

Models of Software Evolution: Life Cycle Model

Magister Teknologi Informasi
Fakultas Ilmu Komputer
Universitas Indonesia

Pengantar

- *Software Evolution*: rangkaian aktivitas yang terjadi selama pengembangan, pemakaian dan pemeliharaan sistem perangkat lunak.



2

Aktivitas dalam daur hidup perangkat lunak

- *System Initiation/Adoption*
 - pengembangan sistem baru ataukah melengkapi sistem yang sudah ada.
- *Requirement Analysis and Specification*
 - identifikasi masalah-masalah yang diharapkan dapat diselesaikan oleh sistem yang baru.

3

Aktivitas dalam daur hidup perangkat lunak

- *Functional Specification or Prototyping*
 - identifikasi dan formalisasi objek-objek, atributnya dan operasi-operasinya beserta batasan-batasannya.
- *Partition and Selection (Build/Buy/Reuse)*
 - dengan hasil aktivitas sebelumnya, sistem dibagi-bagi ke dalam subsistem-subsistem. Kemudian diidentifikasi apakah harus membangun baru, membeli atau me-reuse.

4

Aktivitas dalam daur hidup perangkat lunak

- *Architectural Configuration Specification*
 - pendefinisian inter-koneksi antara modul-modul dari sistem.
- *Detailed Component Design Specification*
 - pendefinisian layanan-layanan tingkat prosedural serta data-data yang ditransformasikan.

5

Aktivitas dalam daur hidup perangkat lunak

- *Component Implementation and Debugging*
 - implementasi spesifikasi yang telah dibuat ke program sumber yang dapat dioperasikan dan divalidasi.
- *Software Integration and Testing*
 - pengecekan keseluruhan integritas konfigurasi arsitektur sistem perangkat lunak verifikasi konsistensi dan kelengkapan modul, verifikasi hubungan antar sumber daya

6


Aktivitas dalam daur hidup perangkat lunak

- *Documentation Revision and System Delivery*
 - penyusunan dokumentasi dari perangkat lunak yang dibangun untuk keperluan *system support*.
- *Training and Use*
 - penyusunan bantuan instruksional dan petunjuk dari kemampuan sistem dan batasan sistem untuk penggunaan sistem yang efektif.

7

Aktivitas dalam daur hidup perangkat lunak


- *Software Maintenance*
 - dukungan terhadap pengoperasian sistem, meliputi: peningkatan fungsi, perbaikan, peningkatan unjuk kerja dan konversi.



8

Software Life Cycle Models

- *Build and Fix Model*
- *Waterfall Model*
- *Prototyping*
- *Incremental Model*
- *Spiral Model*

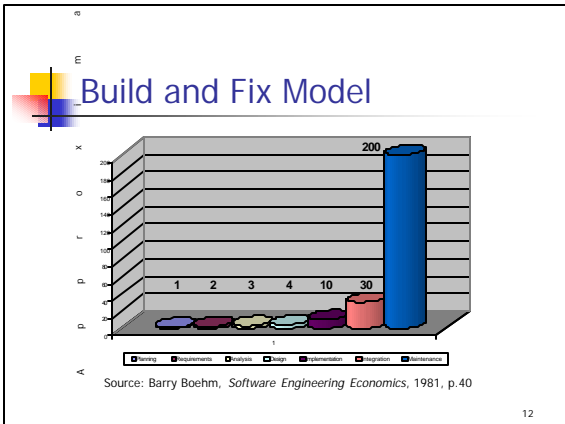
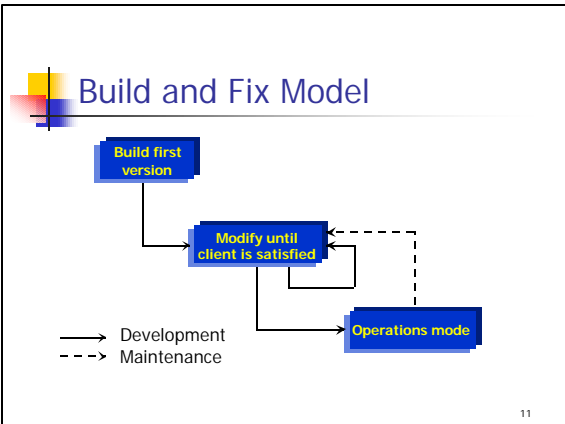


9

Build and Fix Model

- Product is constructed without specifications or any attempt at design.
- Developers simply build a product that is reworked as many times as necessary to satisfy the client.
- This models may work well on short programming (100-200 LOC).

