2. งานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงงานวิจัยต่างๆซึ่งทำมาแล้ว และมีอิทธิพลต่องานวิจัยชิ้นนี้ โดยจะเน้นไปที่งานวิจัยด้านการจินตทัศน์อัลกอริทึมซึ่งศึกษาการแบ่งระบบการจินตทัศน์ออกเป็นส่วนย่อยๆ เพื่อความสะดวกในการพัฒนาการจินตทัศน์อัลกอริทึม งานวิจัยซึ่งจะนำมากล่าวถึงมีอยู่สองงานได้แก่ BALSA-II[2] และงานวิจัยของ ดร.สมชาย ประสิทธิ์จูตระกูล และ ดร.วิทยา วัชระวิทยากุล [3]

นอกจากงานวิจัยทั้งสองนี้แล้วยังมีงานวิจัยทางด้านการจินตทัศน์อัลกอริทึมอื่นอีก ซึ่งจะไม่กล่าวถึงในที่นี้เพราะงานวิจัยเหล่านี้ไม่ได้มุ่งเน้นถึงการแบ่งการจินตทัศน์อัลกอริทึมออกเป็นส่วนย่อยๆ ตัวอย่างงานวิจัยเหล่านี้ได้แก่ xtango[4] และ samba[5] ซึ่งมุ่งเน้นไปในส่วนการพัฒนาส่วนแสดงผลของการจินตทัศน์อัลกอริทึม Lens[6] ซึ่งเป็นงานวิจัยเกี่ยวกับการตรวจสอบการทำงานของโปรแกรมด้วยการจินตทัศน์ (Visual Debugging) และงานวิจัยเรื่อง Java Animation[7] ซึ่งจะเน้นไปที่การนำเสนอการจินตทัศน์อัลกอริทึมผ่านทางเครือข่าย Internet

2.1 BALSA-II

BALSA-II เป็นระบบการจินตทัศน์อัลกอริทึมซึ่งพัฒนาขึ้นโดย Marc H. Brown เพื่อทำงานบนเครื่องคอมพิวเตอร์แมคอินทอช มีจุดประสงค์หลักเพื่อนำไปใช้ในการเรียนการสอนของวิชาทางด้านโครงสร้างข้อมูลและงานอื่นๆซึ่งเกี่ยวข้องกับการวิเคราะห์และออกแบบอัลกอริทึม

2.1.1 โครงสร้างของระบบ

BALSA-II เป็นระบบการจินตทัศน์อัลกอริทึมซึ่งจะแบ่งการจินตทัศน์อัลกอริทึมออกเป็นสี่ประเภทย่อยๆได้แก่ ตัวสร้างข้อมูล (Input Generator) อัลกอริทึม (Algorithm) ตัวแปลงคำสั่ง (Adapter) และส่วนแสดงผล (Moderler/Renderer) ซึ่งยังจะแบ่งย่อยออกเป็น ส่วนแสดงภาพ (Renderer) จะรับผิดชอบในการแสดงผลการทำงานและส่วนกำหนดรูปแบบ (Modeler) จะเป็นผู้รับผิดชอบและควบคุมการจัดวางตำแหน่งของส่วนแสดงภาพที่จะปรากฏบทหน้าจอคอมพิวเตอร์

ส่วนประกอบเหล่านี้สามารถแยกกันไปพัฒนาต่างหาก ซึ่งการพัฒนาส่วนประกอบจะต้องทำตามข้อกำหนดในการพัฒนาของ BALSI-II อย่างเคร่งครัด เมื่อพัฒนาเสร็จก็แล้วจะนำมาแปล (compile) และเชื่อม (link) รวมกันกับส่วนประกอบกลางซึ่งจะเป็นส่วนติดต่อผู้ใช้ (user interface) และส่วนแก่นกลาง (kernel) ของระบบ BALSA-II แล้วสร้างเป็นการจินตทัศน์อัลกอริทึม โดยมีโปรแกรมช่วยงานบางตัวซึ่งช่วยให้การพัฒนาส่วน

---

[2] March H. Brown. <u>Algorithm Animation</u>. The MIT Press, 1988, p. 1-26.

[3] S. Prasitjutrakul and W. Watcharawittaykul. Algorithm Visualization System: An Overview of Internal Structure. Proc. of Third ASEAN Regional Seminar on Microelectronics and Information Technology, Bangkok.

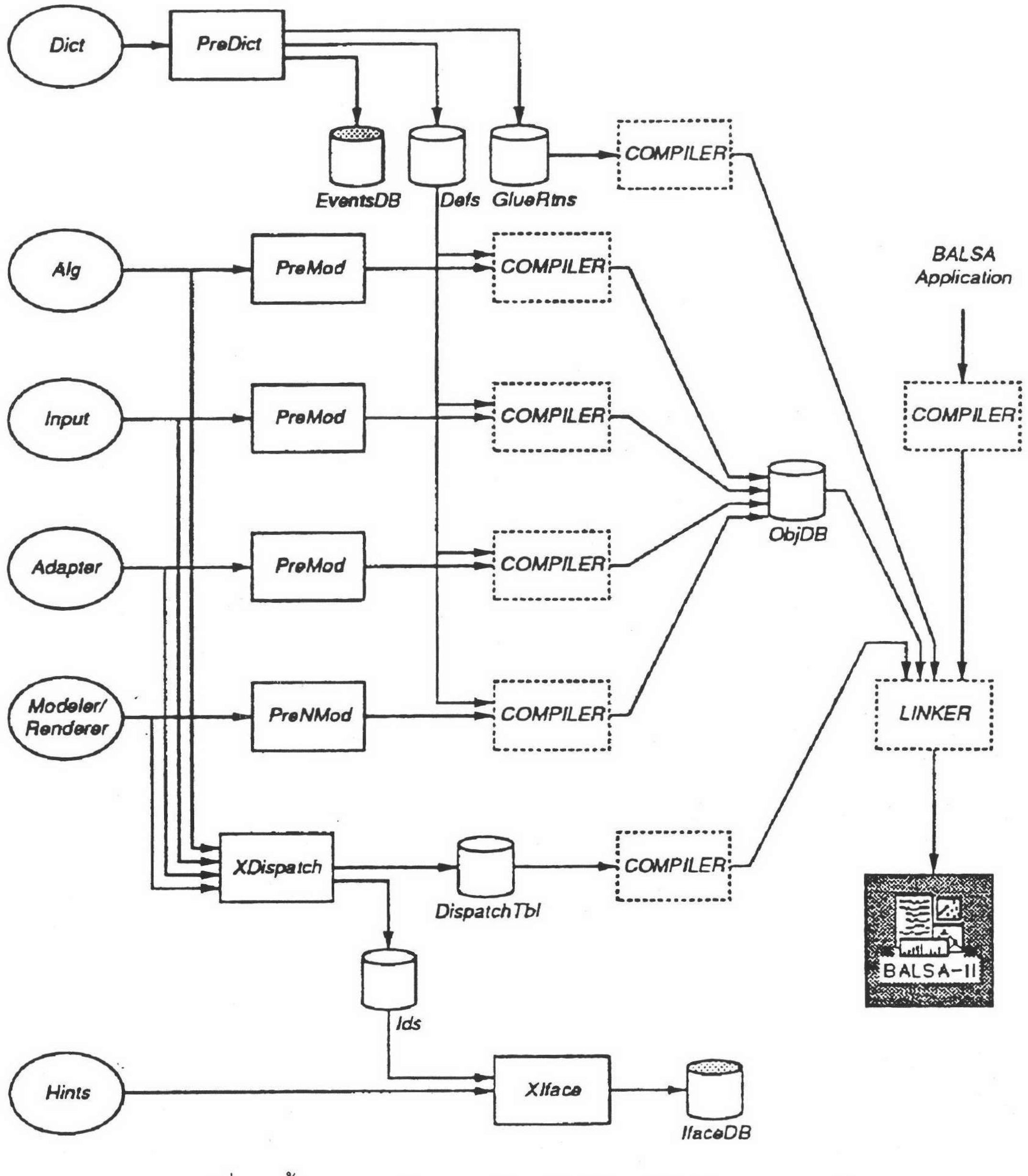
[4] Graphics, Visualization and Usibility Center. "XTANGO Algorithm Animation System". 1992. (INTERNET)

