Konstruksi Perangkat Lunak

Disusun Oleh :
Irman Hariman, M.T
iirmanhariman@gmail.com

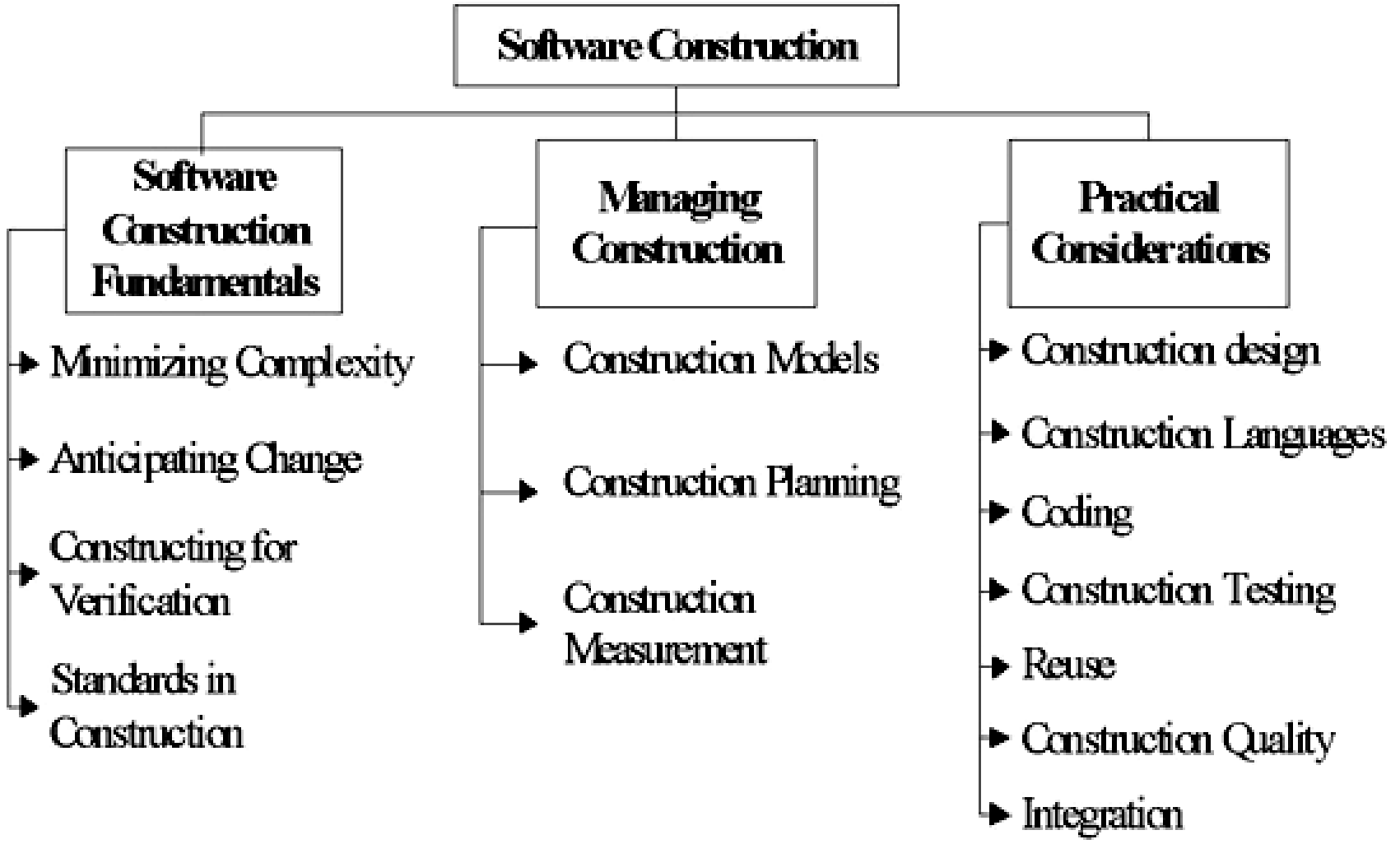
Pendahuluan

- Konstruksi perangkat lunak mengacu pada pembuatan detil kerja, perangkat lunak yang berarti melalui kombinasi *coding*, verifikasi, pengujian unit, integrasi, dan *debugging*.
- Area pengetahuan konstruksi perangkat lunak berhubungan dengan semua area pengetahuan lain, terutama perancangan dan pengujian perangkat lunak
- Walaupun beberapa detil rancangan dikerjakan sebelum konstruksi, masih lebih banyak pekerjaan rancangan yang dikerjakan dalam tahap konstruksi. Dengan demikian area pengetahuan konstruksi perangkat lunak berkaitan erat dengan area pengetahuan perancangan perangkat lunak.
- Melalui konstruksi, pekerjaan unit-test dan integration-test dilakukan. Dengan demikian kaitan dengan area pengetahuan pengujian perangkat lunak sangat erat.

