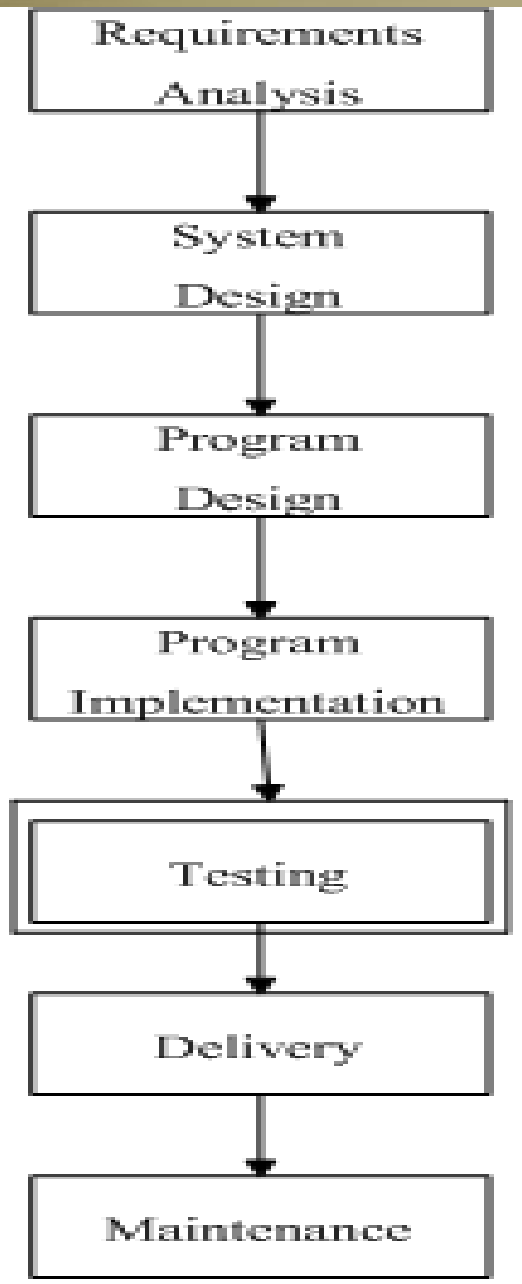
ครั้งที่ 8

การทดสอบโปรแกรม

Logo



What is the problem

What is the solution

What are the mechanisms that best implement the solution?

How is the solution constructed?

Is the problem solved

Can the customer use the solution?

Are enhancements needed?

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดของอัลกอริทึม (**Algorithm error**)
- เป็นความผิดพลาดที่เกิดจากการเขียนขั้นตอนการทำงานผิด เช่น
  - ใช้คำสั่งเงื่อนไขในการทำงานผิดมีการกระโดดข้ามไปยังคำสั่งที่ผิด
  - ลืมกำหนดค่าเริ่มต้นให้กับตัวแปร
  - ลืมทดสอบเงื่อนไขที่มีการหารด้วยศูนย์
  - ทำการเปรียบเทียบข้อมูลที่มีชนิดต่างกัน ซึ่งส่งผลให้การทำงานผิดพลาดทั้งสิ้น

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดด้านไวยากรณ์ (**Syntax error**)
- ความผิดพลาดในลักษณะนี้สามารถตรวจสอบพบในขณะที่ทำการแปลโปรแกรม เช่น
  - ลืมใส่เครื่องหมายในคำสั่งโปรแกรม
  - พิมพ์ตัวแปรผิด เป็นต้น

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดทางด้านการคำนวณ

  (Computation and precision errors)

- เป็นความผิดพลาดจากการคำนวณที่ผิดพลาดโดยเฉพาะค่าที่เป็นเลขทศนิยม ซึ่งผลลัพธ์ที่ได้จะมีความคลาดเคลื่อนเกิดขึ้นถ้าต้องการความเที่ยงตรงมากๆ หรืออาจเป็นการใช้สูตรต่างๆในการคำนวณผิด

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดของเอกสาร (**Documentation error**)
- เอกสารในที่นี้อาจเป็นการบรรยายในโปรแกรมหรือเอกสารที่จัดทำขึ้น ซึ่งการบรรยายในเอกสารกับการทำงานของระบบจริงๆ ไม่ตรงกัน

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดที่เกิดจากการปฏิบัติงานเกินสิ่งที่กำหนดไว้ (**Stress and Overload errors**)
  - เช่น โครงสร้างข้อมูลที่ออกแบบไว้มีความจุไม่เพียงพอ ประกาศตัวแปรชนิดแถวไว้ 20 ช่องแต่มีการระบุการทำงานในช่องที่ 21 ซึ่งเกินสิ่ง ที่กำหนดไว้เป็นต้น

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดที่ระบบทำงานเกินประสิทธิภาพของระบบ **(Capacity or Boundary errors)**
  - เช่น การทำงานกับโปรแกรมที่มีขนาดใหญ่เกินกว่าที่ระบบ เครื่องจะปฏิบัติการได้ หรือการใช้ตัวเลขจำนวนเต็มเกิน 32767 ซึ่งระบบไม่สามารถกระทำได้ เป็นต้น

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดของเวลา(**Timing or Coordination errors**) เกิดขึ้นในระบบ
  - real time ซึ่งมีการใช้คำสั่งร่วมกันโดยการเข้าถึงข้อมูลเดียวกันแล้วเกิดการทำงานที่ไม่สอดคล้องกัน

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดในเรื่องประสิทธิภาพ (**Throughput** หรือ **Performance errors**)
- เป็นความผิดพลาดที่ระบบไม่สามารถกระทำงานได้ตามกฎเกณฑ์ที่ระบุไว้
  - เช่นความเร็วในการทำงานต่ำกว่าที่กำหนด
  - อัตราตอบสนองต่ำกว่าเกณฑ์ที่กำหนดเป็นต้น

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดที่เกิดขึ้นโดยบังเอิญ(**Recovery errors**)
- เป็นความผิดพลาดที่เกิดขึ้นโดยทันทีทันใด ไม่ได้มีการคาดคิดมาก่อน เช่น
  - ไฟฟ้าดับ
  - ไฟฟ้าตก ขณะระบบทำงาน ซึ่งการแก้ไขเพียงแต่ใช้ไฟฟ้าสำรองทำให้ระบบสามารถดำเนินต่อไปได้ เป็นต้น

ชนิดของความผิดพลาด (Type Of Error)

