Software Engineering Practice

Materi 4

- SOFTWARE ENGINEERING PRACTICE
- COMMUNICATION PRACTICES
- Planning practices
- Analysis modeling practices
- Design modeling practices
- Construction practices
- Deployment practices
- Computer-Based System
- System Engineering Hierarchy

(Source: Pressman, R. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 2010)

SOFTWARE ENGINEERING PRACTICE

- Terdiri dari kumpulan konsep, prinsip, metode, dan alat-alat yang engineer software memantau setiap hari
- Melengkapi manajer untuk mengelola proyek perangkat lunak dan software engineer untuk membangun program komputer
- Memberikan diperlukan teknis dan manajemen bagaimana ini dalam mendapatkan pekerjaan yang dilakukan
- Mentransformasi pendekatan terfokus menjadi sesuatu yang lebih terorganisir, lebih efektif, dan lebih mungkin untuk mencapai keberhasilan

THE ESSENCE OF PROBLEM SOLVING

1) Memahami masalah (komunikasi dan analisis)
   - Yang memiliki kepentingan dalam solusi untuk masalah ini?
   - Apa yang tidak diketahui (data, fungsi, perilaku)?
   - Bisakah masalah terkotak?
   - Bisakah masalah diwakili grafis?
2) Rencanakan solusi (perencanaan, pemodelan dan software desain)
   - Pernahkah Anda melihat masalah yang sama seperti ini sebelumnya?
   - Telah masalah yang sama telah dipecahkan dan dapat digunakan kembali solusi?
   - Dapat submasalah didefinisikan dan solusi yang tersedia untuk subproblem?

THE ESSENCE OF PROBLEM SOLVING (CONTINUED)

3) Melaksanakan rencana (pembangunan; kode generasi)
   - Apakah solusinya sesuai dengan rencana? Adalah kode sumber dapat dilacak kembali ke desain?
   - Adalah setiap komponen dari solusi yang benar? Telah desain dan kode diperiksa?
4) Memeriksa hasil untuk akurasi (pengujian dan jaminan kualitas)
   - Apakah mungkin untuk menguji setiap komponen dari solusi?
   - Apakah solusi menghasilkan hasil yang sesuai dengan data, fungsi, dan perilaku yang diperlukan?

SEVEN CORE PRINCIPLES FOR SOFTWARE ENGINEERING

1) Ingat alasan bahwa perangkat lunak yang ada
   - Perangkat lunak ini harus memberikan nilai kepada pengguna dan memenuhi persyaratan
2) Keep it simple, stupid (KISS)
   - Semua desain dan implementasi harus sesederhana mungkin
3) Mempertahankan visi proyek
   - Sebuah visi yang jelas sangat penting untuk keberhasilan proyek
4) Orang lain akan mengkonsumsi apa yang Anda hasilkan
   - Selalu menentukan, merancang, dan mengimplementasikan mengetahui bahwa orang lain nantinya akan harus memahami dan memodifikasi apa yang Anda lakukan
5) Jadilah terbuka untuk masa depan
   - Tidak pernah merancang untuk; membangun perangkat lunak yang dapat dengan mudah diubah dan disesuaikan
6) Rencana ke depan untuk digunakan kembali perangkat lunak
   - Penggunaan kembali perangkat lunak mengurangi biaya jangka panjang dan meningkatkan nilai program dan komponen dapat digunakan kembali
7) Pikirkan, kemudian bertindak
   - Menempatkan jelas, pikiran yang lengkap sebelum tindakan akan hampir selalu menghasilkan hasil yang lebih baik

(Requirements Elicitation)

COMMUNICATION PRACTICES

Communication
Project initiation
Requirements gathering

Planning
Estimating
Scheduling
Tracking

Modeling
Analysis
Design

Construction
Code
Test

Deployment
Delivery
Support
Feedback

COMMUNICATION PRINCIPLES

1) Mendengarkan pembicara dan berkonsentrasi pada apa yang dikatakan
2) Siapkan sebelum Anda bertemu dengan meneliti dan memahami masalah
3) Seseorang harus fasilitas pertemuan dan memiliki agenda
4) Tatap muka komunikasi adalah yang terbaik, tetapi juga memiliki dokumen atau presentasi untuk fokus diskusi
5) Mencatat dan keputusan dokumen
6) Berusaha untuk kolaborasi dan konsensus
7) Tetap fokus pada topik; modularize diskusi
8) Jika sesuatu yang tidak jelas, menggambar
9) Pindah ke topik berikutnya :
    a) Setelah Anda menyetujui sesuatu,
    b) Jika Anda tidak dapat menyetujui sesuatu, atau
    c) Jika fitur atau fungsi tidak jelas dan tidak dapat dijelaskan pada saat ini
10) Negosiasi bukanlah kontes atau permainan; ini bekerja lebih baik ketika kedua belah pihak menang

