Suplement

Bab 3 " Proses Rekayasa Perangkat Lunak "

13 + MODEL SDLC

Overview

Sangat sulit untuk membuat sebuah perangkat lunak tanpa perancangan yang maksimal. Beberapa teknik dalam mengembangkan perangkat lunak terus dikembangkan hingga kini. Masih banyak perdebatan mengenai metode yang paling baik dan paling sesuai untuk segala tipe perangkat lunak. Meski demikian, ada perencanaan lebih baik daripada tidak ada perencanaan sama sekali.

Pemilihan model pengembangan perangkat lunak yang tepat adalah salah satu faktor kunci yang menentukan kesuksesan suatu proyek perangkat lunak. Setiap model pengembangan memiliki karakteristik, kelebihan, dan kekurangan yang berbeda, yang menjadikannya lebih cocok untuk tipe proyek tertentu. Oleh karena itu, pemilihan model pengembangan yang tepat dapat memengaruhi efisiensi tim pengembang, kualitas perangkat lunak yang dihasilkan, serta waktu dan biaya yang dibutuhkan.

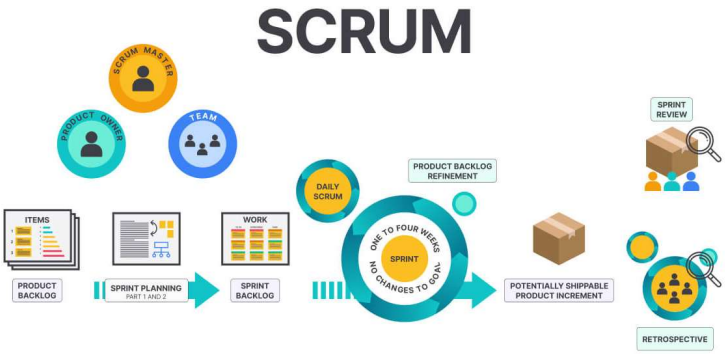
Berikut ini adalah beberapa model pengembangan perangkat lunak yang paling umum digunakan dan faktor-faktor yang perlu dipertimbangkan saat memilih model yang tepat untuk proyek Anda.

Ada banyak model pengembangan *software* yang dapat kita pilih, masing-masing menampilkan dasar-dasar sistem dan pendekatan yang berbeda, antara lain *Waterfall Model, Joint Application Development (JAD), Rapid Application Development (RAD), Information Engineering (IE), Rational Unified Process (RUP), Extreme Programming (XP), Agile Modelling, Scrum* dan lain sebagainya.

Untuk menentukan metode yang paling tepat, kita perlu mempertimbangkan beberapa hal penting dari proyek yang akan kita kerjakan. Tujuan akhir proyek, dana yang akan digunakan, tenggat waktu penyelesaian, tim yang terlibat hingga pendapat dari pengguna adalah faktor yang perlu mendapat perhatian. Semuanya harus bekerja secara terpadu agar proyek dapat berhasil dengan pemilihan metode yang tepat.

Berikut ini model pengembangan *software* yang dapat kita pilih, diantaranya:

1. Scrum Model



Scrum, pada dasarnya merupakan bagian komponen dari metodologi pengembangan sistem Agile .

Akhir-akhir ini Scrum mulai marak diimplementasikan di perusahaan IT di Indonesia, dikarenakan maraknya perusahaan IT mengimplementasikan *agile development.*

Model Scrum adalah sebuah kerangka kerja dalam manajemen proyek yang digunakan untuk pengembangan produk, terutama dalam perangkat lunak. Scrum membantu tim untuk bekerja secara mandiri dan lebih efektif, beradaptasi dengan perubahan, meningkatkan kolaborasi serta mencapai tujuan bersama dengan cara yang efisien untuk menghasilkan produk yang lebih baik.

Berikut adalah beberapa poin penting tentang model Scrum:

- **Tujuan**

  Scrum bertujuan untuk menghasilkan produk yang berkualitas, meningkatkan kepuasan pelanggan, dan mempercepat waktu pemasaran dengan cara mengelola proyek kompleks menjadi tugas-tugas kecil yang dikerjakan dalam siklus pendek yang disebut *sprint*.

- **Pendekatan :**

  Scrum menggunakan pendekatan *iteratif* dan *incremental*, dimana produk dikembangkan sedikit demi sedikit melalui serangkaian *sprint*.

- **Sprint :**

  Proyek dibagi menjadi periode waktu tetap yang disebut *sprint* (biasanya 2-4 minggu), dimana tim menyelesaikan sejumlah pekerjaan yang telah direncanakan.

- **Kerangka Kerja :**

  Scrum menyediakan kerangka kerja yang jelas untuk mengatur proses pengembangan, termasuk peran, acara, dan artefak.

  - Peran :

    Dalam Scrum, terdapat tiga peran utama:

    - *Product Owner :* Bertanggung jawab untuk mewakili kepentingan *stakeholder* dan mengelola *product backlog.*
    - *Scrum Master* : Bertindak sebagai fasilitator untuk tim, membantu menghapus hambatan, dan memastikan penerapan scrum yang efektif.
    - *Development Team* : Tim multidisiplin yang bertanggung jawab untuk menghasilkan produk setiap *sprint*.

  - Acara Scrum :

    Scrum memiliki beberapa acara rutin seperti *sprint planning* (perencanaan *sprint*), *daily scrum* (rapat singkat setiap hari), *sprint review* (peninjauan hasil *sprint*), dan *sprint retrospective* (evaluasi proses *sprint*), yang dirancang untuk meningkatkan transparansi, kolaborasi, dan adaptasi dalam tim.

  - Artefak Scrum :

    Artefak *Scrum* adalah dokumen yang digunakan dan dihasilkan selama pengembangan, seperti *product backlog* (daftar kebutuhan produk), *sprint backlog* (daftar pekerjaan yang akan diselesaikan dalam *sprint*), dan *increment* (produk yang dihasilkan dalam satu *sprint*).

- **Manfaat :**

  - Peningkatan efisiensi

    Scrum membantu tim bekerja lebih efisien dengan memecah proyek menjadi tugas-tugas yang lebih mudah dikelola.

  - Fleksibilitas dan Adaptasi

    Scrum memungkinkan tim untuk merespons perubahan kebutuhan pengguna dan pasar dengan cepat.

  - Peningkatan kualitas produk

    Dengan umpan balik yang berkelanjutan dan proses iterasi, Scrum membantu menghasilkan produk yang lebih berkualitas.

  - Peningkatan kolaborasi

    Scrum mendorong komunikasi yang kuat dan kolaborasi antar anggota tim.

Kelebihan dari Scrum :

- Keperluan berubah dengan cepat.
- Tim berukuran kecil sehingga melancarkan komunikasi, mengurangi biaya dan memberdayakan satu sama lain.
- Pekerjaan terbagi-bagi sehingga dapat diselesaikan dengan cepat.
- Dokumentasi dan pengujian terus menerus dilakukan setelah software dibangun.
- Proses Scrum mampu menyatakan bahwa produk selesai kapanpun diperlukan.

Kelemahan dari Scrum :

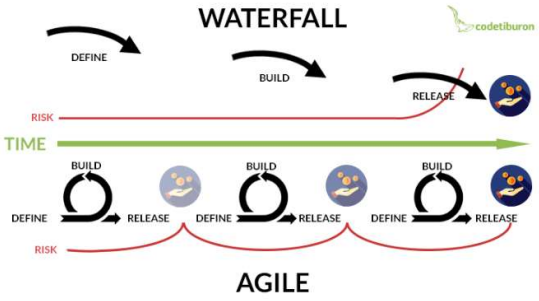
- Developer harus selalu siap dengan perubahan karena perubahan akan selalu diterima.

Kesimpulan : Scrum adalah model yang efektif untuk mengelola proyek terutama dalam pengembangan perangkat lunak dengan cara yang *iteratif* dan *incremental*, memungkinkan tim untuk beradaptasi dengan perubahan dan menghasilkan produk berkualitas tinggi.

Referensi :

- https://id.jobstreet.com/id/career-advice/article/manfaat-metode-scrum-untuk-tim-kerja-lebih-efisien
- https://www.dicoding.com/blog/pengertian-scrum-dan-contoh-kasusnya/
- https://www.dewaweb.com/blog/scrum-methodology-panduan-project-management/
- https://aws.amazon.com/id/what-is/scrum/

2. Agile Development Model



Apa itu *Agile software development* ? Jika nanti kamu bekerja di bidang IT, kamu pasti sering mendengar istilah tersebut.

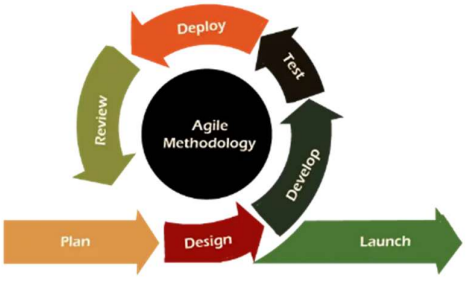
Agile software development merupakan salah satu metode yang digunakan dalam proses pengembangan *software,* baik itu *website, web application,* atau pun *mobile application.*

Agile software development adalah sebuah metode untuk proses pengembangan *software* dengan kualitas yang tinggi namun dengan proses yang ramping.

Agile model, salah satu model pengembangan *software* yang menggunakan urutan kerja *incremental* (berkembang sedikit demi sedikit secara teratur, dengan kolaborasi antar tiap tim secara terorganisir serta terstruktur) dan *iteratif* (proses pengerjaan yang dilakukan secara berulang), yang dikenal dengan istilah *sprint*.

Model Agile ini merupakan pembaharuan dari model *waterfall* dan *spiral*, dimana diyakini dapat mempercepat proses pengembangan suatu proyek karena berlangsung dalam jangka waktu yang pendek dan bertahap.

Metode Agile adalah sebuah pendekatan pengembangan perangkat lunak yang menekankan pada kelincahan, kecepatan, dan adaptasi terhadap perubahan.



Metode ini membagi proyek menjadi iterasi-iterasi kecil yang memungkinkan tim untuk merespons kebutuhan pelanggan dan perubahan pasar dengan lebih efektif. Agile juga mengedepankan kolaborasi tim yang erat, komunikasi terbuka, dan pengujian produk secara berkelanjutan.

