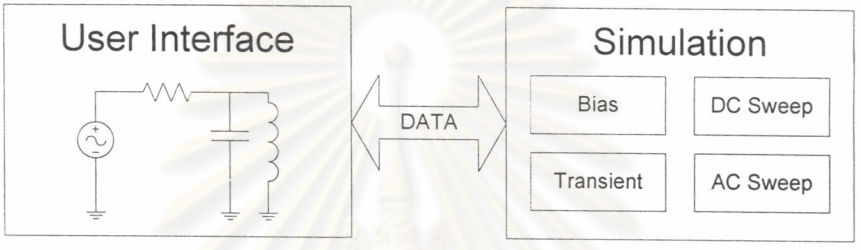
บทที่ 6

การออกแบบโปรแกรม

โปรแกรมวิเคราะห์วงจรรวมที่พัฒนาขึ้นในวิทยานิพนธ์นี้สามารถแบ่งออกเป็น 2 ส่วนสำคัญ ได้แก่ส่วนติดต่อกับผู้ใช้ (User Interface) และส่วนทำการวิเคราะห์ (Simulation) ดังแสดงในรูปที่ 6.1 ในบทนี้จะอธิบายถึงการออกแบบและการทำงานของโปรแกรม



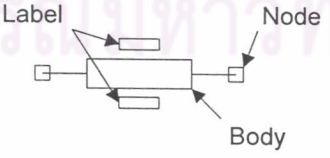
รูปที่ 6.1 โครงสร้างของโปรแกรมวิเคราะห์วงจรรวม

6.1 ส่วนติดต่อกับผู้ใช้

6.1.1 การแสดงผลอุปกรณ์ไฟฟ้า

อุปกรณ์ไฟฟ้าทั่วไปจะมีส่วนประกอบดังรูปที่ 6.2 ซึ่งแบ่งรายละเอียดออกเป็น

- คลาส Body ทำหน้าที่เกี่ยวกับการแสดงรูปร่างของอุปกรณ์ไฟฟ้า
- คลาส Node ทำหน้าที่จัดการกับการเชื่อมต่ออุปกรณ์ไฟฟ้ากับอุปกรณ์ไฟฟ้าตัวอื่น
- คลาส Label ทำหน้าที่แสดงผลรายละเอียดเป็นตัวอักษรกำกับอุปกรณ์ไฟฟ้า

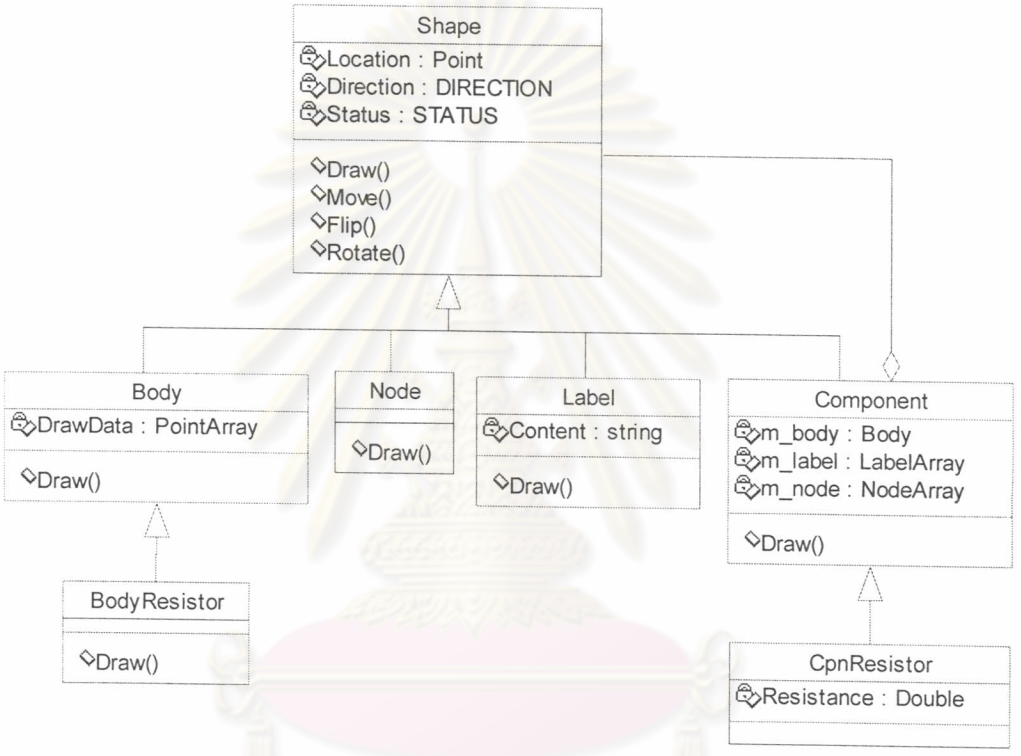


รูปที่ 6.2 รูปร่างทั่วไปของอุปกรณ์ไฟฟ้า

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

6.1.2 คลาส Component

คลาส Component เป็นคลาสที่จัดการกับการวาดรูปอุปกรณ์ไฟฟ้าแต่ละชนิด และรวบรวมค่าพารามิเตอร์ที่ใช้ในการคำนวณเพื่อที่จะส่งให้กับส่วนที่ทำการวิเคราะห์ต่อไป ตัวอย่างของแผนภาพคลาสของอุปกรณ์ไฟฟ้าชนิดตัวต้านทาน แสดงได้ดังรูปที่ 6.3 สัญลักษณ์ ◇— ในรูปที่ 6.3 หมายความว่าคลาส Component มีคลาส Shape เป็นสมาชิกอยู่ภายใน ซึ่งในที่นี้ได้แก่คลาส Body, Label และ Node



