที่ 55 ตามลำดับ

```
visualization
[
    barchart[
        data = "SELECT YEAR(DATE_TRANSACTION) as [Year] ,NUM_PASS,NUM_FAIL
FROM T_IVR_SUMMARY WHERE DATE_TRANSACTION >= '2013-01-01' AND
DATE_TRANSACTION < '2016-01-01'"
        title = "ปริมาณสายที่โทรเข้ามาทำรายการผ่าน IVR ปี 2013-2015"
        series-set CallPass[
        label = <<Year>>
            value = Sum(<<NUM_PASS>>)
        ]
        series-set CallFail[
        label = <<Year>>
        value = Sum(<<NUM_FAIL>>)
        ]
    ]
]
```

รูปที่ 53 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลแผนภูมิแท่ง

```
visualization
[
    linechart[
    title = "ปริมาณสายที่โทรเข้ามาทำรายการผ่าน IVR ในแต่ละเดือน"
        series-set IVRCall2013[
        data = "SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL
as NUM_CALL FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2013'"
            label = <<Month>>
            value = Sum(<<NUM_CALL>>)
        ]
        series-set IVRCall2014[
        data = "SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL
as NUM_CALL FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2014'"
            label = <<Month>>
            value = Sum(<<NUM_CALL>>)
        ]
        series-set IVRCall2015[
        data = "SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL
as NUM_CALL FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2015'"
            label = <<Month>>
            value = Sum(<<NUM_CALL>>)
        ]
    ]
]
```

รูปที่ 54 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลแผนภูมิเส้น

```
visualization
[
    piechart[
        data = "SELECT T2.SEGMENT_DESC as Segment ,NUM_CALL FROM
T_IVR_SUMMARY_SEGMENT T1 JOIN T_IVR_SEGMENT T2 ON T1.SEGMENT = T2.SEGMENT_ID
WHERE T1.DATE_TRANSACTION >= '2015-08-01' AND T1.DATE_TRANSACTION < '2015-09-
01'"
        title = "ปริมาณสายที่โทรเข้ามาทำรายการผ่าน IVR เดือน 8/2015 แบ่งตาม
Segment"
        series-set Segment[
            label = <<Segment>>
            value = Sum(<<NUM_CALL>>)
        ]
    ]
]
```

รูปที่ 55 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลแผนภูมิวงกลม

5.2.5 ทดสอบการแสดงผลแบบหลายส่วนประกอบแสดงผลภาพ

การทดสอบจะทำการเขียนสคริปต์ภาษา สำหรับการแสดงผลส่วนประกอบแสดงผลภาพหลายส่วนประกอบ เช่น ข้อความ รูปภาพ แผนภูมิ ตาราง โดยตัวอย่างสคริปต์ที่ใช้ทดสอบดังรูปที่ 56

```
visualization
[
    header H1 = "IVR Call Summary"
    header H2 = "2013 - 2015"
    image = "Line.png"
    data = "SELECT NUM_PASS+NUM_FAIL as NUM_CALL FROM T_IVR_SUMMARY WHERE DATE_TRANSACTION >=
 '2013-01-01' AND DATE_TRANSACTION < '2016-01-01'"
    text = "Number of calls : " Sum(<<NUM_CALL>>)
    table
    [
        data = "select Top 10 MENU_CODE,SUM(NUM_PASS+NUM_FAIL) as NUM_CALL FROM
T_IVR_SUMMARY WHERE DATE_TRANSACTION >= '2013-01-01' AND DATE_TRANSACTION < '2016-01-01'
GROUP BY MENU_CODE ORDER BY NUM_CALL desc"
        title = "Top 10 IVR Menu Calls"
        select = (<<MENU_CODE>>,<<NUM_CALL>>)
    ]
    linechart[
        title = "ปริมาณสายที่โทรเข้ามาทำรายการผ่าน IVR ในแต่ละเดือน"
        series-set IVRCall2013[
        data = "SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL as NUM_CALL
FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2013'"
            label = <<Month>>
            value = Sum(<<NUM_CALL>>)
        ]
        series-set IVRCall2014[
        data = "SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL as NUM_CALL
FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2014'"
            label = <<Month>>
            value = Sum(<<NUM_CALL>>)
        ]
        series-set IVRCall2015[
        data = "SELECT MONTH(DATE_TRANSACTION) as [Month] ,NUM_PASS+NUM_FAIL as NUM_CALL
FROM T_IVR_SUMMARY WHERE YEAR(DATE_TRANSACTION) = '2015'"
            label = <<Month>>
            value = Sum(<<NUM_CALL>>)
        ]
    ]
]
```

รูปที่ 56 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการแสดงผลหลายส่วนประกอบแสดงผลภาพ

5.2.6 ทดสอบการไม่แสดงผลส่วนประกอบแสดงผลภาพ

การทดสอบจะทำการเขียนสคริปต์ภาษา สำหรับการไม่แสดงผลส่วนประกอบแสดงผลภาพ โดยตัวอย่างสคริปต์ภาษาที่ใช้ทดสอบดัง รูปที่ 57

```
visualization[

]
```

รูปที่ 57 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบการไม่แสดงผลส่วนประกอบแสดงผลภาพ

5.2.7 ทดสอบการเขียนสคริปต์ไม่ถูกต้อง

การทดสอบจะทำการเขียนสคริปต์ภาษาที่ไม่ถูกต้องตามที่ออกแบบไว้ ดังรูปที่ 58

```
visualization[
    data =
]
```

รูปที่ 58 ตัวอย่างสคริปต์ภาษาสำหรับทดสอบสคริปต์ไม่ถูกต้อง

วัตถุประสงค์ของการทดสอบเพื่อยืนยันความถูกต้องของการทำงานของภาษา และเครื่องมือที่สร้างขึ้น โดยการทดสอบจะกระทำตามรายการ ดังตารางที่ 3

ตารางที่ 3 การทดสอบการทำงานของภาษา และเครื่องมือ

รหัส	การทดสอบ	ผลที่คาดหวัง	ผลลัพธ์
TC01	การแสดงข้อความปกติ	สามารถแสดงข้อความปกติได้ถูกต้อง	ถูกต้อง
TC02	การแสดงข้อความหัวเรื่องหลัก	สามารถแสดงข้อความ และรูปแบบหัวเรื่องหลักได้ถูกต้อง	ถูกต้อง
TC03	การแสดงข้อความหัวเรื่องรอง	สามารถแสดงข้อความ และรูปแบบหัวเรื่องรองได้ถูกต้อง	ถูกต้อง
TC04	การแสดงข้อความคำนวณฟังก์ชัน Sum	สามารถแสดงผลข้อความคำนวณฟังก์ชัน Sum ได้ถูกต้อง	ถูกต้อง
TC05	การแสดงข้อความคำนวณฟังก์ชัน Count	สามารถแสดงผลข้อความคำนวณฟังก์ชัน Count ได้ถูกต้อง	ถูกต้อง
TC06	การแสดงข้อความคำนวณฟังก์ชัน Avg	สามารถแสดงผลข้อความคำนวณฟังก์ชัน Avg ได้ถูกต้อง	ถูกต้อง
TC07	การแสดงข้อความคำนวณฟังก์ชัน Min	สามารถแสดงผลข้อความคำนวณฟังก์ชัน Min ได้ถูกต้อง	ถูกต้อง

