Open Source System

Pertemuan 05

Alauddin Maulana Hirzan, S.Kom., M.Kom.
NIDN. 0607069401

Fakultas Teknologi Informasi dan Komunikasi, Universitas Semarang

Model Pengembangan

Definisi Model Pengembangan

Definisi:
Model pengembangan (juga disebut sebagai SDLC) adalah kerangka kerja atau metodologi yang memandu proses pembuatan dan evolusi perangkat lunak. Model ini mendefinisikan bagaimana berbagai tugas, seperti mendesain, membuat kode, menguji, dan merilis, diatur dan dijalankan dalam proyek pengembangan perangkat lunak.

Model pengembangan memberikan pendekatan terstruktur untuk mengelola kompleksitas pengembangan perangkat lunak, memfasilitasi kolaborasi di antara anggota tim, dan memastikan pengiriman produk perangkat lunak berkualitas tinggi.

Model Pengembangan

Definisi Model Pengembangan

Model menentukan bagaimana proyek:

1. Untuk melakukan tugas
2. Untuk mengevaluasi kinerja tim
3. Untuk memantau hasil
4. Untuk berinteraksi dengan pelanggan dan pemangku kepentingan lainnya
5. Untuk membagi pengembangan menjadi beberapa tahapan/siklus
6. Untuk mengelola risiko

Model Konvensional

Definisi Model Konvensional

Definisi:
Model pengembangan konvensional, sering disebut sebagai model pengembangan tertutup atau eksklusif, adalah pendekatan tradisional untuk pengembangan perangkat lunak di mana kode sumber suatu proyek dirahasiakan dan biasanya dikembangkan oleh satu entitas, seperti perusahaan atau tim tertutup dalam suatu organisasi.

Di bawah model ini, proses pengembangan sering kali bersifat hirarkis, dengan kontrol ketat atas akses ke kode sumber dan pengambilan keputusan yang terpusat di dalam organisasi.

Model Konvensional

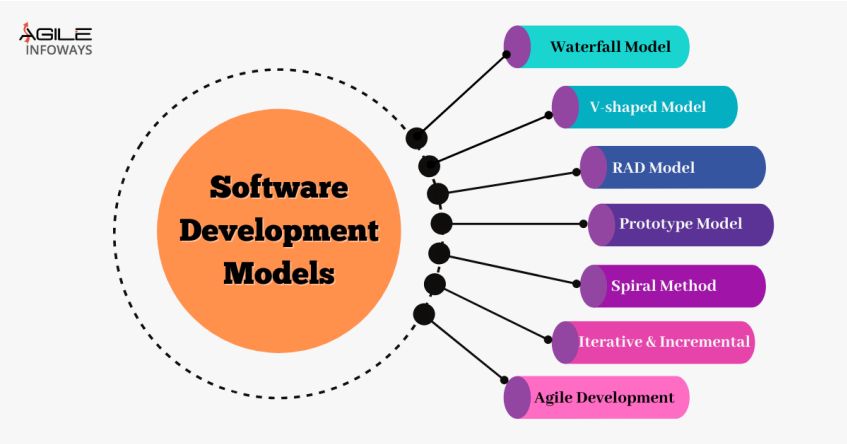
Ciri-ciri Model Konvensional

Ciri khas model konvensional:

1. **Sumber Tertutup**: Kode sumber tidak tersedia untuk umum. Hanya orang yang berwenang dalam organisasi yang memiliki akses ke basis kode.
2. **Kontrol Terpusat**: Keputusan pengembangan, seperti implementasi fitur, perbaikan bug, dan jadwal rilis, dibuat oleh sekelompok kecil individu atau otoritas tunggal dalam organisasi.
3. **Kontribusi Eksternal Terbatas**: Kontributor eksternal biasanya tidak diizinkan untuk memodifikasi atau berkontribusi pada kode sumber

Model Konvensional

Jenis Model Konvensional



Model Konvensional

Jenis Model Konvensional

Jenis-jenis model pengembangan konvensional:

- Model Air Terjun
- Model Inkremental dan Iteratif
- Model Agile
- Model Agile Scrum
- Model Spiral
- Model-V
- Model RUP
- Model Prototipe
- Pemrograman Ekstrim
- Model Ramping
- Model Sistem Dinamis
- Model Berbasis Fitur
- Model Pengembangan Aplikasi Bersama

Model Sumber Terbuka

Definisi Model Sumber Terbuka

Definisi:
Model pengembangan sumber terbuka adalah pendekatan kolaboratif untuk pengembangan perangkat lunak yang memungkinkan siapa pun untuk melihat, menggunakan, memodifikasi, dan mendistribusikan kode sumber proyek secara bebas.

