

LECTURE NOTES

Software Engineering

Minggu 9

Estimation for Software Project And Project Scheduling

LEARNING OUTCOMES

Setelah menyelesaikan pembelajaran ini, mahasiswa akan mampu:

- ☒ LO 4 – Menganalisa proyek *management software*

Outline Materi (Sub-Topic) :

1. *Software Project Planning*
2. *Software Scope*
3. *Resources*
4. *Project Estimation*
5. *Empirical Estimation Models*
6. *Estimation for OO Projects*
7. *Estimation for Agile Projects*
8. *The Make-Buy Decision*
9. *Project Scheduling*
10. *Earned Value Analysis (EVA)*

ISI MATERI

1. *Software Project Planning*

Sebelum Anda melakukan pengerjaan suatu proyek, tahapan perencanaan harus dilakukan untuk memastikan bahwa kita akan membuat perangkat lunak sesuai dengan spesifikasi, biaya, jadwal, dan kualitas yang ditentukan Bersama dengan pelanggan.

Di dalam PMBOK, sebelum perencanaan terdapat proses inisiasi yang berhubungan dengan keputusan formal untuk dimulainya suatu proyek. Gambar di bawah ini memaparkan tahapan-tahapn di dalam proyek, dimulai dari *initiating*, *planning*, *executing*, *monitoring and controlling* sampai proses *closing*.



Tujuan keseluruhan dari perencanaan proyek adalah untuk menetapkan strategi pragmatic untuk mengendalikan, melacak, dan memonitoring proyek teknikal yang kompleks sehingga proyek dapat diselesaikan tepat waktu dengan berkualitas.

Kegiatan-kegiatan dalam perencanaan proyek:

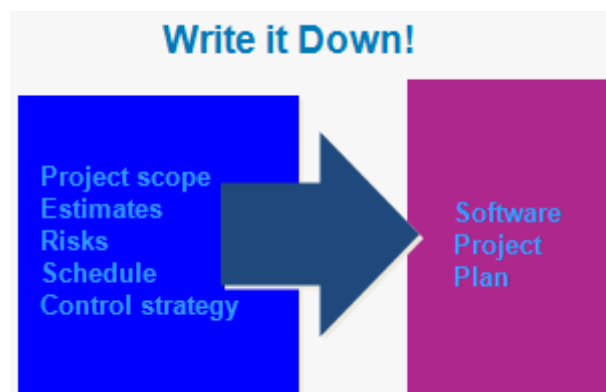
- Menentukan ruang lingkup proyek

- Menentukan kelayakan
- Menganalisa resiko
- Mendefinisikan sumber-sumber daya yang dibutuhkan (meliputi sumber daya manusia, sumber *software* yang dapat digunakan kembali, sumber daya dari lingkungan)
- Memperkirakan biaya dan usaha yang dibutuhkan
- Mengembangkan dua atau lebih prediksi menggunakan ukuran, function point, tugas proses atau *use case*
- Mengembangkan jadwal proyek

Dalam melakukan prediksi terhadap sumber daya, biaya, dan jadwal dibutuhkan:

- Pengalaman
- Akses terhadap informasi *history (metric)*
- Keberanian untuk memberikan prediksi kuantitatif ketika semua informasi kualitatif tersedia.

Prediksi membawa resiko dan resiko memiliki ukuran ketidakpastian.



2. *Software Scope*

Ruang lingkup atau *scope* suatu *software* adalah segala sesuatu yang disepakati untuk dikerjakan pada proyek pembangunan *software*.

Untuk dapat memahami ruang lingkup dari sebuah *software*, maka dibutuhkan pemahaman mengenai:

- Kebutuhan customer
- Konteks bisnis
- Batasan proyek
- Motivasi dari *customer*
- Kemungkinan perubahan

Namun demikian, meskipun semua hal diatas telah dipahami, prediksi tidak dapat dijamin.