TC08	การแสดงข้อความคำนวณฟังก์ชัน Max	สามารถแสดงผลข้อความคำนวณฟังก์ชัน Max ได้ถูกต้อง	ถูกต้อง
TC09	การแสดงรูปภาพ	สามารถแสดงผลรูปภาพ ได้ถูกต้อง	ถูกต้อง
TC10	การแสดงผลตารางแบบกำหนดชื่อหัวตาราง	สามารถแสดงผลตาราง ด้วยชื่อหัวข้อตารางที่กำหนดเอง และข้อมูลที่แสดงผลออกมาได้ถูกต้อง	ถูกต้อง
TC11	การแสดงผลตารางแบบไม่กำหนดชื่อหัวตาราง	สามารถแสดงผลตาราง ด้วยชื่อหัวข้อตารางอัตโนมัติ (จากชื่อฟิลด์ในฐานข้อมูล) และข้อมูลที่แสดงผลออกมาได้ถูกต้อง	ถูกต้อง
TC12	การแสดงผลตารางแบบจำกัดจำนวนแถวการแสดงผล	สามารถแสดงผลตารางแบบจำกัดแถวการแสดงผลได้ถูกต้อง	ถูกต้อง
TC13	การแสดงผลแผนภูมิแท่งแบบ 1 Series	สามารถแสดงผลแผนภูมิแท่งแบบ 1 Series ได้ถูกต้อง	ถูกต้อง
TC14	การแสดงผลแผนภูมิแท่งแบบหลาย Series	สามารถแสดงผลแผนภูมิแท่งแบบหลาย Series ได้ถูกต้อง	ถูกต้อง
TC15	การแสดงผลแผนภูมิเส้นแบบ 1 Series	สามารถแสดงผลแผนภูมิเส้นแบบ 1 Series ได้ถูกต้อง	ถูกต้อง
TC16	การแสดงผลแผนภูมิเส้นแบบหลาย Series	สามารถแสดงผลแผนภูมิเส้นแบบหลาย Series ได้ถูกต้อง	ถูกต้อง
TC17	การแสดงผลแผนภูมิวงกลม	สามารถแสดงผลแผนภูมิวงกลมได้ถูกต้อง	ถูกต้อง
TC18	การแสดงผลแบบหลายส่วนประกอบ	สามารถแสดงผลแบบหลายส่วนประกอบได้ถูกต้องตามลำดับ	ถูกต้อง

TC19	การแสดงผลแบบไม่มีส่วนประกอบ	เป็นหน้าว่าง	ถูกต้อง
TC20	การเขียนสคริปต์ผิดพลาด	มีการแสดงผลความผิดพลาดบนหน้าจอแกรมมาเอกซ์พลอเรอร์	ถูกต้อง

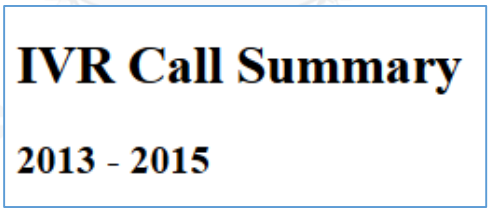
5.3 สรุปผลการทดสอบ

การทดสอบตามข้อ 5.2 ผู้วิจัยขอสรุปผลการทดสอบ ดังต่อไปนี้

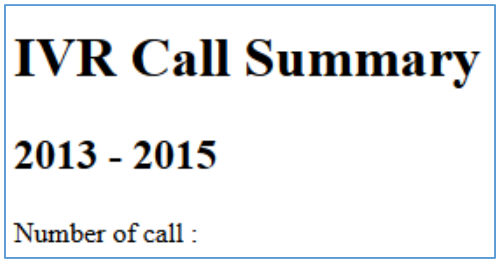
1) การทดสอบการแสดงข้อความ

สามารถแสดงผลข้อความปกติ ข้อความหัวเรื่อง ข้อความหัวเรื่องรอง และ ข้อความคำนวณได้ถูกต้องดัง รูปที่ 59 รูปที่ 60 รูปที่ 61 รูปที่ 62 รูปที่ 63 รูปที่ 64 รูปที่ 65 และรูปที่ 66

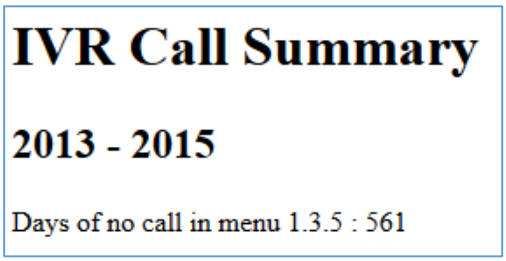
รูปที่ 59 ผลลัพธ์ตัวอย่างสคริปต์แสดงหัวเรื่อง



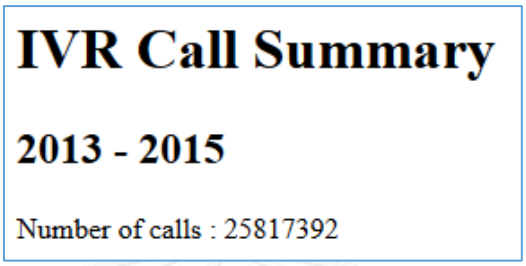
รูปที่ 60 ผลลัพธ์ตัวอย่างสคริปต์แสดงหัวเรื่องรอง



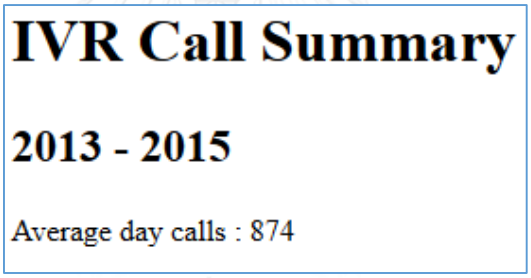
รูปที่ 61 ผลลัพธ์ตัวอย่างสคริปต์แสดงข้อความปกติ



รูปที่ 62 ผลลัพธ์ตัวอย่างสคริปต์แสดงข้อความคำนวณ Count



รูปที่ 63 ผลลัพธ์ตัวอย่างสคริปต์แสดงข้อความคำนวณ Sum



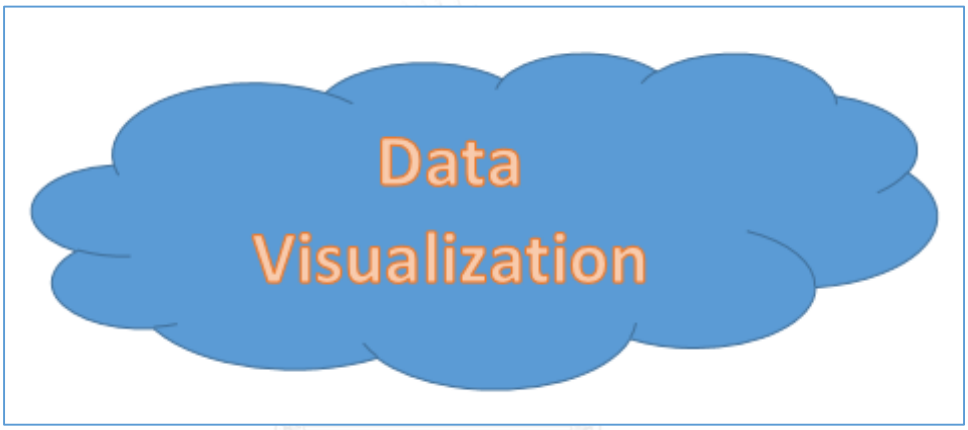
รูปที่ 64 ผลลัพธ์ตัวอย่างสคริปต์แสดงข้อความคำนวณ Avg

