INTRODUCTION TO SOFTWARE ENGINEERING

SOFTWARE ENGINEERING 2 | CS6300

JOHN PAUL M. MANUEL | ACLC COLLEGE OF TAYTAY

ACLC COLLEGE OF TAYTAY
(A Member of AMA Education System)

VERSION 01-250119-2425

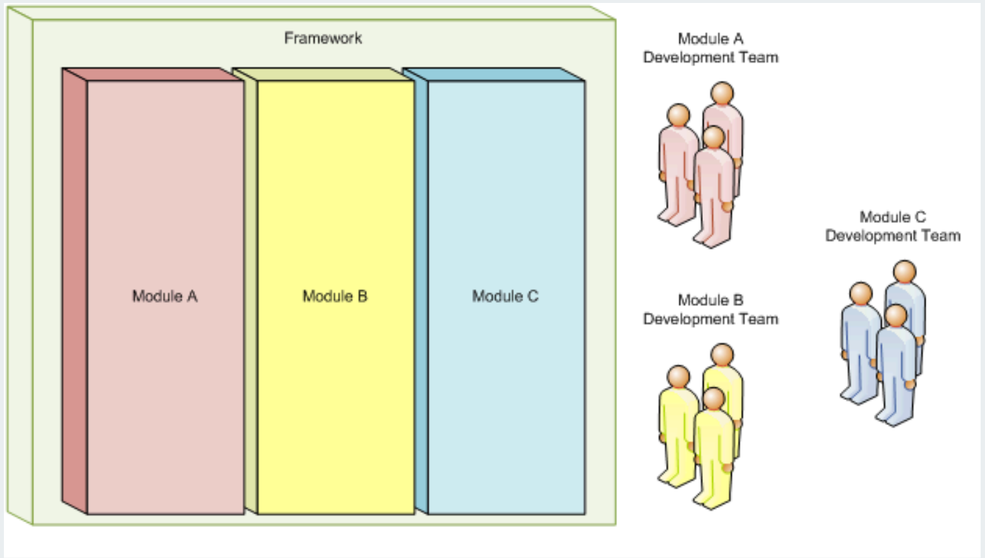
WHAT IS SOFTWARE ENGINEERING?

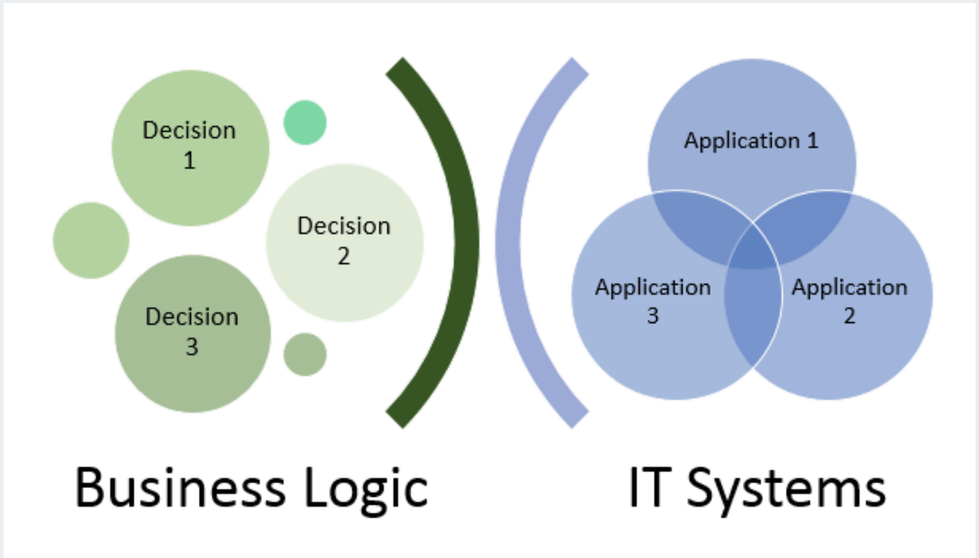
Software engineering is concerned with all aspects of software production, from the early stages of system specification to maintaining the system after it has been used. It involves the systematic application of scientific knowledge to create efficient and reliable software systems.

PRINCIPLES OF SOFTWARE ENGINEERING:

SEPARATION OF CONCERNS

This principle involves dividing different concerns of a system, such as basic functionality and data integrity, into separate components, making the software easier to use and maintain.