Model ini ditandai dengan transparansi, keterlibatan komunitas, dan pengambilan keputusan yang terdesentralisasi.

Model Sumber Terbuka

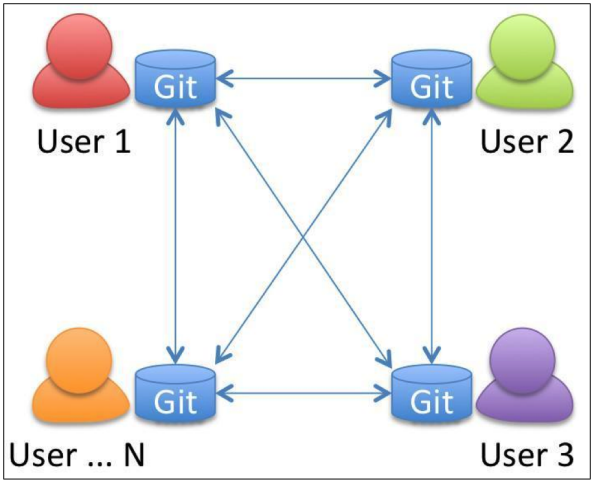
Ciri-ciri Model Sumber Terbuka

Ciri khas modelsumber terbuka:

- **Lisensi Sumber Terbuka**: Proyek-proyek dirilis di bawah lisensi yang sesuai dengan Definisi Sumber Terbuka, seperti GNU General Public License (GPL) atau Lisensi Apache.
- **Pengembangan Terdesentralisasi**: Pengembangan didistribusikan ke seluruh komunitas kontributor, yang mungkin berasal dari berbagai latar belakang dan organisasi.
- **Transparansi dan Kolaborasi**: Proses pengembangan bersifat transparan, dengan diskusi, keputusan, dan perubahan kode yang sering dilakukan di forum publik, milis, atau repositori kontrol versi.

Model Sumber Terbuka

Ciri-ciri Model Sumber Terbuka



Model Sumber Terbuka

Jenis Model Sumber Terbuka

Jenis-jenis model pengembangan sumber terbuka:

- Bazaar
- Cathedral

Informasi

Keduanya memiliki perbedaan yang tidak menonjol, namun memiliki prinsip yang sama

Perbandingan Konvensional dan Sumber Terbuka

Perbandingan Konvensional dan Sumber Terbuka

Aspek	Model	
	Konvensional	**Sumber Terbuka**
Akses Kode	Tertutup	Terbuka
Kontrol	Terpusat	Terdistribusi (Pengembangan Komunitas)
Kontribusi	Kontribusi Luar Terbatas	Terbuka
Lisensi	Lisensi Tertutup	Lisensi Terbuka
Transparansi	Terbatas	Sangat Transparan

Model Bazaar & Katedral

Definisi Model Bazaar

Model Bazaar, yang dipopulerkan oleh Eric S. Raymond dalam esainya ”The Cathedral and the Bazaar,” menggambarkan pendekatan yang terdesentralisasi dan digerakkan oleh komunitas untuk pengembangan perangkat lunak.

Dalam model Bazaar, pengembangan diibaratkan sebagai pasar yang ramai, di mana kontributor dari berbagai latar belakang berkolaborasi secara terbuka untuk meningkatkan perangkat lunak.

Model Bazaar & Katedral

Ciri-ciri Model Bazaar

Model Bazaar memiliki ciri-ciri:

- **Pengembangan Terdesentralisasi**: Pengembangan didistribusikan ke seluruh komunitas kontributor yang besar dan beragam.
- **Meritokrasi**: Kontribusi dinilai berdasarkan kemampuannya, bukan berdasark afiliasi atau status kontributor.
- **Iterasi yang cepat**: Proses pengembangan ditandai dengan rilis yang cepat dan sering, dengan penekanan pada umpan balik dan pengulangan.

Model Bazaar & Katedral

Definisi Model Katedral

Model Katedral, berbeda dengan model Bazaar, mewakili pendekatan yang lebih terpusat dan terkendali untuk pengembangan perangkat lunak.