รูปที่ 6.3 แผนภาพคลาสของอุปกรณ์ตัวต้านทาน

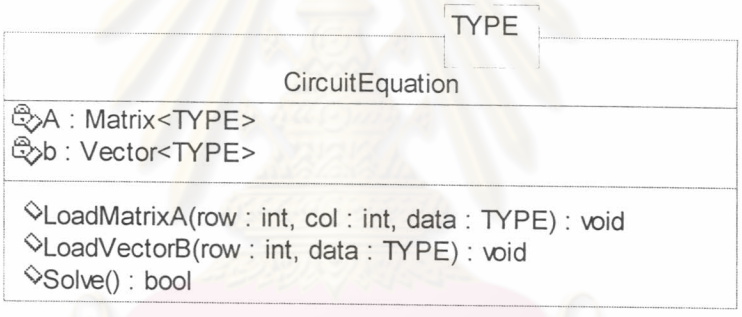
จากรูปที่ 6.3 จะเห็นว่าในการพัฒนาการแสดงผลอุปกรณ์ไฟฟ้าแต่ละชนิดนั้น เราจะสร้างคลาสสำหรับการวาดรูปสัญลักษณ์โดยให้สืบทอดจากคลาส Body จากนั้นจึงสร้างคลาสใหม่ที่สืบทอดจากคลาส Component และให้คลาสนี้เก็บค่าพารามิเตอร์ที่ต้องการ และใช้หลักการการพ้องรูปเพื่อให้ตัวชี้ m_body ชี้ไปยังพื้นที่หน่วยความจำที่จองไว้เป็นวัตถุจากคลาสที่สืบทอดจากคลาส Body ที่สร้างขึ้นใหม่ ส่วนการทำงานพื้นฐานจะอยู่ในคลาส Component อยู่แล้วจึงไม่ต้องไปยุ่งเกี่ยวอีก

6.2 ส่วนที่ทำการวิเคราะห์

โปรแกรมวิเคราะห์วงจรรวมที่พัฒนาขึ้นนี้สามารถวิเคราะห์การทำงานวงจรไฟฟ้าได้ทั้งหมด 4 แบบด้วยกัน ได้แก่การวิเคราะห์หาจุดทำงานสงบ จุดทำงานสงบแบบกวาด ผลตอบสนองเชิงเวลา และผลตอบสนองเชิงความถี่ รายละเอียดของคลาสเกี่ยวข้องกับการวิเคราะห์หาคำตอบมีรายละเอียดดังนี้

6.2.1 คลาส CircuitEquation

คลาส CircuitEquation เป็นคลาสที่มีหน้าที่ควบคุมขั้นตอนในการแก้สมการวงจรไฟฟ้า เนื่องจากในการแก้สมการวงจรไฟฟ้านั้น ข้อมูลที่ใช้จะมีทั้งเลขจำนวนจริงและจำนวนเชิงซ้อน ดังนั้นเราจะใช้คุณสมบัติเทมเพลท (Template) ในภาษา C++ ซึ่งเป็นการกำหนดการทำงานโดยที่ยังไม่กำหนดชนิดของข้อมูลภายใน ในการใช้งานผู้ใช้จึงค่อยระบุชนิดของข้อมูลที่ต้องการ โครงสร้างของคลาส CircuitEquation แสดงได้ดังรูปที่ 6.4



รูปที่ 6.4 คลาส CircuitEquation

จากรูปที่ 6.4 ข้อมูลภายในจะแทนด้วยพารามิเตอร์ TYPE เมื่อเวลาใช้งานจริง ชนิดของข้อมูลที่ผู้ใช้กำหนดจะแทนที่ TYPE ในการทำงาน เราสามารถประกาศการใช้งานคลาส CircuitEquation ได้ด้วยรหัสในภาษา C++ ดังนี้

```
CircuitEquation<double> realEquation;
CircuitEquation<Complex> complexEquation;
```

ซึ่งจะได้ตัวแปร realEquation ที่จัดการกับข้อมูลที่เป็นเลขจำนวนจริงเนื่องจากจะใช้ชนิดข้อมูล double แทน และตัวแปร complexEquation ที่จัดการกับข้อมูลที่เป็นเลขจำนวนเชิงซ้อน

ขั้นตอนในการแก้สมการวงจรไฟฟ้าเราจะใช้การแยกส่วนและการแทนที่ดังที่ได้กล่าวไว้ในบทที่ 2 และแสดงด้วยรหัสเทียมดังนี้

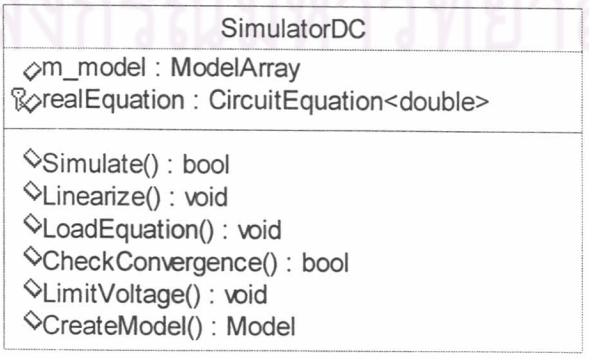
```
bool CircuitEquation::Solve()
{
    1. LU factorization
    2. LU Backsubstitution
}
```