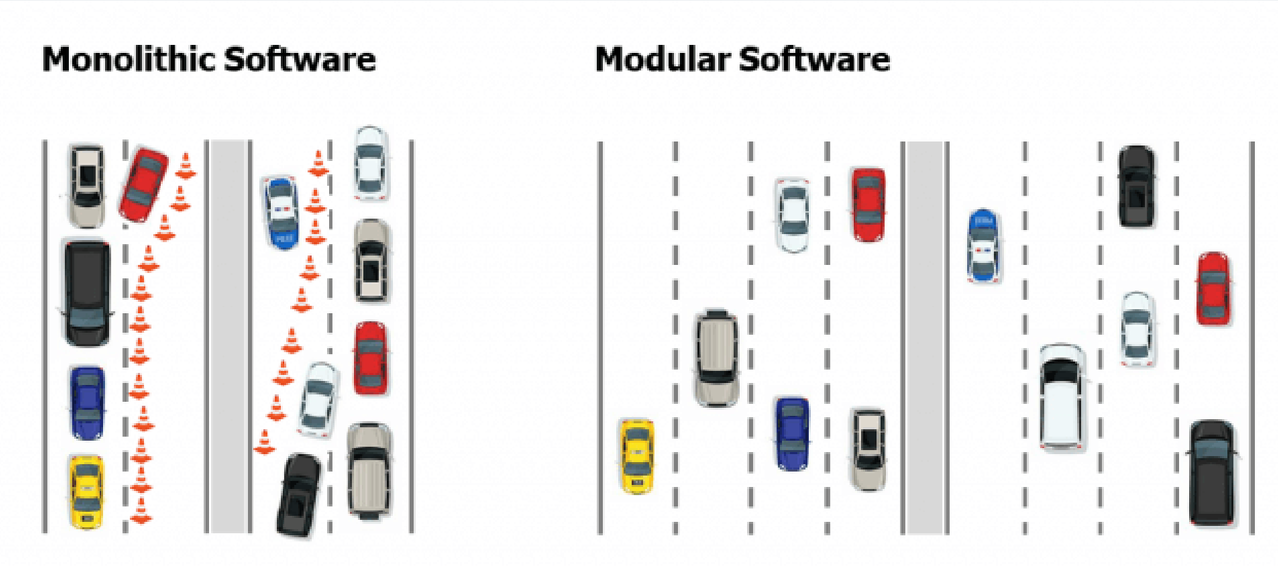




PRINCIPLES OF SOFTWARE ENGINEERING:

MODULARITY

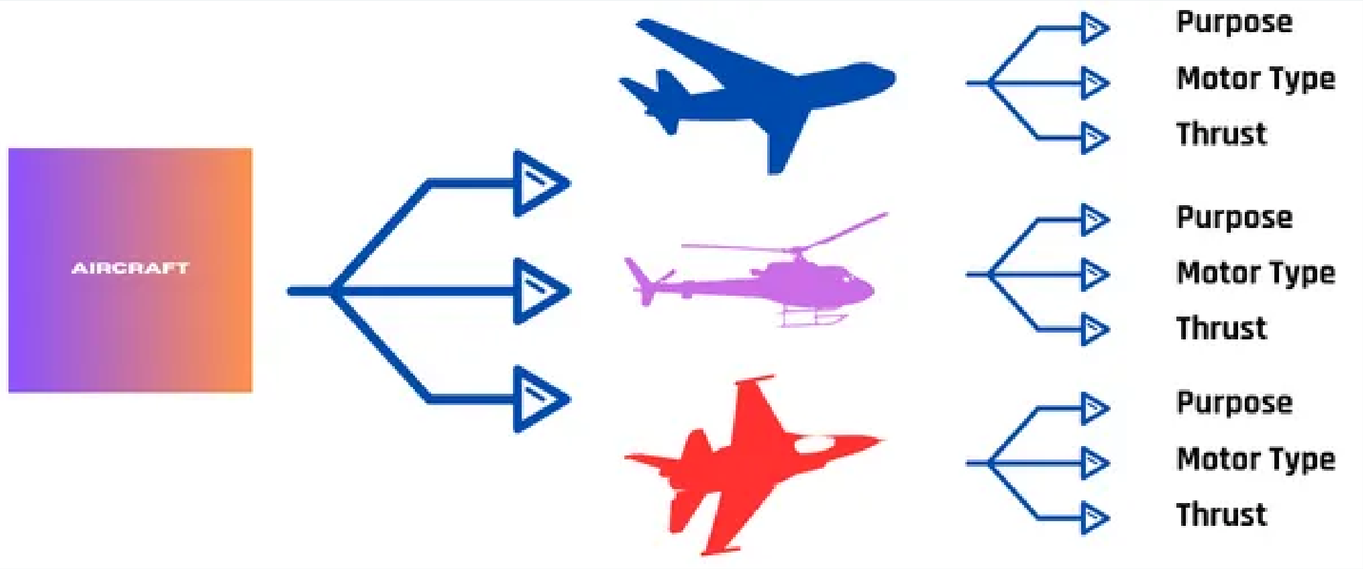
A specialization of separation of concerns, modularity involves separating software into components based on functionality and responsibility, enabling easier development and maintenance.



