Chapter 2

Information System Development

ดร.สันทิฎฐ์ นรบิน

เรียบเรียง

อ.วไลลักษณ์ วงษ์รื่น

Content

1) System Development Life Cycle

2) System Development Process Model

3) Methodologies

4) Two Approaches to System Development

5) Current Trends in Development

6) CASE Tools

7) หลักการพัฒนาระบบ

8) สรุปงานแต่ละเฟสใน SDLC

1 – System Development Life Cycle

SDLC

1.1 ความหมายของ SDLC

1.2 ขั้นตอนในวงจรการพัฒนาระบบสารสนเทศ (Phase)

1.1 ความหมายของ SDLC

- คือ กระบวนการทางความคิด (Logical Process) ในการพัฒนาระบบสารสนเทศ เพื่อแก้ปัญหาทางธุรกิจและตอบสนองความต้องการของผู้ใช้

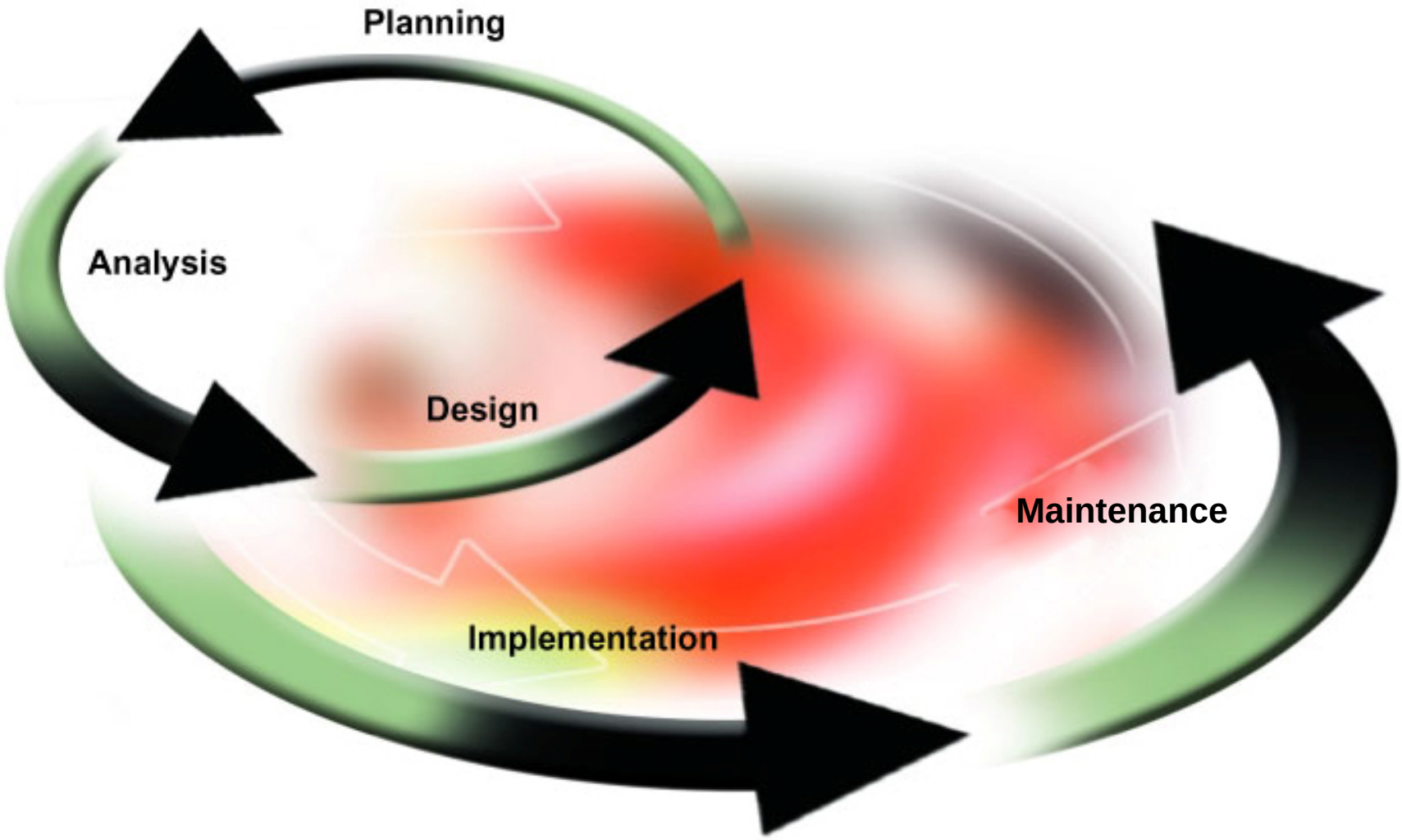
1.2 ขั้นตอนในวงจรการพัฒนาระบบ (Phase)

- ขั้นตอนในวงจรการพัฒนาระบบ ช่วยให้นักวิเคราะห์ระบบสามารถดำเนินการได้อย่างมีแนวทางและเป็นขั้นตอน ทำให้สามารถควบคุมระยะเวลาและงบประมาณในการปฏิบัติงานของโครงการพัฒนาระบบได้

หมายเหตุ

- หนังสือแต่ละเล่มจะแบ่งขั้นตอนในวงจรการพัฒนาระบบไม่เท่ากัน (ตามความคิดเห็นและแนวทางการพัฒนาระบบของผู้แต่งหนังสือเล่มนั้น)
- แต่โดยส่วนใหญ่จะมีการแบ่งขั้นตอนการพัฒนาระบบสารสนเทศไม่ต่ำกว่า 4 ขั้นตอน

วงจร SDLC



วงจร SDLC (ต่อ)

- ภายในวงจร SDLC จะแบ่งกระบวนการพัฒนาระบบออกเป็นระยะ (Phase) ประกอบด้วยระยะการทำงานหลัก ๆ ดังนี้

  1.2.1 Planning Phase

  1.2.2 Analysis Phase

  1.2.3 Design Phase

  1.2.4 Implementation Phase

  1.2.5 Maintenance Phase (Support Phase)

1.2.1 Planning Phase

- สำรวจความต้องการของผู้ใช้ และนำมาวิเคราะห์เพื่อค้นหาโครงการพัฒนาระบบที่สามารถตอบสนองความต้องการของผู้ใช้ได้
- จากนั้นคัดเลือกโครงการที่เหมาะสมและกำหนดขอบเขตของระบบใหม่
- ศึกษาความเป็นไปได้ของโครงการ
- จัดตารางดำเนินงาน
- วางแผนการใช้ทรัพยากร
- จัดทำงบประมาณ

1.2.2 Analysis Phase

- เป็นระยะที่ศึกษาขั้นตอนการดำเนินการของระบบเดิม เพื่อหาปัญหาที่เกิดขึ้น
- รวบรวมความต้องการในระบบใหม่จากผู้ใช้ระบบแล้วนำความต้องการเหล่านั้นมาศึกษาและวิเคราะห์ เพื่อแก้ปัญหาดังกล่าวด้วยการใช้แบบจำลองต่าง ๆ ช่วยในการวิเคราะห์

1.2.3 Design Phase

- เป็นระยะที่ทีมงานจะต้องออกแบบระบบสารสนเทศที่จะนำมาใช้แก้ปัญหาหรือตอบสนองความต้องการที่วิเคราะห์ไว้ใน Analysis Phase
- มีการกำหนดรายละเอียดขององค์ประกอบในส่วนต่าง ๆ ของระบบสารสนเทศ

1.2.4 Implementation Phase

- เป็นระยะของการสร้างระบบสารสนเทศ (เขียนโปรแกรม หรือจัดหาโปรแกรมจากแหล่งอื่น)
- จากนั้นทำการทดสอบโปรแกรม และติดตั้งระบบ พร้อมทั้งจัดทำคู่มือและจัดเตรียมหลักสูตร อบรมให้แก่ผู้ใช้งานที่เกี่ยวข้อง

1.2.5 Maintenance Phase

- เป็นระยะที่ทีมงานต้องทำหน้าที่ดูแลรักษาและเสริมสร้างระบบ
- การดูแลรักษา คือ การแก้ไขข้อผิดพลาดและการปรับเปลี่ยนแปลงตามสิ่งแวดล้อม
- การเสริมสร้างคือ การเพิ่มลักษณะเฉพาะใหม่ ๆ และสิ่งที่จะเป็นประโยชน์กับระบบ

2 – System Development Process Model

แบบจำลองกระบวนการพัฒนาระบบ

- เป็นการจำลองภาพของกระบวนการพัฒนาระบบสารสนเทศ
- เพื่อให้เห็นขั้นตอนของกระบวนการในรูปแบบต่าง ๆ
- เป็นการนำเสนอกระบวนการพัฒนาระบบสารสนเทศในแบบนามธรรม ดังนั้นรายละเอียดที่ปรากฏในแบบจำลองกระบวนการจึงเป็นเพียงรายละเอียดในการพัฒนาระบบเพียงบางส่วนเท่านั้น

รูปแบบของแบบจำลองกระบวนการพัฒนาระบบ

2.1 Waterfall Model

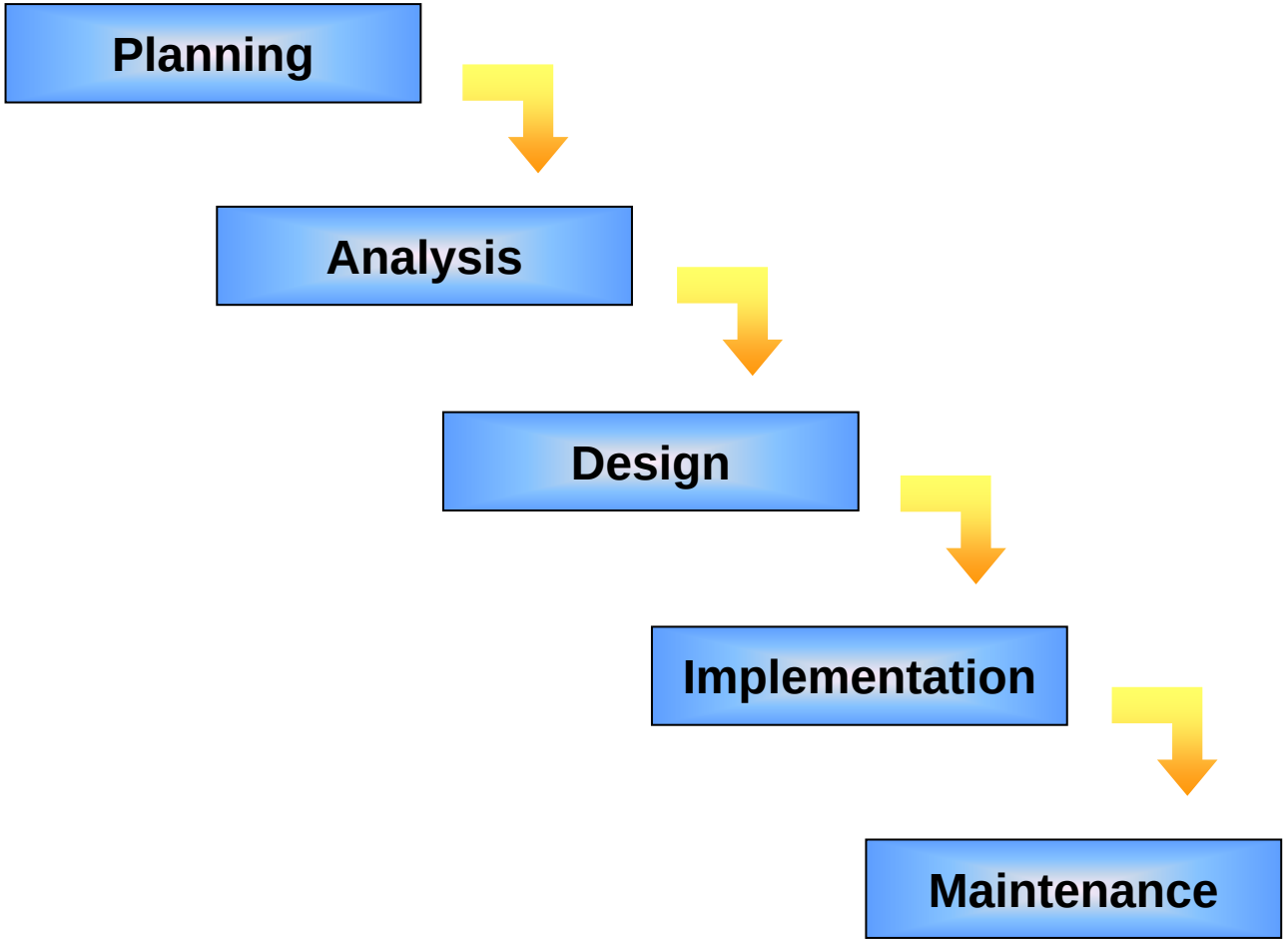
2.2 Spiral Model

2.3 Incremental Model

2.1 Waterfall Model

- เป็นแนวคิดแบบดั้งเดิม (Traditional) ของการพัฒนาระบบงาน
- ใช้หลักการเปรียบเทียบเสมือนกับน้ำตกที่ไหลจากที่สูงลงสู่ที่ต่ำ
- ผลลัพธ์ของแต่ละระยะ ที่เรียกว่า ผลผลิตขั้นสุดท้าย (End Product) จะลดหลั่นลงไปตามลำดับ