6.2.2 คลาส SimulatorDC

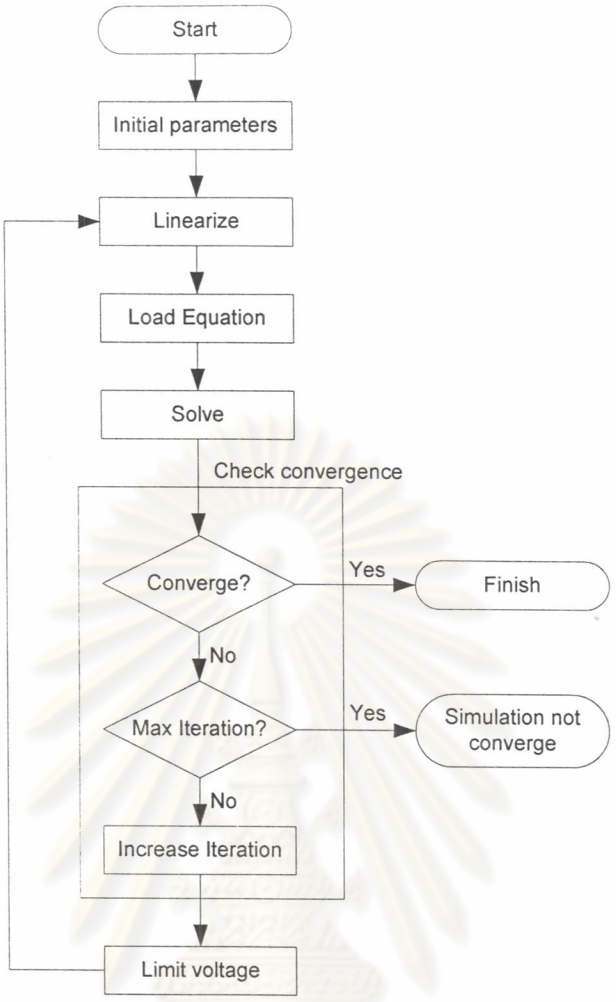
คลาส SimulatorDC เป็นคลาสที่ควบคุมขั้นตอนในการวิเคราะห์หาจุดทำงานสงบ ขั้นตอนการทำงานแสดงได้ดังรูปที่ 6.6 โดยมีรายละเอียดดังนี้

1. Linearize เป็นขั้นตอนคำนวณเชิงเส้น เพื่อให้ได้ค่าที่จะใช้ในการสร้างสมการเมทริกซ์สำหรับอุปกรณ์ไฟฟ้าที่มีการทำงานไม่เชิงเส้น
2. Load Equation เป็นขั้นตอนในการสร้างสมการเมทริกซ์ของวงจรไฟฟ้า
3. Solve เป็นขั้นตอนในการแก้สมการวงจรไฟฟ้า
4. Check convergence เป็นขั้นตอนตรวจสอบการลู่เข้าหาคำตอบของผลลัพธ์ที่ได้จากการแก้สมการวงจรไฟฟ้า
5. Limit voltage เป็นขั้นตอนในการจำกัดแรงดันของอุปกรณ์ไฟฟ้าเช่นไดโอดและทรานซิสเตอร์ชนิดมอส

รูปที่ 6.5 แสดงโครงสร้างคลาสของ SimulatorDC ซึ่งมีตัวแปร realEquation เพื่อใช้ในการแก้สมการวงจรไฟฟ้า และ m_model เป็นตัวแปรจากวัตถุจากคลาส Model ที่ใช้แทนการทำงานของอุปกรณ์ไฟฟ้าชนิดต่างๆ ฟังก์ชัน Simulate() เป็นฟังก์ชันในควบคุมขั้นตอนในการวิเคราะห์หาคำตอบ ฟังก์ชัน Linearize() เป็นฟังก์ชันสำหรับขั้นตอนคำนวณเชิงเส้น ฟังก์ชัน LoadEquation() เป็นฟังก์ชันที่ให้วัตถุจากคลาส Model ทำการสร้างสมการเมทริกซ์ตามชนิดของอุปกรณ์ไฟฟ้าที่กำหนด สำหรับขั้นตอน Solve นั้นจะเป็นหน้าที่การทำงานของคลาส CircuitEquation ฟังก์ชัน LimitVoltage() เป็นฟังก์ชันในขั้นตอนการจำกัดแรงดันของอุปกรณ์ไฟฟ้า รายละเอียดของฟังก์ชัน CreateModel() จะกล่าวถึงในหัวข้อ 6.2.8