Pendahuluan

- Konstruksi perangkat lunak biasanya menghasilkan volume item konfigurasi terbesar yang harus dikelola dalam suatu proyek perangkat lunak (source files, isi, uji kasus, dan seterusnya), dengan demikian konstruksi perangkat lunak berhubungan erat dengan area pengetahuan Manajemen Konfigurasi Perangkat Lunak.
- Ketika konstruksi perangkat lunak banyak menggunakan berbagai peralatan dan metode, maka hal ini berkaitan erat dengan area pengetahuan peralatan dan metode rekayasa perangkat lunak
- Kualitas perangkat lunak penting di semua area pengetahuan, dengan demikian area pengetahuan kualitas perangkat lunak juga berhubungan erat dengan area pengetahuan konstruksi perangkat lunak

Area Pengetahuan Konstruksi Perangkat Lunak



Dasar Konstruksi Perangkat Lunak

- *Dasar konstruksi perangkat lunak meliputi :*
  - *Meminimalkan kompleksitas*
  - *Mengantisipasi perubahan*
  - *Verifikasi konstruksi*
  - *Standar dalam konstruksi*

Meminimalkan Kompleksitas

- Kebutuhan untuk mengurangi kompleksitas pada dasarnya berlaku untuk setiap aspek konstruksi perangkat lunak dan sangat penting untuk proses verifikasi dan pengujian konstruksi
- Dalam konstruksi perangkat lunak, pengurangan kompleksitas dicapai dengan menekankan pembuatan kode yang sederhana dan mudah dibaca.
- Meminimalkan kompleksitas juga dapat dicapai dengan menggunakan standar dan melalui teknik tertentu
- Hal ini juga didukung oleh teknik konstruksi yang berfokus pada kualitas

Mengantisipasi Perubahan

- Perangkat lunak adalah bagian yang tak terpisahkan dari perubahan lingkungan eksternal, dan perubahan lingkungan eksternal mempengaruhi perangkat lunak dalam berbagai cara
- Mengantisipasi perubahan didukung oleh beberapa teknik khusus, yaitu :
  - Metode komunikasi (misalnya : standar format dokumen dan isi)
  - Bahasa pemrograman (misalnya : bahasa standar untuk bahasa seperti Java dan C++)
  - Platform (misalnya : standar interface programmer untuk *operating system call*)
  - Alat (misalnya : diagram standar untuk notasi UML)

Verifikasi Konstruksi

- Verifikasi konstruksi berarti membangun perangkat lunak sedemikian rupa sehingga kesalahan dapat dilacak dengan mudah oleh insinyur perangkat lunak yang menulis perangkat lunak, sebagaimana pengujian independen dan kegiatan operasional
- Teknik khusus yang mendukung verifikasi konstruksi adalah :
  - Standar *coding* untuk mendukung *review* kode.
  - *Unit testing*
  - Mengorganisir kode untuk mendukung pengujian terotomasi
  - Penggunaan terbatas struktur bahasa yang kompleks atau sulit dipahami

Standar dalam Konstruksi

- Konstruksi bergantung pada penggunaan standar eksternal untuk bahasa konstruksi, alat konstruksi, *interface* teknis, dan interaksi antara konstruksi perangkat lunak dan area pengetahuan yang lain.
- Standar berasal dari berbagai sumber termasuk spesifikasi *interface* perangkat keras dan perangkat lunak seperti *Object Management Group* (OMG), dan organisasi internasional seperti IEEE atau ISO.

Mengelola Konstruksi

- Bagian ini membahas tentang
  - Berbagai macam model konstruksi
  - Perencanaan konstruksi
  - Pengukuran konstruksi

Model-model Konstruksi

- Menurut sudut pandang konstruksi beberapa model lebih linier, seperti siklus hidup air terjun (*waterfall*), dan penyampaian bertahap. Model-model ini memperlakukan konstruksi sebagai suatu kegiatan yang terjadi hanya setelah pekerjaan sebelumnya selesai, yang meliputi detail kebutuhan kerja, pekerjaan desain tambahan, dan perencanaan terperinci.
- Model-model lain yang lebih iteratif, seperti prototyping evolusioner, *Extreme Programming*, dan Scrum. Pendekatan ini cenderung memperlakukan konstruksi sebagai suatu kegiatan yang terjadi bersamaan dengan kegiatan pengembangan perangkat lunak lainnya, termasuk kebutuhan, desain, dan perencanaan.
- Akibatnya apa yang dianggap sebagai konstruksi juga bergantung pada model siklus hidup yang digunakan.