[5] Graphics, Visualization and Usibility Center. "SAMBA Animation Program". 1992. (INTERNET)

[6] Graphics, Visualization and Usibility Center. "Visualization for Program Understanding and Debugging".1992. (INTERNET)

[7] Carla Laffra. "Dijkstra's Shortest Path Algorithm Animation in Java". 1996. (INTERNET)

ประกอบเหล่านี้ทำได้ง่ายขึ้น ในรูปที่ 2.1 แสดงขั้นตอนการพัฒนาการจินตทัศน์อัลกอริทึมด้วย BALSA-II วงรีจะแทนโปรแกรมต้นฉบับและแฟ้มข้อมูลซึ่งผู้พัฒนาส่วนประกอบจะต้องสร้างขึ้น สี่เหลี่ยมกรอบเส้นประแทนโปรแกรมซึ่งไม่ได้เป็นส่วนประกอบของ BALSA-II กรอบสี่เหลี่ยมทึบแทนโปรแกรมช่วยงานของ BALSA-II รูปแท่งแทนแฟ้มข้อมูลซึ่ง BALSA-II สร้างขึ้นจากโปรแกรมต้นฉบับ



รูปที่ 2.1 ขั้นตอนการพัฒนาการจินตทัศน์อัลกอริทึมด้วย BALSA-II[8]

ส่วนประกอบทั้งหมดของการจินตทัศน์อัลกอริทึมจะพัฒนาขึ้นเป็นหน่วยย่อยๆ แยกกันต่างหากโดยใช้ภาษาระดับสูง เช่น PASCAL ร่วมกับคำสั่งพิเศษของ BALSA-II จากนั้นโปรแกรมต้นฉบับเหล่านี้จะถูกนำมาผ่านโปรแกรม PreMod/PreNMod ของ BALSA-II เพื่อทำการแปลงคำสั่งพิเศษของ BALSA-II ให้อยู่ในรูปแบบ

---

[8]March H. Brown. Algorithm Animation. The MIT Press, 1988, p. 128.

ซึ่งตัวแปลภาษา (COMPILER) เข้าใจได้ แล้วจึงนำโปรแกรมต้นฉบับทั้งหมดมาแปลจากนั้นจึงนำไปเชื่อมโยงกับส่วนประกอบกลางของ BALSA-II เพื่อสร้างเป็นการจินตทัศน์อัลกอริทึมที่สมบูรณ์ต่อไป

การที่ BALSA-II แยกการจินตทัศน์อัลกอริทึมออกเป็นส่วนย่อยๆมีจุดมุ่งหมายเพื่อ

- ส่วนประกอบที่เป็นอิสระ : โปรแกรมของอัลกอริทึม ตัวสร้างข้อมูล และส่วนแสดงผลจะต้องแยกจากกันอย่างเด็ดขาด เพื่อให้การพัฒนาส่วนประกอบแต่ละส่วนไม่ต้องกังวลกับส่วนประกอบอื่นๆ
- ทำให้การแก้ไขเกิดขึ้นน้อยที่สุด : การเปลี่ยนแปลงซึ่งเกิดขึ้นในส่วนประกอบหนึ่งๆ ไม่ควรมีผลกระทบกับส่วนประกอบอื่น
- การนำส่วนประกอบมาใช้ใหม่ : ส่วนประกอบใดก็ตามเมื่อถูกสร้างขึ้นแล้ว ควรจะนำมาใช้ใหม่ได้ในงานที่คล้ายคลึงกัน
- ความง่ายในการพัฒนา
- ประสิทธิภาพในการทำงาน : การทำงานของส่วนประกอบกลางของระบบจะต้องสิ้นเปลืองเวลาและทรัพยากร (overhead) น้อยที่สุด

จากการที่ส่วนประกอบทั้งหมดจะพัฒนาขึ้นเป็นหน่วยย่อยๆแยกกันต่างหากแล้วนำมาประกอบเพื่อใช้งานร่วมกันในภายหลัง จำเป็นต้องมีการออกแบบและกำหนดส่วนประสานงาน (interface) ที่แน่นอน เพื่อให้ส่วนประกอบต่างๆสามารถทำงานร่วมกันได้อย่างถูกต้องและมีประสิทธิภาพ ดังนั้นส่วนประกอบแต่ละส่วนจึงสามารถแบ่งออกไปเป็นชิ้นส่วนย่อย ๆ สองชิ้นได้แก่ โปรแกรมซึ่งเป็นส่วนทำงานจริงๆ และโปรแกรมซึ่งเป็นส่วนประสานงาน

```
module Alg.Insertionsort imports 'InterbalSorts.Defs';
procedure Alg.Insertionsort.Code;
    var i,j,v : Integer;
          N : Integer;
          a : array[1..MaxN] of Integer;
          eofFg : Boolean;
    begin
          eofFg := InputEvent.HowManyKeys(N);
          for i:= 1 to N do eofFg := InputEvent.ReadKey(a[i]);
          for i:= 1 to N do OutputEvent.NewKey(i,a[i]);
          for I:=2 to N do
          begin
                v := a[i];j := i ;
                while a[j-1] > v do
                begin
                      a[j]:=a[j-1];    j:=j-1;
                      OutputEvent.Swap(j,j+1);
                end;
                a[j]:=v;
          end;
    end;
end.
```