รูปที่ 65 ผลลัพธ์ตัวอย่างสคริปต์แสดงข้อความคำนวณ Min

รูปที่ 66 ผลลัพธ์ตัวอย่างสคริปต์แสดงข้อความคำนวณ Max

2) การทดสอบการแสดงรูปภาพ

สามารถแสดงผลรูปภาพได้อย่างถูกต้องตามที่กำหนดในสคริปต์ภาษาดังรูปที่ 67



รูปที่ 67 ผลลัพธ์ตัวอย่างสคริปต์แสดงรูปภาพ

3) การทดสอบการแสดงผลตาราง

สามารถแสดงผลข้อมูลตาราง หัวตารางได้ถูกต้อง ตามการดึงข้อมูลจากฐานข้อมูล รวมถึงสามารถแสดงผลการจำกัดข้อมูลแถวตารางได้อย่างถูกต้องดัง รูปที่ 68 รูปที่ 69 และรูปที่ 70

Top 10 IVR Menu Calls

MENU_CODE	NUM_CALL
1.1.1	18844750
1.1.2.1	1150669
1	1026954
1.3.1	910488
1.1.2.2	831073
**11*	812998
1.1.2	643403
1.1	478252
1.3	338787
1.1.3.3	278883

รูปที่ 68 ผลลัพธ์ตัวอย่างสคริปต์แสดงตารางแบบไม่กำหนดหัวตาราง

Top 10 IVR Menu Calls

Menu Code	No. of Calls
1.1.1	18844750
1.1.2.1	1150669
1	1026954
1.3.1	910488
1.1.2.2	831073
**11*	812998
1.1.2	643403
1.1	478252
1.3	338787
1.1.3.3	278883

รูปที่ 69 ผลลัพธ์ตัวอย่างสคริปต์แสดงตารางแบบกำหนดหัวตาราง

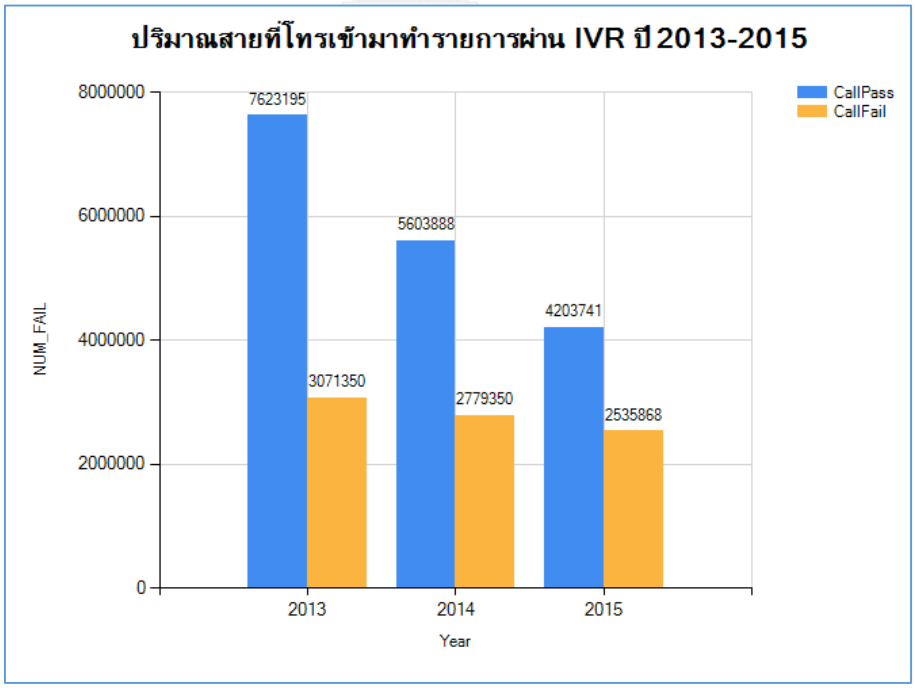
Top 10 IVR Menu Calls

Menu Code	No. of Calls
1.1.1	18844750
1.1.2.1	1150669
1	1026954
1.3.1	910488
1.1.2.2	831073
**11*	812998
1.1.2	643403
1.1	478252
1.3	338787
1.1.3.3	278883

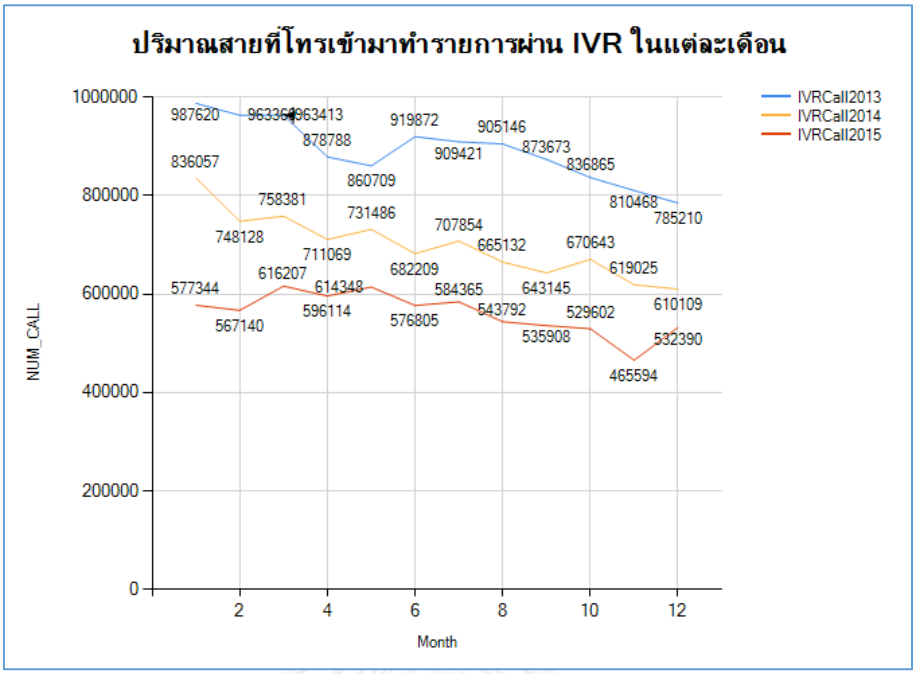
รูปที่ 70 ผลลัพธ์ตัวอย่างสคริปต์แสดงตารางแบบกำหนดจำนวนแถว

4) การทดสอบการแสดงผลแผนภูมิ

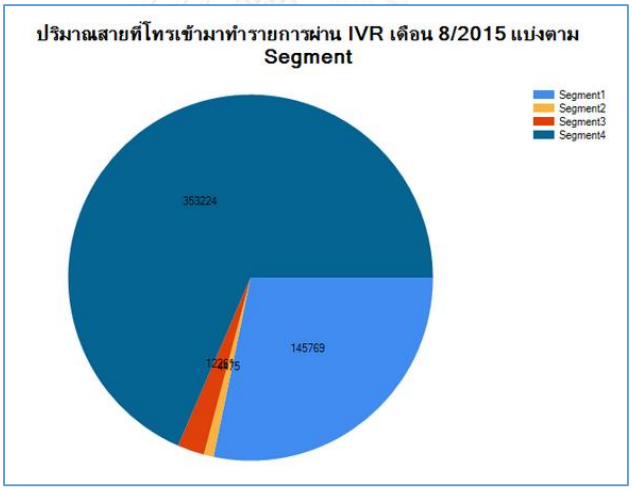
สามารถแสดงผลแผนภูมิแท่ง แผนภูมิเส้น และแผนภูมิวงกลม ได้ถูกต้องตามการดึงข้อมูลจากฐานข้อมูลดังรูปที่ 71 รูปที่ 72 และรูปที่ 73