PLANNING PRACTICES

(DEFINING A ROAD MAP)

Communication
Project initiation
Requirements gathering

Planning
Estimating
Scheduling
Tracking

Modeling
Analysis
Design

Construction
Code
Test

Deployment
Delivery
Support
Feedback

PLANNING PRINCIPLES

1) Memahami ruang lingkupproyek
2) Libatkan pelanggan dalam kegiatan perencanaan
3) Mengakui bahwa perencanaan adalah iteratif; hal akan berubah
4) Memperkirakan hanya berdasarkan apa yang Anda ketahui
5) Pertimbangkan risiko yang Anda menentukan rencana
6) Jadilah realistis tentang berapa banyak yang dapat dilakukan setiap hari oleh setiap orang dan seberapa baik
7) Sesuaikan rincian Anda menentukan rencana
8) Tentukan bagaimana Anda berniat untuk memastikan kualitas
9) Jelaskan bagaimana Anda berniat untuk mengakomodasi perubahan
10) Melacak rencana sering dan membuat penyesuaian yang diperlukan

BARRY BOEHM'S W5HH PRINCIPLE

- **Why** - Mengapa sistem yang dikembangkan?
- **What** - Apa yang akan dilakukan software tersebut nantinya?
- **When** - Kapan akan dicapai?
- **Who** - Siapa yang bertanggung jawab untuk setiap pekerjaan?
- **Where** - Dimanakah peran pihak-pihak diluar tim?
- **How** - Bagaimana pekerjaan dilakukan secara teknis dan manajerial?
- **How Much** - Berapa banyak dari setiap sumber daya yang dibutuhkan?

Jawaban atas pertanyaan-pertanyaan ini mengarah pada definisi kunci karakteristik proyek dan rencana proyek yang dihasilkan

MODELING PRACTICES

(ANALYSIS AND DESIGN)

Communication
Project initiation
Requirements gathering

Planning
Estimating
Scheduling
Tracking

Modeling
Analysis
Design

Construction
Code
Test

Deployment
Delivery
Support
Feedback

ANALYSIS MODELING PRINCIPLES

1) Informasi domain dari masalah (data yang mengalir masuk dan keluar dari sistem) harus diwakili dan dipahami
2) Fungsi yang harus didefinisikan dikerjakan oleh software
3) Perilaku perangkat lunak (sebagai konsekuensi dari peristiwa eksternal) harus direpresentasikan
4) Model yang menggambarkan informasi, fungsi, dan perilaku harus dipartisi dengan cara yang mengungkapkan detail dalam berlapis (atau hirarkis) mode
5) Tugas analisis harus bergerak dari informasi penting menuju implementasi rinci

DESIGN MODELING PRINCIPLES

1) Desain harus dapat dilacak dari model analisis
2) Selalu mempertimbangkan arsitektur perangkat lunak dari sistem yang akan dibangun
3) Desain data adalah sama pentingnya dengan desain fungsi pengolahan
4) Antarmuka (baik internal dan eksternal) harus dirancang dengan hati-hati
5) Desain antarmuka pengguna harus disetel untuk kebutuhan pengguna akhir dan harus menekankan kemudahan penggunaan
6) Desain level komponen harus fungsional independen (kohesi tinggi)
7) Komponen harus longgar digabungkan satu sama lain dan dengan lingkungan eksternal
8) Representasi desain (model) harus mudah dimengerti
9) Desain harus dikembangkan secara iteratif; dengan masing-masing iterasi, desainer harus berusaha untuk kesederhanaan yang lebih besar

Faktor kualitas eksternal: properti-properti yang dapat mudah diamati?

Faktor kualitas internal: properti-properti yang mengarah ke desain berkualitas tinggi dari perspektif teknis

CONSTRUCTION PRACTICES

Communication
Project initiation
Requirements gathering

Planning
Estimating
Scheduling
Tracking

Modeling
Analysis
Design

Construction
Code
Test

Deployment
Delivery
Support
Feedback

CODING PRINCIPLES
(PREPARATION BEFORE CODING)

1) Memahami masalah Anda mencoba untuk memecahkan
2) Memahami prinsip-prinsip desain dasar dan konsep
3) Memilih bahasa pemrograman yang memenuhi kebutuhan perangkat lunak yang akan dibangun dan lingkungan di mana ia akan beroperasi
4) Pilih lingkungan pemrograman yang menyediakan alat-alat yang akan membuat pekerjaan Anda lebih mudah
5) Membuat satu set unit test yang akan diterapkan setelah komponen kode Anda selesai

CODING PRINCIPLES
(AS YOU BEGIN CODING)

