BINUS UNIVERSITY ONLINE LEARNING

People
Innovation
Excellence

LECTURE NOTES

Software Engineering

Minggu 01

Introduction to Software Engineering

LEARNING OUTCOMES

Setelah menyelesaikan pembelajaran ini, mahasiswa akan mampu:

- ☑ LO1 – Menjelaskan konsep dari proses model piranti lunak

Outline Materi (Sub-Topic) :

1. *Software Engineering*
2. *Software* Costs
3. *FAQs about Software Engineering*
4. *What are the Costs of Software Engineering?*
5. *What are Software Engineering Methods?*
6. *What is CASE (Computer-Aided Software Engineering)*
7. *What are the Attributes of Good Software?*
8. *What are the Key Challenges Facing Software Engineering?*
9. *Professional and Ethical Responsibility*
10. Kesimpulan

ISI MATERI

1. *Software Engineering*

Software engineering atau rekayasa perangkat lunak merupakan salah satu cabang keilmuwan pada bidang teknologi informasi yang berkembang sangat pesat. Ilmu ini sangat diperlukan untuk memastikan bahwa software yang dibangun memiliki kualitas yang baik. *Software engineering* berfokus pada pembelajaran teori, metode atau alat-alat pendukung yang secara professional mengembangkan sebuah piranti lunak (*software*). Pada masa ini, hampir semua aspek kehidupan manusia telah menggunakan komputer, misalnya penggunaan perangkat lunak pada bidang:

- Politik: ada aplikasi untuk perhitungan hasil pilpres, kampanye kandidat capres, dll
- Ekonomi: terdapat aplikasi untuk jual beli barang, perhitungan saham
- Hiburan: aplikasi pemutar video, musik, dll
- Perbankan: adanya aplikasi *e-banking, mobile banking*, dll

Apalikasi tersebut tentunya sangat mendukung kebutuhan manusia, sehingga *software* sebagai bagian penting dari komputer mendapatkan perhatian khusus dalam pengembangannya. Bahkan pada negara-negara yang telah maju, bahkan di Indonesia sendiri, hamper semua kegiatan telah bergantung pada *software*.

2. *Software Costs*

Software cost adalah biaya yang berhubungan dengan proses pembuatan dan pemeliharaan perangkat lunak, dapat termasuk:

- Biaya lisensi
- Biaya pembuatan

- Biaya pemeliharaan

Contoh biaya lisensi, adalah biaya untuk pembelian aplikasi, misalnya pembelian software untuk basis data, system operasi, anti virus, dan lain-lain.

Jika dibandingkan dengan biaya yang dikeluarkan untuk membeli *hardware*, biaya yang dikeluarkan untuk pengembangan *software* selalu mendominasi keseluruhan pengembangan sebuah sistem. Dan melihat lebih jauh kedalam tahapan pengembangan *software* (mulai dari analisis kebutuhan, perencanaan, perancangan, pembuatan *software*, hingga maintenance), biaya yang dikeluarkan pada tahapan maintenance memakan porsi yang paling besar. Bahkan biaya maintenance dapat berkali-kali lipat dari biaya pembuatan *software* itu sendiri. Salah satu focus pembelajaran dalam *software engineering* adalah pengembangan *software* yang *cost-effective.*

3. FAQs about *Software Engineering*

Berikut ini adalah beberapa pertanyaan yang seringkali muncul terkait dengan *software engineering*:

a. **Apa itu *software*?**

Software adalah program komputer dan semua dokumentasi yang terkait dengan program tersebut. Dokumentasi dapat berupa kebutuhan-kebutuhan pada tahapan analisis, model-model hasil perancangan, database, maupun dokumen cara menggunakan program tersebut.

Berdasarkan penggunaannya, *software* dapat dikategorikan menjadi dua tipe:

- *Software generic, software generic* dikembangkan untuk dipasarkan pada kalayak umum yang memiliki berbagai jenis pengguna.

- Contoh:
    - aplikasi untuk mendukung pekerjaan sehari-hari: Microsoft seperti Word atau Excel.
    - Aplikasi basis data
    - Aplikasi untuk anti virus

- *Software bespoke (custom), software custom* merupakan *software* yang dikembangkan untuk user secara spesifik berdasarkan kebutuhan user tersebut.

  Dalam pembuatannya, *software* dapat dibuat dengan mengembangkan program baru, melakukan konfigurasi terhadap program general, atau menggunakan kembali (reuse) sistem *software* yang sudah ada.