รูปที่ 6.5 คลาส SimulatorDC



รูปที่ 6.6 ขั้นตอนการหาจุดทำงานสงบ

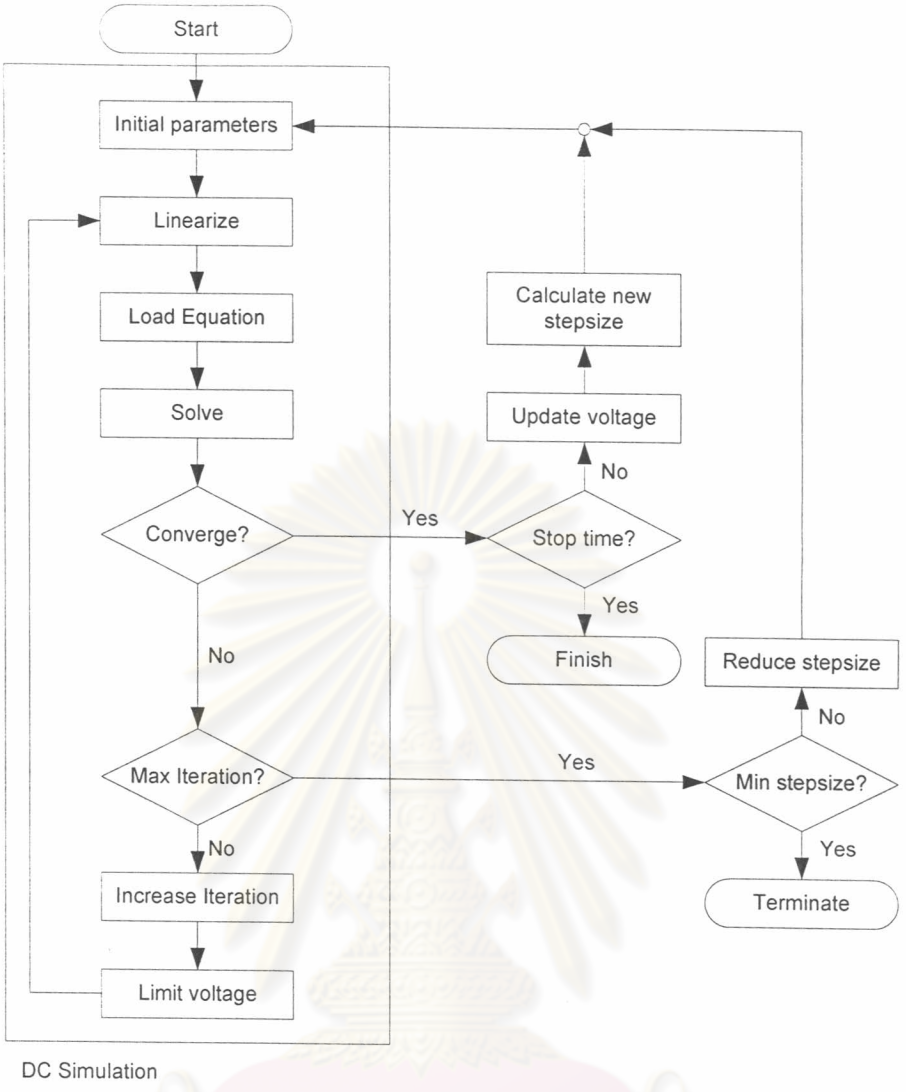
6.2.3 คลาส SimulatorTR

คลาส SimulatorTR เป็นคลาสที่ควบคุมขั้นตอนในการวิเคราะห์หาผลตอบสนองเชิงเวลา ขั้นตอนการทำงานแสดงได้ดังรูปที่ 6.7 ซึ่งขั้นตอนหลักจะเหมือนกับขั้นตอนในการหาจุดทำงานสงบ แต่จะเพิ่มขั้นตอนในการคำนวณขั้นเวลาโดยมีรายละเอียดดังนี้

1. Update voltage เป็นขั้นตอนในการปรับค่าแรงดันในจุดอดีตสำหรับอุปกรณ์ไฟฟ้าที่ใช้วิธีของออยเลอร์เช่นตัวเก็บประจุและตัวเหนี่ยวนำ

2. Calculate new stepsize เป็นขั้นตอนที่จะคำนวณหาขนาดของขั้นเวลาที่ใช้ในการคำนวณจุดเวลาถัดไป หากการหาผลลัพธ์ลู่เข้าก็จะขยายขั้นเวลาให้มากขึ้นเพื่อเร่งการคำนวณ

3. Reduce stepsize เป็นขั้นตอนในการลดขนาดของขั้นเวลา หากการหาผลลัพธ์ไม่ลู่เข้า

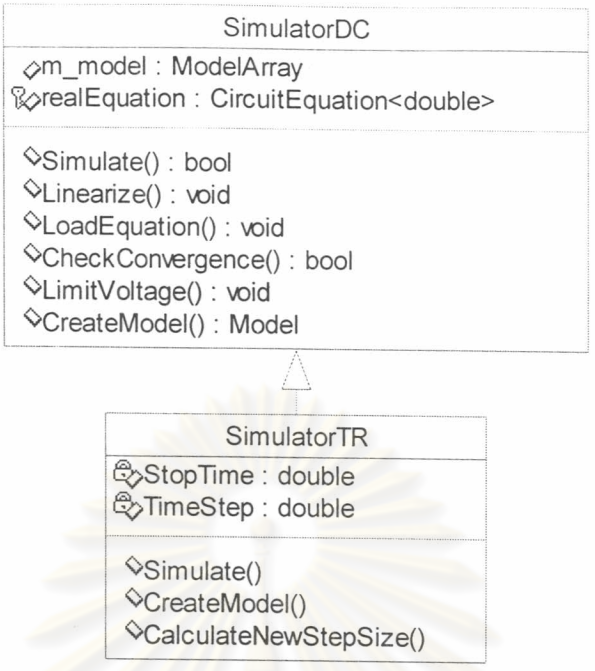


รูปที่ 6.7 ขั้นตอนการหาผลตอบสนองเชิงเวลา

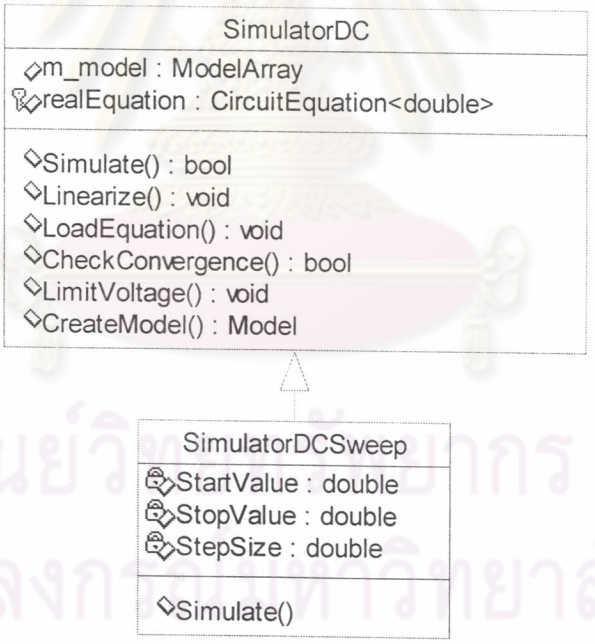
โครงสร้างคลาสของคลาส SimulatorTR แสดงได้ดังรูปที่ 6.8 โดยมีคุณลักษณะที่เพิ่มเติมจากคลาส SimulatorDC คือตัวแปร StopTime เป็นตัวแปรกำหนดเวลาสุดท้ายในการคำนวณ และตัวแปร TimeStep เป็นตัวแปรที่กำหนดขั้นเวลามากที่สุดที่โปรแกรมจะใช้ในการคำนวณ และฟังก์ชัน CalculateNewStepSize() เป็นฟังก์ชันที่ใช้คำนวณขั้นเวลาที่จะใช้คำนวณในจุดเวลาถัดไป

