GROUP FOUR.

Topics to be covered;

Overview of System Analysis.
Attributes of structured Analysis.
Tools and Techniques of System Analysis and Design.

Overview of System Analysis

Definition of System Analysis
Importance of System Analysis
Goals of System Analysis
System Analysis in the Software Development Lifecycle
Stages of System Analysis
Methods of Data Gathering

Attributes of Structured Analysis

Data-centric Approach
Modularity and Hierarchy
Clear Documentation
Use of Standard Notation
Reusability

Tools and Techniques of SAAD

Data Flow Diagrams (DFDs)
Entity-Relationship Diagrams(ERDs)
Structured English
Structured Query Language(SQL)
Unified Modelling Language(UML)
Normalization
Prototyping
Computer-Aided Software Engineering (CASE) Tools

System Analysis is a crucial phase in the software development lifecycle (SDLC) that focuses on understanding and defining the requirements of a system. It involves a systematic and methodical approach to studying existing systems, identifying problems and opportunities, and proposing solutions. The primary goal of system analysis is to ensure that the resulting system, whether it's a software application or an entire information system, meets the needs of its users, is cost-effective, and can be implemented successfully.

Key Aspects of System Analysis:

1. **Understanding User Needs:** System analysts work closely with stakeholders, including end-users, to gather and understand their requirements. This involves conducting interviews, surveys, and observations to capture the needs and expectations of the system's users.

2. **Problem Identification:** System analysts identify issues and inefficiencies within the current system or business processes. They assess how these issues impact the organization's goals and productivity.
3. **Solution Proposal:** After identifying problems and opportunities, system analysts propose solutions. These solutions may involve the development of new software, modifications to existing systems, or changes in business processes.
4. **Requirements Specification:** System analysts document system requirements in a structured and detailed manner. These requirements include both functional requirements (what the system should do) and non-functional requirements (how it should perform).
5. **Data Modeling:** In system analysis, data is a central focus. Data modeling techniques, such as Entity-Relationship Diagrams (ERDs) and data flow diagrams (DFDs), are used to represent how data is stored, processed, and flows within the system.
6. **Process Modeling:** Process modeling involves creating diagrams to illustrate the flow of activities and tasks within the system. Flowcharts and process flow diagrams are commonly used in this context.
7. **Validation and Verification:** System analysts validate and verify requirements by ensuring they are complete, consistent, and aligned with the organization's objectives. This process may involve reviews and validations with stakeholders.
8. **System Design:** While system analysis primarily focuses on defining requirements, it often provides the foundation for system design. It outlines what the system should do, and designers then determine how to build it.
9. **User Feedback:** Throughout the analysis phase, user feedback and validation are essential. System analysts often create prototypes or mock-ups to gather early feedback from end-users.

The Importance of System Analysis:

System analysis plays a pivotal role in the success of software development projects and the efficient operation of information systems within organizations. It:

- Ensures that the final system meets user needs and expectations.
- Identifies and addresses potential issues early in the development process, saving time and resources.
- Provides a clear roadmap for system development and design.
- Supports informed decision-making by stakeholders and project teams.
- Helps align technology solutions with organizational goals and strategies.

Structured analysis is a systematic approach to system analysis that uses graphical tools and techniques to model the system and its components. It is a top-down approach, meaning that it starts with a high-level view of the system and gradually drills down into more detail.

The key attributes of structured analysis are:

- **Modularity:** The system is decomposed into smaller, more manageable modules.

- **Hierarchy:** The modules are organized into a hierarchical structure.
- **Data-driven:** The system is modeled around the data that it processes.
- **Process-driven:** The system is modeled around the processes that it performs.
- **Graphical:** The system is modeled using graphical tools such as data flow diagrams and entity-relationship diagrams.

Modularity

Modularity is a key concept in structured analysis. It involves breaking down the system into smaller, more manageable modules. This makes it easier to understand and design the system.

Hierarchy

The modules in a structured analysis are organized into a hierarchical structure. This means that some modules are higher-level and some are lower-level. High-level modules typically encapsulate the overall functionality of the system, while lower-level modules provide more detailed functionality.