รูปที่ 71 ผลลัพธ์ตัวอย่างสคริปต์แสดงแผนภูมิแท่ง



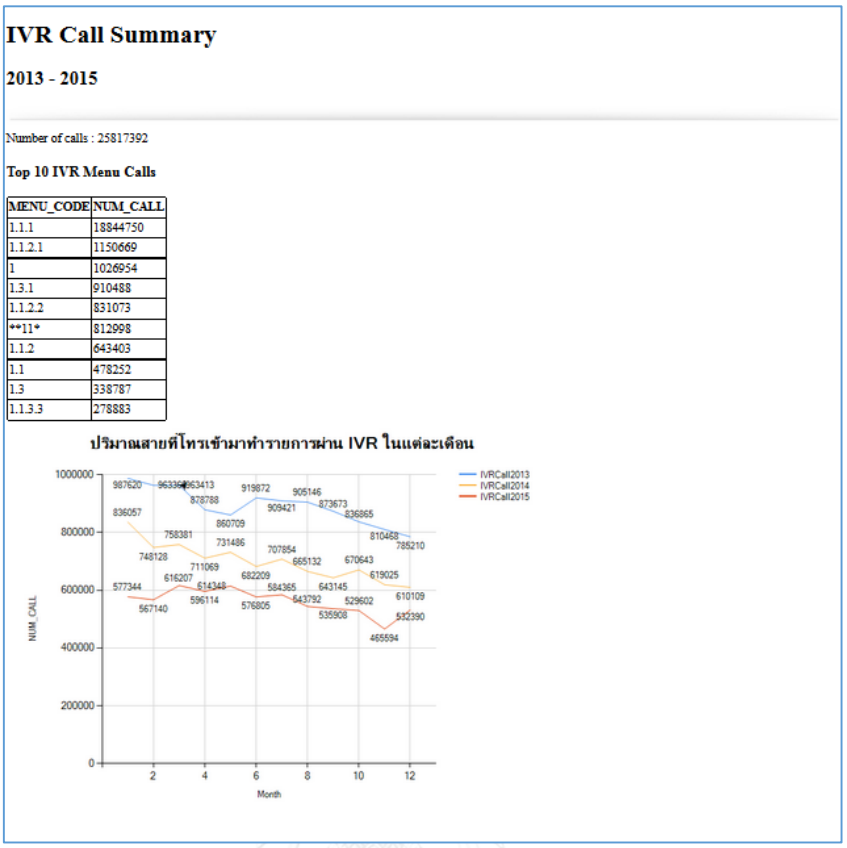
รูปที่ 72 ผลลัพธ์ตัวอย่างสคริปต์แสดงแผนภูมิเส้น



รูปที่ 73 ผลลัพธ์ตัวอย่างสคริปต์แสดงแผนภูมิวงกลม

5) การทดสอบการแสดงผลแบบหลายส่วนประกอบ

สามารถแสดงผลแบบหลายส่วนประกอบได้อย่างถูกต้องตามลำดับในสคริปต์ภาษา ดังรูปที่ 74



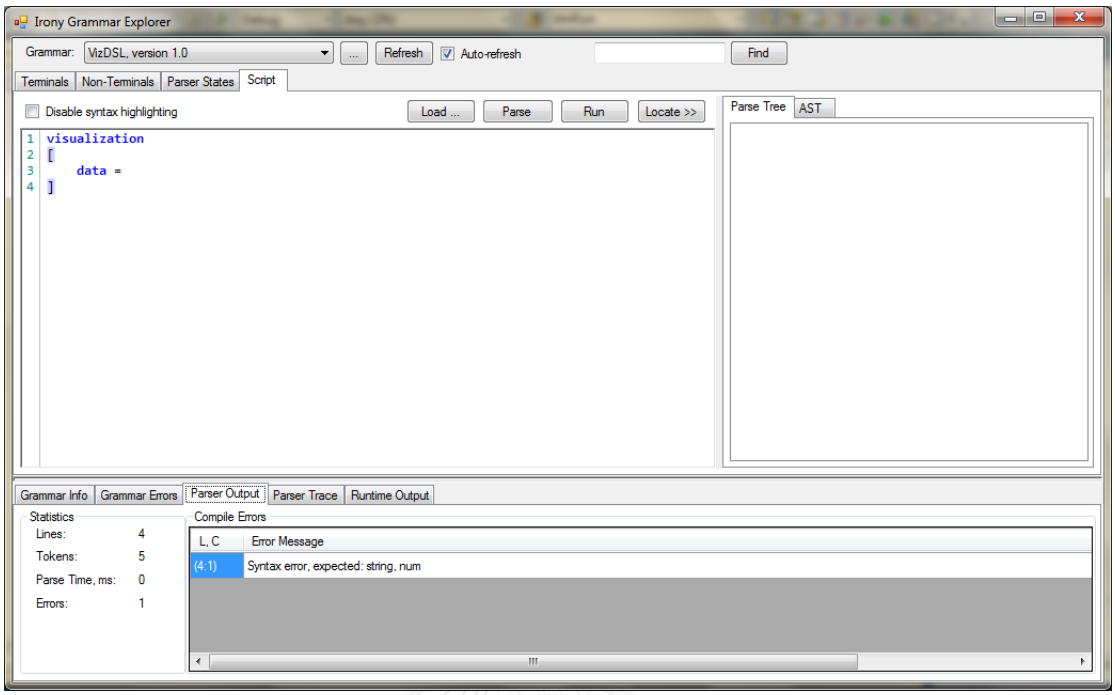
รูปที่ 74 ผลลัพธ์ตัวอย่างสคริปต์แสดงส่วนประกอบแสดงผลภาพหลายองค์ประกอบ

6) การทดสอบการไม่แสดงผลส่วนประกอบ

สามารถไม่แสดงผลส่วนประกอบแสดงผลภาพ

7) การทดสอบการเขียนสคริปต์ไม่ถูกต้อง

สามารถแสดงผลแจ้งเตือนความผิดพลาดบนหน้าจอแกรมมาเอกซ์พลอเรอร์ได้ดังรูปที่ 75



รูปที่ 75 ผลลัพธ์ตัวอย่างการแสดงผลการแจ้งเตือนความผิดพลาดบนแกรมมาเอกซ์พลอเรอร์

บทที่ 6
สรุปผลการวิจัย

6.1 สรุปผลการวิจัย

งานวิจัยนี้ได้นำเสนอวิธีการ และพัฒนาเครื่องมือสำหรับภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพที่ไม่ขึ้นกับแพลตฟอร์ม ซึ่งสามารถทำให้ผู้พัฒนามีอิสระในการเลือกใช้ไลบรารีสำหรับการแสดงผล และยังสามารถให้ผู้เชี่ยวชาญโดเมน สามารถเปลี่ยนแปลงสคริปต์ภาษาเพื่อทำการเปลี่ยนแปลงข้อมูลในการแสดงผลได้ด้วยตนเอง โดยไม่ต้องรอให้ผู้พัฒนาทำการแก้ไขใหม่ สามารถส่งผลให้ผลิตภาพกระบวนการเพิ่มมากขึ้น

ซึ่งจากการทดสอบการแสดงผลส่วนประกอบแสดงผลภาพจากสคริปต์ภาษา ในบทที่ 5 พบว่าภาษาจำเพาะโดเมนที่สร้างขึ้น สามารถใช้อธิบายการสร้างส่วนประกอบแสดงผลภาพได้อย่างถูกต้องตามผลลัพธ์ที่คาดหวัง