Waterfall Model



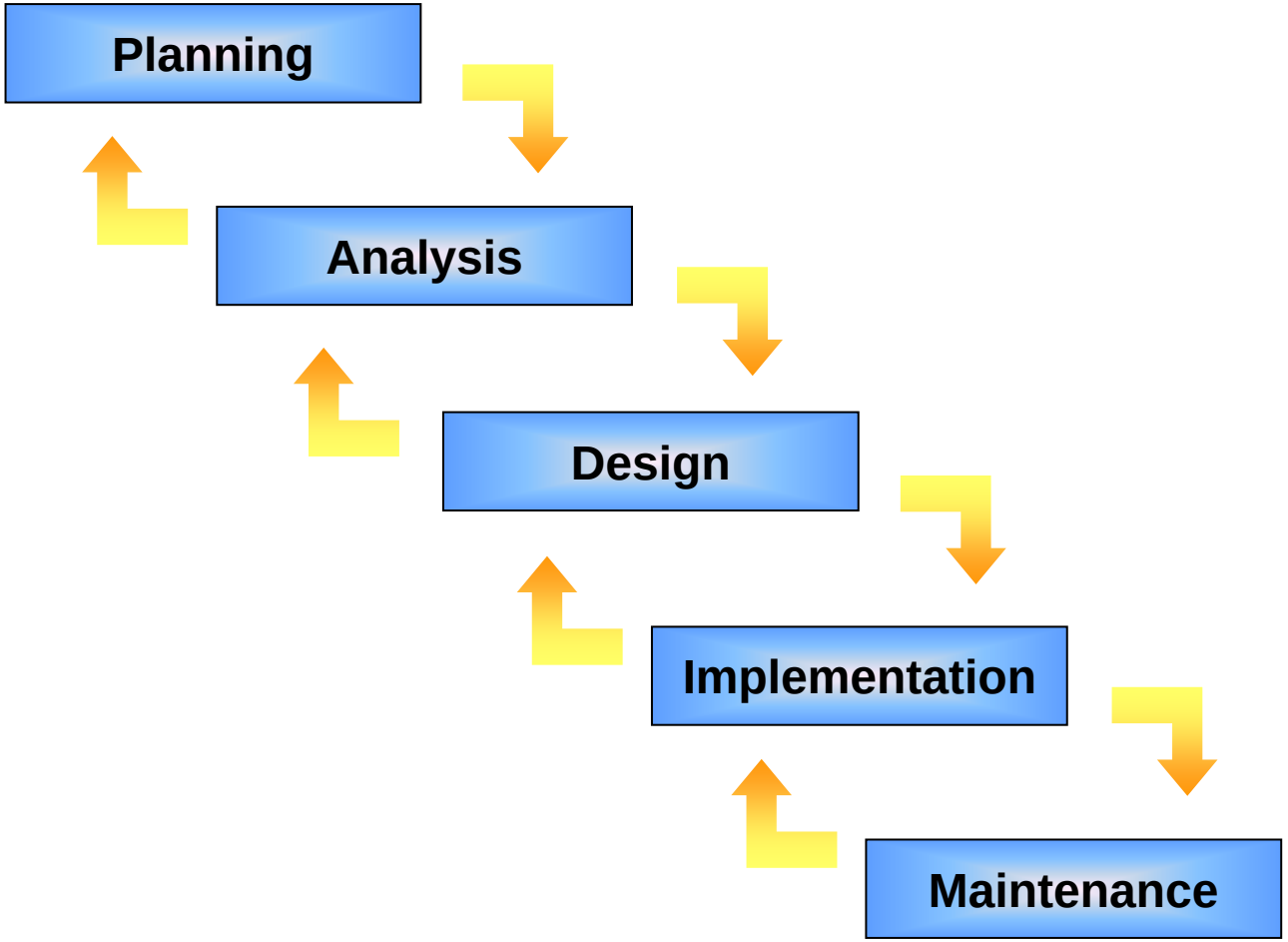
ข้อดีของ Waterfall Model

- มีการสร้างเอกสารในทุกระยะ
- ดำเนินงานทีละขั้นตอน ทำให้ตรวจสอบการทำงานได้ง่าย
- ขอบเขตงานชัดเจน
- เหมาะกับระบบขนาดเล็กที่ไม่ซับซ้อน

ข้อจำกัดของ Waterfall Model

- ใช้เวลาในการวางแผน วิเคราะห์ และออกแบบ มากเกินไป
- ผู้ใช้ได้เห็นระบบเมื่อผ่านขั้นตอนการพัฒนาแล้ว หากมีข้อบกพร่องหรือมีการเปลี่ยนแปลงความต้องการ จะทำให้ไม่สามารถแก้ไขระบบได้ทันตามความต้องการของผู้ใช้

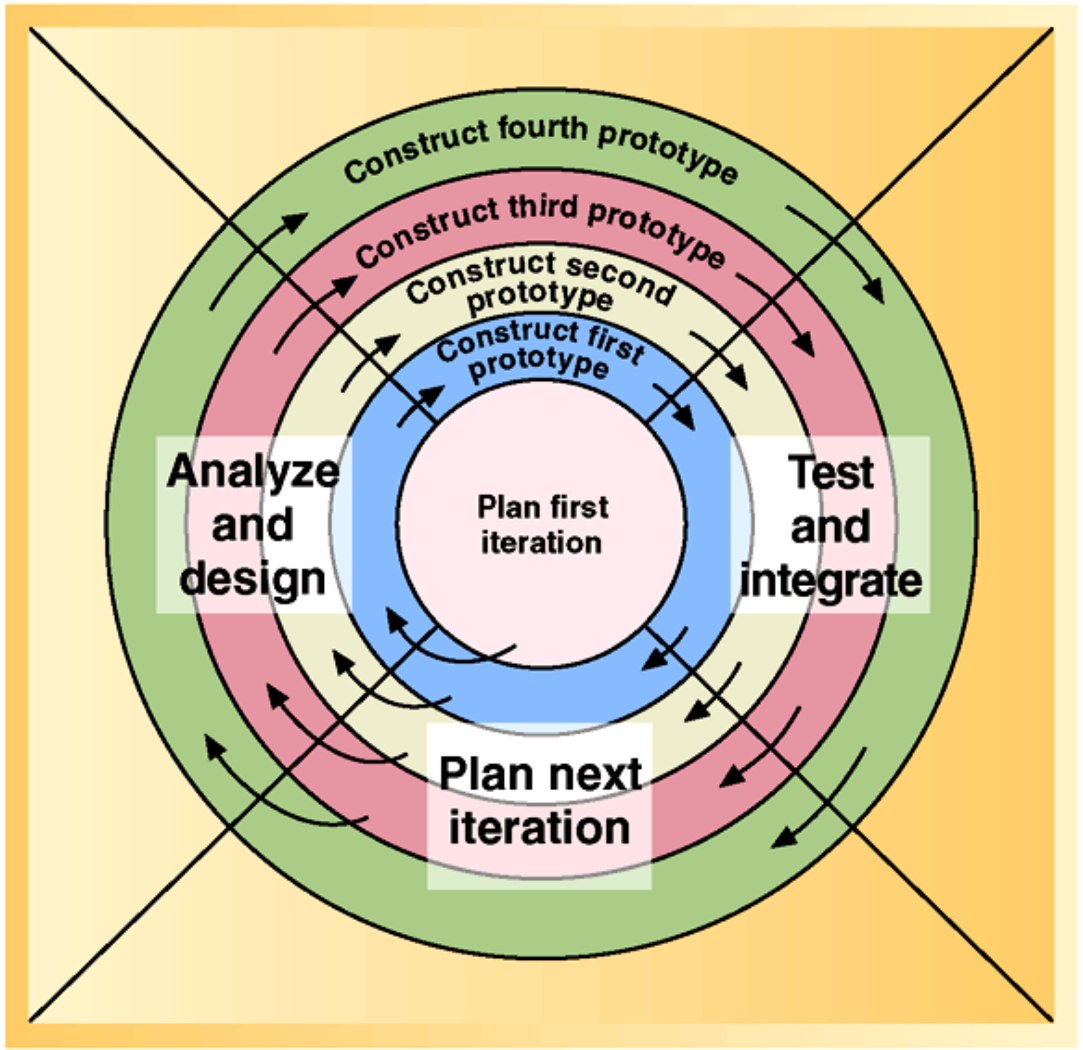
Adaptive Waterfall Model



2.2 Spiral Model

- เป็นรูปแบบที่มีการดัดแปลงการทำงานจาก Waterfall Model
- เพื่อให้การทำงานมีลักษณะที่ยืดหยุ่นได้
- มีลักษณะเป็นวงจรการทำ Analysis, Design, Implementation และ Testing และวนกลับมาในวงจรนี้เรื่อย ๆ จนได้ระบบที่สมบูรณ์

The Spiral Life Cycle Model



ข้อดีของ Spiral Model

- มีความยืดหยุ่นสูง เนื่องจากการทำ Analysis, Design, Implementation และ Testing ในแต่ละรอบนั้น สามารถกำหนดระยะเวลาได้ตามความจำเป็นของงาน
- เหมาะกับระบบที่มีการเปลี่ยนแปลงความต้องการบ่อย

ข้อจำกัดของ Spiral Model

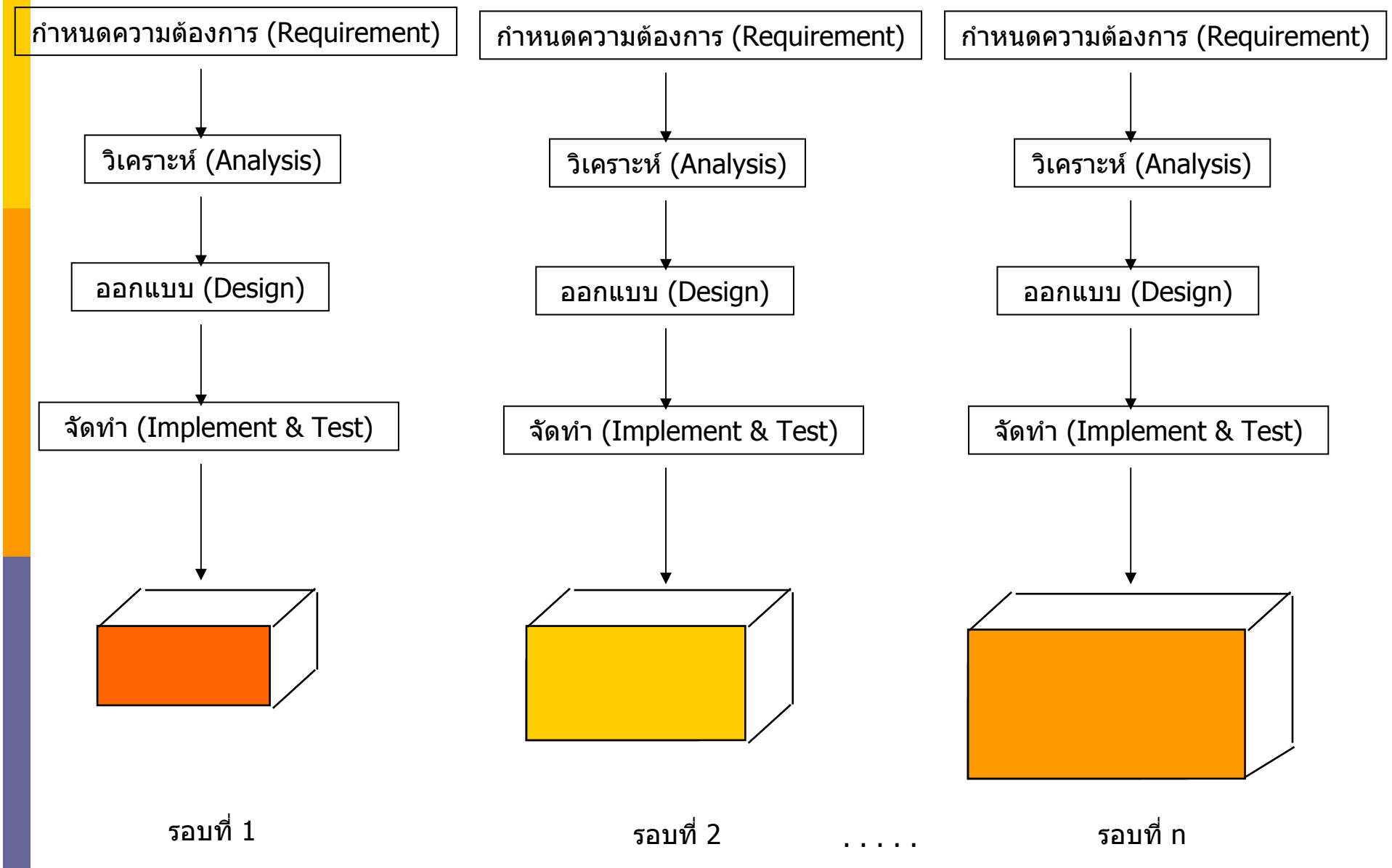
- มีความเสี่ยงสูง
- ต้องมีการวิเคราะห์ความเสี่ยงในการพัฒนาทุกรอบ