6.2.4 คลาส SimulatorDCSweep

คลาส SimulatorDCSweep เป็นคลาสที่ควบคุมขั้นตอนในการวิเคราะห์หาจุดทำงานสงบแบบกวาด ขั้นตอนการทำงานแสดงได้ดังรูปที่ 6.10 ขั้นตอนหลักจะเหมือนกับการหาจุดทำงานสงบแต่จะเพิ่มขั้นตอนในการปรับค่าพารามิเตอร์ตามที่ผู้ใช้กำหนด

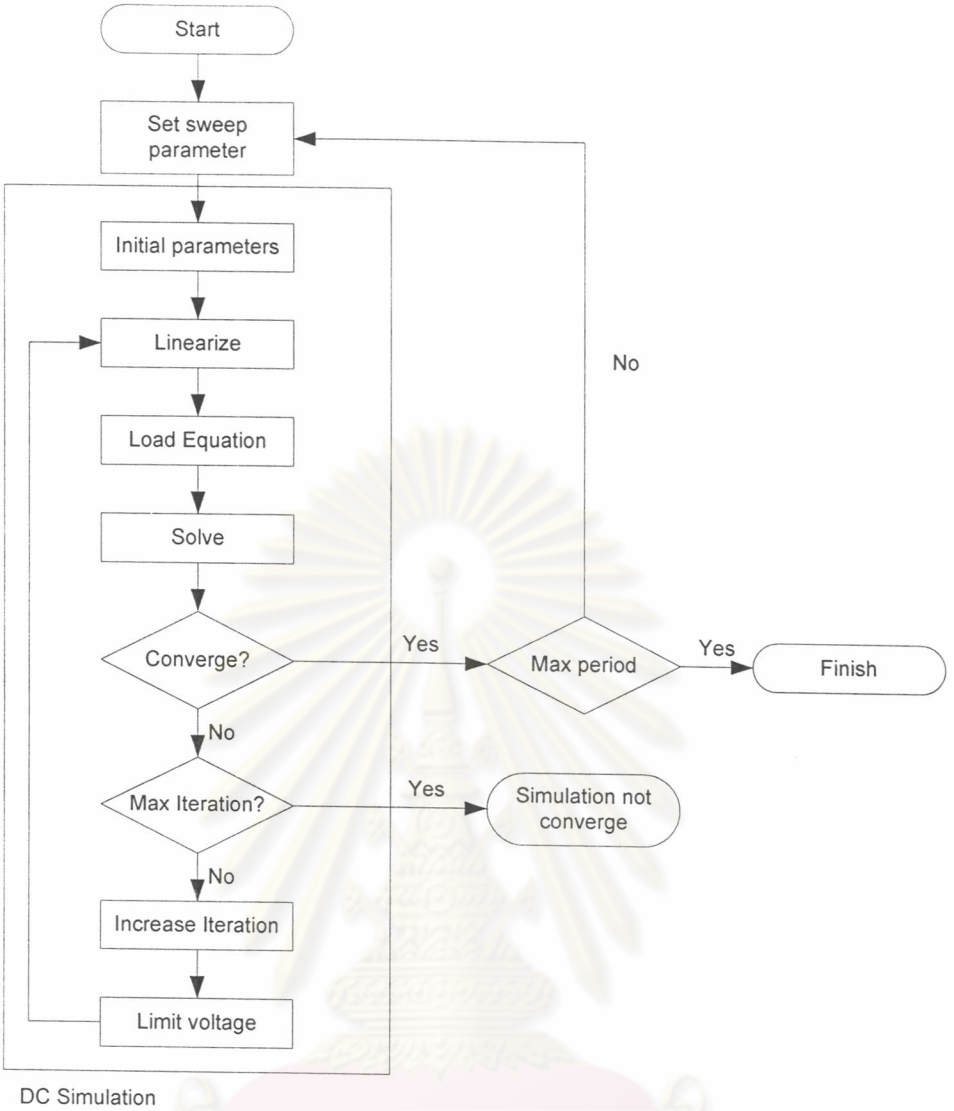


รูปที่ 6.8 คลาส SimulatorTR



รูปที่ 6.9 คลาส SimulatorDCSweep

โครงสร้างของคลาส SimulatorDCSweep แสดงได้ดังรูปที่ 6.9 โดยมีคุณลักษณะที่เพิ่มเข้ามาคือตัวแปร StartValue, StopValue และ StepSize เป็นตัวแปรกำหนดค่าเริ่มต้นการกวาด ค่าสิ้นสุดการกวาด และขั้นของการกวาดค่าพารามิเตอร์ที่ผู้ใช้กำหนดตามลำดับ