รูปที่ 2.2 โปรแกรมต้นฉบับของอัลกอริทึมการจัดเรียงข้อมูลแบบแทรก (insertion sort)[9]

---

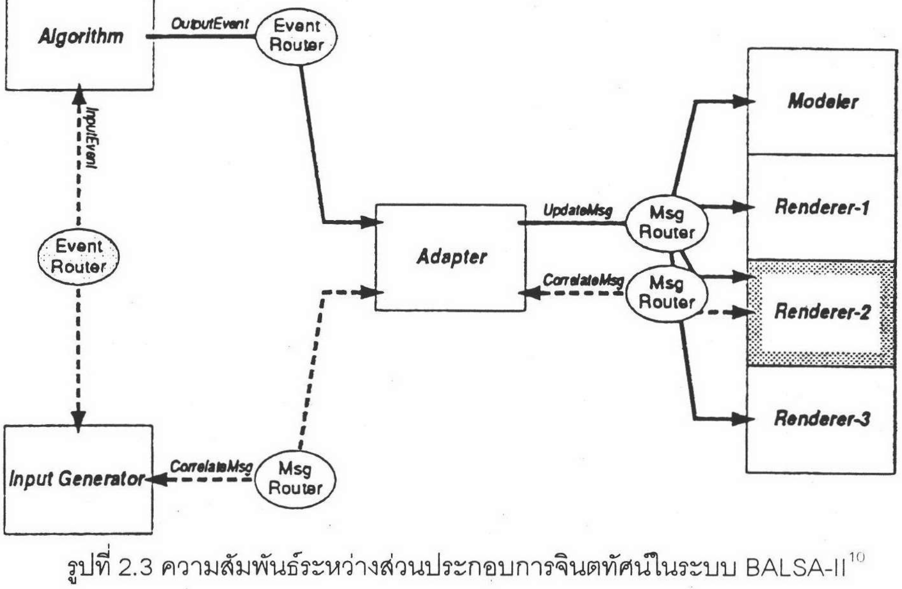
[9]March H. Brown. Algorithm Animation. The MIT Press, 1988, p. 96.

จากรูปจะเห็นว่าจะมีส่วนของโปรแกรมซึ่งใช้ประสานการทำงานกับส่วนประกอบอื่น(แสดงด้วยตัวอักษรเอน)แทรกอยู่ภายในส่วนของโปรแกรมซึ่งใช้จัดเรียงข้อมูล การแทรกโปรแกรมของส่วนประสานงานลงไปจะแทรกลงไปในจุดซึ่งต้องการติดต่อกับส่วนประกอบอื่นหรือ BALSA-II โดยจะใช้คำว่ามีเหตุการณ์ที่น่าสนใจ (interested event) เกิดขึ้น BALSA-II จะแบ่งเหตุการณ์ออกเป็น 2 ประเภทคือ

- เหตุการณ์ขาเข้า (input event) เป็นเหตุการณ์หรือการติดต่อซึ่งส่วนประกอบต้องการข้อมูลจากส่วนประกอบอื่น(เช่นคำสั่งInputEvent.HowManyKeys(N) ที่อัลกอริทึมจัดเรียงข้อมูลส่งไปยังส่วนสร้างข้อมูลเพื่อขอจำนวนข้อมูลที่จะมาจัดเรียง)
- เหตุการณ์ขาออก (output event) ซึ่งจะเกิดขึ้นเมื่อส่วนประกอบต้องการส่งคำสั่งการทำงานหรือแจ้งเตือนให้องค์ประกอบอื่นทราบสถานะของตนเอง เช่น คำสั่ง OutputEvent.Newkey(i,a(i)) ซึ่งอัลกอริทึมจัดเรียงข้อมูลส่งไปยังส่วนแสดงผลเพื่อแสดงว่าเกิดการสลับค่าข้อมูลแล้วให้ส่วนแสดงผลแสดงภาพให้ผู้ใช้เห็นด้วย

หากสังเกตจะพบว่าเหตุการณ์ขาเข้าจะเทียบได้กับคำสั่ง Readln ในภาษา PASCAL และเหตุการณ์ขาออกจะเทียบได้กับคำสั่ง Writeln ในกรณีที่เราพัฒนาโปรแกรมนี้เพื่อนำไปใช้งานภายนอกระบบ BALSA-II

ในรูปที่ 2.3 จะแสดงความสัมพันธ์ในขณะทำการจินตทัศน์อัลกอริทึมของส่วนประกอบทั้งสี่ประเภทนี้ซึ่งนำมาประกอบเป็นการจินตทัศน์อัลกอริทึมตามวิธีการในรูปที่ 2.1 เมื่อผู้ใช้สั่งให้เริ่มทำการจินตทัศน์อัลกอริทึมส่วนอัลกอริทึมจะส่งเหตุการณ์ขาเข้าไปยังส่วนสร้างข้อมูลเพื่อขอข้อมูล จากนั้นจะนำข้อมูลที่ได้รับไปใช้ในการทำงาน เมื่อทำงานมาจนถึงจุดที่ต้องการแสดงผลการทำงานก็จะทำการส่งเหตุการณ์ขาออกไปให้ส่วนแปลงคำสั่ง ซึ่งจะรับทราบเหตุการณ์ขาออกที่ส่วนอัลกอริทึมสร้างขึ้นแล้วนำมาแปลงเป็นคำสั่งแสดงผลเพื่อส่งให้ส่วนแสดงผลนำไปแสดงผลการทำงานของอัลกอริทึมให้ผู้ใช้ได้รับรู้ต่อไป



รูปที่ 2.3 ความสัมพันธ์ระหว่างส่วนประกอบการจินตทัศน์ในระบบ BALSA-II[10]

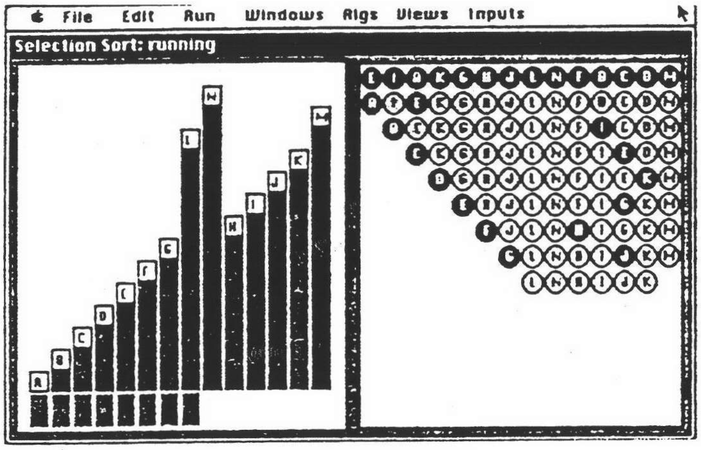
[10]March H. Brown. Algorithm Animation. The MIT Press, 1988, p. 10.