1) Batasi algoritma anda dengan mengikuti praktek pemrograman terstruktur
2) Pilih struktur data yang akan memenuhi kebutuhan desain
3) Memahami arsitektur perangkat lunak dan membuat antarmuka yang konsisten dengan itu
4) Menjaga logika kondisional sesederhana mungkin
5) Buat loop bersarang dengan cara yang membuat mereka mudah diuji
6) Pilih nama variabel yang bermakna, dan ikuti standar lokal lainnya
7) Menulis kode yang mendokumentasikan diri
8) Membuat layout visual (misalnya, lekukan dan garis kosong) yang mempengaruhi kode pemahaman

CODING PRINCIPLES
(AFTER COMPLETING THE FIRST ROUND OF CODE)

1) Melakukan panduan kode
2) Lakukan tes unit (black-box and white-box) dan memperbaiki kesalahan Anda telah menemukan
3) Refactor the code (Mengubah struktur program)

TESTING PRINCIPLES

1) Semua tes harus bisa dilacak dengan persyaratan lunak
2) Pengujian harus direncanakan lama sebelum pengujian dimulai
3) Prinsip Pareto berlaku untuk pengujian perangkat lunak
   - 80% dari kesalahan terungkap dalam 20% dari kode
4) Pengujian harus mulai "dalam kecil" dan kemajuan ke arah pengujian "di besar"
   - Unit testing -> pengujian integrasi -> pengujian validasi -> pengujian sistem
5) Pengujian mendalam tidak mungkin

TEST OBJECTIVES

1) Pengujian adalah proses eksekusi program dengan maksud menemukan kesalahan
2) Sebuah kasus uji yang baik adalah yang memiliki probabilitas tinggi untuk menemukan kesalahan yang belum belum ditemukan
3) Sebuah tes yang sukses adalah salah satu yang mengungkap kesalahan yang belum belum ditemukan

DEPLOYMENT PRACTICES

Communication
Project initiation
Requirements gathering

Planning
Estimating
Scheduling
Tracking

Modeling
Analysis
Design

Construction
Code
Test

Deployment
Delivery
Support
Feedback

DEPLOYMENT PRINCIPLES

1) Harapan pelanggan untuk perangkat lunak harus dikelola
   - Hati-hati untuk tidak menjanjikan terlalu banyak atau untuk menyesatkan pengguna
2) Sebuah pengiriman paket (System Aplikasi) lengkap harus dirakit dan diuji
3) Suatu unit dukungan harus ditetapkan sebelum software tersebut diserahkan
4) Bahan ajar yang sesuai harus diberikan kepada pengguna akhir
5) Software Salah harus diperbaiki dulu, kemudian diserahkan

COMPUTER-BASED SYSTEM

- Defined: Sebuah set atau susunan elemen yang terorganisir untuk mencapai tujuan yang telah ditetapkan dengan memproses informasi
- Tujuannya mungkin untuk mendukung beberapa fungsi bisnis atau mengembangkan produk yang dapat dijual untuk menghasilkan pendapatan bisnis
  - Sebuah sistem berbasis komputer yang menggunakan unsur-unsur sistem
  - Elemen merupakan satu sistem dapat mewakili satu elemen makro dari sistem masih lebih besar
  - Example
    - Sebuah sistem otomatisasi pabrik dapat terdiri dari mesin kontrol numerik, robot, dan perangkat entri data; masing-masing dapat menjadi sistem sendiri
    - Pada tingkat hirarki yang lebih rendah berikutnya, sel manufaktur adalah sistem berbasis komputer sendiri yang dapat mengintegrasikan unsur makro lainnya
  - Peran sistem engineer adalah untuk menentukan unsur-unsur dari sistem berbasis komputer tertentu dalam konteks hirarki keseluruhan sistem