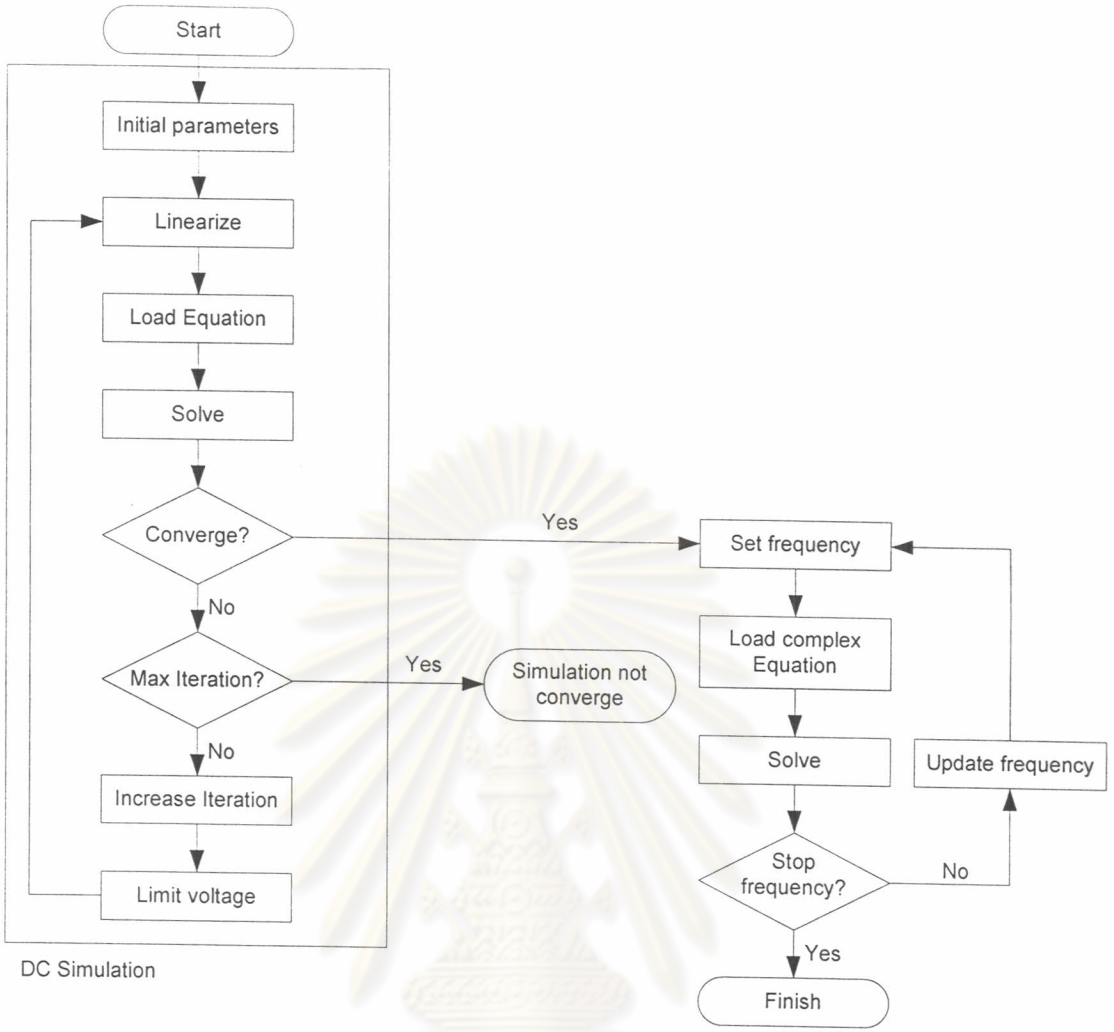


รูปที่ 6.10 ขั้นตอนการหาจุดทำงานสงบแบบกวาด

6.2.5 คลาส SimulatorAC

คลาส SimulatorAC เป็นคลาสที่ควบคุมขั้นตอนการหาผลตอบสนองเชิงความถี่ ขั้นตอนในการหาผลตอบสนองเชิงความถี่จะเป็นการคำนวณหาจุดทำงานสงบ แล้วจึงคำนวณหาผลตอบสนองเชิงความถี่ ดังแสดงในรูปที่ 6.11

โครงสร้างคลาสของคลาส SimulatorAC แสดงได้ดังรูปที่ 6.12 โดยมีคุณลักษณะที่เพิ่มเข้ามาคือตัวแปร StartFreq, StopFreq และ FreqPoints เป็นตัวแปรกำหนดความถี่เริ่มต้นในการกวาด ความถี่สิ้นสุดในการกวาดและจำนวนจุดของความถี่ที่ใช้ในการคำนวณ ตัวแปร complexEquation เป็นตัวแปรเพื่อใช้แก้สมการวงจรไฟฟ้าโดยจะจัดการกับข้อมูลในรูปแบบเลขจำนวนเชิงซ้อน



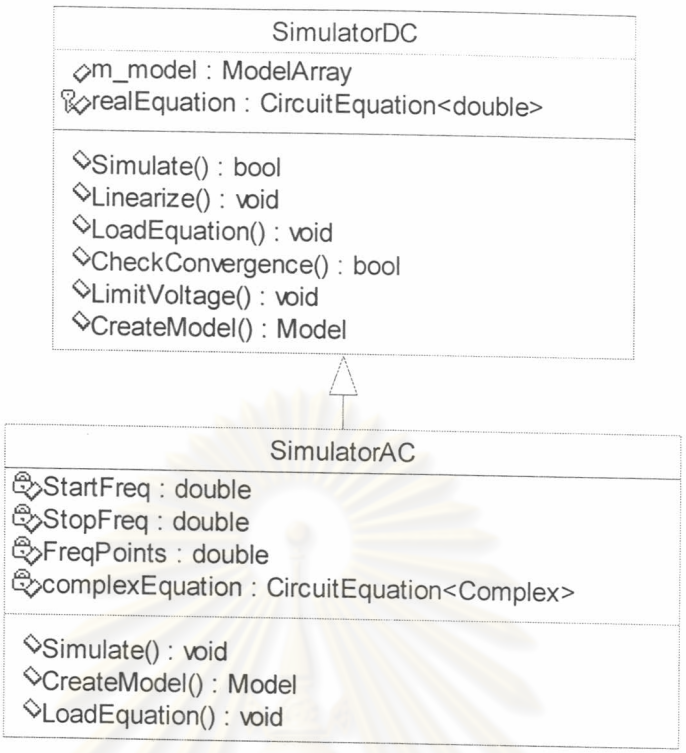
รูปที่ 6.11 ขั้นตอนการหาผลตอบสนองเชิงความถี่

