Chapter 1
The Uniqueness of Software Quality Assurance

UNIT-1 (PART-1)

The Software Quality Challenge

- **This chapter is essentially about two major topics:**
  - **The uniqueness of software quality assurance**
  - **The environments for which SQA methods are developed.**

Introduction

- Why study Quality Assurance and Testing?
- With all the methodology wars, numerous processes, huge number of tools to assist in software development, why this **separate topic**?
- What makes SQA important that it deserves so much attention?
- SQA is a key course in software engineering curricula.

Case study

- In February,1995; opening of the new Denver International Airport (DIA).
- Planned; Serve 110,000,000 passengers/day with 1750 flights daily , 200 gates and 12 runways.

Case study

- ✕ Operations were delayed by 16 months.
- ✕ Failure of Software-base baggage handing system.

$ 2 billion

Differences between Software Products and Industrial Products

- **High complexity**
  - **Product complexity can be measured by no. of operational modes the product permits.**
  - **The potential ways in which a software product can be used with different data / data paths reflecting different incoming data is almost infinite.**
  - **Manner in which industrial products can be used are usually well-defined. E.g. A machine & its settings.**
  - **Think about software:**
    - **every loop with different values of data reflects a different opportunity to see software fail.**
    - **In truth, the number of paths through a non-trivial software product is infinite.**

Differences between Software Products and Industrial Products

- **Invisibility of the product**
  - **In an industrial product, missing parts are obvious.**
    - **Something missing? Easily identified. (e.g. A door missing from new car)**
  - **Not so in software products.**
    - **May not be noticeable for years – if at all!**
    - **Parts may have never been in the software ever!**

Differences between Software Products and Industrial Products

- Product Development and Production Process
- Product Development for **Industrial Products**:
  - Product Development –
    - Designers and QA people check / test the prototype for defects
  - Product Production Planning –
    - Here, production process and tools are designed and prepared.
    - May require special production line to be designed and built
    - Lots of opportunities to check for defects that escaped reviews and tests conducted during development
  - Manufacturing –
    - QA procedures are applied to detect failures of the products themselves during manufacturing.
    - Can be corrected by change in the design or in production tools and this will change the way products are manufactured in the future.

Differences between Software Products and Industrial Products

- Will see:
  - Comparing industrial products with software products we see that the **only phase** when defects can be corrected is really in the **product development phase**.

- Let's look at the same three activities for Software Products:

Differences between Software Products and Industrial Products

- Product Development and Production Process
- Product Development for **Software Products**:
- **Product Development** – Best Chance to Detect Errors!
  - Here, we look for inherent product defects and hope to arrive at an acceptable prototype.
- Product Production Planning
  - This phase is **not required** for software production process. Copies are simply reproduced / printed automatically….
  - Numbers of copies of no consequence.
- Manufacturing
  - Manufacturing limited to copying product / printing copies of manuals.
  - Chances for detecting defects here is quite limited.

Only Chance to Discover Defects:

- **Best chance to really detect defects occurs during the software development process itself!**
- "The need for special tools and methods for the software industry is reflected in the professional publications as well in special standards devoted to SQA, such as ISO 9000-3, "Guidelines for the application of ISO 9001 to the development, supply, and maintenance of software."
- Another: ISO 9004-2: "Quality Management and Quality Systems Elements: Guidelines for the Services."
- These characteristics of software –
  - complexity,
  - invisibility, and
  - limited opportunity to detect bugs
- has led to the development of the ISO Guidelines and an awareness of real SQA methodology.

Software v/s Industrial Product

Table 1.1: Factors affecting defect detection in software products vs. other industrial products

Characteristic	Software products	Other industrial products
Complexity	Usually, very complex product allowing for very large number of operational options	Degree of complexity much lower, allowing at most a few thousand operational options
Visibility of product	Invisible product, impossible to detect defects or omissions by sight (e.g. of a diskette or CD storing the software)	Visible product, allowing effective detection of defects by sight
Nature of development and production process	Opportunities to detect defects arise in only one phase, namely product development	Opportunities to detect defects arise in all phases of development and production: - Product development - Product production planning - Manufacturing

The Environment for which SQA Methods are Developed

- Let's look at the **environment** for professional software development and maintenance
- Main Characteristics:
  1. Contractual Conditions
  2. Subjection to Customer – Supplier Relationship
  3. Required Teamwork
  4. Cooperation and coordination with other teams.
  5. Interfaces with other software systems
  6. The need to continue carrying out a project despite team member changes
  7. The need to continue carrying out software maintenance for an extended period.

The Environment for which SQA Methods are Developed

1. Contractual Conditions
   - These include functional requirements, the project budget, and the project timetable.
   - These are the biggees!
   - Delivering software on time, within budget that meets or exceeds the functional requirements constitutes the thrust of contracts.