2.3 Incremental Model

- โครงการพัฒนาระบบถูกจัดออกเป็นหลายโครงการย่อย
- แต่ละโครงการย่อยมีกำหนดเวลา แผนงานของตัวเอง เรียกแต่ละโครงการย่อยว่า รอบงาน (Iteration)
- แต่ละรอบงานกระทำการพัฒนาระบบตามขั้นตอนวิเคราะห์ออกแบบ จัดสร้างของตัวเอง ได้ผลลัพธ์เป็นระบบที่นำไปประมวลผลได้

Incremental Model (ต่อ)

- ระบบจะถูกเพิ่มพูนให้โตขึ้นตลอดเวลาตามรอบงานแต่ละรอบงานที่ทำเสร็จจนกระทั่งได้ระบบที่สมบูรณ์ เป็นการพัฒนาระบบที่เรียกว่าวนรอบเพิ่มพูนผล (Iterative and Incremental Development)


กำหนดความต้องการ (Requirement)
วิเคราะห์ (Analysis)
ออกแบบ (Design)
จัดทำ (Implement & Test)
กำหนดความต้องการ (Requirement)
วิเคราะห์ (Analysis)
ออกแบบ (Design)
จัดทำ (Implement & Test)
กำหนดความต้องการ (Requirement)
วิเคราะห์ (Analysis)
ออกแบบ (Design)
จัดทำ (Implement & Test)
รอบที่ 1
รอบที่ 2
. . . . .
รอบที่ n

ข้อดีของ Incremental Model

- ผู้ใช้มองเห็นภาพ และได้ใช้งานระบบอย่างรวดเร็ว
- ผู้ใช้ปรับตัวกับระบบใหม่แบบค่อยเป็นค่อยไป
- ลดความเสี่ยง เนื่องจากในแต่ละรอบของการพัฒนา ได้นำระบบที่พัฒนาไว้ในรอบก่อนหน้ามาทดสอบร่วมด้วย

ข้อจำกัดของ Incremental Model

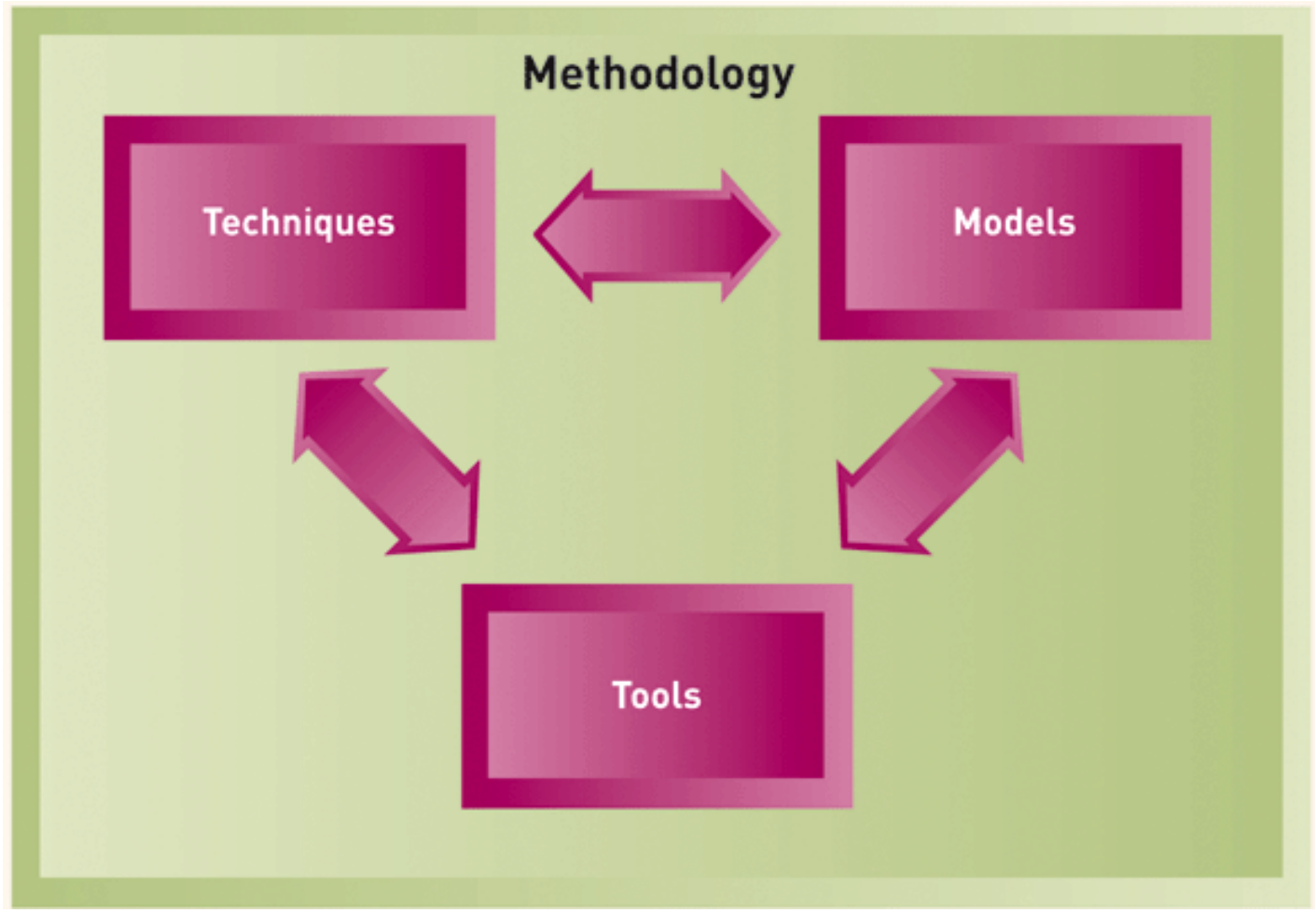
- ต้องมีการวางแผนในการประสานงานการทำงานภายในระบบอย่างดี เพื่อป้องกันข้อผิดพลาดที่อาจเกิดขึ้นได้

3 – Methodologies

ระเบียบวิธีปฏิบัติ

- เป็นวิธีการที่นำกระบวนการ (ที่คิดไว้แล้ว) ของวงจรการพัฒนาระบบมาปฏิบัติจริง เพื่อให้เป็นระบบสารสนเทศที่สามารถใช้งานได้จริง
- ใน Methodology จะระบุขั้นตอนการปฏิบัติ ชนิดของโมเดล เทคนิค และเครื่องมือ ที่ต้องใช้ในแต่ละขั้นตอน

Relationships Among Components of a Methodology



องค์ประกอบของ Methodology

3.1 Models

3.2 Tools

3.3 Techniques

3.1 Models

- เป็นแบบจำลองที่ใช้อธิบายระบบงาน
- ประกอบด้วยสัญลักษณ์ต่าง ๆ ตามที่แบบจำลองประเภทนั้นกำหนดไว้

Some Models Used in System Development

Some models of system components

- Flowchart
- Data flow diagram (DFD)
- Entity-relationship diagram (ERD)
- Structure chart
- Use case diagram
- Class diagram
- Sequence diagram

Some models used to manage the development process

- PERT chart
- Gantt chart
- Organizational hierarchy chart
- Financial analysis models – NPV, ROI

3.2 Tools

- เป็นเครื่องมือที่ใช้ในการพัฒนาระบบ
- เป็นซอฟต์แวร์ที่ช่วยสร้างหรือวาดโมเดลชนิดต่าง ๆ ช่วยตรวจสอบความถูกต้องของโมเดล ช่วยสร้างรายงานและแบบฟอร์ม รวมทั้งช่วยสร้างโค้ดโปรแกรมให้โดยอัตโนมัติ

Some Tools Used in System Development

Project management application
Drawing/graphics application
Word processor/text editor
Computer-aided system engineering (CASE) tools
Integrated development environment (IDE)
Database management application
Reverse-engineering tool
Code generator tool

3.3 Techniques

- คือวิธีการที่เป็นแนวทางที่ช่วยให้ SA สามารถดำเนินกิจกรรมในกระบวนการต่าง ๆ ของการพัฒนาระบบได้อย่างมีประสิทธิภาพ
- เทคนิคจะบอกวิธีในการทำงานใดงานหนึ่งอย่างเป็นลำดับขั้นตอน เช่น เทคนิคในการบริหารโครงการ เทคนิคในการออกแบบฐานข้อมูล เป็นต้น

Some Techniques Used in System Development

Strategic planning techniques
Project management techniques
User interviewing techniques
Data-modeling techniques
Relational database design techniques
Structured analysis technique
Structured design technique
Structured programming technique
Software-testing techniques
Object-oriented analysis and design techniques

4 – Two Approaches to System Development

วิธีการพัฒนาระบบสารสนเทศ

4.1 Traditional approach

- มักถูกเรียกว่า การพัฒนาระบบเชิงโครงสร้าง (structured system development)
- ใช้เทคนิค Structured analysis and design technique (SADT)
- มีการรวมวิศวกรรมสารสนเทศ (Information Engineering : IE) เข้าไปในวิธีการนี้ด้วย

4.2 Object-oriented approach

- มักถูกเรียกว่า OOA, OOD และ OOP
- ใช้วิธีการมองสิ่งต่าง ๆ ในระบบสารสนเทศเป็นวัตถุ (Object)