Data-driven

Structured analysis is data-driven, meaning that the system is modeled around the data that it processes. This ensures that the system is designed to meet the needs of its users and that the data is processed in a way that is efficient and effective.

Process-driven

Structured analysis is also process-driven, meaning that the system is modeled around the processes that it performs. This helps to ensure that the system is efficient and effective.

Graphical

Structured analysis uses graphical tools to model the system and its components. This makes it easier to understand and communicate the design of the system.

Benefits of Structured Analysis

Structured analysis offers a number of benefits, including:

- Improved understanding of the system: Structured analysis helps to improve the understanding of the system by providing a clear and concise way to model the system and its components.
- Improved design of the system: Structured analysis helps to improve the design of the system by identifying and addressing potential problems early in the development process.
- Improved communication: Structured analysis uses graphical tools to model the system, which makes it easier to communicate the design of the system to stakeholders.

- Reduced risk: Structured analysis helps to reduce the risk of project failure by identifying and addressing potential problems early in the development process.

The goals of system analysis are to:

- **Understand the current system.** This includes identifying the system's inputs, outputs, processes, and users. It also involves identifying the system's strengths and weaknesses.
- **Identify opportunities for improvement.** This may involve developing a new system, improving the existing system, or discontinuing the system altogether.
- **Develop recommendations for improvement.** These recommendations should be based on the findings of the system analysis and should be tailored to the needs of the organization.
- **Communicate the findings and recommendations to stakeholders.** This may involve writing a report, giving a presentation, or creating a prototype.

Specific goals of system analysis may include:

- Improving the efficiency and effectiveness of the system.
- Reducing the costs of the system.
- Improving the quality of the system's outputs.
- Responding to changes in the environment.
- Making better decisions about how to use the system.

System analysis is an important stage in the software development lifecycle (SDLC). It is the process of understanding the current system, identifying opportunities for improvement, and developing recommendations for a new or improved system.

The stages of system analysis are:

1. **Define the scope of the project.** What is the purpose of the system analysis? What are the specific problems or opportunities that the system is intended to address?
2. **Gather information about the current system.** This can be done through interviews, surveys, and observation.
3. **Analyze the current system.** This involves identifying the system's inputs, outputs, processes, and users. It also involves identifying the system's strengths and weaknesses.
4. **Develop recommendations for improvement.** This may involve developing a new system, improving the existing system, or discontinuing the system altogether.
5. **Communicate the findings and recommendations to stakeholders.** This may involve writing a report, giving a presentation, or creating a prototype.

Methods of Data Gathering

There are a number of different methods that can be used to gather data for system analysis. The most common methods include:

- **Interviews:** Interviews are a good way to gather information about the current system and the needs of the users.
- **Surveys:** Surveys can be used to gather information from a large number of people.
- **Observation:** Observation can be used to gather information about how the current system is used and to identify any potential problems.
- **Document review:** Document review can be used to gather information about the current system and its requirements.

The specific methods that are used will depend on the scope of the project and the type of information that is needed.

Detailed Explanation

System analysis is an important stage in the SDLC because it helps to ensure that the new or improved system will meet the needs of the users and the organization. By understanding the current system and identifying opportunities for improvement, system analysts can develop recommendations that will help the organization to achieve its goals.

The data gathering stage of system analysis is particularly important. By gathering comprehensive and accurate data, system analysts can develop a deep understanding of the current system and the needs of the users. This information is essential for developing effective recommendations for improvement.

There are a number of different methods that can be used to gather data for system analysis. The most common methods include interviews, surveys, observation, and document review. The specific methods that are used will depend on the scope of the project and the type of information that is needed.

Once the data has been gathered, system analysts will analyze it to identify opportunities for improvement. This may involve identifying areas where the current system is inefficient, ineffective, or costly. It may also involve identifying new features or functionality that would be beneficial to the users or the organization.

Once the system analysts have identified opportunities for improvement, they will develop recommendations for a new or improved system. These recommendations should be based on the findings of the system analysis and should be tailored to the needs of the organization.

The final stage of system analysis is to communicate the findings and recommendations to stakeholders. This may involve writing a report, giving a presentation, or creating a prototype. It is important to communicate the findings and recommendations in a clear and concise way so that stakeholders can understand the benefits of the new or improved system and make a decision about whether to proceed with development.

