IF3210

Proses Pengembangan Perangkat Lunak untuk Aplikasi Mobile

Oleh:
Tim Pengajar IF3210

Semester II 2024/2025

Proses Pengembangan Perangkat Lunak untuk Aplikasi Mobile

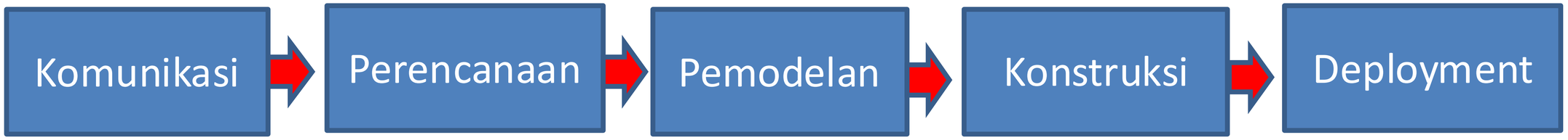
- Proses pengembangan perangkat lunak: serangkaian langkah terstruktur yang dilakukan untuk merancang, mengembangkan, menguji, dan memelihara perangkat lunak.
- Tujuan: memastikan bahwa perangkat lunak yang dihasilkan memenuhi kebutuhan pengguna dan berfungsi dengan baik.
- Proses ini dikenal sebagai **Siklus Hidup Pengembangan Perangkat Lunak** atau **Software Development Life Cycle (SDLC)**.

Manfaat SDLC

- Memastikan kualitas dan konsistensi
- Efisiensi waktu dan biaya
- Kemudahan pemeliharaan dan pembaruan
- Kolaborasi tim yang lebih baik
- Adaptabilitas terhadap perubahan

Aktivitas Pengembangan Perangkat Lunak menggunakan **Proses Perangkat Lunak**

Alur Proses Linear
(Linear Process Flow)