4.1 Traditional approach

4.1.1 Structured programming

4.1.2 Top-Down Programming

4.1.3 Structured Design

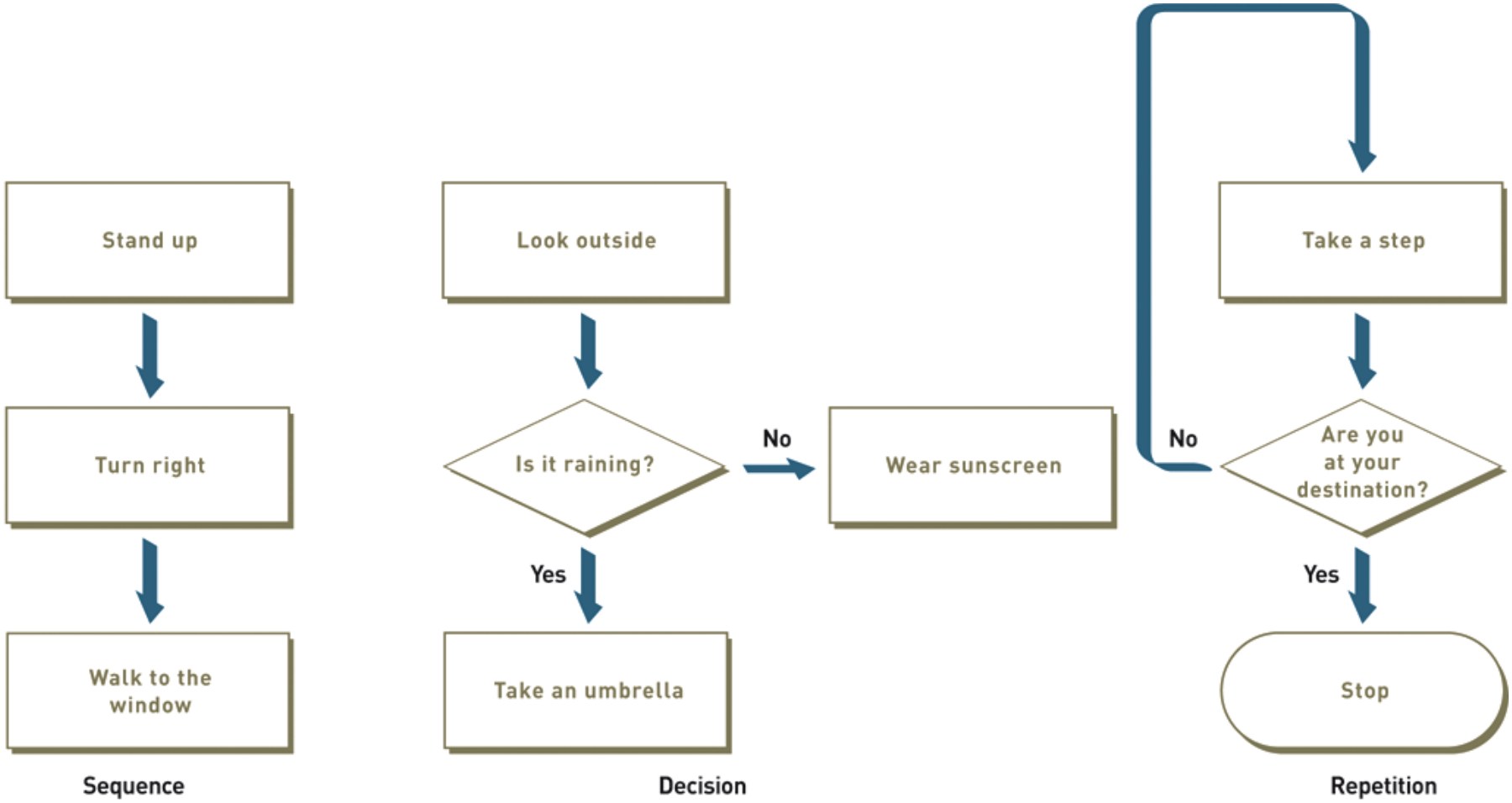
4.1.4 Structured Analysis

4.1.5 Information Engineering

4.1.1 Structured programming

- มีการปรับปรุงคุณภาพของโปรแกรมคอมพิวเตอร์
- อนุญาตให้โปรแกรมเมอร์แก้ไขปรับปรุงโค้ดให้ง่ายขึ้น
- แต่ละโมดูลในโปรแกรมจะมีจุดเริ่มต้นและจุดสิ้นสุดเพียงจุดเดียวเท่านั้น
- มีการควบคุมการทำงานภายในโปรแกรม 3 วิธี (sequence, decision, repetition)

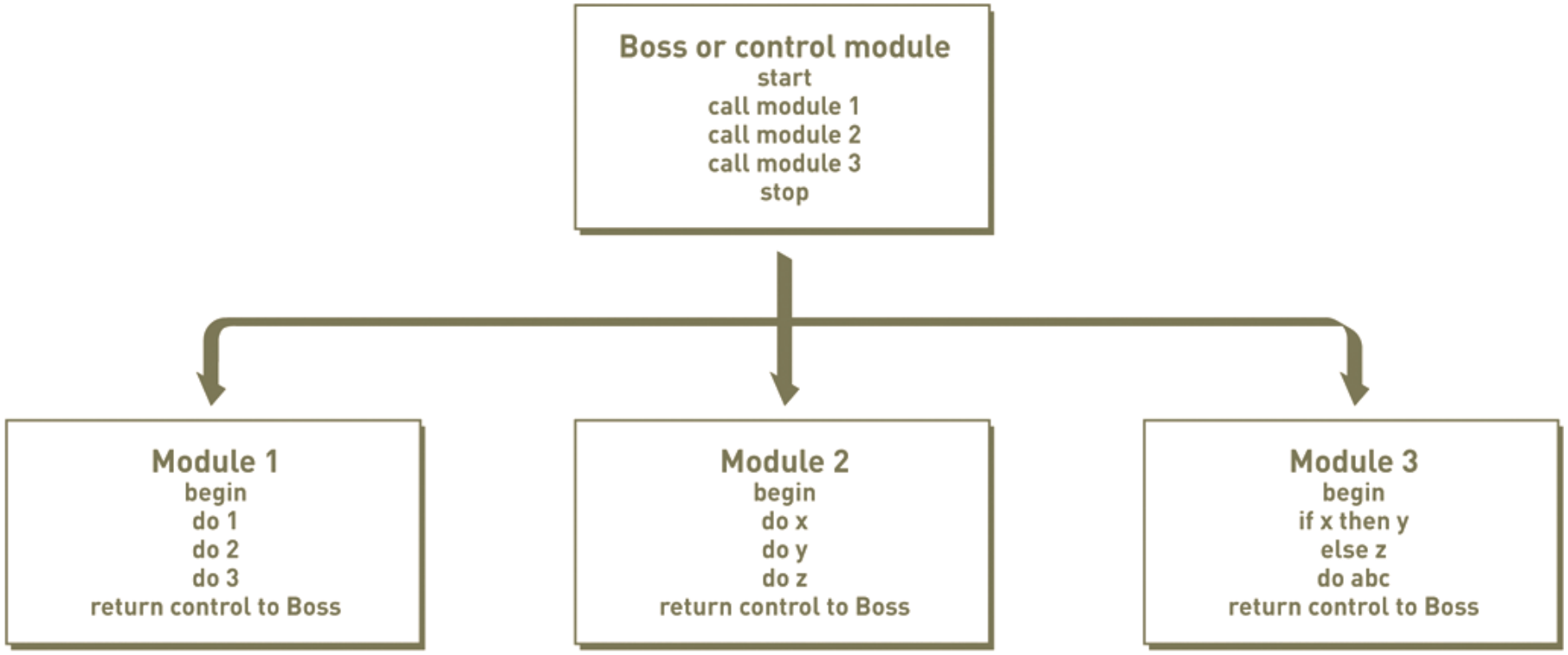
Three Structured Programming Constructs



4.1.2 Top-Down Programming

- มีการแบ่งความซับซ้อนของโปรแกรมให้เป็นแบบลำดับขั้น
- โมดูลสูงสุดจะมีการเรียกใช้โมดูลในระดับล่าง
- เป็นการเขียนโปรแกรมที่ประกอบด้วยโปรแกรมย่อยต่าง ๆ
- โปรแกรมหนึ่ง ๆ สามารถเรียกใช้โปรแกรมอื่น ๆ ได้ ภายในหนึ่งระบบงาน

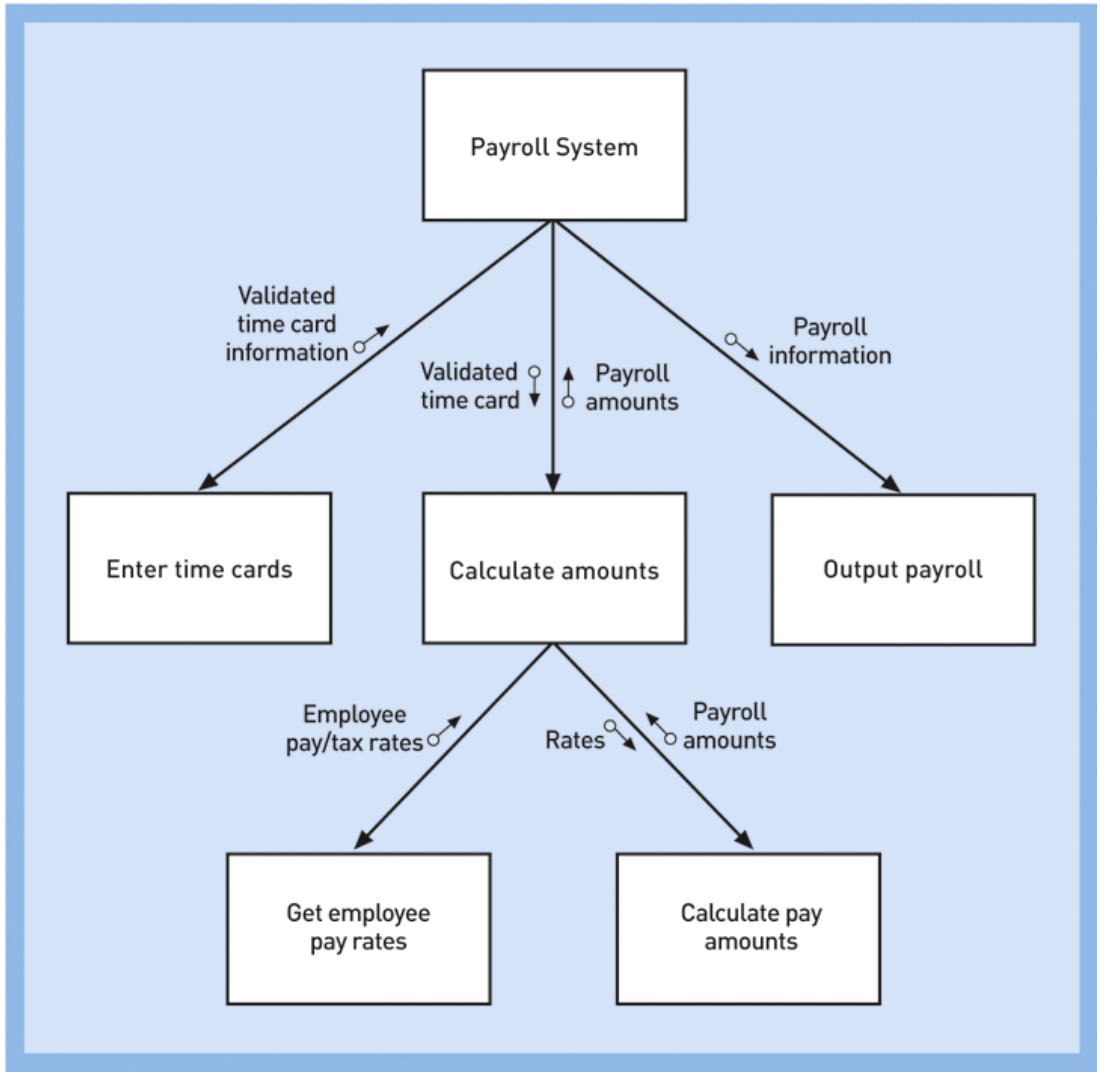
Top-Down or Modular Programming



4.1.3 Structured Design

- เป็นเทคนิคที่ถูกจัดเตรียมไว้เพื่อช่วยในการออกแบบระบบสารสนเทศ
- โมดูลจะถูกแสดงด้วย Structure chart
- องค์ประกอบที่สำคัญประกอบด้วย
  - Loosely coupled – เป็นโมดูลที่เป็นอิสระจากโมดูลอื่น ๆ ภายในระบบงาน
  - Highly cohesive – เป็นโมดูลที่มีภาระงานที่เชื่อมโยงการทำงานที่จะต้องกระทำให้เสร็จสิ้นก่อน