Perencanaan Konstruksi

- Pemilihan metode konstruksi adalah aspek kunci dari kegiatan perencanaan konstruksi
- Pemilihan metode konstruksi mempengaruhi sejauh mana pengerjaan kebutuhan konstruksi, urutan pengerjaan, dan sejauh mana mereka diharapkan akan selesai sebelum pekerjaan konstruksi dimulai
- Pendekatan konstruksi mempengaruhi kemampuan proyek untuk mengurangi kompleksitas, mengantisipasi perubahan, dan verifikasi konstruksi
- Setiap tujuan dapat juga ditujukan pada proses, kebutuhan, dan tingkat desain. Tetapi mereka juga dipengaruhi oleh pilihan metode konstruksi

Perencanaan Konstruksi

- *Perencanaan konstruksi mendefinisikan urutan komponen yang dibuat dan terintegrasi, proses manajemen mutu perangkat lunak, alokasi tugas tertentu untuk insinyur perangkat lunak, dan tugas-tugas lain sesuai dengan metode yang dipilih*

Pengukuran Konstruksi

- *Berbagai kegiatan konstruksi dan artefak dapat diukur, meliputi kode yang dikembangkan, dimodifikasi, digunakan kembali (reused), dihancurkan, kompleksitas, inspeksi statistik, rate penemuan dan perbaikan kesalahan, usaha, dan penjadwalan*
- *Pengukuran ini dapat berguna untuk tujuan pengelolaan konstruksi, memastikan kualitas selama konstruksi, meningkatkan proses konstruksi, serta untuk alasan lain*

Pertimbangan Praktis

- Konstruksi adalah kegiatan di mana perangkat lunak datang untuk menyelesaikan keterbatasan dan kekacauan dunia nyata
- Karena kedekatannya dengan keterbatasan dunia nyata, konstruksi lebih didorong oleh pertimbangan praktis daripada area pengetahuan lainnya

Rancangan Konstruksi

- Beberapa proyek lebih mengalokasikan kegiatan desain pada konstruksi, sementara yang lain mengalokasikan secara eksplisit terfokus pada desain
- Terlepas dari alokasi yang tepat, beberapa pekerjaan desain terinci akan terjadi pada saat konstruksi, dan bahwa pekerjaan desain cenderung dipaksa oleh batasan yang tidak dapat ditawar yang diberikan oleh masalah dunia nyata yang ditangani oleh pernagkat lunak

Bahasa Konstruksi

- Bahasa konstruksi meliputi semua bentuk komunikasi di mana manusia dapat menentukan solusi masalah yang dapat dieksekusi pada komputer
- Jenis bahasa konstruksi yang paling sederhana yaitu bahasa konfigurasi, di mana insinyur perangkat lunak memilih satu set dari pilihan yang terbatas yang telah ditetapkan untuk menciptakan perangkat lunak baru
- Bahasa toolkit digunakan untuk membangun aplikasi di luar toolkit dan lebih kompleks daripada bahasa konfigurasi. Bahasa toolkit secara eksplisit didefinisikan sebagai bahasa pemrograman aplikasi, atau hanya dapat diterapkan oleh set interface dari suatu toolkit

Bahasa Konstruksi

- Bahasa pemrograman adalah jenis yang paling fleksibel dari bahasa konstruksi. Mereka juga mengandung paling sedikit informasi mengenai area aplikasi spesifik dan proses konstruksi, dan juga memerlukan banyak latihan dan keterampilan untuk dapat menggunakannya lebih efektif
- Ada tiga jenis umum notasi yang digunakan untuk bahasa pemrograman, yaitu :
  - Linguistik
  - Formal
  - Visual
- Notasi linguistik dibedakan secara khusus dengan menggunakan string teks untuk mewakili konstruksi prangkat lunak yang kompleks dan kombinasi string tersebut menjadi pola yang memiliki sintaks

Bahasa Konstruksi

- Notasi formal kurang mementingkan intuisi, kata-kata sehari-hari dan string teks dan lebih pada definisi yang didukung dengan definisi yang tepat, jelas, dan formal
- Notasi konstruksi formal dan metode formal berada di pusat kebanyakan bentuk pemrograman sistem, di mana akurasi, perilaku waktu, dan *testability* lebih penting daripada kemudahan pemetaan ke dalam bahasa alami
- Konstruksi formal juga mendefinisikan secara tepat penggunaan penggabungan simbol-simbol untuk menghindari ambiguitas dari banyak konstruksi bahasa alami
- Notasi visual tidak seberapa mementingkan notasi berorientasi teks baik konstruksi linguistik maupun formal