A data-centric approach to structured analysis is one that focuses on the data that the system processes and the relationships between that data. This approach is often used in the design of database systems, but it can also be used in the design of other types of systems, such as business process management systems and enterprise resource planning systems.

One of the key benefits of a data-centric approach to structured analysis is that it helps to ensure that the system is designed to meet the needs of its users. By focusing on the data that the system processes, system analysts can identify the information that users need and how they need to interact with that information.

Another benefit of a data-centric approach is that it can help to improve the efficiency and effectiveness of the system. By organizing the data in a logical way, system analysts can reduce the amount of data that needs to be stored and processed. This can lead to improved performance and reduced costs.

Here are some of the key characteristics of a data-centric approach to structured analysis:

- **The system is designed around the data.** This means that the system is organized in a way that makes it easy to store, process, and retrieve the data that the system needs.
- **The data is normalized.** This means that the data is organized in a way that reduces redundancy and improves accuracy.
- **The data is integrated.** This means that the data is stored in a single place and that different parts of the system can easily access the data.
- **The data is secure.** This means that the data is protected from unauthorized access, modification, or destruction.

Examples of data-centric structured analysis techniques:

- **Entity-relationship diagrams (ERDs):** ERDs are used to model the entities in a system and the relationships between those entities.
- **Data flow diagrams (DFDs):** DFDs are used to model the flow of data through a system.
- **Decision tables:** Decision tables are used to model the decision-making logic in a system.
- **State transition diagrams:** State transition diagrams are used to model the different states that a system can be in and the transitions between those states.

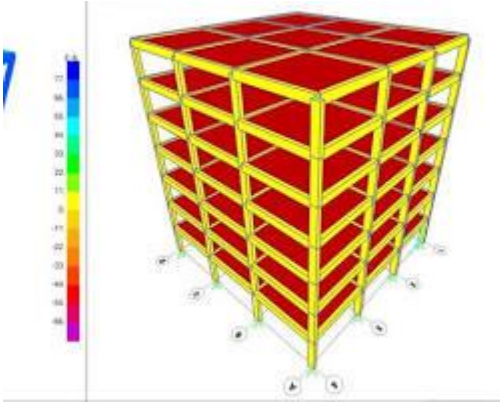
Benefits of a data-centric approach to structured analysis:

- **Improved understanding of the system:** A data-centric approach helps to improve the understanding of the system by providing a clear and concise way to model the system's data and its relationships.
- **Improved design of the system:** A data-centric approach helps to improve the design of the system by identifying and addressing potential problems early in the development process.
- **Improved communication:** A data-centric approach uses graphical tools to model the system's data and relationships, which makes it easier to communicate the design of the system to stakeholders.

- **Reduced risk:** A data-centric approach helps to reduce the risk of project failure by identifying and addressing potential problems early in the development process.

Modularity is the process of breaking down a system into smaller, more manageable modules. This makes it easier to understand, design, and implement the system. Each module should be self-contained and perform a specific function. Modules should also be loosely coupled, meaning that they should not be dependent on each other.

Hierarchy is the organization of modules into a hierarchical structure. This means that some modules are higher-level and some are lower-level. High-level modules typically encapsulate the overall functionality of the system, while lower-level modules provide more detailed functionality.


Opens in a new window prefabex.com

Modularity in structured analysis

Modularity and hierarchy are often used together to design complex systems. By breaking down the system into smaller, more manageable modules and organizing those modules into a hierarchy, system analysts can create more efficient and effective systems.

Benefits of modularity and hierarchy:

- Improved understanding of the system: Modularity and hierarchy help to improve the understanding of the system by providing a clear and concise way to model the system's components and their relationships.
- Improved design of the system: Modularity and hierarchy help to improve the design of the system by identifying and addressing potential problems early in the development process.
- Improved communication: Modularity and hierarchy use graphical tools to model the system's components and relationships, which makes it easier to communicate the design of the system to stakeholders.
- Reduced risk: Modularity and hierarchy help to reduce the risk of project failure by identifying and addressing potential problems early in the development process.

