บทที่ 3

การแปลงและตรวจสอบแผนภาพสถานะ

3.1 แนวคิดในการแปลงและตรวจสอบแผนภาพสถานะ

ในส่วนของเนื้อหางานวิจัยจะครอบคลุมสองส่วน คือ การแปลงแผนภาพสถานะไปเป็นข้อกำหนดแบบรูปนัย และการวิเคราะห์และตรวจสอบความสัมพันธ์ของแผนภาพสถานะจากข้อกำหนดแบบรูปนัยที่ได้จากขั้นตอนการแปลง ดังนี้

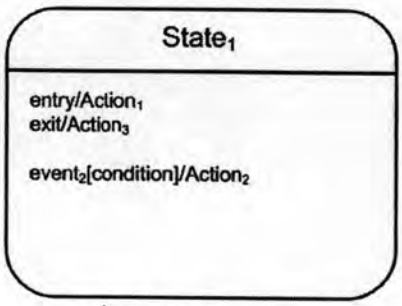
3.1.1 การแปลงแผนภาพสถานะไปเป็นข้อกำหนดรูปนัย

แผนภาพสถานะยูเอ็มแอลนั้นประกอบด้วยองค์ประกอบในการสร้างแผนภาพจำนวนมาก ซึ่งแต่ละองค์ประกอบก็มีความหมายการทำงานในตัวของมันเอง โดยองค์ประกอบแต่ละตัวยังสามารถนำมาเชื่อมต่อกับองค์ประกอบอื่นๆ ได้อีกหลายรูปแบบเพื่อให้เกิดเป็นแผนภาพสถานะ ซึ่งแต่ละแผนภาพจะอธิบายความหมายของพฤติกรรมที่เกิดขึ้นภายในระบบในรูปแบบลำดับของเหตุการณ์ต่างๆ ที่เกิดขึ้น ในลักษณะเดียวกันแต่ละนิพจน์ของแคลคูลัสของกระบวนการก็อธิบายถึงลำดับเหตุการณ์ต่างๆ ที่เกิดขึ้นเช่นกัน จึงมีความเป็นไปได้ในการใช้แคลคูลัสของกระบวนการนี้มาช่วยอธิบายแผนภาพสถานะยูเอ็มแอล ซึ่งในขั้นตอนการแปลงนั้นเราแบ่งกฎการแปลงออกเป็นทั้งหมด 2 ส่วน คือ การแปลงองค์ประกอบแต่ละส่วน และการแปลงรูปแบบการเชื่อมต่อชนิดต่างๆ

1) การแปลงองค์ประกอบแต่ละส่วน

ในเบื้องต้นนั้นองค์ประกอบหลักในการสร้างแผนภาพสถานะ คือ สถานะ และเส้นการเปลี่ยนแปลง

ในส่วนของสถานะนั้นสามารถกำหนดรายละเอียดภายในได้ดังรูปที่ 3.1



รูปที่ 3.1 ตัวอย่างสถานะ

รายละเอียดภายในนั้นจะเป็นการเปลี่ยนแปลงภายใน (internal transition) อยู่ในรูปแบบของ *เหตุการณ์ [เงื่อนไข] /การกระทำ* ซึ่งเมื่อมีเหตุการณ์ที่กำหนดเกิดขึ้นและเงื่อนไขตรงตามที่กำหนดจะทำการกระทำที่ได้กำหนดไว้ แต่สถานะจะยังคงอยู่ในสถานะเดิมไม่เปลี่ยนไปเป็นสถานะอื่น

โดยมีเหตุการณ์พิเศษอยู่สามชนิด คือ *entry*, *exit* และ *do* ซึ่งเหตุการณ์ *entry* จะเกิดเมื่อเกิดการเปลี่ยนสถานะมายังสถานะที่กำหนด *exit* จะเกิดเมื่อสถานะเปลี่ยนไปยังสถานะอื่น และ *do* ซึ่งทำให้เกิดการกระทำที่กำหนดไปเรื่อยๆ จนกว่าสถานะจะเปลี่ยนไปยังสถานะอื่นหรือการกระทำที่กำหนดเสร็จสิ้น

จะเห็นได้ว่าเหตุการณ์ต่างๆ ที่เกิดขึ้นนั้นจะมีลำดับของการเกิดพฤติกรรม ซึ่งหากเราทำการเปรียบเทียบกับแคลคูลัสของกระบวนการนั้นจะพบว่ามีการอธิบายลำดับของสิ่งต่างๆ ที่เกิดขึ้นเช่นกัน โดยเราสามารถกำหนดสถานะแต่ละตัวเป็นกระบวนการ และเหตุการณ์และการกระทำที่เกิดขึ้นเปรียบเสมือนลำดับของอักษรในแคลคูลัสของกระบวนการ จากรูปที่ 3.1 สามารถแทนด้วยนิพจน์ คือ $State_1 = entry \cdot Action_1 \cdot (event_2 \cdot conditionIsTrue \cdot Action_2)^* \cdot exit \cdot Action_3$

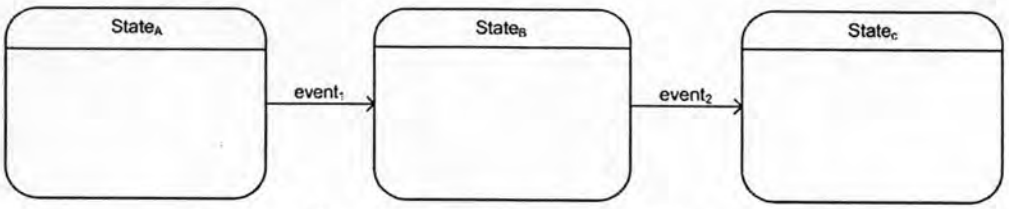
หากสถานะเป็นลักษณะของสถานะประกอบซึ่งมีสถานะภายในเป็นสถานะแบบลำดับก็สามารถเขียนนิพจน์ในลักษณะต่อเนื่องได้เช่นกัน แต่หากสถานะภายในเป็นสถานะแบบทำงานพร้อมกันจะต้องอาศัยตัวดำเนินที่สามารถอธิบายการทำงานแบบพร้อมกันด้วย

รูปที่ 3.2 ตัวอย่างเส้นการเปลี่ยนแปลง

ในส่วนของเส้นการเปลี่ยนแปลงดังรูปที่ 3.2 จะมีคำอธิบายประกอบเส้นการเปลี่ยนแปลงซึ่งอยู่ในรูปแบบ *เหตุการณ์ [เงื่อนไข] /การกระทำ* เช่นเดียวกับการเปลี่ยนแปลงภายในสถานะ โดยจากตัวอย่าง หากสถานะอยู่ที่ $State_A$ และมีเหตุการณ์ *event* เกิดขึ้นแล้วจะมีการพิจารณา *condition* ว่าถูกต้องหรือไม่ หากถูกต้องประมวลผล *Action* และสถานะจะเปลี่ยนจาก $State_A$ ไปเป็น $State_B$ แต่ในกรณีที่ *condition* ไม่ถูกต้อง จะไม่มีการประมวลผล *Action* และสถานะจะยังคงอยู่ที่ $State_A$ โดยไม่มีการเปลี่ยนแปลง จากพฤติกรรมดังกล่าวเราสามารถแทนด้วยนิพจน์คือ $State_A = event \cdot (ConditionIsTrue \cdot Action \cdot State_B + ConditionIsFalse \cdot State_A)$

2) การแปลงรูปแบบการเชื่อมต่อชนิดต่างๆ

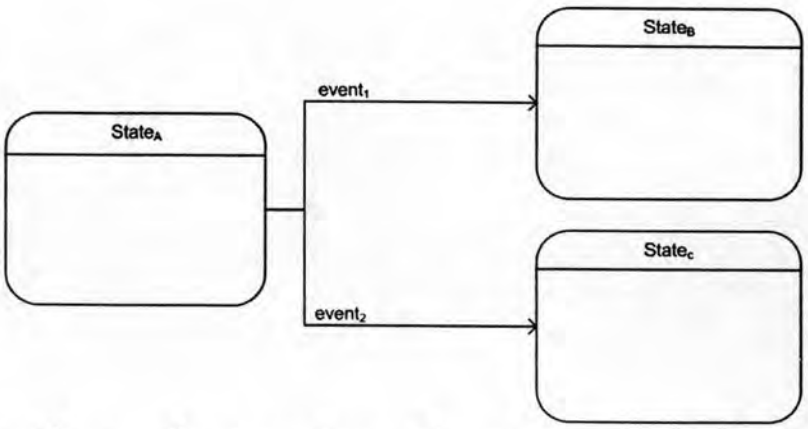
การเชื่อมต่อกันขององค์ประกอบจะแบ่งออกได้สองชนิด คือ การเชื่อมต่อแบบต่อเนื่องกันของเส้นการเปลี่ยนแปลง และการเชื่อมต่อแบบมีเส้นการเปลี่ยนแปลงออกจากสถานะมากกว่าหนึ่งเส้น โดยที่แบบต่อเนื่องเป็นดังรูปที่ 3.3



รูปที่ 3.3 การเชื่อมต่อกันของเส้นการเปลี่ยนแปลงแบบต่อเนื่อง

ลักษณะการเชื่อมต่อกันในแบบดังกล่าวเราสามารถใช้รูปแบบการอธิบายสถานะในหัวข้อ 1) ได้เลย โดยอธิบายสถานะแต่ละตัวด้วยกระบวนการของแคลคูลัสของกระบวนการ ดังนั้นจากตัวอย่างจะได้นิพจน์ดังนี้ $State_A = event_1 \cdot State_B$ และ $State_B = event_2 \cdot State_C$ ซึ่งจะเห็นได้ว่าหากระบบจะเปลี่ยนสถานะจาก $State_A$ ไปยัง $State_C$ ได้นั้นจะต้องเกิดเหตุการณ์ $event_1$ และ $event_2$ ก่อนตามลำดับ ซึ่งเราสามารถอธิบายพฤติกรรมขององค์ประกอบที่เชื่อมต่อกันในลักษณะนี้ได้เป็น $State_A = event_1 \cdot event_2 \cdot State_C$ ซึ่งหาพิจารณาเฉพาะพฤติกรรมของระบบจะพบว่าระบบนี้สามารถเกิดพฤติกรรมได้เพียงรูปแบบเดียวคือ $event_1 \cdot event_2$

ลักษณะการเชื่อมต่ออีกแบบ คือ มีเส้นการเปลี่ยนแปลงมากกว่าหนึ่งเส้นออกจากสถานะ ดังรูปที่ 3.4



