MAKALAH

MODEL AGILE CONTINUOUS INTEGRATION

Dosen Pengampu : Endang Anggiratih, S.T., M.Cs.

Oleh

Muhammad Hafidz (5200411405)

Ristu Aji Wijayanto (5200411407)

Ikhsan Akbar (5200411435)

Tegar Rangga Nur Ridawan (5200411439)

Medotologi Design Perangkat Lunak Praktik XIV

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2021

KATA PENGANTAR

Puji syukur kami ucapkan kehadirat Allah SWT atas segala rahmat-Nya sehingga makalah yang berjudul “Metode Agile Continous Integration” dapat tersusun sampai dengan selesai.

Tidak lupa kami mengucapkan terima kasih kepada Ibu Endang selaku Dosen pengampu Mata Kuliah Metodologi Perangkat Lunak Praktik yang telah memberikan ilmu materi guna menyelesaikan tugas mata kuliah ini.

Kami sangat berharap semoga makalah yang berisi pengetahuan metode agile Continuous Integration ini dapat menambah pengetahuan dan pengalaman bagi pembaca. Bahkan kami berharap lebih jauh lagi agar makalah ini bisa berguna dalam kehidupan sehari-hari. Bagi kami sebagai penyusun merasa bahwa masih banyak kekurangan dalam penyusunan makalah ini karena keterbatasan pengetahuan dan pengalaman Kami. Untuk itu kami sangat mengharapkan kritik dan saran demi perbaikan makalah ini.

Yogyakarta, 24 Desember 2021

Penulis

DAFTAR ISI

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Pengembangan aplikasi sekarang ini sudah berkembanga pesat dan semakin kompleks. Umumnya *software* menjadi salah satu komponen pendukung dalam suatu instansi atau perusahaan untuk membantu mereka dalam mencapai target strategis instansi. Perangkat Lunak (*Software)* merupakan program komputer yang terintegrasi dengan dokumntasi perangkat lunak seperti *Documentation Requirement System*, *Design Model*, *User Manual.* Dan setiap perangkat lunak itu tidak memungkiri adanya sebuah masalah, masalah yang terjadi pada pengembangan perangkat lunak terdapat saat proses integrasi kode program perangkat lunak. Kesulitan ini terjadi karena sulit menggabungkan modul yang satu dengan modul lainnya, terutama jika dikerjakan oleh developer yang berbeda. Selain itu, seringkali muncul masalah yang terjadi pada saat akan melakukan testing kode perangkat lunak dikarenakan proses testing masih dilakukan secara manual, sehingga memerlukan waktu yang cukup lama dalam proses pembuatan aplikasi.

Penggunaan Continuous Integration merupakan salah satu cara yang dapat digunakan untuk mengatasi masalah dari integrasi kode perangkat lunak dan pengujian yang terjadi, sehingga dapat mengurangi waktu proses development dalam pembuatan suatu perangkat lunak. Continuous Integration merupakan sebuah cara atau konsep yang dapat diterapkan untuk membantu proses integrasi dan pengujian kode perangkat lunak.

CI atau *Continous Integration* merupakan bagian dari kernagka metode *Agile* yang merupakan metodologi dalam pengembangan perangkat lunak yang berdasarkan pada prinsip yang memliki kesamaan atau memiliki *system lifecycle* yang pendek dan diperlukan adaptasi yang cepat terhadap suatu perubahan apapun. Pemilihan Metode Agile Continous Integration yang menjadi kerangka perbandingan penelitian untuk metode *Waterfall,* RAD, dan *Prototype* karena ingin melihat bagaimana *build* dari suatu system itu mendapatkan hasil yang optimum dan serta melihat kelebihan dan kekurangan dari masing masing metode dan parameter yang terdapat didalamnya.

1.2 Rumusan Masalah

1. Apa yang dimaksud metode CI?
2. Apa tujuan CI development?
3. Apa manfaat Metode CI?
4. Apa saja kelebihan dan kekurangan metode CI?

1.3 Tujuan

1. Mengetahui apa yang dimaksud dengan metode CI
2. Mengetahui tujuan CI development
3. Mengetahui manfaat metode CI
4. Mengetahui apa saja kelebihan dan kekurangan metode CI

BAB 2

PEMBAHASAN

2.1 Definisi Continuous Integration

Dalam Software Development, *Version Control System* memiliki peranan penting dalam *maintaining* progress code kita, dengan *Version Control System*, kita juga dapat membuat beberapa *environment software* seperti *production* dan *deploy*. Masing masing environment software terdiri dari beberapa stage yang akan melakukan pekerjaan tertentu.