2.1.2 การติดต่อประสานงานกับผู้ใช้

ในส่วนประกอบกลางของ BALSA-II นอกจากแก่นกลางซึ่งเป็นผู้ควบคุมและประสานการทำงานและจัดการเรื่องการส่งผ่านเหตุการณ์ต่างๆแล้ว ยังมีส่วนประกอบอีกชิ้นหนึ่งซึ่งมีความสำคัญอย่างยิ่งต่อระบบ BALSA-II ได้แก่ส่วนติดต่อกับผู้ใช้ ผู้ใช้ที่กล่าวถึงในที่นี้คือผู้ใช้ปลายทาง (End User) ซึ่งจะเป็นผู้นำเอาการจินตทัศน์อัลกอริทึมซึ่งสร้างเรียบร้อยแล้วโดยผู้พัฒนาการจินตทัศน์ไปใช้งานเพื่อสังเกตและศึกษาการทำงานของอัลกอริทึม

BALSA-II เป็นระบบซึ่งทำงานบนเครื่องคอมพิวเตอร์แมคอินทอชซึ่งมีการติดต่อประสานกับผู้ใช้เป็นแบบกราฟิก (Graphic User Interface - GUI) ที่เป็นมาตราฐานอยู่แล้ว ดังนั้นส่วนติดต่อกับผู้ใช้ของ BALSA-II ซึ่งยึดถือตามมาตราฐานนี้จึงโดยการควบคุมการทำงานของระบบจะทำผ่านเมนู สัญรูป (icon) และกรอบโต้ตอบ ดังในรูปรูปที่ 2.4

นอกจากนี้โปรแกรมบนระบบแบบแมคอินทอชยังมีความสามารถที่จะตอบสนองคำสั่งของผู้ใช้ในทันที (Interactive) ได้ ซึ่งทำให้ BALSA-II ใช้งานได้ง่ายยิ่งขึ้น เพราะผู้ใช้สามารถสังเกตผลการทำงานของคำสั่งต่างๆ ที่สั่งให้ BALSA-II กระทำปรากฏขึ้นในทันที

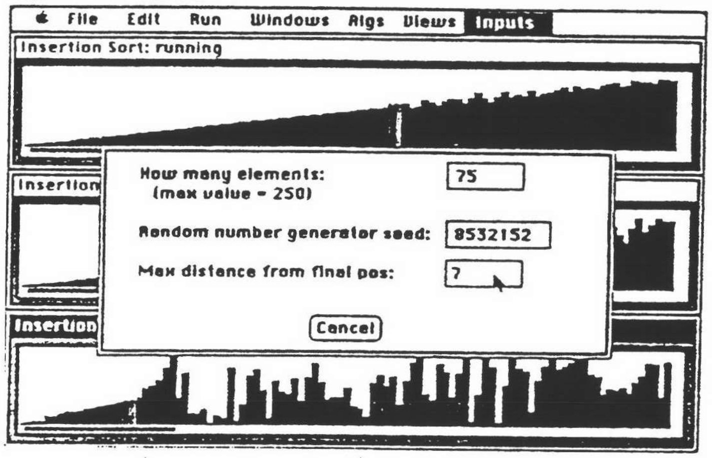


รูปที่ 2.4 ส่วนติดต่อประสานงานกับผู้ใช้ของโปรแกรม BALSA-II[11]

การควบคุมการทำงานของระบบ BALSA-II ทำได้ในสองลักษณะคือ การตั้งค่าพารามิเตอร์ของส่วนประกอบต่างๆ ผ่านทางกรอบโต้ตอบ ดังในรูปที่ 2.5 ซึ่งแสดงการตั้งค่าพารามิเตอร์ของส่วนสร้างข้อมูลสำหรับอัลกอริทึมจัดเรียงข้อมูลที่จะสร้างชุดของตัวเลขซึ่งเกือบจะจัดเรียงไว้แล้วชุดหนึ่ง โดยผู้ใช้สามารถกำหนดจำนวนข้อมูล ค่าสุ่มเริ่มต้น (seed number) และลักษณะของข้อมูลว่าจะแตกต่างจากข้อมูลที่เรียงแล้วมากหรือน้อยเพียงใด

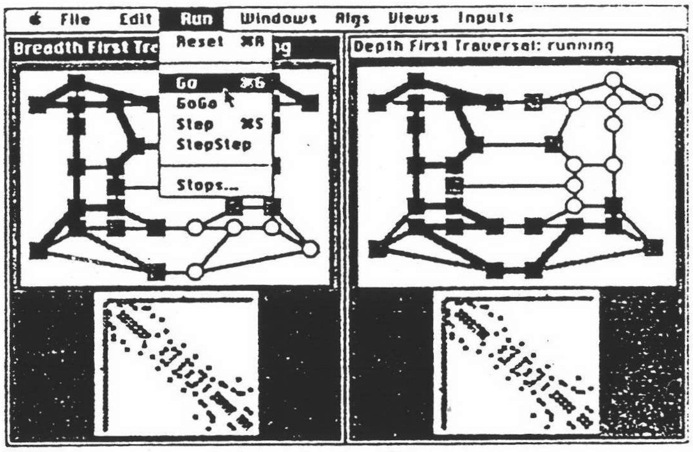
---

[11]March H. Brown. Algorithm Animation. The MIT Press, 1988, p. 48.