  Contoh:

  - Aplikasi *internet banking* yang digunakan sesuai untuk bank tertentu
  - Aplikasi keuangan pada perusahaan tertentu yang dibuat berdasarkan kebutuhan bisnis yang spesifik pada perusaaan tersebut
  - Apkikasi company profile dengan menggunakan web pada perusahaan atau organisasi tertentu.

b. **Apa itu *software engineering*?**

Software engineering seperti dibahas sebelumnya adalah cabang ilmu teknik yang berfokus pada semua aspek pengembangan sebuah *software*. Seorang *software engineer* harus mengadopsi pendekatan yang sistematis dan terorganisir dalam pekerjaannya, serta menggunakan tool dan teknik yang sesuai untuk masing-masing masalah yang dihadapi.

Perkembangan software engineering juga sangat pesat sekali, dan saat ini terdapat banyak sekali metode pengembangan software engineering, seperti:

- Rational Unified Process (RUP)
- Scrum
- Extreme Programming (XP)
- Crystal
- DSDM

c. **Apa perbedaan antara *software engineering* dan *computer science*?**

Perbedaan utama dari *software engineering* dan *computer science* adalah fokus pembelajarannya. *Computer science* fokus pada teori dan konsep yang fundamental, sedangkan *software engineering* berfokus pada praktek dalam pengembangan dan penyampaian sebuah *software*.

Pada dunia industri, perbedaan ini tidak terlalu terlihat, dan salin melengkapi satu sama lain, walau pun di dalam materi pengajarannya terdapat hal-hal yang dapat digunakan, yang satu bersifat konsep, sedangkan lainnya bersifat pada praktik.

d. **Apa perbedaan antara *system engineering* dan *software engineering*?**

Perbedaan antara *system engineering* dan *software engineering* adalah *system engineering* meliputi semua aspek pada pengembangan sebuah sistem, seperti hardware, *software* dan sistem itu sendiri. Sedangkan *software engineering* hanya berfokus pada komponen *software*, proses dari sebuah sistem yang meliputi pengembangan infrastruktur, control, aplikasi serta database sistem.

Di dalam praktiknya, kedua bidang ini sangat saling mendukung. Biasanya system engineering dilakukan terlebih dahulu sebelum software engineering.

Contoh:

Di dalam proyek pengembangan perangkat lunak, sebelu seorang developer membuat program pada komputer, tentunya diperlukan instalasi pada system komputer yang digunakan, seperti:

- Instalasi komponen perangkat keras
- Instalasi jaringan
- Instalasi system operasi

e. Apa itu proses *software*?

Proses pada sebuah *software* merupakan aktifitas yang bertujuan untuk membuat atau mengembangkan sebuah *software*. Berikut ini adalah aktifitas umum dalam pengembangan *software*:

- Spesifikasi, apa saja yang harus dilakukan oleh sistem dan batasan-batasan pengembangannya.
- Pengembangan, pembuatan *software* itu sendiri.
- Validasi, pengujian terhadap *software* yang telah dibuat, apakah sudah memenuhi kebutuhan dari user.
- Evolusi, perubahan dari *software* sebagai bentuk dari tanggapan atas permintaan terhadap perubahan.

f. **Apa itu model proses *software*?**

Model proses dari sebuah *software* merupakan proses *software* yang telah disederhanakan yang menunjukan suatu perspektif yang lebih spesifik. Contoh perspektif dari proses:

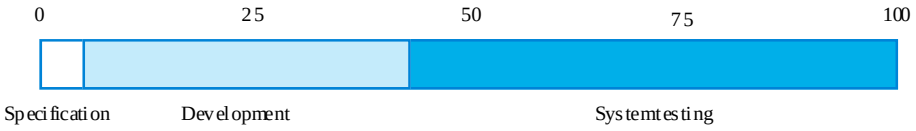
- *Workflow*, urutan aktifitas
- *Data flow*, aliran informasi
- *Role/action*, pengguna aplikasi

Terdapat beberapa proses model umum yang telah dikembangkan yang dapat digunakan langsung dalam pengembangan sebuah *software*:

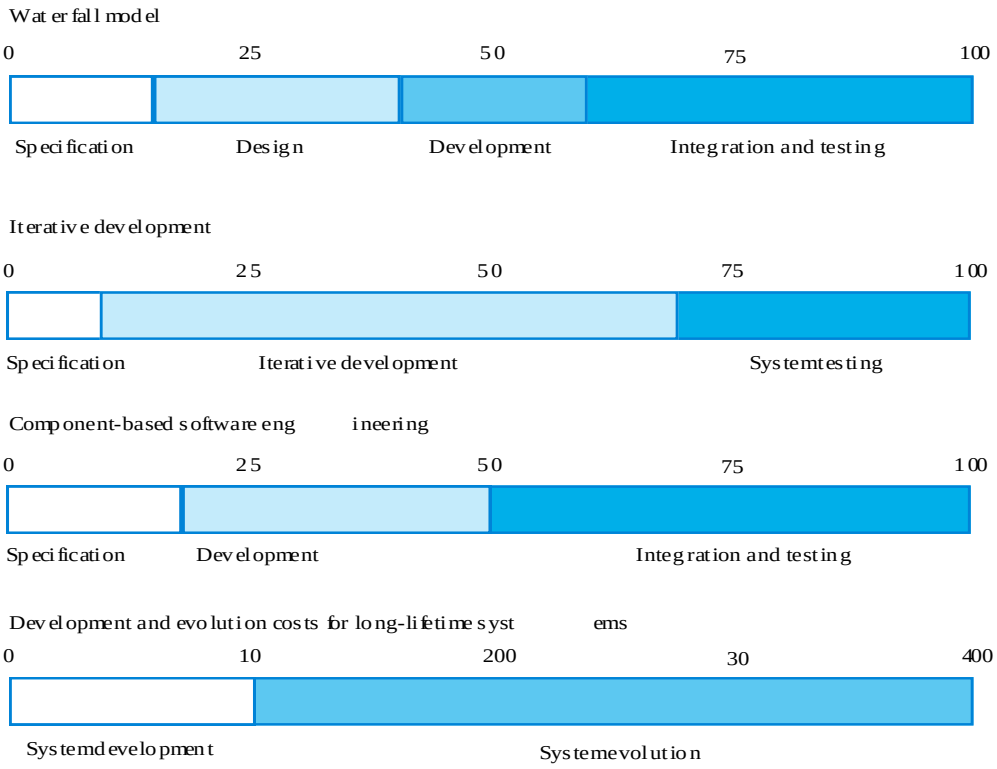
- *Waterfall*
- *Iterative development*
- *Component-based software engineering*

4. *What are the Costs of Software Engineering?*

Biaya dari *software engineering* biasanya dihabiskan 60% untuk proses pembuatannya, dan 40% untuk proses pengujiannya.



Biaya pengembangan *software* tentu saja akan beragam, bergantung pada jenis dan ukuran dari *software* yang akan dikembangkan. Sedangkan biaya distribusi dari sebuah *software* bergantung pada model pengembangan yang digunakan seperti yang digambarkan pada ilustrasi di bawah ini:



5. *What are Software Engineering Methods*?

Metode *software engineering* merupakan pendekatan terstruktur dalam pengembangan *software* yang meliputi model sistem, notasi, aturan, saran perancangan, serta panduan dari proses.

Contoh:

Scrum merupakan salah satu metode software engineering, yang awalnya merupakan suatu framework yang dikembangkan menjadi metode. Di dalam Scrum, terdapat beberapa komponen, yaitu:

- Peran
    - *Product Owner*
    - *Developer*
    - *Scrum Master*

- *Events*
    - *Sprint*
    - *Sprint Planning*
    - *Daily Scrum*
    - *Sprint Review*
    - *Sprint Retrospective*
- *Artifacts*
    - *Product Backlog*
    - *Sprint Backlog*
    - *Increment*

6. ***What is CASE (Computer-Aided Software Engineering)***

CASE merupakan sistem *software* yang bertujuan untuk menyediakan dukungan otomatisasi terhadap aktifitas-aktifitas dalam pengembangan *software*. *Case* dapat dibagi menjadi dua kategori:

- Upper *CASE*, tool untuk mendukung aktifitas pada proses awal pengembangan *software*, seperti analisis kebutuhan dan perancangan model.
- Lower *CASE*, tool yang dikembangkan untuk mendukung aktifitas lebih lanjut dalam pengembangan sebuah *software*, mulai dari pemrograman, debugging serta pengujian program.

Contoh dari CASE misalnya:

- Aplikasi untuk membuat proses perencanaan
- Aplikasi untuk menyediakan fungsi untuk konfigurasi
- Alat bantu testing

7. ***What are the Attributes of Good Software?***

Sebuah *software* harus memenuhi kebutuhan fungsionalitas dan performance dari *user*, dan memiliki sifat *maintainable, dependable, efficient,* serta *acceptable*.

- *Maintainable*: *software* dapat dikembangkan (berevolusi) untuk memenuhi kebutuhan akan perubahan.
- *Dependability*: *software* harus dapat dipercaya.
- *Efficiency*: *software* harus dapat menghemat resource yang dimiliki.
- *Acceptability*: *software* harus dapat diterima oleh user.