10



Build and Fix Model

- Cost of this approach is far greater than the cost of a properly specified and carefully designed product.
- Maintenance can be difficult without specification or design documents, and the chances of a regression fault occurring are considerably greater.

13

Waterfall Model

14

Waterfall Model

- Titik awal dan titik akhir yang eksplisit.
- Setiap tahapan didefinisikan dengan jelas.
- *System Engineering*
 - Pengumpulan kebutuhan seluruh elemen sistem.

15

Waterfall Model

16

Waterfall Model

- *Software Requirements Analysis*
 - pengumpulan kebutuhan dengan berfokus pada perangkat lunak.
 - meliputi :
 - domain informasi
 - fungsi
 - unjuk kerja
 - antar muka

17

Waterfall Model

- *Design*
 - perancangan struktur data
 - arsitektur perangkat lunak
 - rincian prosedural
 - karakteristik antar muka
- *Coding*
 - penerjemahan perancangan ke bentuk yang dapat dimengerti oleh mesin.

18

Waterfall Model

- *Testing*
 - pengujian logikal
 - pengujian fungsional
 - menemukan kesalahan dan memastikan suatu masukan akan diproses menjadi keluaran yang sesuai dengan yang diinginkan.

19

Waterfall Model

- *Maintenance*
 - bagian terujung dari siklus pengembangan dan dilakukan setelah perangkat lunak dipergunakan.
 - Kegiatan:
 - *Corrective Maintenance*
 - mengoreksi kesalahan pada perangkat lunak, yang baru terdeteksi pada saat perangkat lunak dipergunakan.

20

Waterfall Model

- *Adaptive Maintenance*
 - penyesuaian dengan lingkungan baru, misalnya sistem operasi.
 - sebagai tuntutan atas perkembangan teknologi sistem komputer, misalnya penambahan *printer driver*.
- *Perfective Maintenance*
 - bila perangkat lunak sukses dipergunakan serta memuaskan pemakai. Pemeliharaan ditujukan untuk menambah kemampuannya seperti memberikan fungsi-fungsi tambahan, peningkatan kinerja dsb.

21


Waterfall Model

- *Preventive Maintenance*
 - diturunkan dari konsep pemeliharaan perangkat keras. Perubahan perangkat lunak dilakukan dengan tujuan:
 - Perangkat lunak lebih mudah dipelihara
 - kehandalan perangkat lunak meningkat

22

Waterfall Model


- Kelebihan
 - Lebih disiplin
 - Dorongan bahwa dokumentasi selalu tersedia untuk tiap tahapan (*documentation driven*).
 - Dorongan bahwa setiap produk yang dihasilkan selalu dicek



23

Waterfall Model

- Kekurangan
 - Pengguna hanya mendapatkan deskripsi yang panjang, rinci, dan 'agak membosankan' untuk dibaca.
 - Pengguna baru melihat produk setelah selesai diprogram.



24

Prototyping

Pengembang

Pemakai

25

Prototyping

- Teknik untuk membangun versi sistem yang dikurangi fungsionalitasnya sebelum pengembangan yang sebenarnya.

26

Prototyping

- Bisa saja hanya data capture dan cetak report, tanpa file updating, error handling, validasi.
- Mempercepat tahap perancangan.

27

Prototyping

- Pengembang bekerja sama dengan calon pemakai dalam mengevaluasi *prototype*.
- Sebutan lain: *Rapid Prototyping*
- Prototyping dimungkinkan karena tersedianya teknologi pengembangan perangkat lunak baru. Contoh: *software prototyping languages* dan *environment-nya*, *reusable software* dan *application generator*.