The Environment for which SQA Methods are Developed

2. Subjection to Customer-Supplier Relationship
   - We must realize that the customer drives the process in many cases – submitting changes, evaluating deliverables, approving the deliverables
   - This is the relationship that is critical when software is developed by software professionals.
3. Required Teamwork
   - Three very motivating factors for project team v/s an individual:
     - Timetable requirements – team members work together
     - Have a variety of specializations
     - Mutual support and review to enhance product quality

The Environment for which SQA Methods are Developed

4. Cooperating and coordination with other software teams
   - In a large software development organization, there are many teams (hardware devp. As well as software devp.) and may need to cooperate with them
   - Expertise may exist in another team.
   - Software development may be outsourced in part.
   - Other teams may have developed similar software for the client and can offer tremendous help.
   - Conflicts
   - Escalation!

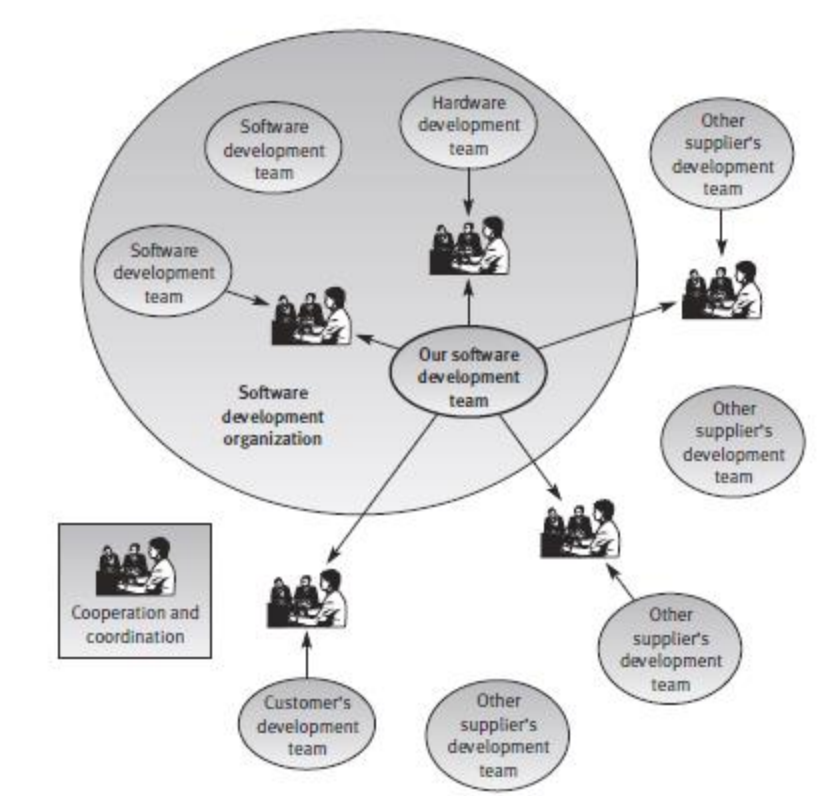


Figure 1.1: A cooperation and coordination scheme for a software development team of a large-scale project