PRINCIPLES OF SOFTWARE ENGINEERING:

ABSTRACTION

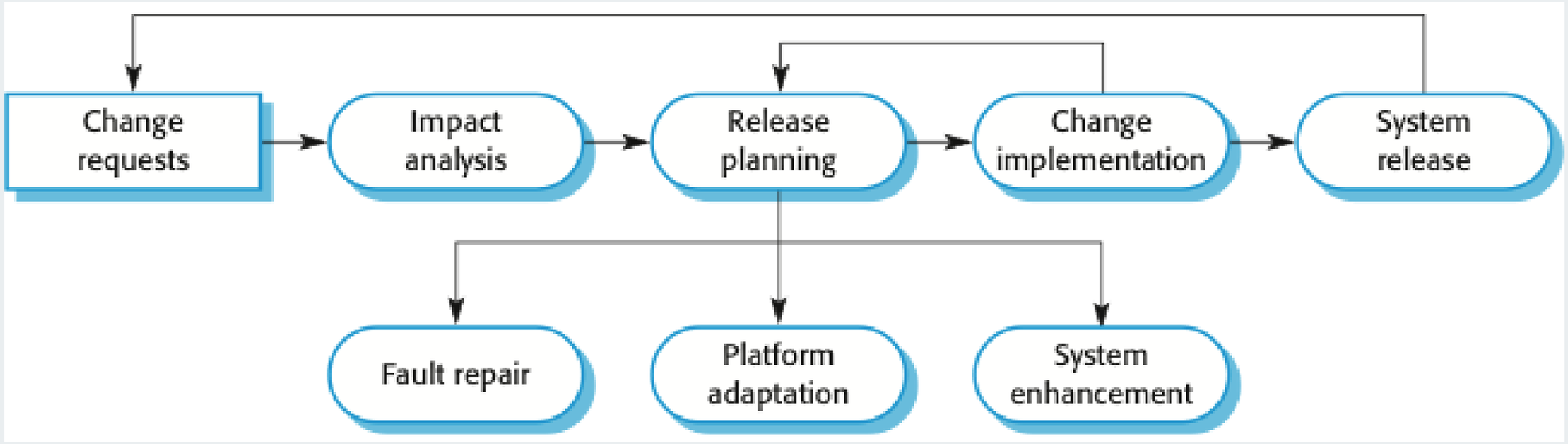
This principle focuses on separating the behavior of software components from their implementation, helping developers understand both what the software does and how it does it.



PRINCIPLES OF SOFTWARE ENGINEERING:

ANTICIPATION OF CHANGE

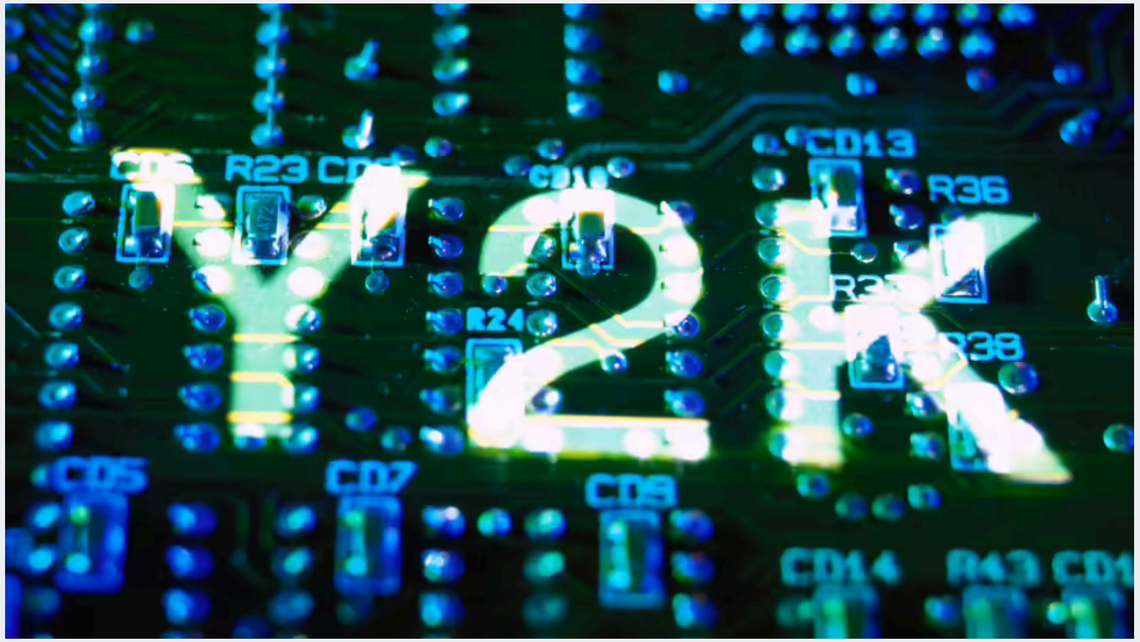
Emphasizes designing software with flexibility to adapt to changing requirements, reducing the need for significant modifications in the future.



PRINCIPLES OF SOFTWARE ENGINEERING:

GENERALITY

A principle related to anticipating change, it focuses on designing software free from unnecessary restrictions or limitations to improve adaptability and longevity.

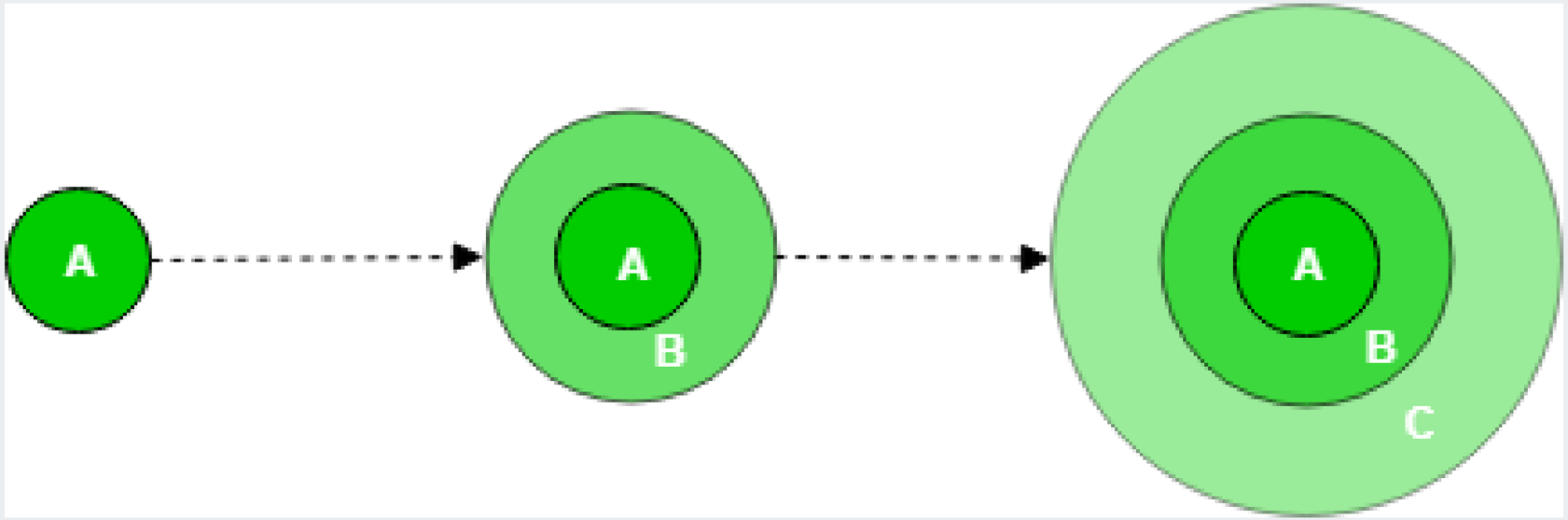


The Y2K Problem was a computer bug caused by systems using two-digit years (e.g., "99" for 1999). When the year changed to 2000, these systems risked interpreting "00" as 1900, potentially causing errors in calculations and operations across industries like banking, utilities, and infrastructure.