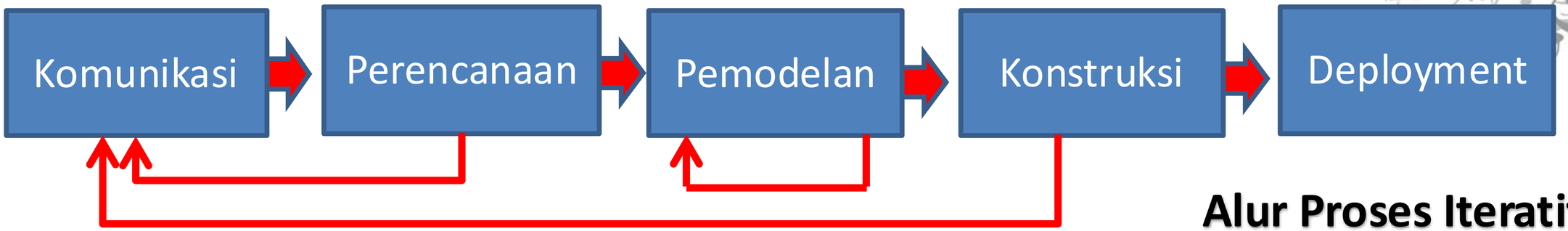
Inisiasi Proyek
Pengumpulan Kebutuhan

Estimasi
Penjadwalan
Tracking

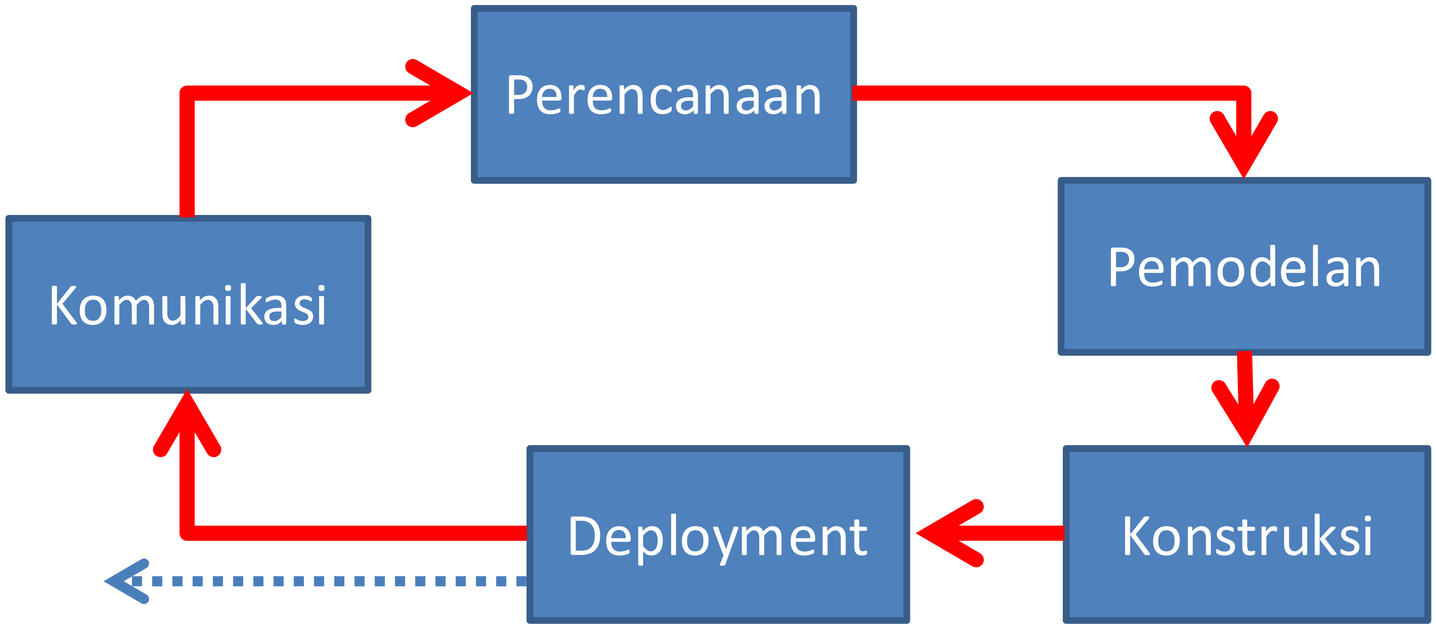
Analisis
Desain

Pemrograman
Pengujian

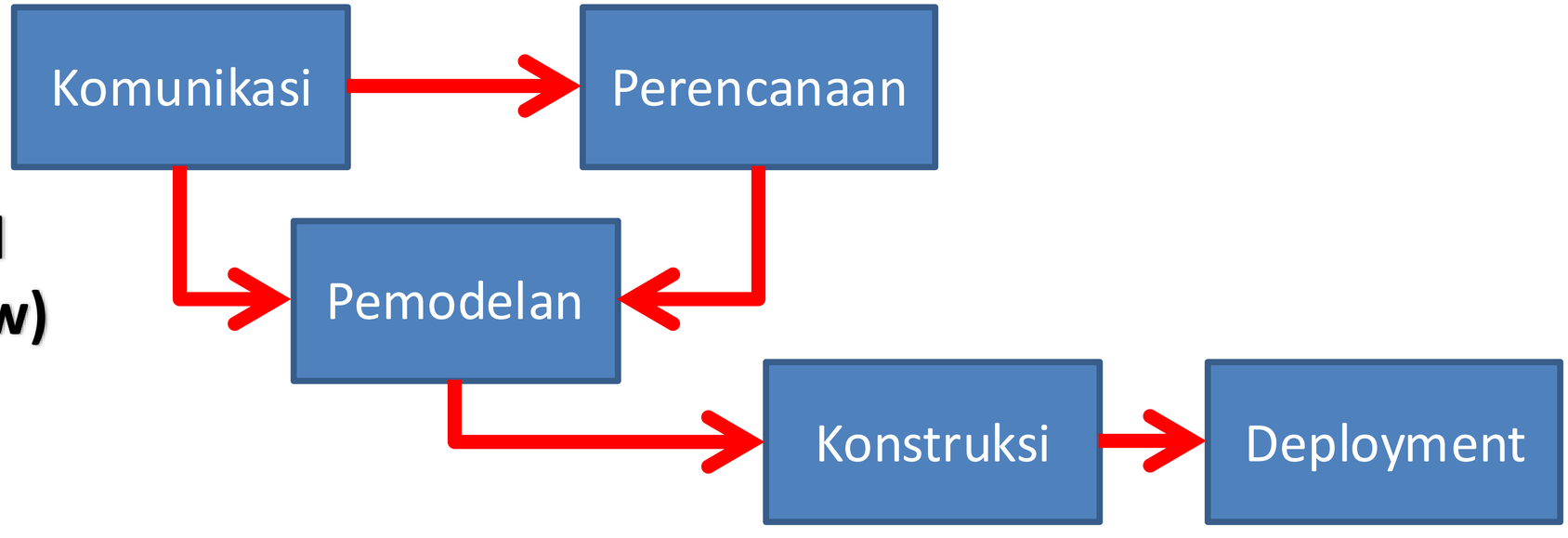
Delivery
Dukungan
Feedback



Alur Proses Iteratif (Iterative Process Flow)