Beberapa karakteristik utama dari metode Agile, sebagai berikut:

- **Iteratif dan Incremental**
  Proyek dikembangkan melalui siklus-siklus pendek (iterasi), di mana setiap iterasi menghasilkan produk yang dapat digunakan dan dapat ditingkatkan pada iterasi berikutnya (incremental).
- **Adaptif**
  Agile memungkinkan tim untuk merespons perubahan kebutuhan pelanggan dan perubahan pasar dengan cepat dan fleksibel.
- **Kolaboratif**
  Agile menekankan kolaborasi yang erat antara tim pengembangan dan pelanggan, serta antar anggota tim sendiri.
- **Fokus pada nilai**
  Agile berorientasi pada penyampaian nilai kepada pelanggan secara cepat dan berkelanjutan.
- **Pengujian berkelanjutan**
  Pengujian dilakukan secara teratur sepanjang proses pengembangan untuk memastikan kualitas produk.

Beberapa model yang termasuk dalam Metode Agile diantaranya:

- ***Scrum***
  Kerangka kerja yang populer untuk manajemen proyek Agile yang menekankan pada iterasi-iterasi pendek *(sprint)*.
- ***Kanban***
  Metode visual untuk mengelola aliran kerja yang membantu tim untuk mengidentifikasi dan menghilangkan hambatan.
- ***Extreme Programming (XP)***
  Metode yang menekankan pada praktik-praktik terbaik dalam pengembangan perangkat lunak, seperti pengkodean berpasangan dan pengujian berkelanjutan.
- ***Lean Software Development***
  Metode yang berfokus pada pengurangan pemborosan dan peningkatan efisiensi dalam proses pengembangan.

Penerapan Agile tidak hanya terbatas pada pengembangan perangkat lunak. Metode ini juga dapat diterapkan pada berbagai jenis proyek, seperti pemasaran, desain produk, dan bahkan dalam kehidupan sehari-hari.

Dengan menerapkan prinsip-prinsip Agile, tim dapat bekerja lebih efektif, menghasilkan produk yang lebih baik, dan memberikan nilai tambah kepada pelanggan dengan lebih cepat.

12 Prinsip Utama Agile Development

Pengembangan software menggunakan metode Agile mempunyai 12 prinsip utama. Prinsip inilah yang dikenal sebagai Agile Manifesto dan inilah rinciannya.

1. Lebih menekankan kepuasan pengguna dengan cara merilis produk secara cepat dan bertahap.
2. Selalu terbuka menerima perubahan, meskipun berdampak pada keterlambatan dalam mengembangkan produk.
3. Dapat menghasilkan perangkat lunak yang dapat bekerja dengan baik dalam jangka waktu yang relatif pendek (14 – 60 hari).
4. Dapat menjalin kerja sama yang baik antara pengembang produk dan klien.
5. Membuat suasana yang berisi anggota dengan motivasi yang tinggi. Dengan adanya lingkungan yang mendukung, maka setiap anggotanya akan menyelesaikan pekerjaan dengan baik.
6. Sedapat mungkin melakukan komunikasi secara langsung, karena metode tersebut dinilai lebih efektif untuk menyampaikan informasi.
7. Kemajuan sebuah proyek IT dinilai dari perangkat lunak yang dapat bekerja dengan baik sesuai harapan.
8. Pengembangan perangkat lunak yang berkelanjutan dengan dukungan dari berbagai pihak, seperti pengguna, klien, dan developer.
9. Memiliki keunggulan dari segi teknis adalah hal utama dalam pengembangan perangkat lunak menggunakan metode agile.
10. Kesederhanaan adalah poin utama dalam agile development untuk memaksimalkan sumber daya yang ada.
11. Setiap anggota tim harus mampu untuk mengorganisir diri sendiri.
12. Melakukan refleksi secara berkala mengenai cara bekerja yang lebih efektif dan menyesuaikannya.

Kelebihan dari Agile Development Model

Beberapa kelebihan dari *agile* diantaranya:

- Menambah produktivitas tim,
- Menambah kualitas perangkat lunak,
- Menambah kepuasan klien,
- Menghemat biaya.

Kekurangan dari Agile Development Model

- *Agile* tidak akan berjalan dengan baik jika komitmen tim kurang,
- Tidak cocok dalam skala tim yang besar (> 20 orang),
- Perkiraan waktu *release* dan harga perangkat lunak sulit ditentukan.

Referensi :

- https://www.telkomsel.com/enterprise/insight/blog/metode-agile-adalah
- https://www.dicoding.com/blog/konsep-agile-pada-software-development/
- https://www.binar.co.id/blog/metode-agile-adalah

3. Waterfall Model

Waterfall model adalah pendekatan SDLC paling awal yang digunakan untuk pengembangan perangkat lunak.

Hal ini juga disebut sebagai *model SDLC linear-sequensial* karena setiap tahapan pengembangan dilakukan secara berurutan, seperti air terjun dari atas ke bawah.

Dalam *Model Waterfall,* setiap tahap harus berurutan, dan tidak dapat meloncat ke tahap berikutnya, harus menyelesaikan tahap pertama baru lanjut ke tahap ke dua dst.

Fase-fase dalam Model Waterfall

- ✓ Analisis Kebutuhan *(Requirement Analysis)*: Mengumpulkan dan mendokumentasikan semua kebutuhan sistem yang diperlukan dari pengguna.
- ✓ Desain *(System Design)*: Merancang arsitektur dan spesifikasi sistem berdasarkan kebutuhan yang telah dikumpulkan.
- ✓ Implementasi *(Implementation)*: Mengembangkan kode perangkat lunak sesuai dengan desain yang telah ditentukan.
- ✓ Pengujian *(Testing)*: Memeriksa dan menguji sistem untuk memastikan bahwa semua fungsionalitas bekerja sesuai spesifikasi.
- ✓ Pemeliharaan *(Maintenance)*: Melakukan perbaikan dan pembaruan setelah sistem telah diterapkan.

Karakteristik Utama dari Model Waterfall

1. Pendekatan Linier dan Berurutan

   Proses pengembangan dalam Waterfall Model memiliki urutan fase yang jelas, dimana proses dimulai dari fase analisis kebutuhan, diikuti oleh desain, implementasi, pengujian, dan pemeliharaan.

2. Dokumentasi yang Mendetail

   Setiap fase memerlukan dokumentasi yang lengkap dan mendalam sebelum melanjutkan ke fase berikutnya; yang mencakup spesifikasi kebutuhan, desain sistem, hasil pengujian, dan lain-lain. Dokumentasi ini berfungsi sebagai panduan resmi bagi tim pengembang dan pemangku kepentingan.

3. Kontrol Proyek yang Ketat

   Karena memiliki struktur yang teratur, Waterfall Model memberikan kontrol yang lebih baik terhadap jalannya proyek, terutama dalam hal perencanaan, penganggaran, dan manajemen waktu.

4. Cocok untuk Proyek dengan Persyaratan Stabil

   Model ini paling efektif digunakan pada proyek yang memiliki kebutuhan yang sudah jelas, terdefinisi dengan baik, dan tidak berubah selama masa pengembangan.

Kelebihan dari Waterfall Model

Keuntungan dari Waterfall Model adalah jadwal dapat diatur dengan tenggang waktu untuk setiap tahap pengembangan dan produk dapat dilanjutkan melalui proses pengembangan model fase satu per satu. Pembangunan bergerak dari konsep, melalui desain, implementasi, pengujian, instalasi, pemecahan masalah, dan berakhir di operasi dan pemeliharaan.

Berikut keuntungan lainnya dari Waterfall Model:

- Simple, mudah dimengerti dan diimplemetasikan
- Mudah untuk mengelola karena model yang sederhana. Setiap fase memiliki spesifik requirement dan proses review diproses dan diselesaikan satu per satu.
- Cocok untuk project skala kecil dimana kebutuhan project dapat mudah dimengerti.
- Jelas dalam mendefinisikan setiap tahap.
- Mudah menentukan pencapaian suatu sistem.
- Mudah dalam menentukan tugas setiap individu.
- Proses pendokumentasian lebih mudah.

Kekurangan dari Waterfall Model

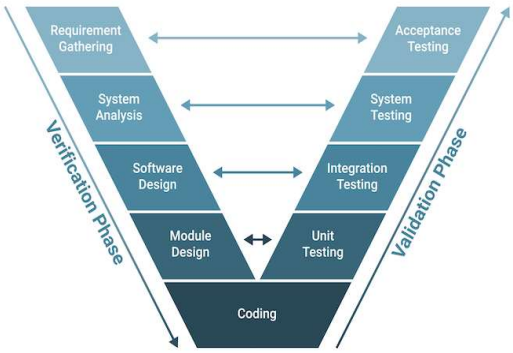
Kerugian dari *Waterfall Model* adalah tidak memungkinkan banyak refleksi atau revisi. Setelah aplikasi dalam tahap pengujian, sangat sulit untuk kembali dan mengubah sesuatu yang tidak terdokumentasi dengan baik atau pikiran pada dalam tahap konsep. Berikut kerugian lainya dari Waterfall Model:

- Aplikasi yang dihasilkan cenderung lama karena *step-step* tidak dapat dilompati.
- Resiko yang tinggi karena proses nya terlalu lama.
- Tidak cocok untuk project yang terlalu kompleks dan *Object Oriented Projects.*
- Tidak cocok untuk project jangka lama dan untuk *project* yang sedang berjalan.
- Tidak cocok untuk project yang mudah berganti-ganti model proses.
- Sulit untuk mengukur kemajuan dalam tahap.

Referensi :

- https://binus.ac.id/bekasi/2024/08/model-pengembangan-sistem-waterfall-buat-kasus-apa-yang-tepat
- https://www.geeksforgeeks.org/software-engineering/waterfall-model/
- https://www.biznetgio.com/news/apa-itu-metode-waterfall

4. V shape - Model



Model ini merupakan perluasan dari model *waterfall* dimana setiap tahap harus dikerjakan berurutan. Model ini adalah model pengembangan perangkat lunak yang sangat disiplin dan memiliki metode pengujian pada setiap tahapnya.

V-model, menekankan pada pengujian yang paralel dengan tahapan pengembangan, dengan visualisasi berbentuk huruf **V - shape**.