Contoh:

Misalnya Anda sebagai seorang software engineer diminta oleh pelanggan untuk membangan suatu aplikasi ERP, maka Anda harus memerhatikan komponen berikut di dalam pembangunan software, misalnya:

- Tampilan aplikasi yang dapat dimengerti oleh pengguna
- Software dapat dikembangkan lagi dengan membuat aturan-aturan dari system yang dapat dibuat parameter, tidak dibuat *hard coded* di dalam program Anda.

- Aplikasi sudah dapat diterima dalam artian misalnya sudah lolos uji *user acceptance test* sebelum digunakan.

8. *What are the Key Challenges Facing Software Engineering?*

Tantangan yang dihadapi dalam *software engineering*, diantaranya adalah:

- *Heterogeneity*. Mengembangkan teknik dalam mengembangkan *software* yang dalam meliputi platform dan lingkungan eksekusi yang beragam.
- *Delivery*. Mengembangkan teknik penyampaian *software* kepada user yang cepat (tidak memakan waktu lama).
- *Trust*. Mengembangkan teknik yang dapat mendemonstrasikan bahwa *software* dapat dipercaya.

9. *Professional and Ethical Responsibility*

Berikut ini adalah beberapa isu dalam tanggung jawab professional seorang *software* engineer:

- Kerahasiaan. Seorang *software* engineer harus dapat menghormati kerahasiaan karyawan atau klien, meskipun tidak ada persetujuan formal yang ditanda tangani.
- Kompetensi. Seorang *software engineer* harus dapat menunjukan kompetensi yang dimiliki.
- Hak kekayaan intelektual. *Software engineer* harus memperhatikan hukum yang mengatur penggunaan properti intelektual seperti paten, *copyright*, dll.
- Penyalahgunaan komputer. *Software engineer* tidak boleh menyalahgunakan kemampuan teknikalnya.

Misalnya:

Anda seorang software engineer yang membuat aplikasi untuk system penggajian di mana Anda mengetahui daftar gaji dari basis data pelanggan Anda, maka Anda tidak boleh menyebarkan informasi ini kepada pihak mana pun.

KESIMPULAN

Sebagai seorang *software engineer*, Anda harus mengetahui prinsip-prinsip dasar dari software, misalnya definisi, metode pembangunan *software*, bagaimana membuat *software* yang dapat digunakan oleh pelanggan. Anda juga harus mengetahui bagaimana etika sebagai seorang *software engineer* di dalam menangani pekerjaan yang dilakukan.

BINUS UNIVERSITY ONLINE LEARNING

People Innovation Excellence

DAFTAR PUSTAKA

- Introduction to *Software Engineering*, http://www.youtube.com/watch?v=Z6f9ckEElsU
- *Software engineering* : a practitioners approach : Chapter 3/ Pages 30, Chapter 4/Pages 40, Chapter 5/Pages 66, Chapter 6/Pages 87
- *Software Engineering* Incremental Model, http://www.youtube.com/watch?v=9cBkihYP1rY
- Video Water Fall, V model,, http://www.youtube.com/watch?v=KaPC0gsEQ68
- *Software engineering* : a practitioners approach : Chapter 8/Pages 131

  Chapter 9/Pages 166, Chapter 10/ Pages 184,Requirements *Engineering* / Specification,, http://www.youtube.com/watch?v=wEr6mwquPLY
- Collaborative Requirements 4Management,, h5ttp://www.youtube.com/watc6h?v=tEXizjE05LA
- UML78 2.0 Tutorial,, http://w9ww.youtube.com/watch?v=OkC7HKtiZC0
- *Software engineering* : a practitioners approach : Chapter 12/Pages 224, Chapter 13/Pages 262, Chapter 14/Pages 285, Chapter 15/Pages 317, 16/Pages 347, 17/Pages 371, dan 18/Pages 391
- Introduction to *Software* Architecture,, http://www.youtube.com/watch?v=x30DcBfCJRI
- Component-based game engine,, http://www.youtube.com/watch?v=_K4Mc3t9Rtc
- *Software* design pattern,, http://www.youtube.com/watch?v=ehGl_V61WJw
- *Software engineering* : a practitioners approach : Chapter 19/Pages 412 & chapter 20/Pages 431
- Introduction to *software* quality assurance,, http://www.youtube.com/watch?v=5_cTi5xBlYg
- *Software* reliability ,, http://www.youtube.com/watch?v=wv51aF_qODA Lean Six Sigma and IEEE standards for better *software engineering*,, http://www.youtube.com/watch?v=oCkPD5YvWqw