Umumnya, ada 3 tahapan yang ada dalam *environment software* yaitu: *test, build*, dan *deploy*. mengapa kita perlu *environtment* dan *stages*? Karena dengan environment dan stages, kita bisa melakukan pengecekan terhadapt *software* kita sebelum siap dipakai oleh public. Salah satu pengecekan adalah dengan *test* terhadap *software* kita, selain itu, dengan adanya *environment* dan *stages* dapat membuat *software development* dan rilis update yang cepat dan kuat.

Proses menjalankan stages di masing masing environment tersebut dikenal dengan *Continuous Integration*, *Continuous Deployment, Continuous Delivery*.

Continuous Integration adalah *software development practice* dimana kita build dan test software kita secara otomatis setiap kali melakukan git push ke repositori.

Untuk disetiap commit, CI/CD akan menjalankan unit test untuk melihat apakah terjadi error, jika terdapat error, maka di pipeline akan terlihat tanda silang yang menandakan bahwa test gagal atau tanda checklist yang menandakan test kita berhasil. Pada *Continuous Integration*, karena kita tidak melakukan deploy produk kita dengan frekuensi yang sering, maka *automatic deploy* tidak akan berjalan. Oleh karena itu disebut *Continuos Integration* karena sudah menjalankan *automatic build* dan test namun *manual deploy*.

2.2 Tujuan Continuos Integration

CI atau Continuous Integration merupakan praktek *engineering* dimana setiap anggota tim developer akan mengintregrasikan kode mereka secara rutin atau dengan frekuensi yang tinggi. Tim yang menerapkan metode CI ini bertujuan agar integrasi kode dapat dilakukan setiap hari atau bahkan setiap jam.

Karena integrasi biasnaya memakan banyak waktu dan tenaga serta juga pikiran kita, metode CI bergantung pada alat otomatisasi yang menjalankan proses building dan testing. Inti dari CI adalah menetapkan *software defined lifecycle* yang dapat mengurangi *effort* yang dibutuhkan selama pengambangan produk dan integrasi. CI juga dapat membantu mendeteksi suatu kesalahan dalam kode sejak pada tahap awal pembuatan *software*.

2.3 Manfaat Continuous Integration

CI memiliki banyak manfaat dalam dunia *software development*, beberapa diantaranya adalah sebagai berikut :

- Mendapatkan Feedback Lebih Cepat

  Dalam pipeline ini setiap kode adakan selalu di test secara bersamaan, agar proses *software development* bisa dilakukan dengan seimbang. Dengan CI tools, feedback atau tanggapan atas masalah/error yang terjadi juga bisa diterima dengan lebih cepat. Akhirnya pihak terkait pun bisa langsung menindaklanjuti feedback atau tanggapan tersebut, dengan apapun bentuknya.

- Visibilitas Lebih Baik

  CI Pipeline memilki sifat yang transparan, sehingga memudahkan developer untuk mengontrol perubahan sekaligus menghindari kerusakan pada *software*/ aplikasi. Developer juga dimudahkan dalam menganalisa pengembangan aplikasi dari awal hingga akhir, sehingga semua masalah bisa diatasi dengan segera.

- Deteksi Bug Lebih Awal

  Seperti yang dibahas diatas, CI adalah proses otomatis, jadi jika ada bug akan langsung terdeteksi. Developer tidak akan kesulitas dalam mengembangkan aplikasi karena semua bug yang muncul akan bisa diketahui untuk kemudian diperbaiki

2.4 Kelebihan dan Kekurangan Metode Continuous Integration

a) Kelebihan Continuous Integration
   - Mengurangi proses manual yang berulang
   - Mengurangi resiko karena mendeteksi dan memperbaiki masalah integrasi yang terus menerus
   - Membuat proyek lebih baik dan jelas
   - Menghasilkan perangkat lunak yang dapat di deploy kapan saja dan dimana saja
   - Menghemat waktu dan biaya selama proyek berlangsung

b) Kekurangan Continuous Integration
   - Memerlukan pengaturan awal terlebih dahulu tahap demi tahap
   - Memerlukan test kode untuk mencapai penujian secara otomatis
   - Refactoring (melakukan perubahan pada kode program dari perangkat lunak dengan tujuan meningkatkan kualitas dari struktur program tersebut tanpa mengubah cara program tersebut bekerja) dalam skala besar dapat mengganggu karena dapat merubah basis kode.