V-model dikenal juga sebagai model verifikasi dan validasi, karena sisi kiri V mewakili tahapan pengembangan *(verification)*, yakni memastikan bahwa produk dibuat sesuai dengan spesifikasi, seperti: analisis kebutuhan, desain sistem, desain modul, dan

pengkodean. Sedangkan sisi kanan V menggambarkan proses pengujian *(validation),* yakni memastikan bahwa produk memenuhi kebutuhan pengguna, seperti: proses pengujian penerimaan, pengujian sistem, pengujian integrasi, dan pengujian unit.

Langkah-langkah Merancang Sistem dengan V-Shape Model

1. Analisis Kebutuhan *(Requirement Analysis)*

   Pada fase ini, kita mengumpulkan dan menganalisis kebutuhan dari pengguna dan membuat spesifikasi kebutuhan perangkat lunak. Semua kebutuhan sistem harus jelas dan mendetail agar desain yang dibuat dapat memenuhi ekspektasi pengguna.

2. Desain Sistem *(System Design)*

   Setelah kebutuhan dikumpulkan, fase berikutnya adalah mendesain (merancang) arsitektur sistem secara keseluruhan. Pada tahap ini, tim pengembang menentukan bagaimana sistem akan beroperasi, termasuk pemilihan teknologi dan platform yang akan digunakan.

3. Desain Detail *(Detailed Design)*

   Fase ini mendesain modul-modul perangkat lunak secara rinci. Setiap modul atau komponen yang ada dalam sistem dijelaskan lebih rinci agar pengembang dapat melanjutkan ke tahap pengkodean dengan panduan yang jelas.

4. Pengkodean *(Coding)*

   Selanjutnya adalah tempat implementasi desain, yaitu menerjemahkan desain menjadi kode program *(source code).* Pengembangan kode program dilakukan sesuai dengan desain detail yang telah disetujui.

5. Pengujian Unit *(Unit Testing)*

   Setelah pengkodean selesai, pengujian unit dilakukan untuk memastikan bahwa setiap unit atau modul perangkat lunak berfungsi dengan baik secara terpisah.

6. Pengujian Integrasi *(Integration Testing)*

   Setelah setiap unit diuji, fase berikutnya adalah pengujian integrasi, dimana unit-unit yang telah dikembangkan diuji bersama untuk memastikan kompatibilitas dan interaksi antar unit / modul perangkat lunak.

7. Pengujian Sistem *(System Testing)*

   Pengujian sistem dilakukan untuk memastikan seluruh sistem perangkat lunak berfungsi sesuai dengan spesifikasi yang telah ditentukan sebelumnya. Fase ini bertujuan untuk memvalidasi bahwa sistem bekerja sebagaimana mestinya.

8. Pengujian Penerimaan *(Acceptance Testing)*

   Fase ini menguji perangkat lunak di lingkungan pengguna untuk memastikan sesuai dengan kebutuhan.

   Setelah sistem diterapkan dan diuji, tahap terakhir adalah fase pemeliharaan. Pemeliharaan melibatkan perbaikan bug, pembaruan perangkat lunak, dan penyesuaian dengan perubahan kebutuhan.

Fungsi Utama Model - V

- ✓ Meningkatkan Kualitas Produk

  Dengan perencanaan pengujian yang terintegrasi sejak awal, model V membantu mengidentifikasi dan memperbaiki kekurangan di setiap tahap, sehingga menghasilkan produk yang lebih berkualitas.

✓ Deteksi Masalah Dini

Tahapan pengembangan dan pengujian dilakukan secara berurutan, keduanya juga berjalan secara paralel. Setiap fase pengembangan memiliki fase pengujian yang sesuai sehingga masalah dapat dideteksi dan diatasi pada tahap awal, mengurangi resiko dan biaya perbaikan di kemudian hari.

✓ Proses yang Terstruktur

Model V menyediakan kerangka kerja yang jelas dan terstruktur, yang membuat proses pengembangan lebih mudah dipahami, dikelola, dan dilacak kemajuannya.

Kelebihan dari V-Model

- Model ini menawarkan alur yang jelas dari analisis hingga pengujian, yang memudahkan tim pengembang untuk mengikuti setiap langkah.
- Dengan adanya pengujian yang dilakukan di setiap tahap pengembangan, masalah dan kesalahan dapat ditemukan lebih cepat, mengurangi biaya dan waktu yang dibutuhkan untuk memperbaikinya.
- Bekerja dengan baik untuk proyek-proyek yang lebih kecil dimana persyaratan dipahami dengan baik.
- Prosesnya jelas dan terstruktur, mudah dimengerti dan mudah digunakan.

Kekurangan dari V-Model

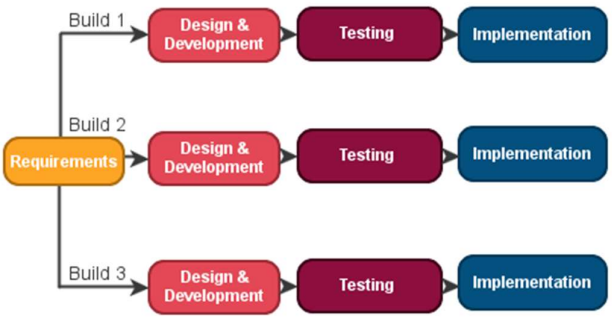
- Kurang efektif untuk proyek-proyek besar dengan kompleksitas tinggi.
- Tidak cocok untuk proyek-proyek dimana persyaratan beresiko tinggi.
- Tidak cocok untuk proyek-proyek yang lama dan berkelanjutan.
- Setelah aplikasi dalam tahap pengujian, sulit untuk kembali dan mengubah fungsionalitas.

Kesimpulan : V shape - Model adalah salah satu pendekatan yang baik untuk pengembangan perangkat lunak, terutama untuk proyek-proyek dengan persyaratan yang jelas dan stabil, serta proyek-proyek yang membutuhkan jaminan kualitas yang tinggi, seperti dalam sistem keselamatan kritis *(safety-critical systems)* di bidang medis, penerbangan, pertahanan atau sistem keamanan tinggi.

Referensi :

- https://www.konsepkoding.com/2022/03/pengertian-model-v-tahapan-kelebihan.html
- https://unydevelopernetwork.com/index.php/2020/02/07/belajar-sdlc-mengenal-v-model-kelebihan-kekurangannya/
- https://t2informatik.de/en/smartpedia/v-model/
- https://kd-cibiru.upi.edu/index.php/component/content/article/model-pengembangan-perangkat-lunak-pemilihan-yang-tepat-untuk-proyek-anda
- https://binus.ac.id/bekasi/2024/12/bagaimana-merancang-sistem-menggunakan-v-shape-model/

5. *Iterative Model*



Iterative adalah melakukan sesuatu secara berulang-ulang, biasanya untuk memperbaiki = proses perulangan.

Iterative development model merupakan suatu pendekatan pengembangan perangkat lunak yang memecah proses pengembangan aplikasi besar menjadi bagian-bagian kecil.

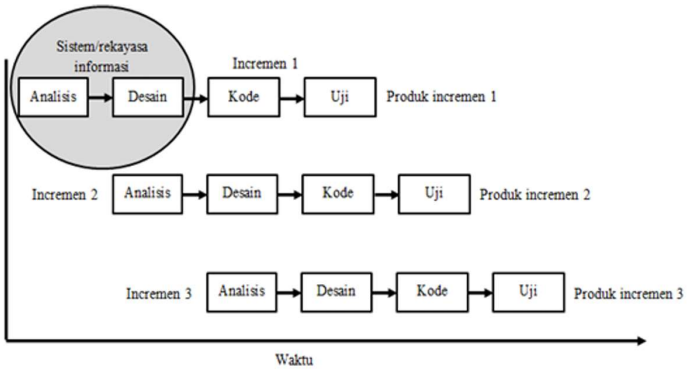
Setiap bagian-bagian kecil ini menjadi siklus-siklus kecil atau iterasi yang berulang. "Iterasi" ini mewakili proses pengembangan keseluruhan serta berisi langkah-langkah perencanaan, desain, pengembangan dan pengujian, yang kemudian diselesaikan secara berulang hingga produk akhir tercapai.

Model iteratif lebih terlihat sebagai proses yang berputar dibanding sebagai tahapan proses selangkah demi selangkah yang kaku.

Saat fase perencanaan awal selesai, beberapa tahapan lain diulang, sehingga menghasilkan sebuah siklus. Setiap satu siklus selesai, perangkat lunak diperbaiki dan diiterasi.

Setiap iterasi menghasilkan versi produk yang lebih baik dari versi sebelumnya, yang memungkinkan identifikasi dan perbaikan masalah sejak dini. Model ini sangat ideal untuk proyek dengan persyaratan yang sering berubah atau tidak pasti.

Langkah-langkah dari Model Iteratif



1. Perencanaan dan Analisis

   Proyek dibagi menjadi bagian-bagian yang lebih kecil, dan persyaratan untuk bagian tersebut dikumpulkan dan dianalisis.

2. Perancangan

   Desain awal dibuat untuk bagian yang akan dikembangkan pada iterasi ini.

3. Pengembangan

   Versi awal dari bagian tersebut dikembangkan dan dibuat prototipe.

4. Pengujian dan Umpan Balik

   Prototipe atau versi yang dikembangkan diuji untuk mengidentifikasi masalah dan mengumpulkan umpan balik dari pengguna dan pemangku kepentingan.

5. Penyempurnaan

   Berdasarkan umpan balik, versi produk disempurnakan dan ditingkatkan.

6. Iterasi berikutnya

   Proses ini kemudian diulang kembali dengan merancang, mengembangkan, menguji, dan menyempurnakan versi produk menjadi lebih baik lagi.

Bagaimana Model Iteratif Bekerja ?

1. Pecah Proyek

   Proyek besar dipecah menjadi bagian-bagian yang lebih kecil dan terkelola.

2. Siklus Berulang (Iterasi)

Setiap bagian ini dikerjakan melalui serangkaian langkah yang berulang, seperti perencanaan, analisis, desain, pengembangan, dan pengujian untuk menghasilkan bagian fungsional dari produk.

3. Produk Berfungsi

Setiap iterasi menghasilkan versi produk yang berfungsi, meskipun mungkin belum lengkap.