6.2 ข้อจำกัดในงานวิจัย

1) ภาษาจำเพาะโดเมนที่สร้างขึ้นรองรับส่วนประกอบแสดงผลภาพ ได้แก่ แผนภูมิแท่ง แผนภูมิเส้น แผนภูมิวงกลม ตาราง รูปภาพ และ ตัวอักษร เท่านั้น

2) ส่วนแก้ไขภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ ยังไม่สามารถตรวจสอบข้อผิดพลาดที่มากไปกว่าการพิมพ์ผิดหลักไวยากรณ์ของภาษา

3) การสร้างแบบจำลองความหมายในงานวิจัยนี้ จะอยู่ในรูปแบบเอกซ์เอ็มแอลเท่านั้น รวมถึงแอปพลิเคชันที่รองรับ จะต้องสามารถอ่านรูปแบบข้อมูลเอกซ์เอ็มแอลที่กำหนดได้

6.3 งานวิจัยในอนาคต

1) นำแนวทางหลักการในการพัฒนาภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ นำไปประยุกต์ให้รองรับส่วนประกอบแสดงผลภาพ และเพิ่มคุณลักษณะอื่น ๆ เช่น แผนภูมิกระจาย (Scatter chart) แผนภูมิหุ้น (Stock chart) เป็นต้น

2) ภาษาจำเพาะโดเมนสำหรับการสร้างส่วนประกอบแสดงผลภาพ อาจสร้างความสับสนกับผู้ใช้ที่ไม่คุ้นเคย หรือไม่เชี่ยวชาญได้ การสร้างส่วนต่อประสานสำหรับการช่วยเหลือผู้ใช้งาน จะทำให้ผู้ใช้งานสามารถใช้งานได้ถูกต้อง และเข้าใจได้ง่ายขึ้น

รายการอ้างอิง

1. Pentaho. *Pentaho*. [cited 2016; Available from: http://www.pentaho.com/.
2. Jaspersoft. *Jaspersoft*. [cited 2016; Available from: http://www.jaspersoft.com/.
3. Microsoft. *Microsoft*. [cited 2016; Available from: https://msdn.microsoft.com/en-us/library/ms159106.aspx.
4. Fowler, M., *Domain-Specific Languages*. 2010: Addison-Wesley Professional.
5. Bryant, B.R., J. Gray, and M. Mernik. *Domain-specific software engineering*. in *Proceedings of the FSE/SDP workshop on Future of software engineering research*. 2010. Santa Fe.
6. Mernik, M., J. Heering, and A.M. Sloane, *When and how to develop domain-specific languages.* ACM Computing Surveys (CSUR), 2005. **37**(4): p. 316-344.
7. Deursen, A.V., P. Klint, and J. Visser, *Domain-specific Languages: An Annotated Bibliography*, in *ACM SIGPLAN Notices Volume 35 Issue 6*. 2000. p. 26 - 36.
8. Hermans, F., M. Pinzger, and A.V. Deursen, *Domain-Specific Languages in Practice: A User Study on the Success Factors*, in *Model Driven Engineering Languages and Systems*. 2009, Springer Berlin Heidelberg. p. 423-437.
9. Tcholtchev, N., et al. *Integrating the Modelica DSL into a Platform for Model-Based Tool Interoperability*. in *Proceedings of the 2014 IEEE 38th International Computer Software and Applications Conference Workshops*. 2014. Vasteras.
10. Wegeler, T., et al. *Evaluating the benefits of using domain-specific modeling languages: an experience report*. in *Proceedings of the 2013 ACM workshop on Domain-specific modeling*. 2013.
11. Gartner, *IT Glossary : Business Intelligence (BI)*, in *Gartner*. 2016.
12. Editor, *Four main types of BI*, in *Technology Advice for Small Businesses*. 2016.
13. Jang, T., *Financial Services Commission English Blog*.
14. Wittwer, J., *New and Interesting Excel Templates and Spreadsheets*. 2013.

15. Larimer, D., *Charts on Grids – Responsive Dashboard Templates with Bootstrap*. 2014.
16. admin, *Gold Price 2015: Forecasts And Predictions*. 2015.
17. Dantra, R., J. Grundy, and J. Hosking. *A domain-specific visual language for report writing using Microsoft DSL tools*. in *Visual Languages and Human-Centric Computing*. 2009. Corvallis, OR.
18. ISO/IEC, *ISO/IEC 14977:1996(E) First edition -- Information technology -- Syntactic metalanguage -- Extended BNF*. 1996.

ภาคผนวก

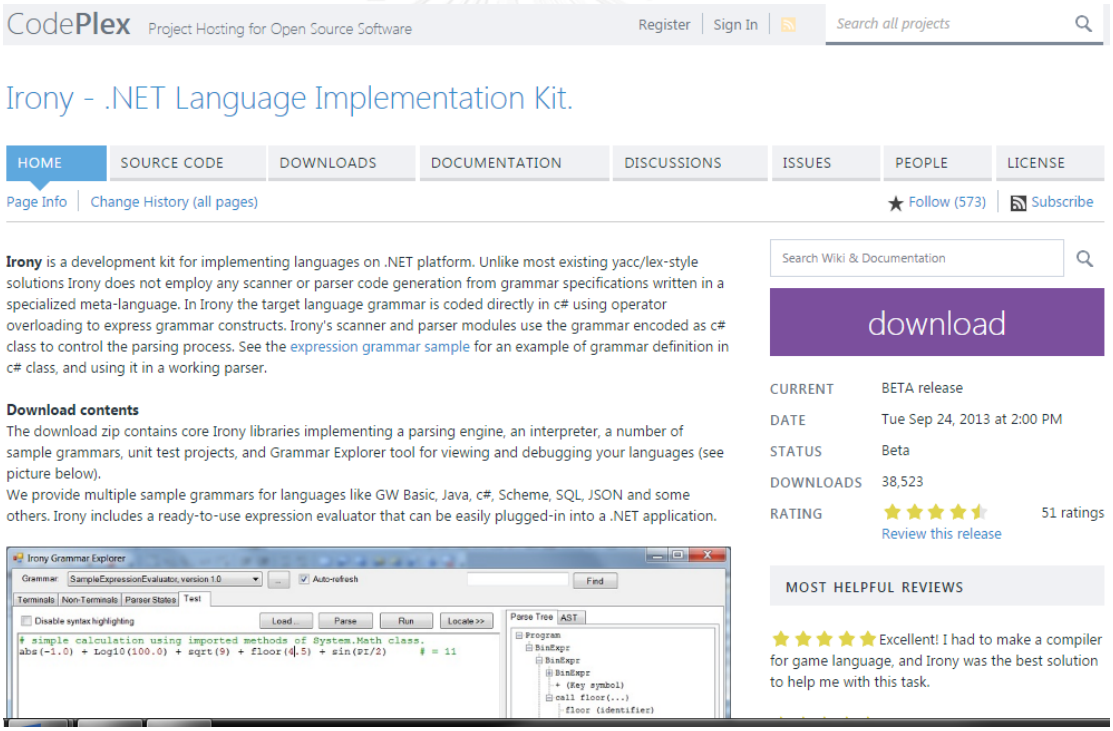
ภาคผนวก

ไอโรนี (Irony -.NET Language Implementation Kit)