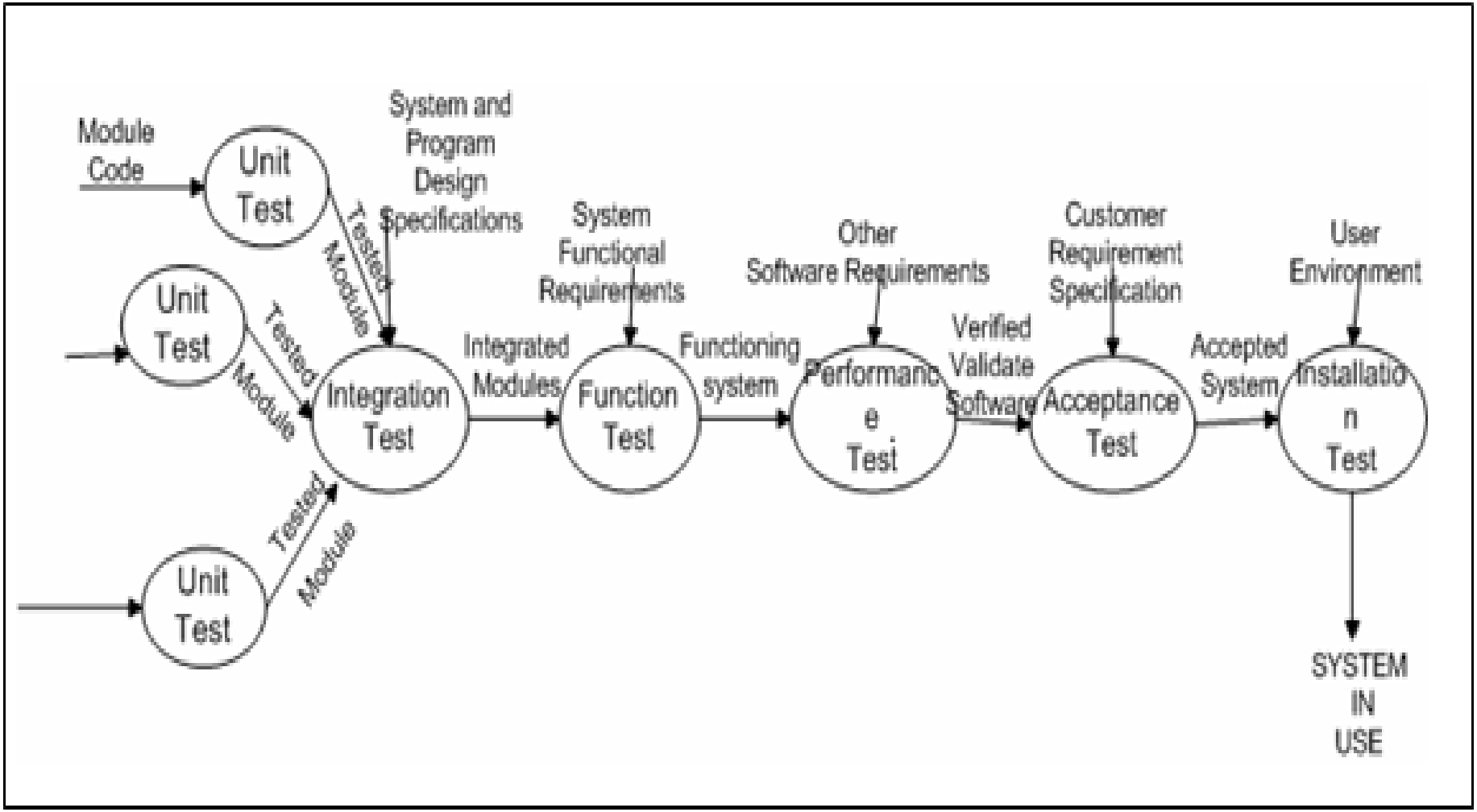
- ความผิดฮาร์ตแวร์และซอฟต์แวร์

  (Hardware and System software errors)
- โดยเกิดจากเอกสารที่ระบุถึงฮาร์ตแวร์และซอฟต์แวร์ที่ใช้ในระบบไม่สอดคล้องกับเงื่อนไขในการปฏิบัติการและกระบวนการ

ชนิดของความผิดพลาด (Type Of Error)

- ความผิดพลาดในเรื่องของมาตรฐานและกระบวนการ **(Standard and Procedure errors)**
- ความผิดพลาดในเรื่องนี้ไม่ส่งผลต่อการประมวลผลโปรแกรม แต่มีผลต่อสภาพแวดล้อมที่เกี่ยวข้อง ซึ่งส่งผลต่อการทดสอบหรือแก้ไขระบบ เมื่อกระบวนการไม่เป็นไปตามที่กำหนด การทำความเข้าใจตรรกของโปรแกรม การค้นหาข้อมูลหรือสารสนเทศที่ต้องการเป็นไปโดยลำบาก

รูปภาพที่ 7.2 ระยะของการทดสอบระบบ



ระยะของการทดสอบ (Stages of testing)

- Module testing หรือ Unit testing เป็นการทดสอบเฉพาะหน้าที่หลักของแต่ละโมดูลโปรแกรม เช่น
  - ตรวจสอบโมดูลการซื้อสินค้า โ
  - มดูลการขายสินค้า ว่าสามารถทำงานได้ตามที่กำหนดหรือไม่

ระยะของการทดสอบ (Stages of testing)

- Integration testing เป็นการทดสอบการทำงานของโมดูลโปรแกรมทั้งหมด
- โดยนำโมดูลทั้งหมดมาทดสอบรวมกัน เพื่อให้แน่ใจว่าสารสนเทศต่างๆที่ส่งผ่านภายในระบบสามารถกระทำได้อย่างถูกต้อง

ระยะของการทดสอบ (Stages of testing)

- **Function testing** เป็นการทดสอบมุ่งเน้นไปที่การทำงานตามที่ลูกค้าต้องการ
- โดยตรวจสอบจากเอกสารระบุความต้องการ โดยทำการเปรียบเทียบระหว่างระบบที่สร้างขึ้นกับเอกสารที่ระบุความต้องการ

ระยะของการทดสอบ (Stages of testing)

- Performance testing เป็นการทดสอบขอบเขตของการทำงาน ประสิทธิภาพของระบบ โดยตรวจสอบในสภาพแวดล้อมจริง (Validated) หรือสภาพแวดล้อมจำลอง**(Verified)** ก็ได้

ระยะของการทดสอบ (Stages of testing)

- Acceptance testing เป็นการทดสอบเพื่อให้ลูกค้ามั่นใจ และพอใจ โดยทดสอบระบบที่พัฒนาขึ้นเพื่อให้ลูกค้าตรวจสอบว่าสามารถกระทำได้ตามความต้องการของลูกค้าหรือไม่

ระยะของการทดสอบ (Stages of testing)

- Installation testing เป็นการทดสอบระบบเพื่อให้แน่ใจว่าสามารถกระทำงานได้จริง โดยทดสอบการติดตั้งระบบ ในสภาพแวดล้อมจริง

UNIT TESTING

- **การตรวจโปรแกรม (Program Reviews)**
- การตรวจสอบคำสั่งโปรแกรมและเอกสารของโมดูลโปรแกรมนั้นว่ามีความถูกต้องหรือไม่

การตรวจโปรแกรม (Program Reviews)