4. Umpan Balik & Penyempurnaan

Setelah setiap iterasi, tim mengumpulkan umpan balik dari pemangku kepentingan dan menggunakannya untuk menyempurnakan dan memperbaiki produk pada iterasi berikutnya.

5. Peningkatan Berkelanjutan

Proses ini berulang sampai produk akhir memenuhi semua persyaratan dan mencapai kualitas optimal.

Karakteristik Utama Model Iteratif

- Berulang:

  Aktivitas pengembangan diulang secara sistematis dalam siklus yang dikenal sebagai iterasi.

- Progresif:

  Produk disempurnakan selangkah demi selangkah, di mana setiap versi dibangun di atas versi sebelumnya.

- Fleksibel:

  Memungkinkan adaptasi dan perubahan yang lebih cepat sebagai respons terhadap perubahan keadaan atau umpan balik.

- Kolaborasi:

  Melibatkan kolaborasi rutin antara tim pengembangan dan pemangku kepentingan untuk mendapatkan umpan balik.

Manfaat Model Iteratif

- ✓ Peningkatan kualitas

  Memungkinkan perbaikan dan peningkatan berkelanjutan pada produk, yang terus disempurnakan di setiap langkah.

- ✓ Adaptif

  Lebih responsif terhadap perubahan kebutuhan dan memberikan fleksibilitas yang lebih besar, ini memungkinkan penyesuaian terhadap perubahan persyaratan atau kondisi.

- ✓ Pengurangan Risiko

  Dengan membangun produk secara bertahap, dapat membantu mengidentifikasi dan menyelesaikan risiko lebih awal dalam proses pengembangan sehingga kesalahan atau kegagalan besar dapat dikurangi.

- ✓ Kolaborasi yang ditingkatkan

  Ada kolaborasi rutin antara tim pengembangan dan pemangku kepentingan, yang meningkatkan efektivitas.

- ✓ Produk berfungsi lebih cepat

  Versi produk yang berfungsi dirilis lebih awal, memungkinkan pemangku kepentingan memberikan umpan balik lebih cepat.

Contoh Penggunaan

- Pengembangan perangkat lunak

  Tim dapat membangun fungsionalitas dasar terlebih dahulu, lalu menambahkan fitur dan menyempurnakan versi yang ada di setiap iterasi berikutnya.

- Desain produk

  Desainer dapat membuat prototipe kecil, mendapatkan masukan dari pengguna, dan kemudian berulang kali menyempurnakan desain berdasarkan umpan balik tersebut.

- Peningkatan kualitas

  Tim dapat menguji berbagai metode iklan, menganalisis hasilnya, dan kemudian menyempurnakan strategi mereka berdasarkan data yang terkumpul.

Berbeda dengan model waterfall yang bersifat linier, model iteratif lebih fleksibel dan memungkinkan perubahan di berbagai tahap proyek, sehingga cocok untuk proyek dengan persyaratan yang berubah-ubah atau tidak pasti.

Kelebihan dari Iterative Model :

- Fleksibel terhadap perubahan : memungkinkan tim untuk menyesuaikan diri dengan perubahan persyaratan kapan saja.
- Deteksi dini masalah : kesalahan dan kekurangan dapat ditemukan dan diperbaiki lebih awal dalam siklus pengembangan.n
- Umpan balik konstan : keterlibatan pelanggan yang terus-menerus memastikan produk akhir sesuai dengan harapan mereka.
- Pengiriman cepat : produk yang berfungsi dapat dikirimkan secara bertahap, memungkinkan pelanggan untuk menggunakan bagian-bagian yang sudah selesai.

Kekurangan dari Iterative Model :

- Membutuhkan perencanaan yang kuat : model ini menuntut perencanaan dan manajemen yang cermat agar tidak terjadi perluasan cakupan proyek (scope creep).
- Sumber daya lebih banyak : Mungkin membutuhkan lebih banyak sumber daya, terutama dalam hal tenaga ahli untuk analisis.
- Progres sulit diprediksi : Menentukan jumlah iterasi yang dibutuhkan dan memperkirakan tanggal penyelesaian proyek bisa jadi sulit.
- Tidak cocok untuk proyek kecil : Proses ini mungkin terlalu rumit untuk proyek yang berskala kecil.

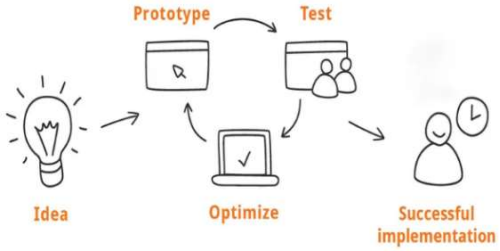
Contoh:

Dalam pengembangan perangkat lunak, model iteratif dapat digunakan untuk membangun aplikasi web dengan fitur bertahap. Iterasi pertama mungkin hanya fokus pada fungsi dasar, kemudian iterasi berikutnya menambahkan fitur yang lebih kompleks. Umpan balik dari pengguna setelah setiap iterasi membantu pengembang untuk terus menyempurnakan aplikasi tersebut.

Referensi :

- https://www.konsepkoding.com/2022/03/pengertian-metode-iterative-fitur-kelebihan-kekurangan.html
- https://glints.com/id/lowongan/iterative-development-model/
- https://kd-cibiru.upi.edu/index.php/component/content/article/model-pengembangan-perangkat-lunak-pemilihan-yang-tepat-untuk-proyek-anda?
- https://asana.com/id/resources/iterative-process
- https://prakom.banjarmasinkota.go.id/2021/10/model-iteratif-sdlc.html
- https://csirt.teknokrat.ac.id/pengembangan-software-lebih-cepat-dengan-model-iteratif/

6. *Prototype Model*



Prototype adalah suatu model awal atau rancangan awal yang dibuat sebagai representasi visual atau fisik dari barang, sistem, atau aplikasi yang sedang dibuat.

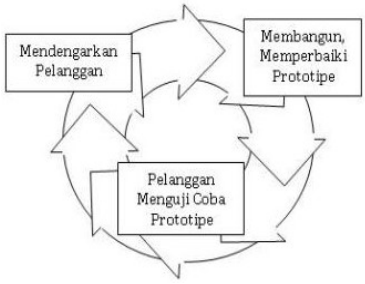
Ini dibuat untuk menguji konsep juga ide-ide baru, serta untuk mengumpulkan umpan balik dari pengguna atau konsumen.

Prototyping telah digunakan oleh banyak industri ; Sebelum memulai membangun sebuah bangunan, arsitek harus menggambarkan *blueprint* dari bangunan dan membuat model dari bangunan. Perusahaan pesawat terbang juga harus merancang sebuah *prototype* dari desain pesawat sebelum mulai membuatnya. Perusahaan yang bergerak di bidang *software*, juga membuat *prototype software* untuk menjelajahi ide sebelum memulai pengembangan aplikasi.

Dalam pengembangan perangkat lunak, *prototipe* biasanya dibuat untuk menguji fungsionalitas dan juga antarmuka pengguna. Hal ini sangat penting untuk memastikan bahwa aplikasi tersebut dapat berfungsi dengan baik dan memenuhi kebutuhan pengguna.

Prototipe dapat berbentuk visual, seperti sketsa tangan atau *mockup* digital, ataupun berupa model fisik atau jenis mekanik.

Jenis visual biasanya digunakan untuk menguji antarmuka pengguna, *layout*, dan desain grafis, sedangkan jenis fisik digunakan dengan tujuan menguji kemampuan serta fungsionalitas hasil akhir produk.



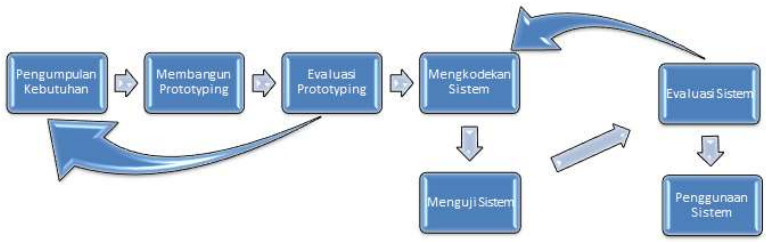
Dalam fase pengembangan aplikasi, *prototipe* sering kali dibuat serta diuji beberapa kali untuk memastikan bahwa aplikasi tersebut dapat berfungsi dengan baik serta memenuhi kebutuhan pengguna. Setelah beberapa iterasi, tahap akhir akan digunakan sebagai dasar untuk produksi massal barang.

Selain untuk pengembangan aplikasi, *prototipe* juga dapat digunakan untuk mempresentasikan ide serta konsep kepada investor atau *client*. Dalam hal ini, *prototipe* digunakan untuk membantu menjelaskan ide serta visi produk secara visual dan interaktif, sehingga memudahkan pemahaman serta pengambilan keputusan.

Secara umum, model awal ini sangat penting dalam pengembangan aplikasi. Dengan membuat *prototipe*, tim pengembang dapat menguji dan mengevaluasi ide-ide baru dengan efisien, serta memperbaiki masalah atau kekurangan yang muncul sejak awal.

Hal ini membantu mengurangi risiko pengembangan seperti kesalahan desain dan kegagalan aplikasi, yang masih dapat diperbaiki sebelum penerapan keseluruhan sehingga meningkatkan kesuksesan hasil pada akhirnya.

Fase - Fase dalam *Model Prototype*



1. Analisa kebutuhan

   Di tahap ini pengembang melakukan identifikasi *software* dan semua kebutuhan sistem yang akan dibuat.

2. Membangun *prototyping*

   Membangun *prototyping* dengan membuat perancangan sementara yang berfokus pada penyajian kepada pengguna (misalnya dengan membuat *input* dan format *output*).

3. Evaluasi *prototyping*

   Evaluasi ini dilakukan untuk mengetahui apakah *prototyping* sudah sesuai dengan harapan pengguna.

4. Mengkodekan sistem

   Pada tahap ini *prototyping* yang sudah disetujui akan diubah ke dalam bahasa pemrograman.

5. Menguji sistem

   Di tahap ini dilakukan untuk menguji sistem perangkat lunak yang sudah dibuat.

6. Evaluasi Sistem

   Perangkat lunak yang sudah siap jadi akan dievaluasi oleh pengguna untuk mengetahui apakah sistem sesuai dengan yang diharapkan.

