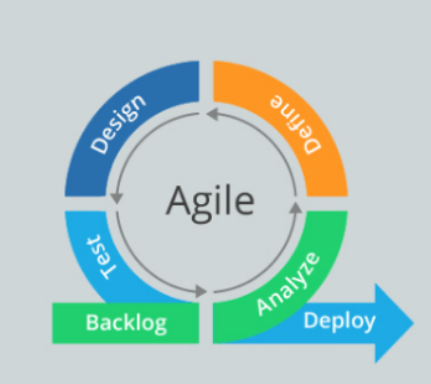
**Nama : Muhammad Arif Safi’i**

**Nim : A11.2019.11818**

**Kls : A11.4617**

**Tugas 1 Proyek Perangkat Lunak**

1. **Model Agile**

****

Agile adalah sekumpulan metode pengembangan software yang dilakukan secara bertahap dan berulang (iterasi). Agile development sering disebut sebagai framework karena di dalamnya memang terdapat berbagai metode yang bisa digunakan sesuai kebutuhan konsumen .

Agile development menggunakan delapan metode yang berbeda, yaitu:

* **Scrum Methodology**

Scrum adalah metode agile yang berfokus pada pengembangan software kompleks. Pengerjaan software pada Scrum dibagi menjadi beberapa proses kecil yang disebut dengan sprints.

Dalam satu sprint, biasanya Anda akan berfokus menyelesaikan satu fitur tertentu. Misalnya, Anda sedang mengembangkan aplikasi dompet digital dan berfokus menambahkan fitur isi saldo.

Nah, agar Scrum berjalan dengan lancar, setiap anggota tim harus diberikan peran, yaitu:

* Product Owner — bertugas memaksimalkan nilai bisnis dari software yang dikembangkan. Plus, ia juga memastikan bahwa list fitur produk telah disusun dengan baik.
* Scrum Master — bertugas memfasilitasi dan memastikan bahwa tim sudah paham dengan proses Scrum. Selain itu, ia juga akan berkoordinasi dengan Product Owner agar bisa memaksimalkan hasil produk dan ROI (Return of Investment).
* Development Team — sekumpulan orang yang memiliki skill tertentu untuk menjalankan project. Misalnya, programmer, designer, writer, dan sebagainya.

* **Scaled Agile Framework (SAFe)**

Scaled Agile Framework (SAFe) ditujukan bagi perusahaan besar atau enterprise yang ingin menerapkan metode Agile. Perusahaan besar cenderung kaku dengan struktur divisi/tim yang banyak. Tak jarang, pengambilan keputusan terkait project bisa memakan waktu lama karena perlu koordinasi antar divisi. Nah, SAFe adalah solusi dari masalah tersebut. Dengan SAFe, kerjasama antar divisi saat menjalankan project akan lebih efektif dan fleksibel.

Sebagai contoh, LEGO menerapkan SAFe pada 20 tim produknya. Salah satu caranya adalah dengan mengadakan meeting antar tim setiap delapan minggu sekali.

Pada meeting tersebut, mereka saling menunjukkan hasil, menjelaskan rencana, mengukur risiko, dan sebagainya. Artinya, semua keputusan penting diambil saat itu juga tanpa harus melalui birokrasi yang rumit.

* **Lean Software Development (LSD)**

Lean Software Development (LSD) adalah metode Agile yang mempunyai satu tujuan: mengembangkan software dengan sumber daya sehemat mungkin. Bagaimana bisa? Caranya, dengan merilis produk dengan fitur terbatas yang disebut dengan Minimum Viable Product (MVP). MVP memiliki fitur standar dan terus dikembangkan sesuai feedback konsumen. Dengan begitu, pengembangan akan menghemat sumber daya karena tak perlu menambahkan fitur yang belum tentu dibutuhkan.Sebagai contoh, awalnya Gojek bukanlah aplikasi kaya fitur. Gojek menggunakan call center yang menghubungkan penumpang dengan driver.