3. *Resources*

Sumber daya yang dibutuhkan dari pengembangan sebuah *software* dapat dibagi menjadi 3 kategori utama:

- **Manusia**

Manusia meliputi skill baik teknis maupun softskill, jumlah orang yang dimiliki serta lokasi. Misalnya, pada proyek pembangunan *software* dibutuhkan peran sebagai berikut:

- *Programmer*
- *System analyst*
- *Tester*
- *Business analyst*

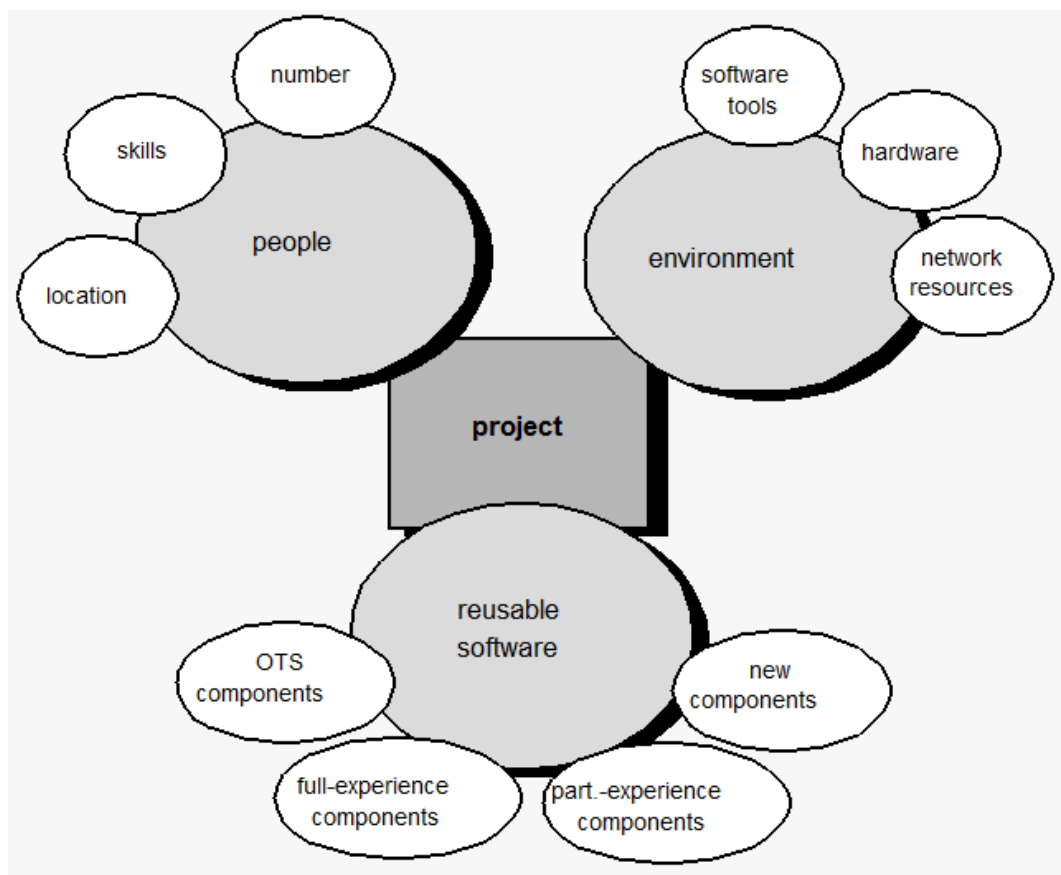
- *Database administrator*
- *Security specialist*

- **Lingkungan**

Lingkungan *software* meliputi *software tools*, *hardware* dan sumber-sumber jaringan.

- **Software yang dapat digunakan kembali (*reuse*)**

Software yang dapat digunakan kembali meliputi: komponen baru, komponen *part-experience*, *component full-experience*, dan *component OTS*.