7. Menggunakan sistem

   Perangkat lunak yang sudah diuji dan disetujui oleh pengguna siap diimplementasikan.

Tujuan *Model Prototype*

Adapun tujuan utama dari penggunaan *Model Prototype* adalah sebagai berikut:

- ☑ Mengembangkan model atau rancangan suatu produk / sistem / aplikasi menjadi suatu produk final yang sesuai dengan permintaan pengguna.
- ☑ Menguji rancangan proses kerja atau konsep produk sebelum dipublikasikan.
- ☑ Memberikan gambaran kepada pengguna prakiraan wujud dari produk/sistem/ aplikasi yang akan dikembangkan sesuai konsep. Dengan demikian, pengguna dapat lebih memahami dan melihat bagaimana produk/sistem/aplikasi bekerja, apa fungsinya dan bagaimana pengguna harus berinteraksi dengannya.
- ☑ Terjalinnya interaksi antara pengembang dan pengguna, sehingga terciptanya dialog dan umpan balik terkait rancangan yang dibuat untuk menentukan hasil yang terbaik.

Jenis-Jenis *Prototype*

1. ***Low-Fidelity Prototype*** (Prototipe Kasar)

   Prototipe jenis ini memiliki tingkat detail yang minim, dibuat dengan cepat, dan biasanya hanya menggunakan tampilan dan interaksi yang sederhana.

   Tujuan utama dari prototipe ini adalah menyampaikan konsep dan struktur dasar produk dengan cepat tanpa memperhatikan terlalu banyak detail visual atau interaksi yang kompleks.

   Umumnya, *low-fidelity prototype* menggunakan sketsa dasar, potongan kertas, atau alat desain sederhana seperti *wireframe*.

2. ***High-Fidelity Prototype*** (Prototipe Lengkap)

Kebalikan dari *low-fidelity prototype*, prototipe ini mencakup elemen visual yang lebih realistis, memiliki desain dan antarmuka yang lebih rapi, dalam menampilkan fungsi asli produk yang akan dikembangkan serta interaksi akhir produk yang lebih kompleks.

Prototipe ini dibuat dengan menggunakan alat desain, seperti *Figma, Adoba XD, Sketch,* dan mencakup elemen visual lebih lengkap, seperti warna, tipografi, dan gambar.

3. ***Interactive Prototype***

Interactive Prototype adalah jenis yang memungkinkan pengguna berinteraksi dengan elemen-elemen antarmuka pengguna dalam desain.

Model ini mencakup navigasi antarmuka pengguna, animasi, dan *respons* interaktif lainnya yang mensimulasikan interaksi, alur pengguna, dan fungsionalitas produk.

4. ***Responsive Prototype***

Responsive prototype adalah model yang menyesuaikan tampilan dan fungsionalitas dengan berbagai ukuran layar dan perangkat.

Prototipe ini memungkinkan pengujian tampilan, responsivitas, dan interaksi yang optimal pada berbagai jenis perangkat.

Keberadaan *responsive prototype* sangat penting dalam desain responsif yang bertujuan untuk memastikan pengalaman pengguna yang konsisten di berbagai *platform*, seperti *mobile*, tablet, dan *desktop*.

5. ***Simulative Prototype***

Terakhir, *simulative prototype* adalah model yang menggunakan perangkat lunak atau alat khusus yang dirancang untuk mensimulasikan interaksi atau pengalaman tertentu.

Contohnya, dalam desain ***UI/UX*** untuk aplikasi *mobile*, alat *prototyping* dapat memungkinkan simulasi gerakan atau sentuhan pada layar. *Simulative prototype* berguna untuk menguji dan memvalidasi aspek khusus yang berhubungan dengan pengalaman pengguna yang unik atau situasi tertentu.

Dengan menggunakan jenis *prototype* ini, tim pengembang dapat mendapatkan wawasan lebih mendalam tentang bagaimana produk akan berkinerja dalam situasi yang spesifik atau dengan fitur-fitur khusus.

Kelebihan Prototype Model :

- ✓ Pelanggan berperan aktif dalam proses pengembangan sistem.
- ✓ Pengembang lebih mudah mengetahui produk yang diharapkan pelanggan, karena sesuai dengan kebutuhan pelanggan.
- ✓ Komunikasi yang baik antara pengembang dan pelanggan.
- ✓ Analisa kebutuhan lebih mudah diwujudkan.
- ✓ Mempersingkat waktu pengembangan produk perangkat lunak.
- ✓ Penerapan menjadi lebih mudah karena pelanggan mengetahui apa yang diharapkannya.

Kekurangan Prototype Model :

- ✓ Proses yang dilakukan untuk analisis dan perancangan terlalu singkat.
- ✓ Kurang fleksibel jika terjadi perubahan.

- ✓ Walaupun pemakai melihat berbagai perbaikan dari setiap versi *prototype*, tetapi pemakai mungkin tidak menyadari bahwa versi tersebut dibuat tanpa memperhatikan kualitas dan pemeliharaan jangka panjang.

Contoh Penerapan *Prototype Model*

Metode prototype dapat digunakan di berbagai bidang, termasuk pengembangan perangkat lunak, perangkat keras, desain antarmuka pengguna, dan bahkan dalam proses pengembangan produk fisik.

Berikut adalah beberapa contoh penerapan *prototype model* dalam berbagai bidang :.

- Pengembangan Aplikasi Mobile

  Tim pengembang perangkat lunak menciptakan *prototype* aplikasi *mobile* dengan fitur dasar yang mencerminkan konsep dan antarmuka pengguna yang direncanakan. *Prototype* ini kemudian diperlihatkan kepada pengguna untuk mendapatkan umpan balik tentang tampilan, navigasi, dan fitur-fitur yang diinginkan sebelum mengembangkan aplikasi secara keseluruhan.

- Desain Antarmuka Pengguna (UI/UX)

  Dalam desain antarmuka pengguna, tim desainer dapat membuat prototipe interaktif untuk menunjukkan bagaimana pengguna akan berinteraksi dengan antarmuka. Prototipe ini memungkinkan desainer untuk menguji desain, navigasi, dan respons antarmuka secara lebih mendalam.

- Perangkat Keras (Hardware)

  Dalam pengembangan perangkat keras, prototipe fisik dapat dibuat untuk menguji fungsi dan kinerja perangkat sebelum memproduksi perangkat secara massal. Prototipe ini memungkinkan pengujian dan penyesuaian desain perangkat sebelum diproduksi secara massal.

- Sistem Informasi atau Aplikasi Bisnis

  Dalam pengembangan sistem informasi atau aplikasi bisnis, prototipe dapat digunakan untuk menunjukkan alur kerja, fitur, dan fungsionalitas dasar dari sistem. Prototipe ini membantu dalam memahami dan memvalidasi kebutuhan bisnis sebelum mengembangkan sistem secara menyeluruh.

- Desain Produk Fisik

  Dalam pengembangan produk fisik, seperti perangkat elektronik, mobil, atau peralatan rumah tangga, prototipe dapat digunakan untuk menguji desain dan fungsi produk sebelum produksi massal. Prototipe ini membantu dalam mengidentifikasi masalah desain dan meningkatkan kualitas produk.

- Pengembangan Permainan

  Dalam pengembangan permainan, prototipe dapat digunakan untuk menciptakan tingkat atau mekanisme permainan yang berfungsi sebagai percobaan awal. Prototipe ini memungkinkan tim pengembang untuk menguji gameplay, kesulitan, dan alur permainan sebelum menciptakan permainan secara keseluruhan.

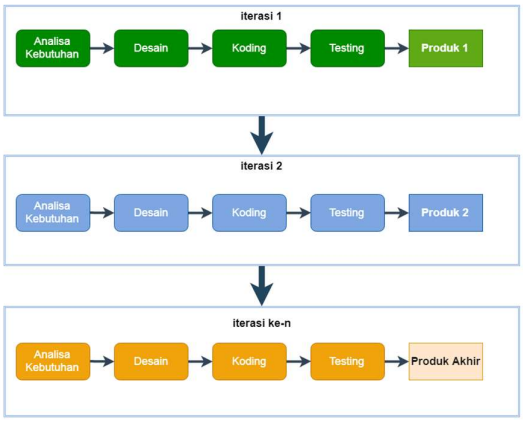
- Desain Web dan E-commerce

  Dalam pengembangan situs web dan *platform e-commerce,* prototipe dapat digunakan untuk menunjukkan tampilan halaman, tata letak, dan alur kerja. Prototipe ini membantu dalam mendapatkan umpan balik dari pengguna tentang pengalaman pengguna sebelum meluncurkan situs atau platform.

Referensi :

- https://sis.binus.ac.id/2022/01/10/prototype-dalam-pengembangan-sistem/
- https://www.dicoding.com/blog/apa-itu-prototype-kenapa-itu-penting/
- https://bsi.today/metode-prototype/
- https://binus.ac.id/bekasi/2024/11/model-prototype-pada-sdlc-pendekatan-inovatif-dalam-pengembangan-sistem/
- https://www.konsepkoding.com/2022/03/pengertian-metode-prototype-tahapan-lengkap.html
- https://www.sekawanmedia.co.id/blog/apa-itu-prototype/
- https://medium.com/@efrenkun123/prototyping-dan-penerapannya-1d6041e65a82
- https://www.ekrut.com/media/prototype
- https://bee.telkomuniversity.ac.id/apa-itu-prototype/
- https://nextgen.co.id/tahapan-metode-prototype-pengertian-metode-prototype-dan-contohnya/

7. Incremental Model



Merupakan kombinasi *"linear sequential model (waterfall model)"* diaplikasikan secara berulang dengan filosofi pengulangan dari *prototyping model* dimana proses pengembangan terus diulangi *(iteration)* untuk mencapai suatu tambahan *(increment)* pada setiap hasil produknya sampai produk memenuhi seluruh kebutuhan atau dinyatakan selesai.

Incremental model adalah sebuah pendekatan dalam pengembangan sistem (terutama perangkat lunak) yang memecah sistem kompleks menjadi bagian-bagian atau modul yang lebih kecil, lalu mengembangkan dan mengirimkan setiap bagian secara bertahap (berulang-ulang) sampai sistem yang lengkap dan berfungsi tercapai.