6.2.6 คลาส Simulator

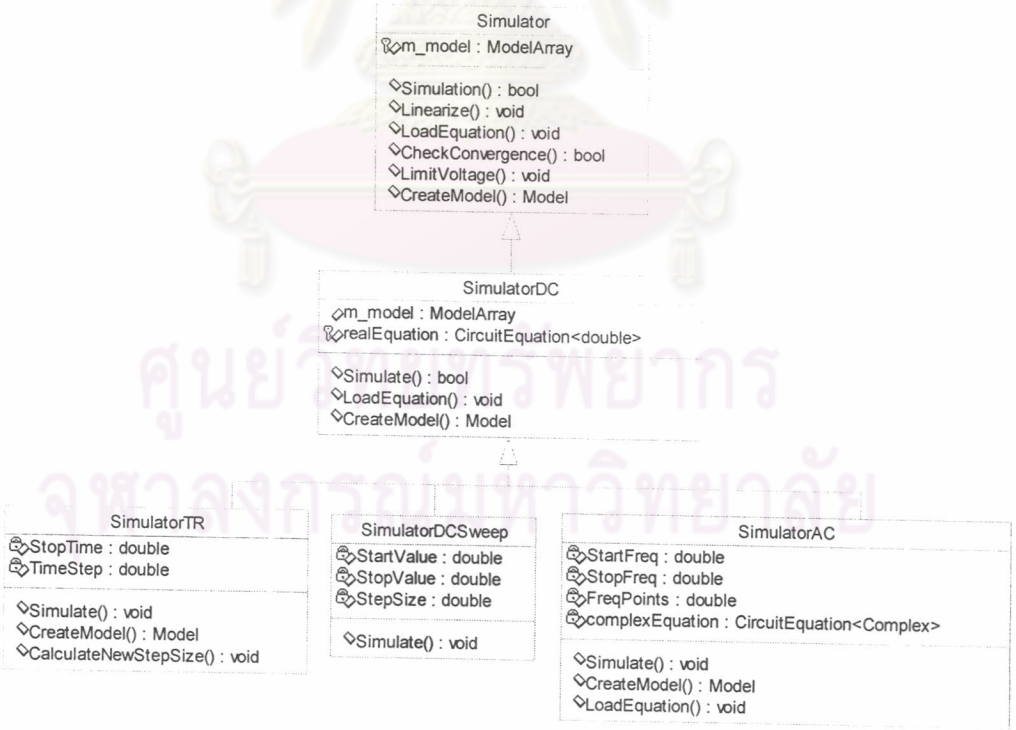
คลาส Simulator เป็นคลาสแม่ของการวิเคราะห์การทำงานวงจรไฟฟ้า ซึ่งจะรวมเอาการทำงานพื้นฐานของคลาส SimulatorDC, SimulatorTR, SimulatorDCSweep และ SimulatorAC ไว้ ซึ่งจะทำให้คลาสลูกเหล่านี้มีการทำงานที่ซับซ้อนน้อยลง เมื่อนำเอาการทำงานพื้นฐานมาไว้ในคลาส Simulator แล้ว จะทำให้ได้โครงสร้างคลาสของคลาส Simulator ใหม่ดังรูปที่ 6.13

6.2.7 คลาส Model

คลาส Model เป็นคลาสที่กำหนดวิธีใส่ตราประจำอุปกรณ์ของอุปกรณ์ไฟฟ้าแต่ละชนิด หน้าที่หลักของคลาส Model นี้คือการกำหนดตราประจำอุปกรณ์ลงที่ตำแหน่งแถวและหลักของเมทริกซ์ A และเวกเตอร์ b ของสมการเมทริกซ์ของวงจรไฟฟ้า รวมถึงการคำนวณเชิงเส้นในอุปกรณ์ที่มีการทำงานไม่เชิงเส้น

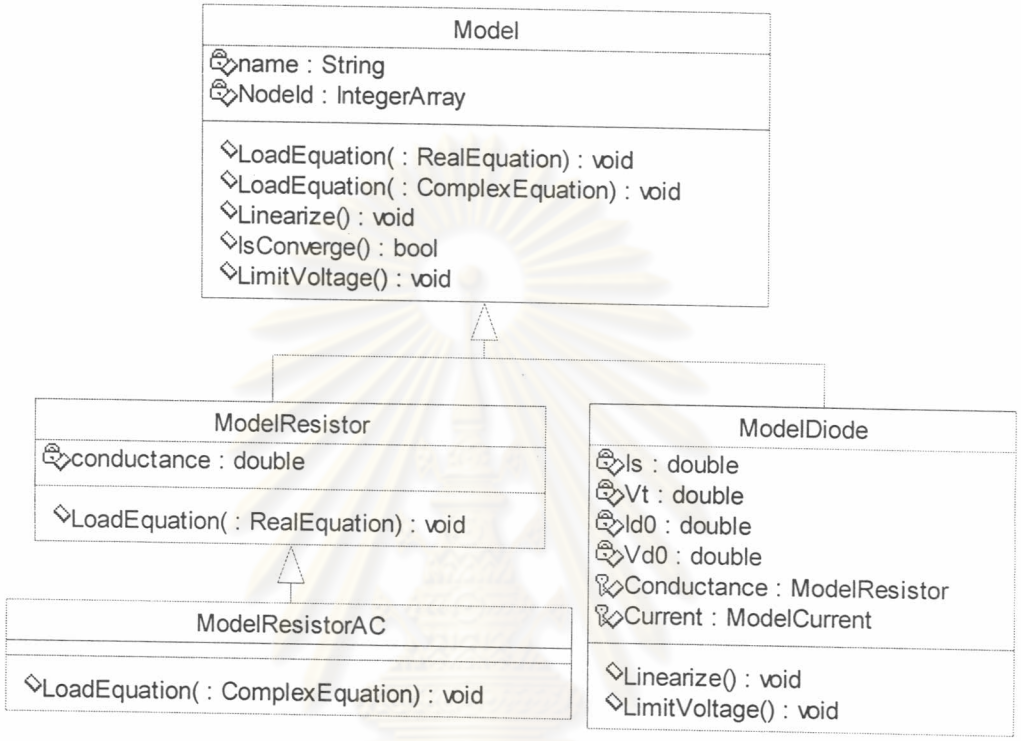


รูปที่ 6.12 คลาส SimulatorAC



รูปที่ 6.13 โครงสร้างคลาส Simulator

คลาสสำหรับคำนวณการทำงานของอุปกรณ์ไฟฟ้าแต่ละตัวจะสืบทอดคลาสมาจากคลาส Model โดยปรับตราประจำอุปกรณ์ให้ตรงกับตราประจำอุปกรณ์ไฟฟ้าที่ต้องการ คลาสของอุปกรณ์ไฟฟ้าที่มีการคำนวณเชิงเส้นก็เพียงแต่นิยามการคำนวณเชิงเส้นใหม่ ตัวอย่างของโครงสร้างคลาส Model แสดงได้ดังรูปที่ 6.14