รูปที่ 3.4 การเชื่อมต่อแบบเส้นการเปลี่ยนแปลงออกมากกว่าหนึ่งเส้น

ลักษณะการเชื่อมต่อกันแบบนี้จะเห็นได้ว่าสถานะสามารถเปลี่ยนจาก $State_A$ ไปเป็นสถานะ $State_B$ หรือสถานะ $State_C$ ก็ได้โดยขึ้นอยู่กับเหตุการณ์ที่เกิดขึ้น ในลักษณะนี้เราอธิบายโดยใช้ตัวดำเนินการทางเลือกช่วย คือ $State_A = event_1 \cdot State_B + event_2 \cdot State_C$ ซึ่งหากพิจารณาเฉพาะพฤติกรรมของระบบจะพบว่าระบบนี้สามารถเกิดพฤติกรรมได้สองรูปแบบ คือ $event_1$ หรือ $event_2$

3.1.2 การตรวจสอบความสัมพันธ์ของแผนภาพสถานะ

ในงานวิจัยนี้ได้ทำการตรวจสอบความสัมพันธ์ของแผนภาพสถานะสองรูปแบบ คือ

1) การตรวจสอบความเท่ากันของพฤติกรรมของวัตถุ โดยเมื่อทำการแปลงแผนภาพสถานะที่ต้องการตรวจสอบให้อยู่ในรูปของแคลคูลัสของกระบวนการแล้ว เราจะทำการเปรียบเทียบรูปแบบพฤติกรรมในจุดที่สนใจของแต่ละแผนภาพว่าถูกต้องตรงกันครบหรือไม่ โดยละเว้นพฤติกรรมย่อยหรือพฤติกรรมเสริมอื่นๆ ภายในระบบ

2) การตรวจสอบพฤติกรรมของวัตถุในการทำงานร่วมกับวัตถุอื่นภายในระบบ โดยวิเคราะห์พฤติกรรมของวัตถุที่เราได้ทำการออกแบบไว้แบบเดี่ยวเปรียบเทียบกับพฤติกรรมของวัตถุเมื่อทำงานร่วมกันกับวัตถุอื่นภายในระบบ ซึ่งวิธีการตรวจสอบจะคล้ายๆ กับข้อ 1) คือ เปรียบเทียบพฤติกรรมที่เป็นไปได้ทั้งหมดของวัตถุว่าเกิดขึ้นได้ทั้งหมดภายในระบบหรือไม่

3.2 การแปลงแผนภาพสถานะไปเป็นซีอาร์อี

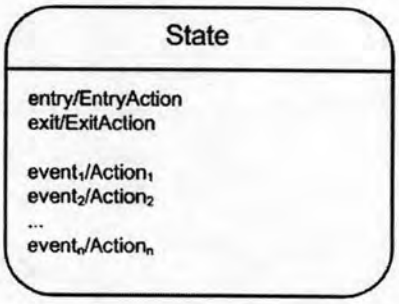
การแปลงพฤติกรรมต่างๆ ของระบบไปเป็นซีอาร์อีนั้น เรากำหนดให้ เหตุการณ์ เงื่อนไข และการกระทำ ที่เกิดขึ้นภายในระบบเปรียบเสมือนตัวอักษรของซีอาร์อี ซึ่งหากพฤติกรรมใดๆ มีลำดับของการเกิดขึ้นที่ต่อเนื่องกัน เราสามารถอธิบายพฤติกรรมนั้นได้ด้วยนิพจน์ของซีอาร์อีในลักษณะการเขียนพฤติกรรมนั้นๆ ต่อเนื่องกัน โดยเชื่อมระหว่างพฤติกรรมสองพฤติกรรมนั้นด้วยสัญลักษณ์ "ตัวดำเนินการเชื่อมต่อ" เช่น ในระบบหนึ่งๆ หากมีเหตุการณ์ *event* เกิดขึ้นแล้วจะต้องมีการกระทำ *action* เกิดขึ้นเพื่อตอบสนองต่อเหตุการณ์นั้น ดังนั้นลำดับพฤติกรรมของระบบที่เกิดขึ้น คือ *event* และ *action* ตามลำดับ ซึ่งเราทำการแทนพฤติกรรมลักษณะนี้ด้วยนิพจน์ของซีอาร์อี คือ $event \cdot action$ แต่หากในกรณีที่ระบบมีพฤติกรรมในลักษณะทางเลือก เราสามารถอธิบายพฤติกรรมนั้นได้ด้วยนิพจน์ของซีอาร์อีในลักษณะการเชื่อมต่อพฤติกรรมนั้นๆ ด้วยตัวดำเนินการทางเลือก เช่น ในระบบหนึ่งๆ ณ เวลาหนึ่ง หากมีเหตุการณ์ที่สามารถเกิดขึ้นกับระบบได้สองเหตุการณ์ คือ $event_A$ และ $event_B$ จะต้องมีการกระทำเพื่อตอบสนองเหตุการณ์

นั้นๆ คือ $action_A$ และ $action_B$ ตามลำดับ ดังนั้นเราจะได้ลำดับของพฤติกรรมที่สามารถเป็นไปได้สองแบบ คือ $event_A \cdot action_A$ หรือ $event_B \cdot action_B$ โดยที่พฤติกรรมของระบบจะเป็นแบบใดนั้นจะขึ้นอยู่กับเหตุการณ์ที่เกิดขึ้น ในลักษณะนี้เราแทนด้วยนิพจน์ที่ใช้ตัวดำเนินการทางเลือกเป็นตัวเชื่อมคือ $event_A \cdot action_A + event_B \cdot action_B$

จากที่ได้อธิบายให้เห็นข้างต้นเป็นเพียงพื้นฐานของการแปลงเท่านั้น แต่องค์ประกอบต่างๆ ที่ใช้สร้างแผนภาพสถานะยูเอ็มแอลนั้นมีพฤติกรรมที่ซับซ้อนทำให้ต้องสร้างกฎในการอธิบายเพื่อให้ครอบคลุมต่อพฤติกรรมต่างๆ ที่สามารถเกิดขึ้นได้ ซึ่งหัวข้อข้างล่างจะเป็นการนำเสนอกฎในการแปลงสำหรับองค์ประกอบต่างๆ ของแผนภาพ

3.2.1 องค์ประกอบต่างๆ ในการสร้างแผนภาพ

1) สถานะทั่วไป (simple state)



รูปที่ 3.5 สถานะทั่วไป

จากสถานะทั่วไปดังรูปที่ 3.5 เมื่อระบบเปลี่ยนแปลงมาสู่สถานะ *State* จะทำให้เกิดเหตุการณ์ *entry* และเกิดการกระทำ *EntryAction* ตามลำดับ ขณะที่ระบบคงอยู่ในสถานะ *State* นี้ หากมีเหตุการณ์ *ie* ใดๆ เกิดขึ้นภายใน ก็จะเกิดการกระทำ $Action_{ie}$ ควบคู่กันไปตามลำดับ ซึ่งขั้นตอนของเหตุการณ์และการกระทำนี้สามารถเกิดขึ้นซ้ำๆ ได้ จนกระทั่งระบบต้องการเปลี่ยนสถานะไปเป็นสถานะอื่น จะทำให้เกิดเหตุการณ์ *exit* และการกระทำ *ExitAction* ตามลำดับ ซึ่งจากพฤติกรรมดังกล่าว เราแทนด้วยนิพจน์ ดังนี้

$$entry \cdot EntryAction \cdot (\sum_{ie \in InternalEvent} ie \cdot Action_{ie})^* \cdot Exit \cdot Action$$

โดยที่

entry คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนสถานะ

EntryAction คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ *entry*

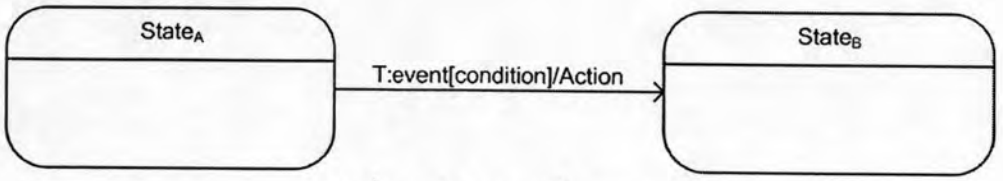
ie คือ เหตุการณ์ที่เกิดขึ้นภายในสถานะ ซึ่งไม่มีผลทำให้สถานะของระบบเปลี่ยนไป

$Action_{ie}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ภายใน *ie*

exit คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนไปยังสถานะอื่น

ExitAction คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ *exit*

2) เส้นการเปลี่ยนแปลง (transition)



รูปที่ 3.6 เส้นการเปลี่ยนแปลง

จากเส้นการเปลี่ยนแปลงดังรูปที่ 3.6 เมื่อเกิดเหตุการณ์กระตุ้นขึ้นเพื่อให้ระบบเปลี่ยนสถานะ จะต้องพิจารณาเงื่อนไขภายในก่อน หากเงื่อนไขถูกต้องจะมีการกระทำการกระทำที่กำหนดไว้ และระบบเปลี่ยนไปยังสถานะที่ต้องการ ในส่วนของเงื่อนไขนั้น เราเลือกใช้สัญลักษณ์ฟังก์ชัน $g()$ แทนการตรวจสอบความถูกต้องของเงื่อนไข โดยหากเงื่อนไขไม่ถูกต้องเส้นการเปลี่ยนแปลงนั้นจะไม่ถูกกระทำ และระบบจะไม่มีการเปลี่ยนแปลงไปยังสถานะปลายทาง ซึ่งเราสามารถแทนด้วยนิพจน์ ดังนี้

$$event \cdot g(condition) \cdot Action$$

โดยที่

event คือ เหตุการณ์ของเส้นการเปลี่ยนแปลง ซึ่งเป็นตัวกระตุ้นให้ระบบเปลี่ยนสถานะ

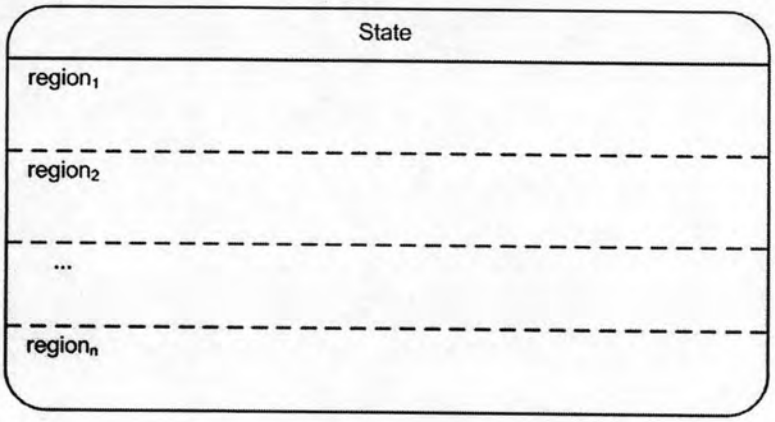
condition คือ เงื่อนไขของเส้นการเปลี่ยนแปลง ซึ่งเป็นตัวพิจารณาการเปลี่ยนแปลงสถานะว่ายอมให้เกิดขึ้นได้หรือไม่