PRINCIPLES OF SOFTWARE ENGINEERING:

INCREMENTAL DEVELOPMENT

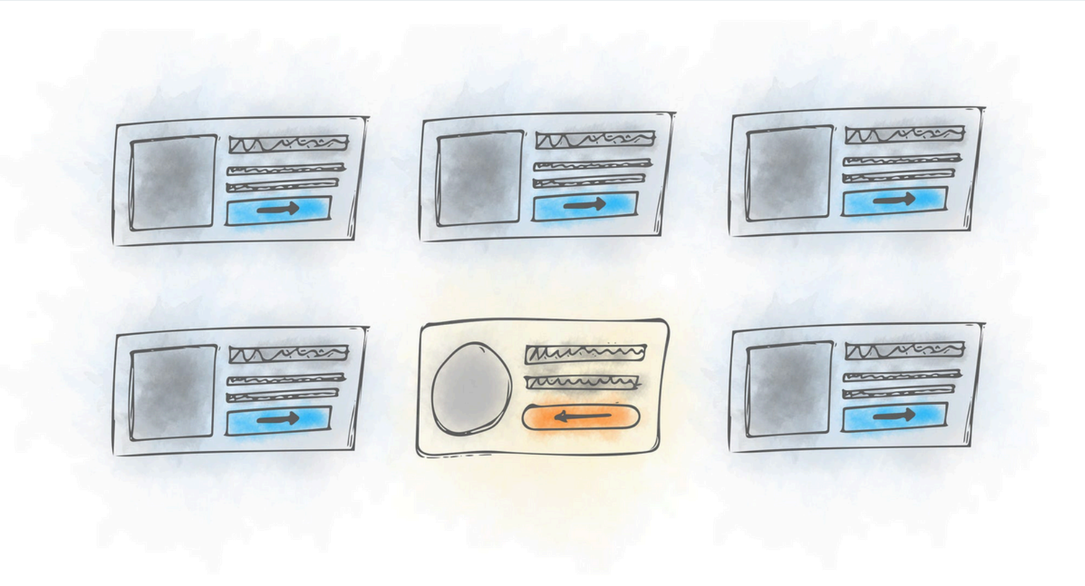
This involves developing software in small, manageable increments, simplifying verification and making it easier to accommodate changes in requirements.



PRINCIPLES OF SOFTWARE ENGINEERING:

CONSISTENCY

Consistency ensures that familiar patterns are used in software design and implementation, making systems easier to use and maintain.



IS SOFTWARE DEVELOPMENT AN ENGINEERING DISCIPLINE?

Software development is often debated as an engineering discipline. It shares characteristics with engineering fields, such as:

- **Systematic Approach:** Software development applies structured methodologies and principles, similar to engineering disciplines, to solve complex problems.
- **Focus on Design and Functionality:** It emphasizes creating efficient, reliable, and maintainable systems.
- **Practical Applications of Theory:** Software engineers use theoretical concepts (e.g., algorithms, data structures) to develop practical solutions.
- **Challenges:** Unlike traditional engineering, software lacks physical constraints (like materials) and involves rapidly evolving technologies, making it unique.

In conclusion, while software development borrows engineering principles, its reliance on creativity and adaptability sets it apart from traditional engineering disciplines.