รูปที่ 2.5 การกำหนดค่าเริ่มต้นของตัวสร้างข้อมูล[12]

การควบคุมการทำงานของการจินตทัศน์อัลกอริทึมอีกวิธีหนึ่งคือทำการควบคุมผ่านทางเมนู (ดูรูปที่ 2.6) โดยผู้ใช้สามารถสั่งให้การจินตทัศน์อัลกอริทึมเริ่มและหยุดทำงานเมื่อใดก็ได้ นอกจากนี้ยังสามารถสั่งให้อัลกอริทึมหยุดทำงานชั่วคราว (pause) หรือทำงานทีละขั้นตอนเพื่อศึกษาการทำงานอย่างละเอียดได้อีกด้วย



รูปที่ 2.6 การควบคุมการทำงานของอัลกอริทึม[13]

2.1.3 บทการจินตทัศน์

ในการควบคุมการทำงานของ BALSA-II นั้นนอกจากผู้ใช้จะสามารถควบคุมการทำงานได้โดยตรงขณะทำงานโดยใช้เมนูและกรอบโต้ตอบแล้ว ผู้ใช้งานระบบ BALSA-II ยังสามารถกำหนดการทำงานล่วงหน้าของการจินตทัศน์ การจัดวางตำแหน่งของหน้าจอแสดงผลต่างๆและอัลกอริทึมที่จะนำมาใช้งาน เตรียมไว้ก่อนในรูปแบบของบทการจินตทัศน์ (script) ได้อีกด้วย ซึ่งการใช้บทการจินตทัศน์มีประโยชน์มากในการนำการจินตทัศน์อัลกอริทึมไปใช้งานในการเรียนการสอน เนื่องจากผู้สอนสามารถเตรียมบทการจินตทัศน์ไว้ก่อน เมื่อถึงเวลาสอนก็สามารถนำมาแสดงให้นักศึกษาดูได้ทันที ทำให้ประหยัดเวลาในการใช้งาน

---

[12] March H. Brown. Algorithm Animation. The MIT Press, 1988, p. 60.

[13] Ibid.,p. 65.

```
AlgDef ALG=1 NAME='GrahamScan' PARAM='Shellsort'
InputDef ALG=1 NAME='FromFile',PARAM='d.x31.r100'
AlgWDef ALG=1 LOC=0.,0.,1.,1. DRESSING
ViewWTiles 'H-SPLIT'
ViewWDef ALG=1 VIEW=1 NAME='Hull Details'  PARAM='' DX=0. DY=0.
     DZ=1. NODRESSING
ViewWDef ALG=1 VIEW=2 NAME='Hull Only'  PARAM='' DX=.5 DY=.5
     DZ=2. DRESSING
Stops ALG=1 TryPt=1,1,5

Start

@127 DeleteView ALG=1 VIEW=2
@127 Freeze 'Let her rip now'
@127 Stops ALG=1 NewPt=0,0,5

@384 ViewWReloc ALG=1 VIEW=1 LOC=.5,0.,1.,1.
@384 ViewWLoc ALG=1 VIEW=2 LOC=0.,0.,.5,1.
@384 ViewWDef ALG=1 VIEW=2 NAME='Classify Pts'  PARAM='COLOR'
     DX=0. DY=0. DZ=1. NODRESSING

@593 Done
```

รูปที่ 2.7 ตัวอย่างบทการจินตทัศน์[14]

จากรูปที่ 2.7 จะเป็นตัวอย่างบทการจินตทัศน์ซึ่งแสดงการจินตทัศน์ของอัลกอริทึมจัดเรียงข้อมูลแบบเชลล์ (Shell Sort) ด้วยวิธีการแบ่งแบบ Graham Scan โดยการจินตทัศน์นี้จะใช้ตัวสร้างข้อมูลที่มีชื่อว่า FromFile ส่วนแสดงผลของการจินตทัศน์อัลกอริทึมนั้น ในช่วงแรกจะมีการสร้างส่วนแสดงผลสองหน้าต่างซึ่งจะมีตำแหน่งและค่าเริ่มต้นต่างๆ ตามที่กำหนดไว้ในบทการจินตทัศน์ หลังจากกำหนดค่าต่างๆเรียบร้อยแล้วก็จะเริ่มทำการจินตทัศน์ เมื่อการจินตทัศน์ทำงานไปจนถึงเวลาที่ 127 ให้ปิดหน้าต่างแสดงผลหน้าต่างที่สอง แล้วทำการสร้างหน้าต่างแสดงผลใหม่ด้วยค่าเริ่มต้นที่ต่างไปเพื่อแสดงการทำงานของอัลกอริทึม แล้วจึงทำงานต่อจนถึงเวลาที่ 593 จากนั้นให้จบการจินตทัศน์ลง

2.2 Algorithm Visualization System

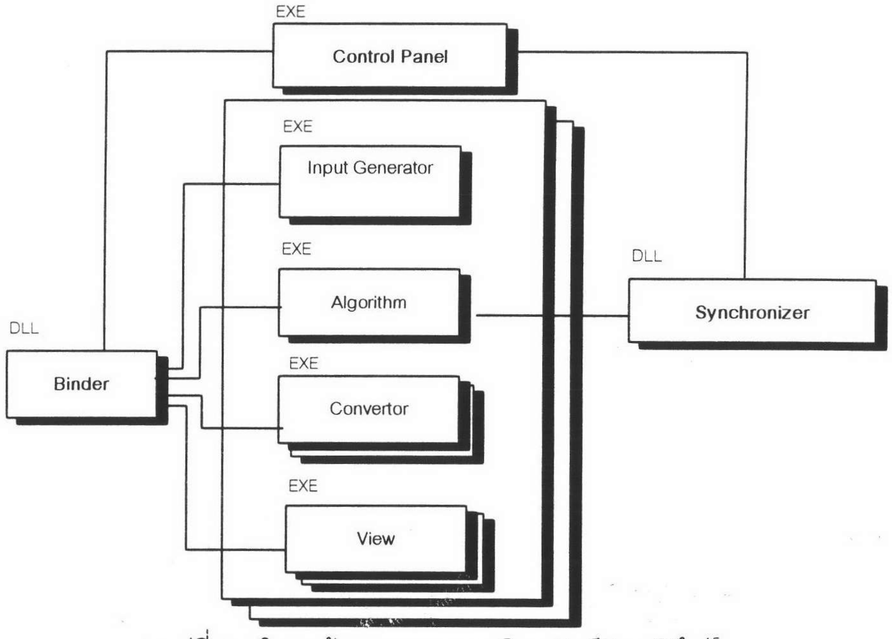
งานวิจัยชิ้นนี้กระทำโดย ดร.สมชาย ประสิทธิ์จูตระกูล และ ดร.วิทยา วัชระวิทยากุล เพื่อพัฒนาระบบการจินตทัศน์อัลกอริทึมบนระบบปฏิบัติการไมโครซอฟต์วินโดวส์ (Microsoft Windows) ซึ่งระบบปฏิบัติการนี้มีข้อดีคือยินยอมให้มีโปรแกรมมากกว่าหนึ่งโปรแกรมทำงานไปพร้อมๆกันได้ในลักษณะของการทำงานร่วมกันแบบไม่มีการขัดจังหวะ (Non-preemptive Multitasking System) จากความสามารถในส่วนนี้ของวินโดว์ทำให้เราสามารถแยกการจินตทัศน์อัลกอริทึมออกเป็นส่วนประกอบย่อยๆ โดยแต่ละส่วนเป็นโปรแกรมที่ทำงานแยกจากกันอย่างอิสระ และจากข้อดีนี้ยังทำให้เราสามารถนำอัลกอริทึมมากกว่าหนึ่งอัลกอริทึมมาทำงานไปพร้อมๆกัน เพื่อเปรียบเทียบการทำงานของแต่ละอัลกอริทึมได้

---

[14]March H. Brown. Algorithm Animation. The MIT Press, 1988 p. 86.