28

Prototyping

29

Prototyping

- Prototype bukan produk akhir, prototype hanyalah model.
- Sistem yang lengkap dapat dibangun dengan teknik:
 - evaluasi/pengembangan
 - buang dan buat baru, hanya demo
 - reuse sebagian/semuanya dari sistem perangkat lunak yang sudah ada.

30

Prototyping

- Kekurangan
 - Pengguna sering menganggap prototype adalah produk akhir.
 - Anggapan di sisi user bahwa perubahan dapat dilakukan dengan mudah dan secepat prototype.
 - User minta lebih banyak setelah melihat prototype.

31

Joint Application Design (JAD)

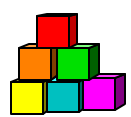
- Dalam penerapan model *prototyping*, peran *user* sangat diharapkan dalam memberikan *feedback*.
- Oleh karenanya, penerapan JAD dimana pengembang dan *user* bekerja bersama dalam satu tim akan sangat mendukung penerapan *prototyping*.



32

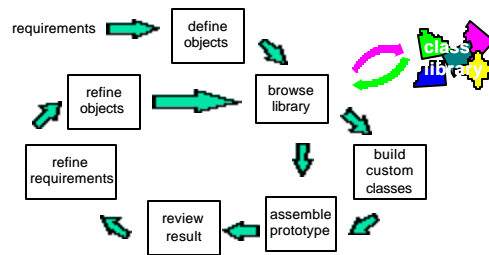
Prototyping: Teknik *Reusability*

- mengkonfigurasi dan mengkhususkan komponen perangkat lunak yang sudah ada ke sistem aplikasi yang akan dibangun.
- cocok untuk pendekatan *object-oriented*.
- Contoh populer: sistem pengaturan interaksi, penggunaan kepastakaan yang sudah ada.



33

Prototyping: Teknik *Reusability*



34

Prototyping: *Application Generation*

- Sangat berguna dalam rangka mempercepat pembuatan *prototype*.
- Sangat populer untuk aplikasi basis data.
- Contoh:
 - sistem interaksi untuk aplikasi basis data
 - *report generator*
 - *menu generator*
 - *screen generator*

35

Incremental Model

- Produk dirancang, diprogram, diintegrasikan, dan diuji sebagai serangkaian serial produk yang incremental.
- Setiap produk terdiri dari program dari berbagai modul yang saling berinteraksi untuk menyediakan sebuah kemampuan fungsional yang spesifik.
- Satu atau beberapa fungsi khusus ditambahkan pada setiap *release*.

36

Incremental Model

- Sistem dikembangkan dengan memberikan fungsi-fungsi dasar terlebih dahulu dan memberikan sistem yang lebih dikembangkan dalam jadwal yang teratur.
- Model ini cukup populer karena termasuk dapat mengakomodir pemeliharaan sistem. Setiap *release* dapat merupakan periodisasi pemeliharaan sistem.
- Sebutan lain: *Incremental Release*.

37

Incremental Model

- Jika produk dibuat dalam *release* yang hanya sedikit maka tidak ada bedanya dengan *build and fix model*.
- Namun jika terlalu sering, akan terlalu banyak waktu dihabiskan untuk melakukan *integration testing*.

38

Incremental Model

- *Incremental* selalu menghasilkan produk operasional pada setiap *release* namun hanya memenuhi sebagian kebutuhan pengguna. Model ini dapat memberikan produk operasional hanya dalam beberapa minggu dibandingkan model *waterfall* atau *prototyping*.
- Perkenalan kepada produk baru yang berlangsung secara bertahap dapat mengurangi 'dampak trauma' dari implementasi produk baru secara keseluruhan.

39

Incremental Model

- *Release* yang bertahap tidak membutuhkan biaya yang sangat besar pada satu saat. *Cash flow* dapat lebih diatur dan klien dapat menghentikan pengembangan setiap saat.
- Setiap *release* yang dibuat harus dapat bekerja sama dan tidak merusak *release* sebelumnya.

40

Spiral Model

- Membagi proses menjadi 4 kuadran
 - *planning*
 - *risk analysis*
 - *engineering*
 - *customer evaluation*
- Memperhatikan *formal risk analysis*
- Hanya cocok untuk *software* berskala besar.

41