ไอโรนีเป็นชุดพัฒนาสำหรับการสร้างภาษาจำเพาะโดเมนบนดอทเนตแพลตฟอร์ม โดยพัฒนาด้วยภาษา C# ไวยากรณ์ที่ถูกเขียนแบบอีบีเอ็นเอฟ (EBNF) สามารถนำมาสร้างเป็นไวยากรณ์ที่ถูกกำหนดด้วยภาษา C# ได้ด้วยการเรียกเอพีไอของไลบรารีไอโรนี ไอโรนีไม่จำเป็นต้องรวมส่วนสแกนเนอร์ และตัวแจงส่วนเข้าไปในการสร้างไวยากรณ์ที่ถูกเขียนโดยระบุบนเมตาภาษา (meta-language) ไอโรนีสามารถสร้างวากยสัมพันธ์แบบต้นไม้ได้จากสคริปต์ภาษาที่เขียน

1. การเริ่มต้นใช้งาน

1) ทำการดาวน์โหลดไอโรนีจากเว็บ https://irony.codeplex.com/ โดยทำการกดปุ่ม download ดังรูปที่ 76

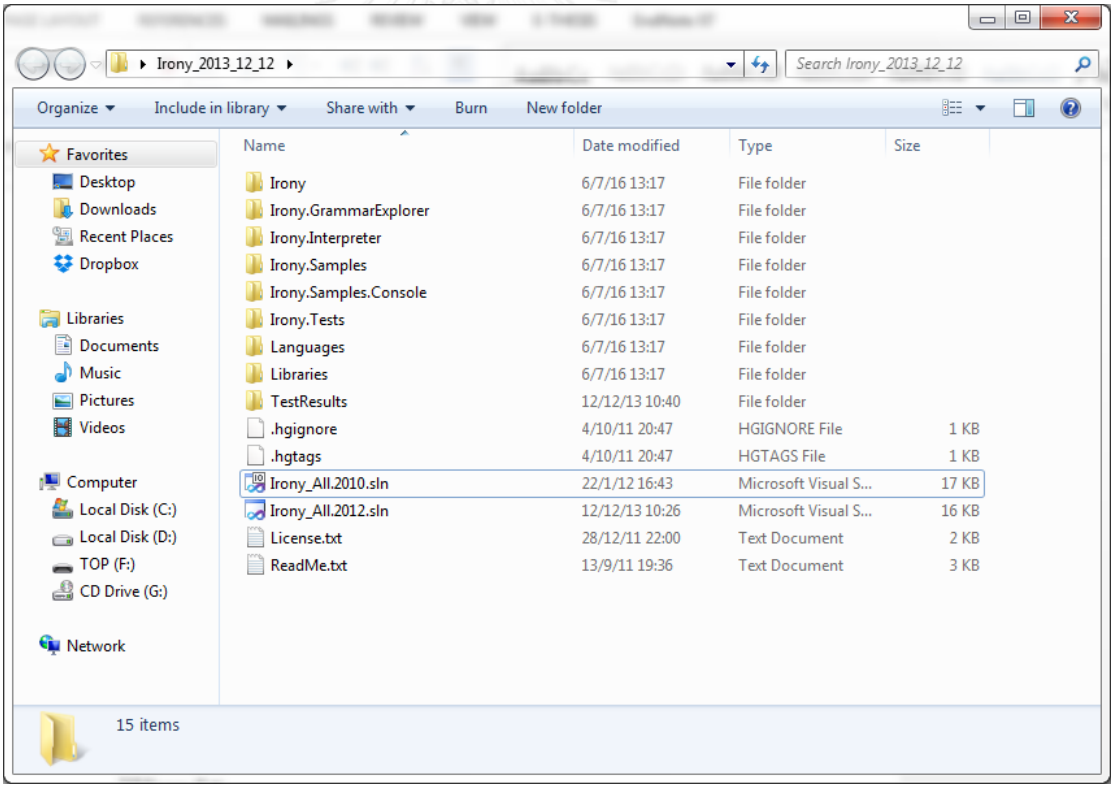


รูปที่ 76 หน้าจอเว็บ https://irony.codeplex.com/

2) เมื่อทำการดาวน์โหลดไอโรนีแล้วจะเห็นว่าไฟล์ที่ได้มาอยู่ในรูปแบบซิปไฟล์ ให้ทำการแตกไฟล์ จะได้โฟลเดอร์ที่เก็บไลบรารีของไอโรนี ดังรูปที่ 77

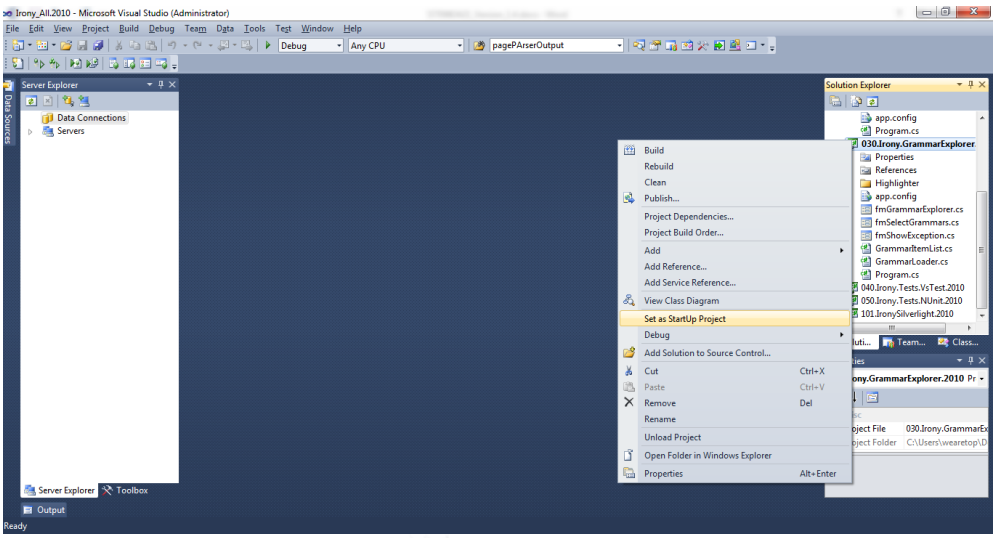
รูปที่ 77 โฟลเดอร์ที่เก็บไฟล์ไลบรารีไอโรนี

3) เมื่อเปิดเข้าโฟลเดอร์มาจะพบโซลูชันของวิชวลสตูดิโอ อยู่สองแบบ คือแบบที่เปิดกับวิชวลสตูดิโอ 2010 และ 2012 ให้ทำการเลือกเปิดโซลูชันตามเวอร์ชันของวิชวลสตูดิโอที่ต้องการใช้พัฒนา ดังรูปที่ 78