Nah, call center itulah yang merupakan MVP Gojek. Setelah pengguna bertambah dan ingin perbaikan fitur, Gojek mulai membuat aplikasi yang dikenal saat ini.

Itulah pentingnya LSD. Bayangkan kalau Anda sudah lembur berhari-hari untuk menciptakan fitur canggih dan ternyata konsumen tidak membutuhkannya. Membuang-buang waktu dan tenaga saja, kan?

Nah, LSD tentu sangat cocok diterapkan pada startup atau bisnis baru yang masih mempunyai sumber daya terbatas dalam pengembangan produk.

Baca juga: Lean Startup: Metode yang Wajib Dipelajari Pebisnis Online!

* **Kanban**

Kanban adalah metode Agile yang menggunakan visual dalam prosesnya. Hal ini membuat flow kerja Anda lebih jelas karena bisa memantau setiap aktivitasnya. Nah, visual yang digunakan metode Kanban disebut sebagai Kanban Board. Umumnya, 

Kanban Board dibagi menjadi tiga tahap, yaitu:

* To Do — aktivitas yang akan dikerjakan.
* In Progress — aktivitas yang sedang dikerjakan.
* Done — aktivitas yang sudah selesai.

Ketiga tahap di atas membuat semua anggota tim bisa mengetahui sampai mana proses pengembangannya dengan mudah dan cepat. Misalnya, ketika mengerjakan halaman About Us.

Task pembuatan halaman About Us yang sudah selesai akan masuk ke tahap Done. Lalu, proses mengisi konten halaman About Us yang diperlukan ternyata masih di tahap On Progress. Dan, masih ada task To Do yaitu memasukkan link halaman About Us di homepage menunggu task on Progress selesai.

Oh ya, Kanban Board bisa berupa papan tulis biasa atau papan virtual seperti Trello dan Quire. Anda tinggal menyesuaikannya dengan kebutuhan dan kemampuan tim Anda.

* **Extreme Programming (XP)**

Extreme Programming (XP) adalah metode agile yang lebih berfokus ke aspek teknis pengembangan. Tujuannya, agar software yang dihasilkan mempunyai kualitas tinggi, sehingga kemampuan tim pengembangnya juga meningkat drastis.

Itulah kenapa metode ini disebut dengan “extreme.” Sebab, untuk mencapai tujuan tersebut, tim harus bekerja dengan extra keras dan dipaksa keluar dari zona nyaman.

Nah, agar Anda mendapat gambaran lebih jelas, berikut penjelasan proses XP yang harus Anda jalankan:

* Planning  — tim developer dan konsumen bertemu untuk merencanakan apa yang akan dikerjakan di setiap iterasinya.
* Designing — mulai mendesain bentuk dasar software yang sederhana.
* Coding — dimulainya proses coding secara intensif oleh tim. Di sini, anggota tim juga saling me-review coding dari anggota tim lainnya.
* Testing — mengetes software berulang kali apakah berfungsi dengan baik (unit tests) dan sesuai kebutuhan konsumen (acceptance tests)
* Listening — berdiskusi dan mendengarkan feedback dari konsumen. Maka dari itu, konsumen harus selalu tersedia on-site dalam XP.

Dengan mengikuti proses di atas, tim Anda tentu lebih bisa memberikan hasil maksimal.

* **Crystal Methodology**

Crystal Methodology adalah metode Agile yang lebih berfokus ke kondisi tim yang mengerjakan, alih-alih ke proses atau tools-nya. Mulai dari interaksi tim, komunikasi, feedback, dokumentasi, dan sebagainya.

Dengan Crystal Methodology, pengembangan software akan lebih optimal karena disesuaikan dengan kondisi tim Anda.Sebagai contoh, kalau tim Anda kecil tidak perlu banyak dokumentasi karena setiap perubahan bisa langsung dikomunikasikan. Sebaliknya, kalau tim sudah besar, perlu dokumentasi yang jelas dan dapat diakses oleh semua anggota tim.

Nah, Crystal Methodology sendiri mempunyai tujuh prinsip utama untuk mendukung pengembangan software secara tim dengan lancar. Apa saja?