Modul akan dibagi menjadi beberapa bagian yang mencakup analisis, desain, *coding*, uji coba, verifikasi, implementasi, dan juga pemeliharaan.

Setiap merilis tahapan, akan diberikan fungsi ke rilis tahapan sebelumnya, dan berlanjut hingga seluruh sistem berfungsi dengan lengkap.

Setiap tahap pengembangan inilah disebut increment, dan setiap increment membangun di atas increment sebelumnya dengan menambahkan fungsionalitas baru.

Dengan menggunakan *incremental model,* tim *developer* akan menyelesaikan setiap tahap dengan cepat. Hal ini bertujuan agar menghasilkan produk yang fungsional secara bertahap.

Tahapan dalam *Incremental Model*

Incremental model adalah proses yang dapat mengidentifikasi kecacatan lebih dini ketika proses pengembangan berlangsung. Model ini memiliki empat tahapan utama dalam eksekusinya. Perhatikanlah ilustrasi *incremental model* diagram berikut ini.

1. *Requirement Analysis*

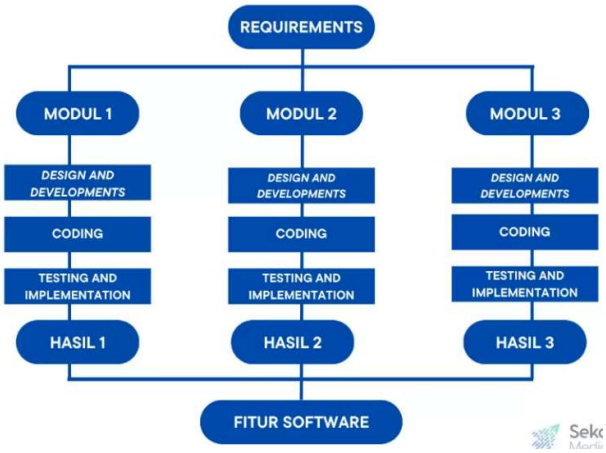
Pada bagan proses model *incremental* di atas, tahapan pertama yang harus dilakukan adalah *requirement analysis* atau analisis kebutuhan.

Tahapan ini merupakan fase yang paling penting dalam pengembangan perangkat lunak dengan model incremental. Tanpa *requirement analysis*, tim

developer tidak akan mengetahui apa saja yang dibutuhkan dalam proses pengembangan.

Pada tahapan ini, *product analyst* akan mengumpulkan kebutuhan dan mengidentifikasi persyaratan-persyaratannya, termasuk tujuan dan kendalanya. Selain itu, tahapan ini juga mencakup identifikasi sumber daya yang dibutuhkan untuk menyelesaikan proyek.

Setelah semua persyaratan terkumpul, maka dokumen akan diolah menjadi *Software Requirement Specification (SRS)* yang akan dikirim kepada klien untuk disetujui.



2. Design and Development

Pada tahapan *design*, sebuah tim *developer* akan mengembangkan desain sistem untuk pengerjaan proyek yang mengacu pada SRS sebelumnya.

Setelah itu, dokumen akan disimpan dalam *Design Document Specification* (DDS) yang akan dikirimkan oleh stakeholder.

3. Coding

Tahapan ini mengacu pada DDS setelah disetujui. Tim *developer* akan melakukan *coding* yang efisien untuk mengatur interaksi dengan modul lainnya. Pada tahapan ini juga memungkinkan untuk melakukan desain secara fisik.

4. Testing and Implementation

Setelah fase *coding*, tim developer akan melakukan *testing* untuk mengetahui efektifitas kode tersebut. *Testing* ini meliputi uji unit atau integrasi aplikasi dan *environment testing*. Tujuannya adalah untuk meminimalisir *bug* dan *error* pada tahap selanjutnya.

Setelah *testing* dilaksanakan, maka akan dilakukan fase implementasi untuk merilis produk. Pada tahap ini, seluruh modul akan digabungkan untuk mendapatkan hasil akhir produk yang dapat digunakan.

Tipe Incremental Model

Model ini memiliki dua tipe dalam SDLC sebagai berikut.

1. Staged Delivery Model

Tipe model ini adalah dengan pengiriman bertahap. Artinya, dalam satu waktu hanya ada satu proyek yang dibangun. Setiap tahap akan dibangun berdasarkan dengan rilis tahap sebelumnya.

2. Parallel Delivery Model

Berbeda dengan tipe sebelumnya, tipe ini mengerjakannya dengan pengiriman paralel. Pada tipe ini, beberapa *sub* yang berbeda akan dikerjakan secara bersamaan dengan sumber daya yang cukup, sehingga dapat menghemat waktu dalam proses pengembangan.

Kapan Harus Menggunakan Incremental Model ?

Jika menimbang kekurangannya, *incremental model* mungkin tidak cocok untuk pengembangan *software* tertentu. Berikut ini waktu yang tepat menggunakan model ini.

- ✓ Ketika mengembangkan aplikasi *web* atau proyek dengan teknologi baru
- ✓ Saat seluruh *requirement* diketahui di awal proses
- ✓ Ketika klien menuntut produk segera diluncurkan
- ✓ Fitur yang dikembangkan memiliki risiko tinggi
- ✓ Ketika durasi pengerjaan proyek sudah terlalu lama

Sebelum menggunakan model ini, kita harus memperhatikan kelebihan dan kekurangan yang dimiliki oleh *incremental model.* Berikut ini beberapa hal yang harus dipertimbangkan.

Kelebihan Incremental Model :

Model ini menawarkan banyak kelebihan dalam pengelolaan risiko dan peningkatan efisiensi pada pengembangan perangkat lunak, diantaranya :

- ☑ Kebutuhan dan persyaratan lebih mudah disesuaikan karena pengerjaannya secara bertahap.
- ☑ Identifikasi dan perbaikan *bug* akan lebih mudah, karena model ini akan melakukan pengujian secara bertahap setiap merilis software.
- ☑ Lebih fleksibel, karena dikerjakan dalam beberapa tahap.
- ☑ Client dapat dengan mudah memonitor progres dari proyek tersebut, sehingga akan lebih mudah untuk menyesuaikan dengan kebutuhan mereka.
- ☑ Meminimalisir risiko kegagalan secara keseluruhan, karena setiap modul akan dikerjakan secara terpisah.
- ☑ Menghasilkan produk yang memiliki fitur lengkap karena adanya penyempurnaan berkala sesuai dengan kebutuhan pengguna.
- ☑ Dapat menghemat sumber daya, karena pengembangan *software* dengan model ini tidak memerlukan jumlah tim yang banyak.

Kekurangan Incremental Model :

Selain memiliki banyak kelebihan, model ini memiliki beberapa kekurangan yang harus diperhatikan diantaranya :

- ☑ Memerlukan perencanaan yang baik agar tidak terjadi gangguan serius pada tahap pengerjaannya untuk meminimalisir kegagalan produk.
- ☑ Membutuhkan koordinasi tim yang baik agar tidak terjadi *miss communication* dalam pengembangan perangkat lunak.
- ☑ Jika ada masalah di satu unit, maka harus memperbaiki semua unit.

Contoh Incremental Model di Kehidupan Nyata

Contoh incremental model sangatlah beragam, salah satunya pengembangan aplikasi e-commerce. Increment yang digunakan untuk mengembangkan aplikasi tersebut cukup beragam, pada saat pertama dilakukan pengembangan akan dibuat fitur utama terlebih dahulu yaitu fitur pencarian produk.

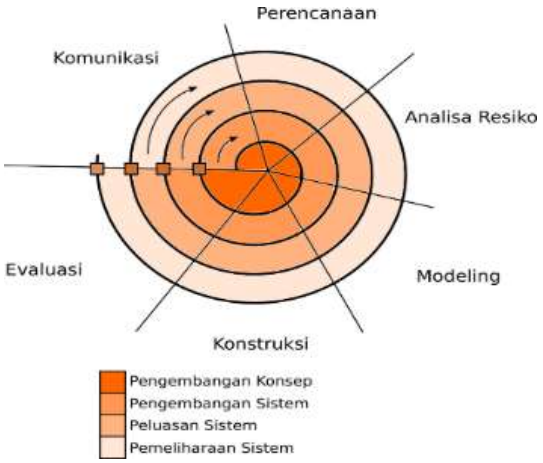
Setelah itu barulah melakukan increment berikutnya yaitu menambahkan fitur-fitur lain yang dibutuhkan pengguna, contohnya menambahkan fitur keranjang belanja, menambahkan integrasi dengan metode pembayaran online dan juga menambahkan fitur pelacakan. Setelah ditambahkan semuanya barulah dilakukan uji coba dan diluncurkan aplikasinya.

Referensi :

- https://www.sekawanmedia.co.id/blog/incremental-model/
- https://codingstudio.id/blog/apa-itu-incremental-model/
- https://sis.binus.ac.id/2019/07/02/software-development-model-incremental-model/
- https://www.revou.co/id/kosakata/incremental

8. *Spiral Model*

Model spiral adalah model pengembangan perangkat lunak yang fokus pada manajemen risiko, menggabungkan aspek iteratif dari model prototyping dengan pendekatan sistematis dari model waterfall.



Model ini memungkinkan proyek untuk berevolusi melalui serangkaian iterasi (disebut "putaran") dalam bentuk spiral, di mana setiap putaran melibatkan evaluasi risiko, pengembangan, dan perencanaan untuk siklus berikutnya.

Model Spiral ini menyediakan pendekatan sistematis dan berulang untuk pengembangan perangkat lunak. Jika kita melihat gambar model ini tampak seperti spiral dengan banyak putaran.

Setiap putaran spiral disebut fase proses pengembangan perangkat lunak. Model Spiral merupakan pendekatan pengembangan perangkat lunak yang berfokus pada pengelolaan risiko melalui serangkaian iterasi. Model ini terdiri dari beberapa fase yang meliputi:

1. Perencanaan

   Pada fase perencanaan, tim proyek menentukan ruang lingkup proyek dan membuat rencana untuk iterasi selanjutnya.

2. Analisis Risiko

   Fase analisis risiko adalah saat risiko terkait proyek diidentifikasi dan dievaluasi.