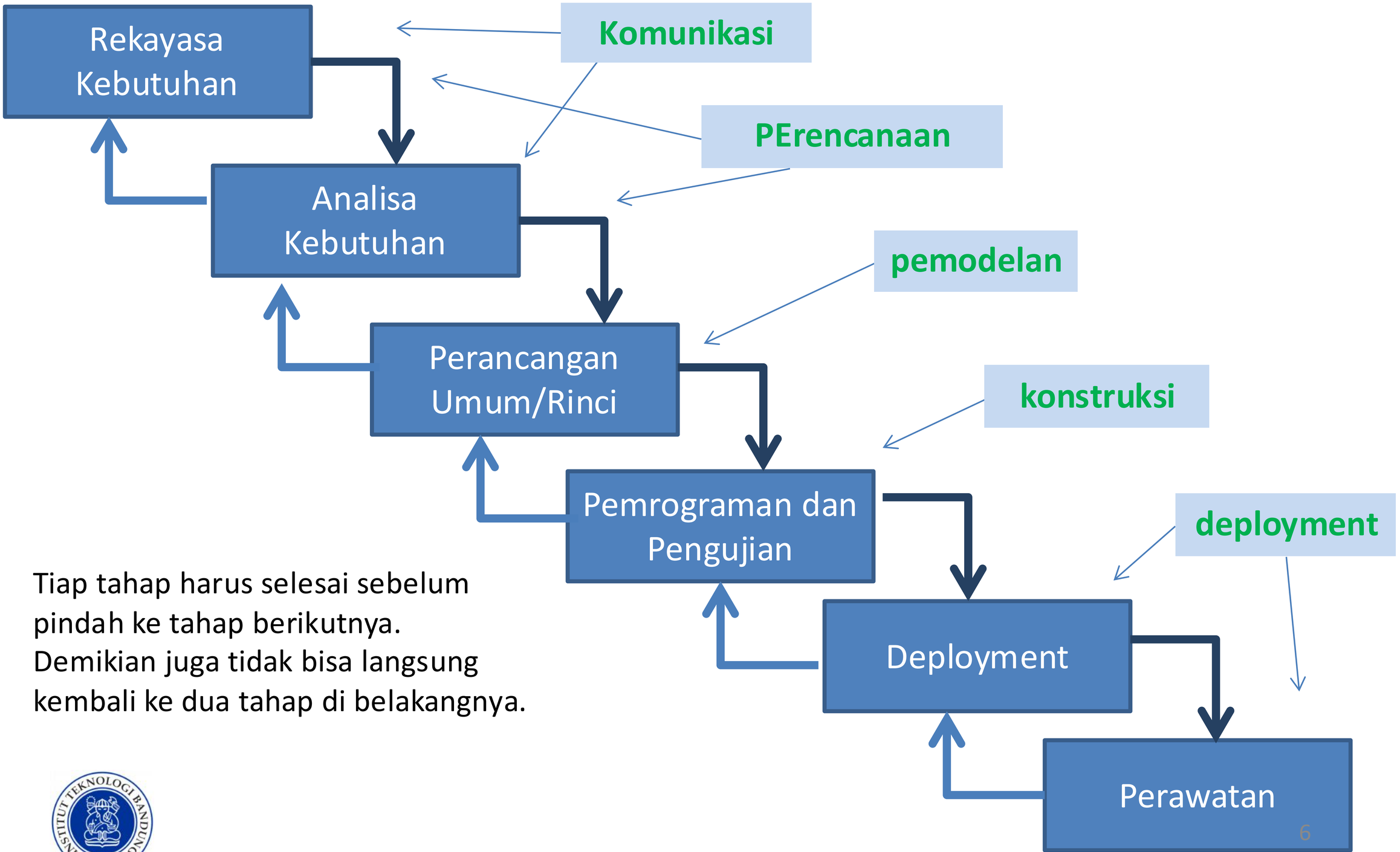


Alur Proses Berevolusi (Evolutionary process flow)



Alur Proses Paralel (Parallel process flow)

Waterfall Model



Tiap tahap harus selesai sebelum pindah ke tahap berikutnya.
Demikian juga tidak bisa langsung kembali ke dua tahap di belakangnya.