Structure Chart Created Using Structured Design Technique

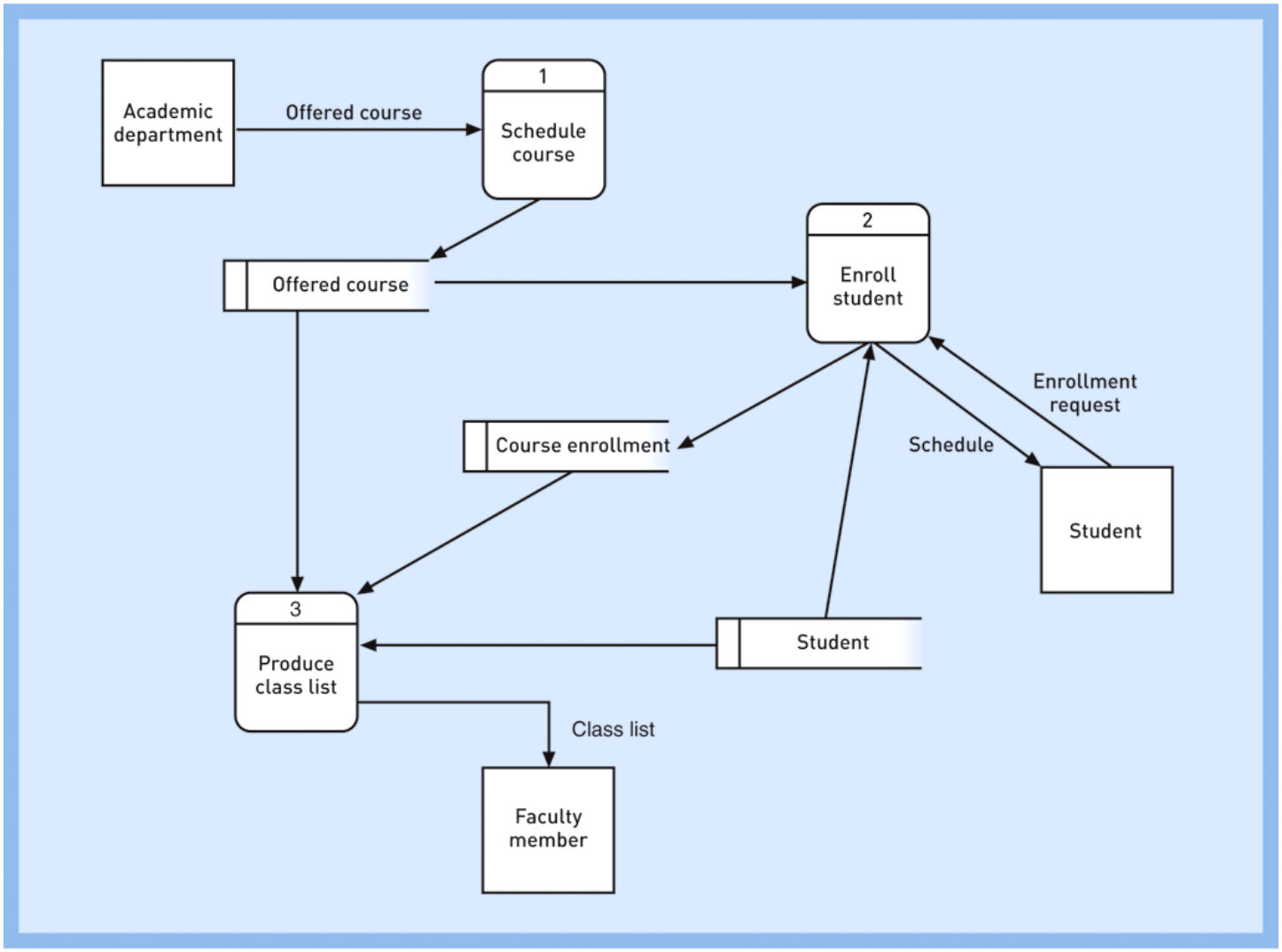


4.1.4 Structured Analysis

- กำหนดว่ามีอะไรบ้างที่ระบบจะต้องทำ (processing requirements)
- กำหนดว่าจะต้องใช้ข้อมูลอะไรบ้าง และเก็บข้อมูลใด ภายในระบบงาน (data requirements)
- กำหนดอินพุตและเอาท์พุต
- กำหนดฟังก์ชั่นที่จะต้องทำร่วมกันภายในระบบ
- ใช้ Data flow diagrams (DFD) และ Entity relationship diagrams (ERD) เพื่อแสดงผลลัพธ์ของการวิเคราะห์ระบบเชิงโครงสร้าง

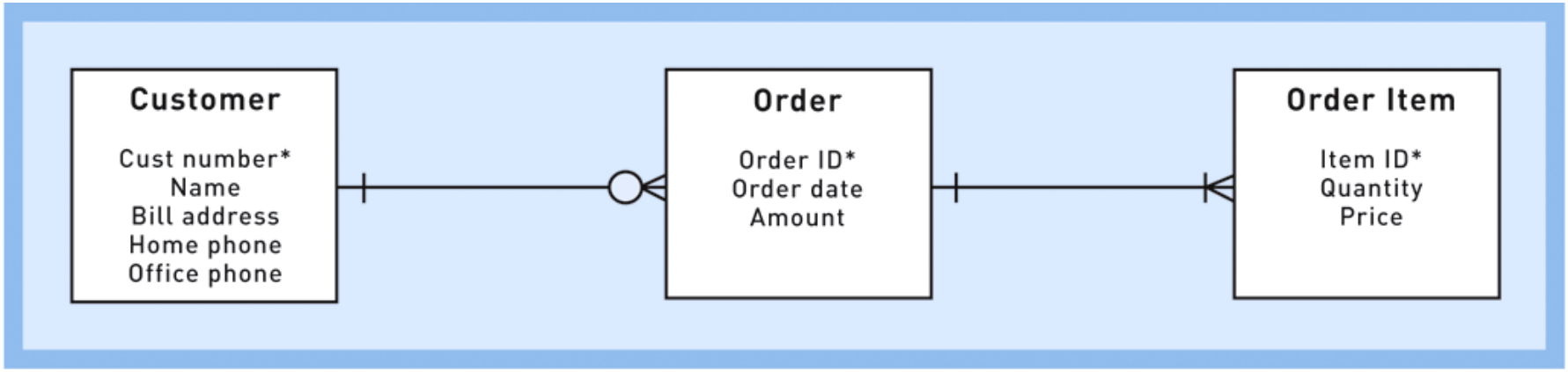
Data Flow Diagram (DFD)

Created Using Structured Analysis Technique

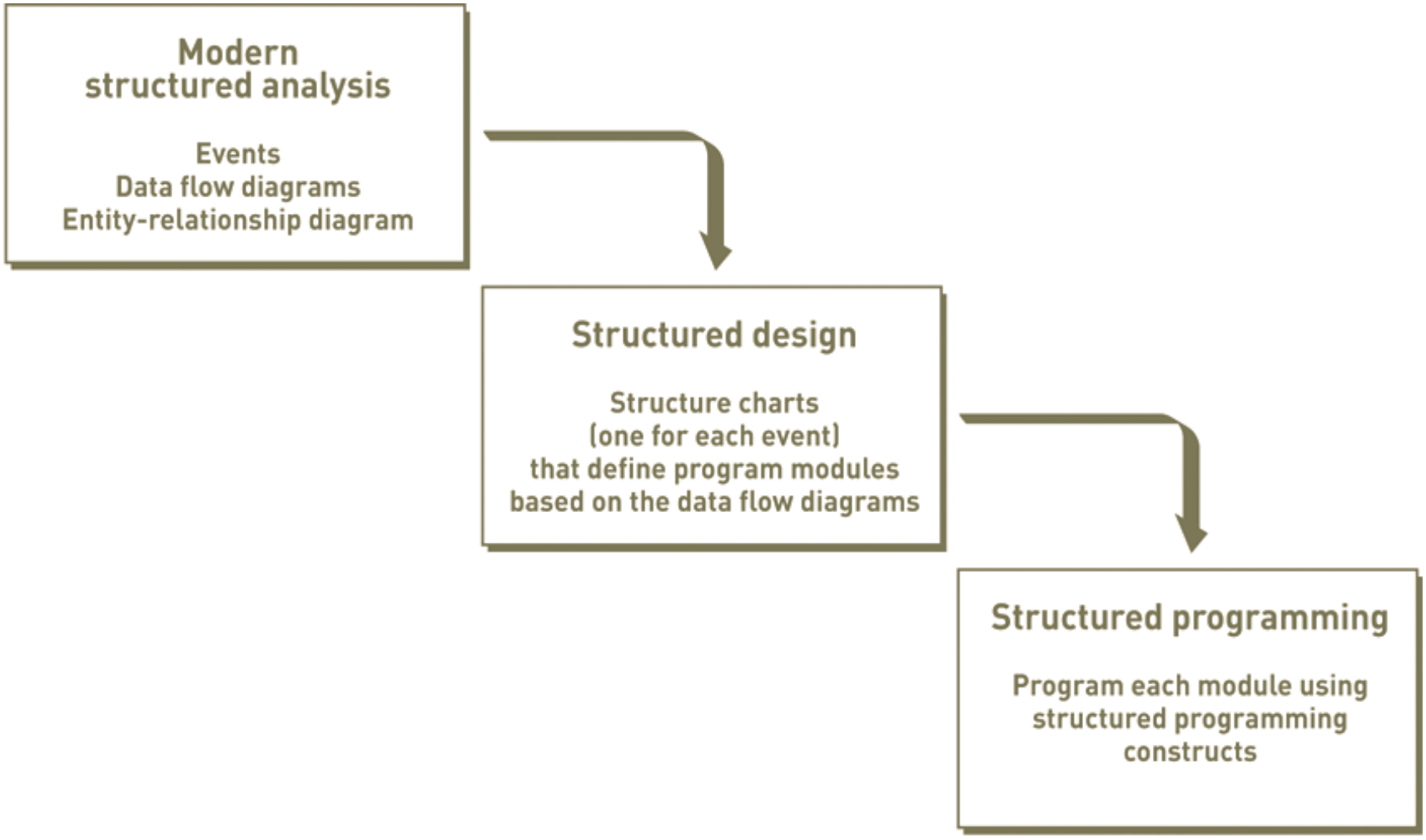


Entity-Relationship Diagram (ERD)

Created Using Structured Analysis Technique



Structured Analysis Leads to Structured Design and Structured Programming



4.1.5 Information Engineering

- เป็นกระบวนการที่ทำให้การพัฒนาระบบเชิงโครงสร้างมีความละเอียดรอบคอบมากยิ่งขึ้น
- วิธีการนี้จะประกอบด้วย strategic planning, data modeling, automated tools focus
- เป็นวิธีการพัฒนาระบบที่เข้มงวดในกระบวนการมากกว่าวิธีการ SADT

4.2 Object-oriented approach

- เป็นวิธีการที่มีมุมมองว่าระบบสารสนเทศเป็นแหล่งที่เก็บวัตถุ (Object) ที่มีการใช้งานร่วมกันภายในระบบ
- ภาษาที่ใช้ในการพัฒนาระบบคือภาษาเชิงวัตถุ (O-O languages) ได้แก่
  - Java
  - C++
  - C# .NET
  - VB .NET

Object-Oriented Approach (continued)

4.2.1 Object-oriented analysis (OOA)

4.2.2 Object-oriented design (OOD)

4.2.3 Object-oriented programming (OOP)

4.2.1 Object-oriented analysis (OOA)

- เป็นขั้นตอนที่กำหนดวัตถุต่าง ๆ ที่ควรมีในระบบ
- มีการใช้ use cases เพื่อแสดงให้เห็นว่ามีกิจกรรมใดที่จะต้องทำภายในระบบสารสนเทศ