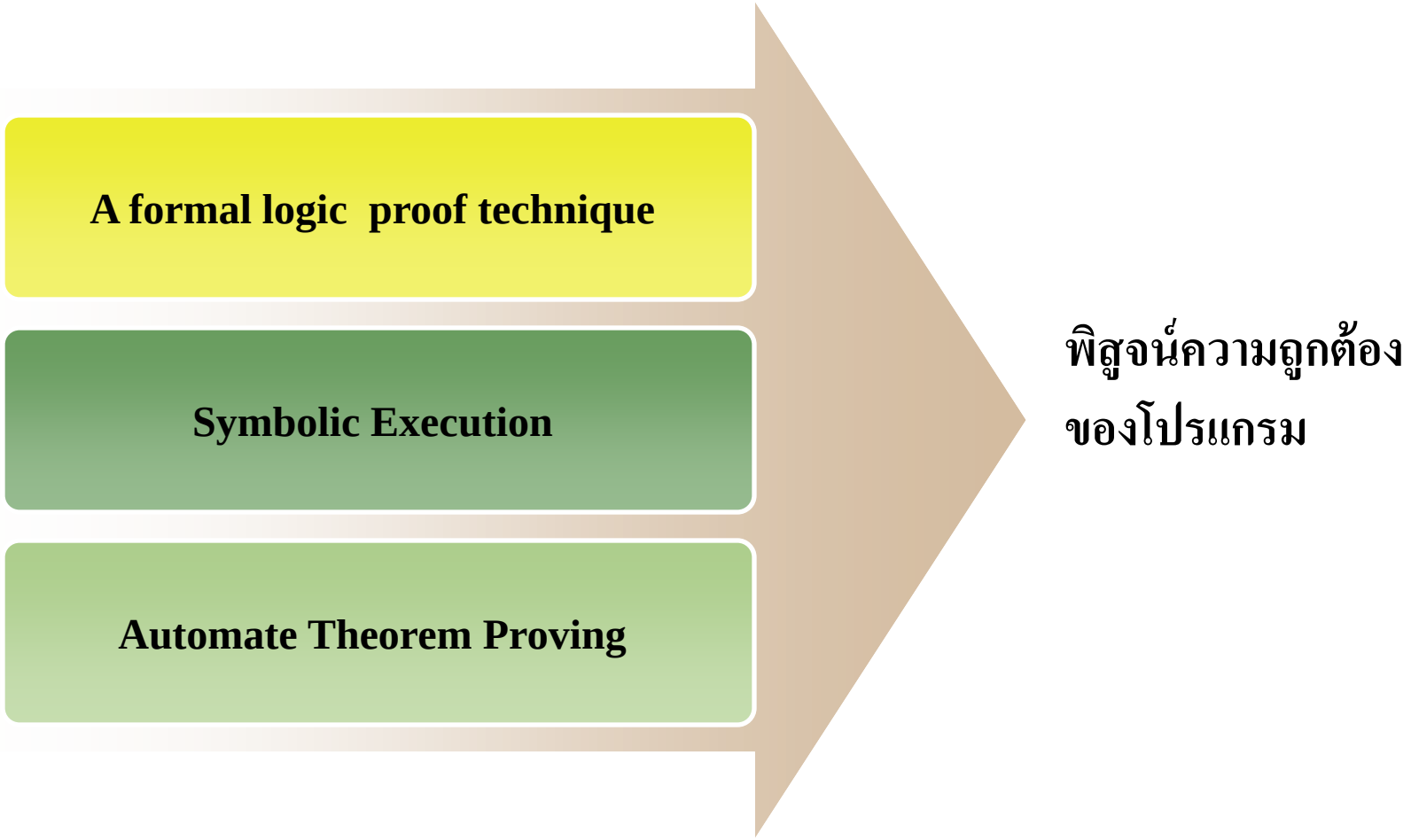
Program Walk-through

ผู้พัฒนาจะทยอยนำคำสั่งโปรแกรมและเอกสารที่ได้พัฒนาเสนอให้กับทีมงานตรวจสอบ เพื่อตรวจสอบดูความถูกต้อง

Program lnspection

การนำเสนอในรูปแบบที่เป็นทางการแก่ทีมงานตรวจสอบ โดยการตรวจสอบวิธีนี้ต้องพัฒนาโปรแกรมโมดูลให้เสร็จเรียบร้อยก่อน

วิธีการในการพิสูจน์ความถูกต้องของโปรแกรมสามารถกระทำได้ดังนี้



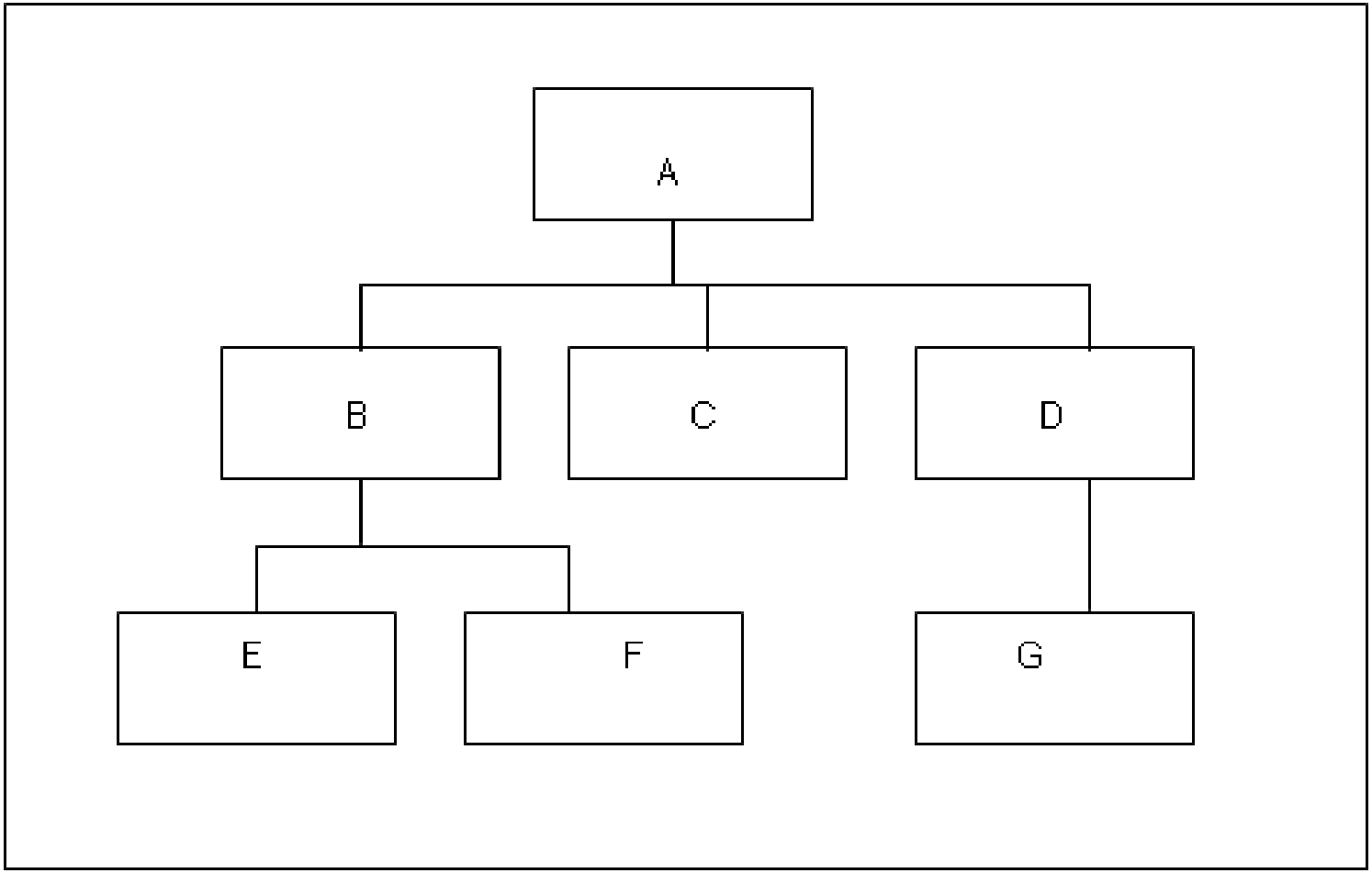
Automate Theorem Proving

- เป็นวิธีการทดสอบโมดูลซึ่งมีการพัฒนาโมดูลพิเศษ เพื่อใช้ตรวจสอบความถูกต้องของโปรแกรมโดยจะทำการอ่าน
  - ข้อมูลเข้าและเงื่อนไข
  - ข้อมูลออกและเงื่อนไข
  - จำนวนบรรทัดที่ถูกทดสอบ
  - ผลลัพธ์ที่ได้จากการทดสอบโมดูล จะเป็นกลุ่มของข้อมูลที่ผิดพลาด