* Frequent Delivery — Anda harus sering merilis dan mengetes kode ke user. Tujuannya, agar Anda terhindar dari menciptakan software yang tak dibutuhkan.
* Reflective Improvement — Tak peduli bagaimana kualitas produk yang diciptakan, pasti ada aspek yang bisa ditingkatkan oleh tim Anda.
* Osmotic Communication  — Komunikasi antar anggota tim seolah mereka berada di satu ruangan yang sama. Artinya, semua anggota bisa mendengar dan mendapat informasi yang sama.
* Personal Safety — Anggota tim bisa mengungkapkan pendapat tanpa rasa takut atau mendapat ancaman. Baik itu tentang masalah atau menyampaikan ide baru.
* Focus on Work —Tiap anggota tim sebaiknya paham dan bisa melakukan pekerjaannya dengan baik. Jadi, mereka bisa fokus untuk bekerja sama mencapai tujuan.
* Easy Access to Expert Users — Anggota tim bisa dengan mudah bertanya atau meminta pendapat dari user yang ahli di bidangnya.
* Technical Tooling — Tim sebaiknya menggunakan tools pendukung seperti test otomatis, manajemen konfigurasi, dan integrasi rutin. Tujuannya, agar kesalahan bisa cepat dideteksi.

* Dynamic Systems Development Method (DSDM)

Dynamic Systems Development Method (DSDM) adalah metode agile yang mengutamakan keterlibatan semua anggota tim secara berlanjut.

DSDM juga mempunyai filosofi utama yaitu, “menciptakan software yang mempunyai manfaat nyata ke bisnis.”  Filosofi ini juga didukung oleh delapan prinsip yang harus dijalankan agar bisa mencapai tujuan tersebut:

* Fokus pada kebutuhan bisnis  — Software yang dikerjakan haruslah mempunyai manfaat yang jelas pada bisnis.
* Selesaikan tepat waktu  — Pengembangan software harus mempunyai tanggal penyelesaian yang jelas.
* Kolaborasi  — Tim Anda harus melibatkan semua stakeholder dari awal sampai akhir.
* Kualitas  — Kualitas akhir software yang akan dikerjakan harus sudah disetujui sejak awal.
* Ciptakan pondasi yang kuat  — Tim Anda harus menciptakan desain awal agar paham software seperti apa yang akan dikembangkan.
* Kembangkan dengan bertahap  — Anda harus mampu menerapkan perubahan dari feedback pada setiap iterasinya.
* Komunikasi yang jelas  — Usahakan Anda melakukan meeting yang rutin dimana setiap anggota bisa menyampaikan masalah atau kendala.
* Tunjukkan kepemimpinan  — Team leader harus memastikan bahwa rencana dan informasi perkembangan bisa diakses siapapun dengan mudah.

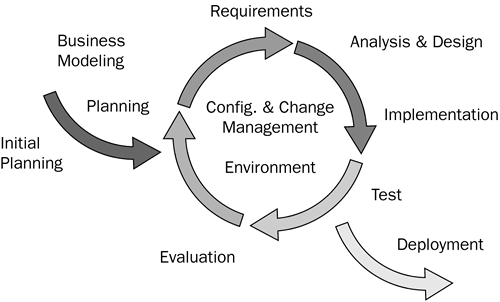
* Feature Driven Development (FDD)

Seperti namanya, Feature Driven Development (FDD) adalah metode Agile yang berfokus untuk menyelesaikan satu fitur. Biasanya, sebelum pengembangan software dimulai Anda harus menulis daftar fitur apa saja yang hendak dimasukkan. Lalu, Anda harus menyelesaikan satu fitur dari daftar tersebut di setiap iterasinya. Yup, mirip dengan Scrum, bukan? Nah, bedanya setiap iterasi pada FDD biasanya hanya berlangsung selama 2 sampai 10 hari. Sedangkan iterasi pada Scrum umumnya dua sampai empat minggu lamanya. Maka dari itu, fitur pada FDD memiliki skala yang kecil dan lebih spesifik agar bisa diselesaikan tepat waktu. Misalnya, buat tombol login atau ganti ikon pencarian.