Problems

- Kebutuhan PL hampir selalu berubah. Dengan Waterfall perubahan ini tidak tertangani dengan baik
  - Dengan waterfall kita harus melewati setiap tahap selengkap mungkin sebelum memasuki tahap aktifitas berikutnya
    - Akibatnya kadang sudah di program, tetapi kebutuhan awalnya sudah mengalami perubahan
- Jarang proyek perangkat lunak mengikuti jalur sekuensial
  - Iterasi tetap terjadi tetapi tidak secara 'langsung'
  - Model ini tidak cukup bisa mengakomodasi tingkat ketidakpastian atas kebutuhan perangkat lunak
- Calon pengguna harus bersabar menunggu versi jadi dari program yang diminta
  - Karena harus menunggu hingga akhir proyek
- Kadang tidak terlalu cocok untuk sistem yang kompleks
  - Sistem yang kompleks biasanya melibatkan pihak yang berbeda-beda
- Beberapa anggota tim pengembang harus menunggu pekerjaan rekannya dalam satu tim menyelesaikan aktivitasnya dulu (*blocking state*)
  - Kadang menghabiskan waktu produktifnya, sehingga pemimpin proyek harus mengatur anggota timnya.
- Blocking state ini biasanya terjadi di awal dan akhir proyek.

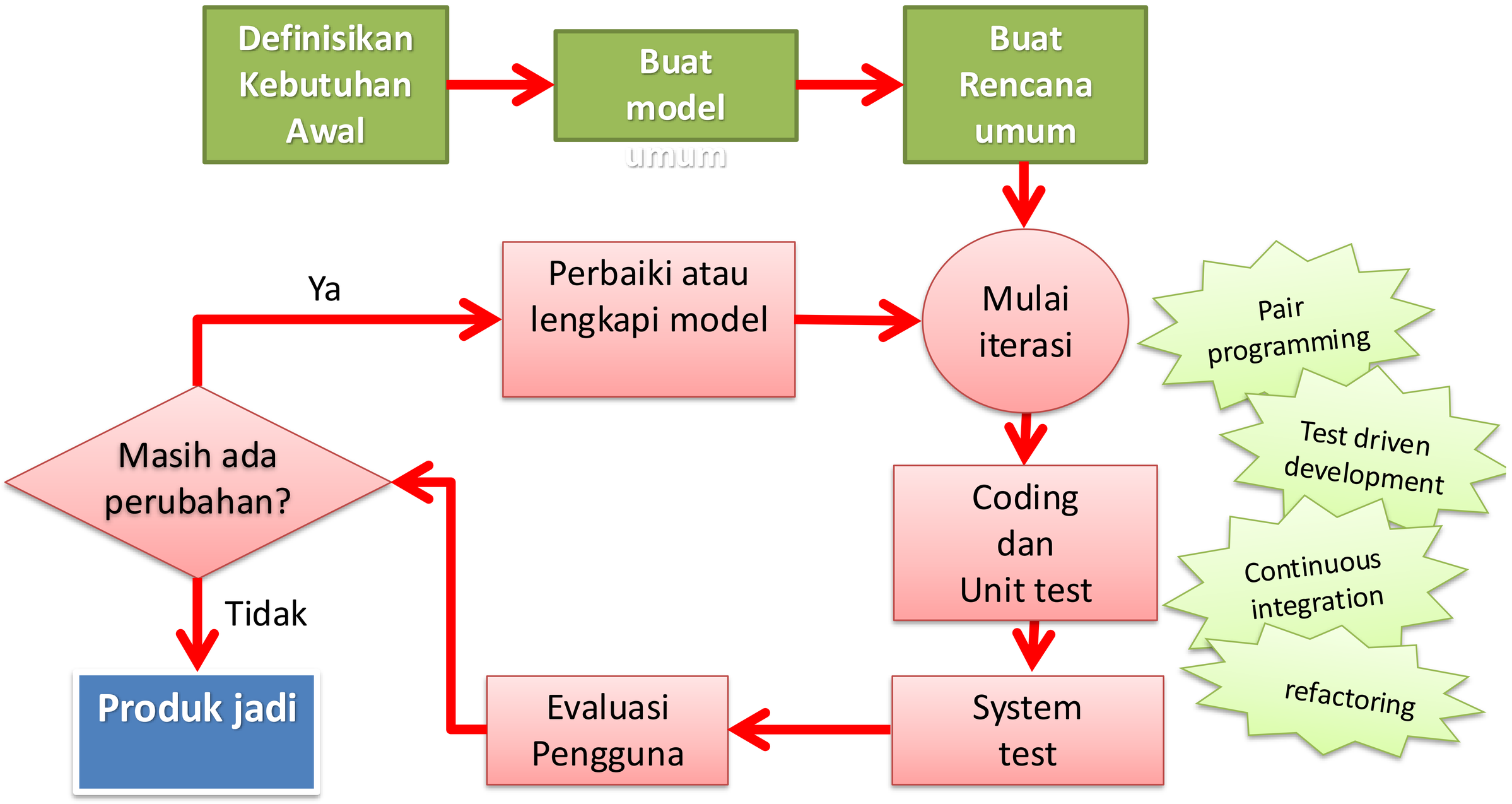
Proses dalam Aplikasi Mobile

Tantangan:
1. Variasi Platform
2. Keterbatasan Perangkat
3. Ekspektasi pengguna

1. **Strategi Pengembangan**
2. **Analisis dan Perencanaan**
3. **Desain UI/UX**: Arsitektur informasi, Alur kerja, *Wireframes*, Panduan *Style*, *Mockup*, *Prototype*
4. **Pengembangan Aplikasi**: *Backend*/*Server*, API, Tampilan Depan Aplikasi
5. **Pengujian**: Pengalaman pengguna, Fungsional, kinerja, keamanan, perangkat dan platform
6. **Penerapan & Dukungan**

Metodologi Agile

Metodologi Agile



Agile Manifesto

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

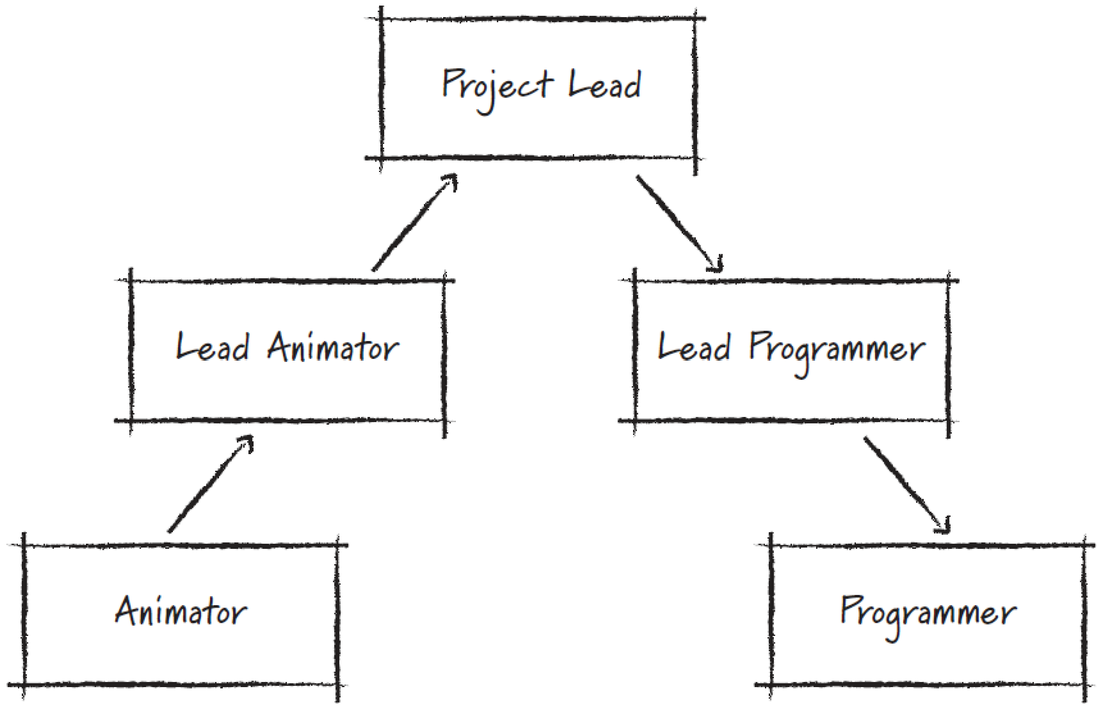
Responding to change over following a plan