BAB 3

CONTOH METODE

Review Jurnal Continuous Integration

Judul	Automatic Deployment System Dengan Menggunakan Metode Continuous Integration di Kakatau
Jurnal	Ilmiah Komputer dan Informatika
Download	https://elib.unikom.ac.id/files/disk1/730/jbptunikompp-gdl-jagamyprie-36468-8-unikom_j-a.pdf
Tahun	2020
Penulis	Jaga My Priera , Robi Tanzil Ganefi
Reviewer	Ristu Aji Wijayanto (5200411407)
Tanggal	26 Desember 2021

Tujuan Penelitian	Memudahkan proses pendistribusian aplikasi ke tim Penguji , Mempermudah komunikasi antara tim penguji dengan developer, dan Mempermudah proses dokumentasi file apk
Subjek Penelitian	Kakatau Startup
Metode Penelitian	Metode yang digunakan adalah turunan dari metode Agile yaitu Continuous Integration dan dapat disimpulkan : 1. Website Automatic Deployment ini dapat mempercepat proses pendistribusian file apk ke tim penguji coba di Kakatu 2. Website Automatic Deployment ini dapat mempermudah komunikasi antara developer dan tim penguji coba di Kakatu

	3. Website Automatic Deployment ini dapat mempermudah proses dalam mendokumentasikan file apk di Kakatu
Hasil Penelitian	Seacra keseluruhan, hasil dari penelitian ini memberikan dampak positif dan development yang dihasilkan sangat lah memuaskan pihak kakatau selaku startup yang menggunakan metode ini , sangatlah efektif untuk development aplikasi yang berjangka pendek dan cepat. Dan automatic deployment system sendiri berintegrasi dengan Version Control System (VCS) dengan begitu semua commit atau pengkodean yang nantinya digunakan pada aplikasi akan terupdate dan jika terdapat error system akan memberikan petunjuk
Kelebihan Penelitian	Kelebihan Penelitian ini adalah metode yang dugunakan oleh peneliti adalah metode yang sangat sederhana dan cepat dan simple dengan memanfaatkan Git sebagai media untuk penggunaan Metode Continuous Integration , tidak dibutuhkan terlalu banyak orang, bisa bekerja secara online, dan aplikasi yang dihasilkan bisa selesai dengan cepat
Kelemahan Penelitian	Kelemahan penelitian ini adalah metode yang digunakan adalah metode yang dikhususkan untuk Senior Developer dan tentunya hanya beberapa perusahaan dan Startup yang memakainya dan metode ini juga tidak cocok untuk para junior.

BAB 4

PENUTUP

Perbandingan Metode Pengembangan Perangkat Lunak

Tahapan Pengembangan Perangkat Lunak	Waterfall	Prototype	RAD	countinous integration (Agile)
Perencanaan Sistem (Systems Planning)	Berawal dari kebutuhan	Berawal dari kebutuhan	Berawal dari kebutuhan	Berawal dari kebutuhan
Analisis Sistem (Systems Analysis)	Kebutuhan data harus dianalisis diawal secara lengkap dan menyeluruh Perubahan data ataupun fungsional akan merubah keseluruh proses pada tahapan berikutnya.	Kebutuhan data dapat ditambah ataupun dikurangi sesuai dengan kebutuhan user, ketika dilakukan testing. Perubahan dapat dilakukan selama sistem atau perangkat lunak masih dalam bentuk prototype	Kebutuhan data dapat ditambah ataupun dikurangi sesuai dengan kebutuhan user, ketika dilakukan testing Kebutuhan fungsi mayor dapat dimodulkan dalam waktu tertentu dan dapat dibicarakan oleh tim RAD yang terpisah.	Kebutuhan data harus dianalisis diawal secara lengkap dan menyeluruh Kebutuhan fungsional dan data dapat dimodulkan dalam waktu tertentu dan dapat dibicarakan oleh tim

				Continous Integration
Perancangan Sistem (Systems Design)	Testing dilakukan ketika semua tahapan pada model sudah selesai. Tidak dapat memberikan gambaran yang jelas mengenai sistem yang dibangun, karena sistem bisa dilihat jika semua tahapan telah dilakukan.	Testing dapat dilakukan ketika prototype telah dibangun, sehingga hasil testing dapat merubah rancangan sistem. Memberikan prototype sebagai gambaran sistem yang akan dibangun, sehingga user dapat melihat dan berinteraksi langsung dengan gambaran sistem. User berperan aktif dalam pengembag an system Sistem yang dibangun akan	Testing dapat dilakukan ketika prototype telah dibangun, sehingga hasil testing dapat merubah rancangan sistem. Memberikan prototype sebagai gambaran system yang akan dibangun, sehingga user dapat melihat dan berinterak si langsung dengan gambaran sistem. User berperan aktif dalam pengembagan system Sistem yang dibangun akan	Testing dilakukan pada saat tahapan integration dapat memberikan gambaran yang jelas mengenai sistem yang dibangun, User berperan aktif dalam pengembagan system Sistem yang dibangun akan