INTEGRATION TESTING

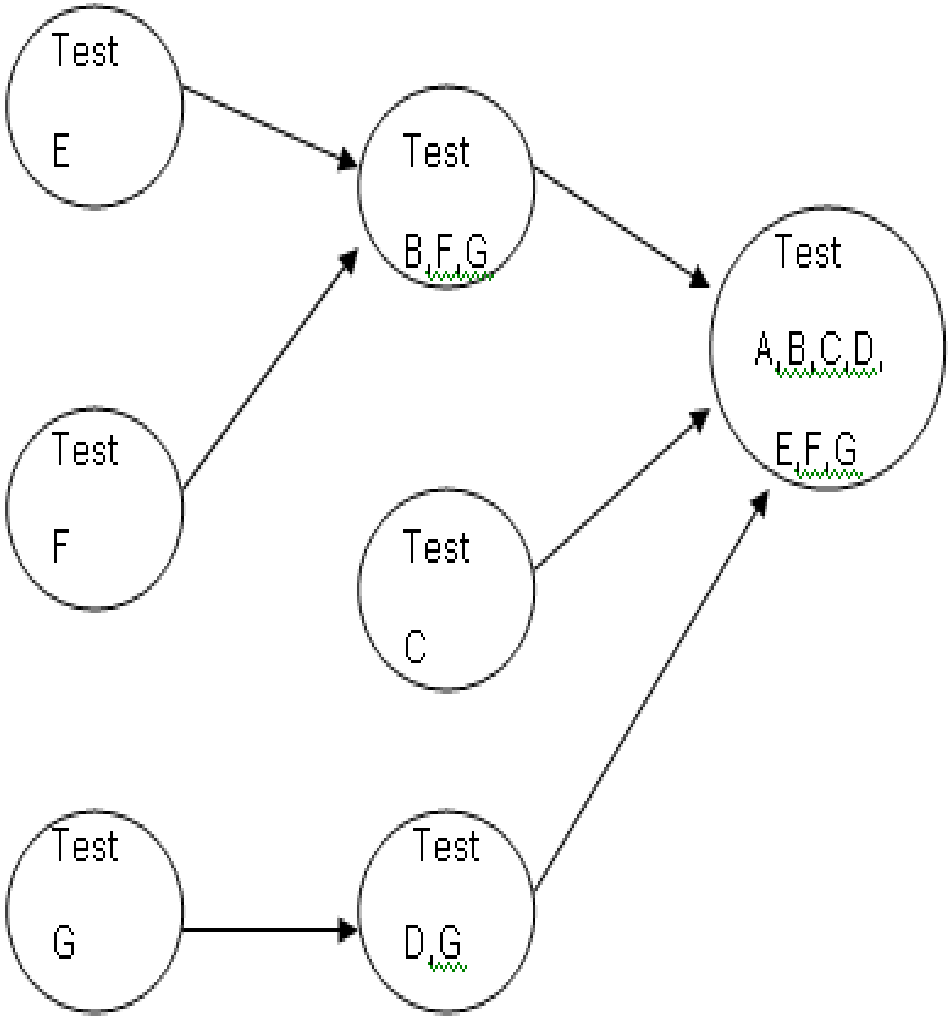
- เป็นการทดสอบการทำงานของโปรแกรมโดยรวม(Integration Testing) หลังจากมีการทดสอบในแต่ละโมดูลแล้ว วัตถุประสงค์เพื่อทดสอบโปรแกรมทั้งระบบว่าสามารถทำงานอย่างถูกต้องและตรงตามวัตถุประสงค์หรือไม่

รูปภาพที่ 7.6 แสดงลำดับชั้นของโมดูลต่างๆของระบบงานหนึ่ง



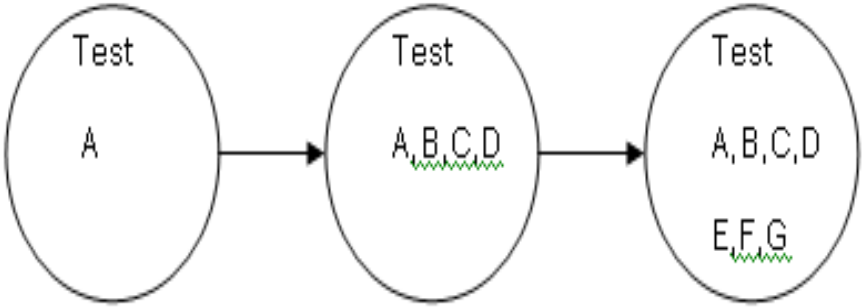
Bottom-up approach

- วิธีนี้เป็นการทดสอบโมดูลโดยเริ่มจากระดับชั้นต่ำที่สุดโดยแยกทดสอบ จากนั้น จะทำการทดสอบรวมกับโมดูลระดับชั้นที่สูงขึ้นไปที่มีความสัมพันธ์กันมีการส่งผ่านข้อมูลระหว่างกัน และทดสอบในระดับชั้นสูงขึ้นไปเรื่อยๆจนเป็นโปรแกรมใหญ่หรืออยู่ในระดับบนสุด ซึ่งเป็นการทดสอบทุกโมดูลโปรแกรมพร้อมกัน ซึ่งวิธีนี้จะใช้ประโยชน์เมื่อมีโมดูลลำดับต่ำมากๆ



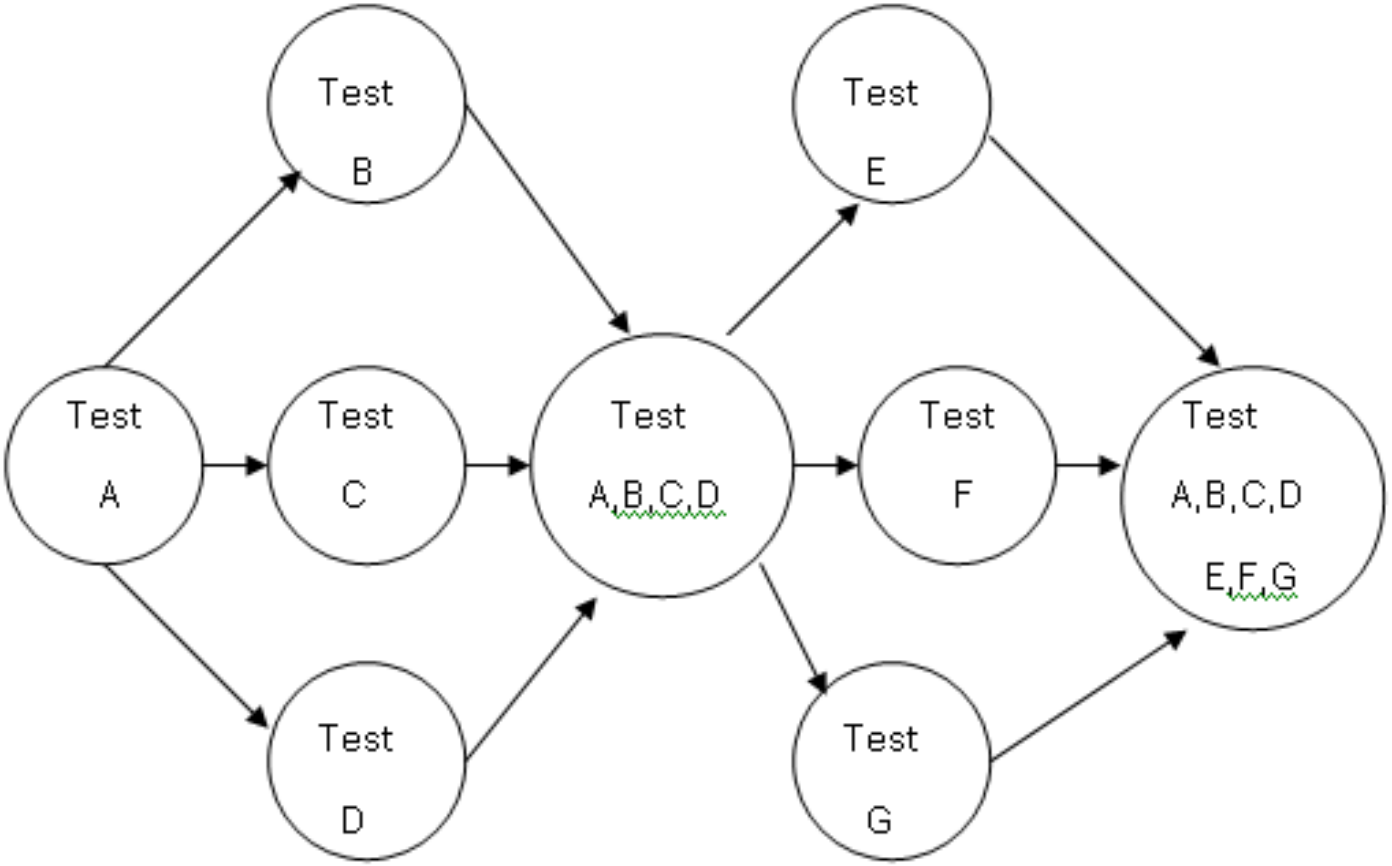
Top-down approach

- จะทำการทดสอบในระดับชั้นที่สูงที่สุดก่อน ต่อจากนั้นจะรวมโมดูลทั้งหมดในระดับชั้นรองลงมาเพื่อทดสอบการทำงาน และจะรวมโมดูลในระดับรองลงมาเพื่อทดสอบจนกระทั่งถึงระดับชั้นที่ต่ำที่สุด
- ปัญหาที่เกิดขึ้นคือ Stub
- Stub คือโปรแกรมที่จำลองกิจกรรมของโมดูลที่หายไป โดยการตอบรับกิจกรรมที่มีการเรียกใช้ เพื่อประมวลผลการทดสอบต่อไป