3. Rekayasa

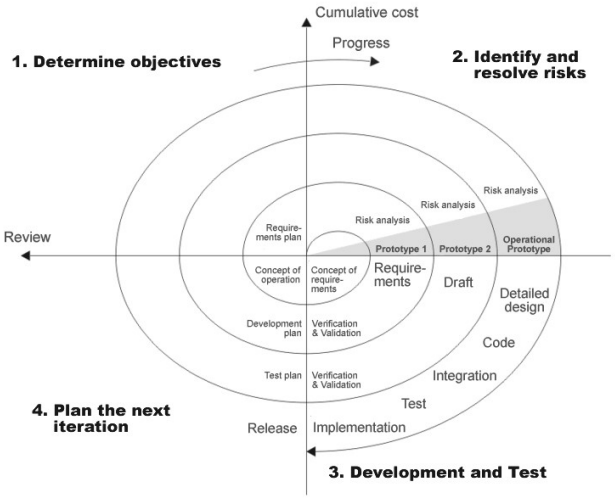
   Selama fase rekayasa, perangkat lunak dikembangkan berdasarkan persyaratan yang telah dikumpulkan sebelumnya.

4. Evaluasi

   Fase evaluasi melibatkan penilaian terhadap perangkat lunak untuk memastikan pemenuhan terhadap persyaratan pelanggan dan kualitas yang tinggi.

5. Perencanaan (kembali)

   Setelah evaluasi, iterasi spiral dimulai kembali dengan fase perencanaan baru berdasarkan hasil evaluasi sebelumnya.



Model Spiral sering digunakan dalam proyek pengembangan perangkat lunak yang kompleks dan besar karena memberikan fleksibilitas dan adaptabilitas yang tinggi. Model ini juga cocok untuk proyek dengan tingkat ketidakpastian atau risiko yang signifikan.Jari-jari spiral dalam model ini merepresentasikan biaya proyek sampai saat ini, sementara dimensi sudut menunjukkan kemajuan pada fase tertentu.

Setiap fase dibagi menjadi empat kuadran dengan fungsi-fungsi berikut:

1. Penentuan Tujuan dan Identifikasi Solusi Alternatif : Pada kuadran ini, persyaratan dari pelanggan diumpulkan dan tujuan serta solusi alternatif dianalisis.
2. Identifikasi dan Penyelesaian Risiko : Kuadran ini melibatkan evaluasi solusi alternatif dan penyelesaian risiko yang terkait dengan solusi yang dipilih. Prototipe juga dibangun pada akhir kuadran ini.

3. Pengembangan Produk versi berikutnya : Fase ini berfokus pada pengembangan fitur dan pengujian untuk versi perangkat lunak berikutnya.
4. Tinjauan dan Perencanaan untuk Fase selanjutnya : Pada kuadran terakhir, versi perangkat lunak dievaluasi oleh pelanggan dan perencanaan untuk iterasi selanjutnya dimulai.

Bagi yang bertanya berapa banyak jumlah putaran dalam model spiral ini tentu jawabannya adalah tidak diketahui dan dapat bervariasi karena tergantung pada besar kecilnya sebuah proyek, diantaranya:

- ☑ Jumlah pasti tahapan yang diperlukan untuk mengembangkan produk dapat bervariasi oleh manajer proyek tergantung pada risiko proyek.
- ☑ Ketika manajer proyek secara dinamis menentukan jumlah fase, ia memiliki peran penting dalam mengembangkan produk menggunakan model spiral.
- ☑ Hal ini didasarkan pada gagasan spiral, dengan setiap iterasi spiral mewakili siklus pengembangan perangkat lunak yang lengkap.

Dalam kehidupan praktis, risiko proyek dapat muncul setelah proses pengembangan dimulai, dan dalam kondisi ini, model prototyping tidak dapat lagi digunakan. Namun, dalam model spiral, setiap fase melibatkan penanggalan dan analisis fitur produk, serta identifikasi dan penyelesaian risiko melalui pembuatan prototipe.

Dengan demikian, model ini menawarkan fleksibilitas yang lebih besar dibandingkan dengan model SDLC *(Software Development Life Cycle)* lainnya. Dengan pendekatan ini, model spiral memberikan cara yang lebih adaptif dalam menghadapi perubahan dan risiko yang mungkin muncul dalam proses pengembangan perangkat lunak.

Model Spiral dijuluki sebagai Meta-Model karena kemampuannya mengintegrasikan berbagai model SDLC menjadi satu kerangka kerja yang komprehensif. Ini berarti model spiral tidak hanya berdiri sendiri sebagai metodologi pengembangan, tetapi juga berfungsi sebagai kerangka kerja yang dapat menyesuaikan dan menerapkan elemen-elemen dari model pengembangan lainnya.

Dengan kata lain, Kemampuan untuk memasukkan berbagai teknik dan metodologi pengembangan menjadikan model spiral sebuah pilihan yang fleksibel dan kuat, terutama untuk proyek-proyek besar dan kompleks dengan tingkat ketidakpastian yang tinggi.

Kelebihan dari Spiral Model :

Model Spiral menawarkan sejumlah kelebihan yang membuatnya menjadi pilihan yang populer dalam pengembangan perangkat lunak, diantaranya:

- ☑ Penanganan Risiko : model spiral sangat efektif dalam menangani risiko yang muncul seiring berjalannya proyek. Dengan melakukan analisis risiko dan penanganan risiko pada setiap tahap pengembangan, model ini memungkinkan tim proyek untuk mengidentifikasi, mengevaluasi, dan mengatasi risiko yang tidak terduga dengan lebih baik.
- ☑ Cocok untuk Proyek Besar : karena kemampuannya untuk mengatasi risiko dan fleksibilitasnya, model ini lebih sesuai untuk proyek-proyek dengan skala yang besar dan persyaratan yang kompleks.
- ☑ Fleksibilitas dalam Persyaratan : model spiral memungkinkan perubahan persyaratan untuk diintegrasikan dengan mudah pada setiap tahap pengembangan. Ini memungkinkan fleksibilitas dalam menanggapi perubahan kebutuhan atau permintaan pelanggan dengan akurat dan tepat waktu.

- ☑ Kepuasan Pelanggan : dengan memungkinkan pelanggan untuk melihat perkembangan produk pada tahap awal pengembangan, Model Spiral dapat meningkatkan kepuasan pelanggan. Pelanggan dapat berpartisipasi aktif dalam proses pengembangan, sehingga mereka lebih akrab dengan sistem dan dapat memberikan umpan balik yang berharga.
- ☑ Pendekatan Iteratif dan Inkremental : model spiral menawarkan pendekatan iteratif dan inkremental terhadap pengembangan perangkat lunak. Ini memungkinkan fleksibilitas dan adaptabilitas dalam menanggapi perubahan persyaratan atau situasi tak terduga selama proses pengembangan.
- ☑ Penekanan pada Manajemen Risiko : model spiral memberikan penekanan yang kuat pada manajemen risiko, membantu tim proyek untuk mengurangi dampak ketidakpastian dan risiko dalam pengembangan perangkat lunak.
- ☑ Peningkatan Komunikasi : dengan menyediakan evaluasi dan peninjauan rutin, model spiral dapat meningkatkan komunikasi antara pelanggan dan tim pengembangan. Hal ini membantu memastikan pemahaman yang baik tentang kebutuhan dan ekspektasi pelanggan serta memperbaiki kolaborasi antara tim.
- ☑ Peningkatan Kualitas : melalui beberapa iterasi dalam proses pengembangan perangkat lunak, model spiral dapat menghasilkan peningkatan kualitas dan keandalan produk akhir. Proses ini memungkinkan identifikasi dan perbaikan kesalahan secara bertahap, sehingga meningkatkan kualitas keseluruhan dari perangkat lunak yang dikembangkan.

Dengan berbagai keuntungan ini, model spiral menjadi salah satu pendekatan yang paling disukai dalam pengembangan perangkat lunak, terutama untuk proyek-proyek yang kompleks dan berisiko tinggi.

Kekurangan dari Spiral Model :

Beberapa kelemahan yang perlu dipertimbangkan sebelum menggunakan model ini, diantaranya:

- ☑ Kompleksitas : model spiral cenderung lebih kompleks dibandingkan dengan model SDLC lainnya. Hal ini disebabkan adanya iterasi dan penanganan risiko yang berulang, sehingga memerlukan pemahaman yang mendalam dan pengelolaan yang cermat dari tim proyek.
- ☑ Mahal : model ini biasanya tidak cocok untuk proyek-proyek kecil karena memerlukan investasi yang signifikan dalam perencanaan, analisis risiko, dan evaluasi. Hal ini membuatnya menjadi pilihan yang kurang ekonomis untuk skala proyek yang lebih kecil.
- ☑ Ketergantungan pada Analisis Risiko : keberhasilan penyelesaian proyek dalam model spiral sangat bergantung pada analisis risiko yang akurat dan efektif. Tanpa keahlian yang memadai dalam mengidentifikasi dan menangani risiko, pengembangan proyek dengan menggunakan model ini dapat menghadapi kesulitan atau bahkan gagal.
- ☑ Kesulitan dalam Manajemen Waktu : karena jumlah tahapan dalam model spiral tidak ditentukan pada awal proyek, perkiraan waktu menjadi sulit dilakukan. Hal ini dapat menyebabkan kesulitan dalam manajemen waktu dan penjadwalan, karena tiap iterasi mungkin membutuhkan waktu yang bervariasi.
- ☑ Memakan Waktu : model spiral dapat memakan waktu karena memerlukan banyak evaluasi dan peninjauan pada setiap tahap iterasi. Proses ini dapat menunda kemajuan proyek dan memperpanjang jadwal pengembangan.

☑ Intensif Sumber Daya : model spiral dapat menjadi intensif sumber daya, karena memerlukan upaya dan investasi yang signifikan dalam perencanaan, analisis risiko, dan evaluasi. Hal ini dapat meningkatkan biaya dan sumber daya yang diperlukan untuk proyek.

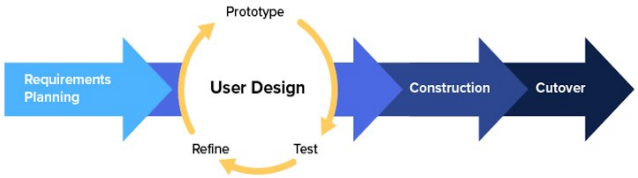
Meskipun model spiral menawarkan pendekatan yang fleksibel dan adaptif dalam pengembangan perangkat lunak, kekurangannya perlu diperhatikan agar dapat mengambil keputusan yang tepat terkait dengan penggunaannya.