Lengkapnya, berikut lima langkah yang harus Anda jalankan saat menerapkan FDD:

* Mengembangkan model dasar
* Menulis daftar fitur
* Merencanakan pengembangan setiap fiturnya
* Mendesain fitur
* Membangun sesuai fitur

1. **Rational Unified Process**



Rational Unified Process (RUP) adalah kerangka kerja proses pengembangan perangkat lunak yang diciptakan oleh Rational Software Corporation. RUP menyediakan panduan pandunan, template, dan contoh semua aspek dari tahapan pengembangan sistem informasi [10].

Terdapat tiga perspektif dalam RUP yaitu Dynamic Perspective and Lifecycle Phases, Static Perspective, dan Practice Perspective and Core Process. Dynamic Perspective menunjukkan fase RUP dari waktu ke waktu yang terus berubah. Static Perspective terbuat dari hal-hal yang tidak mengubah diri mereka sendiri tetapi bekerja untuk mengubah proses dinamis. Practice Prespective dibuat dari implementasi yang efektif berdasarkan pengalaman sebelumnya baik yang digunakan selama proses.

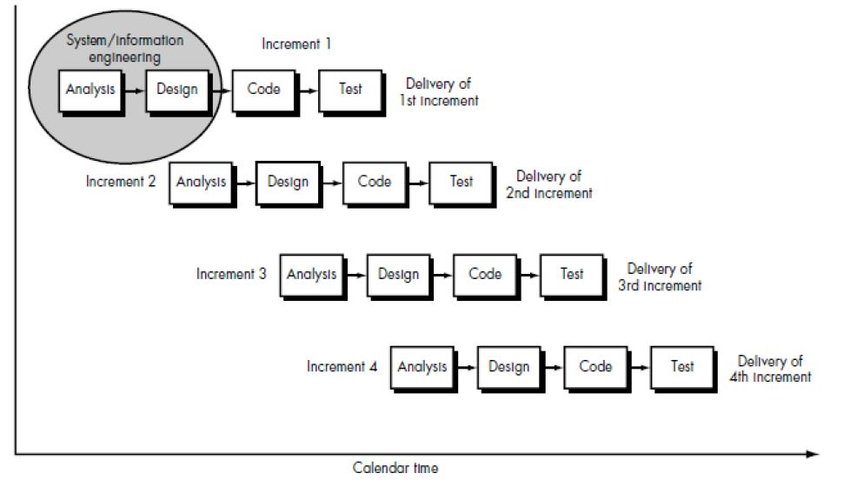
RUP digambarkan seperti pada Gambar 2. Arsitektur RUP. Pada gambar tersebut terdapat dua dimensi yaitu horisontal dan vertikal. Pada dimensi horisontal menunjukkan waktu dan aspek dinamis dari proses yang digambarkan dalam bentuk cycle, phase, iteration dan milestone. Sedangkan pada dimensi vertikal menunjukkan aspek statis dalam proses yang digambarkan dengan activities, artifacts, workers dan workflow.

Dynamic Perspective and Lifecycle Phases – Pada Dynamic Perspective and Lifecycle Phases memiliki empat fase yaitu : inception, elaboration, construction dan transition. Setiap fase memiliki satu atau lebih iterasi dan dilengkapi dengan milestone. Fokus dari iterasi pada setiap fase adalah untuk menghasilkan hasil teknis yang akan memenuhi tujuan fase.

Static Perspective – Model RUP dibangun di atas tiga entitas yang termasuk ke dalam Static Perspective. Ketiga entitas tersebut adalah workers, activities dan artifacts. Workflow berhubungan dengan ketiga entitas tersebut secara berurutan untuk menghasilkan desain dari sistem informasi. Suatu proses menggambarkan siapa (who) yang melakukan apa (what), bagaimana (how) dan kapan (when). RUP digambarkan menggunakan empat elemen yaitu : workers, the who; activities, the how; artifacts, the what; workflows, the when.