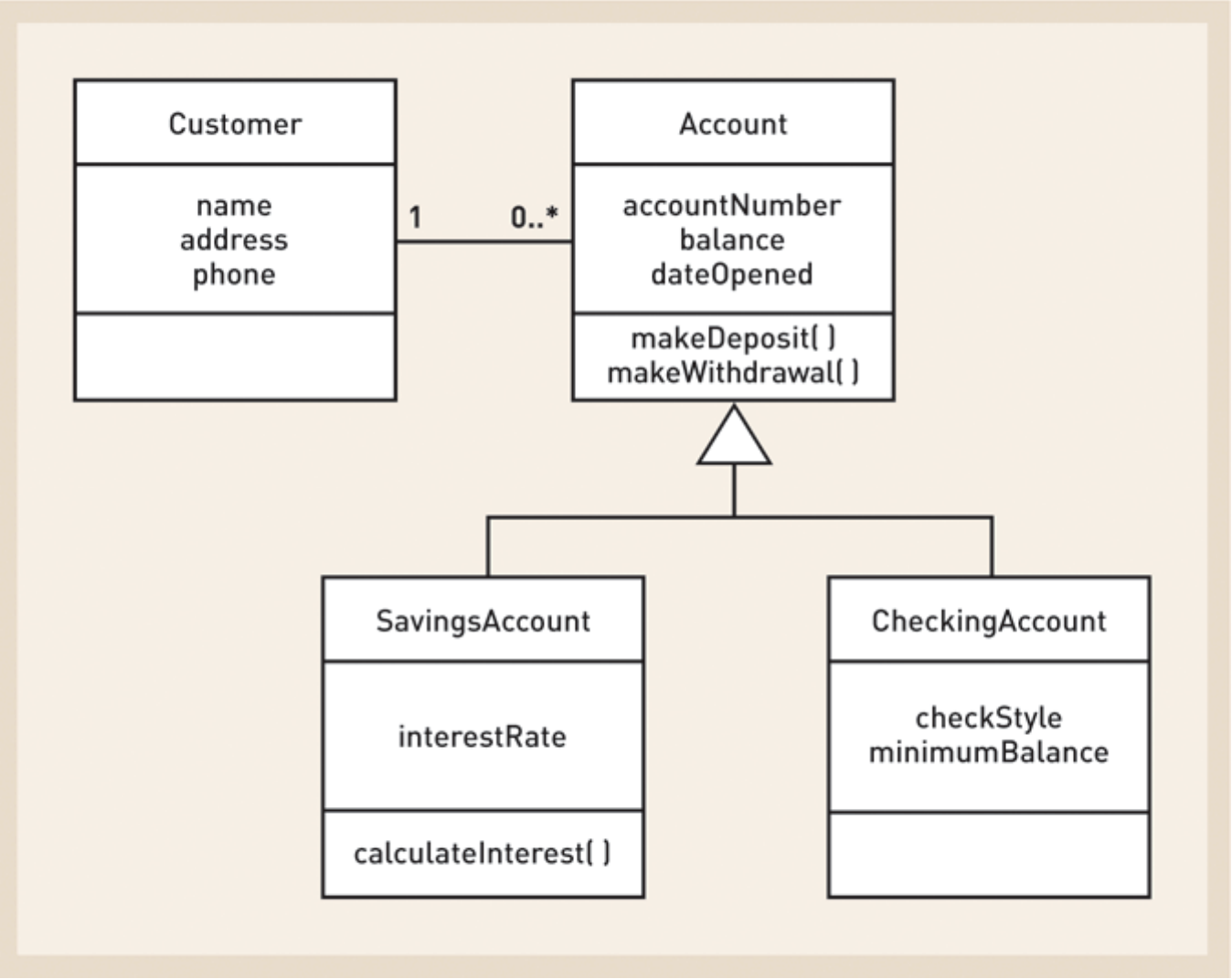
4.2.2 Object-oriented design (OOD)

- เป็นขั้นตอนที่กำหนดวัตถุที่จำเป็นต่อการติดต่อสื่อสารกับบุคคลและอุปกรณ์ต่าง ๆ ภายในระบบ
- เป็นขั้นตอนที่บ่งบอกว่าวัตถุจะมีปฏิสัมพันธ์กันอย่างไรในระบบงาน

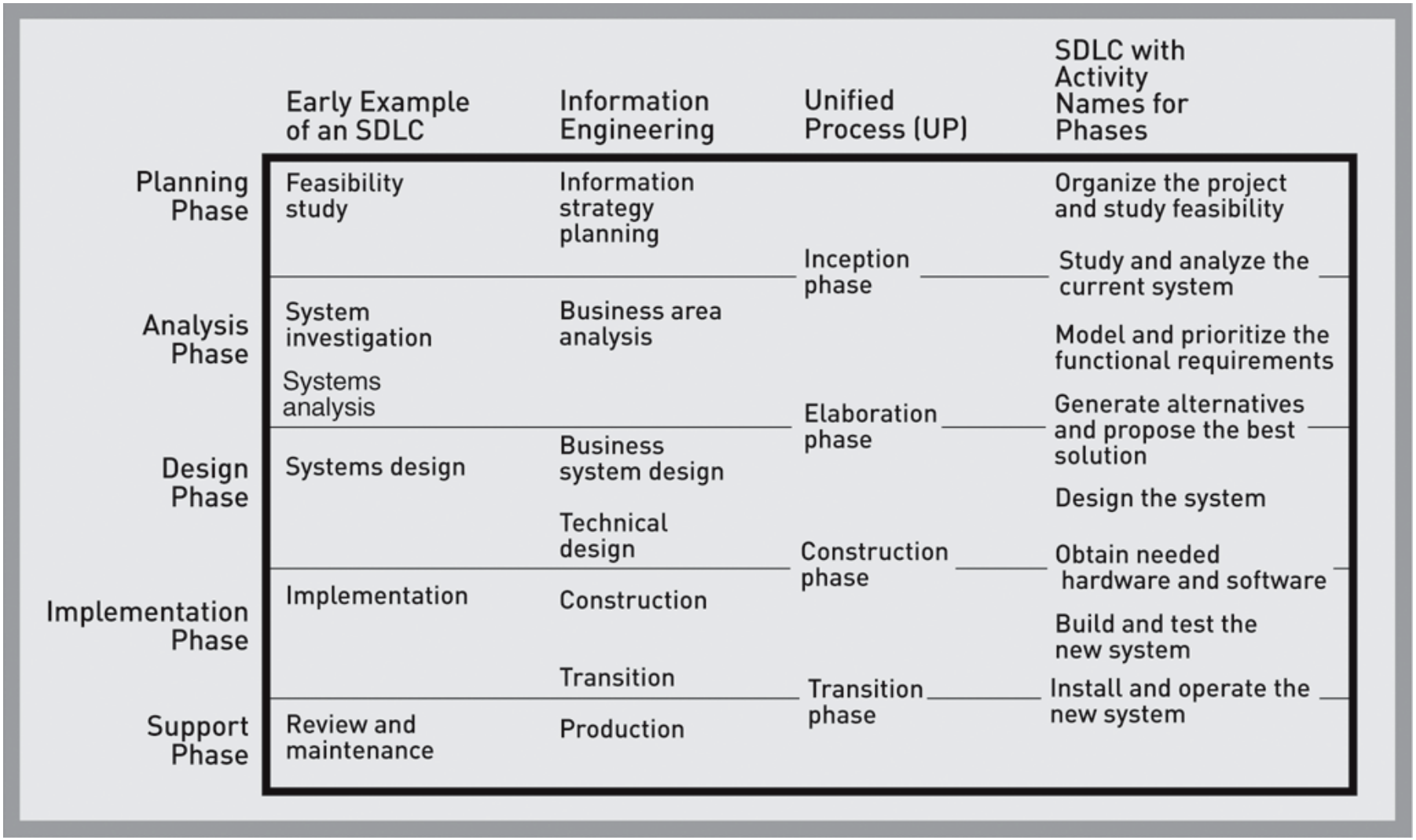
4.2.3 Object-oriented programming (OOP)

- เป็นขั้นตอนของการเขียนโปรแกรมเพื่อกำหนดให้วัตถุต่าง ๆ ที่ออกแบบไว้จะต้องทำงานอย่างไรภายในระบบงาน

Class Diagram Created During OO Analysis



Life Cycles with Different Names for Phases



5 – Current Trends in Development

แนวทางการพัฒนาระบบสารสนเทศในปัจจุบัน

5.1 The Unified Process (UP)

5.2 eXtreme Programming (XP)

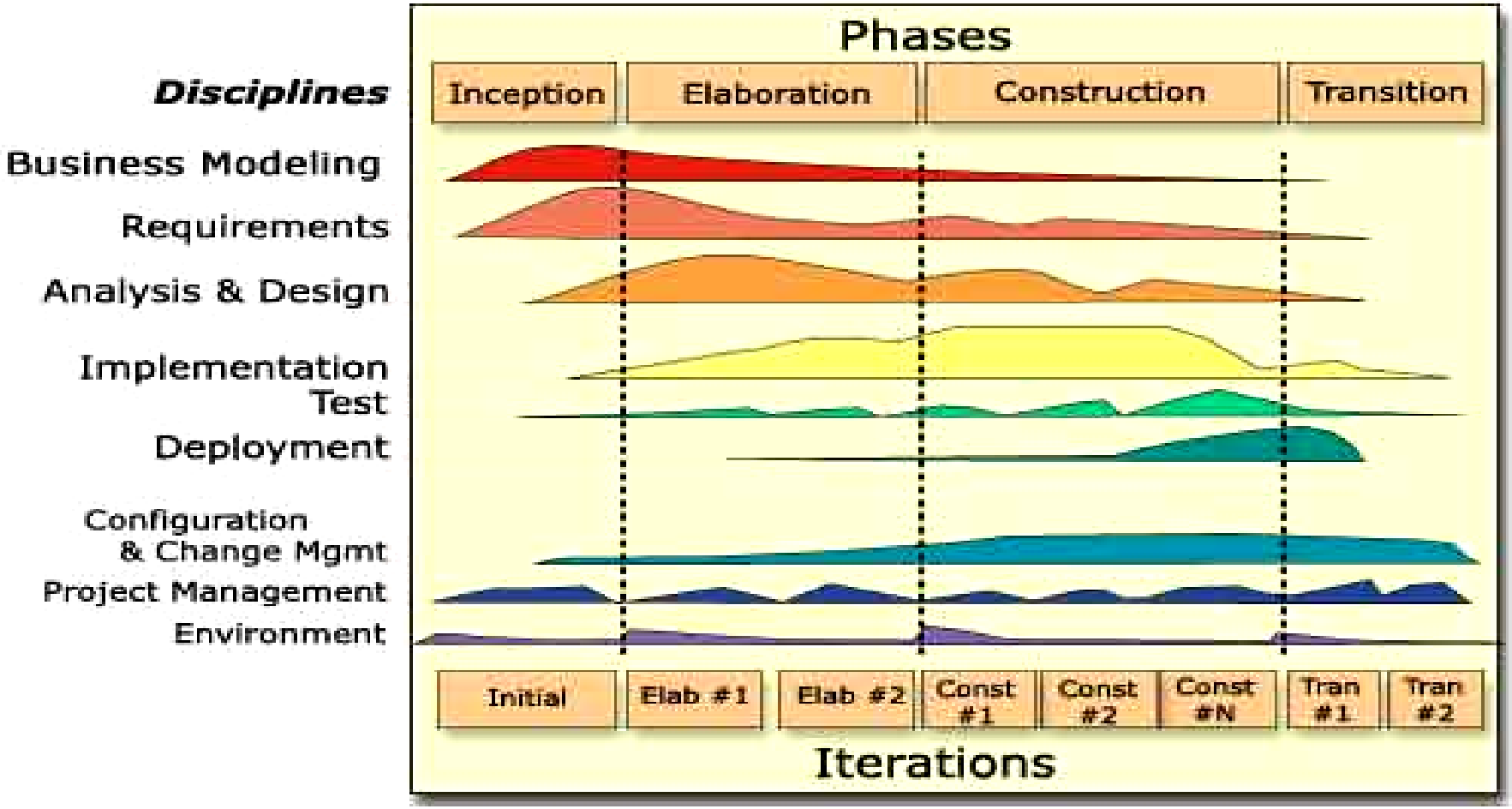
5.3 Agile Modeling

5.4 Scrum

5.1 The Unified Process (UP)

- เป็นวิธีการพัฒนาระบบเชิงวัตถุ (Object-oriented development approach)
- นำเสนอโดยบริษัท IBM และบริษัท Rational
- ใช้ Unified Modeling Language (UML) เป็นภาษาหลักในการสร้างแบบจำลอง
- การพัฒนาระบบแบ่งออกเป็น 4 ระยะ
  - Inception
  - Elaboration
  - Construction
  - Transition

งานที่ต้องทำในแต่ละ Phase ของวิธี UP



กระบวนการพัฒนาระบบแนว UP

5.1.1 Dynamic Perspective

5.1.2 Static Perspective

5.1.3 Practice Perspective

5.1.1 Dynamic Perspective

- เป็นมุมมองที่แสดงให้เห็นขั้นตอนการทำงานที่แบ่งเป็นเฟส ได้แก่
  - Inception : นิยามขอบเขตของโครงการ และกำหนดความสามารถในการพัฒนาระบบของทีมงาน
  - Elaboration : วางแผนโครงการ จัดทำรายละเอียดความต้องการ จัดสร้างสถาปัตยกรรมระบบ
  - Construction : สร้างและทดสอบโปรแกรม
  - Transition : ติดตั้งถ่ายโอนระบบให้กับผู้ใช้