$g()$ คือ การฟังก์ชันตรวจสอบเงื่อนไข ซึ่งหากผลที่ได้จากการประมวลผลเงื่อนไขภายในเป็นจริงจะอนุญาตให้มีการดำเนินการคำสั่งต่อไป

Action คือ การกระทำที่ตอบสนองเมื่อเกิดการเปลี่ยนแปลงสถานะ

3) สถานะประกอบ (composite state)

สถานะประกอบ จะมีสองรูปแบบ คือ สถานะประกอบที่มีสถานะย่อยแบบทำงานพร้อมกัน และสถานะประกอบที่มีสถานะย่อยแบบต่อเนื่อง ซึ่งสถานะประกอบที่มีสถานะย่อยแบบทำงานพร้อมกันนั้น จะประกอบด้วยพื้นที่ย่อยภายในสองพื้นที่ขึ้นไป ซึ่งมีการทำงานพร้อมกัน ส่วนสถานะประกอบที่มีสถานะย่อยแบบต่อเนื่องจะเปรียบเสมือนสถานะประกอบที่มีพื้นที่ย่อยภายในแค่เพียงพื้นที่เดียว ซึ่งการอธิบายจะใช้วิธีการเหมือนการอธิบายแผนภาพทั่วไป



รูปที่ 3.7 สถานะย่อยแบบทำงานพร้อมกัน

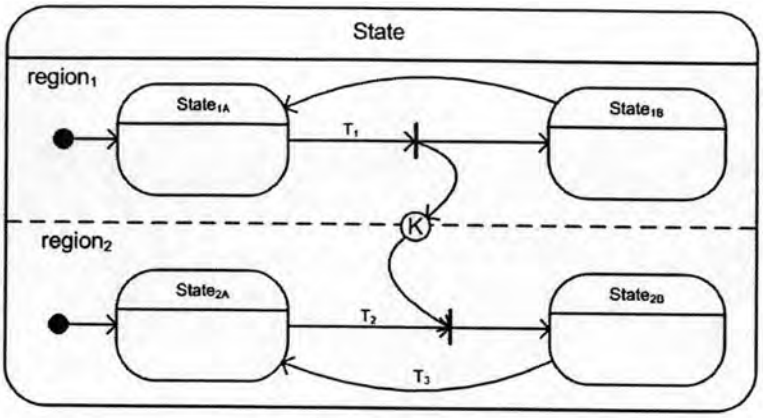
ในส่วนของสถานะย่อยแบบทำงานพร้อมกันดังรูปที่ 3.7 นั้น จะประกอบด้วยพื้นที่ย่อยๆ ภายในสถานะประกอบที่มีการทำงานในลักษณะพร้อมกัน โดยภายในพื้นที่ย่อยๆ ดังกล่าวจะประกอบไปด้วยสถานะที่เชื่อมต่อกันในรูปแบบต่างๆ เช่นกัน ซึ่งจากรูปเราสามารถแทนด้วยนิพจน์ดังนี้

$$region_1[]region_2[]...[]region_n$$

โดยที่

$region_i$ คือ พื้นที่ย่อยภายในสถานะประกอบ ซึ่งภายในจะประกอบด้วยองค์ประกอบต่างๆ เชื่อมต่อกันเสมือนแผนภาพย่อย ซึ่งสามารถอธิบายด้วยกฎวิธีการแปลงทั่วไป

4) สถานะซิง (synch state)



รูปที่ 3.8 สถานะซิง

จากสถานะซิงในรูปที่ 3.8 นั้น แสดงการพฤติกรรมการทำงานเชื่อมต่อกันระหว่างพื้นที่ย่อยภายในสถานะประกอบ โดยพื้นที่ $region_2$ จะมีการเปลี่ยนแปลงสถานะภายในได้จะขึ้นอยู่กับอีกพื้นที่ $region_1$ ซึ่งใน $region_2$ นั้น หากระบบต้องการเปลี่ยนสถานะจาก $State_{2A}$ ไปเป็นสถานะ $State_{2B}$ ที่สถานะซิงจะต้องมีโทเคนภายใน ซึ่งจำนวนโทเคนภายในจะเกิดขึ้นได้จากการเปลี่ยนแปลงของสถานะใน $region_1$ จาก $State_{1A}$ ไปยัง $State_{1B}$ ดังนั้นในการอธิบายจึงใช้รูปแบบของพื้นที่ย่อยๆ ที่มีความสัมพันธ์กันทำงานพร้อมกัน และมีการระบุข้อกำหนดของการเปลี่ยนแปลงสถานะสำหรับช่วงการเชื่อมต่อของสถานะซิง ดังนี้

$$region_1[]region_2[]constraint$$

โดยที่

$region_1$ คือ พื้นที่ย่อยภายในสถานะประกอบเกิดการวนรอบของเส้นเปลี่ยนแปลง T_1 ซึ่งสามารถอธิบายด้วยนิพจน์ คือ T_1^*

$region_2$ คือ พื้นที่ย่อยภายในสถานะประกอบเกิดการวนรอบของเส้นการแปลี่ยนแปลง T_2 และ T_3 ซึ่งสามารถอธิบายด้วยนิพจน์เช่นกัน คือ $(T_2 \cdot T_3)^*$

constraint คือ ข้อกำหนดที่ทำให้พื้นที่ย่อยสองส่วนทำงานสัมพันธ์กันอันเนื่องมาจากสถานะซิง นิพจน์ที่ได้จะขึ้นอยู่กับค่า K ที่กำหนดภายในสถานะซิง ในที่นี้แบ่งได้สองรูปแบบ คือ

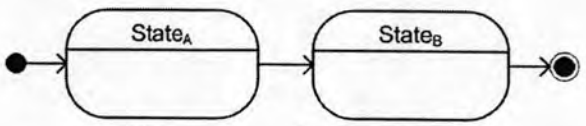
- ค่า K เป็นจำนวนเต็มบวกใดๆ

$$((T_1 \cdot T_1^* \cdot T_2 \cdot T_3)^*)^{(k)} \| (T_2 \cdot T_1 \cdot T_1^* \cdot T_3)^*$$

- ค่า K เป็น *

$$((T_1 \cdot T_1^* \cdot T_2 \cdot T_3)^*)^\alpha \parallel (T_2 \cdot T_1 \cdot T_1^* \cdot T_3)^*$$

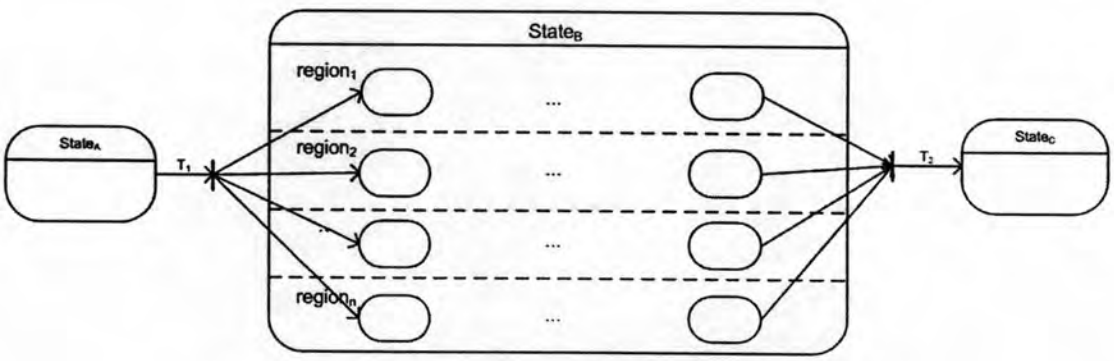
5) สถานะเริ่มต้นและสิ้นสุด (initial & final pseudostate)



รูปที่ 3.9 สถานะเริ่มต้นและสิ้นสุด

สถานะเริ่มและสถานะสิ้นสุดดังรูปที่ 3.9 ไม่มีการกระทำหรือเหตุการณ์ใดๆ ภายใน และเป็นเพียงสัญลักษณ์บอกจุดเริ่มต้นและสิ้นสุดของสถานะเท่านั้น ดังนั้นในที่นี้จึงไม่มีการอธิบายพฤติกรรมของสถานะดังกล่าวด้วยนิพจน์ใดๆ

6) สถานะแยกและเชื่อมสำหรับการทำงานพร้อมกัน (fork & join pseudostate)



รูปที่ 3.10 สถานะแยกและเชื่อม

จากรูปที่ 3.10 ซึ่งจำลองการทำงานของสถานะแยกและเชื่อม ซึ่งลักษณะการทำงานของระบบที่เชื่อมต่อจากสถานะแยกนั้นจะมีลักษณะเกิดขึ้นพร้อมกัน คือ มีสถานะที่ปัจจุบันได้มากกว่า 1 แต่หลังจากการทำงานผ่านสถานะเชื่อมแล้ว สถานะปัจจุบันที่มากกว่า 1 จะถูกรวมเหลือเพียงหนึ่ง เราสามารถอธิบายได้ด้วยนิพจน์ ดังนี้

$$T_1 \cdot (region_1 [] region_2 [] ... [] region_n) \cdot T_2$$

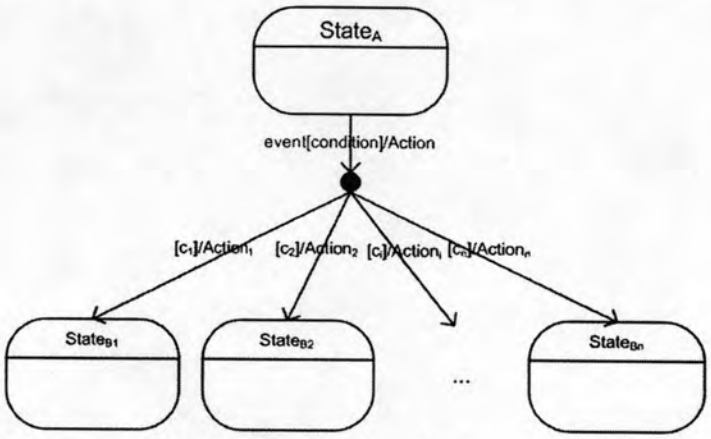
โดยที่

region คือ พื้นที่ย่อยภายในที่เชื่อมต่อจากสถานะแยก

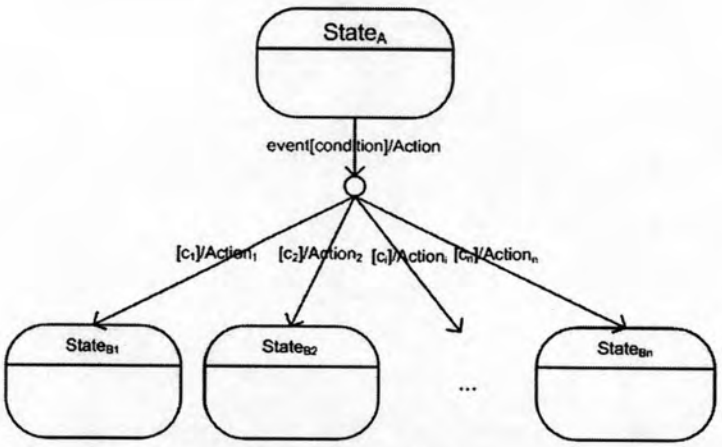
T_1 คือ เส้นการเปลี่ยนแปลงไปยังสถานะแยก

T_2 คือ เส้นการเปลี่ยนแปลงออกจากสถานะเชื่อม

7) สถานะตัวต่อและตัวเลือก (junction & choice point)