Agile Development Model

Traditional Development Model

Individuals and Interactions

over Processes and Tools



- Example in Game development requires different disciplines developers
  - Animators
  - Designers
  - Texture artist
  - Programmers
  - Audio composer

- Incorporating hierarchy in Game Development Management is a waste
- Small problems can be attack between colleagues

Beberapa Variasi Metodologi Agile

- Agile Modeling
- Agile Unified Process (AUP)
- Dynamic Systems Development Method (DSDM)
- Essential Unified Process (EssUP)
- Extreme Programming (XP)
- Feature Driven Development (FDD)
- Open Unified Process (OpenUP)
- Scrum
- Velocity tracking

Apa Beda Agile dengan lainnya?

- Kebutuhan dari pengguna dikumpulkan secara iteratif dan evolusioner
- Suatu fungsionalitas ditentukan skala prioritasnya
- Suatu fungsionalitas dilengkapi dari kebutuhannya, hingga coding dan pengujian unitnya secara lengkap sebelum memulai fungsionalitas lainnya.
- Tidak terlalu banyak dokumentasi
  - Karena jumlah anggota tim kecil, maka komunikasi informal terjadi secara ekstensif yang memungkinkan pengembangan lebih baik
- Sulit menjamin kualitas untuk proyek skala besar, karena keterbatasan metodologi ini

SDLC and Agile fo Mobile Apps

SDLC for Mobile Apps

1. Planning & Requirement Analysis
2. UI/UX Design
3. Development (Iterative & Parallel)
4. Testing & Quality Assurance
5. Deployment & App Store Submission
6. Maintenance & Update

Planning & Requirement Analysis

- Identify user needs, business goals, and technical feasibility.
- Research target audience and competition.
- Define key app features, UI/UX expectations, and monetization models

UI/UX Design

- Wireframing & prototyping (using tools like Figma and Adobe XD).
- User journey mapping for a smooth experience.
- Focus on mobile constraints: screen sizes and touch interactions.

Development

- Frontend (UI, animations, interactions) and backend (APIs, databases) development in parallel.
- Use frameworks like Flutter, React Native, or native development (Swift/Kotlin).
- Implement modular code for easy updates.