		sesuai dengan keinginan user	sesuai dengan keinginan user Mempunyai kemampuan untuk mengguna kan kembali komponen yang ada (reusable object) sehingga pengembang tidak perlu membuat dari awal lagi dan waktu lebih singkat	sesuai dengan keinginan use
Implementasi Sistem(Systems Implement)	Menerapkan proses perancangan yang baik Evaluasi dilakukan setelah system telah dibangun Mengedepankan kebutuhan fungsional sistem	Tidak menerapkan proses perancangan yang baik Evaluasi dilakukan ketika prototype telah dibangun Mengedepa nkan aspek kenyamanan user	Tidak menerapkan proses perancangan yang baik Evaluasi dilakukan ketika prototype telah dibangun Mengedepankan aspek kenyamanan user dan	Menerapkan proses perancangan yang baik Evaluasi dilakukan Ketika Integration Mengedepankan aspek kenyamanan

			kecepatan pembangunan	user dan kecepatan pembangunan
Pemeliharaan Sistem (Systems Maintenance)	Dilakukan sesuai kesepakatan	Dilakukan sesuai kesepakatan	Dilakukan sesuai kesepakatan	Dilakukan sesuai kesepakatan

Hasil yang dicapai berdasarkan penelitian dan perbandingan yang telah dilakukan diatas, maka dapat disimpulkan:

1) Dapat diketahui karakteristik dari keempat model pengembangan perangkat lunak Waterfall, Prototype, RAD dan Continous Integration

2) Model pengembangan Waterfall cocok digunakan untuk sistem atau perangkat lunak yang bersifat generik, artinya sistem dapat diidentifikasi semua kebutuhannya dari awal dengan spesifikasi yang umum serta sesuai untuk perangkat lunak yang memiliki tujuan untuk membangun sebuah sistem dari awal yang mengumpulkan kebutuhan sistem yang akan dibangun sesuai dengan topik penelitian yang dipilih sampai dengan produk tersebut diuji

3) Model pengembangan Prototype lebih cocok untuk sistem atau perangkat lunak yang bersifat customize, artinya software yang diciptakan berdasarkan permintaan dan kebutuhan (bahkan situasi atau kondisi) tertentu dan sesuai untuk perangkat lunak memiliki tujuan untuk mengimplementasikan sebuah metode atau algoritma tertentu pada suatu kasus.

4) Model pengembangan RAD lebih cocok untuk sistem atau perangkat lunak yang bersifat customize, berskala besar dan memerlukan waktu yang lebih singkat artinya software yang diciptakan berdasarkan permintaan dan kebutuhan (bahkan situasi atau kondisi) tertentu dan sesuai untuk perangkat lunak memiliki tujuan untuk mengimplementasikan sebuah metode atau algoritma tertentu pada suatu kasus, serta memiliki kemungkinan untuk kebutuhan pengembangan kembali dalam jangka waktu yang cukup panjang.

5) Model pengembangan Continous Integration hampir sama seperti RAD cocok untuk sistem atau perangkat lunak yang bersifat customize, berskala besar dan Proses

pengembanganny membutuhkan waktu yang relatif cepat dan tidak membutuhkan resources yang besar.

6) Metode-metode yang dianalisa mempunyai kelebihan dan kekurangannya masing-masing sehingga tidak dapat ditentukan mana yang lebih baik. Dari kelebihan dan kekurangan masing-masing metode, pengembang dapat memilih metode mana yang paling cocok untuk dirinya.

DAFTAR PUSTAKA

- https://www.researchgate.net/publication/350240937_A_STUDY_AND_ANALYSIS_OF_CONTINUOUS_DELIVERY_CONTINUOUS_INTEGRATION_IN_SOFTWARE_DEVELOPMENT_ENVIRONMENT, Yogyakarta 2021
- https://www.snet.tu-berlin.de/fileadmin/fg220/courses/WS1112/snet-project/agile-software-development_li.pdf, Yogyakarta 2021
- https://danielcalencar.github.io/papers/Joao_MSR18.pdf, Yogyakarta 2021
- https://elib.unikom.ac.id/files/disk1/730/jbptunikompp-gdl-jagamyprie-36468-8-unikom_j-a.pdf , Yogyakarta 2021
- https://journal.sttindonesia.ac.id/index.php/bangkitindonesia/article/download/153/130/ , Yogyakarta 2021