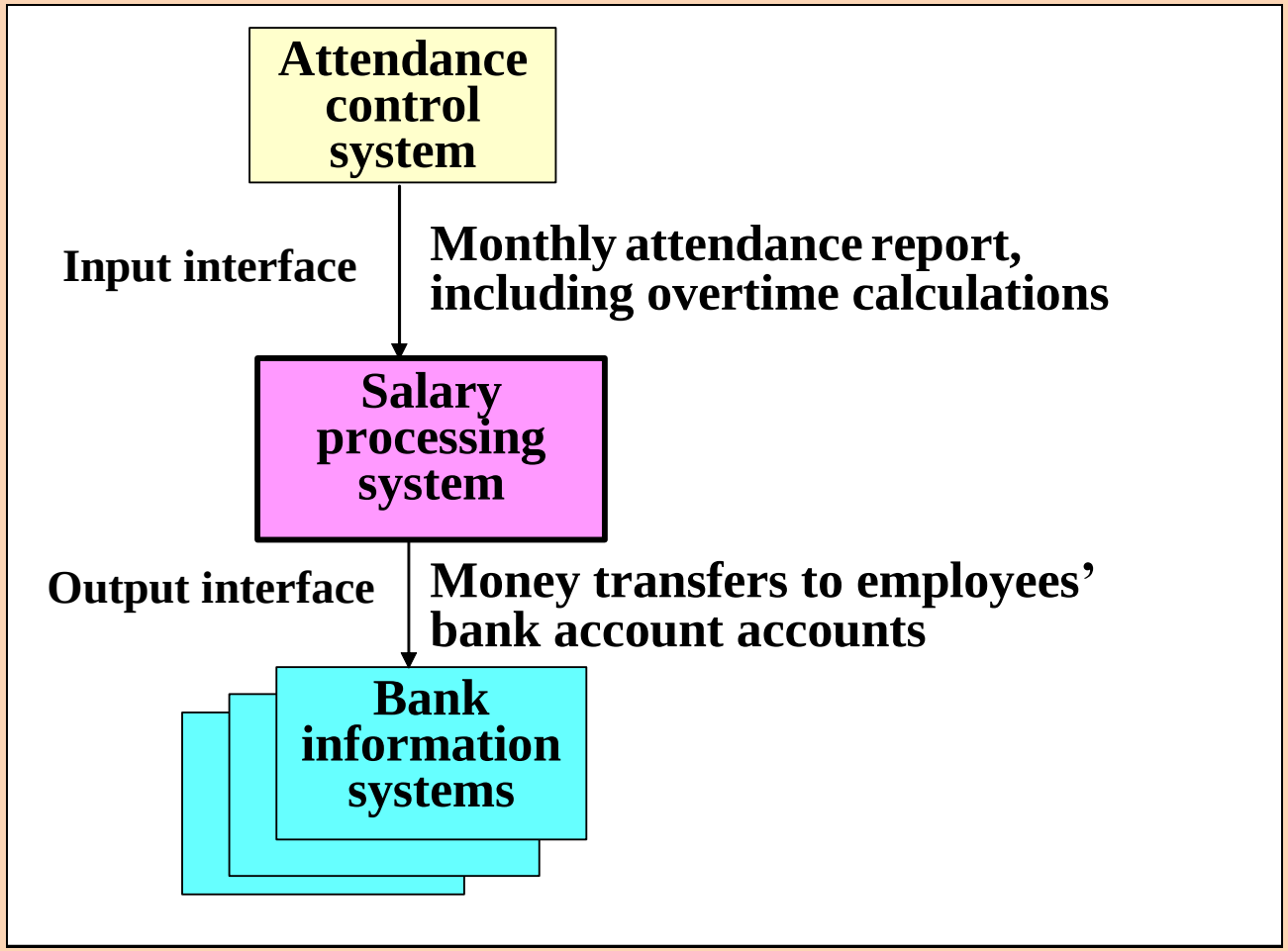
The Environment for which SQA Methods are Developed

5. Interfaces with Other Systems
   - These interfaces allow data in electronic form to flow between the software systems, free from keying in of data processed by the other software systems. One can identify the following main types of interfaces:
     - Input interfaces, where other software systems transmit data to your software system.
     - Output interfaces, where your software system transmits processed data to other software systems.
     - Input and output interfaces to the machine's control board, as in medical and laboratory control systems, metal processing equipment, etc.

The Environment for which SQA Methods are Developed

5. Interfaces with Other Systems
   - A Course Registration System – may well interface with a Class Scheduling System and perhaps a Billing System.
   - Often outputs from one system are inputs to another and vice versa!
   - A course registration system may need to 'interface' with an existing Billing system with different file / database formats, and more.
   - Sometimes outputs from one system are inputs to several others
   - Or, outputs from some system update master files / databases that are processed by other systems.

The Environment for which SQA Methods are Developed



The Environment for which SQA Methods are Developed

6. The Need to continue Carrying out a Project despite Team Member Changes
   - Very commonplace!
   - “The show must go on”
   - Fred Brooks: Adding people to a late project makes it later.
   - Fred Brooks: Communications with new members
   - Fred Brooks: getting new people up to speed.
   - Team members leave, are hired, fired, take unexpected vacations, transferred within the company, and more.
   - No matter how much effort is invested in training the new team member, which means that the original project contract timetable will not change.

The Environment for which SQA Methods are Developed

7. The Need to Continue Carrying out Software Maintenance for an Extended Period
   - Software is developed to run for years.
   - Need for Maintenance arises with time and this is where software development corporations make their money.
   - Remember, when developing software, company is in the 'red.'
   - Takes some time after deployment for transition into the black and make revenue.
   - In most cases, the developer is required to supply these services directly.
   - Internal "customers", in cases where the software has been developed in-house, share the same expectation regarding the software maintenance during the service period of the software system.

Internal Software Development

- Not terribly different from external clients.
- But in-house development normally avoids formal contracts and/or formal customer/supplier relationships.
- Much in-house development or upgrading of in-house software is typical.
- The relationships between ‘internal’ customers and development varies greatly ‘when measured by a formal-informal scale.’
- Some managers claim that the closer the relationships to the formal form, the greater the probability for project success.