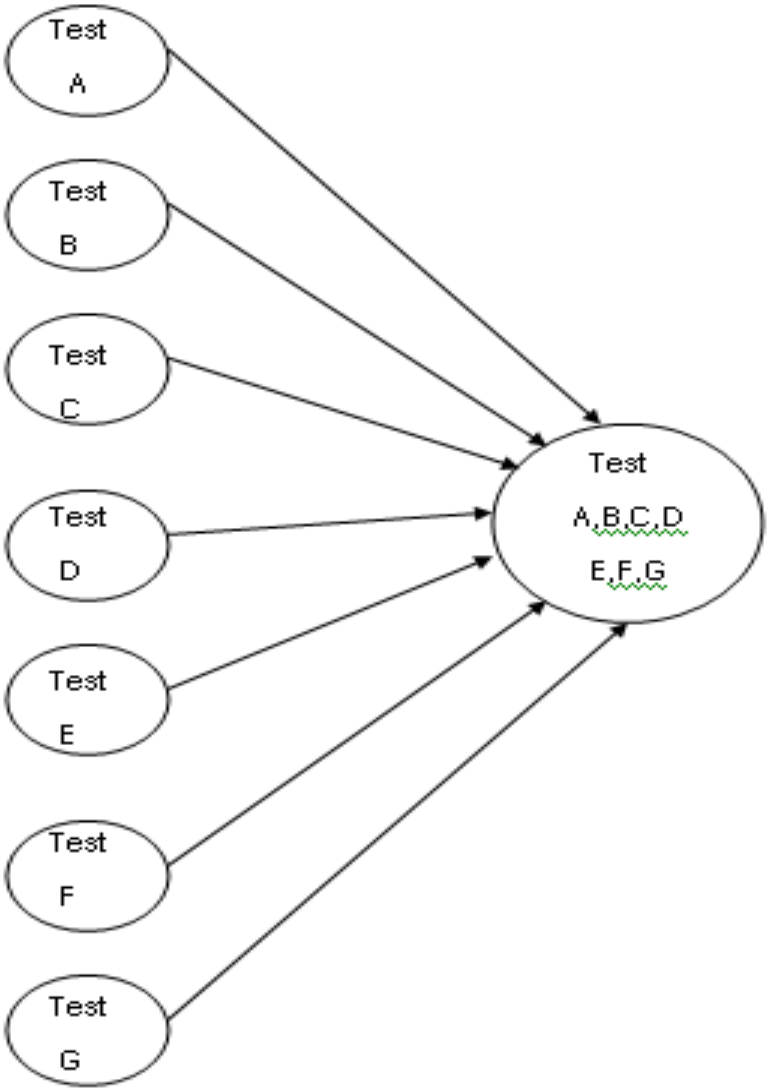
Modified Top-down Testing

- การแก้ไขโดยขจัด Stub ออกไปโดยปรับเปลี่ยนการทดสอบโมดูลในระดับต่ำ โดยแยกการทดสอบแต่ละโมดูลก่อนที่จะรวมกับในระดับที่สูงกว่า



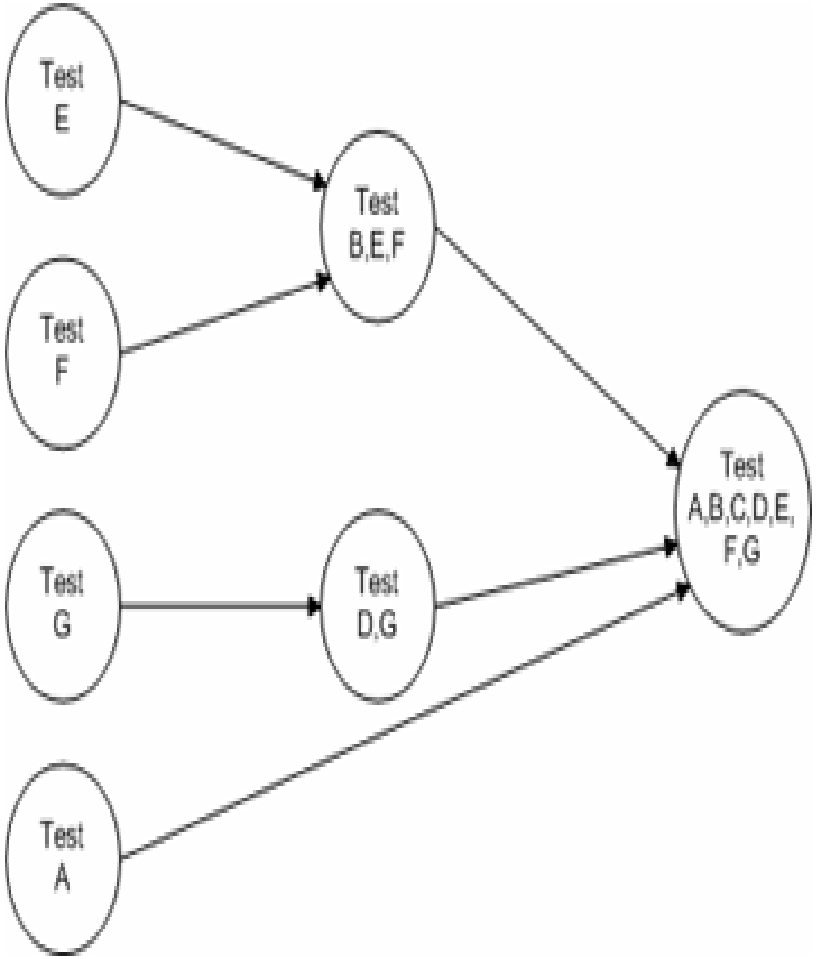
Big-bang Testing

- นำทุกโมดูลโปรแกรมทดสอบแยกหมดทุกตัว ต่อจากนั้นจึงนำโมดูลทั้งหมดมารวมกันทดสอบเพียงครั้งเดียว