Dalam model ini, pengembangan diibaratkan seperti pembangunan sebuah katedral, di mana sekelompok kecil pengembang bekerja secara terpisah untuk menciptakan sebuah mahakarya.

Model Bazaar & Katedral

Ciri-ciri Model Katedral

Model Katedral memiliki ciri-ciri:

- **Kontrol Terpusat**: Keputusan dan perencanaan pengembangan terpusat di dalam sebuah kelompok kecil atau organisasi.
- **Siklus Perilisan yang Lebih Lama**: Rilis lebih jarang dan mungkin memerlukan perencanaan dan koordinasi yang ekstensif.
- **Keterlibatan Eksternal Lebih Sedikit**: Kontribusi dari sumber eksternal terbatas, dan proses pengembangan kurang transparan.

Perbedaan Dua Model Sumber Terbuka

Bazaar vs Katedral

Aspek	Model	
	Bazaar	**Katedral**
Akses Kode	Desentralisasi; Komunitas	Terpusat; Kelompok Kecil
Frekuensi Rilis	Lebih Sering	Jarang
Kontributor	Terbuka	Kontribusi Luar Terbatas
Penentu Keputusan	Terdistribusi; Tergantung Merit	Terpusat; Tergantung Kelompok
Transparansi	Transparan	Kurang Transparan

Sistem Kontrol Versi

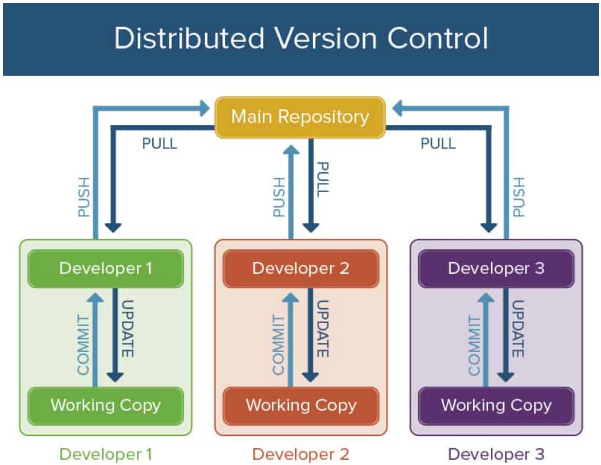
Definisi Sistem Kontrol Versi

Definisi:
Sistem kontrol versi (VCS) adalah alat bantu perangkat lunak yang memungkinkan tim untuk mengelola perubahan pada kode sumber dari waktu ke waktu.

Sistem ini menyediakan fitur-fitur seperti melacak modifikasi, memfasilitasi kolaborasi, dan memungkinkan pengelolaan berbagai versi file. Sistem kontrol versi yang umum termasuk Git, Subversion (SVN), dan Mercurial.

Sistem Kontrol Versi

Definisi Sistem Kontrol Versi



Sistem Kontrol Versi

Cara Kerja Sistem Kontrol Versi

1. **Pembuatan Repositori**: Prosesnya dimulai dengan pembuatan repositori, yang merupakan lokasi penyimpanan terpusat untuk semua file dan riwayat versinya.
2. **Komit Awal**: Pengembang melakukan komit awal dengan menambahkan file mereka ke repositori. Keadaan awal ini sering disebut sebagai cabang ”baseline” atau ”master”.
3. **Melacak Perubahan**: Ketika pengembang mengerjakan proyek, mereka membuat perubahan pada file.

Sistem Kontrol Versi

Cara Kerja Sistem Kontrol Versi

4. **Komitmen**: Ketika pengembang merasa puas dengan perubahan mereka, mereka mengomitnya ke repositori. Komitmen adalah cuplikan perubahan yang dilakukan pada satu atau beberapa file pada waktu tertentu.
5. **Percabangan dan Penggabungan**: VCS memungkinkan pengembang untuk membuat cabang, yang merupakan jalur pengembangan yang terpisah. Cabang berguna untuk mengerjakan fitur baru, perbaikan bug, atau eksperimen tanpa memengaruhi basis kode utama.
6. **Penyelesaian Konflik**: Dalam lingkungan kolaboratif, konflik dapat muncul ketika dua atau lebih pengembang membuat perubahan pada file yang sama secara bersamaan.