5.1.2 Static Perspective

- เป็นมุมมองที่แสดงให้เห็นกิจกรรมที่ต้องดำเนินการ ได้แก่
  - Business Modeling : สร้างแบบจำลองทางธุรกิจ
  - Requirement : เก็บรวบรวมความต้องการ
  - Analysis and Design : วิเคราะห์และออกแบบระบบ
  - Implementation : สร้างระบบ
  - Test : ทดสอบระบบ
  - Deployment : นำระบบไปใช้งาน

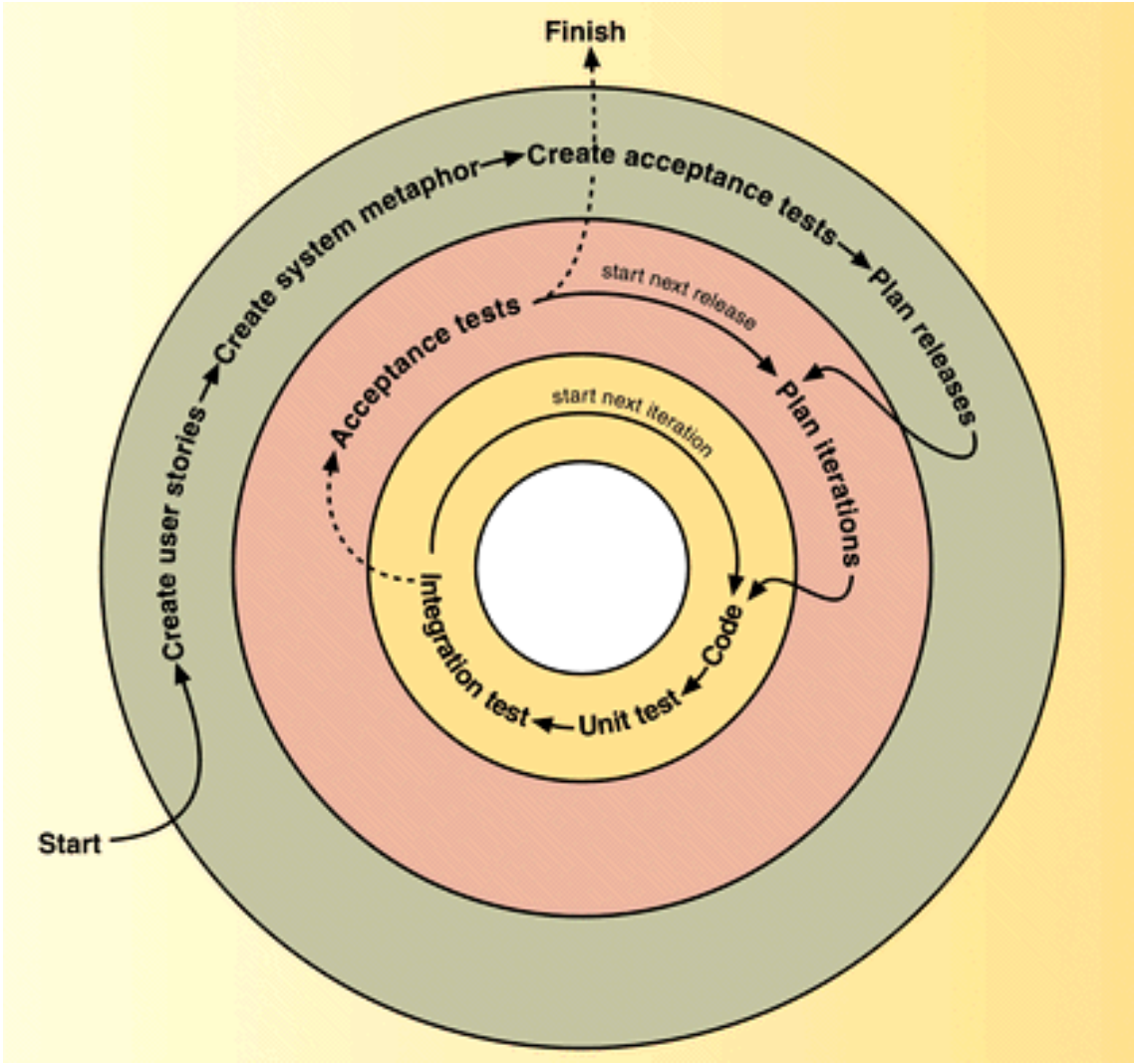
5.1.3 Practice Perspective

- เป็นส่วนสนับสนุนหลักที่ต้องใช้ในกระบวนการพัฒนาระบบ ได้แก่
  - Project Management : การบริหารโครงการ
  - Configuration and Change Management : การจัดการโครงร่างโครงการและการเปลี่ยนแปลงในโครงการ
  - Environment : การคำนึงถึงสภาพแวดล้อมของโครงการ

5.2 eXtreme Programming (XP)

- เป็นวิธีการพัฒนาระบบตามแนวทางการพัฒนาแบบ Iteration and Incremental Development
- ไม่เน้นการสร้างโมเดลและการจัดทำเอกสาร
- แบ่งขั้นตอนการทำงานเป็น 4 ขั้นตอน
  - Planning
  - Design
  - Coding
  - Testing

Extreme Programming System Development Approach



5.3 Agile Modeling

- Hybrid of XP and UP (Scott Ambler)
- มีการสร้างโมเดลในการพัฒนาระบบมากกว่า XP
- เอกสารในการพัฒนาระบบน้อยกว่าแบบ UP
- ใช้วิธีการ Interactive and Incremental Modeling
- มีความคล่องตัวในการทำงานสูง
- ให้ความสำคัญกับทีมงาน

5.4 Scrum

- เหมาะสำหรับการพัฒนาระบบที่มีความยืดหยุ่นสูง
- ตอบสนองต่อความต้องการของลูกค้าอย่างรวดเร็วทุกสถานการณ์เท่าที่สามารถเป็นไปได้
- ทีมงานจะเป็นผู้ควบคุมการดำเนินโครงการ

6 – CASE Tools

สาเหตุที่ต้องมีการใช้ CASE Tools

- ในแต่ละขั้นตอนของการพัฒนาระบบ จะมีการนำเทคนิคแบบจำลอง และแผนภาพ ชนิดต่าง ๆ อธิบายแทนข้อมูลจากเอกสารที่เป็นข้อความอธิบายลักษณะการทำงานของระบบ และวิธีแก้ไขปัญหาที่เกิดขึ้น
- เพื่อให้ขั้นตอนในการทำงานข้างต้นสามารถลดระยะเวลาลงได้
- ทำให้สามารถเพิ่มเวลาในขั้นตอนอื่น ที่เห็นว่าควรใส่ใจในรายละเอียดเพิ่มขึ้นได้
- ส่งผลให้การพัฒนาระบบมีความถูกต้องมากขึ้นและผิดพลาดน้อยลงได้

การใช้ CASE Tools

- ปัจจุบันมีซอฟต์แวร์ที่ช่วยสร้างแผนภาพ รายงาน โค้ดโปรแกรม ในระหว่างการวิเคราะห์และออกแบบระบบให้เป็นไปโดยอัตโนมัติ
- เรียกว่า Computer-Aided Systems Engineering (CASE)
- เป็นโปรแกรมประยุกต์หรือซอฟต์แวร์ชนิดหนึ่งของเทคโนโลยีที่ช่วยในการพัฒนาระบบ
- เพื่อสนับสนุนการทำงานในแต่ละขั้นตอนของการพัฒนา
- ซอฟต์แวร์เหล่านี้มีฟังก์ชันการทำงานต่าง ๆ ที่ทำให้การทำงานแต่ละขั้นตอนมีความรวดเร็วและมีคุณภาพมากขึ้น

การใช้ CASE Tools (ต่อ)

- CASE จะช่วยแบ่งเบาภาระของ SA ได้มาก
- CASE จะช่วยสร้าง Context Diagram, Flowchart, E-R Diagram สร้างรายงานและแบบฟอร์ม
- นอกจากนี้ยังสามารถช่วยสร้างโค้ดโปรแกรม (Source Code) ให้อัตโนมัติอีกด้วย

CASE Tool Framework

- CASE ที่ใช้ในการพัฒนาระบบถูกแบ่งขอบข่ายการทำงานออกเป็น 2 ช่วง โดยการแบ่งนั้นอ้างอิงจากขั้นตอนการพัฒนาระบบในวงจร SDLC ซึ่งมีดังต่อไปนี้
  - 6.1 Upper-CASE
  - 6.2 Lower-CASE

6.1 Upper-CASE

- เป็นเครื่องมือที่ช่วยสนับสนุนการทำงานในขั้นตอนต้น ๆ ของการพัฒนาระบบ
- ได้แก่ ขั้นตอนการวางแผน ขั้นตอนการวิเคราะห์ และขั้นตอนการออกแบบระบบ

6.2 Lower-CASE

- เป็นเครื่องมือที่ช่วยสนับสนุนการทำงานในขั้นตอนสุดท้ายในการพัฒนาระบบ
- ได้แก่ ขั้นตอนการออกแบบ ขั้นตอนการพัฒนาและทดสอบระบบ และขั้นตอนการให้บริการหลังการติดตั้งระบบ
- จะเห็นว่า CASE ทั้งสองระดับนี้ มีการทำงานที่ซ้ำซ้อนกันอยู่ บางครั้งองค์กรอาจเลือกใช้งาน CASE Tools ทั้งสองระดับร่วมกันได้

คุณสมบัติและความสามารถของ CASE

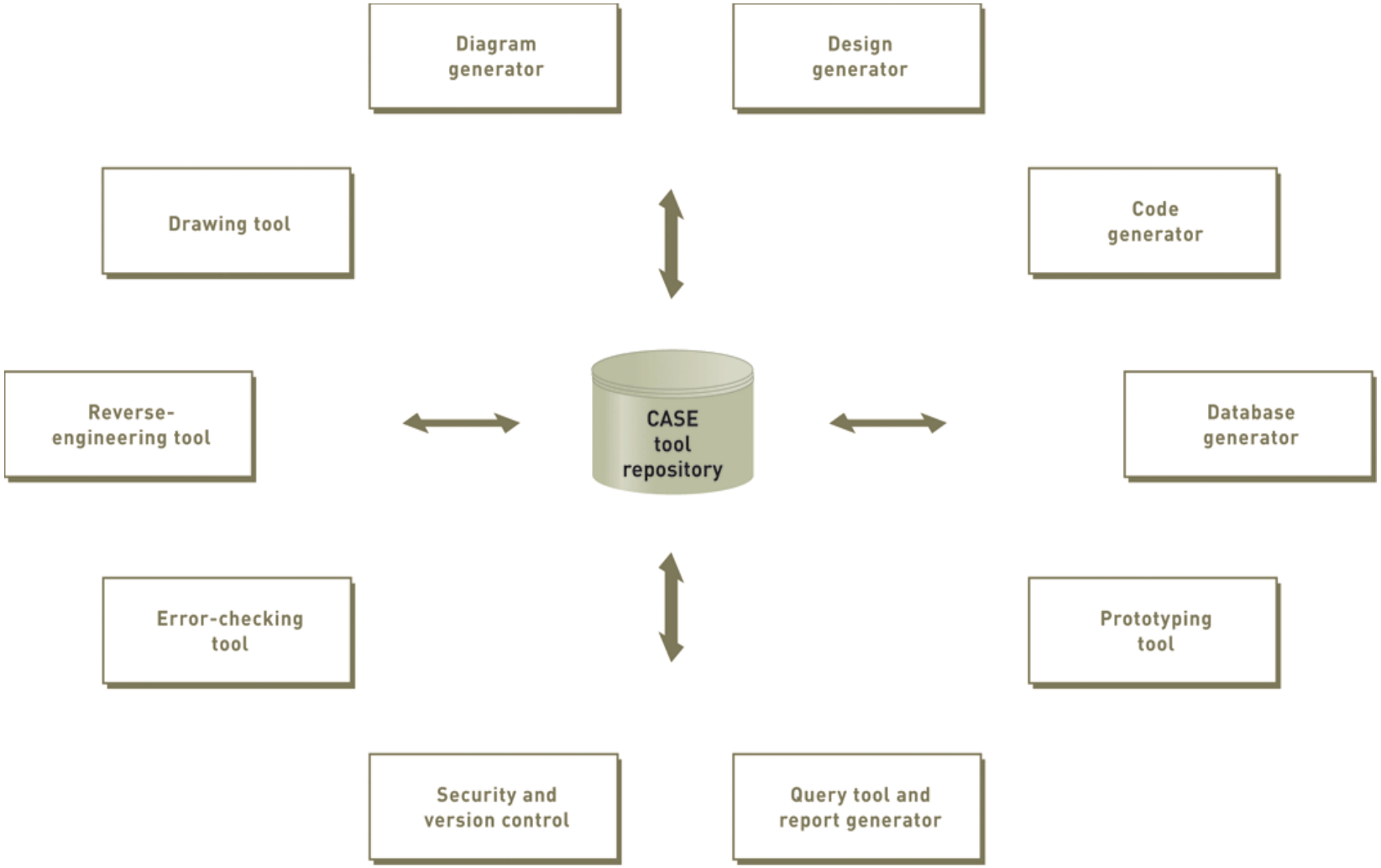
- ในการทำงานของ CASE จะมีการเรียกใช้ข้อมูลจาก Repository ซึ่งจะทำให้ CASE มีความสามารถและจัดเตรียมสิ่งอำนวยความสะดวกให้กับ SAในการพัฒนาระบบได้ ดังนี้