4. Project Estimation

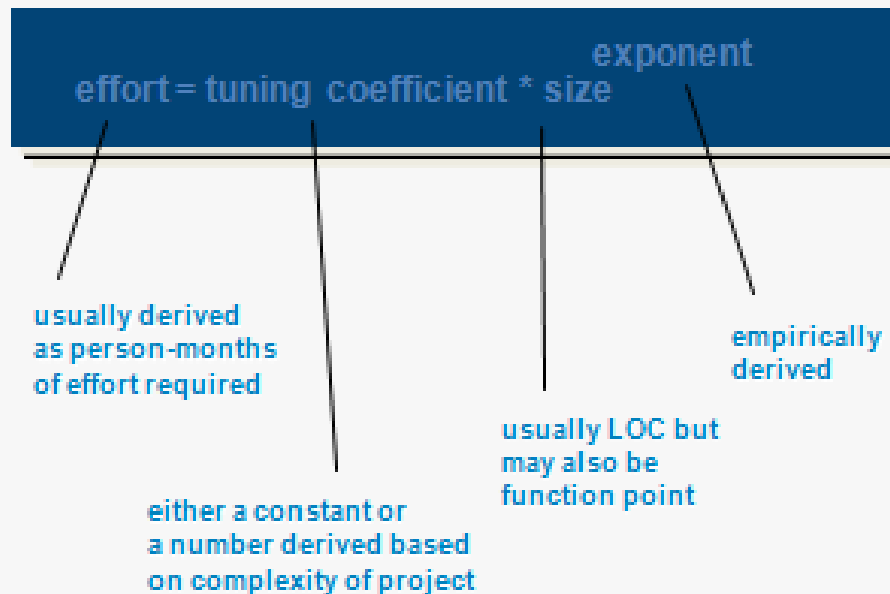
Dalam melakukan prediksi proyek, ruang lingkup harus dipahami seperti yang dibawah pada sub topic di atas. Selain itu, dekomposisi atau elaborasi juga dibutuhkan. Pada banyak kasus, sejarah mengenai metric pada proyek-proyek sebelumnya sangat membantu dalam melakukan prediksi. Dalam melakukan prediksi, setidaknya dibutuhkan dua teknik yang berbeda (contoh di bawah), sehingga hasil dapat dibandingkan dan mendapatkan prediksi yang lebih baik. Dan hal penting yang harus diingat bahwa, prediksi mengandung unsure ketidak pastian, sehingga rencana lebih lanjut perlu dilakukan.

Teknik dalam melakukan prediksi ada 4:

- Masa lampau, dengan melihat pengalaman proyek dari masa lalu untuk proyek-proyek yang sejenis.
- Teknik prediksi yang konvensional
 - o Membagi tugas dan memprediksi usaha
 - o Berdasarkan ukuran (contoh: function point)
- Model empiris
- Tool automasi

5. Empirical Estimation Models

General form:



Gambar diatas adalah contoh model prediksi secara empiris. Selain itu, terdapat COCOMO II yang merupakan sebuah hirarki dari model prediksi yang dalam membantu melakukan prediksi pada area berikut:

- Model komposisi aplikasi

Digunakan pada tahap awal proses pengembangan *software* (*software engineering*), ketika dilakukan prototyping dari *user interface*, pertimbangan terhadap interaksi software dan sistem, penilaian performance, evaluasi kematangan teknologi.

- Model tahap perancangan awal

Digunakan setelah *requirement* ditetapkan dan arsitektur dasar software sudah ditentukan.

- Model tahapan setelah arsitektur

Digunakan selama tahap pembuatan program.

6. *Estimation for OO Projects*

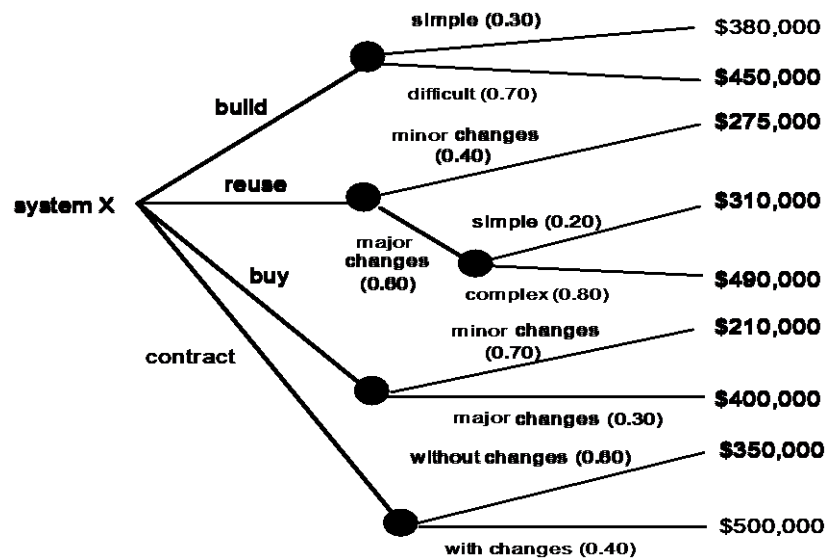
Berikut ini adalah hal-hal yang perlu diperhatikan dalam melakukan prediksi terhadap proyek berorientasi obyek:

- a. Mengembangkan prediksi menggunakan dekomposisi usaha, analisis function point dan metode lainnya yang dapat diterapkan untuk aplikasi konvensional
- b. Menggunakan modeling requirement berorientasi obyek, mengembangkan use case dan menentukan sebuah hitungan
- c. Dari analisis model yang ada, tentukan jumlah dari kelas kunci
- d. Kategorikan tipe interface dari aplikasi dan kembangkan sebuah multiplier untuk kelas pendukung
- e. Kalikan jumlah dari kelas kunci (pada step c) dengan multiplier (pengali) untuk mendapatkan perkiraan jumlah kelas pendukung
- f. Kalikan total kelas (kelas kunci dan kelas pendukung) dengan rata-rata jumlah kerja per kelas.
- g. Cek ulang estimasi berdasarkan kelas dengan mengalikan rata-rata jumlah kerja per use case.

7. *Estimation for Agile Projects*

Dalam melakukan prediksi untuk proyek agile, difokuskan pada *user scenario* (*mini use case*) yang dapat dipertimbangkan secara terpisah untuk kebutuhan perkiraan. *Scenario user* tersebut kemudian didekomposisi menjadi sekumpulan tugas *software engineering* yang dibutuhkan dalam tahap pengembangannya. Setiap tugas diperkirakan secara terpisah (dapat berdasarkan *history*, model empiris atau pengalaman).

8. The Make-Buy Decision



Berdasarkan contoh ilustrasi di atas, diskusikan mengenai keputusan yang sebaiknya diambil dengan alasan yang jelas.

9. Project Scheduling

Berikut ini adalah prinsip-prinsip dalam melakukan penjadwalan:

- Compartmentalization* —mendefinisikan tugas-tugas yang berbeda
- Interdependency* —mengidentifikasi hubungan antar tugas
- Effort validation* —untuk memastikan bahwa sumber daya yang dibutuhkan tersedia
- Defined responsibilities* —orang-orang harus mendapatkan penugasan.
- Defined outcomes* —setiap tugas harus memiliki keluaran (*output*)
- Defined milestones* —melakukan *review* terhadap kualitas

Dalam melakukan penjadwalan untuk proyek aplikasi web atau mobile, terdapat 7 *increments*:

- Informasi dasar dari produk dan perusahaan

- Informasi detail dari produk dan download
- Kuota produk dan urutan proses produk
- Layout space dan rancangan sistem keamanan
- Informasi dan meminta pelayanan monitoring
- Kendali online dari peralatan monitoring
- Kendali akses informasi

10. Earned Value Analysis (EVA)

Earned value adalah ukuran dari sebuah progress/kemajuan yang digunakan untuk menilai kelengkapan suatu software dalam persen. Earned value memberikan analisis kuantitatif. Di dalam organisasi, perhitungan EVA sudah tercakup di dalam tools manajemen proyek.

11. Studi Kasus

Pada organisasi, proyek pengembangan perangkat lunak biasanya menggunakan metode yang sudah biasa dipakai, misalnya PMBOK, SDLC, Agile dan lain-lain. Metode tersebut juga memiliki panduan yang lebih rinci dan sesuai dengan yang dijelaskan pada dokumen ini.

Untuk proses estimasi, Kebanyakan organisasi di dalam menggunakan satuan man-days, misalnya untuk melakukan pekerjaan instalasi operating system dibutuhkan 2 man-days. Untuk estimasi biaya biasa digunakan man-days rate, misalnya 6 juta per man-days. Setiap organisasi dapat memiliki man-days yang berbeda-beda.

DAFTAR PUSTAKA

- Software engineering : a practitioners approach : Chapter 35/Pages 777, Chapter 36/Pages 795
- Sw maintenance and Reengineering,
<http://www.csse.monash.edu.au/~jonmc/CSE2305/Topics/13.25.SWEng4/html/text.html>
- Risk Management & operational Risk,
<http://www.grafp.com/products/risk-manage.html>