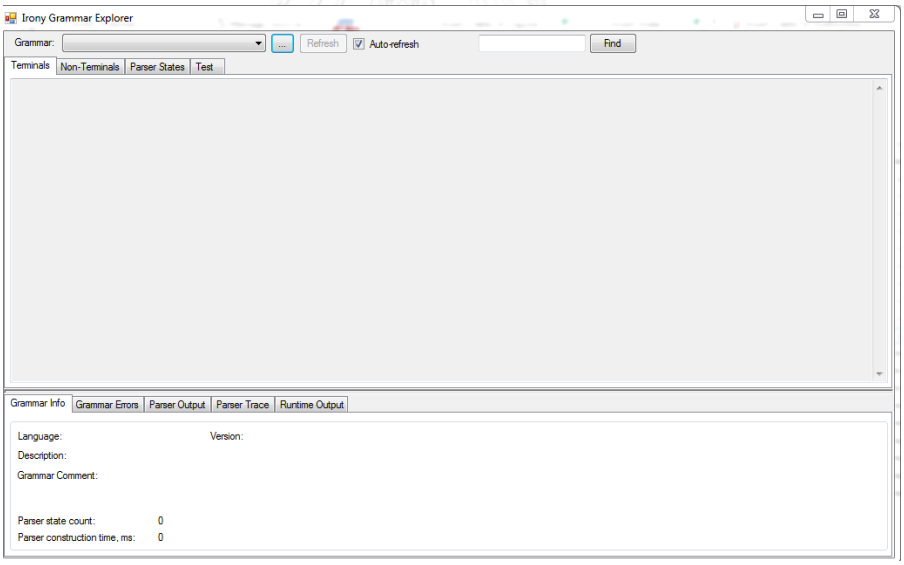
รูปที่ 78 ไฟล์ในโฟลเดอร์ไอโรนี

4) เมื่อเปิดวิชวลสตูดิโอแล้วให้ทำการเลือกโปรเจค Irony.GrammarExplorer และคลิกขวาเลือก Set as Startup Project ดังรูปที่ 79



รูปที่ 79 เซตโปรเจค Irony.GrammarExplorer ให้เป็นค่าเริ่มต้น

5) ทำการกดปุ่ม Debug ( ) จะพบหน้าต่างโปรแกรม Irony Grammar Explorer ซึ่งใช้สำหรับการตรวจสอบไวยากรณ์ที่ถูกสร้างขึ้นดังรูปที่ 80

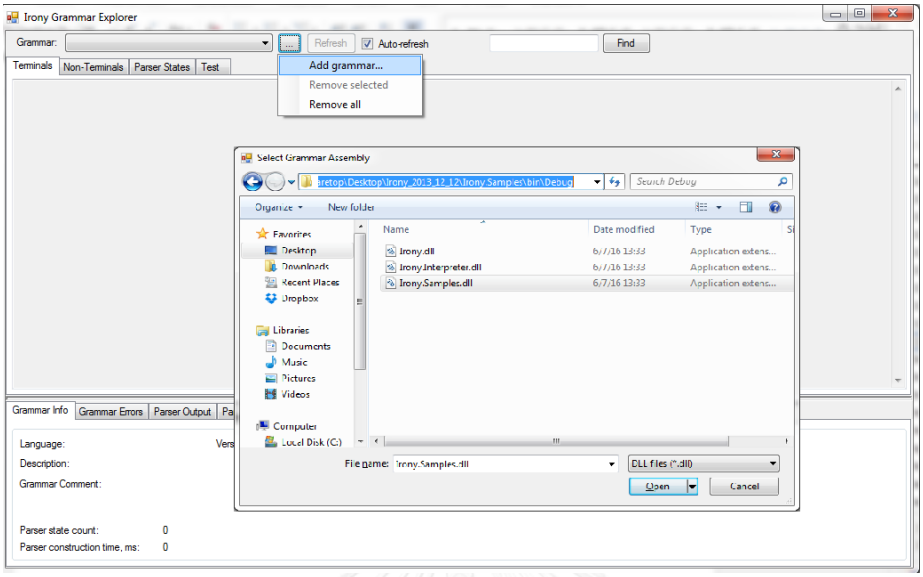


รูปที่ 80 หน้าจอโปรแกรม Irony Grammar Explorer

2. การนำเข้า และการตรวจสอบไวยากรณ์ที่สร้างขึ้น

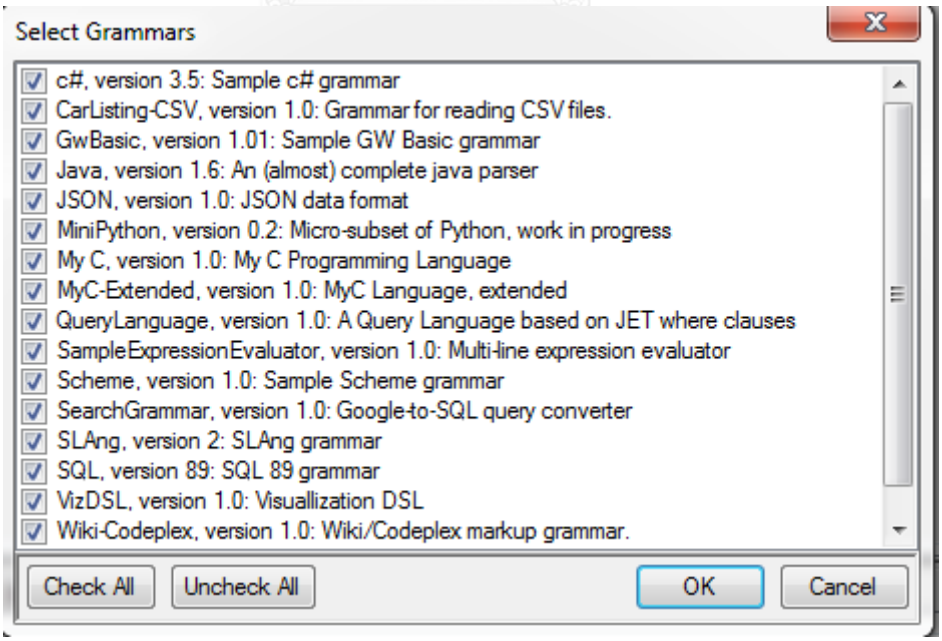
1) จากหน้าจอ Irony Grammar Explorer กดปุ่ม และ เลือก Add grammar จะนำไปสู่หน้าต่างให้เลือกไลบรารีของไวยากรณ์ที่ได้ทำการคอมไพล์แล้ว โดยทางไอโรนี ได้มีตัวอย่าง

ไลบรารีไวยากรณ์ ชื่อว่า Irony.Sample.dll ซึ่งอยู่ภายใต้พาธ Irony.Samples\bin\Debug\ ดังรูปที่ 81



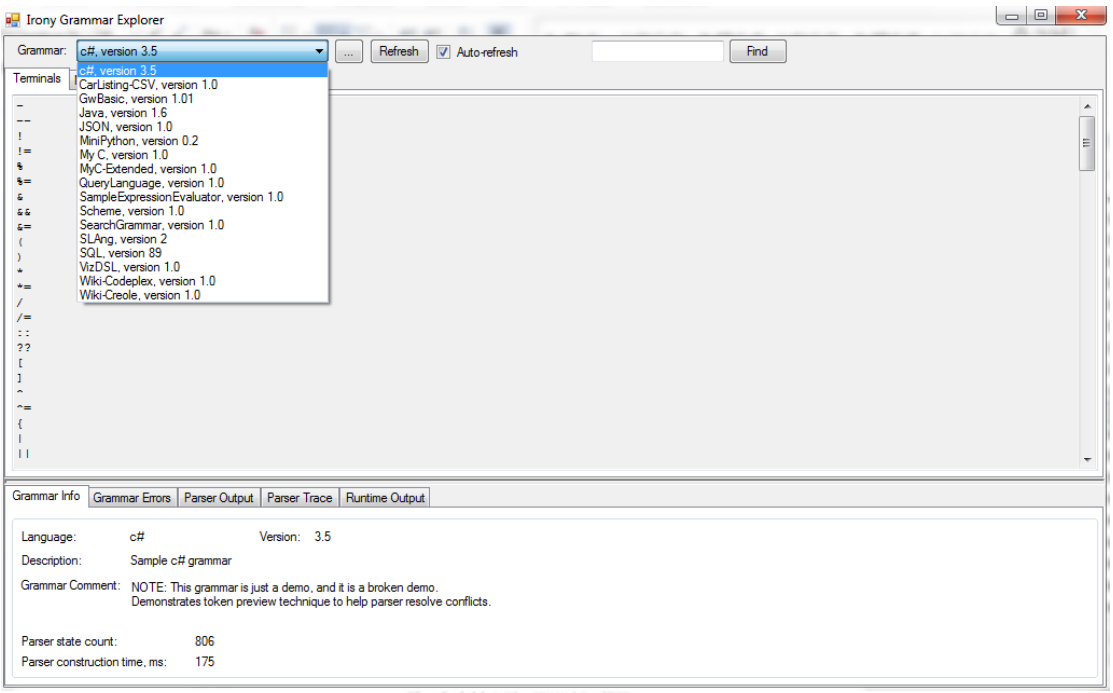
รูปที่ 81 หน้าจอเลือกไลบรารีของไวยากรณ์

2) เมื่อทำการเลือกไลบรารีแล้ว จะพบหน้าให้เลือกไวยากรณ์ที่อยู่ภายใต้ไลบรารีนั้น ให้กด OK ดังรูปที่ 82