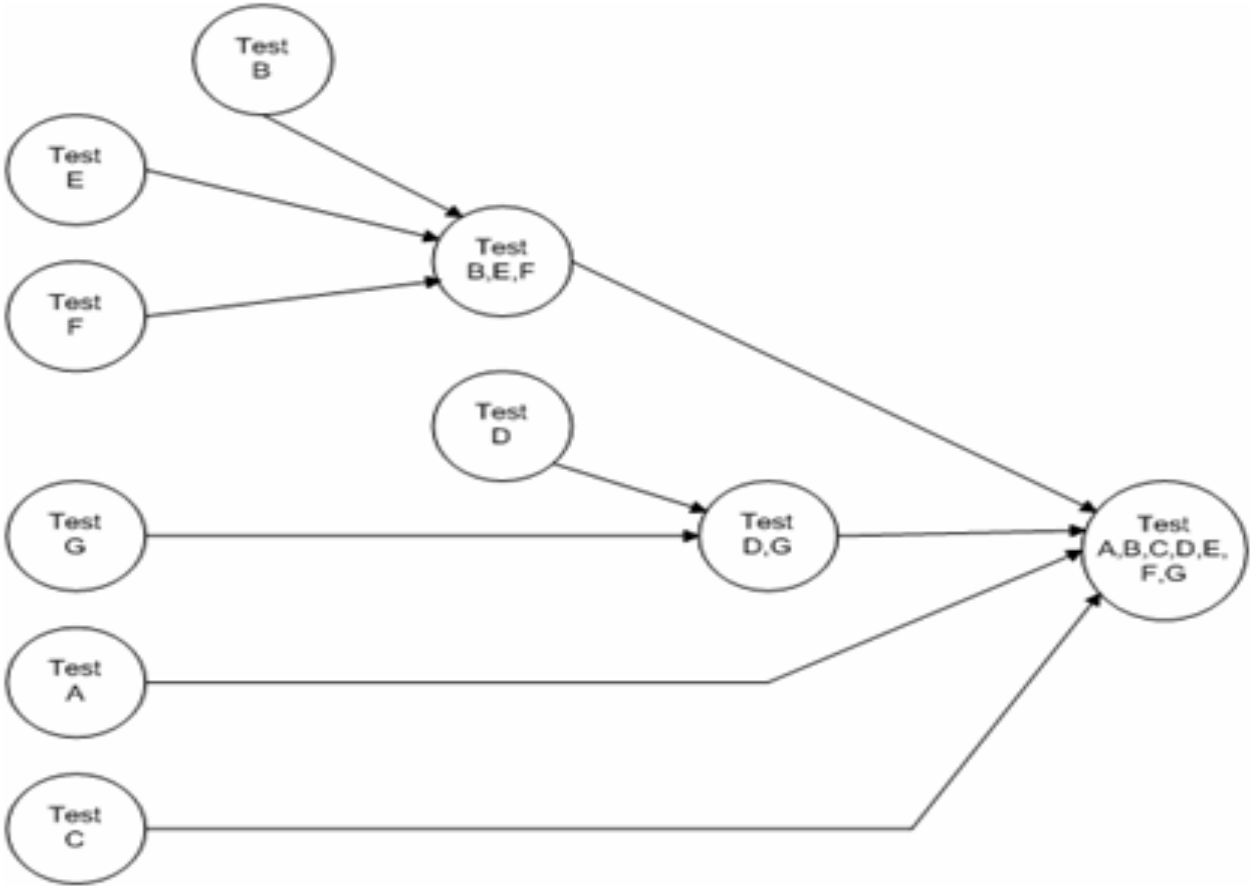
Sandwich Testing

- เป็นวิธีการทดสอบที่ผสมผสานทั้งแบบ Top-down และ Bottom-up เป็นการมองระบบเป็น 3 ระดับ เหมือนกับการทำแซนวิช โดยการทดสอบวิธี Top-down ใช้สำหรับทดสอบกับโมดูลโปรแกรมในระดับบนสุด และ วิธีการทดสอบแบบ Bottom-up ใช้กับการทดสอบโมดูลโปรแกรมในระดับต่ำสุด การทดสอบจะมารวมกันที่ Target Layer (ระดับกลาง)



Modified Sandwich Testing

- เป็นการปรับปรุงวิธีของ Sandwich Testing โดยขจัด Stub ออกจากการทดสอบ โดยทำการทดสอบโมดูลแยกเป็นอิสระก่อนที่จะนำมาทดสอบรวม



Comparison of Integration Strategies

	Bottom-Up	Top-Down	Modified Top-down	Big-Bang	Sandwitch	Modified Sandwitch
Integration	Early	Early	Early	Late	Early	Early
Time to basic Working Program	Late	Early	Early	Late	Early	Early
Module	Yes	No	Yes	Yes	In part	Yes
Drivers needed						
Stubs needed	No	Yes	Yes	Yes	In part	In part
Work paaral-Lelism at Beginning	Medium	Low	Medium	High	Medium	High
Ability to test Particular Parts	Easy	Hard	Easy	Easy	Medium	Easy
Ability to Plan and Control sequence	Easy	Hard	Hard	Easy	Hard	Hard

AUTOMATED TESTING TOOLS AND TECHNIQUES

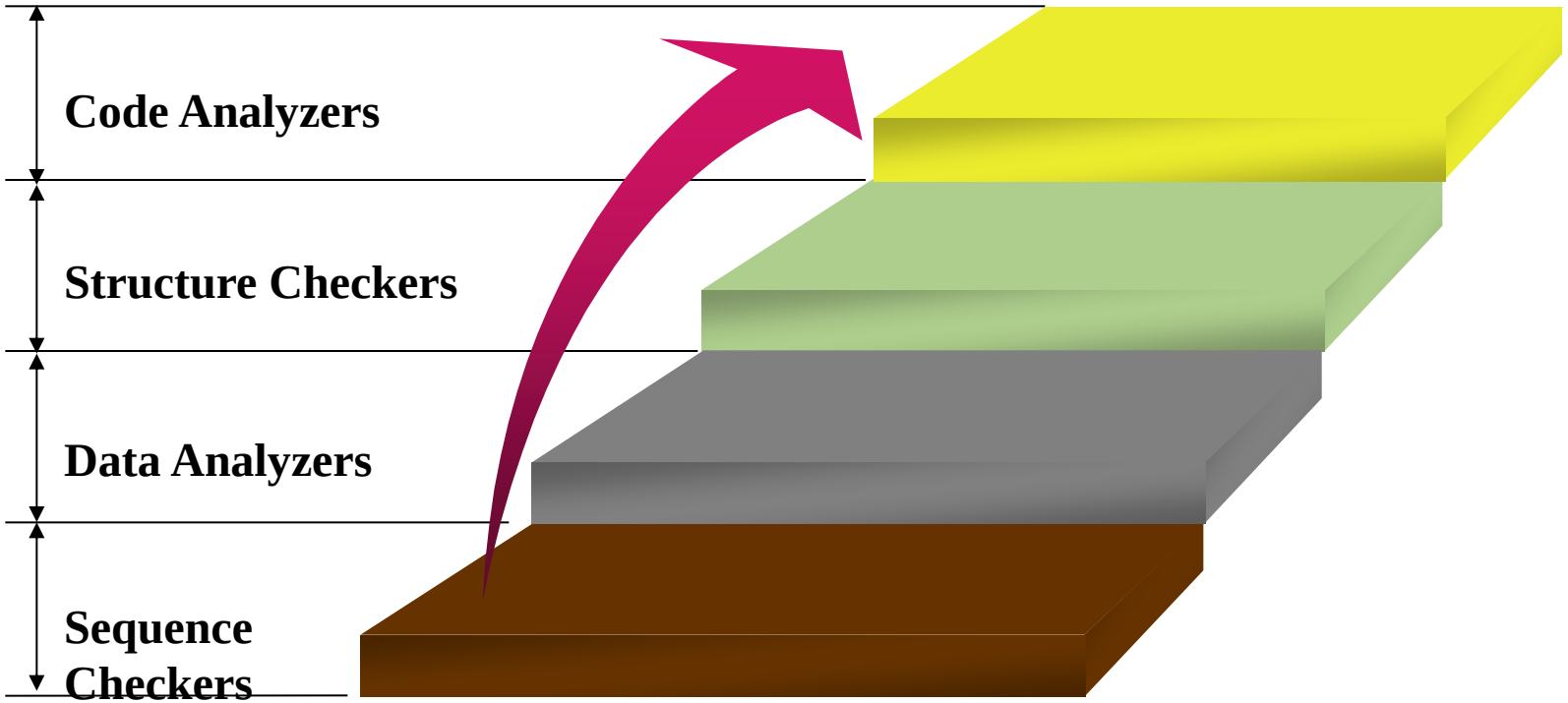
Static Analysis

เป็นเครื่องมือที่ช่วยทีมงานทดสอบและโปรแกรมเมอร์ ในการวิเคราะห์ความถูกต้องของโปรแกรมก่อนการทำงาน **(run)**

Dynamic Analysis

เครื่องมืออีกชนิดหนึ่งที่ใช้ในการทดสอบโมดูลและทดสอบการรวมโมดูล ซึ่งสามารถตรวจสอบความผิดพลาดในระหว่างปฏิบัติงาน

Static Analysis



Code Analyzers

- เป็นโมดูลทดสอบที่ทำหน้าที่วิเคราะห์ความถูกต้องของรูปแบบภาษาหรือไวยากรณ์ภาษาโดยอัตโนมัติ ประโยคหรือคำสั่งใดที่ผิดรูปแบบภาษาจะถูกกำหนดแถบสว่างเพื่อบอกถึงตำแหน่งที่ผิด