1. เครื่องมือช่วยสร้างแผนภาพ (Diagram Tools)
2. เครื่องมือช่วยเก็บรายละเอียดต่าง ๆ ของระบบ (Description Tools)
3. เครื่องมือช่วยสร้างตัวต้นแบบ (Prototyping Tools)
4. เครื่องมือช่วยสร้างรายงานแสดงรายละเอียดของแบบจำลอง (Inquiry and Reporting)

คุณสมบัติและความสามารถของ CASE (ต่อ)

5. เครื่องมือเพื่อคุณภาพของแบบจำลอง (Quality Management Tools)
6. เครื่องมือสนับสนุนการตัดสินใจ (Decision Support Tools)
7. เครื่องมือช่วยจัดการเอกสาร (Documentation Organization tools)
8. เครื่องมือช่วยออกแบบ (Design Generation Tools)
9. เครื่องมือช่วยสร้างโค้ดโปรแกรม (Code Generator Tools)
10. เครื่องมือช่วยทดสอบ (Testing Tools)
11. เครื่องมือช่วยให้สามารถใช้ข้อมูลร่วมกัน (Data Sharing Tools) เตรียมการนำเข้า (Import) และส่งออก (Export) ของสารสนเทศระหว่าง CASE Tools ที่ต่างกันได้

CASE Tool Repository Contains All System Information



ประโยชน์ที่ได้จากการใช้ CASE

1) มีการพัฒนาคุณภาพในการทำงาน เนื่องจาก CASE สามารถตรวจสอบความถูกต้อง สมบูรณ์ของแผนภาพและโปรแกรมได้

2) มีการสร้างเอกสารที่ดี

3) ประหยัดเวลาในการบำรุงรักษาให้ข้อมูลนั้นเป็นปัจจุบันมากที่สุด เพียงเข้าไปทำการแก้ไขในฐานข้อมูล Repository เท่านั้นก็สามารถสร้างเอกสารให้เป็นปัจจุบันได้ โดยไม่ต้องตามไปแก้ไขเอกสารที่เกี่ยวข้องทั้งหมดเอง

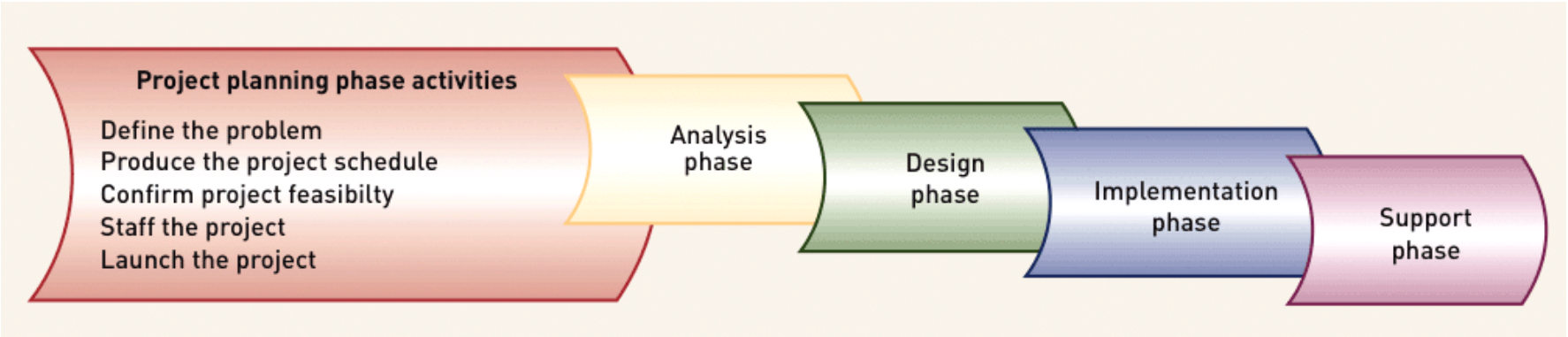
7 – หลักการพัฒนาระบบ

สิ่งที่ต้องคำนึงถึง

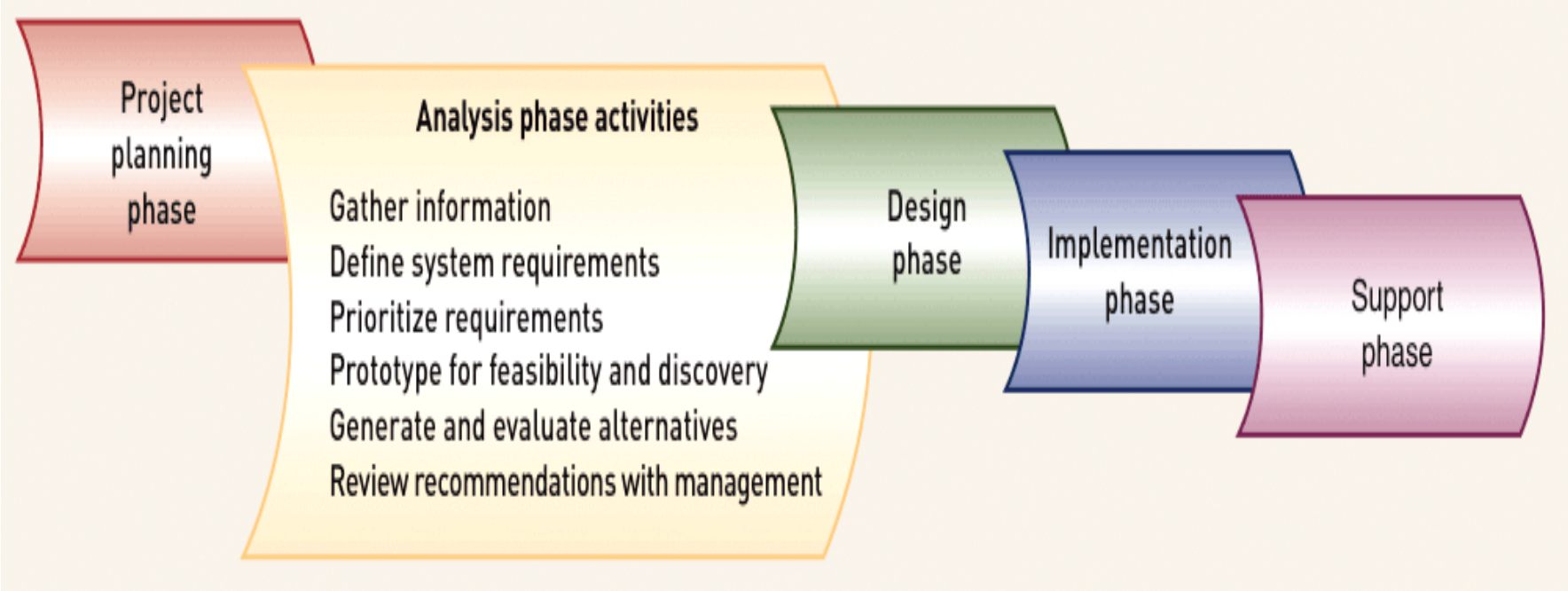
- ให้ความสำคัญต่อความต้องการของเจ้าของระบบและผู้ใช้
- ค้นหาและทำความเข้าใจปัญหา
- กำหนดขั้นตอนการดำเนินงานให้ชัดเจน
- กำหนดมาตรฐานในระหว่างการพัฒนาระบบ และควรจัดทำเอกสารทุกขั้นตอน
- พิจารณาความคุ้มค่าในการลงทุน
- หากเป็นระบบสารสนเทศขนาดใหญ่ให้แตกเป็นระบบย่อยเพื่อความง่ายในการเข้าถึงปัญหาและการวิเคราะห์ระบบ
- ออกแบบระบบสารสนเทศเพื่อรองรับการเติบโตของงานและองค์กรในอนาคต

8-สรุปงานแต่ละเฟสใน SDLC

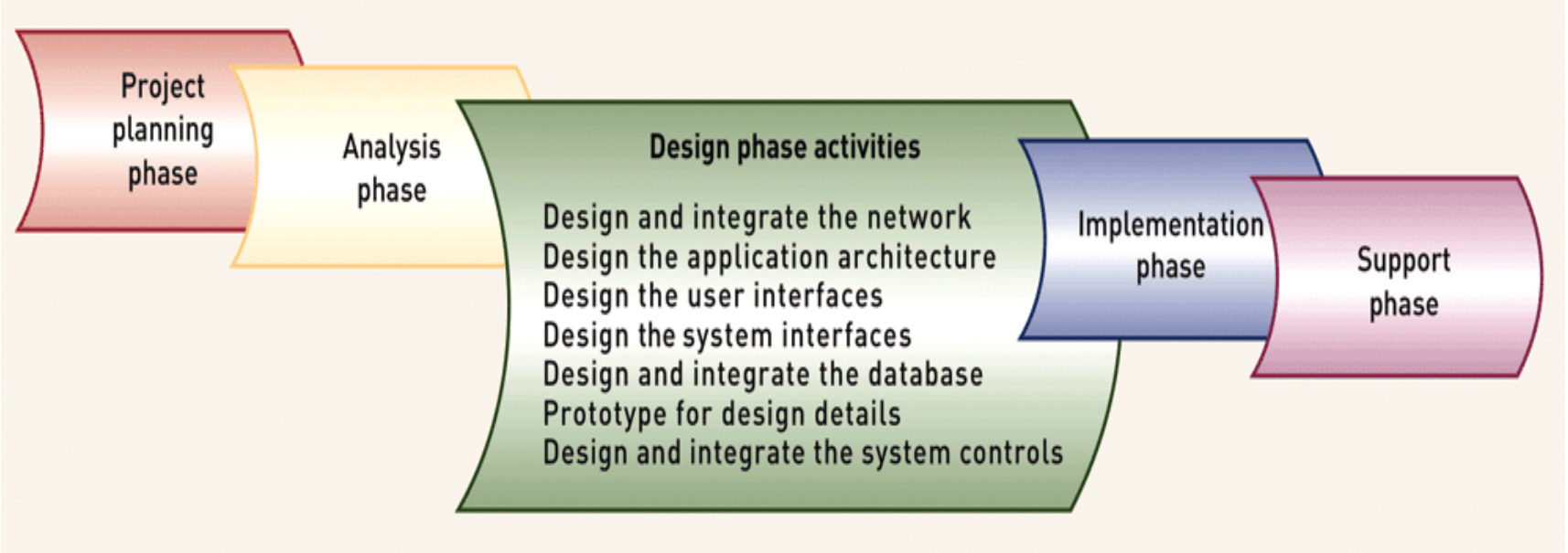
Activities of the Project Planning Phase



The Activities of the Analysis Phase



SDLC Phases with Design Phase Activities



Activities of the Implementation and Support Phases