Examples of modularity and hierarchy:

- A car can be divided into modules such as the engine, transmission, chassis, and body. These modules can then be further divided into smaller modules, such as the pistons and cylinders in the engine.
- A computer can be divided into modules such as the central processing unit (CPU), memory, and storage. These modules can then be further divided into smaller modules, such as the transistors in the CPU.

Clear Documentation

Clear documentation is essential for any system development project. It helps to ensure that the system is properly understood, designed, implemented, and maintained.

Clear documentation should be:

- **Accurate:** The documentation should be accurate and up-to-date.
- **Complete:** The documentation should be complete and should include all of the information that is needed to understand, use, and maintain the system.
- **Concise:** The documentation should be concise and easy to understand.
- **Well-organized:** The documentation should be well-organized and easy to navigate.

Clear documentation can be achieved by following a number of best practices, such as:

- Using plain language that is easy to understand.
- Using diagrams and screenshots to illustrate complex concepts.
- Organizing the documentation into logical sections and subsections.
- Using a consistent style and format throughout the documentation.
- Having the documentation reviewed by others to ensure that it is accurate and complete.

Use of Standard Notation

The use of standard notation is important for system documentation because it makes the documentation more readable and understandable. Standard notation is a set of symbols and conventions that are used to represent different system components, such as data flows, processes, and entities.

There are a number of different standard notations that can be used for system documentation, such as:

- **Data flow diagrams (DFDs):** DFDs are used to model the flow of data through a system.
- **Entity-relationship diagrams (ERDs):** ERDs are used to model the entities in a system and the relationships between those entities.
- **Decision tables:** Decision tables are used to model the decision-making logic in a system.

- **State transition diagrams:** State transition diagrams are used to model the different states that a system can be in and the transitions between those states.

The use of standard notation can help to improve the quality of system documentation and make it more useful for stakeholders.

Reusability

Reusability is the ability to use existing system components in the development of new systems. This can help to save time and money, and it can also help to improve the quality of new systems.

There are a number of different ways to achieve reusability in system development. One way is to use standard components. Standard components are components that have been developed by third-party vendors or by the organization itself. Standard components can be used in a variety of different systems and can help to reduce the amount of time and effort that is needed to develop new systems.

Another way to achieve reusability is to use design patterns. Design patterns are reusable solutions to common software design problems. Design patterns can be used to improve the design and quality of new systems.

Reusability is an important concept in system development because it can help organizations to save time and money, and it can also help to improve the quality of new systems.

1. **Data Flow Diagrams (DFDs):**
   - DFDs are graphical representations that show how data moves through a system. They consist of processes, data stores, data flows, and external entities. DFDs help visualize the flow of data within a system and are used for requirements analysis and design.
2. **Entity-Relationship Diagrams (ERDs):**
   - ERDs are used to model data structures and relationships in a database system. They illustrate entities (such as tables in a database), attributes, and the relationships between entities. ERDs are vital for database design and data modeling.
3. **Structured English:**
   - Structured English is a technique for representing system logic in plain, structured English sentences. It is used to express the logic of processes and algorithms in a way that is easy to understand and can serve as a basis for coding.
4. **Structured Query Language (SQL):**
   - SQL is a specialized language for defining, manipulating, and querying relational databases. It is used to create, modify, and query databases, making it a fundamental tool for data management.
5. **Unified Modeling Language (UML):**
   - UML is a standardized modeling language used in SAAD to represent and visualize object-oriented systems. UML includes various diagrams, such as use

case diagrams, class diagrams, sequence diagrams, and more, to describe system structure and behavior.

6. **Normalization:**
    - Normalization is a process used in database design to organize data efficiently, minimize data redundancy, and ensure data integrity. It involves dividing a database into two or more tables and defining relationships between them.
7. **Prototyping:**
    - Prototyping involves creating a working model of the system to gather user feedback and refine system requirements. It provides a tangible representation of the system's functionality, which can be adjusted based on user input.
8. **Computer-Aided Software Engineering (CASE) Tools:**
    - CASE tools are software applications that provide support for various aspects of SAAD. They assist in modeling, design, documentation, and project management. Examples include tools for drawing diagrams, generating code, and managing project schedules.