รูปที่ 3.11 สถานะตัวต่อ



รูปที่ 3.12 สถานะตัวเลือก

จากรูปที่ 3.11 และ 3.12 ของสถานะตัวต่อและสถานะตัวเลือก การทำงานจะมีลักษณะคล้ายคลึงกัน โดยความแตกต่างคือเงื่อนไขของเส้นการเปลี่ยนแปลงหลังจากสถานะตัวเลือกจะถูกพิจารณาหลังจากเส้นการเปลี่ยนจากสถานะต้นทางมายังสถานะเลือกซึ่งเงื่อนไขอาจมีผลกระทบจากการประมวลผลเส้นการเปลี่ยนแปลงดังกล่าว แต่ในส่วนของซีอาร์อีแล้วการกระทำต่างๆ จะเป็นผลต่อเนื่องจากเหตุการณ์ก่อนหน้าอยู่แล้ว ดังนั้นการอธิบายสถานะตัวต่อและตัวเลือกทั้งสองแบบนี้จึงไม่มีความแตกต่าง โดยสามารถใช้นิพจน์อ้างอิงได้ดังนี้

$$T_1 \cdot (\sum_{i=1}^{n} g(c_i) \cdot Action_i)$$

โดยที่

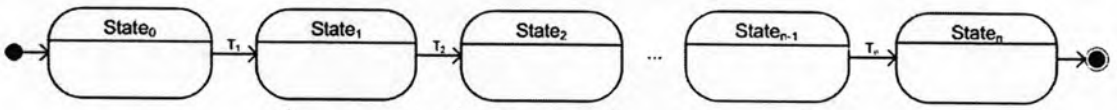
T_i คือ นิพจน์แสดงเส้นเปลี่ยนแปลงจากสถานะต้นทางไปยังสถานะตัวต่อหรือตัวเลือก

c_i คือ เงื่อนไขของเส้นเปลี่ยนแปลงแต่ละเส้นซึ่งเชื่อมจากสถานะตัวต่อหรือตัวเลือกไปยังสถานะปลายทาง

$Action_i$ คือ การกระทำของเส้นเปลี่ยนแปลงแต่ละเส้นซึ่งเชื่อมจากสถานะตัวต่อหรือตัวเลือกไปยังสถานะปลายทาง

3.2.2 การเชื่อมต่อกันระหว่างองค์ประกอบประเภทต่างๆ

1) สถานะที่มีเส้นการเปลี่ยนแปลงออกหนึ่งเส้น



รูปที่ 3.13 สถานะที่มีเส้นการเปลี่ยนแปลงออกหนึ่งเส้น

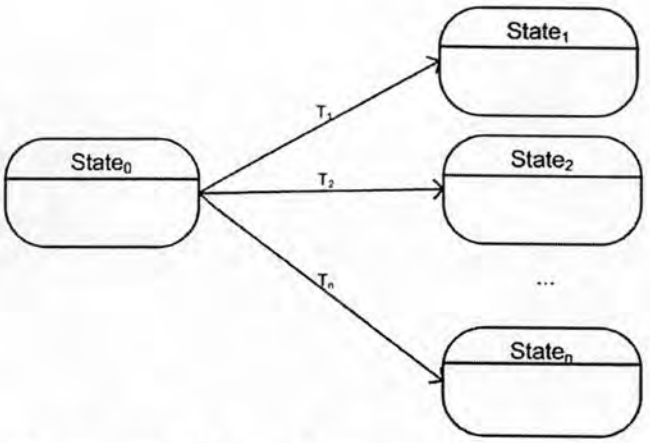
จากรูปที่ 3.13 หากมีเส้นการเปลี่ยนแปลงออกจากสถานะเพียงหนึ่งเส้นเราสามารถเขียนบรรยายระบบได้ด้วย T ซึ่งแทนนิพจน์ของสถานะดังกล่าว และหากมีการเชื่อมต่อกันของสถานะในลักษณะของลำดับ การบรรยายพฤติกรรมของระบบจะสามารถบรรยายในลักษณะของเหตุการณ์ที่เกิดต่อเนื่องกัน คือ $State_0 \cdot T_1 \cdot State_1 \cdot T_2 \cdot State_2 \cdot \ldots \cdot T_n State_n$ ซึ่งในกรณีที่สถานะต่างๆ ไม่มีพฤติกรรมภายใน เราสามารถละสัญลักษณ์ของแต่ละสถานะได้ โดยแทนด้วยนิพจน์ดังนี้

$$T_1 \cdot T_2 \cdot \ldots \cdot T_n$$

โดยที่

T_i คือ นิพจน์ของเส้นเปลี่ยนแปลงแต่ละเส้นที่เชื่อมต่อกัน

2) สถานะที่มีเส้นการเปลี่ยนแปลงออกมากกว่าหนึ่งเส้น



รูปที่ 3.14 สถานะที่มีเส้นเปลี่ยนแปลงออกมากกว่าหนึ่งเส้น

หากสถานะมีเส้นการเปลี่ยนแปลงออกจากสถานะมากกว่าหนึ่งเส้นแล้ว เราสามารถแทนได้ด้วยนิพจน์ $T_1 \cdot State_1 + T_2 \cdot State_2 + ... + T_n State_n$ ซึ่งในกรณีที่สถานะปลายทางต่างๆ ไม่มีเหตุการณ์ภายใน เราสามารถละสัญลักษณ์ของแต่ละสถานะได้ โดยแทนด้วยนิพจน์ ดังนี้

$$\sum_{i=1}^{n} T_i$$

โดยที่

T_i คือ นิพจน์ของเส้นเปลี่ยนแปลงแต่ละเส้นที่ออกจากสถานะต้นทาง

3.3 การแปลงแผนภาพสถานะไปเป็นไพแคลคูลัส

จากส่วนการแปลงแผนภาพสถานะไปเป็นซีอาร์อีนั้น พบว่าการแปลงให้ครอบคลุมส่วนประกอบต่างๆ ทั้งหมดในแผนภาพสถานะยูเอ็มแอลเป็นไปได้ยาก เนื่องจากความซับซ้อนของตัวองค์ประกอบ การเชื่อมต่อองค์ประกอบในลักษณะต่างๆ และข้อจำกัดในการอธิบายของตัวภาษาเอง ดังนั้นจึงได้นำเสนอไพแคลคูลัสขึ้นมาเพื่อเป็นภาษาปลายทาง ซึ่งรูปแบบการแปลงจะมีความคล้ายคลึงกับซีอาร์อี คือ มีการแปลงองค์ประกอบและพิจารณารูปแบบการเชื่อมต่อกัน แต่ได้เปลี่ยนแนวคิดของการแปลงใหม่ในส่วนของการเชื่อมต่อขององค์ประกอบ โดยจากเดิมนั้นจะสังเกตจากรูปแบบการเชื่อมต่อของส่วนประกอบต่างๆ ว่าเชื่อมต่อกันโดยลักษณะใด และทำการแทนการเชื่อมต่อนั้นโดยใช้สัญลักษณ์และนิพจน์ที่เหมาะสม แต่ในรูปแบบใหม่จะมองในลักษณะส่วนประกอบต่างๆ ในระดับเดียวกันมีการทำงานในลักษณะพร้อมกันทั้งหมด และองค์ประกอบแต่ละตัวมีสถานะเป็นสถานะไม่ทำงานและสถานะทำงาน โดยที่ในแผนภาพหนึ่งในระยะเวลาหนึ่งๆ

ที่ระดับเดียวกัน จะมีองค์ประกอบที่อยู่ในสถานะทำงานอยู่เพียงหนึ่งกลุ่ม คือ สถานะเพียงหนึ่งสถานะและเส้นการเปลี่ยนที่เชื่อมออกจากสถานะนั้น ซึ่งการเปลี่ยนสถานะขององค์ประกอบแต่ละส่วนจากสถานะไม่ทำงานไปสู่สถานะทำงานนั้น จะเกิดจากการส่งสัญญาณจากองค์ประกอบที่เชื่อมต่อกันภายในระบบ โดยไม่ได้เกิดจากสัญญาณภายนอก เว้นแต่สถานะเริ่มต้นในแผนภาพซึ่งจะต้องถูกกำหนดให้เป็นสถานะทำงานจากสัญญาณภายนอกเมื่อระบบเริ่มต้นทำงาน

เนื้อหาในหัวข้อนี้จะขอนำเสนอกฎการแปลงแผนภาพสถานะยูเอ็มแอลไปเป็นไพแคลคูลัส โดยแบ่งการแปลงออกเป็นสองส่วน คือ การแปลงองค์ประกอบต่างๆ ในการสร้างแผนภาพ และการแปลงลักษณะการเชื่อมต่อขององค์ประกอบในแบบต่างๆ ซึ่งนิพจน์ที่ได้จะมีความซับซ้อนสูง ดังนั้นผู้เขียนจึงนำเสนอกรณีศึกษาในบทถัดไปซึ่งประกอบไปด้วยขั้นตอนการแปลงอย่างละเอียด เพื่อให้ผู้อ่านสามารถทำความเข้าใจกฎต่างๆ และขั้นตอนการแปลงได้ง่ายยิ่งขึ้น

3.3.1 องค์ประกอบต่างๆ ในการสร้างแผนภาพ

1) สถานะทั่วไป (simple state)