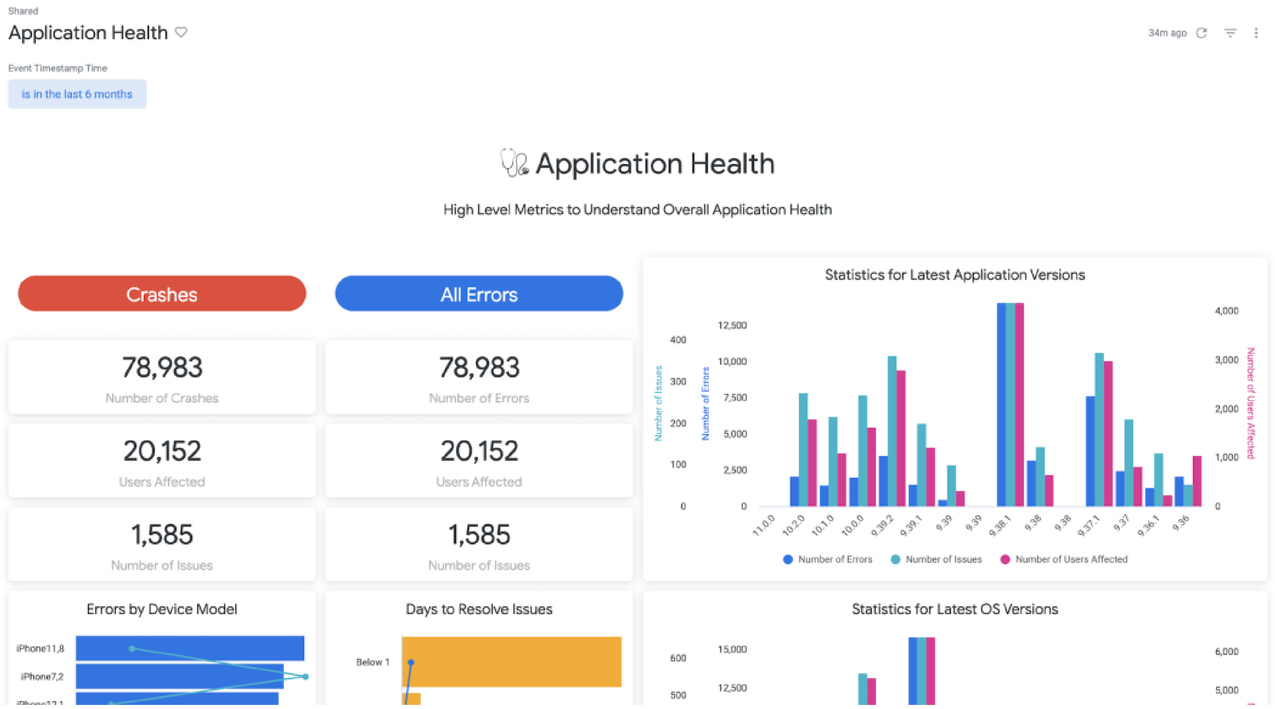
Testing & QA

- Device and platform testing for various screen sizes and OS versions.
- Automated and manual testing (unit tests, UI tests, security testing).
- Beta testing using TestFlight (iOS) or Firebase App Distribution (Android).

Deployment & App Store Submission

- Ensure compliance with Google Play Store & Apple App Store policies.
- Optimize metadata (ASO - App Store Optimization).
- Prepare for approval process delays and re-submissions.

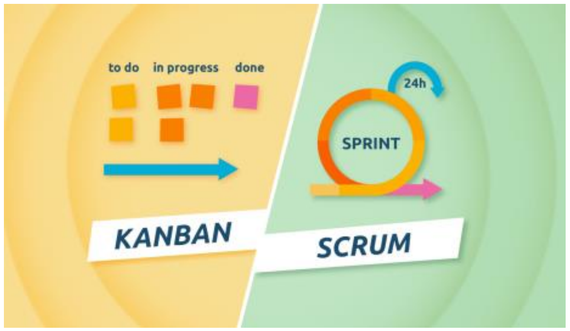
Maintenance & Update



- Collect user feedback and crash analytics (Firebase, Sentry).
- Regular updates to fix bugs and add features.
- Ensure compatibility with new OS versions.

Adapted Agile Methodology for Mobile Apps

1. Scrum or Kanban Framework.
2. Sprint Planning & Prioritization.
3. Continuous Integration and Deployment
4. User Feedback & Rapid Iteration
5. Cross-functional Collaboration



Adapted Agile Methodology for Mobile Apps

1. Scrum or Kanban Framework: *Scrum: Sprint-based development (typically 2-week cycles). Kanban: Continuous development with a flow-based approach.*
2. Sprint Planning & Prioritization: *Define features using User Stories & Epics. Prioritize features based on business value and development complexity. Ensure Minimal Viable Product (MVP) is defined early.*

Adapted Agile Methodology for Mobile Apps

3. Continuous Integration and Deployment: *Automate builds and tests using GitHub Actions, Jenkins, or Bitrise. Deploy frequent updates to internal testers via Firebase App Distribution. Feature toggling for phased rollouts.*
4. User Feedback & Rapid Iteration: *Collect analytics via Mixpanel, Firebase (Crashlytics), or Google Analytics. Conduct A/B testing for UI/UX improvements. Adapt backlog based on user needs.*

Adapted Agile Methodology for Mobile Apps

5. Cross-functional Collaboration: *Product managers, developers, designers, and QA collaborate. Daily stand-ups, sprint reviews, and retrospectives ensure agility. Tasks can be managed using tools like JIRA, Trello, or ClickUp.*

1. Apakah Kotlin kompatible dan juga memakai Java framework layer

2. Apakah kita bisa supply driver HAL sendiri di Android?

3. Apakah ada contoh contoh aplikasi or studi kasus yg memanfaatkan layer layer arsitektur di atas?

References:

1. Roger Pressman, “Software Engineering: A Practitioner’s Approach 7th edition”, McGraw Hill, 2010.
2. Ian Sommerville, “Software Engineering”, 9th Edition, Addison Wesley, 2010.
3. John Hunt, “Agile Software Construction”, Springer-Verlag, 2006.
4. https://aws.amazon.com/id/what-is/sdlc/
5. https://www.youtube.com/watch?v=PxJFFYppMWM&ab_channel=CNNIndonesia
6. https://firebase.google.com/docs/crashlytics