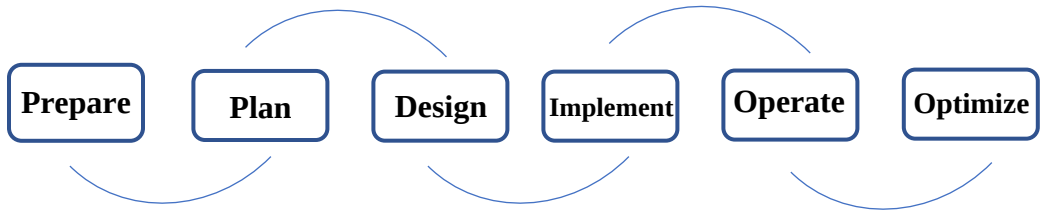
NAMA	:	HENNE IREANI
NIM	:	220110007
KELAS	:	ILMU KOMPUTER-2020
MATKUL	:	MANAJEMEN KONFIGURASI PERANGKAT LUNAK
TANGGAL	:	SABTU, 18 MEI 2024

SOAL

1. Jelaskan definisi manajemen konfigurasi perangkat lunak menurut pemahaman anda
2. Buatkan gambar alur manajemen konfigurasi perangkat lunak kemudian jelaskan
3. Jelaskan sejarah singkat konfigurasi perangkat lunak
4. Apa yang dimaksud dengan git dan github jelaskan menurut pemahaman anda
5. Jelaskan apa yang dimaksud dengan version control system
6. Apa yang dimaksud dengan repository, berikan contoh
7. Apa yang dimaksud dengan commit di dalam git
8. Apa saja komponen SCM, jelaskan
9. Apa saja tahapan control konfigurasi
10. Jelaskan dan berikan contoh teknik-teknik konfigurasi software

JAWAB

1. Manajemen konfigurasi perangkat lunak adalah proses untuk mengidentifikasi dan mengontrol perangkat lunak
2. Alur manajemen konfigurasi perangkat lunak



a. Prepare (Persiapan) : Tahap ini melibatkan persiapan untuk memulai proyek pengembangan perangkat lunak, termasuk identifikasi sumber daya yang diperlukan, menetapkan tujuan, mengumpulkan persyaratan, dan menentukan strategi manajemen konfigurasi yang sesuai.

b. Plan (Perencanaan) : Di tahap ini, rencana manajemen konfigurasi secara detail disusun. Ini mencakup perencanaan pengorganisasian dan dokumentasi konfigurasi, strategi kontrol versi, proses manajemen perubahan, dan pengaturan infrastruktur konfigurasi.
c. Design (Perancangan) : Perancangan konfigurasi perangkat lunak dilakukan di tahap ini. Ini melibatkan desain struktur konfigurasi, termasuk bagaimana komponen-komponen perangkat lunak akan diorganisir, bagaimana kontrol versi akan diimplementasikan, dan bagaimana proses manajemen perubahan akan berjalan.

d. Implement (Implementasi) : Tahap implementasi melibatkan penerapan rencana konfigurasi yang telah dirancang sebelumnya. Ini mencakup pembuatan lingkungan pengembangan, mengatur sistem kontrol versi, mengimplementasikan proses manajemen perubahan, dan memastikan bahwa semua elemen perangkat lunak terintegrasi dengan baik.
e. Operate (Operasi) : Di tahap ini, konfigurasi perangkat lunak beroperasi dalam lingkungan pengembangan atau produksi. Ini melibatkan pemantauan kinerja konfigurasi, menangani perubahan yang mungkin terjadi, menjaga dokumentasi konfigurasi tetap mutakhir, dan menyediakan dukungan bagi pengguna.
f. Optimize (Optimalisasi) : Tahap terakhir adalah tentang meningkatkan efisiensi dan efektivitas konfigurasi perangkat lunak. Ini melibatkan evaluasi kinerja konfigurasi, identifikasi area yang dapat ditingkatkan, menerapkan perbaikan dan peningkatan, serta mengulangi siklus manajemen konfigurasi untuk mencapai hasil yang lebih baik secara keseluruhan.

3. Sejarah konfigurasi perangkat lunak

Konfigurasi perangkat lunak memiliki akar dalam pengelolaan proyek perangkat keras pada tahun 1950-an dan 1960-an. Seiring dengan perkembangan industri perangkat lunak, perlunya pengelolaan versi, pengendalian perubahan, dan pengelolaan konfigurasi menjadi semakin penting. Pada tahun 1970-an, metode formal untuk manajemen konfigurasi mulai muncul, tetapi baru pada tahun 1980-an, konsep ini mulai diterapkan secara luas dalam pengembangan perangkat lunak. Sejak itu, praktik dan alat untuk manajemen konfigurasi perangkat lunak terus berkembang, mencakup pendekatan terpusat seperti sistem kontrol versi dan sistem manajemen konfigurasi perangkat lunak (SCM), serta pendekatan terdistribusi seperti Git.

4. Git adalah system yang mengontrol dan mengelola perangkat lunak dengan cepat dan efisien.
   GitHub adalah platform hosting uang menggunakan Git sebagai basisnya.