ในลักษณะของการแปลง กำหนดให้เมื่อระบบเริ่มต้น สถานะที่ต้องการแปลงจะมีค่าเป็นไม่ทำงาน และเมื่อระบบเปลี่ยนสถานะมายังสถานะดังกล่าว จะเกิดเหตุการณ์ $\overline{entry_S}$ และการกระทำที่ตอบสนองต่อเหตุการณ์นั้น ซึ่งจะมีการส่งสัญญาณ $\overline{setActiveState}\langle entry_S \rangle$ ไปแจ้งถึงสถานะปัจจุบันด้วยเพื่อช่วยให้สถานะประวัติสามารถบันทึกสถานะปัจจุบันของระบบไว้ได้ จากนั้นจะมีการส่งสัญญาณไปให้เส้นการเปลี่ยนแปลงที่เชื่อมออกจากสถานะอยู่ในสถานะทำงานเพื่อรอตอบสนองต่อเหตุการณ์ต่างๆ ที่เกิดขึ้น ซึ่งสถานะที่รอการตอบสนองนี้จะเรียกว่าสถานะทำงาน โดยหากระบบได้รับสัญญาณเพื่อจะเปลี่ยนสถานะไปยังสถานะอื่น จะมีการส่งสัญญาณ $\overline{exit}$ มายังสถานะนี้เพื่อให้สถานะนี้กลับไปสู่สถานะไม่ทำงานเช่นเดิม ซึ่งเมื่อสถานะประมวลผลการกระทำอันสืบเนื่องจากสัญญาณ $\overline{exit}$ ที่ได้รับแล้ว จะทำการส่งสัญญาณ $\overline{exitComplete}$ กลับไปแจ้งว่าได้เปลี่ยนสถานะของสถานะเรียบร้อยแล้ว จากแนวคิดดังกล่าวเราสามารถเขียนอธิบายได้ด้วยนิพจน์ดังนี้

$$S = entry_S \cdot \overline{setActiveState}\langle entry_S \rangle \cdot \overline{entryAction} \cdot \overline{activeT_S} \cdot S'$$

$$S' = \sum_{i=1}^{n} (ie_i \cdot \overline{a_i} \cdot S') + exit \cdot \overline{exitAction} \cdot \overline{exitComplete} \cdot S$$

โดยที่

S คือ สถานะทั่วไปที่ไม่ใช่สถานะทำงานซึ่งจะไม่ตอบสนองต่อเหตุการณ์ต่างๆ ที่เกิดขึ้นภายในระบบ

S' คือ สถานะทั่วไปซึ่งเป็นสถานะทำงานและคอยตอบสนองต่อเหตุการณ์ต่างๆ ที่เกิดขึ้นภายในระบบ

$entry_S$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะ S

$\overline{setActiveState\langle entry_S \rangle}$ คือ การส่งสัญญาณแจ้งสถานะปัจจุบันของระบบ

$\overline{entryAction}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ $entry$ เข้าสู่สถานะ

$\overline{activeT_S}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะนี้ทำงาน

ie_i คือ เหตุการณ์ที่เกิดขึ้นภายในสถานะ ซึ่งไม่มีผลทำให้สถานะของระบบเปลี่ยนไป

$\overline{a_i}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ภายใน ie_i

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะปัจจุบัน

$\overline{exitAction}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ $exit$ ออกจากสถานะ

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

2) เส้นการเปลี่ยนแปลง (transition)

การบรรยายเส้นการเปลี่ยนแปลงนั้น ในแนวคิดของการแปลงนี้จะใช้นิพจน์กลุ่มเดียวอธิบายเส้นการเปลี่ยนแปลงทั้งหมดที่ออกจากสถานะเดียวกัน โดยไม่ได้อธิบายแบ่งแยกจากกันในแต่ละเส้น โดยกลุ่มของเส้นการเปลี่ยนแปลงจะมีสถานะทำงานและไม่ทำงานเช่นกัน โดยเริ่มแรกกลุ่มของเส้นการเปลี่ยนแปลงดังกล่าวจะอยู่ในสถานะไม่ทำงาน แต่เมื่อระบบเปลี่ยนสถานะมาสู่สถานะที่กลุ่มของเส้นการเปลี่ยนแปลงนี้เชื่อมต่ออยู่ จะมีการส่งสัญญาณ $\overline{activeT_S}$ เพื่อกระตุ้นให้เส้นการเปลี่ยนแปลงเปลี่ยนไปสู่สถานะทำงาน และรอตอบรับต่อเหตุการณ์ต่างๆ ที่เกิดขึ้น หากมีเหตุการณ์ที่เกิดขึ้นและระบบต้องเปลี่ยนสถานะไปยังสถานะอื่นแล้ว เส้นการเปลี่ยนแปลงก็จะถูกเปลี่ยนกลับไปยังสถานะไม่ทำงาน พร้อมทั้งส่งสัญญาณ $\overline{exit}$ ไปยังสถานะปัจจุบันเพื่อให้สถานะดังกล่าวเปลี่ยนกลับไปสู่สถานะไม่ทำงานเช่นกัน แต่ในกรณีที่เส้นการเปลี่ยนแปลงซึ่งมีสถานะทำงานและอยู่ในสถานะประกอบ แล้วเกิดเหตุการณ์ที่สถานะปัจจุบันต้องเปลี่ยนจากสถานะประกอบไปเป็นสถานะอื่น เส้นการเปลี่ยนแปลงจึงต้องรองรับเหตุการณ์เพื่อเปลี่ยนสถานะไปเป็นสถานะไม่ทำงานด้วยเมื่อเกิดเหตุการณ์ $inActiveT$ จากแนวคิดดังกล่าวเราสามารถเขียนอธิบายได้ด้วยนิพจน์ดังนี้

$$T_S = activeT_S \cdot T'_S$$

$$T'_S = \sum_{i=1}^{n}(e_i \cdot ([c_i == True] \cdot \overline{a_i} \cdot \overline{exit} \cdot exitComplete \cdot \overline{entry_{T_S}} \cdot T_S + [c_i == False] \cdot T'_S))$$
$$+ inActiveT \cdot \overline{exit} \cdot exitComplete \cdot T_S$$

โดยที่

T_S คือ กลุ่มของเส้นการเปลี่ยนแปลงทั่วไปที่เชื่อมต่อกับสถานะ S

T'_S คือ กลุ่มของเส้นการเปลี่ยนที่เชื่อมต่อกับสถานะ S ซึ่งเป็นสถานะปัจจุบัน

$activeT_S$ คือ เหตุการณ์ที่ทำให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะ S ทำงาน

e_i คือ เหตุการณ์ของแต่ละเส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะ S

c_i คือ เงื่อนไขของแต่ละเส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะ S

$\overline{a_i}$ คือ การกระทำของแต่ละเส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะ S

$\overline{exit}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะอื่น

exitComplete คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานปัจจุบันเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{entry_{T_S}}$ คือ การส่งสัญญาณไปยังสถานะปลายทางของแต่ละเส้นการเปลี่ยนแปลงเพื่อให้สถานะนั้นเป็นสถานะทำงาน

inActiveT คือ เหตุการณ์เพื่อแจ้งให้เส้นการเปลี่ยนแปลงเปลี่ยนสถานะเป็นไม่ทำงาน

3) สถานะประกอบ (composite state)

รูปแบบการอธิบายของสถานะประกอบจะคล้ายกับสถานะทั่วไป แต่เมื่อสถานะประกอบเปลี่ยนเป็นสถานะทำงานแล้ว พื้นที่ภายในก็จะต้องเปลี่ยนเป็นสถานะทำงานด้วย และเมื่อเกิดเหตุการณ์ที่ระบบจะเปลี่ยนออกจากสถานะประกอบ ก็จะต้องเกิดเหตุการณ์ $\overline{activeT}$ เกิดขึ้นที่สถานะย่อยภายในด้วย ดังนิพจน์ต่อไปนี้

$$CS = entry_{CS} \cdot \overline{setActiveState}\langle entry_{CS} \rangle \cdot \overline{entryAction}$$
$$\cdot \overline{activeT_{CS}} \cdot (CS' \mid R_1 \mid R_2 \mid ... \mid R_n)$$

$$CS' = \sum_{i=1}^{n}(ie_i \cdot \overline{a_i} \cdot CS') + exit \cdot \overline{exitAction} \cdot \overline{exitComplete} \cdot CS$$

$$R_i = composition\ of\ all\ components\ in\ region$$

โดยที่

CS คือ สถานะประกอบที่อยู่ในสถานะไม่ทำงาน

CS' คือ สถานะประกอบที่อยู่ในสถานะทำงาน

R_i คือ พื้นที่ย่อยภายในสถานะประกอบแต่ละพื้นที่ ซึ่งการอธิบายองค์ประกอบย่อยภายในพื้นที่ย่อยนั้นจะใช้หลักการเดียวกัน

$entry_{CS}$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะประกอบนี้

$\overline{setActiveState\langle entry_S \rangle}$ คือ การส่งสัญญาณแจ้งสถานะปัจจุบันของระบบ

$\overline{entryAction}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ $entry_{CS}$ เข้าสู่สถานะ

$\overline{activeT_{CS}}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะนี้ทำงาน

ie_i คือ เหตุการณ์ที่เกิดขึ้นภายในสถานะ ซึ่งไม่มีผลทำให้สถานะของระบบเปลี่ยนไป

$\overline{a_i}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ภายใน ie_i

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะปัจจุบัน

$\overline{exit_{R_i}}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันในพื้นที่ย่อยต่างๆ ภายในสถานะประกอบเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนแปลงสถานะไปยังสถานะอื่น

$exitComplete_{R_i}$ คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานปัจจุบันในแต่ละพื้นที่ย่อยเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{exitAction}$ คือ การกระทำที่ตอบสนองเมื่อเกิดเหตุการณ์ $exit$ ออกจากสถานะ

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

4) สถานะซิง (synch state)

เมื่อมีการเปลี่ยนแปลงสถานะเข้าสู่สถานะซิงแล้วจำนวนโทเคนในสถานะจะเพิ่มขึ้น และเมื่อมีการใช้งานโทเคนในสถานะซิงแล้วจำนวนโทเคนในสถานะก็จะลดลงเช่นกัน ในลักษณะการบรรยายนั้นจะใช้การแตกกระบวนการออกเพื่อรอรับการเพิ่มโทเคนต่อไป ซึ่งตัวแปร i ใน SS_i จะเป็นตัวบอกถึงจำนวนโทเคนที่อยู่ในปัจจุบัน ในกรณีที่ค่าจำนวนโทเคนมากที่สุดที่จัดเก็บได้ถูกระบุไว้ในค่า k แล้ว เมื่อมีเหตุการณ์เปลี่ยนมายังสถานะซิงเพื่อเพิ่มโทเคนแล้ว จำนวนโทเคนจะไม่คงที่และไม่เพิ่มขึ้น จากแนวคิดดังกล่าวเราสามารถบรรยายสถานะซิงได้ด้วยนิพจน์ดังนี้

$$SS = entry_{SS} \cdot (\overline{activeT_{SS}} \cdot SS' \mid SS_1) + decToken \cdot SS$$

$$SS_i = entry_{SS} \cdot (\overline{activeT_{SS}} \cdot SS' \mid SS_{i+1}) + decToken \cdot SS_{i-1}$$

$$SS_k = entry_{SS} \cdot SS_k + decToken \cdot SS_{k-1}$$

$$SS' = exit \cdot \overline{exitComplete} \mid \overline{decToken}$$

โดยที่

SS คือ สถานะซิงที่ไม่ใช่สถานะทำงาน

SS' คือ สถานซิงที่อยู่ในสถานะทำงาน

$entry_{SS}$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะซิง SS

$\overline{activeT_{SS}}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะซิงนี้ทำงาน