Structure Checkers

- เป็นเครื่องมือที่ทดสอบโครงสร้างของโมดูล โดยสร้างเป็นกราฟ เป็นลำดับชั้นของโมดูลหรือกลุ่มของโมดูล เพื่อหาข้อบกพร่องของโครงสร้าง

Data Analyzers

- เป็นเครื่องมือตรวจสอบโครงสร้างข้อมูล ,การประกาศข้อมูลและการโต้ตอบของโมดูล เครื่องมือชนิดนี้จะบันทึกความผิดพลาดที่เกิดขึ้นระหว่าง โมดูล รวมทั้งการกำหนดข้อมูลและการใช้ข้อมูลที่ผิดๆ

Sequence Checkers

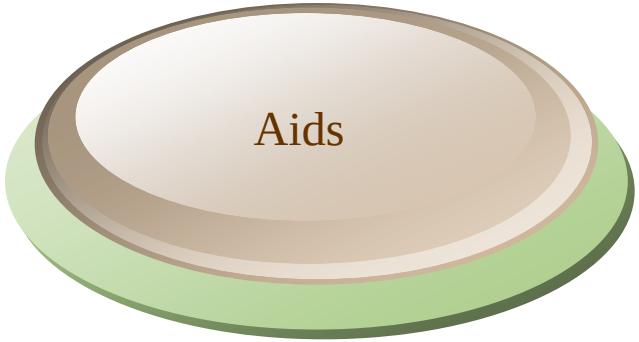
- เป็นเครื่องมือที่ตรวจสอบลำดับของเหตุการณ์ต่างๆที่เกิดขึ้นในโมดูล ถ้าผิดจะทำเครื่องหมายไว้

Dynamic Analysis

- เครื่องมือที่ใช้ในการทดสอบโมดูลและทดสอบการรวมโมดูล ซึ่งสามารถตรวจสอบความผิดพลาดในระหว่างปฏิบัติงาน
- พัฒนาขึ้นมาเพื่อช่วยในกรณีที่ยากต่อการทดสอบระบบ ยากที่จะคาดหมายเงื่อนไขและยากต่อการสร้างจุดทดสอบ
- เครื่องมือชนิดนี้พัฒนาขึ้นมาเพื่อให้ทีมงานทดสอบสามารถทราบระยะของเหตุการณ์ระหว่างการทำงานของโปรแกรมได้ บางครั้งเรียกว่า Program monitor ซึ่งจะทำหน้าที่มอง และรายงานอาการของโปรแกรม รายงานเวลาที่ปฏิบัติงานในแต่ละโมดูล รายงานจุดตัดสินใจที่มีการกระโดดข้ามไปในโปรแกรม เป็นต้น
- Monitor สามารถบันทึกจำนวนครั้งที่โมดูลย่อยถูกเรียก หรือบรรทัดของคำสั่งที่ถูกปฏิบัติการ ซึ่งข่าวสารต่างๆที่ได้รับนี้ช่วยทีมงานทดสอบในการคำนวณประสิทธิภาพของระบบ สถิติต่างๆในการปฏิบัติสามารถที่ใช้สร้างตัวแปรเฉพาะ เช่น ค่าเริ่มต้น,ค่าสุดท้าย,ค่าน้อยสุด, ค่ามากสุด หรือ Breakpoint ในการทดสอบ เป็นต้น

Logo

Other Automated Aids



ฐานข้อมูล
ทีมงานทดสอบสามารถใช้สำหรับบันทึกกรณีทดสอบ โดยเก็บข้อมูลเข้าสำหรับทุกกรณีทดสอบ และอธิบายผลลัพธ์ที่คาดหวัง ทีมทดสอบสามารถใช้ฐานข้อมูลเพื่อเก็บผลลัพธ์ที่เป็นจริงไว้

Test harness
ระบบเตือนอัตโนมัติ ซึ่งบันทึกการทดสอบของข้อมูลเข้าที่ผ่านเข้าไปในโปรแกรม สามารถใช้เปรียบเทียบผลลัพธ์ที่คาดหวังได้จากรายงานที่ได้บันทึกไว้

The Test Life Cycle

- สร้างจุดประสงค์การทดสอบ เนื่องจากการทดสอบมีด้วยกันหลายขั้นตอน การทดสอบนั้นเพื่อแสดงถึงการทำงานของระบบว่าสามารถทำงานได้อย่างถูกต้องตรงกับสิ่งที่ลูกค้าต้องการ ดังนั้นผู้ทดสอบต้องทราบถึงความต้องการ รายละเอียดหน้าที่การทำงาน ระดับชั้นของโมดูลต่างๆ ซึ่งต้องวางแผนการทดสอบถึงการดำเนินการ เกณฑ์การทดสอบ โดยวางแผนถึงกิจกรรมและตารางการทดสอบ
- ออกแบบกรณีทดสอบ(Test case) เป็นการออกแบบข้อมูลเข้าที่ใช้ในการทดสอบโมดูลหรือ เงื่อนไขในการปฏิบัติงาน
- การเขียนกรณีทดสอบ(Test case) โดยเขียนเป็น ตัวขับ (Driver) โดยใช้ภาษาโปรแกรม
- ทดสอบกรณีทดสอบ (Test case) มีความถูกต้องตรงวัตถุประสงค์ที่ได้วางแผนไว้ ข้อมูลมีความถูกต้อง
- ปฏิบัติการทดสอบ ซึ่งอาจได้เป็นรายงานในการทดสอบ
- คำนวณหาผลลัพธ์ของการทดสอบ

Estimating Software Quality

- ความน่าเชื่อถือ(Reliability)
- สามารถหาได้ง่าย (Availability)
- การบำรุงรักษา(Maintainability)

ความน่าเชื่อถือ (Reliability : R)

$$R = MTBF / (1+MTBF)$$

ความน่าจะเป็นเกณฑ์การวัดค่าที่ได้เป็นตัวเลขระหว่าง 0 และ 1
โดยวัดจากเวลาเฉลี่ยระหว่างเกิดความผิดพลาด
(mean time beween failures : MTBF)หน่วยเป็นชั่วโมง

เกณฑ์ประเมินความเชื่อถือ : Availability

- เป็นการวัดว่าระบบพร้อมที่จะทำงานมากน้อยเพียงใด อย่างไร
- เหมาะสำหรับระบบที่อยู่ในระยะหยุดซ่อมแซม และเมื่อเริ่มต้นระบบใหม่จะมีผลต่อความสูญเสียที่เกิดขึ้นจากการให้บริการต่อผู้ใช้เพียงใด
- เช่น อัตราความพร้อม=998/1000 หน่วยเวลา
  - หมายถึง ในทุก ๆ 1,000 หน่วยเวลา ระบบมีความพร้อมในการทำงานถึง 998 ครั้ง

ความพร้อมของซอฟต์แวร์(Avaliable)

- ความน่าจะเป็นที่โปรแกรมสามารถกระทำงานได้ตามช่วงเวลาที่ต้องการได้สำเร็จ ซึ่งการวัดความสามารถในการหาได้ง่ายนั้นต้องเกี่ยวข้องกับการแก้ไขความผิดพลาดของโปรแกรมโดยวัดเป็นเวลาเฉลี่ยหน่วยเป็นชั่วโมง แทนด้วย MTTR กำหนดให้ A เป็นความสามารถหาได้ง่ายของระบบเกี่ยวข้องกับ MTTR และ MTBF สามารถสร้างเป็นสูตรได้ดังนี้