5. Version Control System adalah sistem yang memungkinkan pengembang untuk melacak perubahan dalam kode sumber dan dokumentasi, serta mengelola evolusi proyek perangkat lunak. Dengan menggunakan VCS, pengembang dapat merekam setiap revisi kode, membandingkan perubahan, membatalkan perubahan jika diperlukan, dan bekerja secara kolaboratif dengan tim.

6. Repository adalah tempat penyimpanan yang digunakan untuk menyimpan kode sumber, dokumen, dan artefak terkait lainnya yang terkait dengan proyek perangkat lunak. Ini adalah lokasi sentral di mana semua versi dan perubahan kode disimpan dan dapat diakses oleh anggota tim pengembangan. Contohnya adalah repositori Git di GitHub.

7. Commit dalam Git adalah tindakan menyimpan perubahan yang telah dilakukan pada repositori lokal. Ini mencatat snapshot dari semua perubahan yang telah dilakukan sejak commit sebelumnya, memungkinkan pengembang untuk melacak sejarah perubahan dan mengelola evolusi proyek.

8. Komponen SCM

   a. Item konfigurasi : komponen infrastruktur atau layanan apa pun yang memerlukan manajemen untuk mengaktifkan layana TI.
   b. Kontrol perbuhan : proses mengelola perubahan pada item konfigurasi.
   c. Kontrol versi : proses mengelola berbagai versi perangkat lunak.
   d. Manajemen rilis : proses mengelola rilis perangkat lunak ke lingkungan produksi.

9. Tahapan control konfigurasi
   a. Identifikasi : pengumpulan informasi tentang perubahan, termasuk detail tentang perubahan yang diusulkan, termasuk jenis dan tujuannya.
   b. Evaluasi : penilaian terhadap dampak perubahan pada perangkat lunak secara keseluruhan, Ini melibatkan analisis risiko dan pemilihan perubahan yang paling layak dilakukan.

c. Implementasi : penerapan perubahan pada perangkat lunak, termasuk memperbarui kode sumber, dokumen spesifikasi, dan lainnya.

10. Jenis konfigurasi perangkat lunak:

a. Questionnaire-driven Configuration :
Ini melibatkan penggunaan kuesioner atau formulir untuk mengumpulkan persyaratan konfigurasi dari pengguna atau pemangku kepentingan. Contohnya adalah saat pengguna mengisi kuesioner untuk menentukan preferensi pengaturan pada platform perangkat lunak tertentu, seperti tema, bahasa, atau preferensi fitur.

b. Model-driven Configuration :
Dalam model-driven configuration, konfigurasi didasarkan pada model yang telah didefinisikan sebelumnya. Model ini mungkin berisi berbagai elemen konfigurasi dan hubungan antara mereka. Misalnya, dalam pengembangan perangkat lunak, sebuah model dapat digunakan untuk mendefinisikan struktur aplikasi, serta dependensi antara komponen-komponennya.

c. Ontology-driven Configuration :
Konfigurasi ini didorong oleh penggunaan ontologi, yang merupakan representasi formal dari pengetahuan domain tertentu. Ontologi digunakan untuk mendefinisikan konsep, relasi, dan aturan dalam domain yang relevan. Contohnya adalah penggunaan ontologi untuk mengatur konfigurasi dalam sistem cerdas atau IoT yang bergantung pada konteks dan preferensi pengguna.

d. Architecture-driven Configuration :
Ini berkaitan dengan konfigurasi berdasarkan arsitektur sistem. Konfigurasi perangkat lunak dibangun sesuai dengan arsitektur yang telah ditentukan sebelumnya. Contohnya adalah ketika aplikasi web dikonfigurasi untuk menggunakan arsitektur mikro-servis tertentu, di mana setiap servis memiliki peran dan tanggung jawab yang ditentukan.

e. Requirements-driven Configuration :
Konfigurasi didasarkan pada persyaratan fungsional dan non-fungsional yang ditetapkan untuk sistem. Ini memastikan bahwa konfigurasi perangkat lunak memenuhi kebutuhan pengguna dan kebutuhan sistem. Contohnya adalah ketika konfigurasi perangkat lunak dikembangkan untuk memenuhi

persyaratan kinerja tertentu, seperti waktu respons minimum atau throughput maksimum.

f. Semantically-driven Configuration:
Konfigurasi ini mempertimbangkan makna atau semantik dari konfigurasi yang diterapkan. Ini dapat melibatkan pemahaman konteks dan makna di balik konfigurasi yang diadopsi. Sebagai contoh, dalam konteks Internet of Things (IoT), konfigurasi perangkat dapat disesuaikan dengan konteks lingkungan, seperti suhu, kelembaban, atau lokasi.

g. Policy-driven Configuration :
Konfigurasi ini dikendalikan oleh seperangkat kebijakan atau aturan yang telah ditentukan sebelumnya. Kebijakan ini menentukan batasan dan kriteria untuk konfigurasi perangkat lunak. Contohnya adalah penggunaan kebijakan keamanan untuk mengonfigurasi pengaturan keamanan pada aplikasi atau sistem.

h. Connector-driven Configuration :
Konfigurasi ini bergantung pada konektivitas atau integrasi antara komponen-komponen sistem. Pengaturan konfigurasi dilakukan berdasarkan hubungan atau koneksi antara komponen-komponen tersebut. Sebagai contoh, dalam sistem terdistribusi, konfigurasi dapat ditentukan oleh interaksi antara servis dan sumber daya yang tersedia di dalam jaringan.