$decToken$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบมีการใช้งานโทเคน

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะปัจจุบัน

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{decToken}$ คือ การส่งสัญญาณใช้งานโทเคน

5) สถานะเริ่มต้นและสิ้นสุด (initial & final pseudostate)

การอธิบายพฤติกรรมของสถานะเริ่มต้นจะคล้ายคลึงกับสถานะทั่วไป เพียงแต่สถานะเริ่มต้นจะไม่มีการกระทำตอบสนองเมื่อเข้าสู่สถานะและออกจากสถานะ และจะรอรับสัญญาณแจ้งสถานะปัจจุบันของระบบ $\overline{setActiveState}\langle entry_S \rangle$ แต่จะไม่บันทึกสถานะนั้นไว้ และในส่วนของสถานะสิ้นสุดจะไม่มีสถานะทำงานหรือไม่ทำงาน และไม่มีเส้นการเปลี่ยนแปลงออกจากสถานะ จากแนวคิดดังกล่าวสามารถอธิบายพฤติกรรมได้ด้วยนิพจน์ดังนี้

$$I = entry_I \cdot \overline{activeT_I} \cdot I' + setActiveState(entry_S) \cdot I$$

$$I' = exit \cdot \overline{exitComplete} \cdot I$$

$$F = entry_F \cdot F$$

โดยที่

I คือ สถานะเริ่มต้นที่ไม่ทำงาน

I' คือ สถานะเริ่มต้นที่ทำงาน

$entry_I$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะเริ่มต้น

$\overline{activeT_I}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะนี้ทำงาน

$setActiveState(entry_S)$ คือ เหตุการณ์ที่เกิดขึ้นเพื่อแจ้งสถานะปัจจุบันของระบบ

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะปัจจุบัน

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

F คือ สถานะสิ้นสุด

$entry_F$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะสิ้นสุด

6) สถานะประวัติ (history pseudostate)

จากรูปที่ 3.1 เราสามารถอธิบายด้วยนิพจน์ ดังนี้

$$H = entry_I \cdot \overline{activeT_H} \cdot H' + setActiveState(entry_S) \cdot Hx(entry_S)$$

$$Hx(entry_S) = entry_I \cdot \overline{entry_S} \cdot H + setActiveState(entry_{NS}) \cdot Hx(entry_{NS})$$

$$H' = exit \cdot \overline{exitComplete} \cdot H$$

โดยที่

H คือ สถานะประวัติที่ยังไม่ได้ถูกบันทึกสถานะปัจจุบันและอยู่ในสถานะไม่ทำงาน

H' คือ สถานะประวัติที่ยังไม่ได้ถูกบันทึกสถานะปัจจุบัน และอยู่ในสถานะทำงาน

Hx คือ สถานะประวัติที่ถูกบันทึกสถานะปัจจุบันไว้ผ่านตัวแปร $entry_S$

$entry_I$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะประวัติ

$\overline{activeT_H}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะประวัตินี้

$\overline{entry_S}$ คือ การส่งสัญญาณไปยังสถานะที่ถูกบันทึกไว้ให้เป็นสถานะทำงาน

$setActiveState(entry_S)$ คือ เหตุการณ์ที่เกิดขึ้นเพื่อแจ้งสถานะปัจจุบันของระบบ

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะประวัติ

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะประวัติไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

7) สถานะแยกและเชื่อมสำหรับการทำงานพร้อมกัน (fork & join pseudostate)

ในแผนภาพสถานะยูเอ็มแอลนั้นจะมีการใช้งานสถานะแยกและสถานะเชื่อมเฉพาะเมื่อมีการใช้งานร่วมกับสถานะประกอบเท่านั้นดังรูปที่ 3.10 ซึ่งหากทำการพิจารณารูปแบบการเชื่อมต่อดังกล่าว จะเปรียบได้เสมือนการเชื่อมต่อเส้นการเปลี่ยนแปลงไปยังสถานะประกอบโดยตรงแต่จะทำการกำหนดสถานะทำงานเริ่มต้นของแต่ละพื้นที่ย่อยภายในด้วย โดยที่การทำงาน

จะไม่ได้เริ่มต้นที่สถานะเริ่มต้นของแต่ละพื้นที่ย่อยเช่นเดิม ซึ่งเราจะอธิบายสถานะแยกและเส้นการเปลี่ยนแปลงที่ออกจากสถานะแยกรวมกันด้วยนิพจน์เดียว แต่ในส่วนของสถานะเชื่อมนั้นจะแยกอธิบายเส้นการเปลี่ยนแปลงที่เข้าสู่สถานะเชื่อมกับสถานะเชื่อมออกจากกัน โดยการอธิบายเส้นการเปลี่ยนแปลงจะเป็นดังข้อ 2) ส่วนสถานะเชื่อมนั้นจะมีการรอรับสัญญาณจากเส้นการเปลี่ยนแปลงทั้งหมดก่อน จึงจะสั่งให้เส้นการเปลี่ยนแปลงที่ออกจากสถานะเชื่อมเปลี่ยนเป็นสถานะทำงาน จากแนวคิดดังกล่าวเราสามารถอธิบายได้ด้วยนิพจน์ดังนี้

$$FK = entry_{FX} \cdot \overline{entry_{CS}} \cdot (\overline{entry_{S_{R_1}}} \mid \overline{entry_{S_{R_2}}} \mid ... \mid \overline{entry_{S_{R_n}}}) \cdot FK$$

$$J = (entry_{J_1} \mid entry_{J_2} \mid ... \mid entry_{J_n}) \cdot \overline{exit} \cdot exitComplete \cdot \overline{activeT_J} \cdot J'$$

$$J' = exit \cdot \overline{exitComplete} \cdot J$$

โดยที่

FK คือ สถานะแยก

$entry_{FX}$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะแยก

$\overline{entry_{CS}}$ คือ การส่งสัญญาณไปยังสถานะประกอบให้เปลี่ยนเป็นสถานะทำงาน

$\overline{entry_{S_{R_i}}}$ คือ การส่งสัญญาณไปยังสถานะย่อยในพื้นที่ย่อยแต่ละพื้นที่ให้เปลี่ยนเป็นสถานะทำงาน

J คือ สถานะเชื่อมที่อยู่ในสถานะไม่ทำงาน

J' คือ สถานะเชื่อมที่อยู่ในสถานะทำงาน

$entry_{J_i}$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะเชื่อมจากแต่ละพื้นที่ย่อย

$\overline{exit}$ คือ การส่งสัญญาณไปยังสถานะประกอบเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะอื่น

$exitComplete$ คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานประกอบเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{activeT_J}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะเชื่อมนี้ทำงาน

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะเชื่อม

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะของสถานะเชื่อมไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

8) สถานะตัวต่อและตัวเลือก (junction & choice point)

การอธิบายสถานะตัวต่อและตัวเลือกจะพิจารณาเหมือนกับสถานะทั่วไป ยกเว้นแต่ไม่มีเหตุการณ์และการกระทำต่างๆ ภายใน ซึ่งเราสามารถอธิบายได้ด้วยนิพจน์ดังนี้

$$CJ = entry_{CJ} \cdot \overline{activeT_{CJ}} \cdot CJ'$$

$$CJ' = exit \cdot \overline{exitComplete} \cdot CJ$$

โดยที่

CJ คือ สถานะตัวต่อหรือตัวเลือกที่อยู่ในสถานะไม่ทำงาน

CJ' คือ สถานะตัวต่อหรือตัวเลือกที่อยู่ในสถานะทำงาน

$entry_{CJ}$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบเปลี่ยนมายังสถานะ CJ

$\overline{activeT_{CJ}}$ คือ การส่งสัญญาณให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะ CJ ทำงาน

$exit$ คือ เหตุการณ์ที่เกิดขึ้นเมื่อระบบจะเปลี่ยนสถานะออกจากสถานะ CJ

$\overline{exitComplete}$ คือ การส่งสัญญาณแจ้งถึงการเปลี่ยนสถานะไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

9) แผนภาพย่อย (submachine)

เนื่องจากแผนภาพย่อยมีความหมายเดียวกันกับสถานะประกอบ เพียงแต่แยกการอธิบายสถานะภายในออกไปเป็นแผนภาพใหม่เท่านั้น ดังนั้นการอธิบายจึงใช้รูปรูปแบบเดียวกับสถานะประกอบในหัวข้อ 3) แต่จะมีพื้นที่ย่อยภายในเพียงหนึ่งพื้นที่

3.3.2 การเชื่อมต่อกันระหว่างองค์ประกอบประเภทต่างๆ

จากแนวคิดในการอธิบายแผนภาพโดยมององค์ประกอบต่างๆ ภายในทำงานร่วมกันแบบพร้อมกันนั้น ทำให้ลักษณะการเชื่อมต่อแบบต่างๆ ของสถานะกับเส้นการเปลี่ยนแปลงในแบบต่างๆ ที่อยู่ในระดับเดียวกันนั้นไม่มีความแตกต่างกันในการอธิบาย โดยเราสามารถใช้ตัวดำเนินการ | เชื่อมองค์ประกอบต่างๆ เข้าด้วยกันได้

1) สถานะที่มีเส้นการเปลี่ยนแปลงออกหนึ่งเส้น

จากรูปที่ 3.13 เราสามารถอธิบายการเชื่อมต่อในลักษณะดังกล่าวได้ดังนี้

$$S_0 \mid T_1 \mid S_1 \mid T_2 \mid ... \mid S_n$$

โดยที่

S_i คือ สถานะต่างๆ ที่เชื่อมต่อกัน

T_i คือ เส้นการเปลี่ยนแปลงที่เชื่อมออกจากสถานะแต่ละสถานะ

2) สถานะที่มีเส้นการเปลี่ยนแปลงออกมากกว่าหนึ่งเส้น

จากรูปที่ 3.14 เส้นการเปลี่ยนแปลงทั้งหมดที่ออกจากสถานะเดียวกันจะถูกอธิบายด้วยนิพจน์แสดงเส้นการเปลี่ยนแปลงเดียวดังนี้

