Latar Belakang
System Development Life Cycle
Perkembangan System Development
Peran Life Cycle Management dalam perusahaan
Bayang sistem yang dibatalkan karena analis-analis mencoba bembangung sistem yang “luar biasa” tanpa pemahaman bagaimana sistem itu akan cocok dengan sasaran organisasi

- Standish Group Survey (1996): 42% corporate IS project in US were abandoned
- General Accounting Office (1996) found 53% US government project were abandoned

Target utama dari sistem informasi adalah menciptakan nilai bermanfaat untuk organisasi.

Systems analyst adalah seseorang yang berperan menganalisa bisnis, mengidentifikasi peluang untuk improvement, dan mendisain sistem informasi untuk menerapkan ide-ide yang telah didisain.

Sangat penting untuk memahami dan mengembangkan melalui keterampilan yang diperlukan agar berhasil dalam mendisain dan implementasi sistem informasi yang baru.
SDLC adalah proses tentang pemahaman bagaimana satu sistem informasi (SI) dapat mendukung kebutuhan bisnis, merancang sistem, membangun nya, dan mengirimkan ya (deliver) kepada para pemakai.
**Project Phases**

- **Planning**
  - Why build the system?
- **Analysis**
  - Who, what, when, where will the system be?
- **Design**
  - How will the system work?
- **Implementation**
  - System delivery

**Planning**

- **Project Initiation**
  - Identifying business value (how will the IS lower costs or increase revenue?)
  - Analyze feasibility (technical, economic and organizational)
- **Project Management**
  - Develop work plan
  - Staff the project
  - Control and direct project
## Analysis

- **Analysis Strategy**
  - Analysis current system (as-is system) and new system (to-be system)
- **Requirement gathering**
  - Interview or questionnaires or other method
  - Analysis Model (Process and Data)
- **System Proposal**
  - Describe what business requirements of the new system should met.

## Design

- **Design Strategy**
  - Build it, outsource or buy?
- **Architectural & Interface Design**
  - Describe h/w, s/w, network infrastructure
  - How the users will move through the system
- **Database and file design**
- **Program design**
**Implementation**

- **System Construction**: The system is built and tested to make sure it performs as designed.
- **Installation**: Prepare to support the installed system.
- **Support Plan**: Includes a post-implementation review.

**Processes and Deliverables**

<table>
<thead>
<tr>
<th>Process</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning</td>
<td>Project Plan</td>
</tr>
<tr>
<td>Analysis</td>
<td>System Proposal</td>
</tr>
<tr>
<td>Design</td>
<td>System Specification</td>
</tr>
<tr>
<td>Implementation</td>
<td>New System and Maintenance Plan</td>
</tr>
</tbody>
</table>
What Is a Methodology?

- A formalized approach or series of steps
- A methodology is a formalized approach to implementing the SDLC.
- The methodology will vary depending on whether the emphasis is on businesses processes or on the data that supports the business.

*Writing code without a well-thought-out system request may work for small programs, but rarely works for large ones.*
## Development Methodology Category

- Structured Design
  - Waterfall
  - Parallel
- Rapid Application Development (RAD)
  - Phased Development
  - Prototyping
  - Throw-Away Prototyping
- Agile Development
  - Extreme Programming
  - Scrum

## Structured Design

- Projects move methodically from one to the next step
- Generally, a step is finished before the next one begins
- This design methodology introduces the use of formal modeling or diagramming techniques to describe a system’s basic business processes and follows a basic approach of two structured design categories.
With waterfall development-based methodologies, the analysts and users proceed sequentially from one phase to the next.

Advantages:
- The system requirements are identified long before programming begins.
- Changes to the requirements are minimized as the project proceeds.

Disadvantages:
- The design must be completely specified before programming begins.
- A long time elapses between the completion of the system proposal in the analysis phase and the delivery of the system.
- If team miss important requirement, expensive post-implementation programming may be needed.
Parallel Development

- This methodology attempts to address the long time interval between the analysis phase and the delivery of the system.
- A general design for the entire system is performed and then the project is divided into a series of distinct subprojects.
**Rapid Application Development**

- RAD-based methodologies adjust the SDLC phases to get some part of system developed quickly and into the hands of the users.
- Most RAD-based methodologies recommend that analysts use special techniques and computer tools to speed up the analysis, design, and implementation phases, such as CASE (computer-aided software engineering) tools.
  - CASE tools
  - JAD sessions
  - Fourth generation/visualization programming languages
  - Code generators

**RAD**

- One possible subtle problem with RAD-based methodologies is managing user expectations.
- RAD Categories:
  - Phased development
    - A series of versions
  - Prototyping
    - System prototyping
  - Throw-away prototyping
    - Design prototyping
Phased Development

- This methodology breaks the overall system into a series of versions that are developed sequentially.
- The team categorizes the requirements into a series of versions, then the most important and fundamental requirements are bundled into the first version of the system.
- The analysis phase then leads into design and implementation; however, only with the set of requirements identified for version 1.
- As each version is completed, the team begins work on a new version.
Prototyping

- Prototyping-based methodologies perform the analysis, design and implementation phases concurrently.
- All three phases are performed repeatedly in a cycle until the system is completed.
- A prototype is a smaller version of the system with a minimal amount of features.
### Prototyping

- **Advantage:** Provides a system for the users to interact with, even if it is not initially ready for use.
- **Disadvantage:** Often the prototype undergoes such significant changes that many initial design decisions prove to be poor ones.

### Throwaway Prototyping

- Throwaway prototyping methodologies are similar to prototyping based methodologies.
- The main difference is that throwaway prototyping IS completed during a different point in the SDLC.
- Has relatively thorough analysis phase.
This category focuses on streamlining the SDLC by eliminating much of the modeling and documentation overhead and the time spent on those tasks.

- Projects emphasize simple, iterative application development.
- This category uses extreme programming, which is described next.
Extreme Programming (XP) was founded on four core values:
- Communication
- Simplicity
- Feedback
- Courage

Key principles of XP include:
- Continuous testing
- Simple coding
- Close interaction with the end users to build systems very quickly
Selecting a Methodology

- Selecting a methodology is not simple, as no one methodology is always best.
- Many organizations have their own standards.
- The next figure summarizes some important methodology selection criteria.
Clarity of User Requirements

- RAD methodologies of prototyping and throwaway prototyping are usually more appropriate when user requirements are unclear as they provide prototypes for users to interact with early in the SDLC.

Familiarity with Technology

- If the system is designed without some familiarity with the base technology, risks increase because the tools may not be capable of doing what is needed.
**System Complexity**

- Complex systems require careful and detailed analysis and design.
- Project teams who follow phased development-based methodologies tend to devote less attention to the analysis of the complete problem domain than they might if they were using other methodologies.

**System Reliability**

- System reliability is usually an important factor in system development.
- Throwaway prototyping-based methodologies are most appropriate when system reliability is a high priority.
- Prototyping-based methodologies are generally not a good choice as they lack careful analysis and design phases.
**Short Time Schedules**

- RAD-based methodologies are well suited for projects with short time schedules as they increase speed.
- Waterfall-based methodologies are the worst choice when time is essential as they do not allow for easy schedule changes.

**Schedule Visibility**

- RAD-based methodologies move many of the critical design decisions earlier in the project; consequently, this helps project managers recognize and address risk factors and keep expectations high.
Peran System Life Cycle Management dalam Perusahaan

Why We Need Life Cycle Management?

- IT Governance
  - SDLC
  - NDLC
  - People, Process, Technology
- IT Audit
  - Compliance with standard/best practices
Sample IT Operation

Thank You