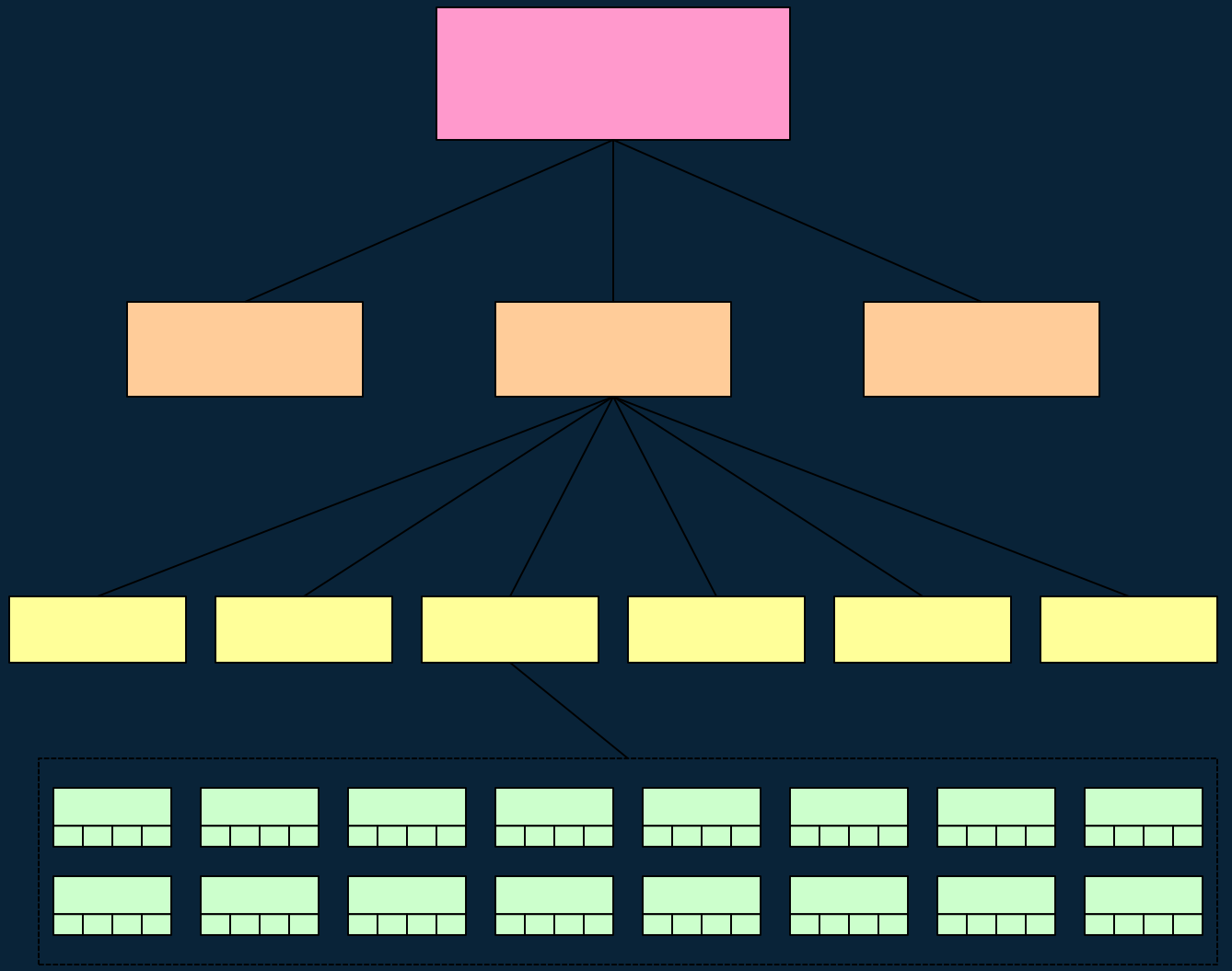
COMPUTER-BASED SYSTEM (CONTINUED)

- Sebuah sistem berbasis komputer yang menggunakan empat elemen sistem yang menggabungkan dalam berbagai cara untuk mengubah informasi
  - **Software**: computer programs, data structures, and related work products that serve to effect the logical method, procedure, or control that is required
  - **Hardware**: electronic devices that provide computing capability, interconnectivity devices that enable flow of data, and electromechanical devices that provide external functions
  - **People**: Users and operators of hardware and software
  - **Database**: A large, organized collection of information that is accessed via software and persists over time
- Penggunaan elemen ini dijelaskan berikut ini:
  - **Documentation**: Informasi deskriptif yang menggambarkan penggunaan dan pengoperasian system
  - **Procedures**: Langkah-langkah yang menentukan penggunaan khusus dari setiap elemen sistem atau konteks prosedural di mana sistem berada

SYSTEM ENGINEERING PROCESS

- Proses sistem rekayasa diawali dengan dunia nyata; bisnis atau produk domain diperiksa untuk memastikan bahwa bisnis atau teknologi konteks yang tepat dapat dibentuk
- Pandangan nyata disempurnakan untuk fokus pada domain tertentu yang menarik
- Dalam domain tertentu, kebutuhan untuk elemen sistem target dianalisis
- Akhirnya, analisis, desain, dan konstruksi dari elemen sistem target diawali
- Pada tingkat pandangan nyata, konteks yang sangat luas dibentuk
- Di tingkat bawah, kegiatan teknis rinci dilakukan oleh disiplin teknik yang relevan (misalnya, rekayasa perangkat lunak)

SYSTEM ENGINEERING HIERARCHY



World View

Domain View

Element View

Component View

PRODUCT ENGINEERING HIERARCHY