2.2.1 โครงสร้างของระบบการจินตทัศน์อัลกอริทึม

งานวิจัยชิ้นนี้ได้แบ่งการจินตทัศน์อัลกอริทึมออกเป็นส่วนประกอบย่อยๆในลักษณะของโปรแกรมบนระบบไมโครซอฟต์วินโดว์ เพื่อที่จะทำให้สามารถนำส่วนประกอบแต่ละส่วนแยกออกไปพัฒนาต่างหาก เมื่อต้องการสร้างการจินตทัศน์อัลกอริทึมก็จะนำเอาส่วนประกอบเหล่านี้มารวมเข้าด้วยกันใหม่



รูปที่ 2.8 โครงสร้างของระบบการจินตทัศน์อัลกอริทึม[15]

โครงสร้างหลักของระบบจะประกอบไปด้วยส่วนต่างๆดังในรูปที่ 2.8 ส่วนประกอบใดที่มีคำว่า EXE กำกับแสดงว่าเป็นส่วนประกอบที่อยู่ในรูปแบบของโปรแกรมที่ทำงานบนระบบวินโดว์ (Windows Application) และส่วนประกอบที่มีคำว่า DLL กำกับแสดงว่าเป็นส่วนประกอบที่พัฒนาขึ้นในรูปแบบของคลังโปรแกรมเชื่อมโยงแบบพลวัต (Dynamic Link Library - DLL) ของวินโดว์

จากรูปที่ 2.8จะพบว่า โครงสร้างของระบบจะประกอบไปด้วยส่วนประกอบดังนี้คือ

- ส่วนสร้างข้อมูล (input generator) จะทำการสร้างข้อมูลเพื่อส่งไปให้ส่วนอัลกอริทึมทำงาน
- ส่วนอัลกอริทึม (algorithm) เป็นตัวอัลกอริทึมซึ่งต้องการทำการจินตทัศน์
- ส่วนแปลงคำสั่ง (converter) เป็นโปรแกรมซึ่งคั่นกลางอยู่ระหว่างส่วนอัลกอริทึมและส่วนแสดงผล รายละเอียดของส่วนประกอบนี้จะขอกล่าวในหัวข้อถัดไป
- ส่วนแสดงผล (view) เป็นส่วนซึ่งจะรับคำสั่งการแสดงผลมาจากส่วนแปลงคำสั่ง แล้วนำมาแสดงผลบนจอภาพ เพื่อให้ผู้ใช้ศึกษาสถานะและลักษณะการทำงานของอัลกอริทึม
- ส่วนควบคุมการทำงาน (controller) เป็นส่วนซึ่งจะต้องรับคำสั่งจากผู้ใช้เพื่อนำมาควบคุมการทำงานต่างๆของการจินตทัศน์อัลกอริทึม

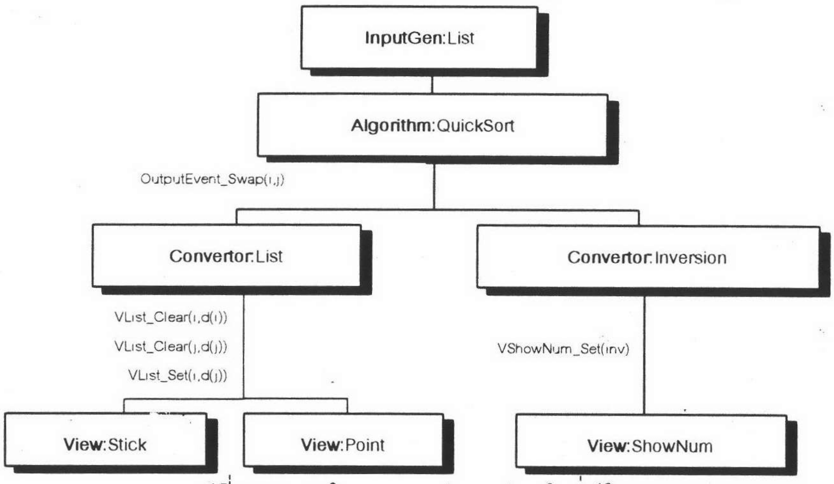
---

[15] S. Prasitjutrakul and W. Watcharawittaykul. Algorithm Visualization System: An Overview of Internal Structure. Proc. of Third ASEAN Regional Seminar on Microelectronics and Information Technology, Bangkok.

- ส่วนเชื่อมประสาน (Binder) ทำหน้าที่เชื่อมประสานส่วนประกอบอื่นๆซึ่งเป็นโปรแกรมบนวินโดว์ (EXE) ให้สามารถทำงานร่วมกันได้
- ส่วนประสานการทำงาน (Synchronizer) ทำหน้าที่ประสานให้อัลกอริทึมแต่ละอัน (หากในการจินตทัศน์นั้นมีอัลกอริทึมมากกว่าหนึ่งอัลกอริทึม) ทำงานไปพร้อมๆกันได้อย่างยุติธรรม

2.2.2 ตัวแปลงคำสั่ง

ตัวแปลงคำสั่งจะเป็นส่วนประกอบซึ่งแทรกระหว่างอัลกอริทึมและส่วนแสดงผล เพื่อทำให้ทั้งสองส่วนนี้มีความสัมพันธ์ระหว่างกันลดลง อันจะทำให้ผู้พัฒนาอัลกอริทึมไม่จำเป็นต้องกังวลถึงความสามารถของส่วนแสดงผลในขณะที่ออกแบบคำสั่งแสดงผลของส่วนอัลกอริทึม ทั้งนี้เพราะหากคำสั่งใดของส่วนอัลกอริทึมที่ส่วนแสดงผลไม่รู้จัก ตัวแปลงคำสั่งจะทำหน้าที่แปลงคำสั่งเหล่านั้นให้อยู่ในรูปที่ส่วนแสดงผลรู้จักและเข้าใจได้



รูปที่ 2.9 การทำงานของส่วนแปลงคำสั่ง[16]