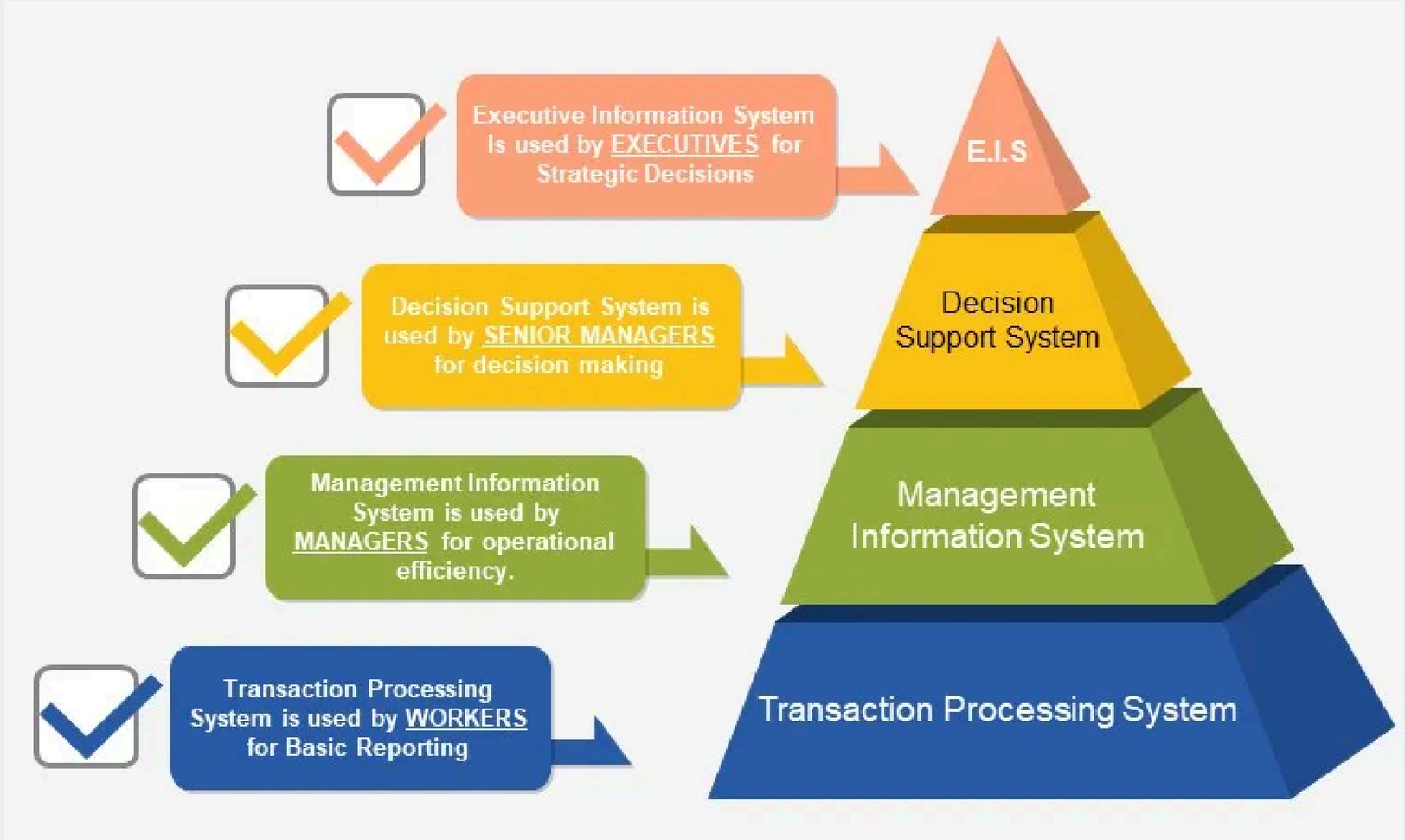
INFORMATION SYSTEMS

Information systems are organized sets of resources and processes designed to manage data and support decision-making in an organization.

Components:

1. **Hardware**: Physical devices like servers, computers, and network devices.
2. **Software**: Programs and applications that process data.
3. **Data**: Raw facts and figures transformed into meaningful information.
4. **Procedures**: Processes and rules for operating the system.
5. **People**: Users and administrators who interact with the system.
6. **Feedback**: Information used to improve system performance.

TYPES OF INFORMATION SYSTEMS



TYPES OF INFORMATION SYSTEMS

Executive Information System (EIS) (Top Layer):

- Used by: Executives.
- Purpose: Supports strategic decision-making by providing high-level summaries and critical information.
- Focus: Long-term goals and organizational strategy.

Decision Support System (DSS) (Third Layer):

- Used by: Senior Managers.
- Purpose: Assists in complex decision-making by analyzing data and presenting actionable insights.
- Focus: Helps in planning and problem-solving.

TYPES OF INFORMATION SYSTEMS

Management Information System (MIS) (Second Layer):

- Used by: Managers.
- Purpose: Provides summarized and structured reports for operational efficiency.
- Focus: Middle management decision-making and monitoring performance.

Transaction Processing System (TPS) (Base Layer):

- Used by: Workers.
- Purpose: Handles basic and routine transactions like payroll, order processing, and billing.
- Focus: Basic reporting and data collection.