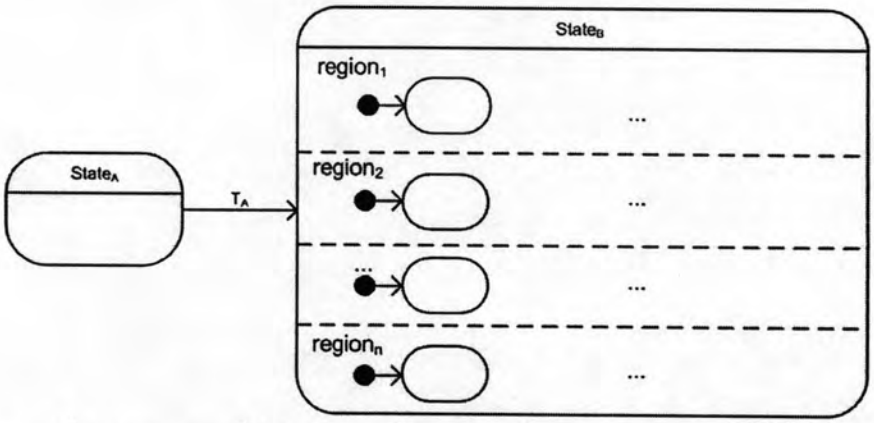
$$S \mid T_S$$

โดยที่

S คือ สถานะต้นทาง

T_S คือ เส้นการเปลี่ยนแปลงที่เชื่อมออกจากสถานะ S

3) สถานะที่มีเส้นการเปลี่ยนแปลงเชื่อมต่อไปยังสถานะประกอบ



รูปที่ 3.15 สถานะที่มีเส้นการเปลี่ยนแปลงเชื่อมต่อไปยังสถานะประกอบ

เส้นการเปลี่ยนแปลงที่ชี้ไปยังสถานะประกอบดังรูปที่ 3.15 นั้น จะพบว่ามีลักษณะคล้ายการอธิบายสถานะแยกในหัวข้อที่ผ่านมา แต่จะไม่มีเส้นการเปลี่ยนแปลงชี้ไปยังสถานะย่อยในแต่ละพื้นที่ย่อย อย่างไรก็ตามความหมายของระบบจะเปรียบเหมือนมีเส้นการเปลี่ยนแปลงชี้ไปยังสถานะเริ่มต้นแทน จากแนวคิดดังกล่าวเราสามารถอธิบายเส้นการเปลี่ยนแปลงดังกล่าวได้ดังนี้

$$T_A = activeT_A \cdot T'_A$$

$$T'_A = e \cdot ([c == True] \cdot \bar{a} \cdot \overline{exit} \cdot exitComplete \cdot \overline{entry_B} \cdot (\overline{entry_{1_{R_1}}} \mid \overline{entry_{1_{R_2}}} \mid ... \mid \overline{entry_{1_{R_n}}}) \cdot T_A + [c_i == False] \cdot T'_A)$$

โดยที่

T_A คือ เส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะประกอบที่อยู่ในสถานะไม่ทำงาน

$T_A{}'$ คือ เส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะประกอบที่อยู่ในสถานะทำงาน

e คือ เหตุการณ์ของเส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะประกอบ

c คือ เงื่อนไขของเส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะประกอบ

$\bar{a}$ คือ การกระทำของเส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะประกอบ

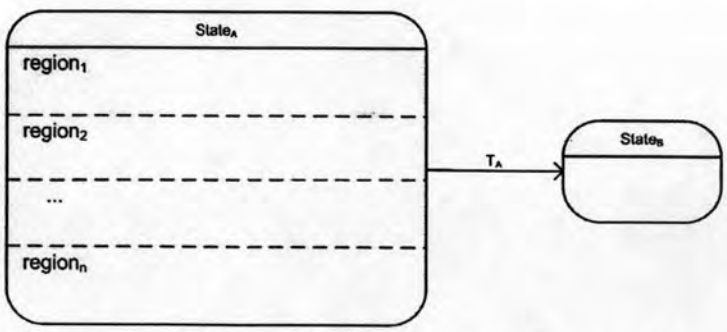
$\overline{exit}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะประกอบ

$exitComplete$ คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานปัจจุบันเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{entry_B}$ คือ การส่งสัญญาณไปยังสถานะประกอบปลายทางเพื่อให้เปลี่ยนเป็นสถานะทำงาน

$\overline{entry_{1_{R_i}}}$ คือ การส่งสัญญาณไปยังสถานะเริ่มต้นในพื้นที่ย่อยแต่ละพื้นที่ให้เปลี่ยนเป็นสถานะทำงาน

4) สถานะที่มีเส้นการเปลี่ยนแปลงเชื่อมออกจากสถานะประกอบ



รูปที่ 3.16 สถานะที่มีเส้นการเปลี่ยนแปลงออกจากสถานะประกอบ

เมื่อมีเหตุการณ์เพื่อเปลี่ยนสถานะปัจจุบันของระบบจากสถานะประกอบไปยังสถานะอื่นดังรูปที่ 3.16 นั้น การอธิบายเส้นการเปลี่ยนแปลงจะต้องเพิ่มการส่งสัญญาณไปยังเส้นการ

เปลี่ยนแปลงภายในพื้นที่ย่อยแต่ละพื้นที่เพื่อให้เปลี่ยนสถานะเป็นไม่ทำงานด้วย จากแนวคิดดังกล่าวเราสามารถอธิบายเส้นการเปลี่ยนแปลงดังกล่าวได้ดังนี้

$$T_A = activeT_A \cdot T'_A$$

$$T'_A = e \cdot ([c == True] \cdot \bar{a} \cdot (\overline{inactiveT_{R_1}} \mid \overline{inactiveT_{R_2}} \mid ... \mid \overline{inactiveT_{R_n}}) \cdot \overline{exit} \cdot exitComplete \cdot \overline{entry_B} \cdot T_A + [c == False] \cdot T'_A)$$

โดยที่

T_A คือ เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะประกอบไปยังสถานะอื่นซึ่งอยู่ในสถานะไม่ทำงาน

T'_A คือ เส้นการเปลี่ยนที่เชื่อมต่อกับสถานะประกอบซึ่งอยู่ในสถานะทำงาน

$activeT_A$ คือ เหตุการณ์ที่ทำให้เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะประกอบทำงาน

e คือ เหตุการณ์ของเส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะประกอบ

c คือ เงื่อนไขของแต่ละเส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะประกอบ

$\bar{a}$ คือ การกระทำของแต่ละเส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะประกอบ

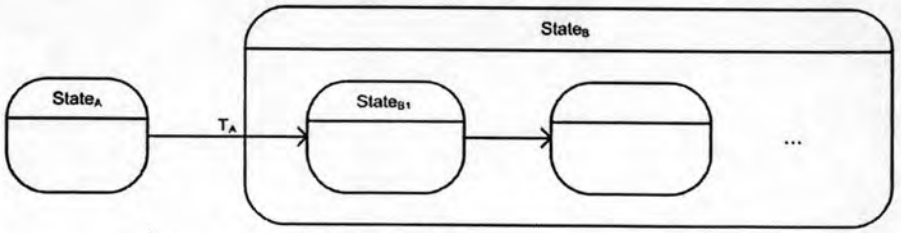
$\overline{inactiveT_{R_i}}$ คือ การส่งสัญญาณไปยังเส้นการเปลี่ยนแปลงที่ทำงานอยู่ภายในพื้นที่ย่อยแต่ละพื้นที่ให้เปลี่ยนสถานะตัวเองและสถานะที่เชื่อมต่อไปเป็นสถานะไม่ทำงาน

$\overline{exit}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะอื่น

$exitComplete$ คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานปัจจุบันเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{entry_B}$ คือ การส่งสัญญาณไปยังสถานะปลายทางของแต่ละเส้นการเปลี่ยนแปลงที่ออกจากสถานะประกอบเพื่อให้สถานะนั้นเปลี่ยนเป็นสถานะทำงาน

5) สถานะที่มีเส้นการเปลี่ยนแปลงเชื่อมต่อไปยังสถานะย่อยภายในสถานะประกอบ



รูปที่ 3.17 สถานะที่มีเส้นการเปลี่ยนแปลงเชื่อมต่อไปยังสถานะย่อยภายในสถานะประกอบ

ลักษณะของเส้นการเปลี่ยนแปลงเชื่อมต่อไปยังสถานะย่อมภายในสถานะรวมนั้น แบ่งได้สองรูปแบบ คือ แบบที่หนึ่ง สถานะรวมมีพื้นที่ย่อยภายในมากกว่าหนึ่งพื้นที่ ซึ่งรูปแบบดังกล่าวเส้นการเปลี่ยนแปลงจะต้องลากผ่านสถานะแยกดังรูปที่ 3.10 ซึ่งการอธิบายจะใช้นิพจน์ในรูปแบบที่ได้กล่าวถึงในหัวข้อสถานะแยก และรูปแบบที่สอง สถานะรวมมีพื้นที่ย่อยภายในเพียงหนึ่งพื้นที่ดังรูปที่ 3.17 ซึ่งการอธิบายจะเหมือนหัวข้อที่ 3) เพียงแต่กำหนดสถานะเริ่มต้นให้เป็นสถานะย่อยปลายทางของเส้นการเปลี่ยนแปลงแทนที่สถานะเริ่มต้น ซึ่งจากแนวคิดดังกล่าวเราสามารถอธิบายเส้นการเปลี่ยนแปลงได้ดังนี้

$$T_A = activeT_A \cdot T'_A$$

$$T'_A = e \cdot ([c == True] \cdot \bar{a} \cdot \overline{exit} \cdot exitComplete \cdot \overline{entry_B} \cdot \overline{entry_{B_1}} \cdot T_A + [c_i == False] \cdot T'_A)$$

โดยที่

T_A คือ เส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะรวมที่อยู่ในสถานะไม่ทำงาน

T_A' คือ เส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะรวมที่อยู่ในสถานะทำงาน

e คือ เหตุการณ์ของเส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะรวม

c คือ เงื่อนไขของเส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะรวม

$\bar{a}$ คือ การกระทำของเส้นการเปลี่ยนแปลงที่เชื่อมต่อไปยังสถานะรวม

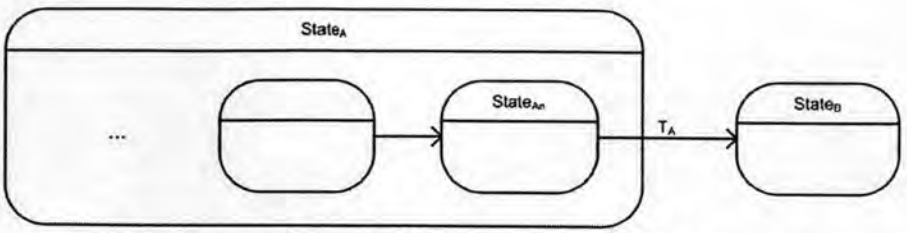
$\overline{exit}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะประกอบ

exitComplete คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานปัจจุบันเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{entry_B}$ คือ การส่งสัญญาณไปยังสถานะประกอบปลายทางเพื่อให้เปลี่ยนเป็นสถานะทำงาน

$\overline{entry_{B_1}}$ คือ การส่งสัญญาณไปยังสถานะย่อยปลายทางของเส้นการเปลี่ยนแปลงในพื้นที่ย่อยที่ให้เปลี่ยนเป็นสถานะทำงาน