รูปที่ 2.9 แสดงการจินตทัศน์อัลกอริทึมของการจัดเรียงข้อมูลแบบเร็ว (quick sort) เพื่อส่งผลการทำงานไปให้ส่วนแสดงผลสามรูปแบบคือ ส่วนแสดงผลที่แทนการเคลื่อนที่ของข้อมูลด้วยจุด (point view) ส่วนแสดงผลที่แทนการเคลื่อนที่ของข้อมูลด้วยแท่งสี (stick view) และส่วนแสดงผลที่จะแสดงถึงค่าการสลับตำแหน่ง (number of inversion) จากรูปจะเห็นว่าตัวแปลงคำสั่งจะทำการแปลงคำสั่ง OutputEvent_Swap(i,j) ออกเป็นคำสั่งซึ่งง่ายขึ้นและส่วนแสดงผลรู้จักเพื่อส่งให้ส่วนแสดงผลทำงานต่อไป ทำให้ผู้พัฒนาตัวอัลกอริทึมไม่ต้องกังวลว่าส่วนแสดงผลจะรู้จักคำสั่งแสดงผลที่ตนสั่งหรือไม่

2.2.3 ตัวเชื่อมประสาน

จากที่กล่าวมาข้างต้นแล้วว่าส่วนประกอบการจินตทัศน์จะถูกแบ่งออกเป็นส่วนประกอบย่อยๆ ตัวเชื่อมประสานจะทำหน้าที่เชื่อมส่วนประกอบเหล่านี้เข้าด้วยกันเพื่อให้ส่วนประกอบทั้งหมดสามารถทำงานเพื่อแสดงการจินตทัศน์อัลกอริทึมที่ต้องการได้ ตัวเชื่อมประสานจะทำหน้าที่เป็นผู้ส่งผ่านคำสั่งแสดงผลขององค์ประกอบต่างๆนั่นคือ แทนที่องค์ประกอบจะส่งข้อความคำสั่งข้างต้นโดยตรง ก็จะส่งข้อความคำสั่งมาให้ส่วนเชื่อม

---

[16] S. Prasitjutrakul and W. Watcharawittaykul. Algorithm Visualization System: An Overview of Internal Structure. Proc. of Third ASEAN Regional Seminar on Microelectronics and Information Technology, Bangkok.

ประสานก่อน จากนั้นส่วนเชื่อมประสานก็จะใช้ข้อมูลซึ่งตัวมันเองมีอยู่ โดยข้อมูลนี้อาจจะได้จากผู้ใช้ในขณะทำงานหรือจากบทการจินตทัศน์ที่เตรียมไว้ก่อนก็ได้ ตรวจสอบดูว่าจะต้องส่งข้อความคำสั่งเหล่านี้ไปให้ส่วนประกอบใด ดังนั้นผู้ที่พัฒนาส่วนประกอบต่างๆจึงไม่ต้องกังวลว่าในขณะทำงานจะต้องส่งข้อความคำสั่งให้ส่วนประกอบใดบ้างเพราะตัวเชื่อมประสานจะทำหน้าที่เหล่านี้ให้แทน

2.2.4 กลไกการส่งคำสั่งแบบขนาน

จากรูปที่ 2.9 จะเห็นว่าอัลกอริทึมหนึ่งๆสามารถต่อกับตัวแปลงคำสั่งมากกว่าหนึ่งตัวได้ หากการส่งข้อความคำสั่งของอัลกอริทึมไปให้ตัวแปลงคำสั่งเป็นไปแบบอนุกรม จะส่งผลการแสดงผลเป็นไปอย่างไม่สอดคล้องกัน ทั้งนี้เพราะส่วนแสดงผลบางอันอาจทำงานเร็วกว่าส่วนอื่น หากให้ส่วนอัลกอริทึมสามารถทำงานต่อได้ทันทีหลังจากส่งคำสั่งแสดงผลแต่ละคำสั่ง อัลกอริทึมอาจส่งคำสั่งแสดงผลคำสั่งใหม่ไปให้ส่วนแสดงผล ส่วนแสดงผลที่ทำงานตามคำสั่งเดิมเสร็จเรียบร้อยแล้วก็สามารถทำงานตามคำสั่งใหม่ แต่ส่วนแสดงผลที่ทำงานช้าซึ่งยังทำงานตามคำสั่งเดิมไม่เสร็จก็จะไม่สามารถทำงานตามคำสั่งใหม่ได้ในทันที ทำให้ดูเหมือนว่าส่วนแสดงผลแต่ละอันทำงานไม่สอดคล้องกัน ทางแก้ปัญหานี้ก็คือเราจะต้องออกแบบให้ระบบส่งคำสั่งแบบขนาน โดยเมื่อส่งคำสั่งไปให้ตัวแปลคำสั่งทุกตัวแล้วจะต้องหยุดรอจนกว่าตัวแปลงคำสั่งทุกตัวจะทำงานตามคำสั่งที่ได้รับเสร็จเรียบร้อยแล้วระบบจึงจะยอมให้ส่วนอัลกอริทึมทำงานต่อไปได้

2.2.5 ตัวประสานจังหวะการทำงาน

เนื่องจากระบบยินยอมให้การจินตทัศน์อัลกอริทึมหนึ่งๆประกอบไปด้วยอัลกอริทึมหลายๆอัลกอริทึมทำงานไปพร้อมๆกันได้ เพื่อเปรียบเทียบการทำงาน ระบบจึงจะต้องรับประกันให้ได้ว่าอัลกอริทึมแต่ละอัลกอริทึมมีโอกาสในการทำงานเท่าเทียมกัน เพื่อจะได้เปรียบเทียบการทำงานได้ว่าหากในการจินตทัศน์อัลกอริทึมใดทำงานเสร็จก่อน แสดงว่ามีประสิทธิภาพดีกว่า การที่จะออกแบบอัลกอริทึมให้ทำการเปรียบเทียบแบบนี้ได้ ผู้ออกแบบอัลกอริทึมจะต้องกำหนดคำสั่งการทำงานพื้นฐาน (basic operation) ของอัลกอริทึมเสียก่อน จากนั้นให้ใช้จำนวนครั้งซึ่งคำสั่งการทำงานพื้นฐานถูกเรียกใช้งานเป็นเครื่องวัดประสิทธิภาพการทำงานของอัลกอริทึม เช่น คำสั่งการทำงานพื้นฐานของการจัดเรียงข้อมูลคือการเปรียบเทียบค่า เมื่อกำหนดเช่นนี้ได้แล้วเมื่อใดก็ตามที่มีการเปรียบเทียบค่าในอัลกอริทึมก็จะทำการแจ้งเตือนว่า ตัวประสานจังหวะการทำงานอัลกอริทึมได้ทำงานไปหนึ่งการทำงานพื้นฐานแล้ว (ในรูปที่ 2.10 จะใช้การเรียกฟังก์ชัน SyncNotify เป็นการแจ้งเตือน) เพื่อให้ตัวประสานการทำงานสลับการทำงานไปยังอัลกอริทึมอื่นๆต่อไป เมื่อทุกอัลกอริทึมทำงานเสร็จแล้วก็จะเริ่มวนเช่นนี้กลับมายังอัลกอริทึมแรกอีกครั้ง ทำเช่นนี้เรื่อยไปจนกว่าอัลกอริทึมทุกตัวจะทำงานเสร็จ