Referensi :

- https://sis.binus.ac.id/2019/04/29/pengembangan-sistem-spiral-model/
- https://eko.co.id/1488/mengenal-model-spiral-dalam-pengembangan-perangkat-lunak-kelebihan-dan-kekurangannya/
- https://www.konsepkoding.com/2022/03/pengertian-metode-spiral-tahapan-kelebihan-kekurangan.html
- https://kantinit.com/programming/metode-spiral-adalah-pengertian-tahapan-dan-kelebihan/

9. *RAD (Rapid Application Development) Model*

Rapid Application Development (RAD) adalah metodologi pengembangan perangkat lunak (SDLC) yang menggunakan pengabungan antara prototype model dengan iterative model. Prototipe adalah model kerja yang secara fungsional setara dengan komponen produk.



Dalam *model RAD (Rapid Application Development),* modul fungsional dikembangkan secara paralel sebagai prototip dan terintegrasi untuk membuat produk yang lengkap untuk pengiriman produk yang lebih cepat, dikarenakan tidak ada rincian planning yang detail maka memudahkan untuk melakukan perubahan pada saat development berjalan.

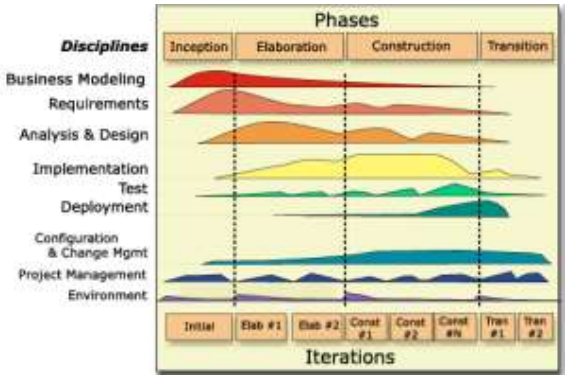
Kelebihan dari RAD

- Mudah mengakomodasi perubahan sistem
- Progress *development* bisa di ukur.
- Waktu iterasi bisa diperpendek menggunakan *RAD Tools.*
- Mengurangi waktu *development*.
- Mudah dalam menentukan dasar system.
- Mempermudah feedback customer.
- Cocok untuk proyek yang membutuhkan waktu pengembangan yang lebih pendek.
- Cocok untuk sistem yang berbasis komponen dan terukur.

Kelemahan dari RAD

- Ketergantungan pada anggota bisnis tim untuk mengidentifikasi persyaratan bisnis.
- Hanya sistem yang bisa di modularized yang bisa dibangun menggunakan RAD.
- Membutuhkan *developer / designer* yang berpengalaman.
- Ketergantungan pada keterampilan model.
- Kompleksitas manajemen.
- Tidak dapat diterapkan pada proyek yang kecil / murah.

10. *Rational Unified Process (RUP) Model*



Pengertian *Rational Unified Process (RUP)* menurut IBM adalah kerangka proses yang menyediakan simulasi sistem pada industri untuk sistem, *software,* implementasi, dan manajemen proyek yang efektif.

RUP adalah salah satu dari sekian banyak proses yang terdapat di dalam *Rational Process Library,* yang memberikan simulasi terbaik untuk pengembangan atau kebutuhan proyek.

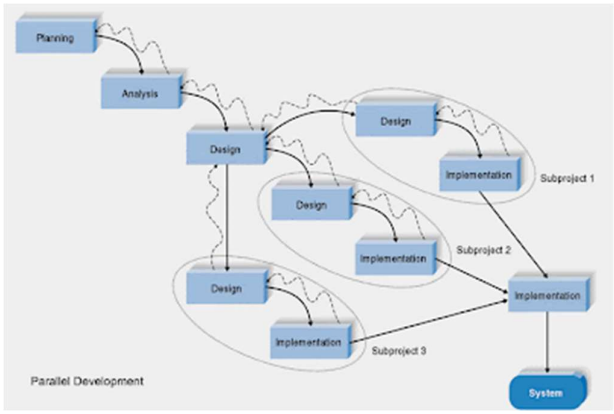
Kelebihan dari RUP Model :

- Menyediakan akses yang mudah terhadap pengetahuan dasar bagi anggota tim.
- Menyediakan petunjuk bagaimana menggunakan UML secara efektif.
- Mendukung proses pengulangan dalam pengembangan software.
- Memungkinkan adanya penambahan-penambahan pada proses.
- Memungkinkan untuk secara sistematis mengontrol perubahan-perubahan yang terjadi pada software selama proses pengembangannya.
- Memungkinkan untuk menjalankan test case dengan menggunakan *Rational Test Manager Tool.*

Kekurangan dari RUP Model :

- Metodologi ini hanya dapat digunakan pada pengembangan perangkat lunak yang berorientasi objek dengan berfokus pada UML *(Unified Modeling Language)*

11. *Parallel Development Model*



Parallel development model adalah sebuah pendekatan pengembangan dimana beberapa versi objek atau fitur perangkat lunak dikerjakan secara bersamaan, seringkali oleh beberapa tim atau dengan membuat jalur pengembangan terpisah, untuk mempercepat proses pengembangan keseluruhan atau untuk mempersiapkan rilis, pemeliharaan, dan adaptasi varian produk.

Metode ini mencoba mengatasi masalah lamanya waktu antara fase analisis dengan implementasi sistem. Desain secara umum akan ditampilkan terlebih dahulu kemudian proyek dibagi-bagi menjadi beberapa sub proyek yang dikerjakan secara bersama-sama secara parallel. Jadi, parallel development model memungkinkan beberapa fase dilakukan secara bersama-sama untuk mempersingkat waktu.

Karakteristik Utama Model Pengembangan Paralel

- Bekerja pada beberapa fitur atau objek secara bersamaan:
  Tim pengembangan dapat mengerjakan berbagai fitur atau varian produk secara simultan.

- Penggunaan cabang terpisah:
  Jalur pengembangan terpisah sering kali dibuat untuk memungkinkan berbagai modifikasi dilakukan tanpa saling mengganggu.
- Peningkatan kecepatan dan efisiensi:
  Dengan memecah pekerjaan, proyek dapat diselesaikan lebih cepat karena banyak bagian dapat dikerjakan dalam periode waktu yang sama.
- Mengelola beberapa konfigurasi:
  Pengembangan paralel menciptakan beberapa versi "terbaru" atau varian produk secara bersamaan.

Kapan Model Pengembangan Paralel Digunakan ?

Pengembangan paralel biasanya dibutuhkan untuk:

- ✓ Persiapan rilis : Memungkinkan pekerjaan dilakukan pada rilis mendatang sementara rilis sebelumnya masih dalam pemeliharaan.
- ✓ Pemeliharaan pasca-rilis : Memisahkan perbaikan *bug* dan pemeliharaan dari pengembangan fitur baru.
- ✓ Perangkat lunak yang disesuaikan : Memungkinkan pembuatan varian khusus untuk pelanggan yang berbeda.
- ✓ Diferensiasi pekerjaan : Membagi tugas di antara tim yang berbeda atau berdasarkan fitur yang berbeda.

Manfaat Model Pengembangan Paralel

- ☑ Pengurangan waktu siklus proyek : Waktu yang dibutuhkan untuk menyelesaikan proyek secara keseluruhan dapat berkurang secara signifikan.
- ☑ Peningkatan fleksibilitas : Memungkinkan penanganan kebutuhan yang berbeda, seperti pemeliharaan dan pengembangan fitur baru secara bersamaan.

Tantangan Model Pengembangan Paralel

- Koordinasi yang kompleks :
  Mengelola banyak jalur pengembangan bisa jadi rumit, terutama untuk proyek besar.
- Potensi konflik :
  Adanya banyak konfigurasi "terbaru" dapat menyebabkan konflik saat mencoba menggabungkan perubahan.

Kelebihan dari Parallel Development Model :

- ☑ Jadwal waktu yang dibutuhkan untuk mengerjakan proyek menjadi lebih cepat, hanya ada sedikit kesempatan jika terjadi perubahan rencana pada lingkungan bisnis yang cepat berubah dapat menyebabkan pengerjaan kembali.

Kekurangan dari Parallel Development Model :

- ☑ Model ini akan mengalami kendala pada lamanya pekerjaan jika ada subproyek yang bermasalah yang berimbas pada sub proyek yang lain.
- ☑ Pada akhir proyek, integrasi hasil kerja dari sub proyek juga dapat menimbulkan masalah.
- ☑ Tidak cocok untuk proyek-proyek perangkat lunak yang kompleks..

12. *Big Bang Model*

Pada Big Bang Model ini, kita tidak mengikuti proses tertentu. Perkembangan hanya dimulai dengan uang dan usaha yang dibutuhkan sebagai masukan, dan hasilnya adalah perangkat lunak yang dikembangkan yang mungkin atau mungkin tidak sesuai dengan kebutuhan pelanggan.

Model Big Bang ini tidak mengikuti dan hanya ada sedikit perencanaan yang diperlukan. Bahkan pelanggan pun tidak yakin dengan apa yang sebenarnya dia inginkan dan persyaratannya diimplementasikan dengan cepat tanpa banyak analisis.

Biasanya model ini diimplementasi untuk proyek kecil dimana tim developernya sangat sedikit.

Kelebihan dari Big Bang Model :

- Model yang sangat sederhana.
- Sedikit atau tidak ada perencanaan yang dibutuhkan.
- Mudah dikelola.
- Sangat sedikit sumber daya yang dibutuhkan.
- Memberikan fleksibilitas kepada pengembang.
- Bagus untuk developer yang ingin belajar atau developer pendatang baru.

Kekurangan dari Big Bang Model :

- Memiliki resiko tinggi dan kepastian dari requirement yang tidak jelas.
- Tidak cocok untuk project skala besar dan berorientasi objek.
- Model yang buruk untuk proyek yang panjang dan sedang berlangsung.
- Bisa berubah menjadi sangat mahal jika persyaratan disalah pahami.