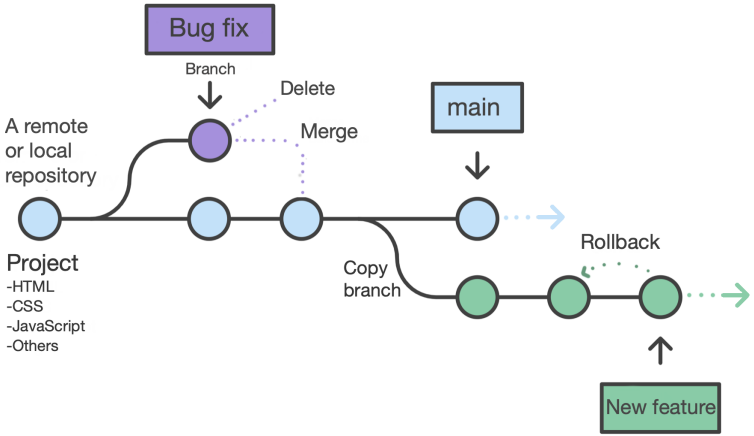
Sistem Kontrol Versi

Cara Kerja Sistem Kontrol Versi

7. **Sejarah dan Revisi**: VCS menyimpan riwayat lengkap semua perubahan yang dilakukan pada proyek, memungkinkan pengembang untuk melihat versi file sebelumnya,
8. **Repositori Jarak Jauh**: Dalam sistem kontrol versi terdistribusi (DVCS), pengembang dapat mengkloning repositori ke mesin lokal mereka.
9. **Autentikasi dan Otorisasi**: VCS sering kali menyertakan fitur autentikasi dan otorisasi untuk mengontrol akses ke repositori dan mengelola izin untuk pengguna dan tim.

Sistem Kontrol Versi

Cara Kerja Sistem Kontrol Versi



Sistem Kontrol Versi

Definisi Workflow Sistem Kontrol Versi

Workflow sistem kontrol versi (VCS) mendefinisikan proses dan konvensi yang diikuti oleh pengembang ketika menggunakan kontrol versi untuk mengelola basis kode mereka.

Alur kerja ini menentukan bagaimana perubahan diusulkan, ditinjau, dan diintegrasikan ke dalam basis kode utama. Ada beberapa alur kerja yang umum, masing-masing dengan keunggulan dan kasus penggunaannya sendiri

Terdapat dua jenis alur:

1. Alur Kerja Terpusat
2. Fitur Alur Kerja Cabang

Sistem Kontrol Versi

Alur Kerja Terpusat

- Dalam alur kerja terpusat, biasanya ada satu repositori pusat yang berfungsi sebagai sumber kebenaran otoritatif untuk proyek.
- Pengembang mengkloning repositori pusat ke mesin lokal mereka, membuat perubahan, dan kemudian mendorong perubahan tersebut langsung ke repositori pusat.
- Namun, hal ini dapat menjadi tidak praktis ketika beberapa pengembang bekerja pada fitur yang berbeda secara bersamaan, karena mereka dapat saling menginjak satu sama lain dan menimbulkan konflik.

Sistem Kontrol Versi

Alur Kerja Cabang

- Dalam alur kerja cabang fitur, setiap fitur baru atau perbaikan bug dikembangkan di cabangnya masing-masing.
- Pengembang membuat cabang baru untuk setiap fitur yang mereka kerjakan, berdasarkan status terbaru dari cabang utama.
- Mereka membuat perubahan pada cabang fitur, melakukan perubahan tersebut secara lokal, dan kemudian mendorong cabang tersebut ke repositori pusat.
- Setelah fitur selesai, pengembang memulai pull request (atau permintaan penggabungan) untuk mengusulkan penggabungan cabang fitur ke dalam cabang utama.

Sistem Kontrol Versi

Alur Kerja Cabang

- Pull request akan ditinjau oleh rekan-rekan, yang akan memberikan umpan balik dan menyetujui perubahan jika memenuhi standar proyek.
- Setelah disetujui, cabang fitur digabungkan ke dalam cabang utama, dan perubahannya menjadi bagian dari basis kode utama.
- Alur kerja ini mendorong kolaborasi, memungkinkan pengembangan fitur secara independen, dan membantu menjaga cabang utama yang bersih dan stabil.

Terima Kasih