ในรูปที่ 2.10 แสดงตัวอย่างโปรแกรมต้นฉบับของอัลกอริทึมจัดเรียงข้อมูลแบบเลือกเพื่อทำกสรจัดเรียงข้อมูลของเลขจำนวนเต็ม โดยกำหนดให้คำสั่งการทำงานพื้นฐานของอัลกอริทึมนี้คือคำสั่งเปรียบเทียบค่าของจำนวนเลขที่นำมาจัดเรียงกัน

```
Alg_SelectionSort_Code(hAlg)
{
    int i,n, iMaxVal,iMaxPos, iData[MAXN];
    n = InputEvent_HowManyKeys();
    for(i=0;i<n;i++) InputEvent_ReadKey(iData[i]);
    for(i=0;i<n;i++) OutputEvent_NewKey(i,iData[i]);
    for(i=n-1;i>=1;i--) {
        iMaxPos = i;
        for(j=0;j<=i;j++) {
            if ( iData[iMaxPos] < iData[i] ) iMaxPos = j;
            SyncNotify(hAlg);
        }
        swap(iData,i,iMaxPos);
        OutputEvent_Swap(i,iMaxPos);
    }
}
```

รูปที่ 2.10 ตัวอย่างโปรแกรมต้นฉบับ[17]

2.3 สรุปและวิเคราะห์

จากงานวิจัยทั้งสองชิ้นจะพบว่าจะมีการแบ่งการจินตทัศน์อัลกอริทึมออกเป็นส่วนประกอบย่อยๆในลักษณะที่คล้ายกัน และทำไปด้วยเหตุผลซึ่งเหมือนกันเป็นส่วนใหญ่ สิ่งที่แตกต่างกันระหว่างงานวิจัยทั้งสองคือ ใน BALSA-II นั้นส่วนประกอบที่แยกออกมาจะอยู่ในรูปของฟังก์ชันย่อยซึ่ง BALSA-II จะเป็นผู้เรียกมาทำงาน ซึ่งทำให้การจินตทัศน์อัลกอริทึมที่พัฒนาด้วยระบบ BALSA-II ทำงานได้เร็วกว่าระบบของดร.สมชาย ประสิทธิจูตระกูล และ ดร.วิทยา วัชระวิทยากุลซึ่งกำหนดให้ส่วนประกอบการจินตทัศน์อัลกอริทึมแต่ละส่วนเป็นโปรแกรม แต่ก็มีข้อเสียคือ การเลือกส่วนประกอบต่างๆมาใช้งานต้องทำในขั้นของการพัฒนาการจินตทัศน์ ทั้งนี้เพราะต้องนำส่วนประกอบทั้งหมดมาเชื่อมรวมกันเป็นโปรแกรมเพียงโปรแกรมเดียวซึ่งจะมีขนาดใหญ่มาก นอกจากนี้การปรับปรุงแก้ไขส่วนประกอบแต่ละส่วนในภายหลังทำได้ยากมาก หรืออาจทำไม่ได้เลย หากไม่สามารถหารหัสจุดหมาย (object code) ของส่วนประกอบอื่นๆมาเพื่อเชื่อมรวมกันได้ และการที่ส่วนประกอบทั้งหมดรวมกันเป็นโปรแกรมเดียว หากส่วนประกอบใดทำงานผิดพลาดก็จะส่งผลให้ระบบทั้งหมดหยุดทำงานได้ ในขณะที่ระบบของดร.สมชาย ประสิทธิจูตระกูล และ ดร.วิทยา วัชระวิทยากุล ซึ่งกำหนดให้ส่วนประกอบของการจินตทัศน์พัฒนาขึ้นทำงานในลักษณะของโปรแกรมซึ่งแยกจากกันต่างหาก แล้วทำการติดต่อประสานงานกันผ่านตัวเชื่อมประสานซึ่งเป็นคลังคำสั่งร่วมแบบพลวัต วิธีการนี้มีข้อเสียก็คือทำงานได้ช้ากว่าวิธีการเชื่อมรวมกันของ BALSA-II มาก เนื่องจากต้องมีการติดต่อระหว่างโปรแกรม (interprocess communication) แต่ก็มีข้อดีคือการแก้ไขส่วนประกอบใดๆทำได้ง่าย ไม่จำเป็นต้องนำส่วนประกอบอื่นมาเชื่อมรวมกัน ทั้งนี้เพราะการเชื่อมต่อจะกระทำในขณะทำงานอยู่แล้ว นอกจากนี้ก็ยังมีความมั่นคงกว่า เพราะหากส่วนประกอบใดทำงานผิดพลาดจนหยุดทำงาน ส่วนประกอบอื่นก็ยังสามารถทำงานต่อไปได้

ข้อแตกต่างอีกประการหนึ่งคือ ในระบบ BALSA-II ไม่ได้ออกแบบให้การทำงานของอัลกอริทึมต่างๆทำงานไปพร้อมๆกันได้อย่างยุติธรรม โดยการกำหนดให้มีการสลับการทำงาน (task switching) เมื่อมีการทำงานตามการทำงานพื้นฐาน แต่กำหนดให้สลับการทำงานทุกครั้งที่อัลกอริทึมสร้างเหตุการณ์ขึ้น ซึ่งในบางครั้งอาจไม่

---

[17]S. Prasitjutraku and W. Watcharawittaykul. Algorithm Visualization System: An Overview of Internal Structure. Proc. of Third ASEAN Regional Seminar on Microelectronics and Information Technology, Bangkok.

มีความยุติธรรมพอ เนื่องจากความเร็วที่ได้จะขึ้นกับจำนวนเหตุการณ์ที่อัลกอริทึมสร้าง ไม่ใช้จำนวนการทำงานพื้นฐานที่มักจะใช้ในการวิเคราะห์การทำงานของอัลกอริทึมโดยทั่วไป