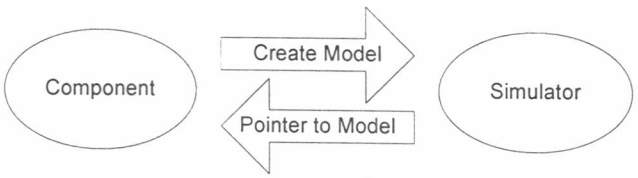
รูปที่ 6.14 ตัวอย่างโครงสร้างคลาส Model

รูปที่ 6.14 เป็นการแสดงโครงสร้างคลาสบางคลาสที่ได้เสนอไว้ในบทที่ 5 โดยแสดงโครงสร้างที่ได้เมื่อสืบทอดมาจากคลาส Model

6.2.8 การสร้างวัตถุจากคลาส Model

เนื่องจากแผนภาพวงจรหนึ่งสามารถเลือกทำการวิเคราะห์ได้หลายแบบ ดังนั้นเพื่อเป็นการลดขั้นตอนในการเลือกสร้างวัตถุ เราจะใช้คลาส Simulator เป็นคลาสที่ควบคุมในการเลือกวัตถุจากคลาส Model ให้ตรงกับความต้องการ ยกตัวอย่างเช่น แหล่งจ่ายแรงดันแบบพัลส์ ในการวิเคราะห์หาจุดทำงานสงบ จะใช้การคำนวณของแหล่งจ่ายแรงดันตรง แต่ถ้าเป็นการวิเคราะห์หาผลตอบทางเวลา จะใช้การคำนวณของแหล่งจ่ายแรงดันแบบพัลส์ หรืออีกตัวอย่างหนึ่งเช่น ตัวเก็บประจุ ซึ่งในการวิเคราะห์หาจุดทำงานสงบจะเปรียบเสมือนวงจรเปิดซึ่งเป็นวัตถุจากคลาส Capacitor แต่ในการวิเคราะห์หาผลตอบทางเวลาจะต้องใช้ขั้นตอนวิธีของออยเลอร์ซึ่งเป็นวัตถุ

จากคลาส CapacitorTR หรือในการวิเคราะห์หาผลตอบสนองเชิงความถี่จะใช้วัตถุจากคลาส CapacitorAC รูปแบบการติดต่อในการสร้างวัตถุจากคลาส Model แสดงได้ดังรูปที่ 6.15



รูปที่ 6.15 การติดต่อระหว่างคลาส Component และ Simulator ในการสร้างวัตถุจากคลาส Model

รูปที่ 6.15 แสดงให้เห็นถึงการติดต่อระหว่างวัตถุจากคลาส Component และวัตถุจากคลาส Simulator โดยที่วัตถุจากคลาส Component จะส่งคำร้องไปยังวัตถุจากคลาส Simulator เพื่อให้สร้างวัตถุจากคลาส Model ที่ตรงกับการวิเคราะห์ จากนั้นวัตถุจากคลาส Simulator จะส่งตัวชี้ไปยังวัตถุจากคลาส Model กลับมาเพื่อให้ทำการปรับค่าพารามิเตอร์ให้ตรงกับความต้องการ

ตัวอย่างรหัสในการสร้างวัตถุจากคลาส ModelCapacitor ในคลาส Simulator แสดงได้ดังรหัสเทียมดังนี้

```
Model* SimulatorDC::CreateModel(MODEL_TYPE type)
{
      switch(type)
      {
            …
            case CAPACITOR_MODEL:
            /* code to create object from ModelCapacitor */
            …
      }
}
Model* SimulatorTR::CreateModel(MODEL_TYPE type)
{
      switch(type)
      {
            …
            case CAPACITOR_MODEL:
            /* code to create object from ModelCapacitorTR */
            …
      }
}
Model* SimulatorAC::CreateModel(MODEL_TYPE type)
{
      switch(type)
      {
            …
            case CAPACITOR_MODEL:
            /* code to create object from ModelCapacitorAC */
            …
      }
}
```

6.3 สรุปท้ายบท

โปรแกรมวิเคราะห์วงจรรวมที่พัฒนาขึ้นนี้ได้แยกการทำงานของส่วนติดต่อผู้ใช้และส่วนที่ทำการวิเคราะห์ออกจากกัน ข้อดีของโครงสร้างนี้คือทำให้สามารถที่จะพัฒนาโปรแกรมในระบบปฏิบัติการอื่นได้โดยง่าย เนื่องจากส่วนที่ทำการวิเคราะห์นั้นเราใช้ไลบรารี่พื้นฐานของภาษา C++ เท่านั้น ส่วนที่ต้องเพิ่มเติมคือการปรับแต่งส่วนติดต่อผู้ใช้ให้สามารถเข้ากับระบบปฏิบัติการที่ต้องการพัฒนาเพิ่มเติมได้

ในการวิเคราะห์แต่และแบบ หน้าที่ในการเลือกอุปกรณ์แต่ละชนิดจะอยู่ในคลาส Simulator ข้อดีของการออกแบบนี้คือทำให้วงจรหนึ่ง ๆ สามารถคำนวณได้หลายแบบโดยที่ไม่ต้องพัฒนาส่วนหนึ่งส่วนใดเป็นพิเศษ ผู้ใช้จะไม่จำเป็นต้องรู้รายละเอียดการทำงานของส่วนการวิเคราะห์เลยก็สามารถทำการวิเคราะห์ได้ เพียงแต่กำหนดค่าให้กับส่วนการวิเคราะห์ตามปกติเท่านั้น

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย