SINGAPORE INSTITUTE OF TECHNOLOGY

ICT2108: SOFTWARE MODELLING & ANALYSIS

LEC1: Introduction/Recap/Overview

Dr Lu Liming & Dr Ryan Kirwan

Today's Agenda

- Module plan:
  - Lecture schedule
  - Assignment weightages/dates
- Overview:
  - (revision) Software Engineering
  - Terminology to remember
- ICT2101 recap:
  - Terminology/Standards
  - Requirements Engineering + recap

Module schedule Trimester 2

Wk	Topic
1	Introduction to module + Recap + Overview
2	Requirements Engineering (RE) + Requirements Elicitation
3	Requirements Elicitation + Evaluation + Acceptance
4	Requirements Evaluation + Risk Management
5	Requirements Prioritization + Documentation
6	Requirements Documentation + Requirements Engineering Recap
7	Recess Week
8	UML + RML
9	Scenario Modelling + Agile
10	Structural/Behavioural Modelling
11	Requirements validation and Prototyping
12	Formal Methods + State Diagrams
13	Revision
14-15	Exam Weeks
16-18	Trimester Break

module overview

Teaching activities (every Tuesday LEC: (09:00-12:00), Friday LABs: (14:00-18:00))

• Lectures: 3 hours per wk
• Lab/Tutorials: 2 hours per wk, starts in wk2 (*project info released wk1*)

Assessments

- Group Project, 40%

• Interview skills (CCS) A0*, interview practice (wk2), Assessed (wk3)		4%
• Req Eng A1: Interview 1 (wk4)		4%
• Req Eng A2: Req Document 1: Submission (wk5)		6%
• Req Eng A3: Prototype Demo 1: Assessment (wk6)		8%
• Req Eng A4: Interview 2 (wk9)	~*tentative*~	4%
• Req Eng A5: Req Document 2: Submission (wk11)	~*tentative*~	6%
• Req Eng A6: Prototype Demo 2: Assessment (wk13)	~*tentative*~	8%

- Peer Assessment: via Teammates, applies to all Group Project Submissions (wk13)
- 1st half LMS Quiz, 10% (wk6), 2nd half LMS Quiz, 10% (wk11 or 12)
- Exam, 40 %

- Interview skills (CCS)*:

Is an individual assessment that we'll cover in detail at the end.

Assessed in wk3, 4%

module overview

Teaching activities (every Tuesday LEC: (09:00-12:00), Friday LABs: (14:00-18:00))

- Lectures: 3 hours per wk
- La… …wk1)

Asse…

- G…

- Int…
- Re…
- Re…
- Re…
- Re…
- Req Eng A5: Req Document 2: Submission (wk11) ~*tentative*~ **6%**
- Req Eng A6: Prototype Demo 2: Assessment (wk13) ~*tentative*~ **8%**

- Peer Assessment: via Teammates, applies to all Group Project Submissions (wk13)
- 1st half LMS Quiz, 10% (wk6), 2nd half LMS Quiz, 10% (wk11 or 12)
- Exam, 40 %

- **Interview skills (CCS)*UPDATE 2:**

Lab groups Will be moved to Tuesday 14th and Tuesday the 21st for those respective weeks (remove the Friday labs for those weeks).

Timeslots: 1-3pm and 3-5pm.
Rooms: SR4A and SR4F
(see the LMS for your specific rooms and timings).

- **Interview skills (CCS)*:**

Is an individual assessment that we'll cover in detail at the end.

Assessed in wk3, **4%**

SINGAPORE INSTITUTE OF TECHNOLOGY

Software Engineering

What is software engineering?

"The application of a *systematic*, *disciplined*, *quantifiable* approach to the *development*, *operation* and *maintenance* of software."

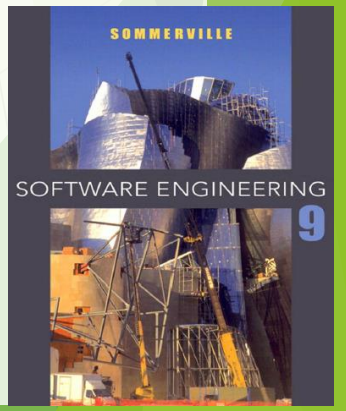
SINGAPORE INSTITUTE OF TECHNOLOGY

IEEE
Advancing Technology for Humanity

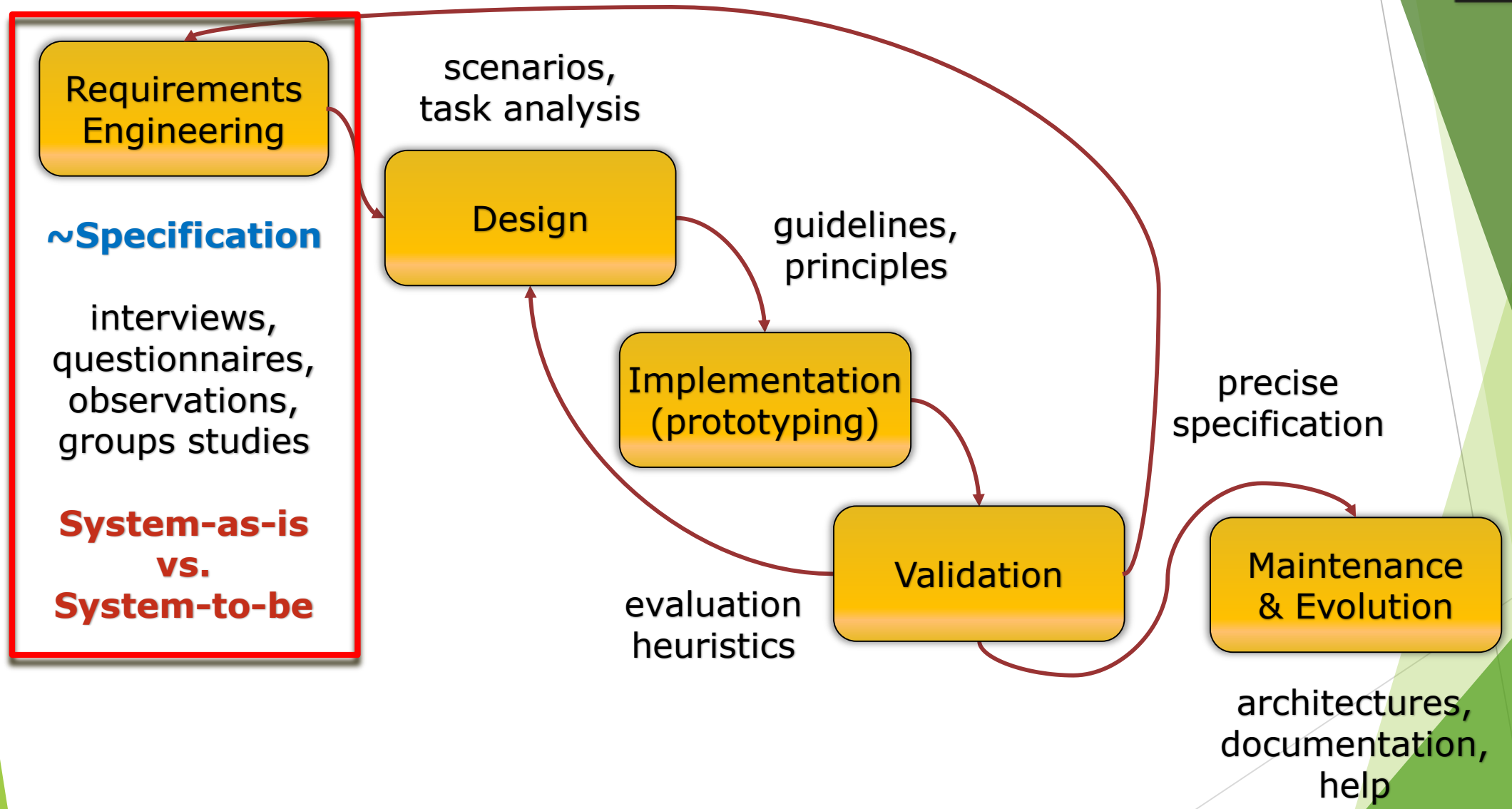
the Software Process (lifecycle)

SINGAPORE INSTITUTE OF TECHNOLOGY

- A structured set of **activities** required to develop a software system.
- Represent Software processes (lifecycle) as **models**:
  an **abstract representations** of the processes
- Many different activities in each Software Process, but all **models** involve:
  - **Specification** – defining what the system should do;
  - **Development** – design and implementation, defining the organization of the system and implementing the system;
  - **Validation** – checking that it does what the customer wants;
  - **Evolution** – changing the system in response to bugs/changing customer needs.

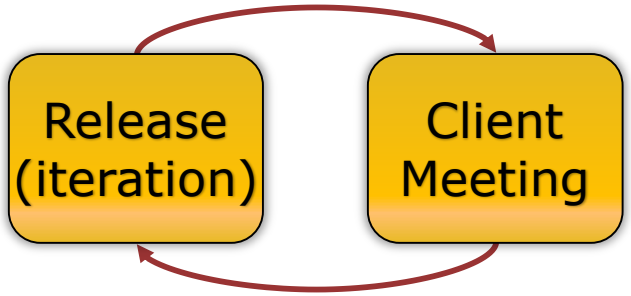


general (*ideal*) Software Engineering Process



Plan-driven and Agile processes

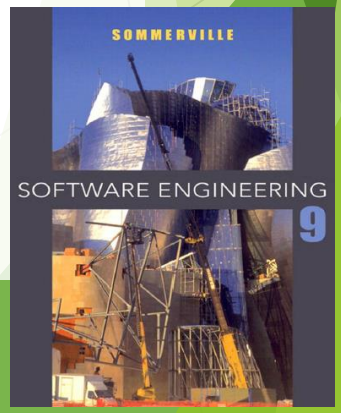
- Plan-driven processes:
  - Structured development where all activities are planned in advance
  - progress is measured against this plan (ridged)
- Agile processes:
  - Planning is incremental and it is easier to change the process to reflect changing customer requirements (reactive)



- In practice, most practical processes include elements of both plan-driven and agile approaches.
- There are no right or wrong software processes.

Process/Activities summary

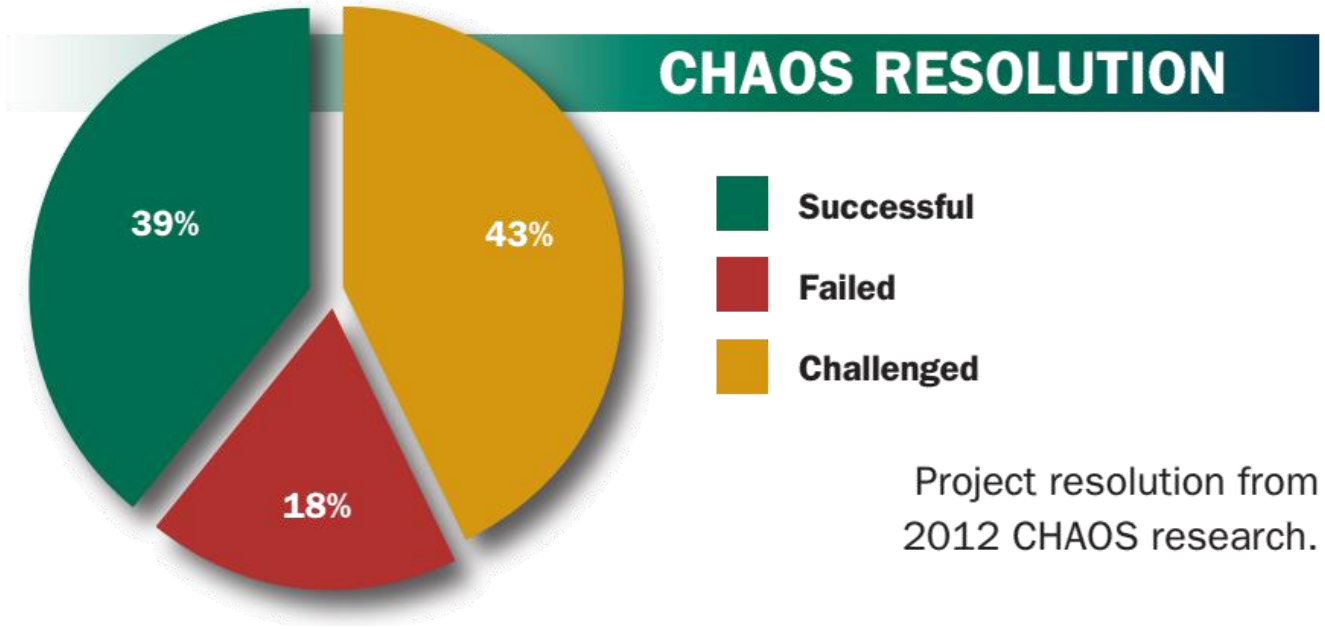
- Real software processes are **inter-leaved** sequences of technical, collaborative and managerial activities
  - Hence, **not necessarily one process fits all**
- Four general process activities of: specification, development, validation and evolution
- *So how do the processes/activities translate into successful/failed Software Engineering projects?*



SINGAPORE INSTITUTE OF TECHNOLOGY

SE Projects:
Data Analysis

Software Engineering: the reality in numbers



Project resolution from 2012 CHAOS research.

RESOLUTION

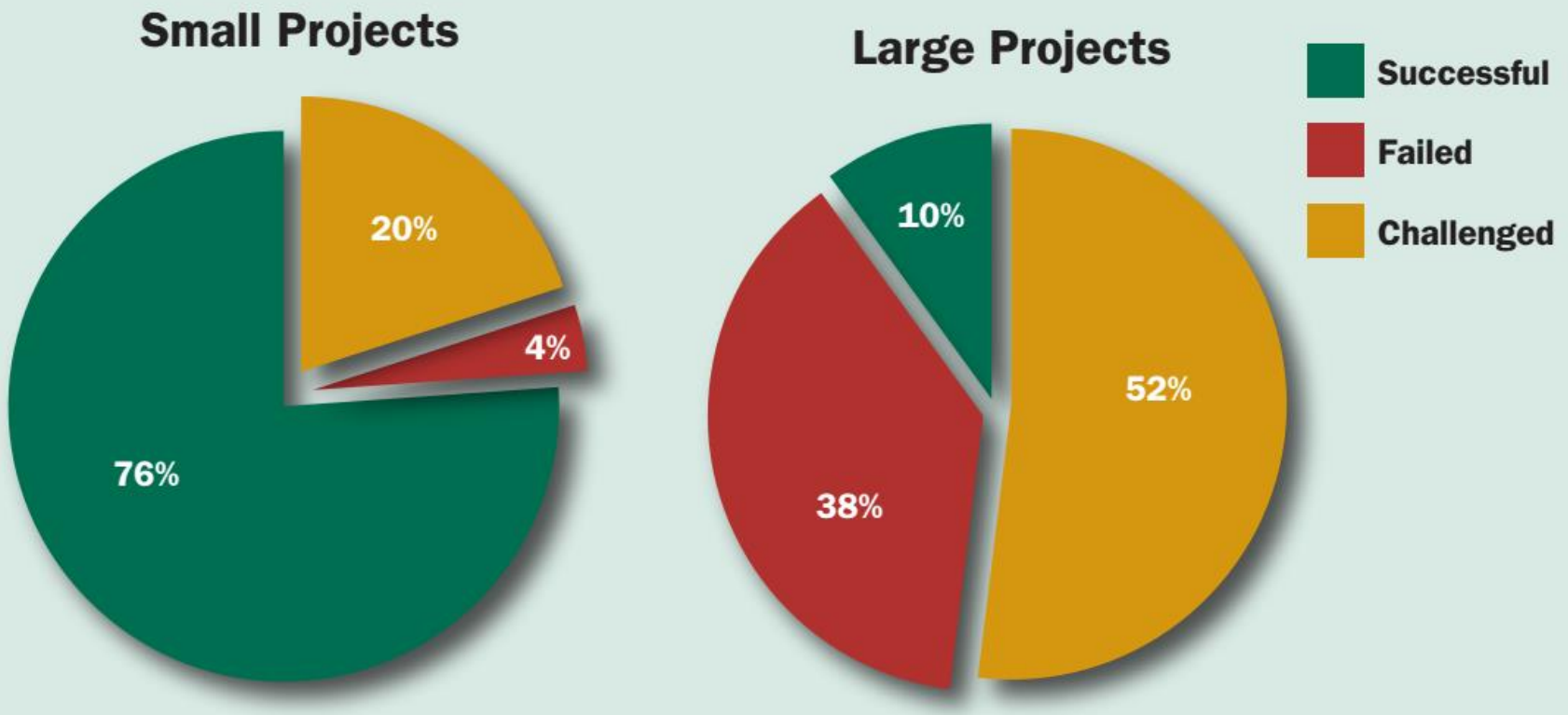
	2004	2006	2008	2010	2012
Successful	29%	35%	32%	37%	39%
Failed	18%	19%	24%	21%	18%
Challenged	53%	46%	44%	42%	43%

Project resolution results from CHAOS research for years 2004 to 2012.

Large projects are a major problem

CHAOS RESOLUTION BY LARGE AND SMALL PROJECTS

Project resolution for the calendar year 2012 in the new CHAOS database. Small projects are defined as projects with less than $1 million in labor content and large projects are considered projects with more than $10 million in labor content.

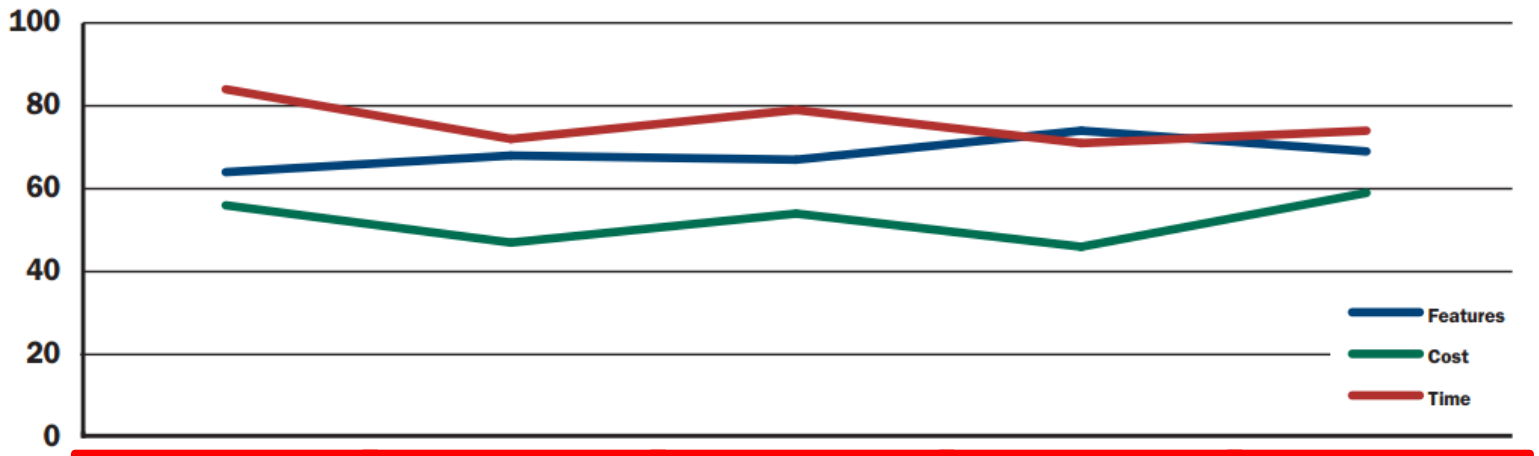


Software Engineering: the reality in numbers

SINGAPORE INSTITUTE OF TECHNOLOGY

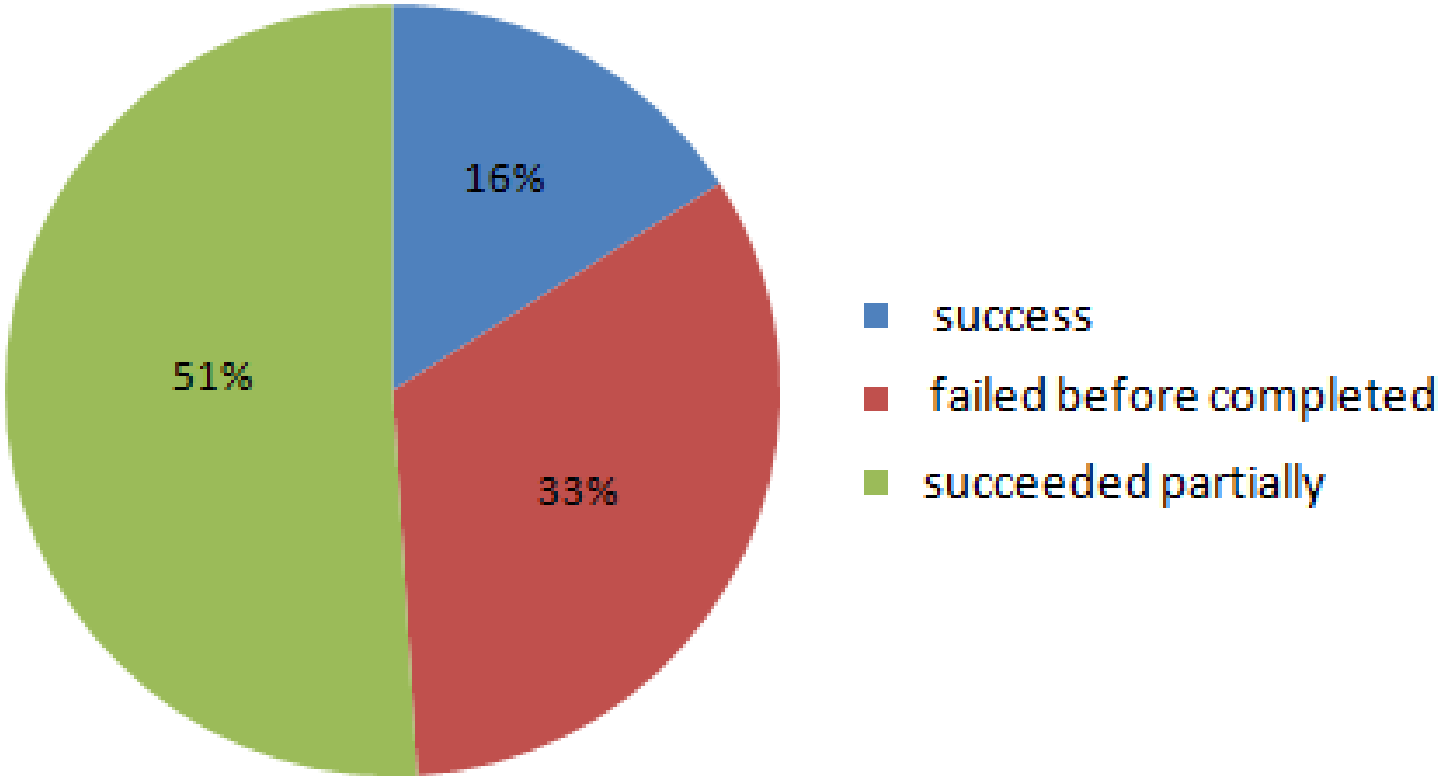
OVERRUNS AND FEATURES

Time and cost overruns, plus percentage of features delivered from CHAOS research for the years 2004 to 2012.



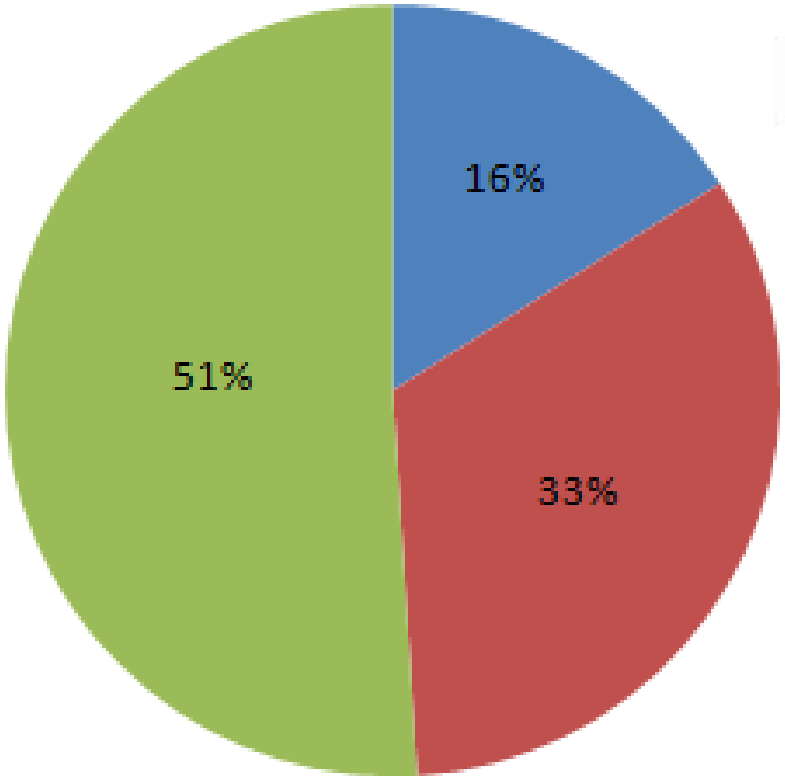
	2004	2006	2008	2010	2012
TIME	84%	72%	79%	71%	74%
COST	56%	47%	54%	46%	59%
FEATURES	64%	68%	67%	74%	69%

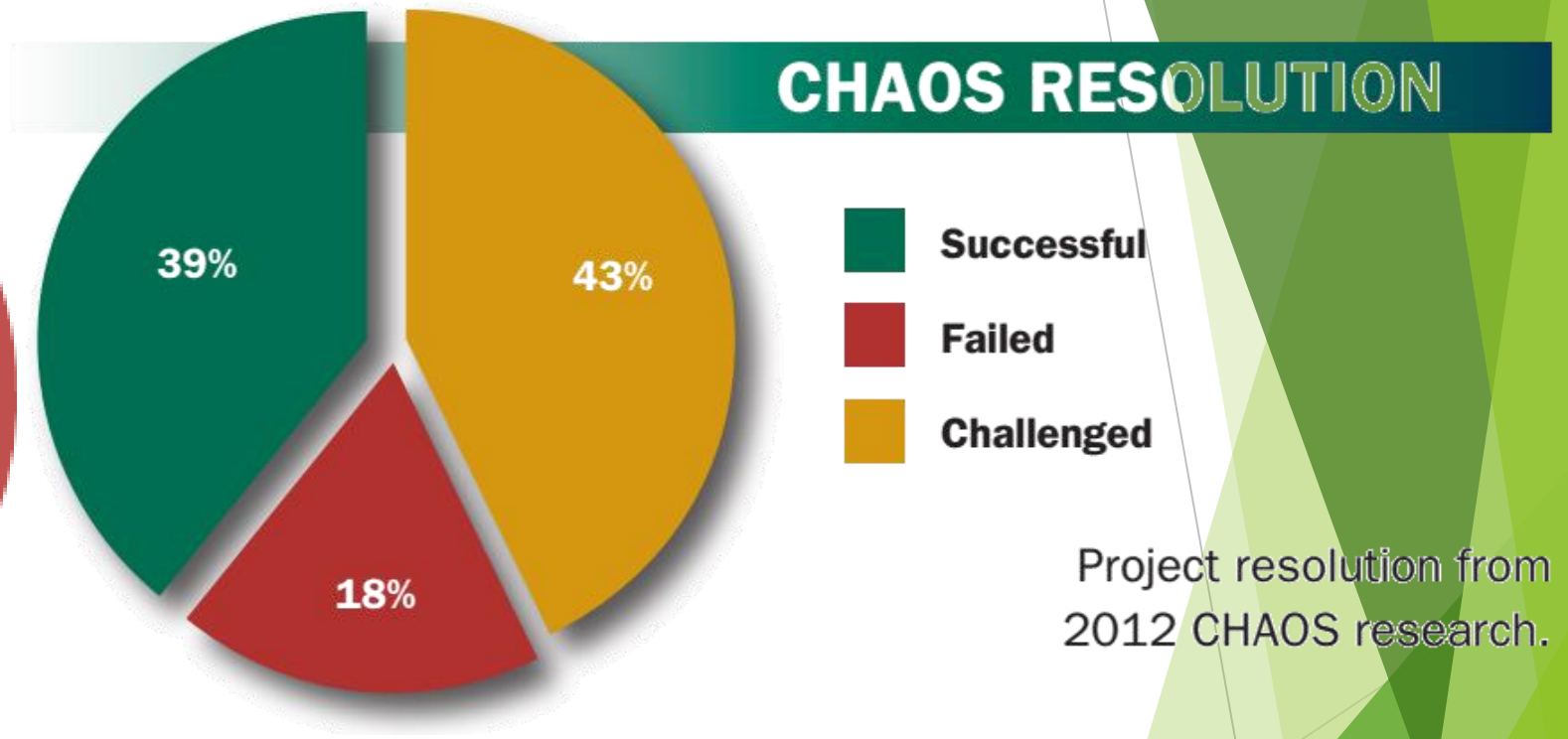
We're getting better…



Standish report, 1995

We're getting better…

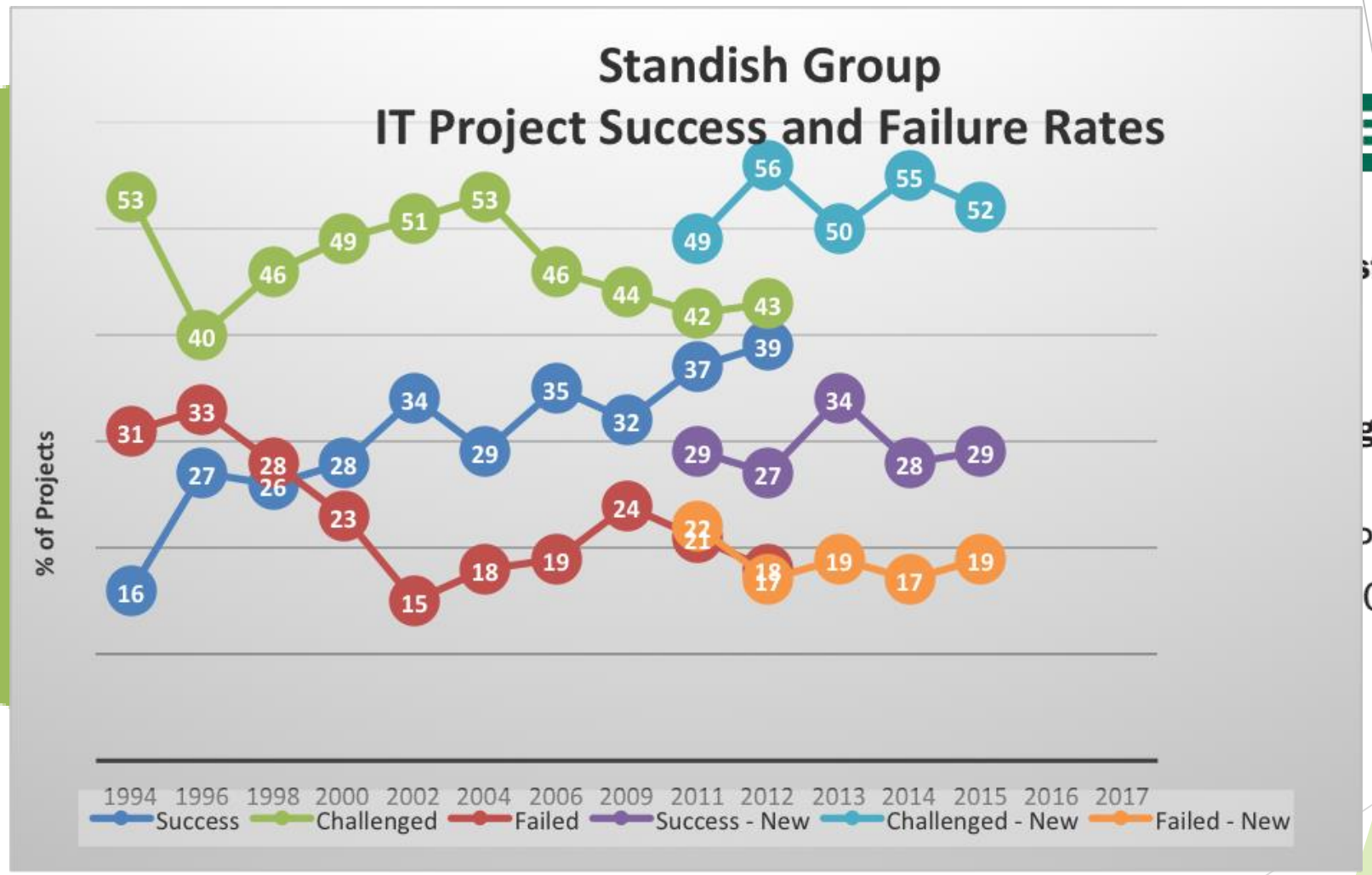




Project resolution from 2012 CHAOS research.

Standish report, 1995

We're getting better…



Standish report, 1995

SINGAPORE INSTITUTE OF TECHNOLOGY

ESOLUTION

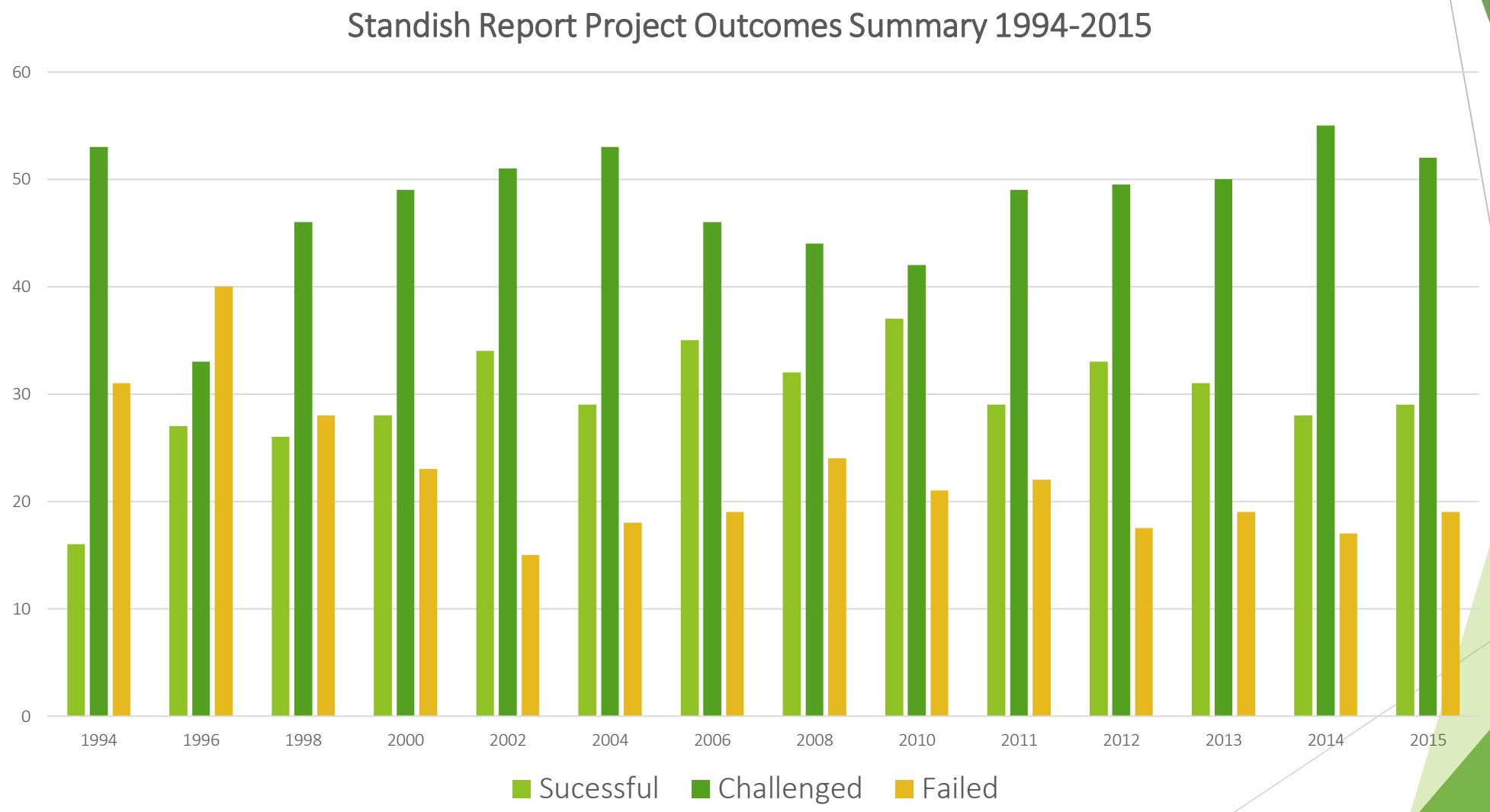
sful

ged

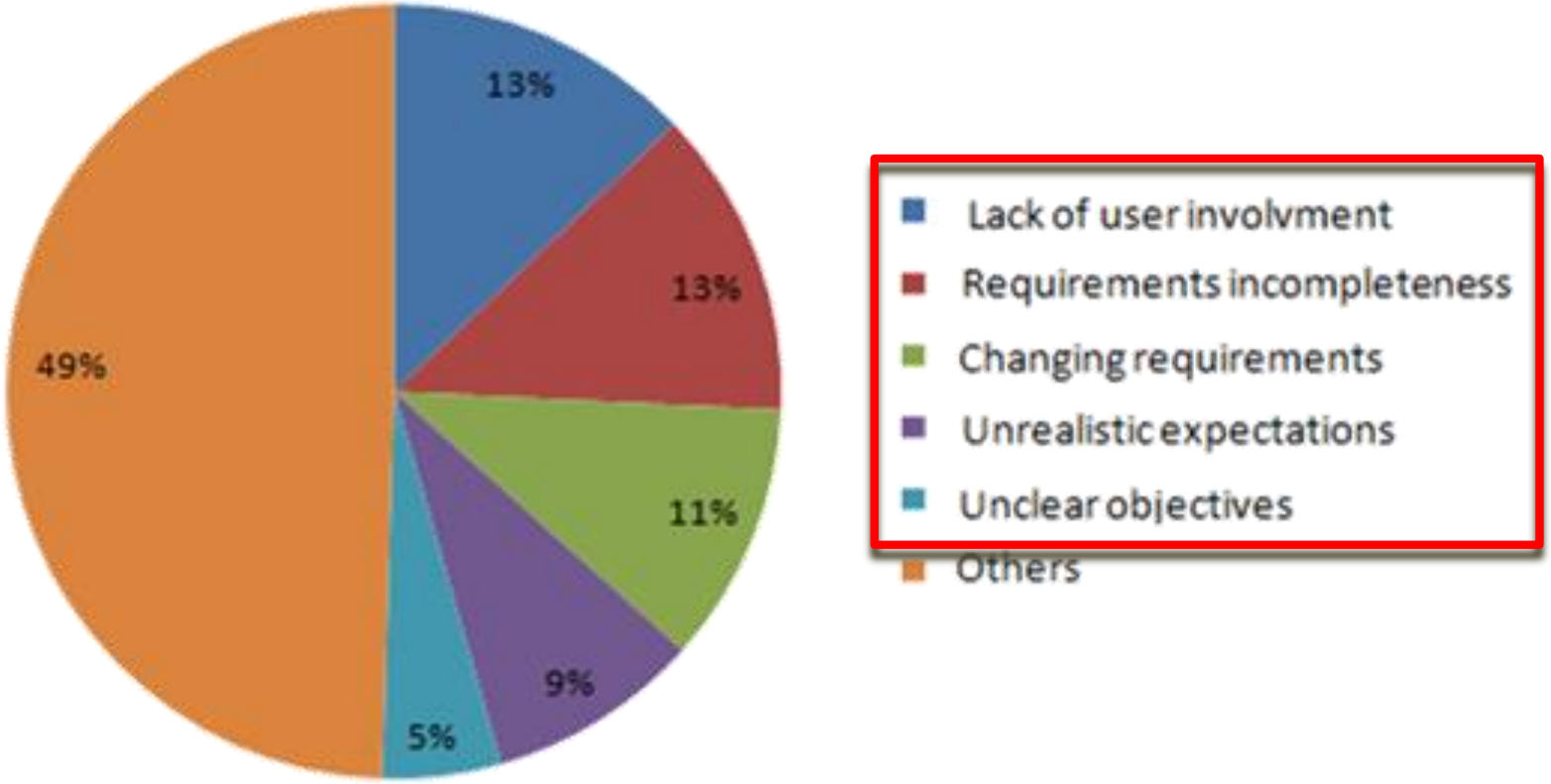
Project resolution from
012 CHAOS research.

51%

We're getting better…?

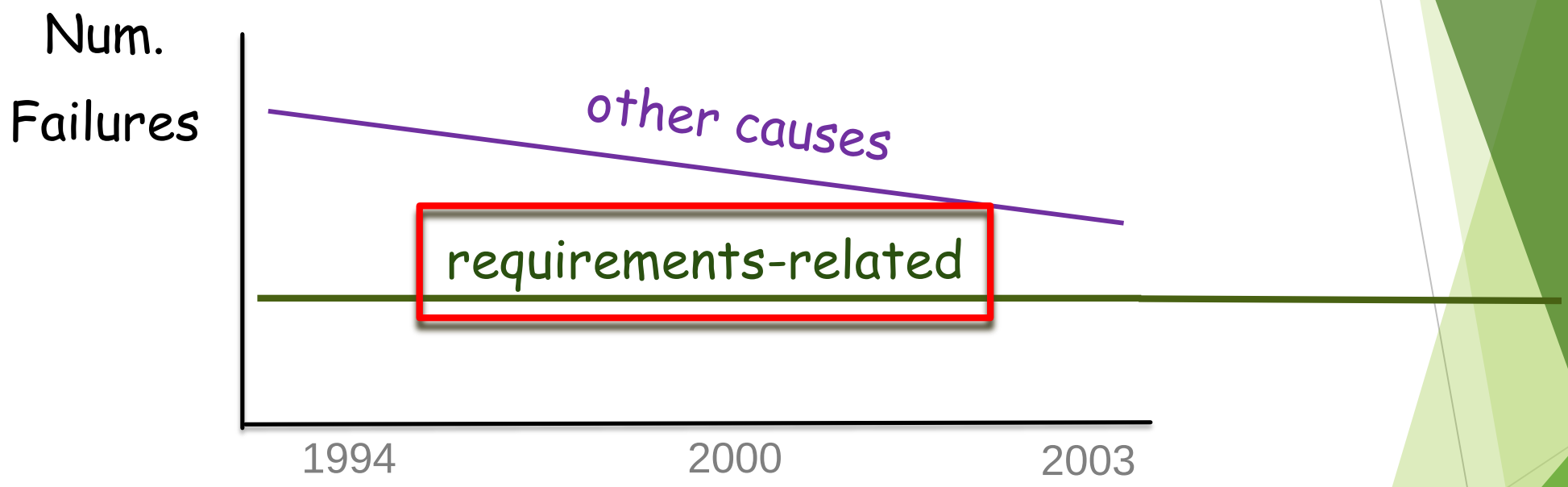


We're getting less failed projects… but… Reasons for project failure:



Standish report, 1995

However, requirements are still a major problem



- Hence we need better Requirements Engineering!

[J. Maresco, IBM developersWork, 2007]

SINGAPORE INSTITUTE OF TECHNOLOGY

MINI-QUIZ 1

miniQuiz

- Questions to help you retain this information!
- You can discuss with the people around you (2/3 people)
- *Write down*/`type up` your answers as a group
- Pass them to another group or marking
- This is a MEMORY exercise, so there is no point in cheating!

miniQuiz

- Questions

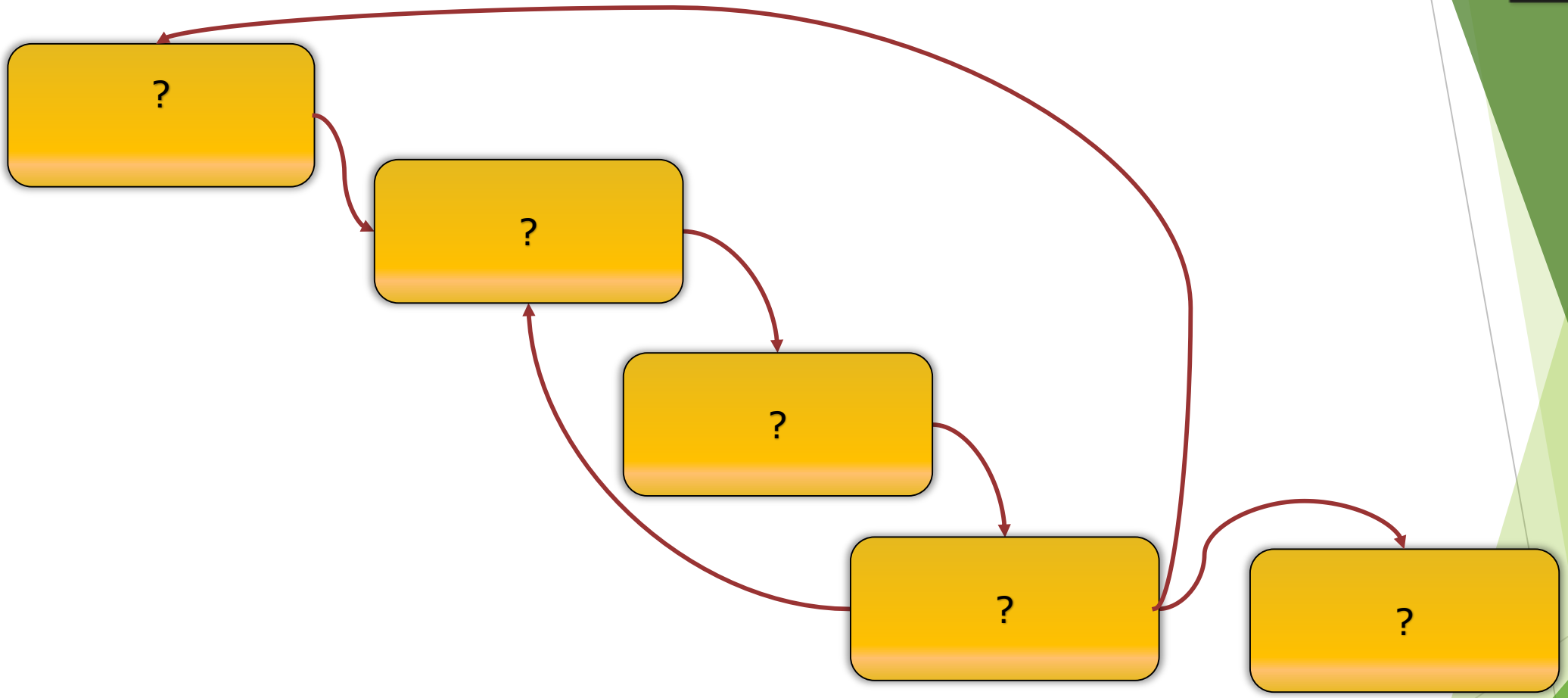
(1) Fill in the missing words:

*"The application of a **s**_____, **d**_____, **q**_____ approach to the **d**_____, **o**_____ and **m**_____ of software."*

[4 total: 0.5 marks for each correct word.
1 bonus for all 6 correct.]

miniQuiz

(2) Fill in the missing process activities:



[7 total: 1 marks for each correct activity,
2 bonus for all 5 correct.]

miniQuiz

(3) Describe what Plan-driven and Agile processes are (a few words each)

[5 total: 2 marks for each correct description,
1 bonus for all both descriptions.]

(4) Why is Requirements Engineering very important to focus on, given the project failure data we covered?

[2 total: 2 marks for reason]

Answers

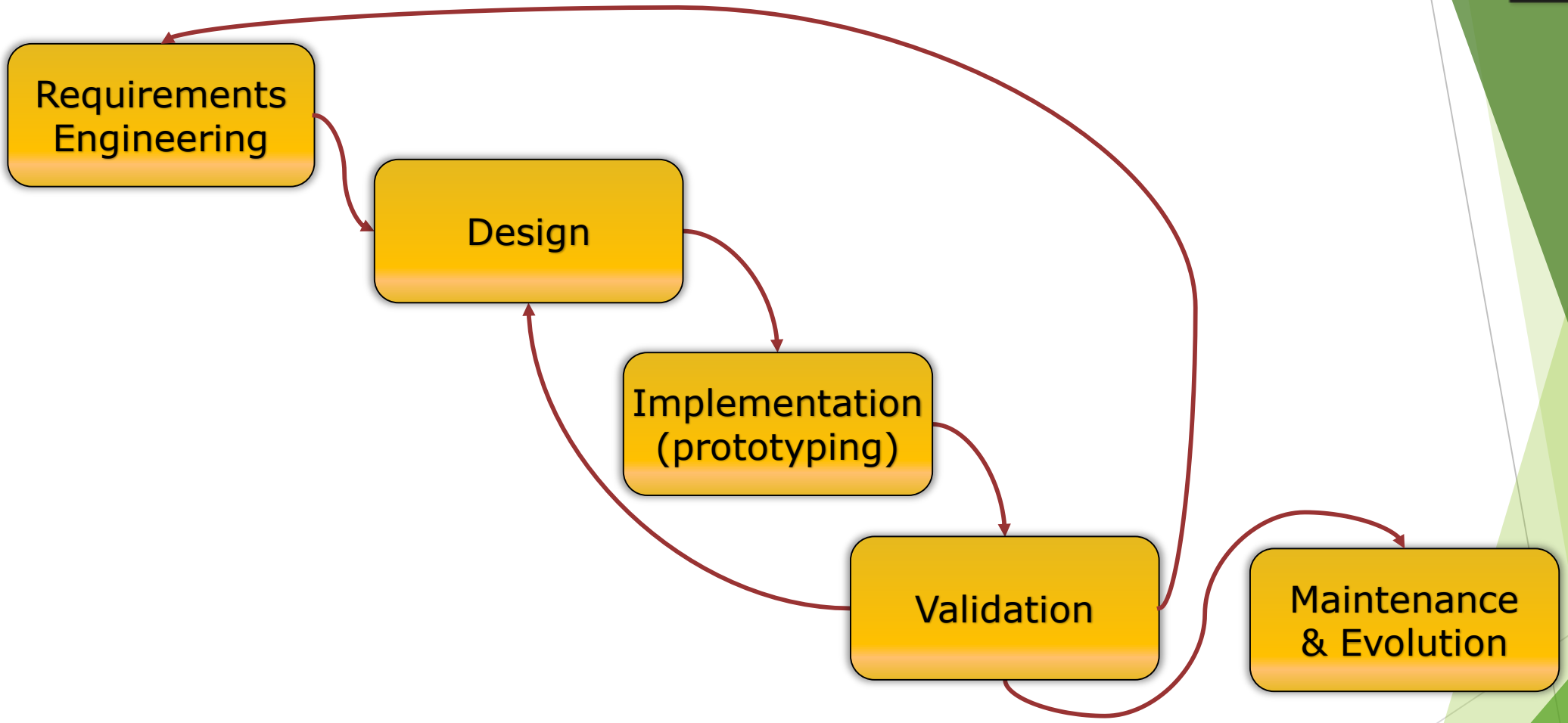
(1) Fill in the missing words:

"The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software."

[4 total: 0.5 marks for each correct word.
1 bonus for all 6 correct.]

Answers

(2) Fill in the missing process activities:



[7 total: 1 marks for each correct activity,
2 bonus for all 5 correct.]

Answers

(3) Describe what Plan-driven and Agile processes are
(a few words each)

- **Plan Driven:** Structured development where all activities are planned in advance. Progress is measured against this plan (ridged)
- **Agile:** Planning is incremental and it is easier to change the process to reflect changing customer requirements (reactive)

[5 total: 2 marks for each correct description,
1 bonus for all both descriptions.]

Answers

(4) Why is Requirements Engineering very important to focus on, given the project failure data we covered?

- It's consistently the cause of failure in Software Engineering projects. Even as Software Processes improve success rates in Software Engineering, Requirement problems remain a primary cause for projects to fail (hence our focus on learning the potential problems and mitigations here—in this module).

[2 total: 2 marks for reason]

miniQuiz

Total up marks

- Score out of 18

SINGAPORE INSTITUTE OF TECHNOLOGY

ICT2101

recap

ICT2101 Topics

- Requirements identification
  - functional, non-functional
- Requirements Terminology
  - terminology/standards
  - moving target problem
- Use Case Diagram
  - actors, actions, associations

Types of requirements

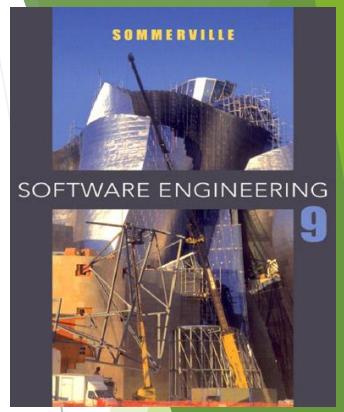
There are two major categories of requirements

- A *Functional requirement* specifies an action that the software product must be able to perform
  - Often expressed in terms of inputs and outputs
    - E.g., User can withdraw money from their account
- A *Non-functional requirement* specifies properties of the software product itself, constraints, such as
  - Platform constraints, Response times, Reliability
    - E.g., User should be able to withdraw money in less than 5mins per withdrawal

Functional requirements

- "…describe the interaction between the system and its environment, independent of its implementation"
- E.g.
  - **What** does the system do?
  - **What** are the input/output?

- Functionalities that are **externally visible** by some form of test



Non-functional requirements

- Requirements which **constrain** the implementation of the system

Product requirements

Organizational requirements

External requirements

Non-functional requirements

- Requirements which **constrain** the implementation of the system

Product requirements
Specify or constrain the behaviour of the Software • Performance requirements • Reliability requirements • Security requirements

Non-functional requirements

- Requirements which **constrain** the implementation of the system

Organizational requirements
Broad system requirements derived from policies and procedures in the organization • Operational process requirements • Development process requirements • Environmental requirements

Non-functional requirements

- Requirements which **constrain** the implementation of the system

External requirements

Derived from factors external to the system and its development process

- Regulatory requirements
- Legislative requirements
- Ethical requirements

Non-functional requirements

- Requirements which **constrain** the implementation of the system

Product requirements

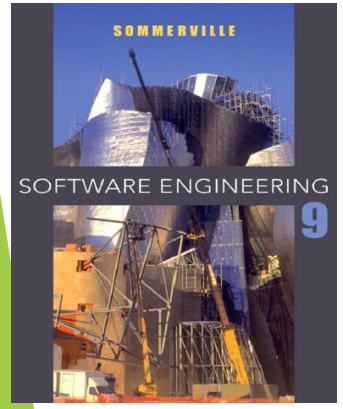
Organizational requirements

External requirements

Non-functional requirements: the bad and the good

"The system should be easy to use by medical staff and should be organized in such a way that user errors are minimized."

"Medical staff shall be able to use all the system functions after four hours of training. After this training, the average number of errors made by experienced users shall not exceed two per hour of system use."



Functional vs Non-functional requirements

- Functional requirements are derived from the requirements engineering and analysis activities

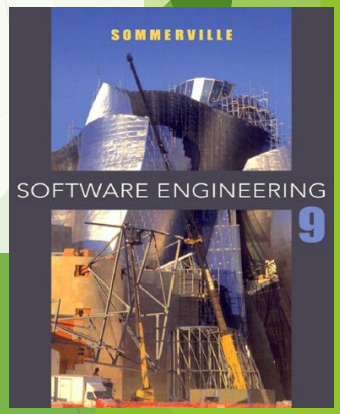
- Some Non-functional requirements have to wait until the design activities
  - Detailed information for Non-functional requirements may be **unavailable** until requirements and analysis activities are complete

Requirements
Terminology/Standards

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- Everything that matters is covered in the document. All conditions under which the requirement applies are stated.



Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- The facts related to the requirement are accurate, and it is technically and legally possible.

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- There are no conflicts between requirements

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- Every statement has one interpretation that can be easily derived from the requirement statement.

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- The built system can be checked to see if requirement is met.

SINGAPORE INSTITUTE OF TECHNOLOGY

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- Changes to the requirements can be accommodated.

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- The source of the requirement can be traced, and it can be tracked throughout the system (e.g., to the design, code, test, and documentation).

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- The requirement is doable and can be accomplished within budget and schedule.

Requirements should be…

- Complete
- Correct
- Consistent
- Unambiguous
- Verifiable
- Modifiable
- Traceable
- Feasible
- Design independent

- It does not pose a specific implementation solution.
  - *Ideally*

Moving Target problem

Dangers

- Requirements changed as software develops

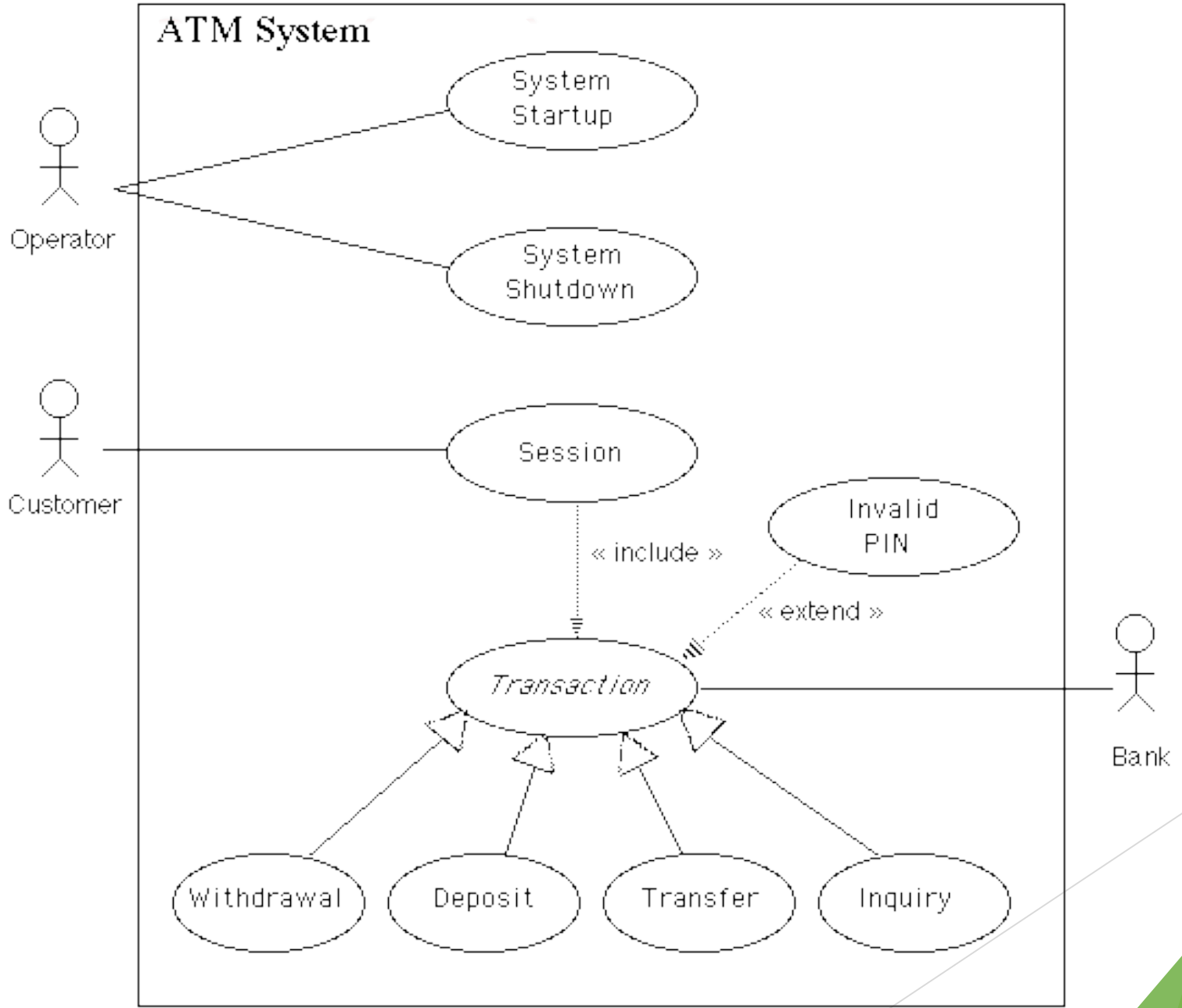
- Regression faults
  - Changes in some areas cause errors in seemingly unrelated aspects of the software
- Redesign/Reimplementation
  - Too many changes to the requirements lead to a huge alteration in the system

Moving target problem

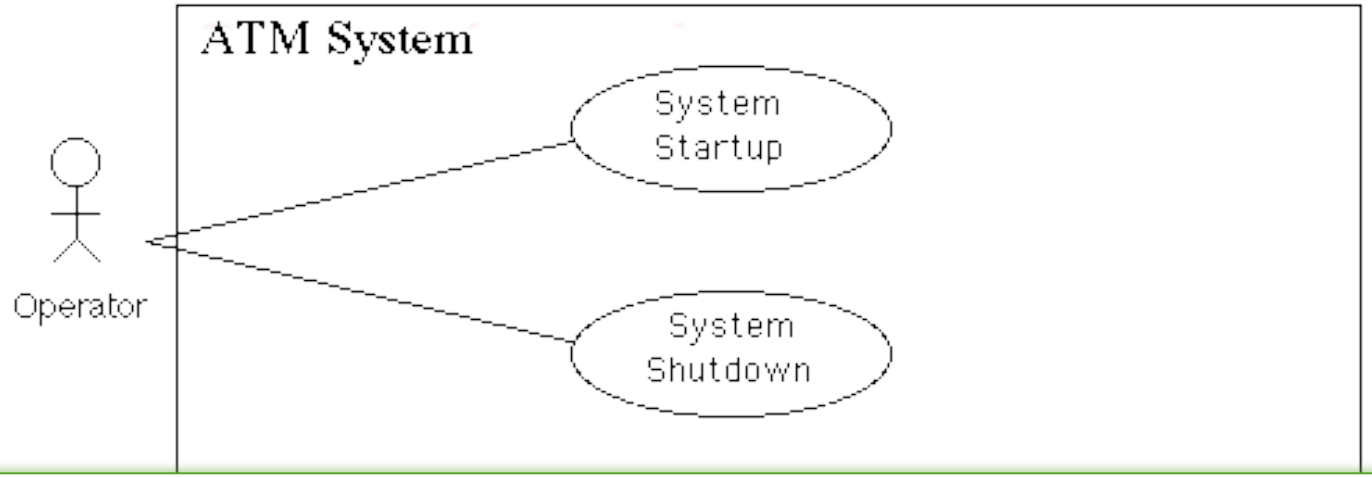
- Change is **inevitable**
  - Growing companies are always going to change
  - If the **client** calling for changes has sufficient clout, nothing can be done about it
- Even if the reasons for the change are good, the software product can be adversely impacted
- There is *no solution* to the moving target problem

UML Use Case Diagram (basics)

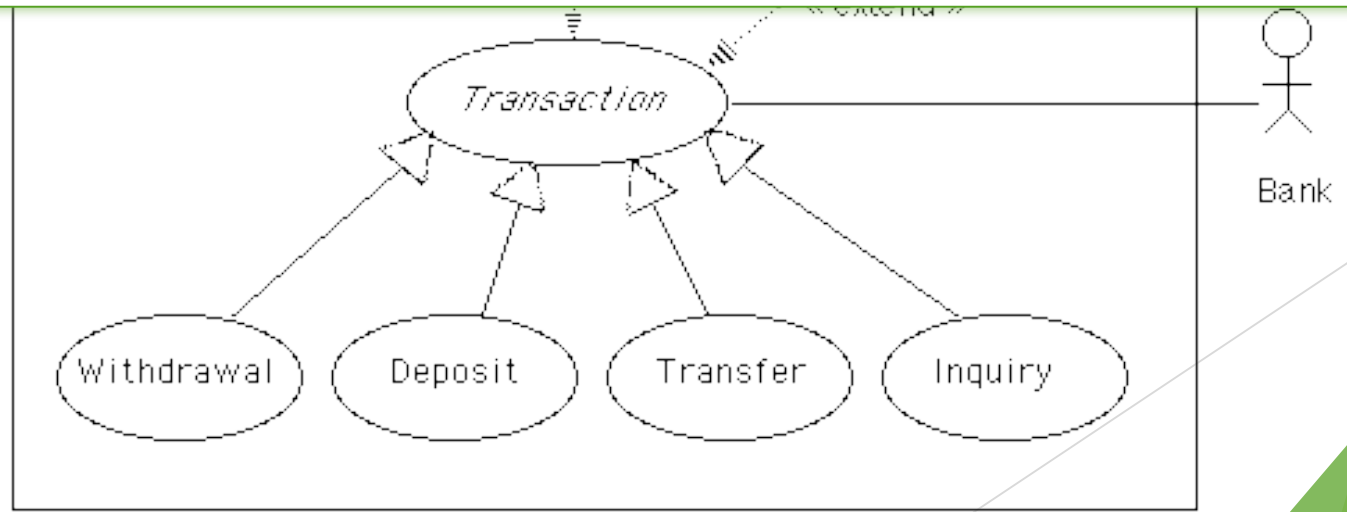
Use Case diagram (UML)



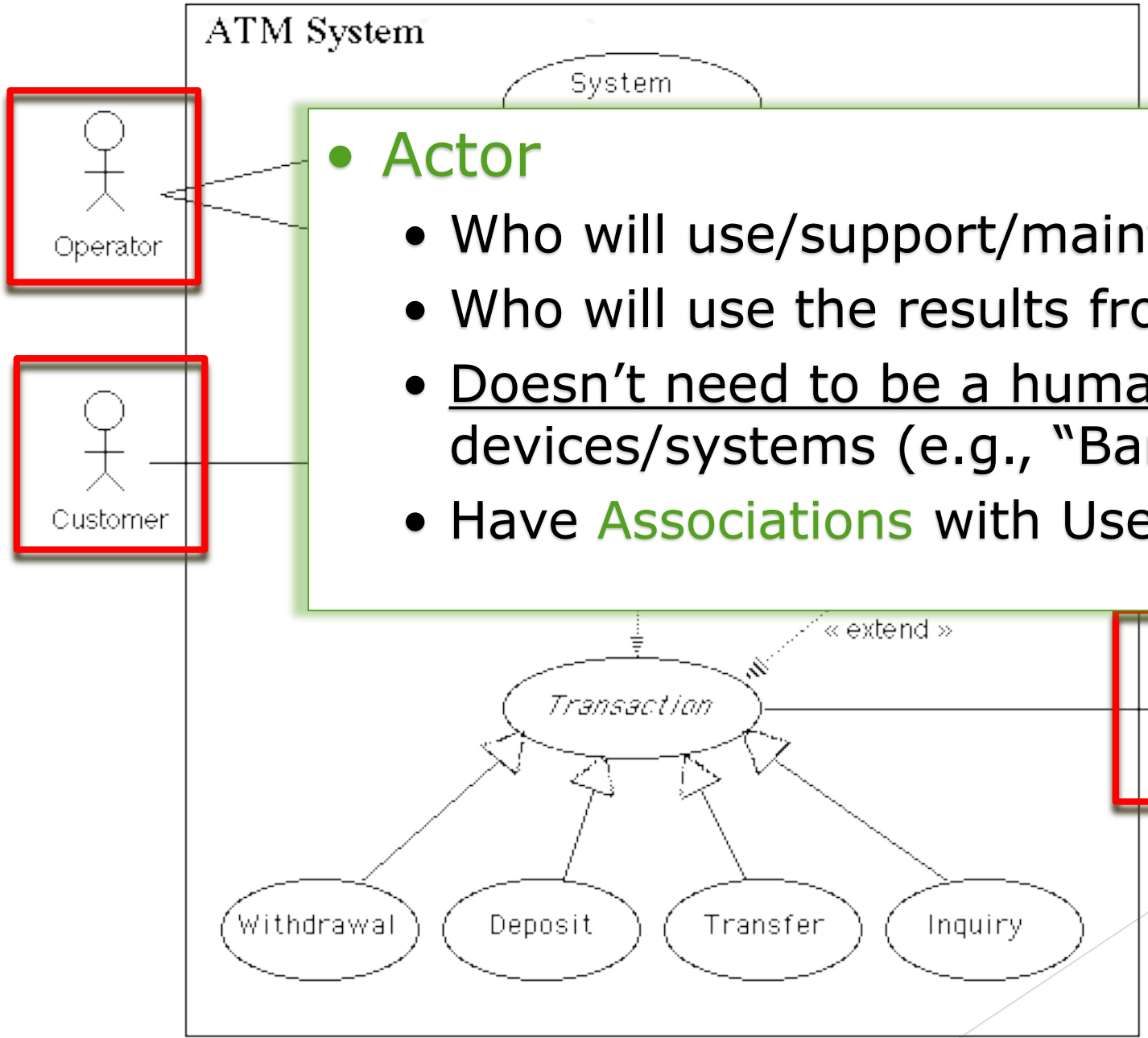
Use Case diagram (UML)



- Represents: Actors, Actions and Associations, and how they relate to Use Cases

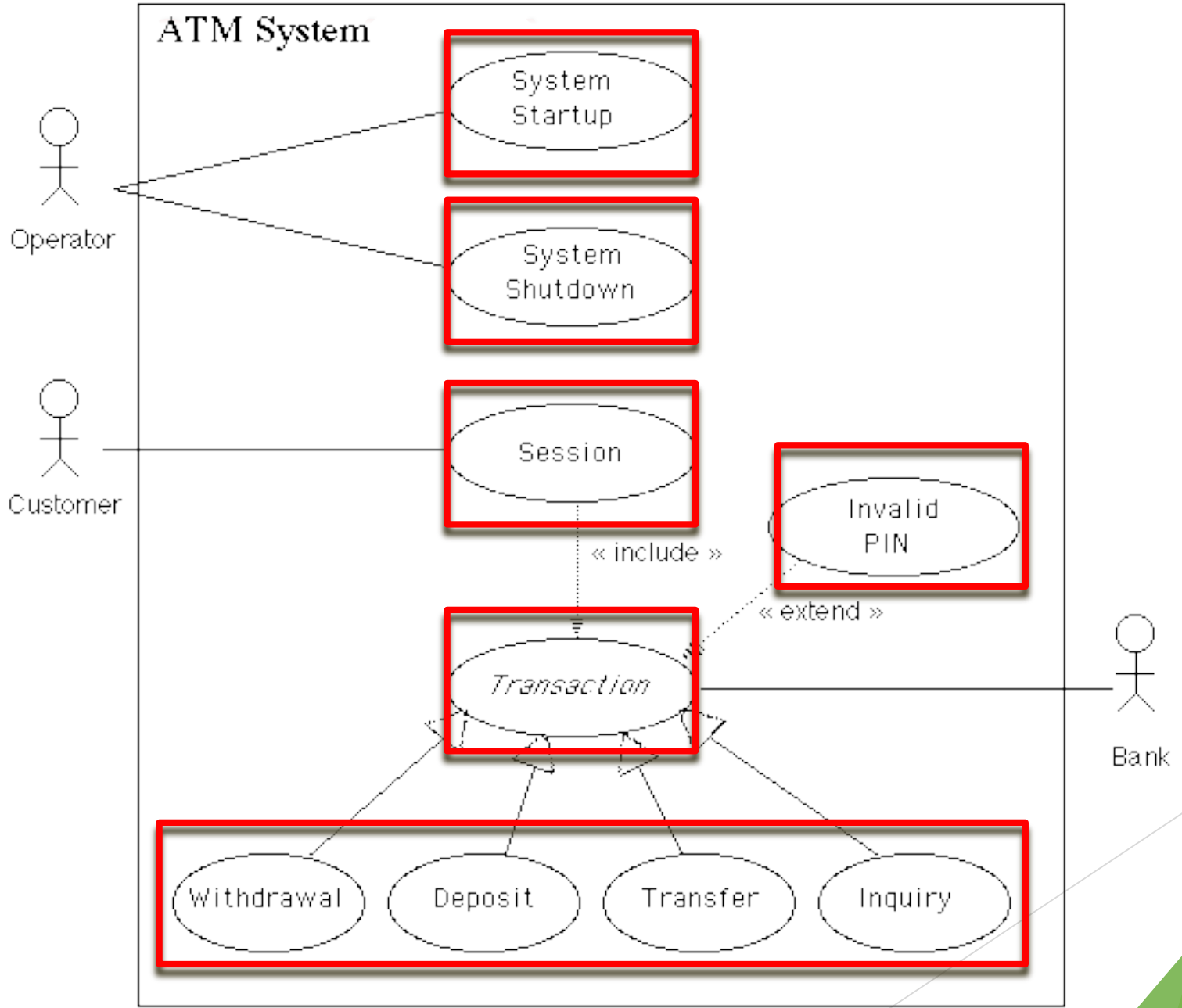


Use Case diagram (UML)

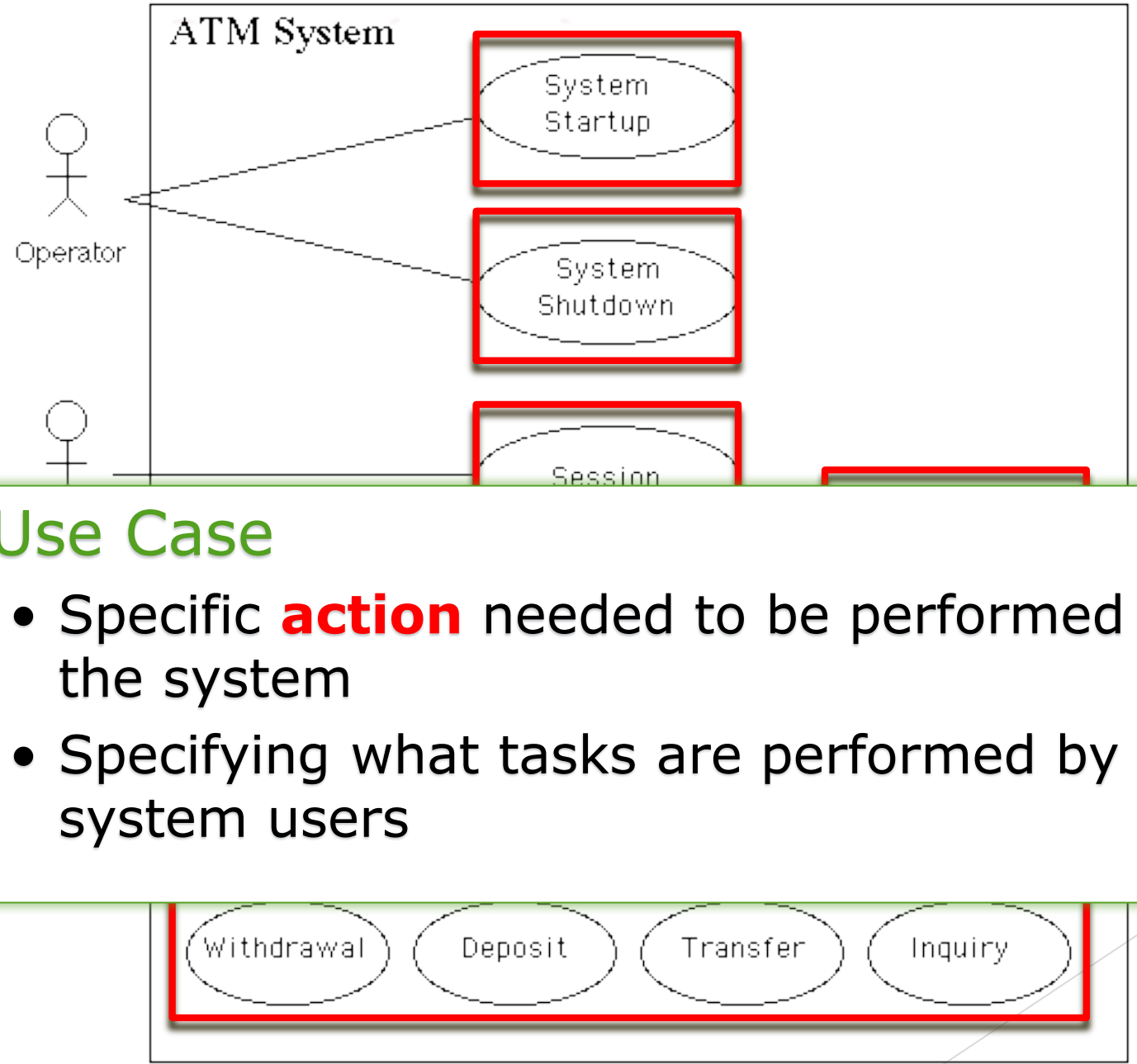


- Actor
  - Who will use/support/maintain the system?
  - Who will use the results from the system?
  - Doesn't need to be a human: external devices/systems (e.g., "Bank")
  - Have Associations with Use Cases

Use Case diagram (UML)



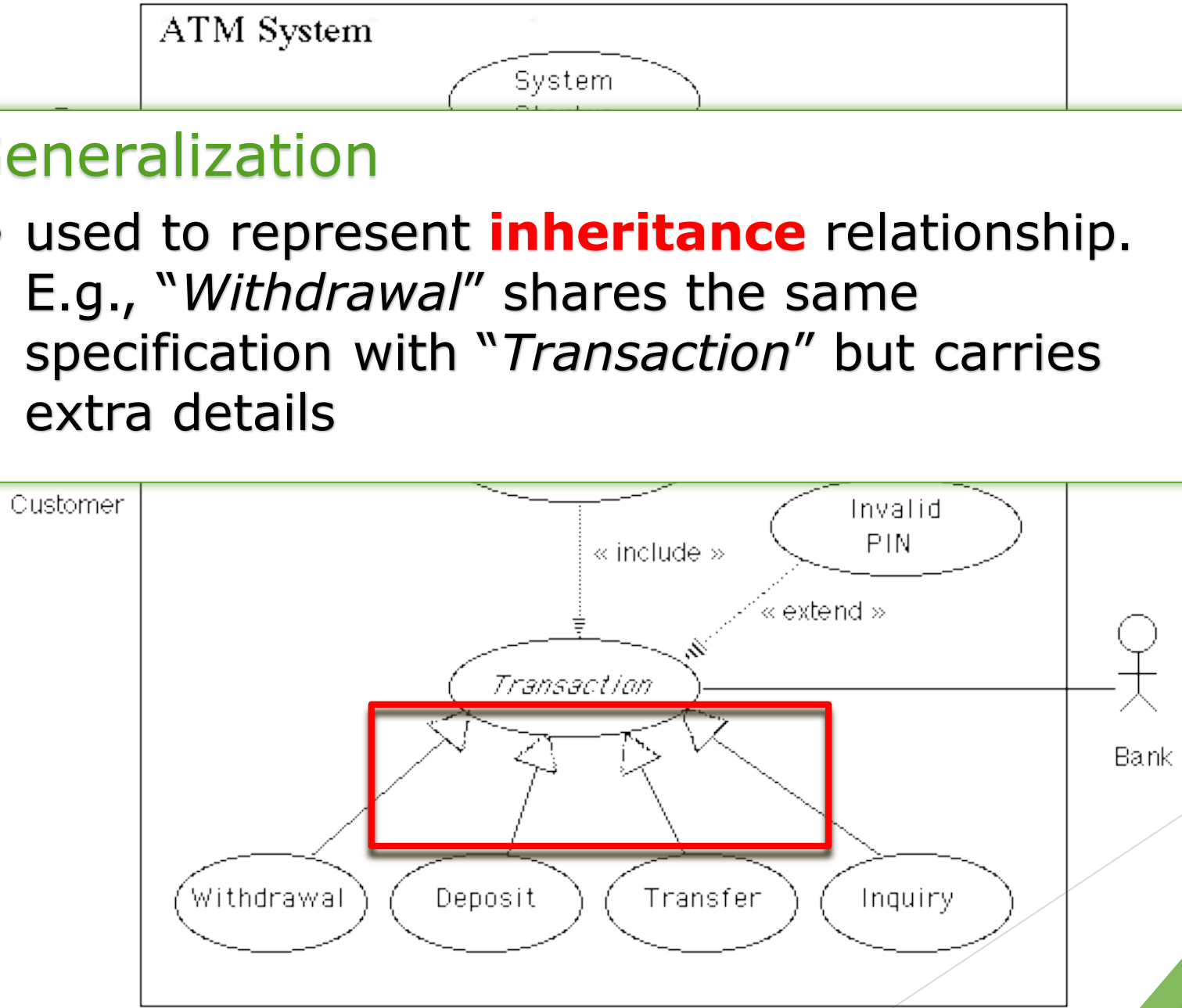
Use Case diagram (UML)



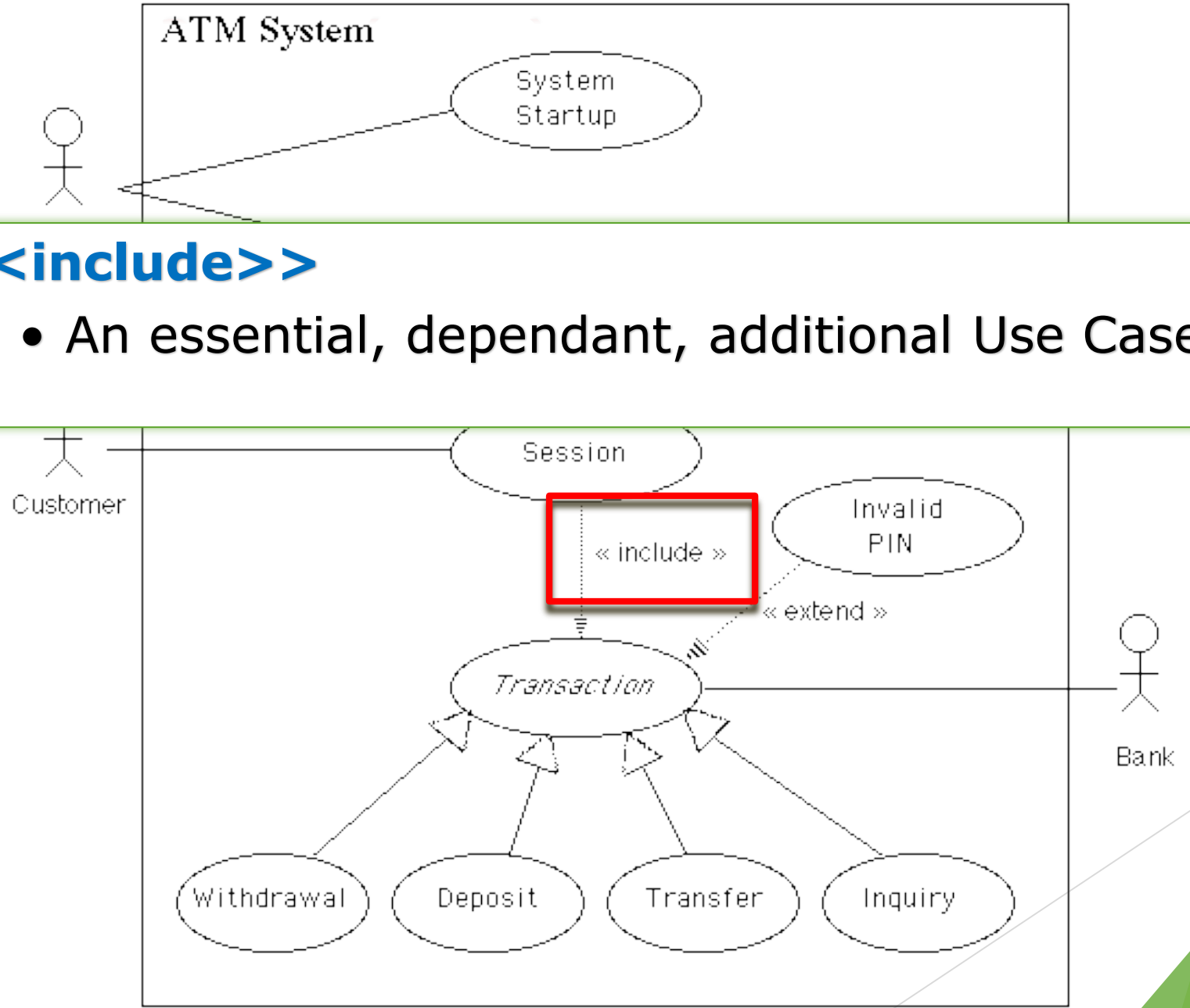
- Use Case
  - Specific **action** needed to be performed in/by the system
  - Specifying what tasks are performed by the system users

Use Case diagram (UML)

- Generalization
  - used to represent **inheritance** relationship. E.g., "*Withdrawal*" shares the same specification with "*Transaction*" but carries extra details



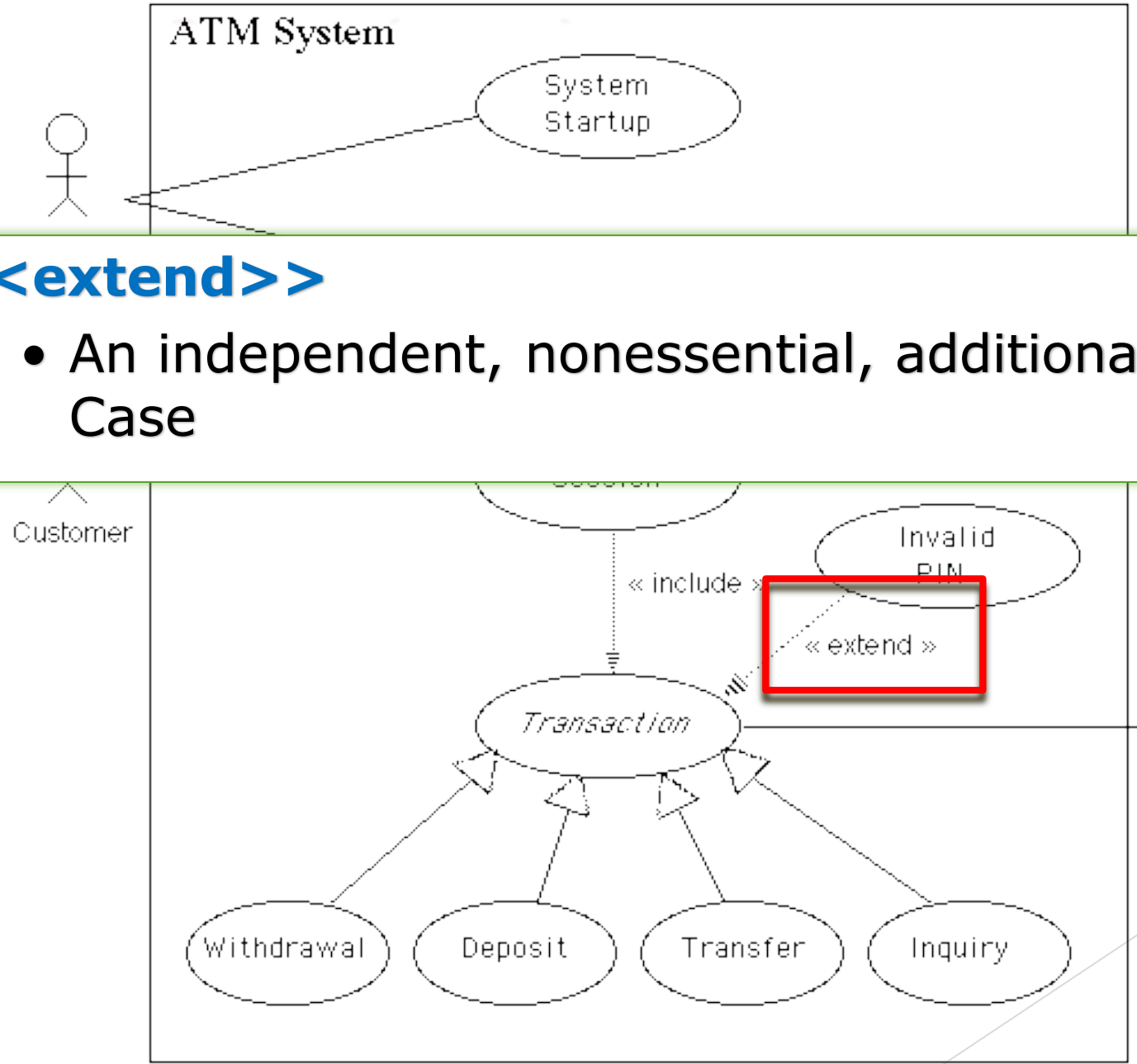
Use Case diagram (UML)



<<include>>

- An essential, dependant, additional Use Case

Use Case diagram (UML)



<<extend>>

- An independent, nonessential, additional Use Case

SINGAPORE INSTITUTE OF TECHNOLOGY

MINI-QUIZ 2

miniQuiz

- Questions to help you retain this information!
- You can discuss with the people around you (2/3 people)
- *Write down*/`type up` your answers as a group
- Pass them to another group or marking
- This is a MEMORY exercise, so there is no point in cheating!

miniQuiz

- Questions

(1) What is the difference between a functional and non-functional requirement (define each)?

[5 total: 2 marks for each correct definition. 1 bonus for both]

(2) What is the moving target problem?

[3 total: for description]

(3) What are the 2 main dangers of the moving target problem?

[5 total: 2 for each. 1 bonus for both]

miniQuiz

(4) Name 3 things that “Requirements should be…”, with each name write a short description.

[12 total: 1 marks for each correct name, 2 for each description. 3 bonus for all names and descriptions]

(5) Name the 3 categories for Non-functional requirements, and describe one of them.

[7 total: 1 for each name. 2 for correct description. 2 bonus for all names and description]

miniQuiz

(6) In a UML Use Case diagram, what are the following:

- Actor
- Use Case
- Generalisation relationship
- Include relationship
- Extend relationship

[14 total: 2 marks for each correct description.
4 bonus for all]

Answers

(1) What is the difference between a functional and non-functional requirement (define each)?

- *Functional:* specifies an action that the software product must be able to perform
- *Non-functional:* specifies properties of the software product itself, constraints,

[5 total: 2 marks for each correct definition. 1 bonus for both]

Answers

(2) What are is the moving target problem?

- the problem with requirements changing as software develops

[3 total: for description]

(3) What are the 2 main dangers of the moving target problem?

- Regression faults
  - Changes in some areas cause errors in seemingly unrelated aspects of the software
- Redesign/Reimplementation
  - Too many changes to the requirements lead to a huge alteration in the system

[5 total: 2 for each. 1 bonus for both]

Answers

(4) Name 3 things that "Requirements should be…", with each name write a short description.

- **Complete** — Everything that matters is covered in the document. All conditions under which the requirement applies are stated.
- **Correct** — The facts related to the requirement are accurate, and it is technically and legally possible.
- **Consistent** — There are no conflicts between requirements.
- **Unambiguous** — Every statement has one interpretation that can be easily derived from the requirement statement.
- **Verifiable** — The built system can be checked to see if requirement is met.
- **Modifiable** — Changes to the requirements can be accommodated.
- **Traceable** — The source of the requirement can be traced, and it can be tracked throughout the system (e.g., to the design, code, test, and documentation).
- **Feasible** — The requirement is doable and can be accomplished within budget and schedule.
- **Design independent** — It does not pose a specific implementation solution.

[12 total: 1 marks for each correct name, 2 for each description.
3 bonus for all names and descriptions]

Answers

(5) Name the 3 categories for Non-functional requirements, and describe one of them.

- ***Product** req:* Specify or constrain the behaviour of the Software
- ***Organisational** req:* Broad system requirements derived from policies and procedures in the organization
- **External** req: Derived from factors external to the system and its development process

[7 total: 1 for each name. 2 for correct description. 2 bonus for all names and the one description]

Answers

(6) In a UML Use Case diagram, what are the following:

- Actor: entity that interacts with the system
- Use Case: specific **action** needed to be performed in/by the system
- Generalisation relationship: inheritance between Use Cases
- Include relationship: An essential/dependant additional Use Case
- Extend relationship: An independent/nonessential, additional Use Case

[14 total: 2 marks for each correct description.
4 bonus for all]

miniQuiz

Total up marks

- Score out of **46**
- *Total score today out of **64***

☺!Well Done!☺

Interview skills (CCS)*:

- *"…an individual assessment that we'll cover in detail at the end"….*

IE., now

- Assessed in wk3, 4% of module grade (individually assessed)

- Each lab group ***will*** be divided into two rooms. This info will be confirmed by next lecture.
  - Please check the LMS for updates

- During the session you'll be advised on one of the most important aspects of Requirements Engineering: interview skills.
  - Focus on *soft-skills* (critical to creating constructive conversation)
  - Week 2 will be a formative session
  - Assessment will have allotted timeslots in week 3 TBC (CCS will inform more next week)

SINGAPORE INSTITUTE OF TECHNOLOGY

This Week:
Project Info Released
Next Week:
Requirements Engineering