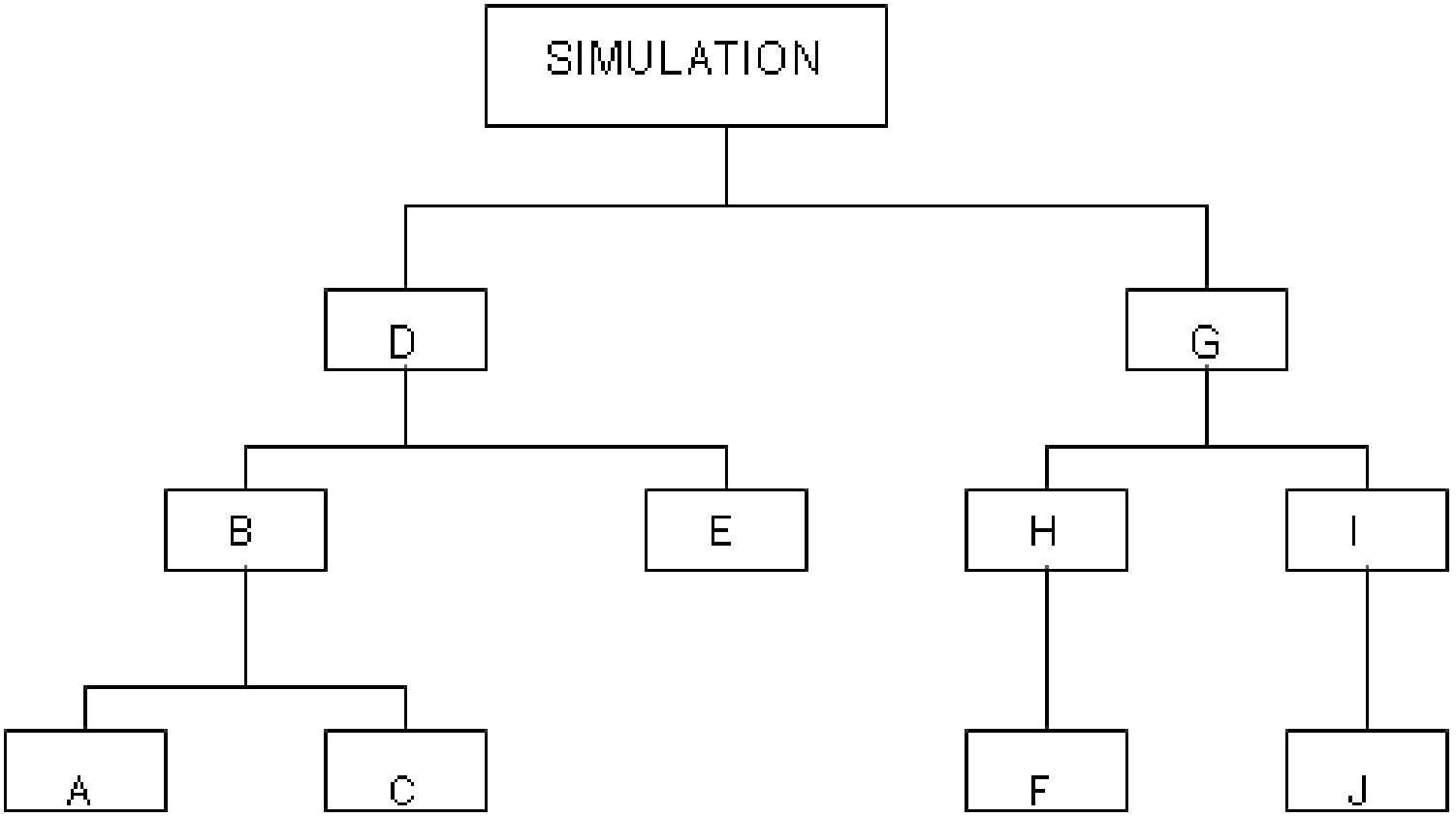
$$A = MTBF / (MTBF+MTTR)$$

การบำรุงรักษาของซอฟต์แวร์(Software Maintainability)

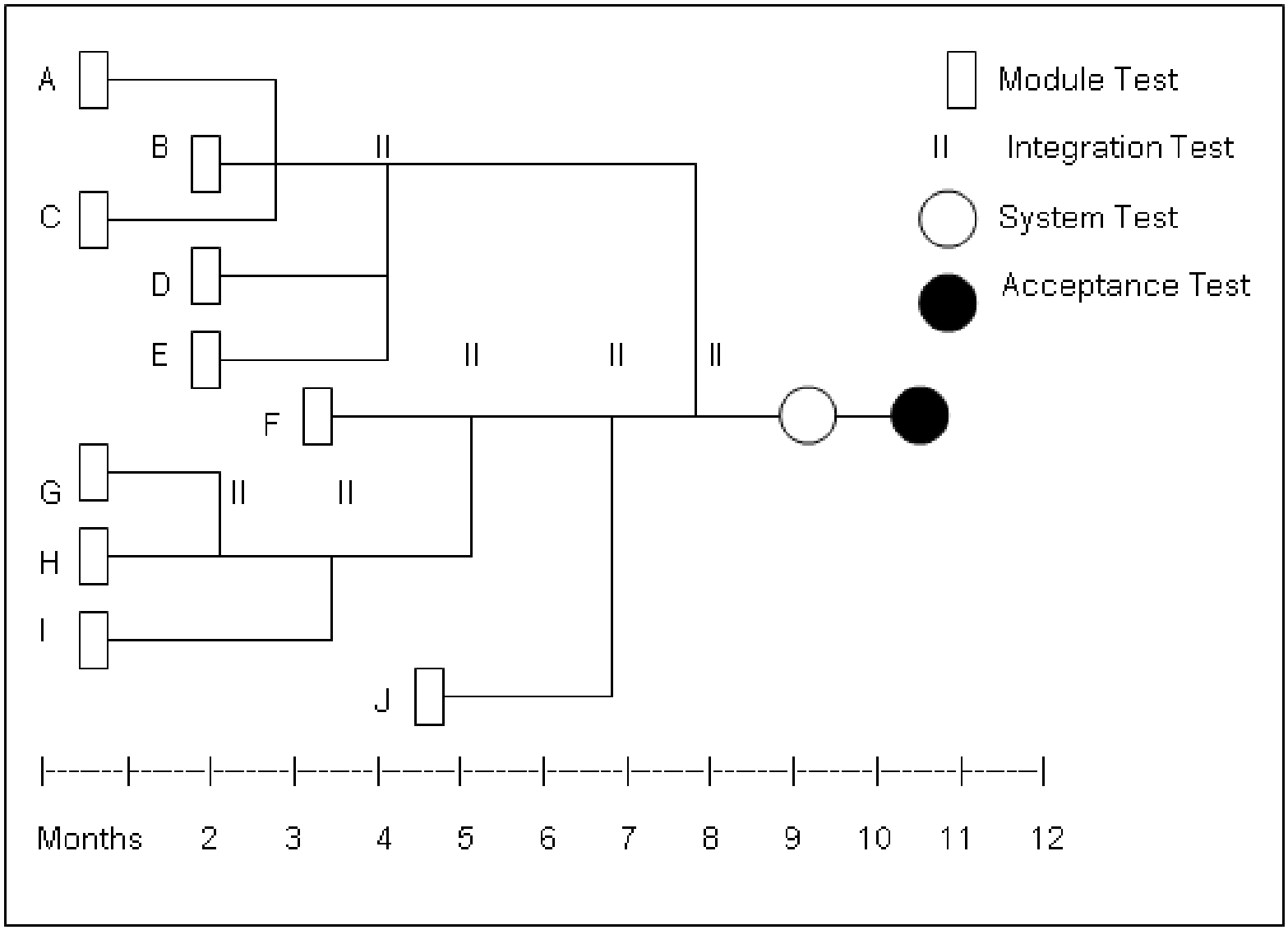
- ความเป็นไปได้ที่ความ ผิดพลาดของซอฟต์แวร์สามารถซ่อมแซมได้ ซึ่งเกี่ยวข้องกับเวลาเฉลี่ยในการซ่อมแซม หรือแก้ไขความผิดพลาดที่เกิดขึ้น (MTTR) สามารถสร้างความสัมพันธ์เป็นสูตรดังนี้

$$M = 1 / (1+MTTR)$$

ตัวอย่าง



ตัวอย่าง


A
B
C
D
E
F
G
H
I
J
Module Test
Integration Test
System Test
Acceptance Test
Months
2
3
4
5
6
7
8
9
10
11
12