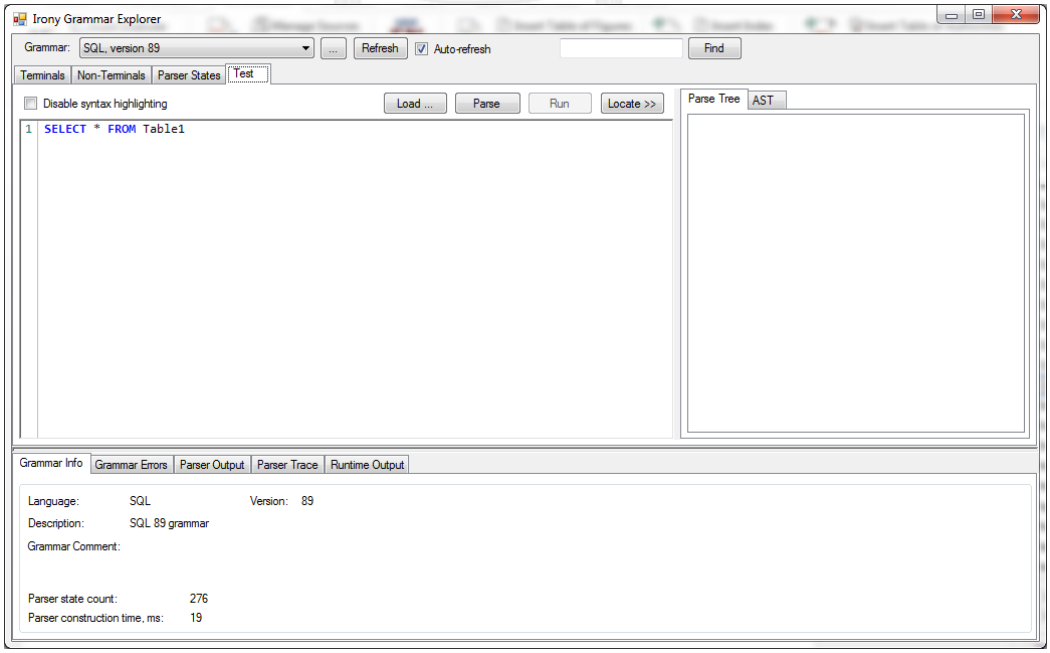
รูปที่ 82 หน้าต่างเลือกไวยากรณ์

3) ที่ดรอปดาวน์ลิสต์นั้น จะมีลิสต์ของไวยากรณ์ที่ถูกนำเข้ามา ดังรูปที่ 83



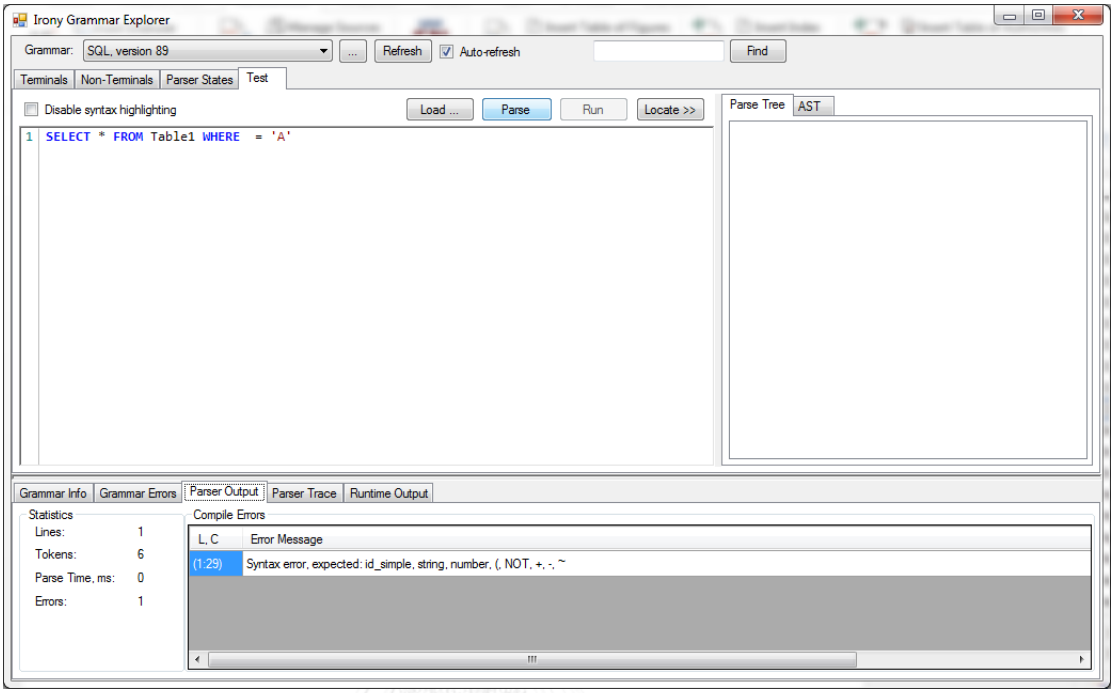
รูปที่ 83 ลิสต์ของไวยากรณ์ที่ถูกนำเข้า

4) เมื่อเลือกแทป Test จะพบพื้นที่ให้ทดสอบเขียนสคริปต์ภาษา เมื่อทำการทดสอบแล้ว ให้กดปุ่ม Parse โปรแกรมจะทำแสดงต้นไม้แจงส่วนขึ้น เมื่อสคริปต์ภาษาที่เขียนนั้นถูกต้องตามไวยากรณ์ที่เลือกมา ดังรูปที่ 84



รูปที่ 84 การทดสอบไวยากรณ์ที่นำเข้า

5) เมื่อสคริปต์ภาษาที่เขียนไม่ตรงตามหลักไวยากรณ์ที่นำเข้ามาจะขึ้นข้อความแสดงความผิดพลาดดังรูปที่ 85



รูปที่ 85 หน้าจอพบข้อผิดพลาดของสคริปต์ภาษา

ประวัติผู้เขียนวิทยานิพนธ์

นายอิทธิพล นันทรุจิ เกิดเมื่อวันที่ 17 กรกฎาคม พ.ศ. 2534 ที่จังหวัดลำพูน สำเร็จการศึกษาปริญญาตรีหลักสูตรวิศวกรรมศาสตรบัณฑิต (วศ.บ.) สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยมหิดล ในปีการศึกษา 2555 และเข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิศวกรรมซอฟต์แวร์ ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2557