Practice Perspective and Core Process – Alur kerja Core Process dibagi menjadi enam bagian yaitu: business modelling workflow, requirement workflow, analysis and design workflow, implementation workflow, testing workflow, dan deployment workflow. Selain itu RUP juga memiliki tambahan Core Process yaitu: environment workflow, configuration and change workflow, dan project management workflow

1. **Model Incermental**

Seperti yang kita tahu bahwa software development model adalah suatu model yanmg mereresentasikan suatu proses software. Terdapat banyak model dalam software development model, salahsatunya adalah incremental model.

Incremental model adalah model pengembangan sistem pada software development berdasarkan requirement software yang dipecah menjadi beberapa fungsi atau bagian sehingga model pengenbangannya secara bertahap. Dengan menggunakan incremental model dapat membantu kita untuk mengurangi biaya sebelum mencapai level dari initial productivity dan mengakselerasi proses dari pembuatan suatu fungsi sistem.

Incremental model menggambarkan suatu proses dimana mengutamakan perhatian pada system requirement dan mengimplementasinya dalam team development.

Tahap – tahap pada incremental model, yaitu:

* Requirement: proses penentuan kebutuhan atau analisis kebutuhan.
* Specification: proses spesifikasi dimana menggunakan analisis kebutuhan sebagai acuannya.
* Architecture Design: perancangan software yang terbuka agar dapat diterapkan sistem pembangunan per-bagian pada tahapan selanjutnya.
* Code: melakukan koding.
* Test: melakukan testing dalam model ini.

More Risky Incremental Model adalah suatu solusi untuk mengantisipasi kondisi yang tidak diinginkan saat mengimplentasikan incremental model.

More Risk Incremental Model menerapkan sistem kerja parallel dimana tahap awal yaitu membuat daftar kebutuhan (requirement) user kemudian dianalisa.

Setelah spesifikasi pertama telah selesai, tim design akan  membuat design untuk modul pertama. Pada saat itu juga, tim spesifikasi akan membuat spesifikasi untuk modul kedua dan jika sudah selesai, maka tim design akan membuat design untuk modul ke dua, dan seterusnya. Sehingga tidak harus menunggu modul pertama selesai hingga dikirim ke user, baru lanjut ke modul kedua.

**Beberapa Kelebihan Dari Mode Incremental atara lain :**

* Merupakan model dengan manajemen yang sederhana
* Pengguna tidak perlu menunggu sampai seluruh sistem dikirim untuk mengambil keuntungan dari sistem tersebut. Increment yang pertama sudah memenuhi persyaratan mereka yang paling kritis, sehingga perangkat lunak dapat segera digunakan.
* Resiko untuk kegagalan proyek secara keseluruhan lebih rendah. Walaupun masalah masih dapat ditemukan pada beberapa increment. Karena layanan dengan prioritas tertinggi diserahkan pertama dan increment berikutnya diintegrasikan dengannya, sangatlah penting bahwa layanan sistem yang paling penting mengalami pengujian yang ketat. Ini berarti bahwa pengguna akan memiliki kemungkinan kecil untuk memenuhi kegagalan perangkat lunak pada increment sistem yang paling bawah.
* Nilai penggunaan dapat ditentukan pada setiap increment sehingga fungsionalitas sistem disediakan lebih awal.
* Memiliki risiko lebih rendah terhadap keseluruhan pengembagan sistem,
* Prioritas tertinggi pada pelayanan sistem adalah yang paling diuji

**Kelemahannya adalah :**

* Kemungkinan tiap bagian tidak dapat diintegrasikan
* Dapat menjadi build and Fix Model, karena kemampuannya untuk selalu mendapat perubahan selama proses rekayasa berlangsung
* Harus Open Architecture
* Mungkin terjadi kesulitan untuk memetakan kebutuhan pengguna ke dalam rencana spesifikasi masing-masing hasil increment.
* Membutuhkan waktu yang relative lama untuk menghasilan product yang lengkap