Bahasa Konstruksi

- *Notasi visual lebih mementingkan interpretasi visual langsung dan penempatan entitas visual yang mendasari perangkat lunak*

Coding

- Pertimbangan berikut ini dapat digunakan untuk kegiatan *coding* konstruksi perangkat lunak :
  - Teknik untuk pembuatan *source code* yang dapat dipahami, termasuk penamaan dan tata letaknya
  - Penggunaan *class, enumerated type*, variabel, nama konstanta, dan entitas serupa lainnya
  - Penggunaan struktur kontrol
  - Penanganan kondisi error, baik yang direncanakan maupun pengecualian
  - Pencegahan pelanggaran keamanan (misalnya buffer overruns, atau array index overflow)
  - Penggunaan sumber daya melalui penggunaan mekanisme pengecualian dan disiplin dalam mengakses sumber daya yang dapat digunakan kembali secara serial

Coding

- Organisasi *source code* (ke dalam *statement, routine, class, package,* atau struktur lainnya)
- Dokumentasi kode
- *Tuning* kode

Pengujian Konstruksi

- Konstruksi melibatkan dua bentuk pengujian, yaitu :
  - Pengujian unit
  - Pengujian integrasi
- Tujuan pengujain konstruksi adalah mengurangi jarak waktu antara penyisipan kesalahan ke dalam kode dan waktu kesalahan terdeteksi
- Dalam kasus lain uji kasus dapat dibuat sebelum kode ditulis
- Pengujian konstruksi biasanya melibatkan subset jenis pengujian seperti dijelaskan dalam area pengetahuan pengujian perangkat lunak
- Pengujian konstruksi biasanya tidak termasuk pengujian sistem, pengujian alfa, pengujian beta, pengujian stress, pengujian konfigurasi, pengujian penggunaan, dan lain-lain

Reuse

- Sebagaimana dalam standar IEEE std 1517-1999, IEEE Standard for Information Technology – Software Life Cycle Processes – Reuse : "Penerapan *reuse* perangkat lunak lebih dari sekedar penciptaan dan penggunaan pustaka aset. Hal ini membutuhkan formalisasi praktek *reuse* dengan mengintegrasikan proses dan aktifitas *reuse* ke dalam siklus hidup perangkat lunak".
- Tugas-tugas yang terkait dengan *reuse* dalam konstruksi perangkat lunak selama coding dan pengujian adalah :
  - Pemilihan unit yang *reusable*, database, prosedur pengujian, atau data uji
  - Evaluasi *reusability* dari kode atau pengujian
  - Pelaporan *reuse* pada kode baru, prosedur pengujian, atau data uji

Kualitas Konstruksi

- Ada beberapa teknik untuk menjamin kualitas kode yang dibangun. Teknik utama yang digunakan untuk konstruksi meliputi :
  - Pengujian unit dan pengujian integrasi
  - Pengembangan *test-first*
  - *Code stepping*
  - Penggunaan pernyataan (*use of assertions*)
  - *Debugging*
  - Tinjauan teknis
  - Analisis statis
- Teknik tertentu atau teknik-teknik yang dipilih tergantung pada sifat perangkat lunak yang dibangun, sebagaimana juga pada keterampilan insinyur perangkat lunak yang mengerjakan

Kualitas Konstruksi

- Kegiatan kualitas konstruksi dibedakan dengan aktifitas kualitas lainnya. Kegiatan kualitas konstruksi berfokus pada kode dan artefak yang terkait erat dengan kode : desain skala kecil – sebagai lawan dari artefak lain yang kurang berhubungan secara langsung dengan kode, seperti kebutuhan, desain tingkat tinggi, dan rencana

Integrasi

- Sebuah kegiatan utama dari konstruksi adalah integrasi dari rutin, class, komponen, dan subsistem yang dibangun secara terpisah
- Selain itu sistem perangkat lunak tertentu mungkin perlu diintegrasikan perangkat lunak lain atau sistem perangkat keras
- Hal penting yang berkaitan dengan konstruksi meliputi perencanaan urutan di mana komponen akan diintegrasikan, menciptakan tahapan untuk mendukung versi interim perangkat lunak, menentukan tingkat pengujian dan kualitas kerja yang dilakukan pada komponen sebelum mereka terintegrasi, dan menentukan titik-titik dalam proyek di mana versi interim perangkat lunak diuji

Ada
Yang INGIN
Ditanyakan ?