6) สถานะย่อยที่มีเส้นการเปลี่ยนแปลงเชื่อมโยงไปยังสถานะอื่นๆ ภายนอกสถานะประกอบที่สถานะย่อยนั้นอยู่ภายใน



รูปที่ 3.18 สถานะที่มีเส้นการเปลี่ยนแปลงเชื่อมโยงไปยังสถานะอื่นๆ ภายนอกสถานะประกอบ

ลักษณะของเส้นการเปลี่ยนแปลงเชื่อมต่อไปยังสถานะอื่นๆ ภายนอกสถานะรวมนั้น แบ่งได้สองรูปแบบ คือ แบบที่หนึ่ง สถานะย่อยนั้นอยู่ในพื้นที่ที่อยู่ในสถานะรวมที่มีพื้นที่ย่อยภายในมากกว่าหนึ่งพื้นที่ ซึ่งรูปแบบดังกล่าวเส้นการเปลี่ยนแปลงจะต้องชี้ไปยังสถานะเชื่อมดังรูปที่ 3.10 ซึ่งการอธิบายจะใช้นิพจน์ในรูปแบบที่ได้กล่าวถึงในหัวข้อสถานะเชื่อม และรูปแบบที่สอง สถานะรวมมีพื้นที่ย่อยภายในเพียงหนึ่งพื้นที่ดังรูปที่ 3.18 ซึ่งการอธิบายจะคล้ายกับหัวข้อที่ 4) คือ พิจารณาเสมือนเส้นการเปลี่ยนแปลงนั้นเชื่อมโยงออกจากสถานะประกอบเอง แต่เส้นการเปลี่ยนแปลงจะถูกเปลี่ยนให้เป็นสถานะทำงานก็ต่อเมื่อมีการเปลี่ยนสถานะของระบบมายังสถานะย่อยที่เส้นการเปลี่ยนแปลงนั้นเชื่อมโยงอยู่เท่านั้น

$$T_A = activeT_A \cdot T'_A$$

$$T'_A = e \cdot ([c == True] \cdot \bar{a} \cdot \cdot \overline{exit_A} \cdot exitComplete \cdot \overline{exit} \cdot exitComplete \cdot \overline{entry_B} \cdot T_A + [c == False] \cdot T'_A)$$

โดยที่

T_A คือ เส้นการเปลี่ยนแปลงที่เชื่อมต่อกับสถานะย่อยในสถานะประกอบไปยังสถานะอื่นซึ่งอยู่ในสถานะไม่ทำงาน

T'_A คือ เส้นการเปลี่ยนที่เชื่อมต่อกับสถานะย่อยในสถานะประกอบซึ่งอยู่ในสถานะทำงาน

$activeT_A$ คือ เหตุการณ์ที่ทำให้เส้นการเปลี่ยนแปลง T_A ทำงาน

e คือ เหตุการณ์ของเส้นการเปลี่ยนแปลง T_A

c คือ เงื่อนไขของแต่ละเส้นการเปลี่ยนแปลง T_A

$\bar{a}$ คือ การกระทำของแต่ละเส้นการเปลี่ยนแปลง T_A

$\overline{exit_A}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันในพื้นที่ย่อยของสถานะประกอบเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะอื่น

$\overline{exit}$ คือ การส่งสัญญาณไปยังสถานะปัจจุบันเพื่อแจ้งให้ทราบว่าระบบจะมีการเปลี่ยนสถานะไปยังสถานะอื่น

exitComplete คือ เหตุการณ์ตอบกลับหลังจากสถานะทำงานปัจจุบันเปลี่ยนไปสู่สถานะไม่ทำงานเรียบร้อยแล้ว

$\overline{entry_B}$ คือ การส่งสัญญาณไปยังสถานะปลายทางของแต่ละเส้นการเปลี่ยนแปลงที่ออกจากสถานะประกอบเพื่อให้สถานะนั้นเปลี่ยนเป็นสถานะทำงาน

3.4 การตรวจสอบความสัมพันธ์ระหว่างแผนภาพสถานะ

ในหัวข้อนี้การตรวจสอบความสัมพันธ์ระหว่างแผนภาพสถานะจะบรรยายในลักษณะของทฤษฎี ซึ่งสามารถนำไปประยุกต์ในการตรวจสอบแผนภาพสถานะที่ถูกแปลงไปอยู่ในรูปของซีอาร์อีหรือไพแคลคูลัสได้โดยอาศัยแนวคิดเดียวกัน ซึ่งกรณีศึกษาที่จะได้นำเสนอในบทถัดไปจะแสดงให้เห็นถึงการประยุกต์ใช้กฎการตรวจสอบในรูปแบบของไพแคลคูลัส

3.4.1 การตรวจสอบความเท่าเทียมกันในด้านพฤติกรรมของวัตถุ

ในการพัฒนาซอฟต์แวร์ในระบบขนาดใหญ่นั้น โดยส่วนมากมักมีการพัฒนาในรูปแบบของส่วนประกอบย่อยๆ หลายๆ ส่วนประกอบกัน ซึ่งในระยะยาวนั้นส่วนประกอบต่างๆ เหล่านี้อาจต้องมีการเปลี่ยนแปลงหรือแทนที่ด้วยส่วนประกอบใหม่เพื่อเพิ่มความสามารถของระบบเดิมที่มีอยู่ แต่อย่างไรก็ตามในบางครั้งระบบก็อาจต้องคงความสามารถเดิมที่มีอยู่แล้วไว้ ดังนั้นจึงต้องมีการพิจารณาถึงส่วนประกอบใหม่ที่เข้ามาแทนที่ว่าสามารถตอบสนองต่อความต้องการพื้นฐานเดิมที่มีอยู่แล้วหรือไม่ ดังนั้นจากหลักการดังกล่าวเราจะทำการตรวจสอบความครอบคลุมของพฤติกรรมการตอบสนองของส่วนประกอบใหม่เทียบกับส่วนประกอบเดิม ซึ่งส่วนประกอบใหม่นั้นต้องตอบสนองต่อสิ่งแวดล้อมจากเหตุการณ์ต่างๆ ที่เกิดขึ้นเหมือนกับส่วนประกอบเดิม

ตัวอย่าง เช่น ให้ระบบหนึ่งๆ ประกอบด้วยส่วนประกอบ A ซึ่งมีพฤติกรรมการทำงาน $\bar{a}\cdot\bar{b}\cdot\bar{c}$ และมีการเปลี่ยนแปลงระบบโดยการแทนที่ส่วนประกอบ A ด้วย ส่วนประกอบ B ซึ่งภายในประกอบด้วยส่วนประกอบ X และ Y ทำงานด้วยกัน คือ $X \mid Y$ โดยที่ $X = \bar{a}\cdot m\cdot\bar{b}\cdot n$ และ $Y = \bar{m}\cdot\bar{n}\cdot\bar{c}$ ซึ่งจะได้พฤติกรรมการทำงานร่วมกัน คือ $\bar{a}\cdot m\cdot\bar{b}\cdot n\cdot\bar{c}$ แต่ m และ n เป็นการสื่อสารกันภายในระหว่าง X และ Y ไม่ใช่การสื่อสารกับสิ่งแวดล้อมภายนอก จึงไม่จำเป็นต้องสนใจพฤติกรรมดังกล่าว ซึ่งจะทำให้ได้พฤติกรรมที่ตอบสนองต่อสิ่งแวดล้อม คือ $\bar{a}\cdot\bar{b}\cdot\bar{c}$ ซึ่งพบว่าจะเหมือนกับส่วนประกอบ A ทำให้เราสรุปได้ว่า เราสามารถแทนที่การทำงานของส่วนประกอบ A ด้วยส่วนประกอบ B

3.4.2 การตรวจสอบพฤติกรรมการทำงานของวัตถุที่เกิดขึ้นเมื่อทำงานร่วมกับวัตถุอื่นภายในระบบ

ในการออกแบบระบบด้วยแผนภาพสถานะนั้น จะมีการออกแบบการตอบสนองของวัตถุแต่ละวัตถุแยกจากกัน แต่เมื่อระบบมีการทำงานจริงวัตถุดังกล่าวจะมีการทำงานเกี่ยวเนื่องกัน ซึ่งการสื่อสารที่เกิดขึ้นจะมีทั้งระหว่างภายนอกระบบกับวัตถุภายใน และระหว่างวัตถุภายในระบบเอง ในบางครั้งในระบบที่มีขนาดใหญ่หรือมีพฤติกรรมการทำงานที่ซับซ้อน การตรวจสอบพฤติกรรมต่างๆ ที่เกิดขึ้นย่อมเป็นไปได้ยากตามกัน งานวิจัยนี้จึงนำเสนอวิธีการตรวจสอบพฤติกรรมที่เกิดขึ้นดังกล่าวในส่วนหนึ่ง คือ การตรวจสอบพฤติกรรมบางส่วนของวัตถุที่ถูกออกแบบไว้ แต่เมื่อทำงานภายในระบบกลับไม่มีพฤติกรรมนั้นๆ เกิดขึ้น ซึ่งการตรวจสอบดังกล่าวจะนำเสนอให้เห็นถึงข้อบกพร่องของการออกแบบได้ 2 รูปแบบ คือ มีการออกแบบวัตถุให้มีความสามารถเกินความจำเป็น หรือไม่ได้มีการออกแบบวัถตุที่ทำงานร่วมกันให้สนับสนุนการทำงานวัตถุนั้นๆ

ตัวอย่าง เช่น ให้ระบบหนึ่งๆ ประกอบด้วย ส่วนประกอบ X ซึ่งถูกออกแบบให้มีพฤติกรรมการทำงานเป็น $a\cdot\bar{x}+b\cdot\bar{y}$ ซึ่งจะทำงานร่วมกันกับส่วนประกอบ Y ซึ่งมีพฤติกรรมการทำงานเป็น $\bar{a}+\bar{b}$ โดยมี a และ b เป็นการสื่อสารภายในระหว่าง X และ Y ซึ่งเมื่อสองส่วนประกอบระบบทำงานร่วมกันจะได้พฤติกรรมที่สื่อสารกับสิ่งแวดล้อมเป็น $\bar{x}+\bar{y}$ แต่หากเราทำการแก้ไขระบบใหม่โดยให้ $Y=\bar{a}$ จะได้ว่าพฤติกรรมของระบบที่สื่อสารต่อสิ่งแวดล้อมจะเป็น $\bar{x}$ เท่านั้น ซึ่งเมื่อเทียบกับ X ที่ถูกออกแบบให้มีพฤติกรรมสื่อสารกับสิ่งแวดล้อมเป็น $\bar{x}+\bar{y}$ จะพบว่าพฤติกรรม $\bar{y}$ ไม่สามารถเกิดขึ้นได้ เนื่องจากการออกแบบที่ผิดพลาดของส่วนประกอบ Y